Universidad de Valladolid

Escuela de Ingeniería Informática

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN COMPUTACIÓN

Aplicación de Business Intelligence para las retransmisiones de la Fórmula 1

Autor:
Alberto González de la Plaza
Tutores:
Félix Villafáñez Cardeñoso
Pablo Sánchez Mayoral

Agradecimientos

A mi familia, en especial a mis padres por apoyarme y ayudarme en todo lo que pueden.

A mis amigos, por los ánimos recibidos y la motivación para crear cosas nuevas.

A mi tutor Félix Villafáñez Cardeñoso, por compartir conmigo sus conocimientos de Business Intelligence y por involucrarse en ser mi tutor pese a no ser alumno de su Facultad, gracias por el tiempo dedicado.

A mi tutor Pablo Sánchez Mayoral, por dar el consentimiento para que pueda realizar este TFG de lo que me apasiona, la visualización de datos y por darme la libertad que deseaba para realizarlo, gracias por sus consejos.

A toda la familia del CD Parquesol por generar ilusión y alegría en mí y en todo el barrio.

Gracias a todos

Resumen

El propósito principal de este proyecto es crear una aplicación de visualización de datos interactiva que permita a una cadena de televisión amenizar una retransmisión de la Fórmula 1 en directo. Esta aplicación muestra a los comentaristas de la carrera, datos relevantes para que se puedan diferenciar de la competencia. Los datos son extraídos desde los orígenes de manera automatizada mediante una API. Se realizan las tres etapas de la ETL (Extracción, Transformación y Carga), para terminar en un almacén de datos que sea de fácil acceso para la herramienta de visualización.

Para crear esta aplicación llamada *Fórmula 1 BI* se ha desarrollado una aplicación de Business Intelligence mediante la herramienta Power BI ya que permite crear aplicaciones de visualización con cuadros de mando.

Adicionalmente, se incluyen procedimientos de análisis de datos para enriquecer la información mostrada. Para ello se realiza una métrica para asignar una puntuación a cada piloto en una temporada reduciendo el efecto del monoplaza que conduce. Esta métrica tiene en cuenta factores como ser buen piloto en lluvia, tener pocos accidentes, obtener buenos resultados respecto a su compañero de equipo...

Esta aplicación de visualización de datos se enfoca como un proyecto de Business Intelligence, siguiendo todos los pasos para completarlo correctamente. Desde la definición conceptual del proyecto e identificación de requerimientos hasta la implementación de la aplicación para la ayuda en la toma de decisiones utilizando la herramienta Power BI.

Palabras clave: Inteligencia de negocio, Fórmula 1, cuadro de mandos, Power BI, visualización de datos, ETL, análisis de datos.

Abstract

The main purpose of this project is to create an interactive data visualization application that allows a television channel to entertain a live Formula 1 broadcast. This application shows the race commentators relevant data so that they can differentiate themselves from the others television channels. The data is extracted from the sources in an automated way via API. The three stages of ETL (Extraction, Transformation and Load) are performed, ending up in a data warehouse that is easily accessible to the visualisation tool.

To create this application called *Formula 1 BI*, a Business Intelligence application has been developed using the Power BI, because it allows the creation of visualization applications with dashboards.

Furthermore, it includes data analysis procedures to enrich the information displayed. For this purpose, a metric is also created to assign a score to each driver in a season by reducing the effect of the single-seater he drives. This metric takes into account factors such as being a good driver in the rain, having few accidents, achieving good results compared to his teammate...

This data visualization application is approached as a Business Intelligence project, following all the steps to complete it correctly. From the conceptual definition of the project and identification of requirements to the implementation of the application for decision support using the Power BI.

Key words: Business Intelligence, BI, Formula 1, dashboard, Power BI, data visualization, ETL, data analysis.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Este capítulo presenta el contexto que enmarca el proyecto, la motivación del mismo, los objetivos a alcanzar y la estructura del resto del documento.

1.1. Contexto

La Fórmula 1 es la principal competición de automovilismo internacional, tuvo su inicio en 1950 y desde entonces ha experimentado una increíble evolución en cuanto a sus retransmisiones. Este deporte ha logrado atraer a una audiencia global masiva a lo largo de los años. Según estadísticas oficiales de la FIA (Federación Internacional del Automóvil), cada carrera en 2021 fue seguida por una media de más de 70 millones de personas en todo el mundo con un aumento del 13 % respecto al año anterior, convirtiéndola en una de las competiciones deportivas más populares y seguidas a nivel mundial. [1]

Las retransmisiones de la Fórmula 1 han desempeñado un papel fundamental en su crecimiento y en su capacidad para llegar a un público muy amplio. Inicialmente, las retransmisiones se limitaban a la radio, donde los comentaristas transmitían en vivo la emoción de las carreras. Sin embargo, con la llegada de la televisión en los años 50 y 60, las carreras de Fórmula 1 comenzaron a ser transmitidas en vivo, permitiendo a los espectadores ver las emocionantes batallas en la pista.

Con el avance de la tecnología, las retransmisiones de la Fórmula 1 han seguido evolucionando. Actualmente se han introducido cámaras a bordo de los vehículos y sistemas de telemetría que generan multitud de información, brindando a los espectadores una experiencia aún más inmersiva y cercana a la acción en la pista. La telemetría permite la recogida de múltiples datos mediante sensores en los coches lo que genera mucha información que puede ser tratada y analizada.

En las últimas décadas, el auge de Internet y las nuevas plataformas de transmisión en línea ha revolucionado aún más la forma en que se transmiten las carreras de Fórmula 1. Plataformas como YouTube y Twitch han ganado popularidad, brindando a los aficionados la posibilidad de ver retransmisiones en vivo y acceder a contenido relacionado, como entrevistas, resúmenes y análisis. Esto ha ampliado aún más el alcance de la Fórmula 1, llegando a audiencias más jóvenes y conectadas digitalmente.

En cuanto a la dimensión económica, la Fórmula 1 es un deporte que mueve grandes sumas de dinero. La revista Forbes señala que la Fórmula 1, en el año 2022, generó ingresos anuales superiores a los 2 mil millones de dólares.[2]

Estos ingresos provienen de diversas fuentes, como los derechos de televisión, patrocinios, acuerdos comerciales y ventas de entradas. Las retransmisiones televisivas y las nuevas plataformas en línea han desempeñado un papel crucial en la generación de estos ingresos, ya que han permitido alcanzar audiencias más amplias y atractivas para los anunciantes.

1.2. Motivación

El motivo de este proyecto es crear una aplicación de visualización de datos profesional, que proporcione a los narradores de las retransmisiones de Fórmula 1, una herramienta para amenizar sus comentarios con

estadísticas precisas y relevantes. Estas estadísticas se obtienen a partir de la recopilación y análisis de datos, es decir, se genera información a partir de los datos.

La Fórmula 1 es un deporte donde la velocidad, la estrategia y la habilidad de los pilotos juegan un papel crucial. Sin embargo, para que los espectadores realmente puedan apreciar la intensidad de cada carrera, es fundamental contar con información adicional que enriquezca la experiencia.

Es ahí donde la aplicación de visualización de datos entra en juego. Gracias a ella, los narradores podrán acceder de manera ágil y precisa a estadísticas detalladas sobre los pilotos, las escuderías, los circuitos y los resultados históricos, entre otros aspectos relevantes.

Brindar a los narradores una aplicación que les permita ofrecer a los espectadores información valiosa y entretenida, basada en datos actualizados es fundamental para enriquecer la experiencia de los aficionados y generar un mayor impacto en la audiencia.

Por tanto, este proyecto también es una oportunidad para profundizar en mi pasión por la visualización datos y su aplicación en el mundo real. Se han recopilado, limpiado y analizado grandes volúmenes de datos relacionados con la Fórmula 1. Esta información se ha utilizado para crear visualizaciones interactivas que puedan ser utilizadas de manera efectiva y sencilla por los narradores.

1.3. Objetivos

El objetivo principal de este trabajo es utilizar los datos de la Fórmula 1 disponibles así como otras fuentes de datos abiertas para crear una aplicación de Business Intelligence útil para narradores de Fórmula 1 y poder ayudarles en sus retransmisiones. Otro objetivo es crear una medida que evalúe a los pilotos en un año concreto, reduciendo el efecto del monoplaza que conducen.

Surgen los siguientes subobjetivos que podemos evaluar durante la realización del trabajo:

- Centralizar los datos y tratar los mismos para presentarlos correctamente al usuario.
- Desarrollar una aplicación de Business Intelligence utilizando Power BI que permita visualizar información de pilotos, escuderías y circuitos de la Fórmula 1.
- Desarrollar diferentes gráficos descriptivos de los datos que sean de rápida interpretación.
- Diseñar una interfaz moderna, atractiva y usable.
- Crear una métrica a partir del análisis de datos que permita evaluar el rendimiento de un piloto en un año concreto reduciendo el efecto del monoplaza que conduce, el valor de esta métrica debe ser explicable mediante otras métricas.

1.4. Estructura de la memoria

En la presente memoria tiene el siguiente esquema:

- **Capítulo 1. Introducción**. Presenta el contexto que enmarca el proyecto, la motivación del mismo, los objetivos a alcanzar y la estructura del resto del documento.
- **Capítulo 2. Marco teórico**. Se explica la historia de la Fórmula 1, de las retransmisiones de Fórmula 1 y por último una explicación teórica del Business Intelligence.
- **Capítulo 3. Planificación**. Se abordan los temas relacionados con la gestión y planificación del proyecto.
- **Capítulo 4. Análisis**. Se realiza el análisis correspondiente a un proyecto de Business Intelligence empezando con la toma de requisitos resultado del análisis de las necesidades del cliente y cómo voy a dar respuesta a los mismos (indicadores, métricas, KPI's).

- **Capítulo 5 Diseño** Se detalla el diseño de la solución y se presentan los bocetos iniciales de la futura aplicación de Business Intelligence.
- **Capítulo 6. ETL**. Se realiza la implementación del proceso ETL. Desde la extracción de los datos desde los orígenes, siguiendo con la transformación en la que se aplica una serie de reglas para garantizar la calidad de los datos y por último la carga de los datos en la que los datos ya transformados pasan al entorno de explotación para poder realizar análisis posteriores.
- **Capítulo 7: Métrica *scoreAI***. Se explica detalladamente el proceso de creación de la métrica *scoreAI* que evalúa pilotos en un año mitigando el efecto del monoplaza que conduce.
- **Capítulo 8. Aplicación *Fórmula 1 BI***. Se describe detalladamente los cuadros de mando realizados junto con una explicación de cómo utilizar la aplicación. También se detalla el manual de ayuda al usuario integrado dentro de la propia aplicación.
- **Capítulo 9. Pruebas**. Se explican las pruebas realizadas tanto los test de la ETL como los tests de usabilidad sobre la aplicación de visualización simulando situaciones reales de uso de la aplicación.
- **Capítulo 10. Conclusiones**. Se presentan las conclusiones obtenidas del trabajo y se enumeran algunas de las posibles líneas de trabajo futuro a partir de este proyecto.

Capítulo 2

Marco Teórico

2.1. La Fórmula 1 (F1)

La Fórmula 1, considerada la máxima categoría del automovilismo, posee una enorme importancia y una popularidad en todo el mundo. Desde su creación, ha capturado la atención de millones de aficionados y se ha convertido en un fenómeno global que trasciende las barreras culturales y geográficas.

La Fórmula 1 es una competición de automovilismo que se caracteriza por ser una mezcla perfecta de velocidad, habilidad y tecnología. Los monoplazas de última generación, diseñados y construidos con todo tipo de detalles son pilotados por un único piloto que compiten en circuitos de todo el mundo.

Además de su impacto en el deporte en sí, la Fórmula 1 ha tenido un alcance mucho más amplio. Ha influido en la industria automotriz, impulsando el desarrollo de tecnologías avanzadas que luego se aplican en los vehículos de uso diario como son el uso de fibra de carbono o sistemas de propulsión híbridos. También ha sido una gran plataforma publicitaria, atrayendo patrocinadores y marcas de renombre a nivel mundial como las grandes marcas de automóviles como Mercedes o Renault pero también marcas totalmente alejadas de los automóviles como la empresa de bebidas energéticas Red Bull.

La Fórmula 1, como una competición que combina la pasión por el automovilismo con la emoción de la competencia deportiva, ha ganado su lugar como uno de los eventos más destacados en el calendario deportivo global. Su importancia y popularidad son evidentes pors la enorme base de seguidores, tanto en los circuitos como frente a las pantallas de televisión, y en la gran cantidad de recursos que se invierten en ella.

La Fórmula 1 es uno de los deportes más populares del mundo, es la categoría más importante de las carreras internacionales de monoplazas y está organizada por la Federación Internacional del Automóvil, a partir de ahora mencionada con sus siglas FIA. La Fórmula 1 se inauguró el 13 de mayo de 1950 como "Campeonato Mundial de Pilotos" en Silverstone (Reino Unido). [3]

A lo largo de una temporada se celebran en todo el mundo varias carreras denominadas "Grandes Premios", el conjunto de estas carreras se denomina temporada de Fórmula 1. La palabra "Fórmula" se refiere a un conjunto de normas que deben cumplir todos los equipos participantes.

Cada Gran Premio se celebra en un circuito determinado que debe cumplir una serie de normativas estipuladas por la FIA, los monoplazas que participan en las carreras también deben seguir unas normas establecidas y todo piloto que participe en una carrera de Fórmula 1 debe estar en posesión de una superlicencia válida expedida por la FIA.

El rendimiento de los pilotos y de los constructores del coche se evalúa al final de cada carrera mediante un sistema de puntos. Al final de una temporada, la FIA suma los puntos obtenidos por cada uno y se celebran dos Campeonatos del Mundo anuales: uno para los pilotos y otro para los constructores. El mundial de pilotos se celebra desde 1950 pero el de constructores no se comenzó a celebrar hasta el año 1958.

En esta sección, se explorará a fondo todo el universo que engloba la Fórmula 1 comenzando con la explicación de su historia desde el comienzo en el año 1950 hasta la actualidad, también se explicará los elementos característicos de la Fórmula 1 como circuitos, pilotos o escuderías y finalmente se analizará el impacto y relevancia que tiene hoy en día.

2.1.1. Historia de la Fórmula 1

Desde Farina y Fangio hasta Schumacher, Fernando Alonso y Hamilton, desde Alfa Romeo y Ferrari hasta Red Bull y Alpine, la historia de la Fórmula 1 está viva con el triunfo de la habilidad y la tecnología, la aplastante decepción de las oportunidades perdidas y la tragedia de las vidas cortadas. La historia de las carreras de Grand Prix (Grandes Premios en francés) es la saga del esfuerzo y los logros humanos, de la pura voluntad de ganar, de las cuales se pueden dividir en varias etapas:

Orígenes de la F1

La era moderna de la Fórmula 1 comenzó en 1950, pero las raíces de las carreras de Grand Prix son mucho más antiguas, desde las pioneras carreras en ruta en Francia en la década de 1890. La primera carrera automovilística propiamente dicha, fue una carrera en ruta de 1.200 km de París a Burdeos ida y vuelta, en 1895, ganada por Émile Levassor con su Panhard et Levassor en 48 horas. La primera carrera que usó la denominación "Gran Premio" fue el Gran Premio de Francia de 1901 en Le Mans, ganado por Ferencz Szisz con un Renault, quien realizó las 700 millas a 100 km/h.

Durante el comienzo del siglo XX se realizaron Grandes Premios por múltiples países de Europa como el "Gran Premio de Italia" ,"Gran Premio de España", "Gran Premio de Mónaco"...

El año 1931 vio la formación del Gran Premio Internacional, más tarde conocido también como Campeonato Europeo de Automóviles. Fue el primer campeonato internacional de pilotos en la historia del automovilismo. El reglamento original estipulaba carreras de 10 horas en Francia, Bélgica, Italia y España, con dos pilotos por cada coche debido a la duración de las carreras. Esta competición fue conocida como "Fórmula Libre" ya que los coches participantes no tenían restricción de peso o capacidad del motor. Los ganadores serían la pareja de conductores que en el período de 10 horas cubriera la distancia más larga. En esta época ya corrían con coches de marcas muy conocidas en la actualidad como Alfa Romeo, Bugatti, Maserati, Audi y Mercedes-Benz. [4]

Los primeros años

Después de la Segunda Guerra Mundial, el automovilismo inició una nueva fórmula, originalmente llamada Fórmula A pero que pronto se conocería como Fórmula 1, para autos con motores de 1.500 centímetros cúbicos (cc) sobrealimentados y 4.500 cc sin sobrealimentar. La distancia mínima de carrera se redujo de 500 km a 300 km.

En 1950 la FIA anunció planes para un Campeonato Mundial en una reunión celebrada ese año. El 10 de abril de 1950, Juan Manuel Fangio, a los mandos de un Maserati, ganó el Gran Premio de Pau, el primer certamen calificado como carrera de "Fórmula 1 Internacional". Un mes después Silverstone fue sede del Gran Premio de Gran Bretaña, la primera carrera de campeonato autorizada para autos de Gran Premio de Fórmula 1, y así nació el Campeonato Mundial de F1. Giuseppe "Nino" Farina ganó la primera carrera en Silverstone y posteriormente se coronó como primer campeón de la Fórmula 1 en 1950. Un año después en 1951 el campeonato mundial fue para Juan Manuel Fangio y coincidió con la primera victoria de la *Scuderia Ferrari*, la escudería más prestigiosa de la historia de la Fórmula 1. En esta época el piloto que más destacó fue el mencionado Fangio que ganó cinco campeonatos mundiales para cuatro constructores diferentes (Alfa Romeo, Maserati, Mercedes-Benz y Ferrari) y cuatro consecutivamente entre 1954 y 1957. En esta época los cascos eran de cuero y los pilotos utilizaban gafas protectoras, (ver Figura 2.1). [5]

Figura 2.1: Juan Manuel Fangio (extraído de [6])

La era británica

La era británica realmente comenzó en la temporada de 1958, cuando Moss ganó el Gran Premio de Argentina de 1958 en el Cooper T45 con motor trasero revolucionario, utilizando un motor Coventry-Climax de 2500 cc junto con el radiador montado en la parte delantera, obteniendo una combinación de excelente distribución de peso y manejo. Desde el descubrimiento de Cooper todos los campeones mundiales de F1 han tenido un motor trasero. En estos años también hubo un cambio de reglamento que prohibieron la gasolina de aviación.

La escudería británica Cooper reconoció que el consumo reducido de combustible y las carreras más cortas inevitablemente daban una ventaja a los autos más livianos, esto logró que fueran unas de las mejores escuderías de la época.

Entre 1962 y 1973, los equipos británicos de Fórmula 1 ganaron 12 campeonatos mundiales con pilotos de la talla de los escoceses Jim Clark y Jackie Stewart, el australiano Jack Brabham y el inglés Graham Hill con escuderías como Cooper, Lotus o Brabham Racing.

El mejor piloto de esta "era" fue Jim Clark, dos veces campeón de la Fórmula 1 que puede haber sido el piloto con más talento natural que jamás haya aparecido en la Fórmula 1, el cual fue magistral en condiciones de lluvia. Su temporada dominante fue en 1965 con el Lotus 33 , en la que lideró cada vuelta de cada carrera que terminó. Clark rompió el récord del inmortal Fangio de victorias en GP en su carrera en 1968 , pero murió unos pocos meses después pilotando en una carrera en Hockenheim.

Uno de los desarrollos más significativos en la historia de la Fórmula 1 ocurrió antes de la muerte de Jim Clark, en 1967, cuando Ford se asoció con el equipo Lotus para desarrollar el legendario motor Cosworth F1. Si bien las innovaciones de ingeniería anteriores, como el chasis monocasco, fueron técnicamente más importantes, el motor Ford Cosworth DFV 3.0 V8 dominaría la Fórmula 1 durante casi dos décadas. En este periodo ganó más de 150 Grandes Premios y ha sido el más persistente y exitoso de todos los motores de F1, el cuál marcó el comienzo de una era de constructores de Fórmula 1. [7]

Figura 2.2: Jim Clark con el motor Ford Cosworth en su espalda (extraído de [8])

La era de los alerones y el suelo

La tercera década de la Fórmula 1 moderna, la década de 1970, comenzó con el único Campeón Mundial de F1 póstumo Jochen Rindt y cerró con la llegada de los autos turboalimentados. La ingeniería aerodinámica de Fórmula 1 se desarrolló a un ritmo vertiginoso, comenzando con la introducción inicial de alerones. Los alerones permitieron la generación de carga aerodinámica, sujetando los autos a la pista para una mayor tracción y una velocidad en las curvas mucho mayor.

Ferrari volvió a la vanguardia de la F1 en 1975 con el 312T de 12 cilindros planos y los pilotos Niki Lauda y Clay Regazzoni. Para el año 1976 Lauda luchó con James Hunt (conduciendo el McLaren M23 Cosworth) para ganar seis de las primeras nueve carreras de la temporada de 1976. Pero en el Gran Premio de Alemania el 1 de agosto, Lauda estrelló su Ferrari en el circuito de Nürburgring con graves quemaduras faciales e inhalando gases tóxicos de la carrocería en llamas del automóvil, se esperaba que Lauda muriera pero logró recuperarse casi milagrosamente para regresar a la parrilla de salida sólo seis semanas después en el GP de Italia, donde terminó cuarto.

Los diseñadores de Fórmula 1 crearon en 1978 el "efecto de suelo" con el Lotus 78/79 . Los efectos de suelo convirtieron todo el coche en un gran alerón invertido, utilizando faldones laterales y diseño de la parte inferior de la carrocería para pegar literalmente el coche al circuito.

Mientras tanto, la economía y el negocio de las carreras de automovilismo Grand Prix también estaban cambiando rápidamente. Bernie Ecclestone reorganizó la gestión de los derechos comerciales de la Fórmula 1 en 1978, convirtiendo el deporte en un negocio global de mil millones de dólares. [9]

La era turbo

En la temporada de 1977 Renault volvió a entrar en la Fórmula 1 con un motor turboalimentado pero no es hasta 1983 cuando la era turbo comenzó a florecer, cuando Nelson Piquet ganó su segundo Campeonato del Mundo por dos puntos, esta vez usando un motor BMW turboalimentado.

En 1984, McLaren y su motor TAG turbo dominaron la temporada, ganando 12 de 16 carreras y el campeonato de constructores. Niki Lauda se llevó el título de pilotos por medio punto.

McLaren continuó su dominio en la década de 1980, ya sea con el motor TAG o con los motores turbos Honda. Alain Prost y Ayrton Senna ganaron múltiples títulos de pilotos con McLaren.

El año 1988 marcó el final de la era turbo, ya que los motores de aspiración normal se volvieron obligatorios a partir de 1989. También fue el último año de rivalidad pacífica entre Prost y Senna en McLaren, ya que su rivalidad se intensificaría en las siguientes temporadas.

Se puede destacar en esta era a los pilotos Alain Prost, Ayrton Senna, Niki Lauda, Gilles Villeneuve, Nelson Piquet y Emerson Fittipaldi. [10]

La era post-turbo

En la temporada 1987 el equipo Lotus presentó el primer coche de F1 con un sistema de "suspensión activa" controlado por ordenador. Más tarde se sumaron la caja de cambios semiautomática, el control de tracción, los programas de arranque controlados por "caja negra", los controles fly-by-wire y los frenos antibloqueo que produciría coches fabulosamente complejos y rápidos.

Al comienzo de la era post-turbo, McLaren seguía siendo supremamente dominante, pero sus dos estrellas, Ayrton Senna y Alain Prost , comenzarían una batalla personal que nunca llegó a su fin. Aunque acordaron no pelear por la primera curva de las carreras, su rivalidad se intensificó en el Gran Premio de San Marino de 1989 cuando Senna superó a Prost después de un accidente. Esta ruptura llevó a varios enfrentamientos entre los dos pilotos en los años siguientes que llevó a la marcha de Alain Prost a la *Scuderia Ferrari* tras asegurar que había un trato de favor de McLaren a Senna. Williams introdujo el revolucionario FW14 con una caja de cambios semiautomática y control de tracción. Senna, conduciendo un McLaren inferior, ganó las primeras cuatro carreras de la temporada, pero Nigel Mansell de Williams finalmente se llevó el campeonato de 1992 con un dominio impresionante. En 1993, Alain Prost regresó a Williams y ganó su cuarto y último campeonato mundial. Ese año también marcó el fin de una era con la prohibición de las "ayudas al conductor" en la Fórmula 1, como la suspensión activa y el control de tracción.[11]

Figura 2.3: Ayrton Senna con el McLaren (izquieda) y Alain Prost con el Ferrari (derecha) (extraído de [12])

Después del accidente de Tamburello

A raíz de la impactante muerte de Ayrton Senna en 1994 con 34 años y tres títulos de la Fórmula 1 en la infame curva Tamburello de Imola mientras dirigía el GP de San Marino, se impusieron nuevos reglamentos técnicos para reducir la velocidad de los autos y mejorar la seguridad de los conductores.

En esta era apareció el que luego se convertió en uno de los mejores pilotos de la historia, el alemán Michael Schumacher que tras la retirada de Prost y la ausencia de Mansell fue el único que compitió con Senna y consiguió dos mundiales consecutivos con la escudería Benetton en 1994 y 1995 compitiendo contra Damon Hill.

A la escudería Williams regresó Nigel Mansell que llevaba unos años retirado y el joven Jacques Villeneuve (hijo del mencionado en la era turbo Gilles Villeneuve).

Schumacher posteriormente se unió a Ferrari en 1996, llevándolos a la victoria en el GP de Italia tras casi una década sin victorias y desencadenando una nueva era en la Fórmula 1.[13]

Finales del siglo XX

La Fórmula 1 a finales del siglo XX fue testigo del surgimiento de nuevos pilotos y equipos preparados para tomar el mando de este deporte.

Llegó una nueva generación de pilotos talentosos como Schumacher, Jacques Villeneuve y Hakkinen. Fue una época llena de rivalidades que comenzó con la rivalidad entre Schumacher y Hill, realizando grandes batallas en pista. También hubo cambios en las reglas (neumáticos ranurados, monocascos estrechos) diseñadas para reducir la velocidad de los coches. La temporada de 1997 es destacada por el enfrentamiento entre Villeneuve y Schumacher en la última carrera, que resultó en la victoria y el título mundial para Villeneuve.

La temporada de 1998 se describe como una de las más emocionantes en años, con el dominio inicial de McLaren-Mercedes y junto a Ferrari protagonizaron una intensa batalla por el campeonato. Mika Hakkinen, de McLaren, y Michael Schumacher, de Ferrari, se enfrentaron en una lucha épica. Finalmente, Hakkinen se coronó campeón mundial con su habilidad y consistencia, superando a Schumacher por solo 14 puntos. Fue la primera vez que McLaren ganó el campeonato de constructores desde 1991 con Ayton Senna.

Esta rivalidad McLaren-Mercedes contra Ferrari y Hakkinen contra Schumacher también estuvo en las termporadas 1999 y 2000, ganando Hakkinen en 1999 y en el año 2000 Schumacher logró el campeonato. Así acabó con la crisis de Ferrari que llevaban sin ganar un título de pilotos desde 1979, alcanzando su tercer campeonato.

Comienzos del siglo XXI

El comienzo del siglo XXI de la Fórmula 1 fue testigo de como la *Scuderia Ferrari* logró ganar el mundial de pilotos después de una ausencia de 20 años, liderado por el incomparable Michael Schumacher, cuyos

cinco campeonatos mundiales consecutivos reescribieron literalmente el libro de récords de la F1.

La temporada 2001 de Fórmula 1 comenzó con caras nuevas que se unieron al circo de la F1. Dos futuros campeones del mundo comenzaron en la parte trasera de la parrilla, eran Fernando Alonso en la escudería Minardi y Kimi Räikkönen en Sauber, y otro ex campeón de CART/IndyCar, Juan Pablo Montoya, fue fichado por Williams.

Se realizaron nuevos cambios en los que se permitió nuevamente el uso del control de tracción, los neumáticos Michelin regresaron a la Fórmula 1, los equipos experimentaron con diferentes diseños aerodinámicos para mejorar el rendimiento. Se introdujo la clasificación de una vuelta para aumentar la exposición televisiva de los equipos más pequeños. Se introdujeron las pruebas opcionales de los viernes a mayores de la clasificación y la propia carrera. Se realizaron cambios en los chasis y motores para mejorar el rendimiento y la fiabilidad.

Los primeros años del siglo XXI fueron un dominio absoluto del Ferrari de Schumacher, el único piloto que le pudo competir fue Kimi Räikkönen en el año 2003.

En el año 2004 el equipo Jaguar de Ford fue comprado por el fabricante de bebidas energéticas Red Bull y continuó compitiendo en la Fórmula 1 como Red Bull Racing. En 2004 Schumacher consiguió su séptimo título de la Fórmula 1 logrando el récord que sigue vigente hoy en día igualado por el piloto británico Lewis Hamilton.

Pero el dominio de Ferrari se acabó en la temporada 2005 cuando el piloto español Fernando Alonso apareció con la escudería Renault. Alonso y Schumacher crearon una de las rivalidades históricas de la Fórmula 1 que se saldaron con dos campeonatos seguidos para el piloto español.

Con todos los records ganados (títulos, poles, vueltas rápidas, victorias y puntos) Michael Schumacher se retiró tras la temporada de 2006 finalizando con una victoria en el Gran Premio de Italia de 2006 en Monza en el circuito favorito de los Tifosi (fans de la *Scuderia Ferrari*). [14]

Figura 2.4: Fernando Alonso con el Renault seguido por el Ferrari de Michael Schumacher (extraído de [15])

La era después de Michael Schumacher

Todo cambió en la F1 con la desaparición de Schumacher, ya que el deporte se volvió más joven. Fue un cambio refrescante, con el arte del pilotaje y la estrategia primordiales.

En 2007, el debutante Lewis Hamilton sorprendió a todos al luchar por el campeonato mundial en su primera temporada con McLaren. Sin embargo, Kimi Räikkönen de Ferrari se alzó con el título en la última carrera, superando a Hamilton y al piloto de McLaren, Fernando Alonso.

En 2008, Hamilton se tomó la revancha. En una emocionante última vuelta en el Gran Premio de Brasil, adelantó a Timo Glock de Toyota en condiciones de lluvia y se coronó campeón del mundo por un solo punto, dejando atrás a Felipe Massa de Ferrari.

El año 2009 trajo consigo cambios significativos en la reglamentación técnica y vio el surgimiento de Brawn GP creado por Ross Brawn, que fue director técnico de Ferrari y Benetton en los años de Schumacher, el cuál compró lo que quedaba de Honda después de que el constructor japonés se retirara de la F1, llevándose sus motores con ellos. Brawn rápidamente renombró al equipo con su propio nombre,

instaló la potencia de Mercedes en la parte trasera del chasis y, gracias a una lectura atenta de las nuevas normas técnicas de la temporada, ideó un "difusor doble" trasero que funcionó años luz mejor que el de cualquier otro. El equipo, liderado por Jenson Button y con un sorprendente rendimiento, dominó la temporada, ganando ocho de las primeras nueve carreras. Button se llevó el título de manera dominante, con Brawn GP asegurando el campeonato de constructores en su única temporada en la Fórmula 1.

En 2010, comenzó una intensa rivalidad entre los pilotos de Red Bull Racing, Sebastian Vettel y Mark Webber. Vettel se llevó el campeonato tras una emocionante batalla con Alonso y Hamilton en la última carrera en Abu Dhabi. Este fue el primer título mundial de Vettel y también el primero para Red Bull Racing.

La era de Vettel continuó en 2011 y 2012, donde dominó la Fórmula 1. Ganó ambos campeonatos de manera consecutiva, convirtiéndose en el tricampeón mundial más joven en la historia de la Fórmula 1. Red Bull Racing demostró ser el equipo a vencer durante estos años, gracias a su enfoque innovador y al talento de Vettel.

Mientras tanto, Ferrari, McLaren y Mercedes también estuvieron en la lucha por el título. Alonso, en particular, se mantuvo como un contendiente constante, logrando el subcampeonato en 2010 y 2012.

En 2013, Sebastian Vettel continuó su dominio al ganar su cuarto campeonato mundial consecutivo con Red Bull Racing. Sin embargo, el cambio de reglamentación en 2014 introdujo los motores híbridos V6 turbo, lo que llevó a un cambio en el panorama competitivo. [16]

La era híbrida

Después de un notable cuarto campeonato mundial para Sebastian Vettel con Red Bull, las reglas de la F1 y los motores cambiaron una vez más, pero marcaron el comienzo de un período de dominio sostenido de Mercedes AMG y Lewis Hamilton.

A partir de 2014, Mercedes AMG Petronas se estableció como el equipo a vencer, liderado por Lewis Hamilton y Nico Rosberg. Los dos pilotos crearon una intensa rivalidad por el campeonato, con Hamilton llevándose los títulos en 2014, 2015, 2017 y 2018, y Rosberg ganando el campeonato en 2016 antes de retirarse.

Durante estos años, hubo un aumento en la competitividad de otros equipos. Red Bull Racing, con Max Verstappen como piloto destacado, desafió a Mercedes en varias ocasiones, y en 2021, Verstappen se convirtió en el piloto más joven en ganar el campeonato mundial de Fórmula 1.

En cuanto a Ferrari, el equipo italiano experimentó altibajos en su rendimiento. En 2017, Sebastian Vettel lideró la lucha por el campeonato contra Hamilton, pero finalmente quedó en segundo lugar. Ferrari luchó por mantenerse competitivo en los años siguientes y experimentó cambios en la dirección del equipo.

En 2022, la Fórmula 1 introdujo cambios en el reglamento técnico con el objetivo de promover carreras más igualadas y emocionantes. Estos cambios incluyeron modificaciones aerodinámicas y un nuevo formato de fin de semana de carrera con la inclusión de las carreras a sprint. [17]

Concluyendo la historia de la Fórmula 1, se puede apreciar en la Figura 2.5 una notable evolución en los monoplazas, los cuales han experimentado una transformación hacia vehículos más veloces y al mismo tiempo más seguros. Esta evolución se refleja en la implementación de tecnologías avanzadas, mejoras aerodinámicas y sistemas de seguridad cada vez más sofisticados.

Figura 2.5: Evolución de los coches de Fórmula 1 de la *Scuderia Ferrari* entre el año 1950 y 2012 (extraído de [18])

2.1.2. Federación Internacional del Automóvil (FIA)

La Federación Internacional del Automóvil (FIA), desempeña un papel fundamental en el mundo de la Fórmula 1, siendo el organismo encargado de regular y supervisar este deporte a nivel mundial. Como máxima autoridad la FIA tiene la responsabilidad de establecer las normas y reglamentos que rigen la competición, garantizando la seguridad de los pilotos, promoviendo la equidad y fomentando la competitividad entre las escuderías.

Fundada en 1904, con sede en París es una asociación sin ánimo de lucro. Reúne a 244 organizaciones automovilísticas y deportivas internacionales de 145 países de los cinco continentes.

Sus responsabilidades son: [19]

- Crear y modificar las reglas deportivas como función legislativa principal
- Tomar y revisar decisiones ejecutivas sobre la gestión de recursos financieros y la organización de competiciones deportivas
- Resolver disputas entre miembros, participantes deportivos y otras partes relevantes

2.1.3. Pilotos de la Fórmula 1

Para garantizar la seguridad y la igualdad de condiciones en la competición, existen normas y regulaciones estrictas que deben seguir tanto los pilotos como los equipos.

En primer lugar, los pilotos deben cumplir con requisitos específicos para obtener una superlicencia de la FIA. Estos requisitos incluyen la experiencia en otras categorías de automovilismo y la obtención de suficientes puntos en su carrera deportiva. Además, los pilotos deben someterse a exámenes médicos regulares para garantizar su aptitud física y mental.

En cuanto al equipamiento, los pilotos de Fórmula 1 deben utilizar trajes de competición ignífugos, diseñados para protegerlos en caso de incendios. Estos trajes están hechos de materiales resistentes al fuego (Nomex) y están equipados con sistemas de refrigeración para mantener la temperatura corporal bajo condiciones extremas. Los pilotos también deben llevar ropa interior ignífuga, pantalón, pasamontañas, guantes y zapatos Nomex.

El casco es la única protección para la cabeza del conductor junto con el HALO del que se hablara más adelante. Es la parte más importante en la seguridad del conductor e incluso juega un papel importante en la aerodinámica del automóvil. Los cascos se pueden construir de tal manera que guíe el aire de una manera determinadas y así aumentar la potencia del motor y, por lo tanto, la velocidad general.

Los componentes individuales de los cascos son un secreto altamente guardado: pero si se conocen los tres materiales principales: fibra de carbono para la rigidez, Nomex ignífuga y polietileno, que también se utiliza en la producción de chalecos antibalas. A estos se suman aluminio, magnesio y resina epoxi. Los compuestos de fibra de carbono (CFC) utilizados hacen que los cascos sean extremadamente ligeros, con un peso aproximado de 1.25 kg. Los cascos disponen de una radio en el área de la barbilla proporciona los medios para la comunicación con la escudería.

Una buena visibilidad, incluso en las situaciones meteorológicas más difíciles, es vital para los conductores. Esto lo proporciona la visera de 3 mm de espesor del casco. Está fabricado en policarbonato ignífugo. El tinte se ajusta en fracciones de segundo, similar a algunas gafas de sol, solo que mucho más rápido. Por ejemplo, en la entrada del túnel en Mónaco, la visera se ilumina y se vuelve a oscurecer en una fracción de segundo a la salida del túnel. Todas las partes de un casco de Fórmula 1 se pueden observar en la Figura 2.6.

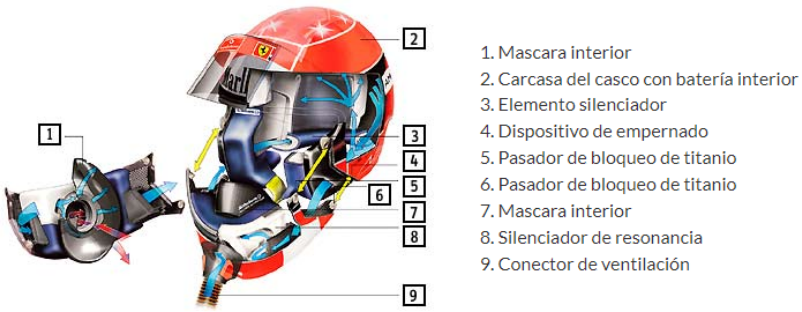


Figura 2.6: Partes de un casco de la Fórmula 1 (extraído de [20])

Para que la FIA, otorgue su aprobación para un casco, debe pasar una serie de pruebas de choque diferentes. Durante la llamada “prueba de penetración”, se deja caer un objeto metálico puntiagudo de 3 kg desde una altura de tres metros sobre el casco, que debe permanecer intacto. También se prueba

el ajuste seguro del casco en la cuál la correa se somete a una carga de 38 kg. El visor es bombardeado con proyectiles que viajan a aproximadamente 500 km/h. Los puntos de impacto no pueden tener una profundidad superior a 2,5 mm. Por último, pero no menos importante, el casco se somete a una llama de 800 °C durante 45 segundos en la prueba de fuego. Durante este tiempo, las temperaturas en el interior del casco no pueden superar los 70 °C.

Para la protección del cuello de un piloto existe el sistema de soporte para la cabeza y el cuello (HANS) ya que los conductores se enfrentan a tensiones de desaceleración teóricas de hasta 80 veces la fuerza de la gravedad en un accidente. En tal situación, el peso de la cabeza y el casco aumenta rápidamente de 7 kg a 560 kg. HANS ayudaría a absorber esta tensión, así como a evitar que la cabeza del conductor golpee el volante o el borde delantero de la cabina. El sistema fue diseñado en la Universidad de Michigan por el Dr. Robert Hubbard. [20]

En la Figura 2.7 se puede observar todo el equipamiento de un piloto actual de la Fórmula 1, guantes, traje ignífugo, casco y HANS.

Figura 2.7: Equipamiento de piloto de Fórmula 1 (extraído de [21])

Durante las carreras, los pilotos están sujetos a reglas de competición que dictan su comportamiento en pista. Deben respetar las banderas y las señales de los comisarios de pista, seguir los límites de la pista y evitar maniobras peligrosas que pongan en riesgo a otros competidores. También se aplican reglas estrictas para adelantar y para evitar bloquear o empujar a otros pilotos de manera antideportiva.

En resumen, los pilotos de Fórmula 1 deben cumplir con normas estrictas, tanto en términos de licencia y requisitos médicos como en el uso de equipamiento adecuado. Además, están sujetos a reglas de competición y a regulaciones técnicas para garantizar la seguridad.

2.1.4. Circuitos de la Fórmula 1

Las carreras se disputan en circuitos especialmente construidos y homologados por la FIA. La mayoría de los circuitos están situados en lugares remotos bien comunicados con las ciudades como puede ser el circuito de Silverstone, son los llamados circuitos permanentes. Hay otras carreras, como el Gran Premio de Mónaco y el Gran Premio de Bakú, que se celebran en carreteras públicas cerradas dentro de las propias ciudades, son los llamados circuitos urbanos (también llamados temporales), ambos tipos de circuito se pueden ver en la Figura 2.8.

Figura 2.8: Vista aéra del circuito permanente de Silverstone (izquierda) y del circuito urbano de Mónaco (derecha) (extraídos de [22] y [23])

Las carreras se disputan en circuitos clasificados de grado "1" por la FIA. De ahí que se adoptara el nombre de Fórmula 1. Los circuitos están realizados con asfalto debido a su capacidad para ofrecer buen agarre y tracción, así como su flexibilidad para adaptarse a diferentes condiciones climáticas y niveles de desgaste.

No hay reglas que dicten qué forma debe tener una pista, pero aún hay una serie de regulaciones que los circuitos deben cumplir. Las rectas no pueden tener más de 2 km, y las pistas deben tener al menos 3,5 km de largo en total. La única excepción es el Gran Premio de Mónaco, que se queda corto con 3,337 km.

Se recomienda que los circuitos nuevos no superen los 7 km de longitud. Los circuitos permanentes deben tener al menos 12 metros de ancho en todos los puntos, aunque en ciertos circuitos urbanos temporales no se cumple.

Mientras tanto, la parrilla de salida debe tener al menos 15 metros de ancho, manteniendo el ancho de la pista en la primera curva para minimizar el riesgo de colisiones en la primera vuelta. La primera esquina debe tener un cambio de dirección de al menos 45 grados, con un radio inferior a 300 metros.

El pitlane, el cual es el carril adyacente a la pista principal que usan los vehículos cada vez que necesitan entrar en boxes, también está sujeto a un ancho mínimo: debe tener al menos 12 metros de ancho y estar adyacente a la recta de salida y llegada. Los puntos de entrada y salida de boxes no pueden interferir con la línea de carrera para que los autos no choquen cuando se reincorporan a la pista.

La FIA también entra en detalles sobre la pendiente de los circuitos. La pendiente de la recta de inicio-meta no puede exceder el 2 %, e idealmente no debe cambiar en zonas de frenado de alta velocidad o en curvas con un alto nivel de aceleración. La inclinación no debe exceder los 5.7 grados, aunque la FIA permite "posibles excepciones en casos especiales". [24]

Dado que los conductores pueden correr un grave riesgo de sufrir lesiones en caso de un choque a alta velocidad, es obligatorio que los circuito tengan un centro médico permanente.

2.1.5. Gran Premio (GP)

Un Gran Premio de Fórmula 1 se lleva a cabo durante un fin de semana. El viernes se designa como día de sesión de práctica cuando se realizan dos sesiones de práctica libre. Las sesiones de práctica se adelantan un día en Mónaco y se llevan a cabo el jueves.

Las sesiones de clasificación se llevan a cabo los sábados y la sesión determinará el orden de salida de la carrera. El Gran Premio se lleva a cabo el domingo y es el evento principal del fin de semana.

Las sesiones de calificación de F1 son cuando los pilotos se esfuerzan por establecer los tiempos de vuelta más rápidos que determinarán su lugar de inicio en la parrilla de salida. Hoy en día la calificación F1 se divide en tres partes: Q1, Q2 y Q3. Cada sesión tiene una duración determinada, con Q1 que dura 18 minutos, Q2 que dura 15 minutos y Q3 que dura 12 minutos.En cada sesión los pilotos compiten para establecer el tiempo de vuelta más rápido durante un período de tiempo específico. Cada fase es un sistema de eliminatorias.

En la Q1, la primera sesión de calificación, a los cinco pilotos más lentos se les asignan los últimos cinco lugares en la parrilla. Los 15 pilotos restantes (solo hay 20 autos en una carrera actualmente) deciden las siguientes cinco peores posiciones de salida en la Q2.

En la Q3 (Clasificación 3), los últimos diez pilotos se disputan el mejor puesto de la parrilla de salida. Los pilotos tienen 12 minutos para establecer sus tiempos de vuelta más rápidos. El piloto con el tiempo de vuelta más rápido obtiene la pole position en la parrilla de salida (lograr salir primero el día de la carrera). Los pilotos restantes se clasifican de acuerdo con sus tiempos de vuelta más rápidos y ocuparán sus lugares designados al comienzo de la carrera.

Si un automóvil no cumple con las normas de la FIA, el conductor no podrá participar en las sesiones de calificación. El coche tiene que salir desde la última posición de la parrilla para la carrera. Cualquier penalización durante la calificación se aplica al final de la calificación. La penalización determinará la posición de salida del piloto en la carrera. Las infracciones pueden incluir cambiar una caja de cambios o un componente del motor. [25]

Pero toda la organización del Gran Premio ha cambiado en el año 2021 con la inclusión de las carreras a sprint, las cuales en ciertos Grandes Premios (en 2022 fueron 3 y en 2023 serán 6) han incluido una sesión en la que todos los monoplazas recorrerán una distancia de al menos 100 kilómetros sin necesidad de entrar a cambiar neumáticos en boxes. Esta sesión se realiza en sábado y desde el 2023, ya no determina el orden de salida de la carrera habitual del domingo. Anteriormente, el ganador del sprint partía primero en el GP. Ahora, el GP tiene su propia sesión de clasificación (viernes), lo mismo que el sprint. Esta tomó el nombre de sprint shootout. En el sprint, se otorgan puntos a los ocho primeros en orden descendente, 8-7-6-5-4-3-2-1. [26, 27]

Una carrera de Gran Premio comienza con una vuelta de calentamiento, llamada vuelta de formación. Al final de la vuelta, los pilotos se reúnen en sus lugares designados en la parrilla de salida. Cinco luces rojas se encienden en un intervalo de un segundo por encima de la pista. La carrera comienza cuando todas estas cinco luces se apagan simultáneamente.

La carrera podrá reiniciarse en caso de accidente grave o inclemencias del tiempo. Los pilotos se volverán a reunir en el formato de salida y la carrera se reiniciará desde cero.

Una carrera de Grand Prix está limitada a un máximo de dos horas. La FIA ha estandarizado la distancia de carrera a 305 km. Sin embargo, las pistas más lentas como Mónaco tienen una distancia más corta que cubrir para permitir que los autos completen la carrera dentro del tiempo estipulado. Los conductores pueden parar en boxes para cambiar neumáticos u otros componentes dañados.

2.1.6. Puntuación de los pilotos

En el sistema actual de otorgamiento de puntos del campeonato, el piloto debe terminar una carrera entre los diez primeros para obtener un punto. El piloto con la vuelta más rápida recibe un punto, ya sea que termine entre los diez primeros o no. Los que terminan fuera de los diez primeros no reciben ningún punto. La siguiente tabla muestra los puntos otorgados. [28]

El piloto completó el 90 % de la distancia de carrera del ganador	
1er lugar	25 puntos
Segundo lugar	18 puntos
3er lugar	15 puntos
4to lugar	12 puntos
5to lugar	10 puntos
6to lugar	8 puntos
7mo lugar	6 puntos
8vo lugar	4 puntos
9no lugar	2 puntos
décimo lugar	1 punto
Vuelta más rápida	1 punto (siempre que el conductor termine entre los 10 primeros)
11° lugar en adelante	Sin puntos

Figura 2.9: Puntuación actual en la Fórmula 1 (extraído de [28])

Pero esto no ha sido así siempre, el sistema de puntuación de pilotos en la Fórmula 1 ha ido evolucionando con los años, al principio sólo puntuaban los 5 primeros, esto se ha ido aumentando hasta los 10 primeros pilotos reciben algún punto en la actualidad. [29]

Posición	1950-59(*)	1960	1961-1990	1991-2002	2003-09	2010-actual(**)
1º	8 puntos	8 puntos	9 puntos	10 puntos	10 puntos	25 puntos
2º	6 puntos	6 puntos	6 puntos	6 puntos	8 puntos	18 puntos
3º	4 puntos	4 puntos	4 puntos	4 puntos	6 puntos	15 puntos
4º	3 puntos	3 puntos	3 puntos	3 puntos	5 puntos	12 puntos
5º	2 puntos	2 puntos	2 puntos	2 puntos	4 puntos	10 puntos
6º	-	1 punto	1 punto	1 punto	3 puntos	8 puntos
7º	-	-	-	-	2 puntos	6 puntos
8º	-	-	-	-	1 punto	4 puntos
9º	-	-	-	-	-	2 puntos
10º	-	-	-	-	-	1 punto

Figura 2.10: Evolución de la puntuación en la Fórmula 1 (extraído de [29])

Los puntos anotados por el piloto se suman al final de la temporada. El piloto que ha acumulado la mayor cantidad de puntos en una temporada es premiado con el Campeonato Mundial de Pilotos. Incluso si un piloto cambia de equipo a mitad de temporada, los puntos anotados por él aún cuentan y se sumarán al final de la temporada. Un piloto también puede recibir puntos si se detiene a mitad de la carrera pero ha completado el 90 por ciento de la distancia recorrida por el ganador de la carrera.

La escudería que sume el máximo de puntos sumando los puntos de los pilotos que conforman la escudería en una temporada obtendrá el Campeonato Mundial de Constructores.

2.1.7. Escuderías

La Fórmula 1 es una competición en la que participan varias escuderías, cada una con dos pilotos (suelen tener un tercer piloto reserva).

Las escuderías de la Fórmula 1 desempeñan un papel fundamental en la competición, encargándose de preparar y gestionar los monoplazas para cada carrera. Antes de la carrera, realizan una serie de tareas que incluyen el diseño y construcción del monoplaza, la preparación y puesta a punto de los coches, la revisión y ajuste de los sistemas, y la estrategia de carrera. Los equipos trabajan en colaboración con ingenieros, mecánicos y pilotos para optimizar el rendimiento del coche y adaptarlo a las características de la pista.

Durante la carrera, las escuderías monitorean constantemente el rendimiento del coche y toman decisiones estratégicas. Utilizan telemetría y sistemas de comunicación en tiempo real para recopilar información y ajustar la configuración del monoplaza. También se encargan de la logística, el suministro de combustible y neumáticos, y coordinan las paradas en boxes.

Las escuderías de Fórmula 1, son organizaciones complejas que cuentan con numerosos empleados altamente especializados. Esto incluye ingenieros, diseñadores, mecánicos, analistas de datos, estrategas, personal de logística y más. El tamaño de las escuderías puede variar, pero generalmente emplean a cientos de personas dedicadas a diversas áreas para asegurar el funcionamiento eficiente del equipo.

Las escuderías de Fórmula 1, a menudo, están asociadas con marcas automovilísticas reconocidas. Muchos equipos son propiedad de fabricantes de automóviles, como Mercedes, Ferrari,y McLaren, pero también existen marcas alejadas del mundo del motor como la marca de bebidas energéticas Red Bull. Estas marcas aprovechan la Fórmula 1 como plataforma de marketing para promover su imagen y tecnologías avanzadas.

Los pits stops, o paradas en boxes, son momentos cruciales durante la carrera en los que los equipos realizan cambios de neumáticos, ajustes aerodinámicos, reabastecimiento de combustible y reparaciones rápidas. Durante los pits stops, los mecánicos trabajan en equipo para realizar estas tareas en el menor tiempo posible, ya que cada segundo cuenta. Las paradas en boxes son una parte esencial de la estrategia

de carrera, ya que una parada rápida y eficiente puede marcar la diferencia en el resultado final. El record de pit stop lo tiene la escudería Red Bull Racing con 1.82 segundos en 2019. [30]

Figura 2.11: Parada en boxes (pit stop) de la escudería Red Bull Racing (extraído de [31])

2.1.8. Monoplazas

Un automóvil de Fórmula 1 es un automóvil de carreras de un solo asiento, de cabina abierta y de cuatro ruedas con el propósito de ser utilizado en competencias de Fórmula 1. Está equipado con dos alerones (delantero y trasero) más un motor, que se encuentra detrás del conductor.

Cada monoplaza de Fórmula 1 se compone de alrededor de 14 500 componentes individuales y cada elemento se fabrica a medida con el diseño asistido por ordenador (CAD) y la fabricación asistida por ordenador (CAM) o incluso el procesamiento manual para construirlos.

A diferencia de otras competiciones de carreras, los monoplazas son completamente únicos para cada equipo, a diferencia de la F2, que tiene un auto estándar, o incluso la Fórmula E, que tiene un chasis estándar, pero en la F1 ha habido una tendencia hacia la estandarización de algunas partes.

El reglamento técnico de la FIA define una serie de reglas que limitan los diseños de los equipos. Eso incluye establecer dimensiones específicas para alerones, definir áreas aerodinámicas prohibidas y prohibir ciertas aleaciones de alto costo en los motores.

Cada año se produce un automóvil nuevo, pero a diferencia de un automóvil de carretera, que generalmente permanece igual una vez que sale de la fábrica, un automóvil de F1 se desarrolla continuamente durante la temporada, con nuevas piezas que se introducen carrera por carrera.

La aerodinámica juega un papel crucial en el diseño de los monoplazas de Fórmula 1. Con alerón delantero, difusores, elementos de carrocería y alerón trasero ajustable, se busca maximizar la carga aerodinámica y mejorar la estabilidad del vehículo a altas velocidades. Estos componentes se diseñan con precisión para generar el equilibrio adecuado entre agarre aerodinámico y resistencia al aire.

Además, los monoplazas de Fórmula 1 están equipados con sistemas de telemetría y electrónica que permiten a los ingenieros monitorear y ajustar el rendimiento del vehículo en tiempo real. Los datos recopilados durante las carreras se analizan detalladamente para optimizar el rendimiento y la estrategia.

Alrededor del 80 por ciento del automóvil está hecho de resinas compuestas y de fibra de carbono. Cada monoplaza de F1 se compone de dos componentes principales: el chasis y el motor.

El chasis de un monoplaza de Fórmula 1 es la estructura principal del vehículo que proporciona rigidez y soporte a todos los componentes, los chasis actualmente están hechos de fibra de carbono y componentes ultraligeros. El peso no debe ser inferior a 702 kg incluido el conductor y los neumáticos, pero sin incluir el combustible.

Las dimensiones de un automóvil de Fórmula 1 en 2022 deben ser como máximo de 180 cm (ancho) × 95 cm (alto); no hay un número específico para la longitud máxima, pero todos los automóviles tienden a tener casi la misma longitud.

El motor de acuerdo con los cambios normativos de 2014, todos los coches de F1 deben utilizar motores V6 turboalimentados de 1.6 litros, el combustible utilizado por los autos de Fórmula 1 es una mezcla

estrictamente controlada de gasolina común. Los monoplazas de Fórmula 1 han estado usando neumáticos lisos desde 2009. Las dimensiones de los neumáticos de un auto de F1 son 245 mm (ancho) para el neumático delantero y 355 mm (ancho) para el trasero.

El volante de un automóvil de F1 está equipado para realizar múltiples funciones. En la parte frontal del volante, se encuentra un conjunto de botones y diales que permiten al piloto realizar ajustes en tiempo real. Estos controles pueden incluir ajustes de la mezcla de combustible, modos de motor, distribución de frenado, ajustes de diferencial y configuraciones de tracción. Estas opciones permiten al piloto personalizar y optimizar el rendimiento del coche según las condiciones de la pista y la estrategia de carrera. En la parte superior del volante se encuentran las levas de cambio, que permiten al piloto cambiar de marcha sin necesidad de quitar las manos del volante. El volante también puede estar equipado con una pantalla LCD o un panel de información en el centro, que muestra información muy importante como la velocidad, las revoluciones del motor, la temperatura y la presión de los neumáticos. Esto proporciona al piloto información en tiempo real sobre el rendimiento del coche y le permite tomar decisiones rápidas y precisas durante la carrera. [32]

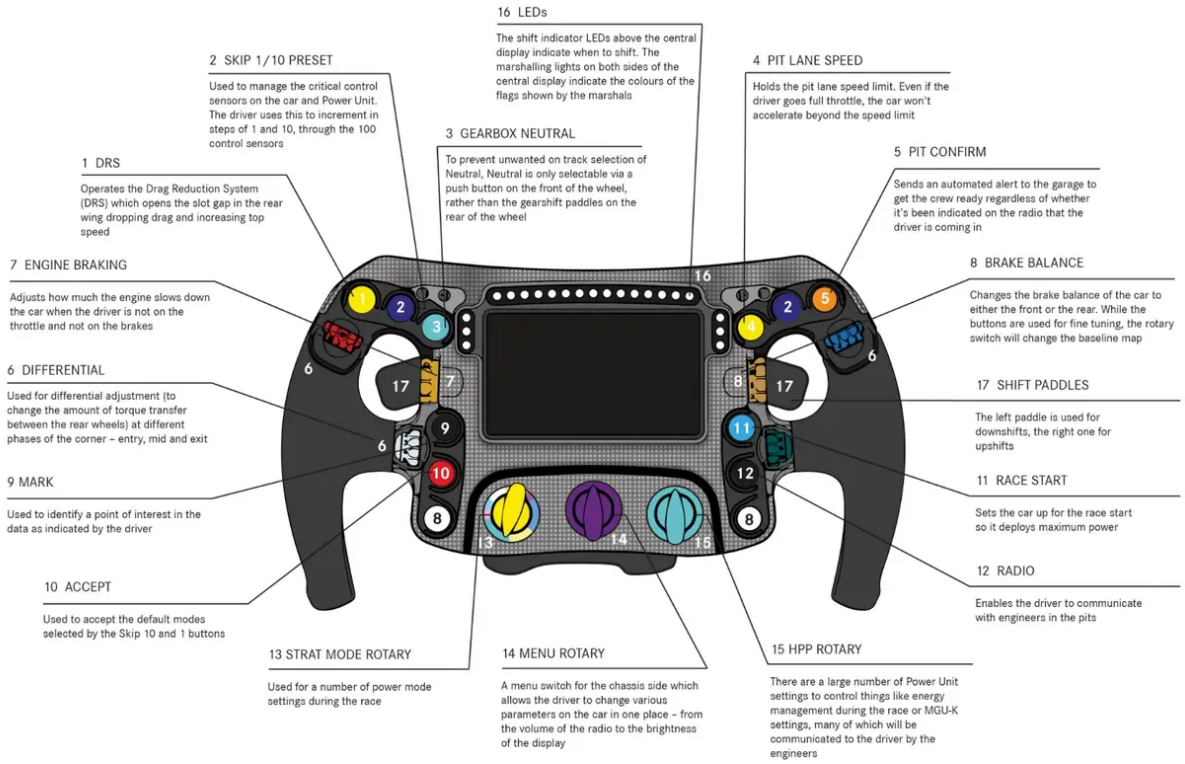


Figura 2.12: Volante y descripción de sus partes (extraído de [32])

El halo es un dispositivo de seguridad introducido en la Fórmula 1 con el objetivo de proteger la cabeza de los pilotos durante los accidentes y mejorar la seguridad en el deporte. Consiste en una estructura en forma de arco de titanio que se encuentran la parte superior de la cabina del automóvil y está compuesta por una barra central vertical y dos soportes laterales en forma de aro, está diseñada para soportar un peso de 12 toneladas. El halo fue introducido en la Fórmula 1 en la temporada 2018. Su objetivo principal es prevenir lesiones graves en la cabeza en caso de impacto de objetos como neumáticos, piezas de otros coches o fragmentos de barreras de seguridad. Aunque inicialmente hubo cierta controversia sobre su estética y la posibilidad de afectar la visibilidad de los pilotos, el halo se ha convertido en un elemento estándar en los monoplazas de Fórmula 1. Ha demostrado su eficacia en varios accidentes en los que ha protegido a los pilotos de impactos potencialmente letales. [33]

Figura 2.13: Arriba monoplaza de F1 sin halo y debajo monoplaza de F1 con halo

Todos los monoplazas de F1 pueden acelerar de 0 a 160 km/h y desacelerar de nuevo a 0 en menos de 5 segundos. Los monoplazas de F1 han alcanzado velocidades máximas de alrededor de 300 km/h. [34]

Cada escudería tiene un tope de 140 millones de dólares por temporada en 2022, y se reduce a 135 millones de dólares a partir de 2023. Esto cubre todos los costos de rendimiento del automóvil y excluye el marketing y los salarios de los conductores y los tres miembros más caros del equipo. Las estimaciones de los costos de un monoplaza sugieren que ronda los 6 millones de libras esterlinas. Los motores son la parte más cara, se estima que construir una sola unidad ronda los 3,5 millones de libras esterlinas. [35]

2.1.9. Accidentes y seguridad

La Fórmula 1, como muchos deportes de motor, solía ver la seguridad como una preocupación secundaria a la velocidad. Sus primeros años estuvieron plagados de muertes no solo entre los pilotos, sino también entre los comisarios de pista y los espectadores.

El tres veces campeón mundial Jackie Stewart fue uno de los defensores más enérgicos de la seguridad en la F1, inspirado por un accidente aterrador que tuvo en 1966. Stewart hizo campaña por la introducción obligatoria de cinturones de seguridad y cascos integrales, y por mejoras en la seguridad en la pista. El progreso fue lento, Niki Lauda intentó liderar un boicot de pilotos al Gran Premio de Alemania de 1976 por motivos de seguridad, el resto de pilotos se negaron a apoyarlo y, en un giro cruel de los acontecimientos, Lauda sufrió un terrible accidente en el que su automóvil explotó en llamas, dejándolo con quemaduras faciales permanentes. En la carrera final de esa temporada, liderando el campeonato, Lauda se retiró de la contienda en lugar de arriesgar su vida por otro título, su gran rival James Hunt corrió y finalmente se llevó el título por un solo punto. Esta historia fue adaptada al cine de la mano del director Ron Howard dirigiendo la película Rush en el año 2013.

Figura 2.14: Película Rush (2013) que narra la historia del accidente de Niki Lauda (extraído de [36])

Desde el Gran Premio de San Marino en 1994, donde los accidentes se cobraron la vida del novato Roland Ratzenberger y el icónico tres veces campeón mundial Ayrton Senna, la Fórmula 1 se ha tomado en serio la seguridad. Desde ese fatídico fin de semana, ha habido una única muerte en la Fórmula 1: Jules Bianchi sufrió una extraña colisión con un tractor grúa que estaba retirando otro automóvil accidentado en el Gran Premio de Japón de 2014 y sucumbió a sus heridas nueve meses después.

Actualmente están incorporados nuevos métodos para aumentar la seguridad: los cascos son obviamente obligatorios y deben someterse a vigorosas pruebas de calor y estrés antes de que se considere seguro; el HANS (soporte para la cabeza y el cuello) de fibra de carbono protege las vértebras del conductor en caso de choque, su traje (como los que debe usar cada equipo de boxes) está hecho de material ignífugo multicapa, y los guantes biométricos miden sus signos vitales y los transmiten al personal médico. Los cinturones de seguridad están tan apretados que los conductores necesitan que un miembro del equipo los abroche.

Los reglamentos técnicos ponen más énfasis en las consideraciones de seguridad que en el rendimiento, y los autos mismos se someten a pruebas rigurosas de resistencia al fuego y estrés. La celda de supervivencia del automóvil está diseñada para proteger al conductor en caso de choque, y los automóviles están equipados con extintores de incendios. Los recorridos en sí están meticulosamente planificados y mejorados para proteger a los pilotos, comisarios y espectadores.

Un gran cambio que proporcionó un aumento de la seguridad de los pilotos fue la inclusión el del previamente mencionado halo en la temporada de 2018, al comienzo tuvo muchas críticas de fanáticos y de pilotos pero cuando Fernando Alonso fue desviado por encima del auto de Charles Leclerc unas semanas después de añadirlo, se aceptó universalmente que el halo había evitado otra tragedia, y el halo ahora es ampliamente aceptado como una necesidad moderna. Entre otros incidentes, el halo probablemente salvó la vida de Romain Grosjean en su aterrador accidente en el Gran Premio de Bahrein del año pasado. [37]

Otro elemento útil para reducir el número y gravedad de los accidentes son las banderas, actualmente muchos lugares utilizan pantallas electrónicas para indicar banderas y dar varios mensajes a los conductores. Sin embargo, los comisarios de carrera siguen utilizando banderas físicas como mecanismo de redundancia en caso de fallo de la pantalla electrónica. Los comisarios se colocan en numerosos puntos alrededor de la pista durante cada carrera. Las banderas tienen diferentes significados según su color.

Hay 10 banderas principales en la F1, y los conductores pueden ser sancionados si no cumplen con los requisitos de cada una cuando están en uso. [38]

- Bandera a cuadros: Pone fin a una sesión, salvo la finalización de la vuelta en la que se encuentran para los competidores. En una carrera, el primer piloto en cruzar la línea de meta mientras se ondea esta bandera es el ganador.

- Bandera roja: Una bandera roja significa que se suspendió una sesión y los pilotos deben regresar lentamente a boxes, y la sesión podría reanudarse con los autos en el orden en que se detuvieron. Esta bandera se puede usar debido a una pista bloqueada o clima peligroso que podría causar un peligro inminente para los competidores o espectadores.

- Bandera amarilla: Una sola bandera amarilla significa que los conductores deben reducir su velocidad, evitar adelantar debido a un peligro en la pista o cerca de ella. Dos banderas amarillas es una orden para reducir significativamente la velocidad, evitar adelantar y prepararse para detenerse debido a un peligro que bloquea la pista.

- Bandera amarilla y roja: Esta bandera indica un cambio en la superficie de la pista, como agua o aceite que reduce el agarre de la sección.

- Bandera verde: Indica que se ha solucionado un peligro anterior y que se pueden reanudar las condiciones normales de carrera.

- Bandera azul: A los conductores que están a punto de ser doblados se les muestra esta bandera para indicarles que dejen pasar al automóvil que está detrás de ellos. Una bandera azul estacionaria significa que otro automóvil se acerca por la pista cuando un conductor abandona el pit lane. Los conductores pueden ser penalizados por ignorar las banderas azules repetidamente.

- Bandera blanca y negra: Dividida entre los dos colores por una línea diagonal, esta bandera es fija e incluye el número de un conductor que está siendo advertido por un comportamiento antideportivo.

- Bandera Negra: Acompañada de un número blanco, esto significa la descalificación para el piloto en cuestión.

- Bandera blanca: La bandera blanca simboliza que un vehículo de movimiento lento, como un automóvil médico, está adelante. También se utiliza para indicar que una sesión de práctica ha terminado.

- Bandera negra y naranja: Una bandera negra con un círculo naranja, con el número del piloto al que va dirigida, se muestra cuando un coche tiene un problema mecánico que se considera peligroso y, por tanto, debe volver a boxes lo antes posible.

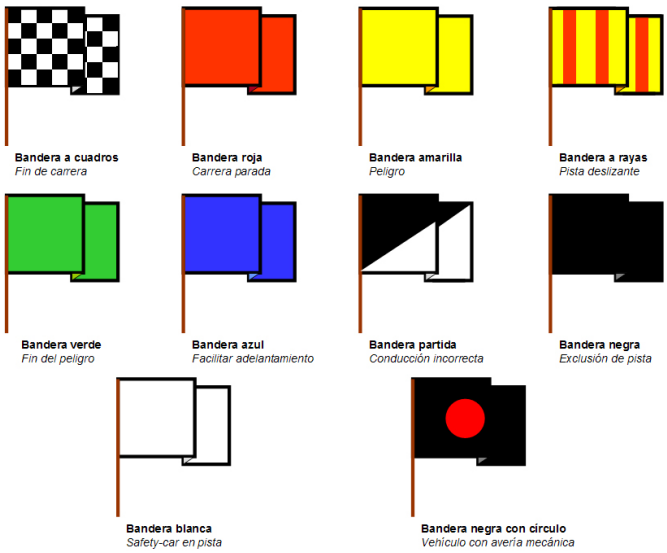


Figura 2.15: Todas las banderas de la Fórmula 1 y su descripción (extraído de [39])

Por último otro añadido referente a la seguridad es el safety car, tal como su nombre indica, es un auto de seguridad. Este ingresa cuando hay un accidente o por condiciones climáticas adversas. El vehículo, que está acompañado de la bandera amarilla, lidera la fila de los monoplazas de la F1 y marca la velocidad de carrera. Ninguno puede superarlo ni tampoco puede haber adelantamientos. El coche escogido como safety Car no es un monoplaza de Fórmula 1, es un coche deportivo para que sea claramente distinguible.

Esta regla se implementó por primera vez en el Gran Premio de Gran Bretaña en 1973, aunque se volvió una tradición desde 1993. Desde el accidente fatal de Jules Bianchi, la FIA sumó una nueva herramienta: el virtual safety car.

El virtual safety car (VSC), a diferencia del safety car, no pone un vehículo en pista, sino que neutraliza la carrera de manera virtual. Los monoplazas son limitados en cuanto a su velocidad, aunque las distancias entre pilotos se mantienen: todos marchan al mismo paso en la posición en la que estaban al momento del "ingreso" del VSC. Su aparición se da cuando el accidente no es tan grave y la pista se puede despejar rápidamente. [40]

Figura 2.16: Safety car (extraído de [41])

2.1.10. Impacto y relevancia de la Fórmula 1

La Fórmula 1 continúa siendo uno de los deportes más populares y de mayor relevancia en la actualidad, con un impacto significativo en diversas áreas. En términos de audiencias, la Fórmula 1 atrae a millones de espectadores en todo el mundo, convirtiéndose en uno de los eventos deportivos más seguidos a nivel global. Las carreras de Fórmula 1 generan una enorme atención mediática y son transmitidas en numerosos países, lo que contribuye a su alcance masivo y su influencia en la cultura popular. La revista Forbes señala que la Fórmula 1 generó ingresos anuales superiores a los 2 mil millones de dólares en el año 2022.

Las principales escuderías y pilotos cuentan con patrocinadores de renombre, como marcas de automóviles, empresas de tecnología, aerolíneas y marcas de lujo. Estas empresas buscan aprovechar la visibilidad global de la Fórmula 1 para promocionar sus productos y servicios.

Además de su impacto en términos monetarios y de audiencias, la Fórmula 1 ha impulsado el desarrollo de numerosas tecnologías que han tenido un impacto significativo en nuestra vida cotidiana. El enfoque constante en la innovación y la búsqueda de rendimiento ha llevado a importantes avances como la suspensión activa, el KERS (Sistema de recuperación de energía cinética), el uso de fibra de carbono y nuevos sistemas de frenado entre otros. Estas tecnologías han sido transferidas a la industria automotriz de consumo, lo que ha permitido mejorar la eficiencia, la seguridad y el rendimiento de los vehículos de carretera. [42]

2.1.11. Ciencia de datos en la Fórmula 1

La importancia de la ciencia de datos en la Fórmula 1 es indiscutible y ha revolucionado la forma en que los equipos compiten en la pista. En un deporte altamente tecnológico como este, la recopilación, análisis y aplicación inteligente de datos se ha convertido en una parte integral del éxito.

Los coches de Fórmula 1 están equipados con más de 300 sensores y sistemas de telemetría que capturan información en tiempo real sobre diversos aspectos del rendimiento del vehículo y producen 1.5 terabytes de datos por cada carrera. La información recogida incluye la velocidad, las revoluciones por minuto del motor, la temperatura de los neumáticos, la presión aerodinámica, el consumo de combustible y muchos otros parámetros relevantes. Además, se utilizan técnicas avanzadas de recopilación, como cámaras de alta velocidad y sistemas de GPS, para recopilar información adicional y proporcionar una visión más completa de lo que sucede en la pista.

Una vez recopilada, esta información se procesa y analiza utilizando herramientas de ciencia de datos. Se aplican algoritmos y modelos estadísticos para extraer información valiosa, identificar patrones, tendencias y correlaciones, y tomar decisiones basadas en evidencias estadísticas. Los ingenieros y estrategas de equipo utilizan estos análisis para comprender el rendimiento del coche en diferentes situaciones, evaluar el desgaste de los neumáticos, optimizar la configuración del vehículo y diseñar estrategias de carrera más efectivas gracias a los datos en tiempo real.

Además, la ciencia de datos también ha abierto nuevas oportunidades en la simulación y el modelado predictivo. Los equipos utilizan modelos matemáticos y simuladores avanzados para recrear situaciones de carrera, evaluar diferentes escenarios y predecir el rendimiento del coche en condiciones específicas.

Esto les permite probar estrategias antes de implementarlas en la pista y ajustar variables clave para maximizar el rendimiento. [43]

2.2. Retransmisiones en la Fórmula 1

2.2.1. Descripción de las retransmisiones de F1

Las retransmisiones de la Fórmula 1 son una parte esencial de la experiencia de los aficionados y juegan un papel fundamental en la difusión y popularidad del deporte en todo el mundo. Estas retransmisiones televisivas capturan cada detalle de la carrera, desde la emoción de la salida hasta las estrategias en boxes y los adelantamientos en pista.

Los equipos de producción trabajan duramente para dar una cobertura de alta calidad a los espectadores. Utilizan una combinación de cámaras en pista, helicópteros y cámaras aéreas para capturar los momentos más emocionantes de la competición.

Además, se emplean gráficos y animaciones para proporcionar información adicional sobre las posiciones de los pilotos, los tiempos por vuelta y otra información relevante.

Las retransmisiones de la Fórmula 1 también incluyen comentarios de expertos y narradores que ofrecen análisis en tiempo real, describen las estrategias de los equipos y proporcionan contexto histórico sobre los pilotos y los circuitos. Estos comentarios ayudan a los espectadores a comprender mejor las complejidades del deporte y a sumergirse en la emoción de la competición.

Además de las retransmisiones en vivo, se utilizan repeticiones instantáneas para mostrar momentos clave y acciones destacadas de la carrera. Esto permite a los espectadores revivir los adelantamientos, los incidentes y los momentos emocionantes que pudieron haberse perdido durante la transmisión en vivo.

2.2.2. Evolución de las retransmisiones de F1

Las retransmisiones de la Fórmula 1 han experimentado una notable evolución a lo largo de los años, adaptándose a los avances tecnológicos y a las demandas de los aficionados. Todo comenzó con las retransmisiones de radio, donde los aficionados podían escuchar narraciones detalladas de las carreras, imaginando la emoción y la acción en sus mentes.

Con la llegada de la televisión en blanco y negro, las retransmisiones adquirieron una nueva dimensión visual. Los espectadores podían ver los coches en acción y seguir las carreras de una manera más inmediata. La televisión en color amplió aún más la experiencia, brindando una mayor riqueza de detalles y permitiendo una mejor apreciación de los colores vibrantes de los coches y los circuitos.

A medida que la tecnología avanzaba, las retransmisiones de la Fórmula 1 se expandieron a nuevas plataformas. La llegada de la televisión por cable y satélite proporcionó una mayor cobertura y acceso a más canales especializados en deportes. Los espectadores podían elegir entre diferentes ángulos de cámara y disfrutar de repeticiones instantáneas, lo que les permitía capturar cada detalle de la acción en la pista.

En la era digital, las retransmisiones de la Fórmula 1 se han adaptado a las nuevas plataformas de transmisión en línea. Plataformas como YouTube y Twitch han brindado a los aficionados la oportunidad de seguir las carreras en vivo a través de transmisiones por internet. Esto ha abierto un mundo de posibilidades, con la interacción en tiempo real, la participación de los espectadores a través de chats y la posibilidad de ver las carreras en dispositivos móviles en cualquier momento y lugar.

Además, las redes sociales han jugado un papel importante en la evolución de las retransmisiones de la Fórmula 1. Los equipos, pilotos y medios de comunicación utilizan plataformas como Twitter, Facebook e Instagram para compartir contenido exclusivo, realizar transmisiones en vivo y mantener a los aficionados informados sobre las últimas noticias y actualizaciones del deporte.

2.2.3. Ciencia de datos en la retransmisiones de F1

La ciencia de datos ha transformado las retransmisiones de la Fórmula 1 al proporcionar análisis en tiempo real, información detallada y visualizaciones avanzadas. Gracias a estos avances, los espectadores pueden disfrutar de una experiencia más inmersiva, comprender mejor los aspectos técnicos y estratégicos de las carreras, y personalizar su experiencia de visualización según sus intereses individuales. La ciencia

de datos continúa desempeñando un papel crucial en la evolución de las retransmisiones de la Fórmula 1, mejorando constantemente la manera en que vivimos y disfrutamos este emocionante deporte.

Los científicos de datos de Fórmula 1 están entrenando modelos de aprendizaje profundo con más de 65 años de datos históricos de carreras. Con esta información, la Fórmula 1 puede extraer estadísticas críticas de rendimiento de carrera para hacer predicciones de carreras y dar a los aficionados una idea de las decisiones y estrategias adoptadas en una fracción de segundo por los equipos y los pilotos.

La Fórmula 1 puede capturar y procesar información de rendimiento clave para cada automóvil durante cada vuelta de los circuitos de Fórmula 1 con una precisión y velocidad inigualables. La Fórmula 1 transmite información de carrera en tiempo real y, mediante la implementación de aprendizaje automático, la Fórmula 1 puede identificar cómo se está desempeñando un conductor y si los conductores se han excedido o no al límite. Al compartir estos conocimientos con los aficionados a través de transmisiones de televisión y plataformas digitales, la Fórmula 1 está mejorando la experiencia, permitiéndoles profundizar en el funcionamiento interno de sus equipos y pilotos favoritos. [44]

2.3. Business Intelligence (BI)

De acuerdo con el economista Carlo Revelli, el Business Intelligence o Inteligencia de Negocio "es el proceso de recolección, tratamiento y difusión de la información que tiene un objetivo: la reducción de incertidumbre en el proceso de toma de decisiones estratégicas". [45]

Por otra parte, la empresa Gartner define BI como "un término que engloba las aplicaciones, infraestructuras y herramientas y las buenas prácticas que permiten acceder y analizar la información para mejorar y optimizar tanto las decisiones como el desempeño", los catedráticos de computación Nedim Dedić y Clare Stanier de Staffordshire University entienden por BI las "estrategias, procesos, aplicaciones, datos, productos, tecnologías y arquitecturas técnicas empleadas para sustentar la recolecta, análisis, presentación y difusión de la información del negocio". [46, 47]

Se puede ver que todas las definiciones coinciden en que el Business Intelligence es un conjunto de metodologías, estrategias, datos, herramientas e infraestructura que permiten tomar mejores decisiones. Por tanto, el objetivo final del BI no es analizar, estructurar o procesar datos sino mejorar la toma de decisiones en una organización.

2.3.1. Historia del BI

Se ha realizado un resumen de la historia del BI teniendo en cuenta los artículos de Cristina Lago y Paulo Limpio sobre los 150 años de historia del Business Intelligence. [48, 49]

BI en la era pre-digital

El primer registro escrito que tenemos del término "Business Intelligence" proviene de la obra Cyclopaedia of Commercial and Business Anecdotes de Richard Miller Devens de 1865. [50]

El autor estadounidense utilizó estas palabras para describir cómo el banquero Sir Henry Furnese se había posicionado por delante de su competencia al recopilar, analizar y utilizar la información a su disposición para ayudarlo a tomar decisiones comerciales sólidas y sensatas.

Década de 1950: inicio de la revolución digital

No fue hasta la década de 1950, durante los comienzos de la Revolución Digital, que el BI se convirtió en un proceso científico independiente adoptado por los empresarios para informar sus estrategias comerciales.

En ese año, el investigador de ciencias informáticas de IBM, Hans Peter Luhn, escribió un artículo seminal en el IBM Systems Journal titulado "Un sistema de inteligencia comercial ". Luhn describió un sistema para la "difusión selectiva" de documentos a "puntos de acción" basado en los "perfiles de interés" de los puntos de acción individuales.

Dicho sistema tenía la flexibilidad de identificar información conocida, encontrar a quién necesitaba conocerla y la posibilidad de distribuirla de manera eficiente.

Su trabajo tiene un significado notable hasta el día de hoy, ya que predijo varias tendencias de inteligencia empresarial que son de vanguardia en la actualidad, como la capacidad de los sistemas de información para aprender y predecir en función de los intereses de los usuarios. Hoy lo llamamos aprendizaje automático (machine learning). Luhn es reconocido popularmente como el padre de la inteligencia empresarial.

1960: primeras computadoras y bases de datos

La década de 1960 vio un aumento en el uso de las computadoras, las cuales eran máquinas que ocupaban pisos enteros y que debían ser operadas por trabajadores cualificados que comenzaron a generar grandes cantidades de datos. Si bien se pudieron recopilar cantidades inmensas de datos, todavía no se tenían las herramientas o la tecnología necesarias para hacer algo útil con ellos.

No obstante, el principal problema era la falta de un método centralizado que pudiera reunir todos los datos disponibles ya que, los datos en sí mismos no generan información.

En 1970 Edgar Codd mientras trabajaba en IBM, inventó el modelo relacional para la gestión de bases de datos, la base teórica para las bases de datos relacionales y los sistemas de gestión de bases de datos relacionales.

Década de 1970: los primeros proveedores de BI

Con la aparición de los primeros proveedores de BI (SAP, Siebel y JD Edwards, los dos últimos ahora parte de Oracle Corporation), las herramientas estuvieron disponibles para ayudar a acceder y organizar los datos de manera más efectiva.

El software de BI comenzó a dar estructura a la gran cantidad de información que habíamos recopilado durante las décadas anteriores.

Cerca de finales de los 70, Larry Ellison lanzó la primera versión comercial de la base de datos Oracle. Fue el primer verdadero gestor de bases de datos relacionales del mercado, reemplazando las ideas hasta entonces utilizadas de bases de datos jerárquicas y bases de datos en red por una estructura más robusta, que permitía búsquedas mucho más flexibles. Esta tecnología dictaría la historia y las tendencias de BI en las próximas décadas.

1980: nacimiento de los almacenes de datos

La década de 1980 vio la llegada de los almacenes de datos, que son sistemas utilizados para el análisis y la generación de informes.

Ralph Kimball y Bill Inmon propusieron dos estrategias diferentes pero similares al problema de tener todos los datos del negocio en el mismo lugar para poder analizarlos. Estos eran almacenes de datos (Data Warehouse).

Los almacenes de datos se utilizan como repositorios centrales de datos integrados de una o más fuentes diferentes. Almacenan información actual e histórica en un solo lugar que se utiliza para crear informes analíticos para departamentos separados en una empresa. Ahora se consideran un componente central de BI.

Décadas de 1990 y 2000: Business Intelligence 1.0 y Business Intelligence 2.0

En los años 90, los costos del almacenamiento de datos disminuyeron a medida que más competidores ingresaron al mercado y más profesionales de TI se familiarizaron con la tecnología. En la historia y evolución de la inteligencia empresarial, este fue el período de "Business Intelligence 1.0".

Para reducir el esfuerzo de realizar consultas, se desarrollaron algunas herramientas nuevas para acelerar el proceso de diferentes consultas:

- ETL (extraer, transformar y cargar): Es un conjunto de herramientas, similar a un lenguaje de programación, que facilitaba el diseño del flujo de datos dentro de un almacén de datos.

- OLAP (procesamiento analítico en línea): Ayudó a crear diferentes opciones de visualización para los datos consultados, lo que permitió a los analistas extraer mejores conclusiones de la información disponible.

Hasta el día de hoy, las herramientas ETL y OLAP siguen siendo una parte crucial de las soluciones de inteligencia empresarial.

La década de 2000 (conocida como Business Intelligence 2.0) agregó más velocidad al desarrollo de BI y vio una concentración de BI en manos de IBM, Microsoft, SAP y Oracle.

El análisis predictivo proporcionó un nuevo método de uso de datos, algoritmos y aprendizaje automático para pronosticar cambios futuros en el negocio.

Las tecnologías en la nube y el software basado en Internet pasan a primer plano a medida que las fuentes en tiempo real y la visualización mejorada cambian la forma en que se ven los datos.

2010 – actualidad

Actualmente estamos en la etapa 3.0 de BI.

BI se ha convertido en una herramienta estándar para todas las medianas o grandes empresas, desde finanzas y banca hasta TI y comunicaciones.

Las herramientas de BI actuales funcionan en múltiples dispositivos y utilizan análisis visuales para aplicar el razonamiento analítico a los datos a través de interfaces visuales interactivas.

Los esfuerzos se centran en hacer que las herramientas y aplicaciones de BI sean lo más intuitivas posible y en adquirir las habilidades necesarias para aplicarlas con éxito.

En la Figura 2.17 se observa la evolución temporal del BI desde su primera referencia escrita por Richard Miller Devens en 1865 del término Business Intelligence hasta la actualidad. [51]

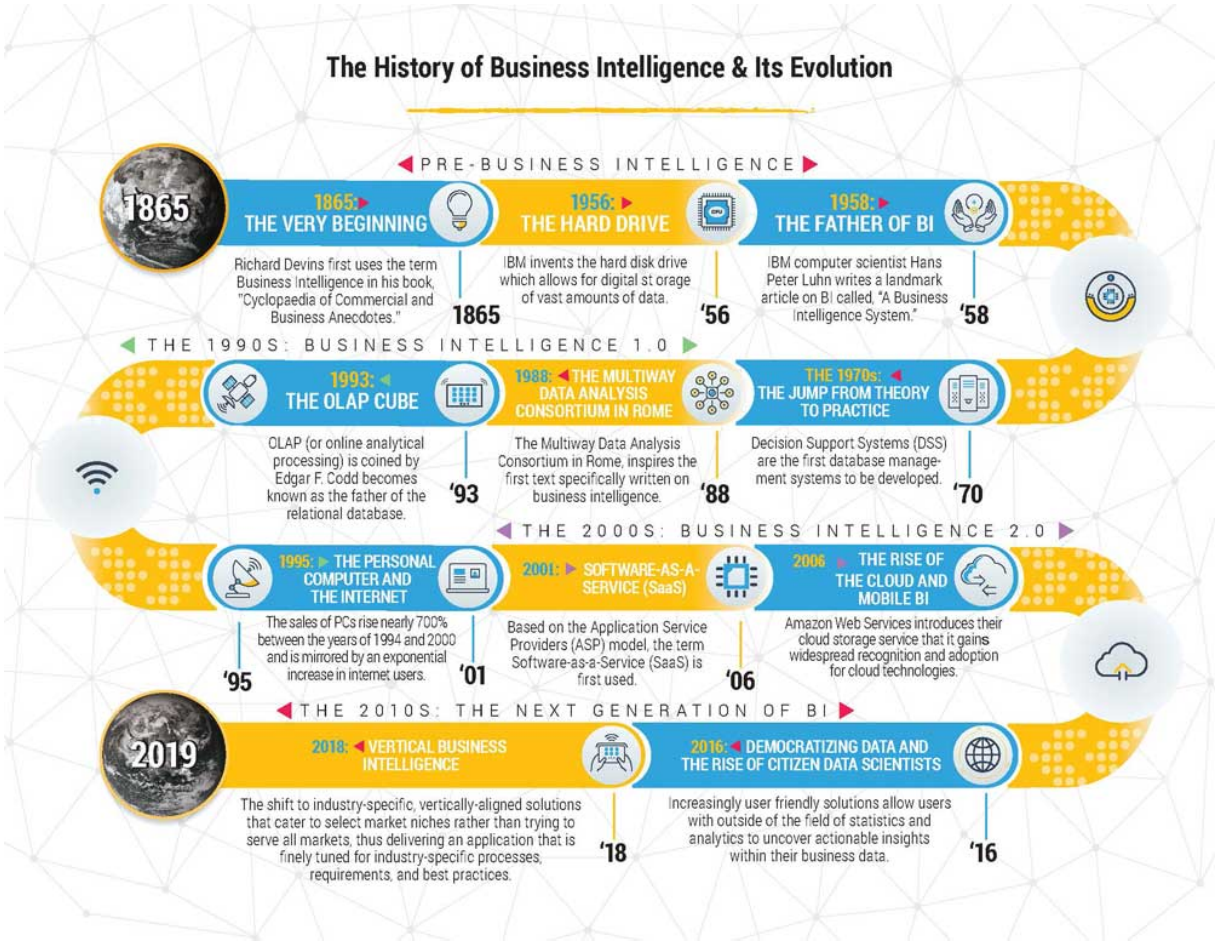


Figura 2.17: Evolución temporal del Business Intelligence (extraído de [51])

2.3.2. Datos, información, conocimiento

Los conceptos que se muestran a continuación se basan en las definiciones de Thomas Davenport y Laurence Prusak en su libro "Working Knowledge: How Organizations Manage what They Know" [52] y descritas en el artículo de la empresa Sinnexus [53].

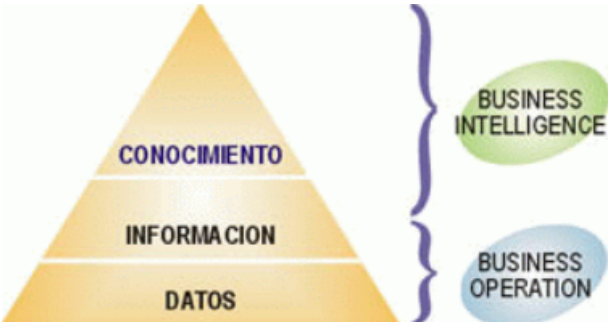


Figura 2.18: Datos, información y conocimiento (extraído de [53])

Datos

Los datos son la mínima unidad semántica y se corresponden con elementos primarios de información que por sí solos son irrelevantes como apoyo a la toma de decisiones. También se pueden ver como un conjunto discreto de valores, que no dicen nada sobre el por qué de las cosas y no son orientativos para la acción.

Un número telefónico o un nombre de una persona, por ejemplo, son datos que, sin un propósito, una utilidad o un contexto no sirven como base para apoyar la toma de una decisión.

Los datos pueden provenir de fuentes externas o internas a la organización, pudiendo ser de carácter objetivo o subjetivo, o de tipo cualitativo o cuantitativo, etc.

Información

La información se puede definir como un conjunto de datos procesados y que tienen un significado (relevancia, propósito y contexto), y que por lo tanto son de utilidad para quién debe tomar decisiones, al disminuir su incertidumbre. Los datos se pueden transformar en información añadiéndoles valor:

- Contextualizando: se sabe en qué contexto y para qué propósito se generaron.
- Categorizando: se conocen las unidades de medida que ayudan a interpretarlos.
- Calculando: los datos pueden haber sido procesados matemática o estadísticamente.
- Corrigiendo: se han eliminado errores e inconsistencias de los datos.
- Condensando: los datos se han podido resumir de forma más concisa (agregación).

Por lo tanto, la información es la comunicación de conocimientos.
Información = Datos + Contexto (añadir valor) + Utilidad (disminuir la incertidumbre).

Conocimiento

El conocimiento es una mezcla de experiencia, valores e información que sirve como marco para la incorporación de nuevas experiencias e información. En las organizaciones, con frecuencia no sólo se encuentra dentro de documentos o almacenes de datos, sino que también está en rutinas organizativas, procesos, prácticas, y normas.

El conocimiento se deriva de la información, así como la información se deriva de los datos. Para que la información se convierta en conocimiento es necesario realizar acciones como:

- Comparación con otros elementos.
- Predicción de consecuencias.
- Búsqueda de conexiones.
- Conversación con otros portadores de conocimiento.

2.3.3. Arquitectura y elementos del BI

En la Figura 2.19 se presenta la arquitectura común genérica que acompaña un modelo de inteligencia de negocios. [54]

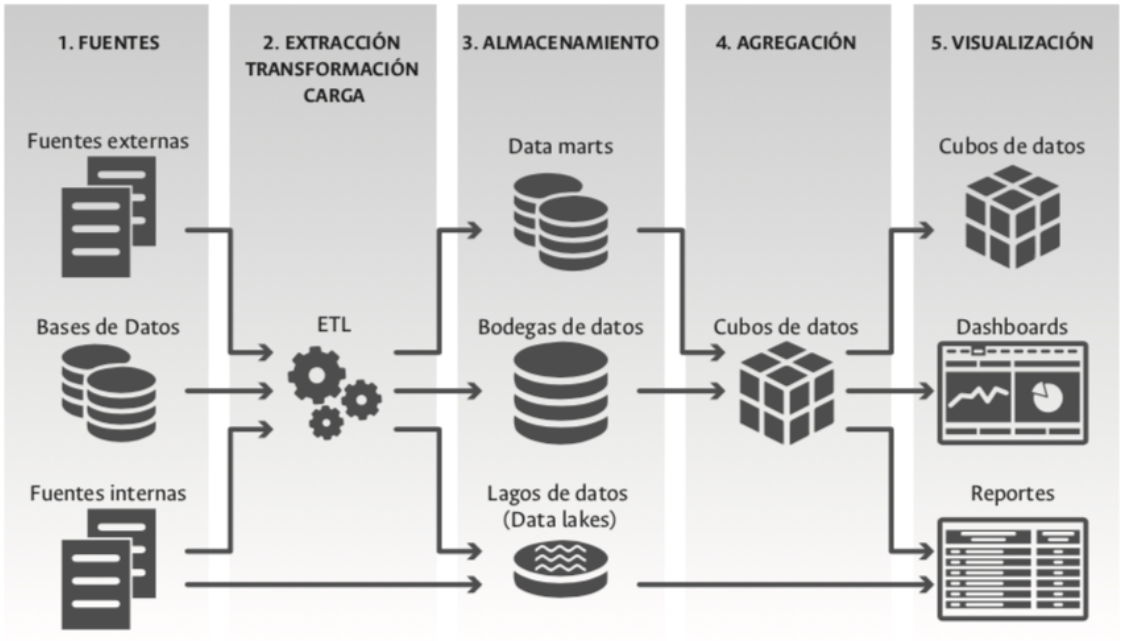


Figura 2.19: Arquitectura tecnológica común de un modelo de inteligencia de negocios (extraído de [54])

Como se observa en la ilustración existen cinco elementos comunes en la arquitectura de un proyecto de BI:

1. **Fuentes de datos**: Los datos que se utilizan pueden ser generados por la propia organización (fuentes internas) o de origen externo (fuentes externas) pero la distinción más importante que se debe hacer es entre datos estructurados y no estructurados: [55]

    - Datos estructurados: Se encuentran organizados mediante una serie de filas y columnas bien definidas. Son los que se usan de manera habitual en la mayor parte de las bases de datos relacionales (RDBMS). El lenguaje de programación mediante el cual se gestionan es el Structured Query Language o SQL.
    - Datos semiestructurados: Son aquellos con un nivel medio de estructuración y rigidez organizativa. Se encuentran a medio camino entre los estructurados y los no estructurados. Algunas fuentes de datos semiestructurados son los archivos XML.
    - Datos no estructurados: No están estructurados a través de modelos o esquemas de datos fijos y predefinidos. Pueden ser textuales o no y generados por humanos o máquinas. Ejemplos de datos no estructurados pueden ser archivos de audio, imágenes o presentaciones en Power Point.

2. **Extracción, transformación y carga (ETL)**:

- Extracción (E) donde se explica el proceso de ingesta de los datos desde los orígenes.
- Transformación (T) en la que se aplica una serie de reglas para garantizar la calidad de los datos.
- Carga (L) de los datos en la que los datos ya transformados pasan al entorno de explotación para poder realizar análisis posteriores.

3. **Almacenamiento**: Se pueden almacenar los datos transformados en la ETL en diferentes lugares como pueden ser Data marts, Data Warehouse y Data lakes. [56]

   - Data Mart: Es un subconjunto de los datos guardados en un almacén de datos, destinado a satisfacer las necesidades de un segmento de negocio en particular. Este área de clasificación de datos enfoca la información, logrando un ajuste máximo al propósito de los usuarios de la unidad de negocio.
   - Almacén de datos (Data Warehouse): Es un repositorio unificado que permite integrar todos los datos de la organización. Su estructura facilita la extracción de datos, su procesamiento y la posterior puesta a disposición del usuario. Un Data Warehouse debe presentar las siguientes características:
     - Estabilidad: Los datos y el acceso a ellos no puede ser volátil. Deben estar accesibles para los miembros de la organización.
     - Coherencia: Los datos deben estar organizados. Las relaciones entre ellos han de estar correctamente definidas.
     - Fiabilidad: Los datos almacenados en el Data Warehouse deben ser fiables ya que son la base de la información extraída.
   - Data Lake: Es un repositorio centralizado diseñado para almacenar, procesar y proteger grandes cantidades de datos estructurados, semiestructurados o no estructurados. Puede almacenar datos en su formato nativo y procesar cualquier variedad de datos, ignorando los límites de tamaño.

4. **Agregación**: Un cubo de datos (data cube) generalmente se usa para interpretar fácilmente los datos. Es especialmente útil cuando se representan datos junto con dimensiones como ciertas medidas de los requisitos comerciales. Cada dimensión de un cubo representa cierta característica de la base de datos, por ejemplo, ventas diarias, mensuales o anuales. Los datos incluidos dentro de un cubo de datos permiten analizar casi todas las cifras de prácticamente todos los clientes, agentes de ventas, productos y mucho más. Crear cubos tiene muchas ventajas a la hora de realizar análisis, la principal es la eficiencia ya que tiene menor coste de almacenamiento y menor tiempo para generar los informes.

5. **Visualización**: La visualización de datos es la práctica de representar información gráficamente para comunicar y contextualizar datos. Destaca cambios, patrones y tendencias importantes utilizando un formato visual, como tablas, gráficos, mapas y diagramas, para ayudar a que los datos sean más fáciles de entender. Se realiza mediante Reportes o mediante Dashboards.

   Los reportes son una estrategia empleada para suministrar tecnológicamente información tabular que contiene cifras de interés institucional, de alta utilidad para aquellas organizaciones o entidades que requieren suministrar informes o microdatos de manera periódica a actores internos o externos.

   Los cuadros de mando o dashboards conforman la apuesta gráfica para el seguimiento y la presentación de las datos, principalmente de naturaleza descriptiva, de una organización y son, de lejos, el mecanismo tecnológico más empleado para gestionar la información cuantitativa institucional en una organización guiada por una apuesta de inteligencia de negocios. Aunque los dashboards pueden ser empleados en una entidad para presentar información de manera gráfica y tabular proveniente de datos de cualquier naturaleza y complejidad, desde sus orígenes han sido ampliamente empleados para representar información gráfica de tipo descriptivo y derivada de fuentes estructuradas o, a lo sumo, semiestructuradas. Los gráficos más comunes en los dashboards son diagramas de tarta,

diagramas de barras, histogramas, diagramas de caja (box plots), mapas, gráficos de líneas, gráficos de dispersión, barras de progreso, velocímetros...

2.3.4. Herramientas de BI

Los diferentes tipos de herramientas coinciden y vienen dados por las diferentes etapas del BI. Se puede sintetizar dicho proceso mediante 3 tipos de herramientas de BI: [57]

- **Herramientas para la gestión de datos** (data management tools): Engloba los pasos 1 y 2. Permiten desde la depuración y estandarización de datos de procedencia diversa hasta su extracción, transformación y traslado a un determinado sistema de almacenamiento.
- **Aplicaciones para descubrir nuevos datos** (data discovery applications): Engloba el paso 3. Permiten recopilar y evaluar nueva información (data mining o minería de datos), y aplicar técnicas de análisis predictivo para realizar proyecciones de futuro.
- **Herramientas de reporting**: Engloba los pasos 4 y 5.Una vez recopilada y tratada toda esa información preexistente o nueva, ayudan a las organizaciones a visualizarla de manera gráfica e intuitiva. También sirven para integrarla en cuadros de mando que midan si se cumplen o no determinados KPIs, o pueden incluso generar todo tipo de informes de reporting.

Las mayores empresas tecnológicas como Microsoft Google o Amazon han creado sus propias herramientas de BI siendo estas las más importantes: [58, 59, 60]

- **Herramientas de BI de Microsoft**: La herramienta Power BI de Microsoft ofrece capacidades de visualización, análisis e informes para datos locales o de origen en la nube en su plataforma Power BI. También se incluye preparación de datos, paneles interactivos y descubrimiento basado en imágenes. La interfaz de usuario de Power BI es similar a Excel y permite una curva de aprendizaje corta. También proporciona videos y documentos de aprendizaje detallados para seguir aprendiendo.

  Ventajas:

  - No es necesario saber SQL para mezclar y limpiar datos de sus diferentes sistemas.
  - Puede conectar Power BI a muchas fuentes de datos y también lee datos de Microsoft Excel y archivos de texto como XML y JSON.
  - Permite visualizar gráficos creados mediante scripts de R.

  Contras:

  - Poca capacidad de manejo de datos para las versiones gratuitas.
- **Herramienta de BI de Salesforce**: Tableau, una herramienta de BI de Salesforce, proporciona software para inteligencia comercial que se puede ejecutar en local, en la nube o alojado por Tableau. Tableau utiliza el procesamiento de lenguaje natural para la comunicación, aprovecha la inteligencia artificial, presenta análisis visuales en vivo con botones de arrastrar y soltar, admite una amplia variedad de fuentes de datos, permite compartir paneles fácilmente y puede conectarse a cualquier base de datos.

  Ventajas:

  - Admite múltiples fuentes de datos como base de datos relacionales, bases de datos NoSQL como Mongo DB, diferentes fuentes de datos de archivos (Excel,csv, txt, Json...).s
  - Impresionante visualización de datos. Las capacidades incomparables de visualización de información se encuentran en la parte superior de la lista de beneficios del software Tableau.
  - El software detecta automáticamente qué dispositivo está utilizando un usuario y ajusta el informe para una visualización óptima.

  Contras:

- La preparación de datos requiere de una herramienta independiente (Tableau Prep), lo que provoca la necesidad de cambio de software durante el trabajo.
- Escasa información y asistencia al usuario para análisis avanzado.
- Además, los formatos de exportación e impresión son limitados, lo que es inconveniente para compartir resultados.

- **Herramienta de BI de Google**: Looker, una herramienta de BI de Google, es una plataforma de inteligencia empresarial que ofrece visualizaciones en tiempo real, capacidades de modelado, paneles interactivos, herramientas de generación de informes de inteligencia empresarial en tiempo real, colaboración, descubrimiento de datos y puede funcionar encima de cualquier base de datos analítica.

  Ventajas:

  - Se ejecuta en la nube, por tanto se puede acceder a los informes y realizar cambios en cualquier lugar.

  Contras:

  - Las visualizaciones de datos resultan deficientes.
  - Es necesario conocer el lenguaje SQL para poder utilizarlo.

- **Herramienta de BI de Qlik**: QlikView es una herramienta de inteligencia empresarial de software que ayuda a crear informes y a obtener conocimientos empresariales de forma rápida. Tiene una interfaz intuitiva que se puede usar de manera intuitiva y tiene una función de búsqueda inteligente. No requiere la construcción de cubos y es más adecuado para el análisis ad hoc que para el análisis de rutina.

  Ventajas:

  - El motor de procesamiento de datos es increíblemente rápido y puede manejar más datos de múltiples fuentes.
  - Integración de R y Python disponible para preparación de datos y análisis visual.
  - Obtiene poderosas capacidades de descubrimiento de datos (data discovery) con la capacidad única de mostrar datos ausentes o incorrectos en las dimensiones.

  Contras:

  - Carencias en la visualización de datos en comparación con sus competidores.
  - No es compatible con gráficos de mapas.
  - No es necesaria otra herramienta de preparación de datos, pero suele ser útil porque no es el punto fuerte de Qlik.

- **Herramientas de BI de Oracle**: Oracle Business Intelligence es un conjunto integral de tecnología y aplicaciones diseñado para proporcionar soluciones integrales que optimizan el rendimiento comercial y permiten una toma de decisiones móvil más rápida e informada. Fundada en 1977, Oracle ofrece una amplia gama de funciones de gestión de datos, que incluyen aprendizaje automático, inteligencia artificial y consultas integradas, informes y análisis móvil.

  Ventajas:

  - Tableros Oracle BI. Los tableros personalizados ofrecen acceso interactivo a la información adaptada a los roles e identidades individuales, lo que permite a los usuarios trabajar con informes, cuadros, gráficos y tablas dinámicas en vivo en una arquitectura web pura.

  Contras:

- Velocidad lenta en algunas ocasiones especialmente al generar informes. Algunas funciones, como agregar información a los modelos de datos, pueden ser complicadas y llevar mucho tiempo.

2.3.5. Etapas de un proyecto de BI

Un ciclo de vida típico de un proyecto de inteligencia de negocios tiene cinco pasos:

1. **Identificación y Análisis:** El proceso de inteligencia empresarial (BI) se inicia con la identificación de las necesidades de los usuarios y la formulación de preguntas clave para definir los objetivos del proyecto y definición de métricas o KPIs claves para el proyecto. Los analistas de negocios desempeñan un papel fundamental en la recolección de información y en la búsqueda de soluciones. Antes de implementar un sistema de BI, es crucial realizar un análisis exhaustivo de las necesidades específicas del negocio, involucrando a diferentes partes interesadas y obteniendo diferentes perspectivas. Esto garantizará la construcción de un sistema de Business Intelligence que cumpla con los requisitos identificados en esta primera etapa y que ayude a mejorar la eficiencia y la toma de decisiones en la organización. [61, 62, 63]

2. **Proceso de Extracción, Transformación y carga (ETL):**Es el proceso que permite a las organizaciones extraer datos desde múltiples fuentes, transformarlos y limpiarlos, y cargarlos en el destino . Consta de las siguientes etapas:

   - Extracción (E) donde se explica el proceso de ingesta de los datos desde los orígenes.
   - Transformación (T) en la que se aplica una serie de reglas para garantizar la calidad de los datos.
   - Carga (L) en la que los datos ya transformados pasan al entorno de explotación para poder realizar análisis posteriores.

3. **Diseño e implementación del almacén de datos :** En la etapa de diseño del sistema de almacenamiento, se debe tener en cuenta la necesidad de actualizaciones y evolución. El equipo de diseño no solo debe desarrollar una herramienta que satisfaga los requisitos actuales, sino también dejar espacio para futuras actualizaciones sin necesidad de crear una solución completamente nueva desde cero. Además, en esta etapa se procede a la confección de modelos de datos, que pueden ser en forma de estrella o de copo de nieve. El modelo estrella consta de una tabla central de "Hechos" y varias "dimensiones". Por otro lado, el modelo de copo de nieve es una variante del modelo estrella en el cual una tabla de hechos está conectada a múltiples tablas de dimensiones, que a su vez pueden estar conectadas a otras tablas de dimensiones. [61]

4. **Desarrollo de las visualizaciones:** En esta etapa se elaboran y presentan reportes, cuadros de mando (dashboards) y otros elementos de visualización para facilitar la toma de decisiones. Se utilizan diversos tipos de gráficos para presentar de manera clara, visual y esquemática para facilitar el entendimiento y el trabajo del equipo.

   Es fundamental que el proyecto de BI se comunique de manera efectiva a los usuarios y al resto del equipo de trabajo. Aunque el proyecto pueda ser excelente en sí mismo, si no se sabe comunicar correctamente, perderá su eficacia. Por lo tanto, la comunicación clara y efectiva a través de los informes y elementos visuales es esencial para asegurar el éxito del proyecto de BI.

5. **Toma de decisión:** La última etapa del proyecto de Business Intelligence es el causante del comienzo del proyecto, ya que un proyecto de BI se realiza para facilitar mediante la visualización de datos la toma de decisión de una organización. A partir de los informes y cuadros de mando realizados los líderes de la organización toman las decisiones estratégicas y operativas respaldadas por datos. [64]

Para finalizar este apartado se ha realizado un esquema con cada una de las cinco fases de un proyecto de BI, (ver Figura 2.20).

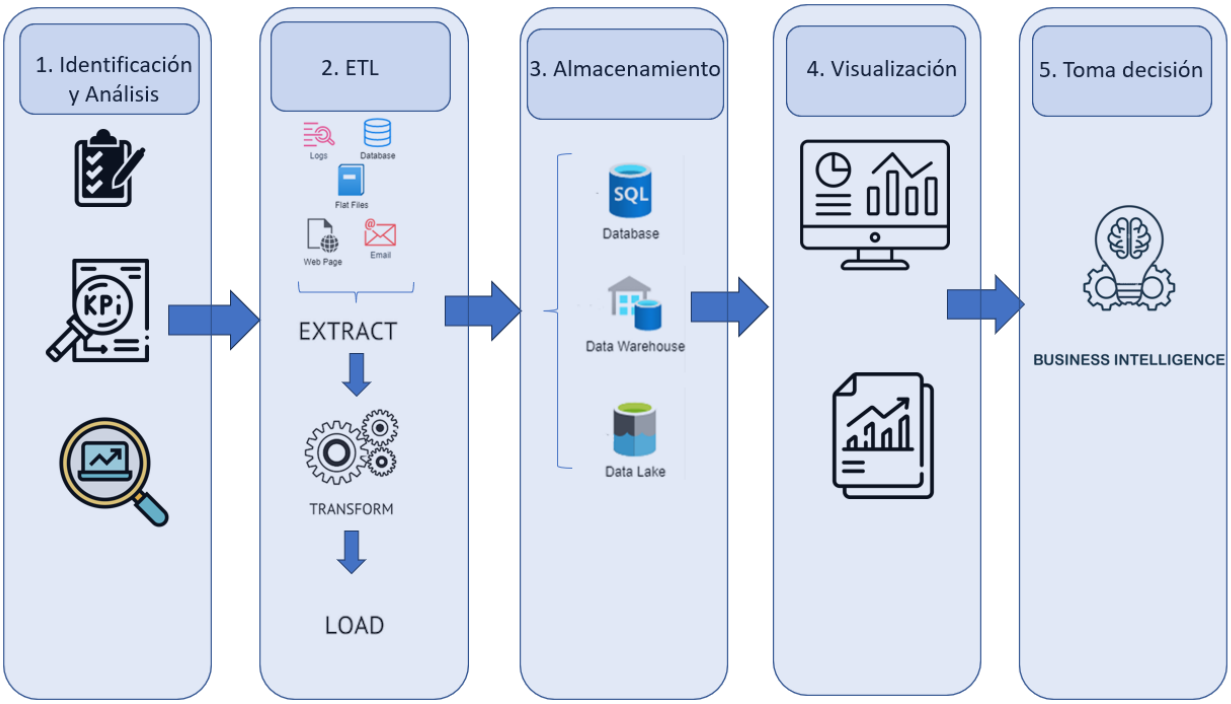


Figura 2.20: Etapas de un proyecto de Business Intelligence (elaboración propia)

Capítulo 3

Planificación

El presente Trabajo de Fin de Grado constituye 12 *European Credit Transfer System* (ECTS) de la formación académica de la titulación de Grado en Ingeniería Informática de la Universidad de Valladolid.

Teniendo en cuenta el actual marco de traducción a horas de la Universidad de Valladolid, implicaría una dedicación estimada de 300 horas. El inicio data del 16 de enero de 2023 y su fecha estimada de entrega es el 9 de junio del mismo año. Por lo que, tendríamos un marco temporal de 21 semanas con una carga promedio de trabajo de 15 horas semanales aproximadamente.

Antes de nada lo primero es definir la metodología empleada por el presente TFG, después se realiza una planificación temporal del proyecto definiendo las actividades con sus respectivos tiempos, como una planificación de costos y recursos. Por último también se realiza una estimación de riesgos.

3.1. Metodología empleada

La metodología empleada en este TFG es una metodología híbrida entre la metodología en cascada (waterfall) y la metodología Agile muy utilizada en los proyectos de BI llamada metodología Blended. [65]

En el comienzo del proyecto de BI, que corresponde con las etapas de planificación del proyecto e identificación de requerimientos en los que se define el alcance del proyecto, se empieza con una metodología en cascada, este proceso es completamente lineal, es decir, no se avanza a otra fase hasta que se completa la fase anterior esto asegura que los requisitos estén bien entendidos, documentados y acordados.

Pero la poca flexibilidad al cambio de esta metodología provoca que después de realizar la etapa de análisis y diseño con la metodología en cascada se realice una metodología Agile mediante sprints para las etapas de desarrollo de la aplicación de BI ya que se ha ido adaptando el trabajo adecuándolo a medida de que objetivos se iban alcanzando en las reuniones realizadas.

Por tanto, este enfoque híbrido reduce los riesgos, al describir claramente los requisitos gracias a la metodología en cascada y gracias a la metodología Agile que proporciona un cambio rápido aclarando interpretaciones confusas de la interfaz realizando entregas iterativas.

3.2. Actividades a realizar

El proyecto se divide en cuatro fases dentro de las cuales están formadas por diferentes actividades:

1. Planificación del proyecto de BI
    1. Identificación de requisitos
    2. Definición de métricas y KPIs
    3. Definición de tablas de hechos y dimensiones
    4. Diseño de mockups
2. Extracción, transformación, carga de los datos (ETL) y creación de métricas
    1. Acceso al origen de los datos

2. Transformación de los datos
3. Creación de métricas

3. Desarrollo de los informes

1. Formación de la herramienta a utilizar
2. Creación de los cuadros de mando
3. Pruebas

4. Documentación

1. Desarrollo de la memoria

Después de identificar las actividades de las que se compone el proyecto se realiza un diagrama de Gantt para organizar las actividades en el calendario, comenzando el día 16 de enero de 2023 y finalizando como fecha estimada el 9 de junio del mismo año, (ver Figura 3.1).

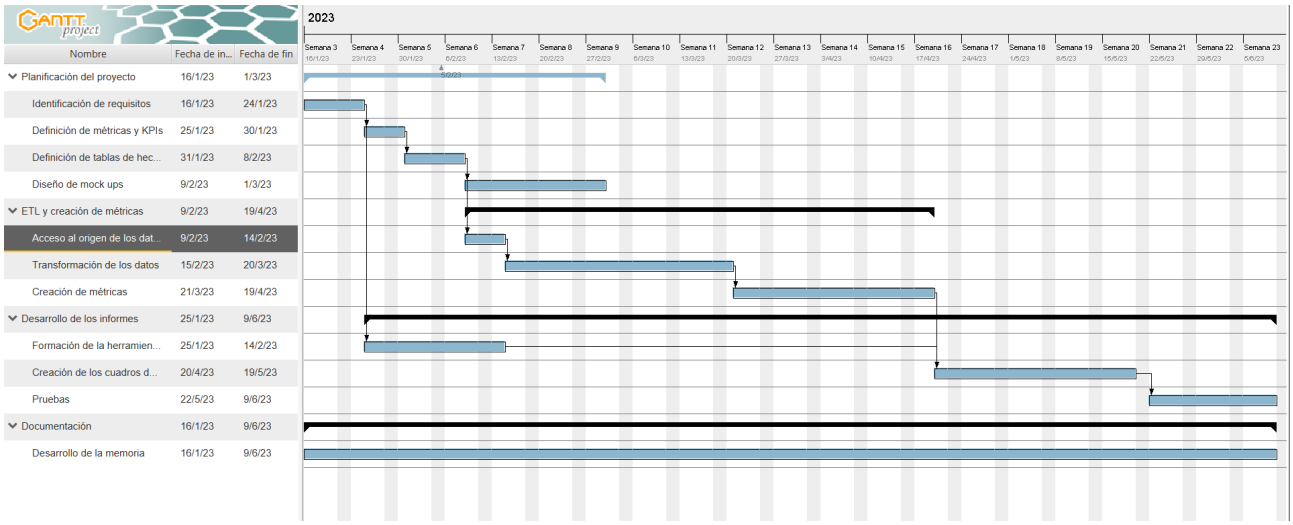


Figura 3.1: Diagrama de Gantt de la planificación

3.3. Recursos a emplear

Tras la definición de las actividades se debe definir los recursos necesarios para el desarrollo del proyecto, de los que se diferencian tres tipos: recursos humanos, recursos materiales y recursos tecnológicos.

3.3.1. Recursos humanos

Atendiendo a recursos humanos la unidad de medición en recursos humanos para un proyecto de Business Intelligence es de horas de trabajo.

Por ello con las actividades presentadas y el tiempo disponible, se realiza una estimación del esfuerzo en horas de trabajo para desarrollar la planificación del mismo.

Fase	Actividad	Tiempo estimado en horas	Estimación total
Planificación del proyecto de BI	Identificación de requisitos	10	50
	Definición de métricas y KPIs	5	
	Definición de tablas de hechos y dimensiones	10	
	Diseño de mockups	25	
Extracción, transformación, carga de los datos (ETL) y creación de métricas	Acceso al origen de los datos	5	90
	Transformación de los datos	45	
	Creación de métricas	40	
Desarrollo de los informes	Formación de la herramienta a utilizar	25	90
	Creación de los cuadros de mando	40	
	Pruebas	25	
Documentación	Desarrollo de la memoria	70	70
Total			300

Tabla 3.1: Estimación de recursos humanos

3.3.2. Recursos materiales

Para el desarrollo del proyecto atendiendo a los recursos materiales se requiere:

- Oficina (coworking).
- Ordenador (renting).
- Monitor (renting).

3.3.3. Recursos tecnológicos

Para el desarrollo del proyecto se requiere del siguiente software:

- Licencia de Windows 10 Profesional.
- Licencia Power BI Pro para poder publicar la aplicación final.
- Licencia del paquete Office.

3.3.4. Costos del proyecto

Por último se van a presentar los costos del proyectos sumando todos los tipos de recursos requeridos. El sueldo por hora de los recursos humanos se ha obtenido mediante la web Talent [66].

Para ello, se ha escogido el sueldo en 2023 en España de un profesional especializado en cada fase, correspondiendo con un Business Analyst para la fase de planificación del proyecto de BI y documentación, un Data Scientist para la etapa de ETL y creación de métricas y por último un Business Intelligence Analyst para la etapa de desarrollo de los informes. Obteniendo un coste total del recurso humano de 5954.4 €, (ver Tabla 3.2).

Por tanto obtenido el costo total de los recursos humanos se presenta el presupuesto total para el desarrollo del proyecto que se estima en 6355.48 €, (ver Tabla 3.3).

Rol	Tiempo (h)	Coste (€/h)	Coste total (€)
Business Analyst	120	19.23	2307.6
Data Scientist	90	23.08	2077.2
Business Intelligence Analyst	90	17.44	1569.6
Total	300		5954.4

Tabla 3.2: Desglose de los recursos humanos

Recurso	Unidades equivalentes	€Unidad	€Total
Recurso humano	300 horas	Ver Tabla 3.2	5954.4
Oficina coworking	1.875[1] meses	125€/mes [67]	234.38
Ordenador renting	1.875[1] meses	50 €/mes	93.75
Monitor renting	1.875[1] meses	24 €/mes	45
Licencia Windows 10	1	260 €	8.13[2]
Licencia Power BI Pro	1.875[1] meses	9.40 €/mes	17.63
Licencia Office	1	70 €	2.19[2]
Total			6355.48

Tabla 3.3: Presupuesto total del proyecto

[1] Si fuera un proyecto real en el que se trabaje 40 horas semanales (160 horas mensuales) habría durado el proyecto $\frac{300}{160} = 1{,}875$ meses en vez de 6 como es lo que está planeado. Por ese motivo no se utiliza el ordenador o el monitor el 100 % del tiempo y hay que tenerlo en cuenta en los costes.

[2] Al ser un precio fijo hay que tener en cuenta los años de amortización (el software se estima que se amortiza en 5 años = 60 meses). Como se van a utilizar 1.875 meses el precio final será igual a $\frac{Preciofijo \cdot 1{,}875}{60}$.

3.4. Riesgos

3.4.1. Identificación de riesgos

Como en todos los proyectos, se debe llevar a cabo un análisis de los riesgos asociados al desarrollo de este. En la Tabla 3.4 podemos encontrar los riesgos más comunes en el mundo del BI extraídos de varios estudios sobre los errores cometidos habitualmente en el desarrollo de proyectos BI.[68, 69].

A partir de la probabilidad e impacto de cada riesgo, se ha creado una matriz de Probabilidad-Impacto, para visualizar rápidamente qué riesgos son los más peligrosos para el proyecto.

Gracias a la matriz de Probabilidad-Impacto, (ver Figura 3.2) se observa que el riesgo más peligroso para el proyecto es el riesgo 04 (Probabilidad = Probable e Importancia = Peligroso) ya que es el que supera la línea de tolerancia. También hay que tener precaución con los riesgos 01, 02, 03 y 07 que se encuentran cerca de la línea de tolerancia.

Id	Descripción	Probabilidad	Impacto
01	Por una planificación poco realista, las tareas de documentación duran más de lo esperado, retrasando el proyecto	Probable	Moderado
02	Un mal prototipo y análisis de requisitos puede hacer que se desarrolle un cuadro de mando defectuoso y que el usuario acabe descontento	Posible	Peligroso
03	Los requisitos del proyecto cambian durante el transcurso del proyecto, lo que puede requerir ajustes en el diseño y desarrollo del sistema de Business Intelligence.	Ocasional	Peligroso
04	Los datos pueden contener errores, inconsistencias o outliers, lo que afecta a las métricas y a las visualizaciones de los cuadros de mando	Probable	Peligroso
05	Las herramientas utilizadas en el proyecto de BI pueden cambiar, lo que requiere adaptación, una posible formación y posibles retrasos en la implementación.	Improbable	Moderado
06	El conocimiento limitado del dominio de la Fórmula 1 al que pertenece el proyecto de BI, puede dificultar la comprensión y satisfacción de las necesidades de los usuarios finales.	Improbable	Peligroso
07	Tener experiencia insuficiente en tecnologías de BI, que puede llevar a retrasos y dificultades en la implementación del proyecto.	Ocasional	Moderado

Tabla 3.4: Identificación de riesgos

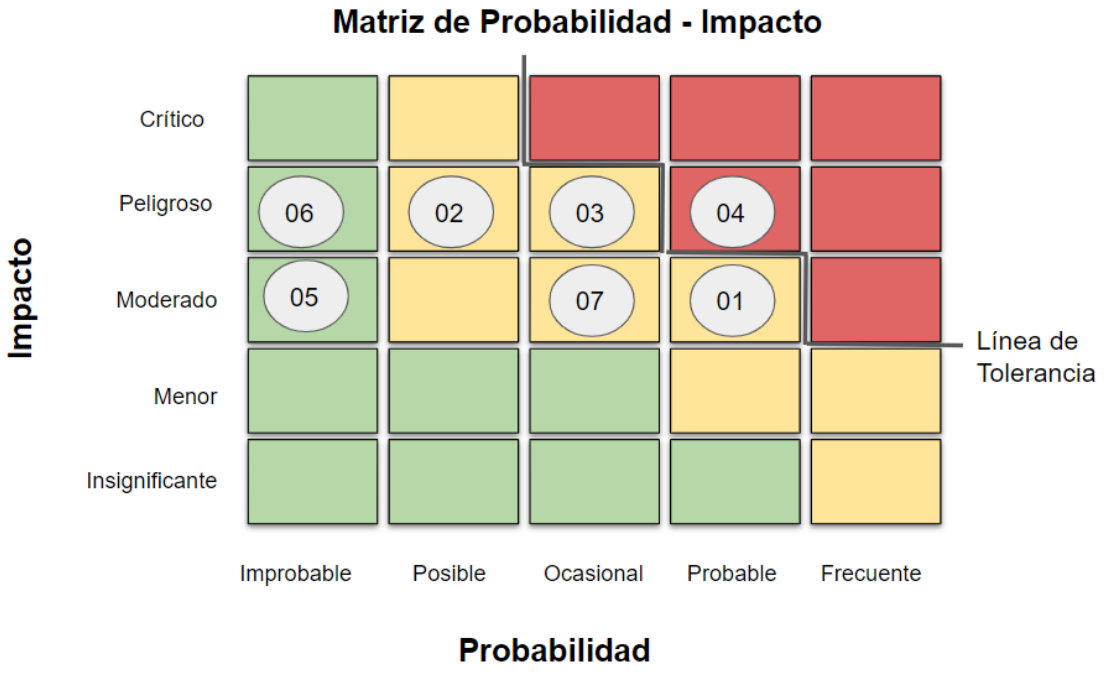


Figura 3.2: Matriz de Probabilidad-Impacto de riesgos con los IDs de riesgos (elaboración propia)

3.4.2. Respuesta a los riesgos

- **ID 01**: Definición de las etapas contando con un tiempo extra de cara a poder mitigar los imprevistos que puedan surgir.
- **ID 02**: Revisiones continuas, en primer lugar de los mockups teniendo en cuenta toda la impresiones del cliente, después estás revisiones deben hacerse con las versiones iniciales de la aplicación.
- **ID 03**: Llevar a cabo de forma muy cuidadosa la toma de requisitos con el cliente.
- **ID 04**: Realizar una exhaustiva limpieza de los datos para garantizar su calidad.
- **ID 05**: Antes de comenzar la implementación de la aplicación, analizar los requisitos del proyecto y realizar una evaluación exhaustiva de las tecnologías y herramientas de BI antes de su implementación.
- **ID 06**: Prestar especial atención a la etapa de conocimiento del sector del cliente, investigando sobre el tema en cuestión.
- **ID 07**: Realizar una formación en la herramienta que se va a utilizar antes de comenzar la implementación.

Capítulo 4

Análisis

Según un estudio de la empresa consultora Gartner, estimó en 2016, que un 60 % de los proyectos de Big Data fracasan. Como explica Mark Tossell en su artículo "6 Reasons Why BI and Analytics Projects Fail – And How to Avoid It " [70] existen múltiples razones por las que los proyectos de Business Intelligence fracasan pero hay ciertos motivos que se repiten con asiduidad, en el que destaca, la de falta de claridad para establecer los objetivos. Muchos proyectos fracasan por no tener claro los objetivos y los resultados, el cliente no tiene muy claro que quiere o necesita y sus objetivos son conflictivos, variables y ambiguos. Esta falta de cooperación, comunicación y colaboración es una de las causas más comunes de fracaso.

En un proyecto de Business Intelligence es necesario seguir una planificación que comienza definiendo el objetivo del proyecto y cuál va a ser su respectivo alcance, suele finalizarse elaborando un informe o un cuadro de mando para facilitar la toma de decisiones.

En la Figura 4.1 se observa las etapas que debería tener la fase de análisis de un proyecto de BI correctamente realizado.

Figura 4.1: Etapas de la fase de análisis de un proyecto de BI (elaboración propia)

Este caso práctico participan dos empresas ficticias, la empresa cliente *Motorland TV* la cuál es una cadena de televisión especializada en el mundo del motor y la empresa proveedora *BI Analytics* que es una consultora especializada en crear aplicaciones de Business Intelligence para todo tipo de sectores diferentes, pese a ser un caso inventado podría tratarse de un caso real.

Esta aplicación es un proyecto de Inteligencia de Negocio ya que utiliza el análisis y visualización de los datos para facilitar la toma de decisiones y generar una ventaja competitiva respecto al resto de cadenas televisivas.

4.1. Definición conceptual del Proyecto de BI

El máximo dirigente de *Motorland TV* al enterarse de las continuas críticas en las redes sociales sobre la monotonía de sus retransmisiones deportivas en vivo, se le ocurre una idea de cómo atraer nuevos clientes y recuperar la ilusión por ver retransmisiones de la Fórmula 1 en directo. Esta idea consiste en un aplicación de Business Intelligence que ayude a los narradores a visualizar información histórica sobre pilotos, escuderías y circuitos con la cual puedan amenizar la carrera comentando información relevante, enlazándolo con lo que ocurre en directo.

Tras varias reuniones con el cliente, este ha planteado, que necesita una herramienta de Inteligencia de Negocio que le permita tener disponible la siguiente información:

- **Datos referentes a cada circuito que haya existido en la Fórmula 1**. Como son número de victorias, de Grandes Premios (GP), de podios y de poles por pilotos y escuderías, vuelta rápida, abandonos (DNF) más comunes, número de Grandes Premios celebrados, longitud del circuito y velocidad máxima del circuito. Desea poder consultar dicha información en base al circuito y a la fecha.
- **Datos referentes a cada piloto que ha pilotado en la Fórmula 1**. Como son número de victorias, de Grandes Premios (GP), de podios y de poles, vuelta rápida, tiempos por vuelta, tiempos de pit stop, puntos que ha obtenido, número de abandonos (DNF) y velocidad máxima. Desea poder consultar dicha información en base al circuito, al piloto y a la fecha.
- **Datos referentes a cada escudería que ha participado en la Fórmula 1**. Como son número de victorias, de Grandes Premios (GP), de podios y de poles, vuelta rápida, tiempos por vuelta, tiempo de pit stop, puntos que ha obtenido, número de abandonos (DNF) y velocidad máxima. Desea poder consultar dicha información en base al circuito, a la escudería y a la fecha.
- **Datos globales**. Como número de Grandes Premios (GP), número de escuderías, número de pilotos, número de circuitos, lista de ganadores y localización de los circuitos. Desea poder consultar dicha información en base a la fecha.
- **Datos globales de los pilotos**. Como número de Grandes Premios (GP), de podios, de poles, de abandonos (DNF), de campeonatos mundiales, posición en el ranking de pilotos, una puntuación general que evalúe a los pilotos reduciendo el efecto del monoplaza. Esa puntuación estará creada por una puntuación respecto a su compañero de equipo, puntuación en lluvia, puntuación respecto al resto de pilotos, puntuación en base a la seguridad del piloto y una puntuación en base a la regularidad en sus resultados. Todas estas puntuaciones con las que se crea la puntuación general también se desean consultar. Desea consultar dicha información en base al piloto y a la fecha.
- **Datos globales de las escuderías**. Como número de Grandes Premios (GP), de podios, de poles, de abandonos (DNF), campeonatos mundiales de constructores y posición en el ranking de constructores. Desea consultar dicha información en base a la escudería y a la fecha.

El cliente ha adquirido los derechos para utilizar los datos procesados por Rohan Rao y disponibles en Kaggle [71] los cuales se irán actualizando mensualmente. Para el resto de información complementaria se plantea utilizar otras fuentes de uso libre como Wikipedia, respetando siempre los correspondientes derechos de propiedad comercial e intelectual.

A partir de las necesidades identificadas junto con el cliente, y la documentación facilitada proporcionada por este, vamos a realizar la fase inicial de planificación de un proyecto de BI que posteriormente, una vez sea implementado, obtengamos como resultado una herramienta final consistente en varios dashboards que permitan visualizar fácilmente los datos de la Fórmula 1 y satisfacer de manera integrada esas necesidades.

4.2. Identificación de requerimientos

Tras las subsiguientes reuniones celebradas con el cliente se han identificado y acordado con el mismo, unos requisitos para abordar cada una de esas necesidades en el Proyecto de BI.

- Qué datos se deben de calcular.

- Qué campos se deben de utilizar para segmentar esos datos.

Se ha sintetizado esta información en las siguientes tablas:

Dato / Evaluado por	Longitud	NºGP	Vuelta rápida	Nº abandonos	Velocidad Max
Circuito					
Fecha					

Tabla 4.1: Tabla de requerimientos de los datos del Circuito

Dato / Evaluado por	Nº Victorias	NºGP	NºPodios	NºPoles	Vuelta rápida	Escuderías
Circuito						
Piloto						
Fecha						

Dato / Evaluado por	T.por vuelta	T. Pitstop	Puntos	Velocidad Max	Nº abandonos
Circuito					
Piloto					
Fecha					

Tabla 4.2: Tabla de requerimientos de los datos de cada piloto en un circuito concreto

Dato / Evaluado por	Nº Victorias	NºGP	NºPodios	NºPoles	Vuelta rápida	Pilotos
Circuito						
Escudería						
Fecha						

Dato / Evaluado por	T.por vuelta	T. Pitstop	Puntos	Velocidad Max	Nº abandonos
Circuito					
Escudería					
Fecha					

Tabla 4.3: Tabla de requerimientos de los datos de cada escudería en un circuito concreto

Dato / Evaluado por	NºGP	NºEscuderías	NºPilotos	NºCircuitos	Ganadores
Fecha					

Tabla 4.4: Tabla de requerimientos de los datos globales

Dato / Evaluado por	NºGP	NºPodios	NºPoles	Nº Abandonos	Campeonatos	Ranking
Piloto						
Fecha						

Dato / Evaluado por	Puntuación General	Puntuación vs compañero	Puntuación vs resto pilotos
Piloto			
Fecha			

Dato / Evaluado por	Puntuación en lluvia	Puntuación seguridad	Puntuación regularidad
Piloto			
Fecha			

Tabla 4.5: Tabla de requerimientos de los datos globales de los pilotos

Dato / Evaluado por	NºGP	NºPodios	NºPoles	Nº abandonos	Campeonatos	Ranking
Escudería						
Fecha						

Tabla 4.6: Tabla de requerimientos de los datos globales de las escuderías

4.3. Definición de indicadores

Posteriormente a la identificación de requerimientos se debe definir los Indicadores, Medidas y KPI´s (key performance indicator) para cada uno de los elementos a medir.

Un indicador corresponde a un dato sencillo, una medida a una medida calculada a partir de los datos y un KPI es un indicador o medida junto con unos valores objetivo.

Cada indicador debe estar descrito en base a los siguientes campos:

- Definición: ¿qué representa el indicador?
- Cálculo: ¿cómo se calcula el indicador?
- Excepciones: ¿existen casos excepcionales?
- Fuentes de datos: ¿qué datos son necesarios para calcular el indicador?

Victoria	
Definición	Indicador que define si un piloto ha ganado una carrera, es decir, ha finalizado la carrera primero.
Cálculo	Victoria = 1 si posición final carrera = 1 Victoria = 0 en caso contrario
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular la posición final en carrera.

Tabla 4.7: Tabla de definición del indicador Victoria

Podio	
Definición	Indicador que define si un piloto ha terminado en el podio de una carrera, es decir, ha finalizado la carrera entre las tres primeras posiciones.
Cálculo	Podio = 1 si posición final carrera <= 3 Podio = 0 en caso contrario
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular la posición final en carrera.

Tabla 4.8: Tabla de definición del indicador Podio

Pole	
Definición	Indicador que define si un piloto ha comenzado en la "pole position" de una carrera, es decir, ha comenzado la carrera en la primera posición.
Cálculo	Pole = 1 si posición inicial carrera = posición clasificación = 1 Pole = 0 en caso contrario
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular la posición inicial en carrera.

Tabla 4.9: Tabla de definición del indicador Pole

KPI_vs_compañero_equipo	
Definición	Métrica que indica la puntuación de un piloto respecto a su compañero de equipo. Es una medida flotante acotada en el rango 0-100. Esta variable recoge las diferencias entre compañeros de equipo en múltiples variables como posición final en carrera, diferencias en el número de accidentes... Valores grandes de está variable indican grandes diferencias a favor del piloto en comparación con su compañero de equipo. En esta variable apenas influye la calidad del monoplaza. Más detalles de la métrica en el Capítulo 7.
Cálculo	Siendo X el conjunto de variables que utilizaremos para definir la métrica de cada piloto y siendo β los pesos asignados a cada variable, las variables que se utilizarán son: Posición inicial y final en carrera y posición en alcanzar la vuelta rápida, también el número de victorias, de podios, de poles, de accidentes y colisiones, todas las variables no utilizarán el valor total sino las diferencias con su compañero de equipo. KPI_compañero_equipo = $\frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (\sum_{i=1}^{p} X_i \cdot \beta_i)$
Excepciones	Si un piloto ha tenido varios compañeros de equipo en un año se realiza la media de resultados de todos los compañeros de equipo.
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular posición inicial y final carrera, número de accidentes...

Tabla 4.10: Tabla de definición del indicador KPI_compañero_equipo

KPI_Lluvia	
Definición	Métrica que indica la puntuación de un piloto respecto a los resultados en carreras con lluvia. Es una medida flotante acotada en el rango 0-100. Recoge información respecto a los resultados en lluvia de un piloto. Valores grandes de está variable indican buenos resultados del piloto en carreras con lluvia. Se define como carrera con lluvia si en la descripción del clima de la Wikipedia aparece alguna de las siguientes palabras: “rain”,“rainy”,“wet”,“showers”,“Drizzly”. Más detalles de la métrica en el Capítulo 7.
Cálculo	Siendo X el conjunto de variables que utilizaremos para definir la métrica de cada piloto y siendo β los pesos asignados a cada variable, las variables que se utilizarán son: Posición final en carrera y en alcanzar la vuelta rápida, en carreras con lluvia. KPI_Lluvia $= \frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (\sum_{i=1}^{p} X_i \cdot \beta_i)$
Excepciones	Sólo se tienen en cuenta carreras dónde ha llovido el día de carrera. Si ha llovido el día de clasificación pero no de carrera no se tienen en cuenta.
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular posición en carrera, posición de clasificación, número de accidentes, meteorología de la carrera...

Tabla 4.11: Tabla de definición del indicador KPI_Lluvia

KPI_evitar_accidentes	
Definición	Métrica que mide la habilidad evitando accidentes de un piloto. Es una medida flotante acotada en el rango 0-100. Recoge información respecto a la cantidad de accidentes que ha provocado y número de colisiones múltiples en las que ha participado. Valores grandes de está variable indica que el piloto ha participado en pocos accidentes y colisiones múltiples. Más detalles de la métrica en el Capítulo 7.
Cálculo	Siendo X el conjunto de variables que utilizaremos para definir la métrica de cada piloto y siendo β los pesos asignados a cada variable, las variables que se utilizarán son: Número de accidentes provocados y número de colisiones en las que ha participado, utilizarán tanto el valor total como la comparativa con su compañero de equipo. KPI_evitar_accidentes $= \frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (\sum_{i=1}^{p} X_i \cdot \beta_i)$
Excepciones	Sólo se han tenido en cuenta accidentes en carrera no en clasificación o tiempos libres
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular número de accidentes y número de colisiones

Tabla 4.12: Tabla de definición del indicador KPI_evitar_accidentes

KPI_resultados_generales	
Definición	Métrica que indica la puntuación de un piloto respecto al resto de pilotos de la Formula 1 en ese año concreto. Es una medida flotante acotada en el rango 0-100. Esta variable recoge las diferencias entre todos los pilotos del año en múltiples variables como posición final en carrera, diferencias en el número de accidentes... Valores grandes de está variable indican grandes diferencias a favor del piloto en comparación con el resto de competidores en las que sí influye la calidad del monoplaza pero se reduce la influencia de la calidad del compañero de equipo. Más detalles de la métrica en el Capítulo 7.
Cálculo	Siendo X el conjunto de variables que utilizaremos para definir la métrica de cada piloto y siendo β los pesos asignados a cada variable, las variables que se utilizarán son: Posición final en carrera, clasificación y en alcanzar la vuelta rápida, también el número de victorias, de podios, de poles, de accidentes y colisiones, todas las variables utilizarán el valor total. KPI_resultados_generales = $\frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (\sum_{i=1}^{p} X_i \cdot \beta_i)$
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular posición en carrera, posición de clasificación, número de accidentes...

Tabla 4.13: Tabla de definición del indicador KPI_resultados_generales

KPI_regularidad	
Definición	Métrica que indica la regularidad de un piloto. Es una medida flotante acotada en el rango 0-100. Esta variable recoge la variabilidad de un piloto en la posición obtenida en carrera, en la salida, en el ranking para alcanzar la vuelta rápida y en el número de posiciones ganadas en carrera. Valores grandes de está variable indica que un piloto es regular y consistente en sus resultados obtenidos. Más detalles de la métrica en el Capítulo 7.
Cálculo	Siendo X el conjunto de variables que utilizaremos para definir la métrica de cada piloto y siendo β los pesos asignados a cada variable, las variables que se utilizarán son: Desviación típica en la posición final en carrera, clasificación y en alcanzar la vuelta rápida. KPI_regularidad = $\frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (\sum_{i=1}^{p} X_i \cdot \beta_i)$
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular posición en carrera y posición de clasificación

Tabla 4.14: Tabla de definición del indicador KPI_regularidad

scoreAI	
Definición	Métrica que indica la puntuación de un piloto en una temporada reduciendo el efecto del monoplaza que conduce. Teniendo en cuenta sus habilidades respecto a su compañero de equipo, en lluvia, evitando accidentes, respecto al resto de pilotos y siendo consistente. Es una métrica con rango (0-100), siendo 0 un mal rendimiento y 100 un rendimiento excelente del piloto. Más detalles de la métrica en el Capítulo 7.
Cálculo	Siendo X el conjunto de variables que utilizaremos para definir la métrica de cada piloto y siendo β los pesos asignados a cada variable, las variables que se utilizarán son: `KPI_compañero_equipo`, `KPI_lluvia`,`KPI_evitar_accidentes` `KPI_resultados_generales` y `KPI_regularidad`. $scoreAI = \frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (\sum_{i=1}^{p} X_i \cdot \beta_i)$
Excepciones	Si un piloto ha pilotado en varias escuderías en un mismo año el *scoreAI* del piloto en ese año es el *scoreAI* promedio en cada una de las escuderías
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular posición en carrera, posición de clasificación, número de accidentes...

Tabla 4.15: Tabla de definición del indicador *scoreAI*

Ranking Piloto	
Definición	Indicador que calcula la posición de un piloto en el ranking de pilotos de un año concreto, siendo 1 la mejor posición. Para crear el ranking, se tiene en cuenta primero la suma total de puntos y si hay empate a puntos, es la suma de victorias como indican en el reglamento oficial de la Formula 1 en el apartado 7.2 establecido por la FIA. [72]
Cálculo	Se ordenan los pilotos de manera descendente en base al criterio = "Suma de puntos totales", si hay empates en la suma de puntos entre dos o más pilotos, se utiliza el segundo criterio: el número de victorias. Cuando ya están ordenados en base a los criterios calcula el ranking.
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular los puntos de cada carrera y la posición final en carrera.

Tabla 4.16: Tabla de definición del indicador Ranking Piloto

Campeonato Piloto	
Definición	Indicador que define si un piloto ha ganado el campeonato de pilotos, es decir, ha finalizado primero el ranking de pilotos en un año concreto.
Cálculo	Campeonato Piloto = 1 , si ranking piloto = 1, Campeonato Piloto = 0, en caso contrario
Excepciones	Hasta que no haya datos de la temporada siguiente no se asigna el campeonato de ese año. Así se evita dar el campeonato a un piloto antes de que haya finalizado la temporada.
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular los puntos de cada carrera y la posición final en carrera.

Tabla 4.17: Tabla de definición del indicador Campeonato Piloto

Ranking Escudería	
Definición	Indicador que calcula la posición de una escudería en el ranking de escuderías de un año concreto, siendo 1 la mejor posición. Para crear el ranking, se tiene en cuenta primero la suma total de puntos y si hay empate a puntos, es la suma de victorias, igual que con el ranking Piloto.
Cálculo	Se ordenan las escuderías de manera descendente en base al criterio = "Suma de puntos totales", si hay empates en la suma de puntos entre dos o más escuderías, se utiliza el segundo criterio: el número de victorias. Cuando ya están ordenados en base a los criterios calcula el ranking.
Excepciones	NA
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular los puntos de cada carrera y la posición final en carrera.

Tabla 4.18: Tabla de definición del indicador Ranking Escudería

Campeonato Constructores	
Definición	Indicador que define si una escudería ha ganado el campeonato de constructores, es decir, ha finalizado primero el ranking de escuderías en un año concreto.
Cálculo	Campeonato Constructores = 1 , si ranking escudería = 1, Campeonato Escudería = 0, en caso contrario
Excepciones	Hasta que no haya datos de la temporada siguiente no se asigna el campeonato de ese año. Así se evita dar el campeonato a una escudería antes de que haya finalizado la temporada.
Fuente de datos	Documentos CSV proporcionados por la empresa cliente. En particular los puntos de cada carrera y la posición final en carrera.

Tabla 4.19: Tabla de definición del indicador Campeonato Constructores

4.4. Profiling de los datos disponibles

Después de definir los indicadores se debe comprobar la disponibilidad de todos los datos requeridos para calcularlos. Los 14 archivos CSV proporcionados por la empresa cliente son los siguientes:

- **circuits.csv**: Contiene los datos respecto a los circuitos de la Fórmula 1. Descripción de cada columna de circuits.csv:
  - circuitId: Id del circuito.
  - circuitRef: Nombre referencia del circuito.
  - name: Nombre completo del circuito.
  - location: Ciudad donde se sitúa el circuito.
  - country: País donde se sitúa el circuito.
  - lat: Latitud.
  - lng: Longitud.
  - alt: Altitud.
  - url: Enlace del circuito en la Wikipedia.

circuitId	circuitRef	name	location	country	lat	lng	alt	url
1	albert_park	Albert Park Grand Prix Circuit	Melbourne	Australia	-378497	144968	10	http://en.wikipedia.org/wiki/Melbourne_Grand_Prix_Circuit

Figura 4.2: Columnas de circuits.csv

- **constructor_results.csv**: Contiene los datos respecto a los resultados de las carreras del campeonato de constructores. Descripción de cada columna de constructor_results.csv:
  - constructorResultsId: Id del resultado de la escudería.
  - raceId: Id de la carrera.
  - constructorId: Id de la escudería.
  - points: Puntos obtenidos por la escudería constructorId en la carrera raceId.
  - status: Estatus.

constructorResultsId	raceId	constructorId	points	status
1	18	1	14	\N

Figura 4.3: Columnas de constructor_results.csv

- **constructor_standings.csv**: Contiene los datos respecto a los resultados finales del campeonato de constructores. Descripción de cada columna de constructor_standings.csv:
  - constructorStandingsId: Id del resultado final de la escudería.
  - raceId: Id de la carrera.
  - constructorId: Id de la escudería.
  - points: Puntos obtenidos por la escudería constructorId en la carrera raceId.
  - position: Posición de la escudería constructorId en la carrera raceId.
  - positionText: Posición (texto) de la escudería constructorId en la carrera raceId.
  - wins: Número de victorias de la escudería constructorId en el campeonato de constructores al finalizar la carrera raceId.

constructorStandingsId	raceId	constructorId	points	position	positionText	wins
1	18	1	14	1	1	1

Figura 4.4: Columnas de constructor_standings.csv

- **constructors.csv**: Contiene los datos respecto a las escuderías de la Fórmula 1. Descripción de cada columna de constructors.csv:
  - constructorId: Id de la escudería.
  - constructorRef: Nombre referencia de la escudería.
  - name: Nombre completo de la escudería.
  - nationality: País de la escudería.
  - url: Enlace de la escudería en la Wikipedia.

constructorId	constructorRef	name	nationality	url
1	mclaren	McLaren	British	http://en.wikipedia.org/wiki/McLaren

Figura 4.5: Columnas de constructors.csv

- **driver_standings.csv**: Contiene los datos respecto a los resultados finales del campeonato de pilotos. Descripción de cada columna de driver_standings.csv:
  - driverStandingsId: Id del resultado final del piloto.
  - raceId: Id de la carrera.

- driverId: Id del piloto.
- points: Puntos obtenido por el piloto driverId en la carrera raceId.
- position: Posición del piloto driverId en la carrera raceId.
- positionText: Posición (texto) del piloto driverId en la carrera raceId.
- wins: Número de victorias del piloto driverId en el mundial de pilotos al finalizar la carrera raceId.

driverStandingsId	raceId	driverId	points	position	positionText	wins
1	18	1	10	1	1	1

Figura 4.6: Columnas de driver_standings.csv

- **drivers.csv**: Contiene los datos respecto a los pilotos de la Fórmula 1. Descripción de cada columna de drivers.csv:

  - driverId: Id del piloto.
  - driverRef: Nombre referencia del piloto.
  - number: Número del piloto en su monoplaza.
  - code: Código de tres letras que identifica al piloto.
  - forename: Nombre del piloto.
  - surname: Apellido del piloto.
  - dob: Fecha de nacimiento del piloto.
  - nationality: Nacionalidad del piloto.
  - url: Enlace del piloto en la Wikipedia.

driverId	driverRef	number	code	forename	surname	dob	nationality	url
1	hamilton	44	HAM	Lewis	Hamilton	lunes, 7 de enero de 1985	British	http://en.wikipedia.org/wiki/Lewis_Hamilton

Figura 4.7: Columnas de drivers.csv

- **lap_times.csv**: Contiene los datos respecto a los tiempos de las vueltas en las carreras de la Fórmula 1. Descripción de cada columna de lap_times.csv:

  - raceId: Id de la carrera.
  - driverId: Id del piloto.
  - lap: Número de vuelta.
  - position: Posición del piloto driverId en la vuelta lap.
  - time: Tiempo de la vuelta en formato mm:ss.fff (Minutos:segundos.milisegundos).
  - milliseconds: Tiempo de la vuelta en milisegundos.

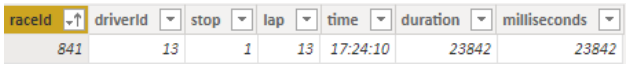
raceId	driverId	lap	position	time	milliseconds
1	67	46	11	1:30.180	90180

Figura 4.8: Columnas de lap_times.csv

- **pit_stops.csv**: Contiene los datos respecto a los tiempos de los pit stops en las carreras de la Fórmula 1. Descripción de cada columna de pit_stops.csv:

  - raceId: Id de la carrera.

- driverId: Id del piloto.
- stop: Número de parada.
- lap: Número de vuelta.
- time: Hora a la que se hizo la parada en formato hh:mm:ss (horas:minutos:segundos).
- duration: Tiempo del pit stop en formato ss.fff (segundos.milisegundos).
- milliseconds: Tiempo del pit stop en milisegundos.

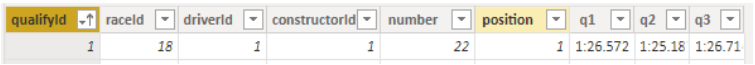


raceId	driverId	stop	lap	time	duration	milliseconds
841	13	1	13	17:24:10	23842	23842

Figura 4.9: Columnas de pit_stops.csv

- **qualifying.csv**: Contiene los datos respecto a las clasificatorias de las carreras de la Fórmula 1. Descripción de cada columna de qualifying.csv:

  - qualifyId: Id de la clasificatoria.
  - raceId: Id de la carrera.
  - driverId: Id del piloto.
  - constructorId: Id de la escudería.
  - number: Número del piloto en su monoplaza.
  - position: Posición en la clasificatoria final.
  - q1: Tiempo en la clasificatoria q1 en formato mm:ss.fff (Minutos:segundos.milisegundos).
  - q2: Tiempo en la clasificatoria q2 en formato mm:ss.fff (Minutos:segundos.milisegundos).
  - q3: Tiempo en la clasificatoria q3 en formato mm:ss.fff (Minutos:segundos.milisegundos).



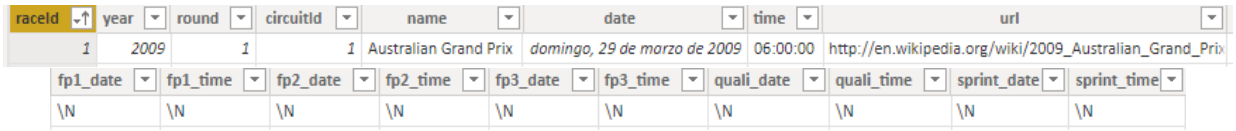
qualifyId	raceId	driverId	constructorId	number	position	q1	q2	q3
1	18	1	1	22	1	1:26.572	1:25.18	1:26.71

Figura 4.10: Columnas de qualifying.csv

- **races.csv**: Contiene los datos respecto a las carreras de la Fórmula 1. Descripción de cada columna de races.csv:

  - raceId: Id de la carrera.
  - year: Año.
  - round: Número de la carrera en esa temporada.
  - circuitId: Id del circuito.
  - name: Nombre completo del circuito.
  - date: Fecha de la carrera en formato AAAA-MM-DD (Año-mes-día).
  - time: Hora de la carrera en formato hh:mm:ss (horas:minutos:segundos).
  - url: Enlace de la carrera en la Wikipedia.
  - fp1_date: Fecha de los primeros entrenamientos libres en formato AAAA-MM-DD.
  - fp1_time: Hora de los primeros entrenamientos libres en formato hh:mm:ss.
  - fp2_date: Fecha de los segundos entrenamientos libres en formato AAAA-MM-DD.
  - fp2_time: Hora de los segundos entrenamientos libres en formato hh:mm:ss.
  - fp3_date: Fecha de los terceros entrenamientos libres en formato AAAA-MM-DD.

- fp3_time: Hora de los terceros entrenamientos libres en formato hh:mm:ss.
- quali_date: Fecha de la clasificatoria en formato AAAA-MM-DD.
- quali_time: Hora de la clasificatoria en formato hh:mm:ss.
- sprint_date: Fecha de la carrera a sprint en formato AAAA-MM-DD.
- sprint_time: Hora de la carrera a sprint en formato hh:mm:ss.



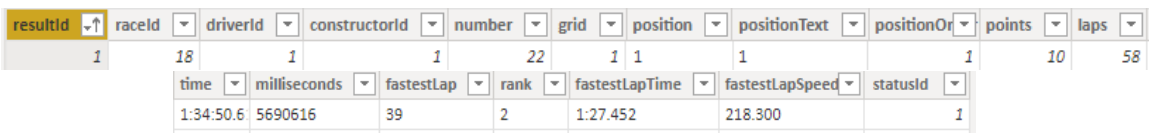
raceId	year	round	circuitId	name	date	time	url
1	2009	1	1	Australian Grand Prix	domingo, 29 de marzo de 2009	06:00:00	http://en.wikipedia.org/wiki/2009_Australian_Grand_Prix

fp1_date	fp1_time	fp2_date	fp2_time	fp3_date	fp3_time	quali_date	quali_time	sprint_date	sprint_time
\N	\N	\N	\N	\N	\N	\N	\N	\N	\N

Figura 4.11: Columnas de races.csv

- **results.csv**: Contiene los datos respecto a los resultados de cada piloto en cada carrera de la Fórmula 1. Descripción de cada columna de results.csv:

  - resultId: Id del resultado de la carrera.
  - raceId: Id de la carrera.
  - driverId: Id del piloto.
  - constructorId: Id de la escudería.
  - number: Número del piloto en su monoplaza.
  - grid: Posición del piloto driverId en la salida.
  - position: Posición final del piloto driverId, los abandonos son nulos.
  - positionText: Posición(texto) final del piloto driverId, los abandonos se indican con una letra.
  - positionOrder: Ranking final de la carrera, los abandonos no son nulos, los que abandonan primero obtienen las últimas posiciones.
  - points: Puntos obtenidos.
  - laps: Número de vueltas realizadas.
  - time: Tiempo total en realizar la carrera, se indica el tiempo en formato hh:mm:ss.fff (horas:minutos:segundos.milisegundos) si ha finalizado primero y en el resto se indica a cuanta diferencia ha tenido respecto al primero en formato + mm:ss.fff.
  - milliseconds: Tiempo total en realizar la carrera en milisegundos, aquí es tiempo total no diferencias respecto al primero.
  - fastestLap: Número de la vuelta en la que el piloto driverId ha realizado la vuelta rápida.
  - rank: Ranking en el que ha estado el piloto driverId en alcanzar la vuelta rápida.
  - fastestLapTime: Tiempo de la vuelta rápida del piloto driverId en formato mm:ss.fff.
  - fastestLapSpeed: Máxima velocidad media por vuelta del piloto driverId alcanzada en la carrera.
  - statusId: Estatus de la carrera del piloto driverId.



resultId	raceId	driverId	constructorId	number	grid	position	positionText	positionOr	points	laps
1	18	1	1	22	1	1	1	1	10	58

time	milliseconds	fastestLap	rank	fastestLapTime	fastestLapSpeed	statusId
1:34:50.6	5690616	39	2	1:27.452	218.300	1

Figura 4.12: Columnas de results.csv

- **seasons.csv**: Contiene los datos con los enlaces a la Wikipedia de todas las temporadas de la Fórmula 1. Descripción de cada columna de seasons.csv:
  - year: Año de la temporada.
  - url: Enlace de la temporada en la Wikipedia.

year	url
1950	http://en.wikipedia.org/wiki/1950_Formula_One_season

Figura 4.13: Columnas de seasons.csv

- **sprint_results.csv**: Contiene los datos respecto a las carreras a sprint, introducidas en 2021. [27] Descripción de cada columna de sprint_results.csv:
  - resultId: Id del resultado de la carrera a sprint.
  - raceId: Id de la carrera.
  - driverId: Id del piloto.
  - constructorId: Id de la escudería.
  - number: Número del piloto en su monoplaza.
  - grid: Posición del piloto driverId en la salida.
  - position: Posición final del piloto driverId, los abandonos son nulos.
  - positionText: Posición(texto) final del piloto driverId, los abandonos se indican con una letra.
  - positionOrder: Ranking final de la carrera, los abandonos no son nulos, los que abandonan primero obtienen las últimas posiciones.
  - points: Puntos obtenidos.
  - laps: Número de vueltas realizadas.
  - time: Tiempo total en realizar la carrera, se indica el tiempo en formato hh:mm:ss.fff (horas:minutos:segundos.milisegundos) si ha finalizado primero y en el resto se indica a cuanta diferencia ha tenido respecto al primero en formato + mm:ss.fff.
  - milliseconds: Tiempo total en realizar la carrera en milisegundos, aquí es tiempo total no diferencias respecto al primero.
  - fastestLap: Número de la vuelta en la que el piloto driverId ha realizado la vuelta rápida.
  - fastestLapTime: Tiempo de la vuelta rápida del piloto driverId en formato mm:ss.fff.
  - statusId: Estatus de la carrera del piloto driverId.

resultId	raceId	driverId	constructor	number	grid	position	positionText	positionOrde
1	1061	830	9	33	2	1	1	1

points	laps	time	milliseconds	fastestLap	fastestLapTime	statusId
3	17	25:38.426	1538426	14	1:30.013	1

Figura 4.14: Columnas de sprint_results.csv

- **status.csv**: Contiene los datos respecto al estado de cómo puede finalizar una carrera un piloto. Descripción de cada columna de status.csv:
  - statusId: Id del status.
  - status: Descripción del status.

Existen 140 valores de status diferentes entre los que se encuentra finalizado, accidente o fallo en la caja de cambios.

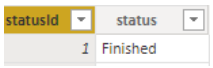


Figura 4.15: Columnas de status.csv

Tras analizar con el cliente sus requerimientos y haber comprobado la disponibilidad de la información proporcionada, de los 14 archivos proporcionados solo se utilizan los siguientes 8 archivos.

- circuits.csv
- constructors.csv
- drivers.csv
- lap_times.csv
- pit_stops.csv
- races.csv
- results.csv
- status.csv

Toda la información requerida por el cliente se encuentra en esos 8 archivos excepto un dato, el de longitud del circuito, esa información podría obtener a partir de la Wikipedia.

Revisando los archivos proporcionados por el cliente, se observa en la Figura 4.2 y posteriores que están todos los datos disponibles excepto la longitud del circuito. También podemos fijarnos en la Figura 4.9 como en la información proporcionada de pit stops han empezado a recogerse a partir del Id 841 que son los correspondientes a partir del año 2011.

Para comprobar la calidad de los datos me ayudo de la herramienta de Power BI, en este paso solo es utilizada para realizar un análisis previo de los mismos, Power BI tiene un apartado llamado PowerQuery para tratar la limpieza de los datos. Una opción que tiene es mostrar la calidad de las columnas mostrando (% de valores ausentes) e incluso te muestra la distribución de la clase.

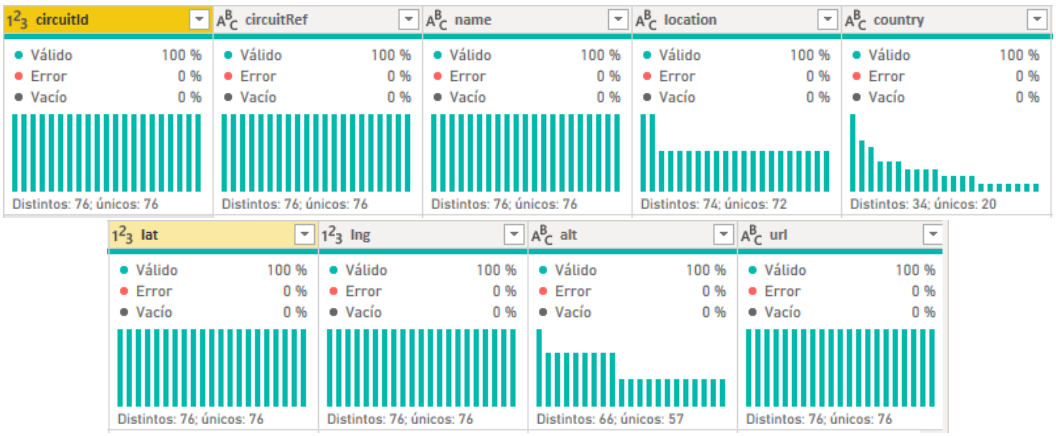


Figura 4.16: Calidad y distribución de las columnas del archivo circuits.csv

La herramienta muestra las tablas con los datos y nos permite observar la calidad de los mismos, por lo que podemos comprobar si realmente está funcionando como deseamos.

En todas las columnas de todos los archivos nos indica que hay 0 % de valores nulos pero si vemos las tablas con los datos eso no es así.

En el origen hay un carácter especial "\N", el cuál se utiliza para indicar un valor especial, Power BI lo detecta como un texto y provoca que transforme columnas numéricas que tienen "\N" como columnas de texto. Si se investiga el porqué de estos caracteres especiales no es por un fallo en el origen sino que suelen tener un sentido.

Muchos de ellos tienen que ver con abandonos de pilotos en un GP, cuando esto ocurre la columna que indica la posición de la carrera se indica como "\N" porque no ha llegado a finalizarla.

Es por ello que antes de realizar los cálculos debemos realizar una tarea de limpieza de datos, para que podamos utilizar los mismos de una manera correcta.

4.5. Identificación de hechos y dimensiones

El siguiente paso en el proyecto de BI es identificar los hechos y las dimensiones, es importante remarcar:

- Los hechos representan ¿qué queremos medir?
- Las dimensiones representan ¿cómo lo queremos medir?

Para identificar los hechos hay que tener en cuenta los indicadores definidos en el apartado 4.3. Concretamente, en los datos que deben utilizarse para calcular cada indicador.

Por tanto, hemos identificado los siguientes hechos:

- Número de Grandes Premios (GPs)
- Posición final en una carrera
- Posición inicial en una carrera
- Posición piloto en alcanzar vuelta rápida
- Velocidad Máxima de una carrera
- Tiempo vuelta rápida
- Tiempo vuelta
- Tiempo pit stop
- Puntos
- Número de abandonos

Para identificar las dimensiones hay que conocer en base a qué se quieren medir estos indicadores. Esta información fue proporcionada en el apartado 4.2. Concretamente qué segmentaciones de datos queremos realizar.

Analizando todos los indicadores, el cliente desea obtener la información en base a:

- Circuito
- Piloto
- Escudería
- Fecha

Circuito, piloto, escudería y fecha son dimensiones distintas y no forman ninguna jerarquía entre ellas, las denominaremos "Circuito", "Piloto", "Escudería" y "Fecha".

La fecha se subdivide en tres dimensiones: "Día", "Mes" y "Año" que nos permiten agrupar la información en base a esos campos. De hecho, los propios programas de Visualización, ETL, suelen crear estas jerarquías de fecha de forma automática como ocurre con Power BI.

4.6. Definir tablas de hechos

Después de realizar la identificación de hechos y dimensiones se deben definir las tablas de hechos.

Se tiene cinco procesos de negocio diferentes como son:

- Resultado de los pilotos y escuderías en cada carrera.
- Tiempo de cada vuelta de cada piloto en cada carrera.
- Tiempo de cada pit stop de cada piloto en cada carrera.
- Resultados de los pilotos en cada temporada.
- Resultados de las escuderías en cada temporada.

Los cinco procesos de negocio generan cinco tablas diferentes de hechos:

- **CarreraResultado**: Esta tabla agrupa todos los resultados de cada piloto y su escudería en cada carrera que ha celebrado.
- **TiempoVuelta**: Esta tabla agrupa todos los tiempos de vuelta de cada piloto y su escudería en cada carrera que ha celebrado.
- **TiempoPitStop**: Esta tabla agrupa todos los tiempos de parada en pit stops de cada piloto y su escudería en cada carrera que ha celebrado.
- **TemporadaPiloto**: Esta tabla agrupa los hechos relacionados con los resultados finales de un piloto en una temporada: *scoreAI*, número de puntos, puesto en el ranking...
- **TemporadaEscuderia**: Esta tabla agrupa los hechos relacionados con los resultados finales de una escudería en una temporada: número de puntos, número de Grandes Premios, puesto en el ranking...

4.7. Definir tablas de dimensiones

Para cada dimensión identificada en el apartado 4.5 se debe realizar una tabla de dimensiones. También se ha incluido la dimensión CarreraDescripcion la cuál dispone de información respecto a carreras. Esta tabla permite relacionar la dimensión Circuito con la tabla de hechos de Resultado de los pilotos y escuderías en cada carrera.

Dimensión	Definición
Circuito	La dimensión Circuito consta de los siguientes atributos: • circuitId: Id del circuito • circuitRef: Nombre referencia del circuito • name: Nombre completo del circuito • location: Ciudad donde se sitúa el circuito • country: País donde se sitúa el circuito • lat: Latitud • lng: Longitud • alt: Altitud • url: Enlace del circuito en la Wikipedia • image_url: Imagen del circuito en la Wikipedia
Piloto	La dimensión Piloto consta de los siguientes atributos: • driverId: Id del piloto • driverRef: Nombre referencia del piloto • number: Número del piloto en su monoplaza • code: Código de tres letras que identifica al piloto • forename: Nombre del piloto • surname: Apellido del piloto • dob: Fecha de nacimiento del piloto • nationality: Nacionalidad del piloto • url: Enlace del piloto en la Wikipedia • country: País de origen del piloto • complete_name: Nombre y apellido del piloto • image_url: Imagen del piloto en la Wikipedia
Escudería	La dimensión Escudería consta de los siguientes atributos: • constructorId: Id de la escudería • constructorRef: Nombre referencia de la escudería • name: Nombre completo de la escudería • nationality: Nacionalidad de la escudería • url: Enlace de la escudería en la Wikipedia • image_url: Imagen de la escudería en la Wikipedia • country: País de la escudería
CarreraDescripcion	La dimensión CarreraDescripcion consta de los siguientes atributos: • raceId: Id de la carrera • year: Año • round: Número de la carrera en esa temporada • circuitId: Id del circuito • name: Nombre completo del circuito • date: Fecha de la carrera en formato AAAA-MM-DD (Año-mes-día) • time: Hora de la carrera en formato hh:mm:ss (horas:minutos:segundos) • url: Enlace de la carrera en la Wikipedia • circuitLengthCleaned: Longitud en kilómetros del circuito
Status	La dimensión Status únicamente consta de los atributos statusId y status es por ello que no se crea una tabla para la dimensión status y corresponderá con un campo en la tabla de hechos carreraResultado.
Year	La dimensión Year únicamente consta del atributo year.

Tabla 4.20: Tabla de definición de las dimensiones

4.8. Establecer bus de dimensiones

Después de definir las tablas de hechos y de dimensiones se debe relacionar que dimensiones aplican a cada uno de los hechos definidos.

En la tabla siguiente se muestra el bus de dimensiones para el proyecto de BI del caso práctico definido.

Para rellenar el bus de dimensiones hay que fijarse en las tablas de requerimientos definidas en el apartado 4.2 y en los datos necesarios para calcular los indicadores definidos en el apartado 4.3

Que\Como	Circuito	Piloto	Escudería	CarreraDescripcion	Status	Year
NºGPs	X	X	X			
Posición final	X	X	X	X		X
Posición inicial	X	X	X	X		X
Posición en alcanzar vuelta rápida	X	X	X	X		X
Velocidad máxima	X	X	X	X		X
Tiempo vuelta rápida	X	X	X	X		X
Tiempo vuelta	X	X	X	X		X
Tiempo pit stop	X	X	X	X		X
Puntos	X	X	X	X		X
Nº abandonos	X	X	X	X	X	X

Tabla 4.21: Tabla del bus de dimensiones

Capítulo 5

Diseño

5.1. Arquitectura de la solución

En esta sección se especifica la arquitectura de la que se compone el proyecto desarrollado. La arquitectura del proyecto comprende la definición de los componentes lógicos del sistema y como estos se relacionan entre ellos. La arquitectura lógica del sistema consta de las siguientes fases: Extracción, Transformación y Carga (ETL), Almacenamiento y Visualización, (ver Figura 5.1).

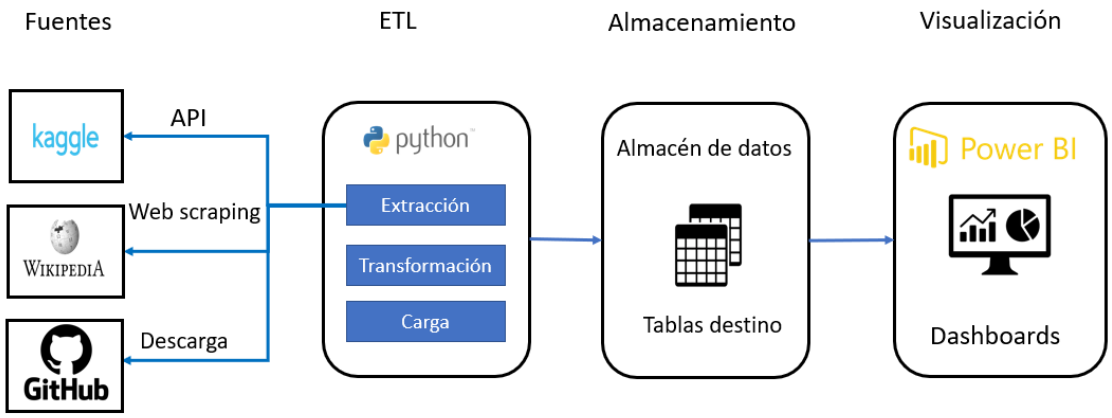


Figura 5.1: Esquema de la arquitectura de la solución (elaboración propia)

1. **Módulo Python de ETL**. Desde este módulo se establece conexión con la API de Kaggle donde se obtiene la información de la Fórmula 1. Una vez extraída esta información, desde el mismo módulo es transformada y estructurada en dataframes listos para ser cargados en el almacén de datos. En este módulo se realiza el web scraping de la Wikipedia y la descarga de datos de GitHub para completar la información inicial. También se crea la métrica *scoreAI* y las métricas parciales las cuales se detallan en el Capítulo 7.

2. **Módulo Almacenamiento**. Una vez transformada la información se realiza un volcado de datos a archivos CSV los cuáles son fácilmente accesibles con la herramienta Power BI.

3. **Módulo de visualización**. Cuando se tienen los datos almacenados en el almacén de datos se desarrollan los diferentes cuadros de mando con la herramienta Power BI que van a componer la aplicación.

5.2. Establecer e implementar el esquema del almacén de datos

Tras establecer el bus de dimensiones, se emplea esta información junto con las tablas de hechos y las tablas de dimensiones para relizar el esquema del almacén de datos (Data Warehouse).

Si se implementa en forma de base de datos relacional, esta contendrá cinco tablas de hechos con los campos identificado en el apartado 4.6, junto con las relaciones a las tablas de dimensiones descritas en el bus de dimensiones. Además contendrá las tablas de dimensiones definidas en el apartado 4.7.

El esquema elegido para construir el almacén de datos es fundamental a la hora de buscar un mejor rendimiento a la hora de implementarlo. Existen 2 esquemas fundamentales y principales a la hora de hablar de Business Intelligence: esquema en estrella y esquema en copo de nieve.

- **Esquema en estrella (Star schema)**: En el centro de la estrella puede tener una tabla de hechos y varias tablas de dimensiones asociadas. Se conoce como esquema en estrella porque su estructura se asemeja a una estrella. El modelo de datos Star schema es el tipo más simple de esquema de almacén de datos. Un ejemplo de esquema en estrella se observa en la Figura 5.2.

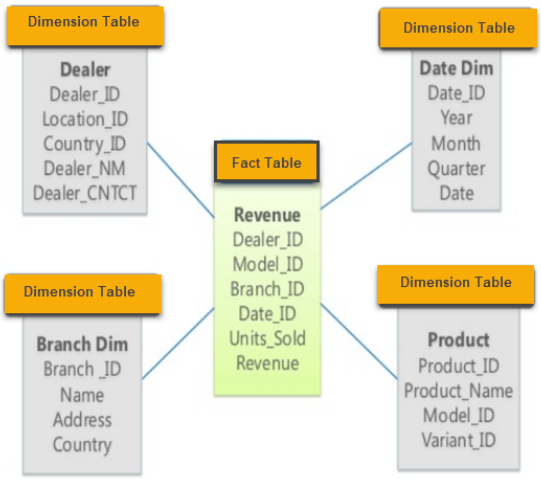


Figura 5.2: Ejemplo de esquema de estrella (extraído de [73])

- **Esquema en copo de nieve (Snowflake schema)**: Es una disposición lógica de tablas en un almacén de datos multidimensional de modo que se parece a la forma de un copo de nieve. Un esquema de copo de nieve es una extensión de un esquema de estrella y agrega dimensiones adicionales. Las tablas de dimensiones están normalizadas, lo que divide los datos en tablas adicionales. Un ejemplo de esquema en copo de nieve se observa en la Figura 5.3.

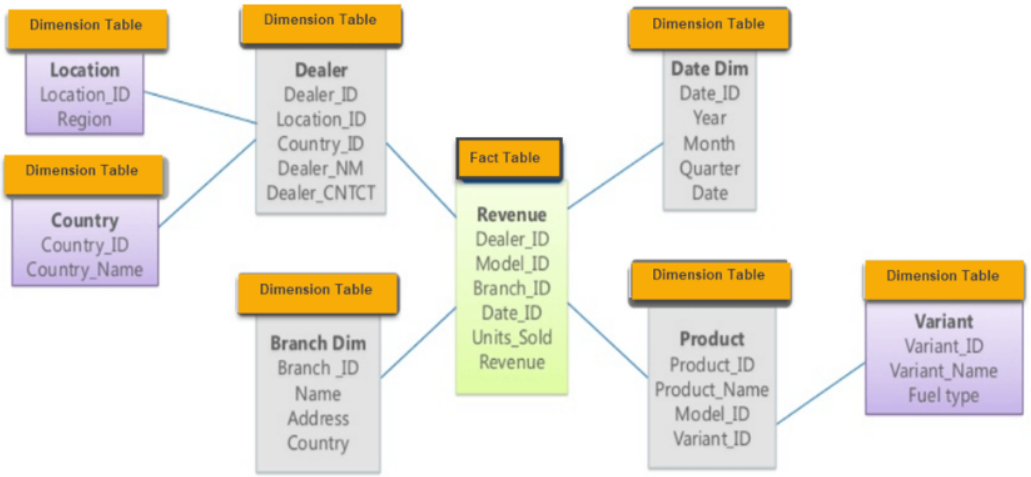


Figura 5.3: Ejemplo de esquema de copo de nieve (extraído de [73])

Se ha detallado las características de cada uno de los dos esquemas en la Tabla 5.1.[73]

Teniendo en cuenta los datos que se disponen y las características de ambos esquemas, se ha optado por realizar un esquema híbrido entre ambos esquemas, ya que se trata de un esquema en estrella ex-

cepto por la dimensión Circuito, la cuál está conectada a la dimensión CarreraDescripción en vez de a una tabla de hechos como sería en un esquema de estrella puro. Igual ocurre con la dimensión Year y CarreraDescripcion.

Después de haber elegido el esquema de almacén de datos, el esquema de nuestro proyecto con los campos contraídos se observa en la Figura 5.4 mientras que el esquema con los campos expandidos se observa en la Figura 5.5.

En ellos se observan las relaciones entre las diferentes tablas de hechos y de dimensiones. Las tablas de dimensiones poseen una clave primaria simple y las tabla de hechos están formadas por estas claves primarias de las tablas de dimensiones (actuando como "Foreign Key", FK).

Esquema de estrella	**Esquema de copo de nieve**
Las jerarquías para las dimensiones se almacenan en la tabla de dimensiones.	Las jerarquías se dividen en tablas separadas.
Contiene una tabla de hechos rodeada de tablas de dimensiones.	Una tabla de hechos rodeada por una tabla de dimensiones que a su vez están rodeadas por una tabla de dimensiones
En un esquema en estrella, solo la combinación única crea la relación entre la tabla de hechos y las tablas de dimensiones.	Un esquema de copo de nieve requiere muchas uniones para obtener los datos.
Diseño de base de datos simple.	Diseño base de datos muy complejo.
La tabla de dimensión única contiene datos agregados.	Datos divididos en diferentes tablas de dimensiones.
El procesamiento de cubos es más rápido.	El procesamiento de cubos puede ser lento debido a la unión compleja.
Ofrece consultas de mayor rendimiento. Las tablas se pueden conectar con múltiples dimensiones.	El esquema Snowflake está representado por una tabla de hechos centralizada que es poco probable que esté conectada con múltiples dimensiones.

Tabla 5.1: Características de ambos esquemas

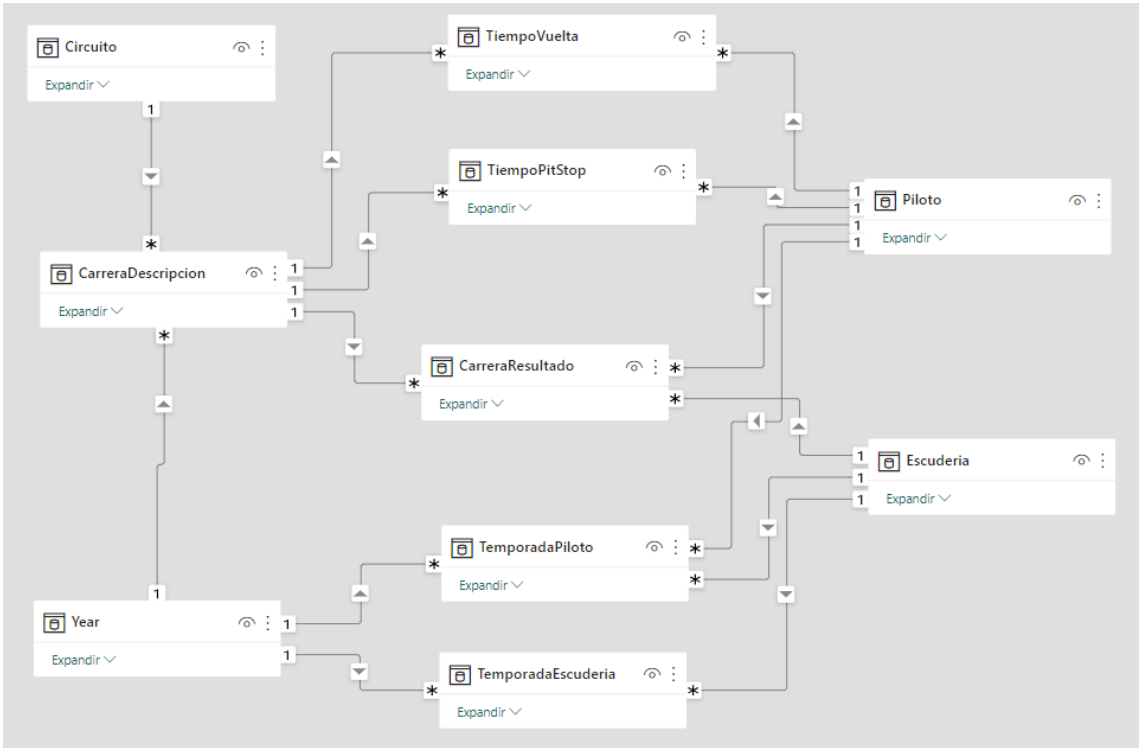


Figura 5.4: Esquema del almacén de datos con los campos contraídos

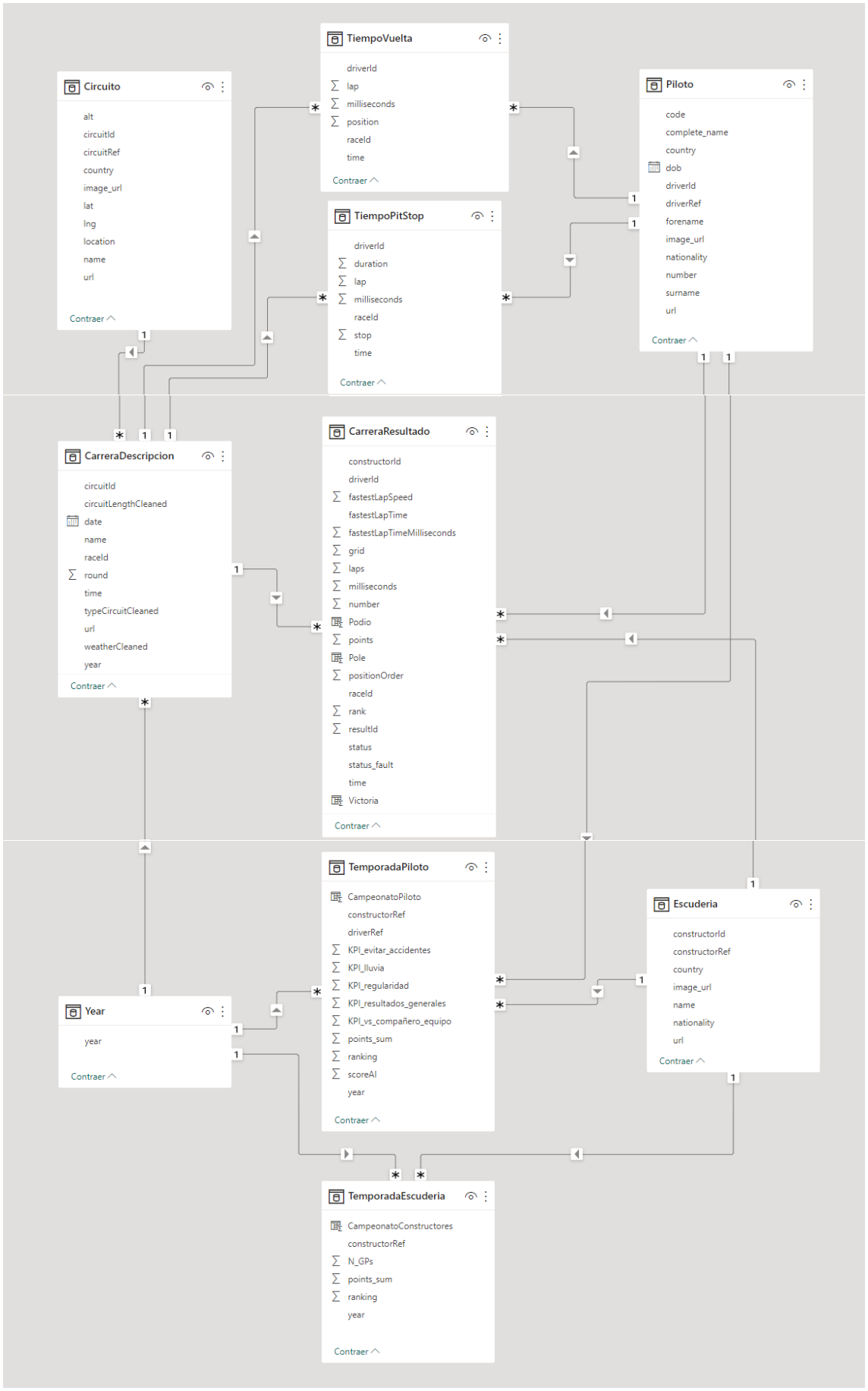


Figura 5.5: Esquema del almacén de datos con los campos expandidos

5.3. Diseño de bocetos iniciales (mockups)

La interfaz ha sido diseñada mediante mockups, es decir, representaciones del prototipo del proyecto que hemos realizado.

Dentro de estos bocetos iniciales, se ha indicado de color rosa los filtros en los que el usuario interactúa con la aplicación, en azul se han escrito anotaciones para aclarar dudas sobre el boceto realizado. El diseño final de la aplicación puede tener ligeros cambios respecto al diseño de estos mockups.

En total se han creado cinco mockups uno para cada dashboard de la aplicación final:

- **Mockup Circuito**: Prototipo inicial para el dashboard Circuito, (ver Figura 5.6).
- **Mockup Piloto**: Prototipo inicial para el dashboard Piloto, (ver Figura 5.7).
- **Mockup Escudería**: Prototipo inicial para el dashboard Escudería, (ver Figura 5.8).
- **Mockup Temporada**: Prototipo inicial para el dashboard Temporada, (ver Figura 5.9).
- **Mockup 1 vs 1**: Prototipo inicial para el dashboard 1 vs 1, (ver Figura 5.10).

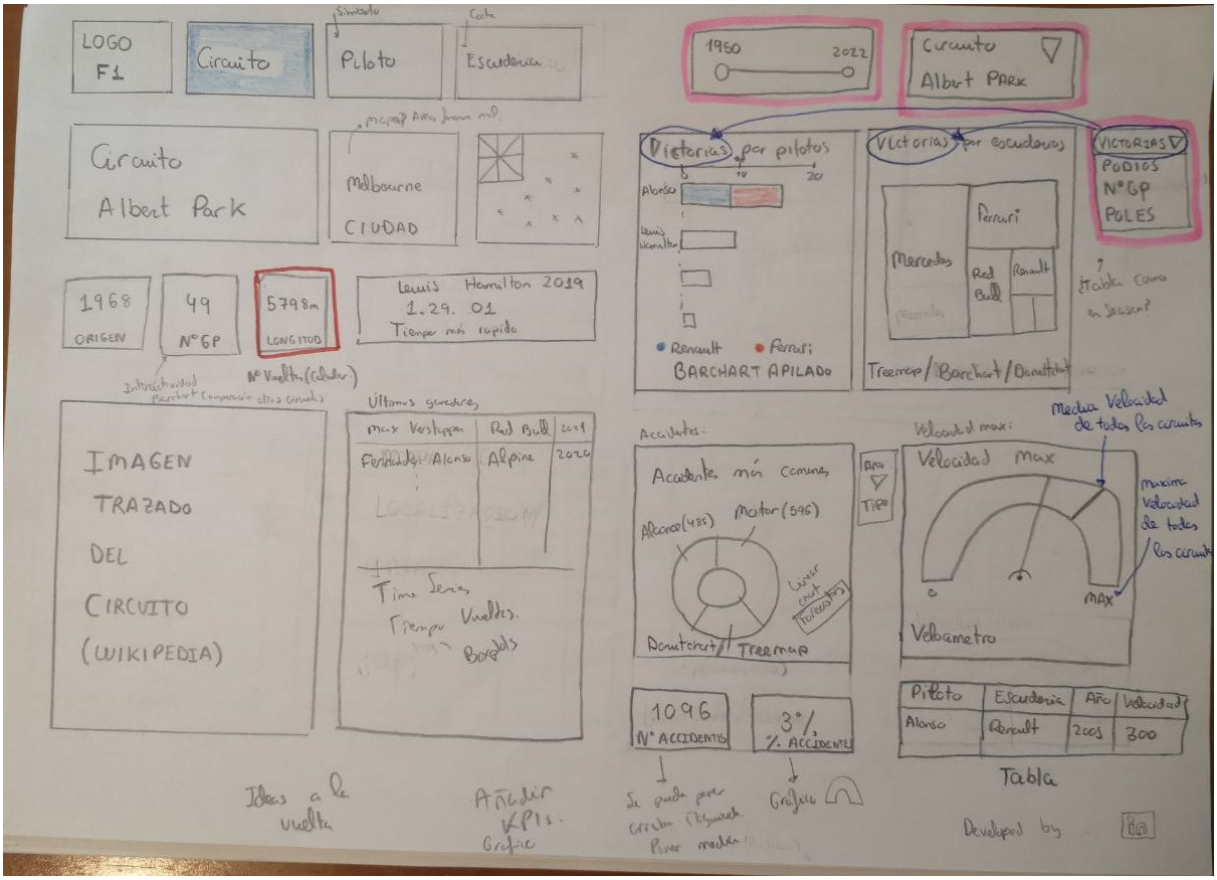


Figura 5.6: Mockup Circuito

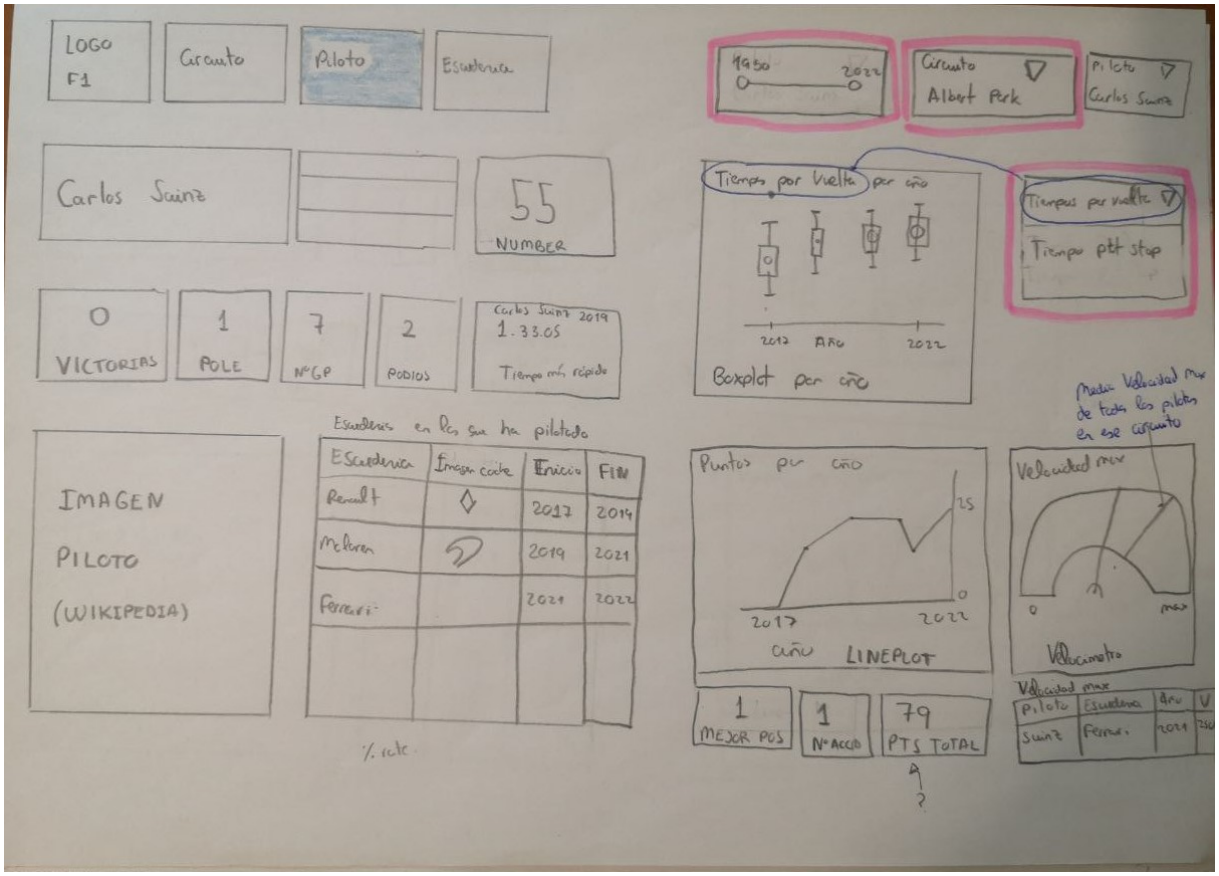


Figura 5.7: Mockup Piloto

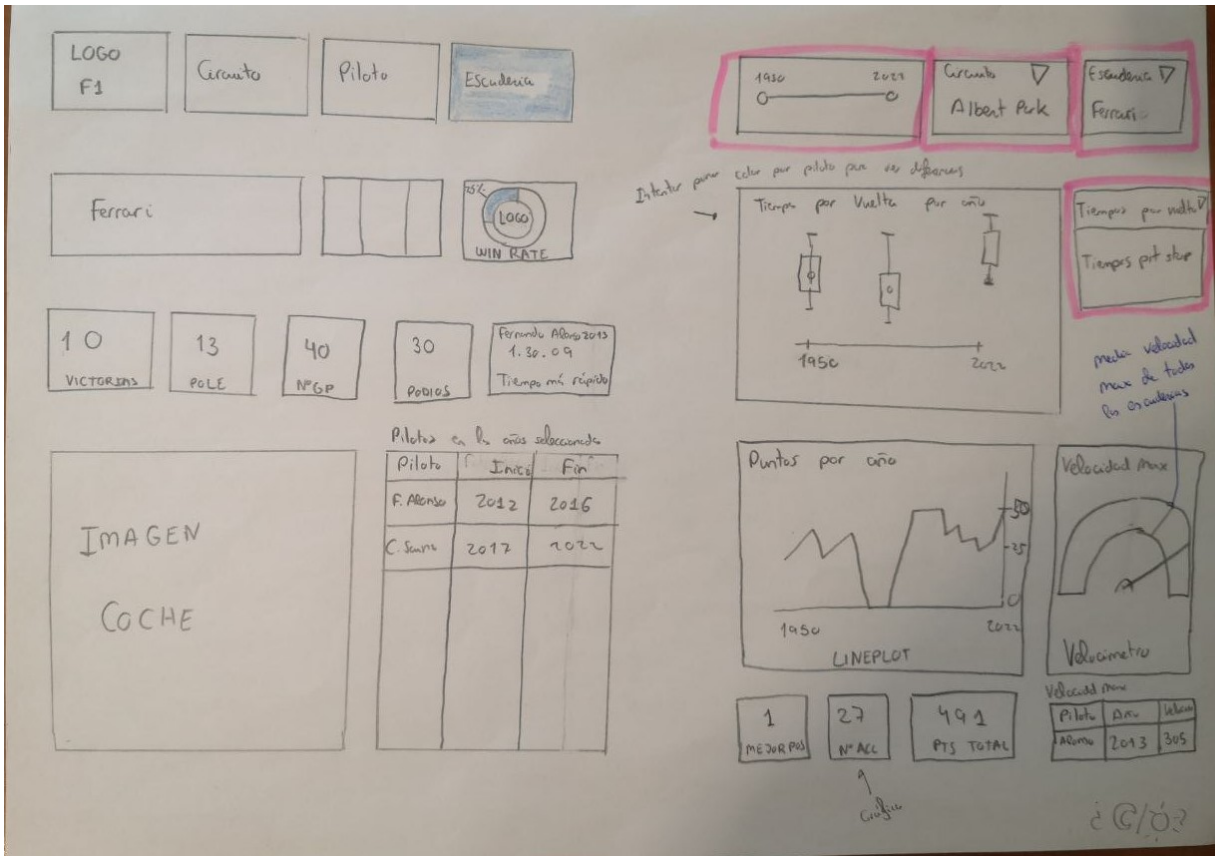


Figura 5.8: Mockup Escudería

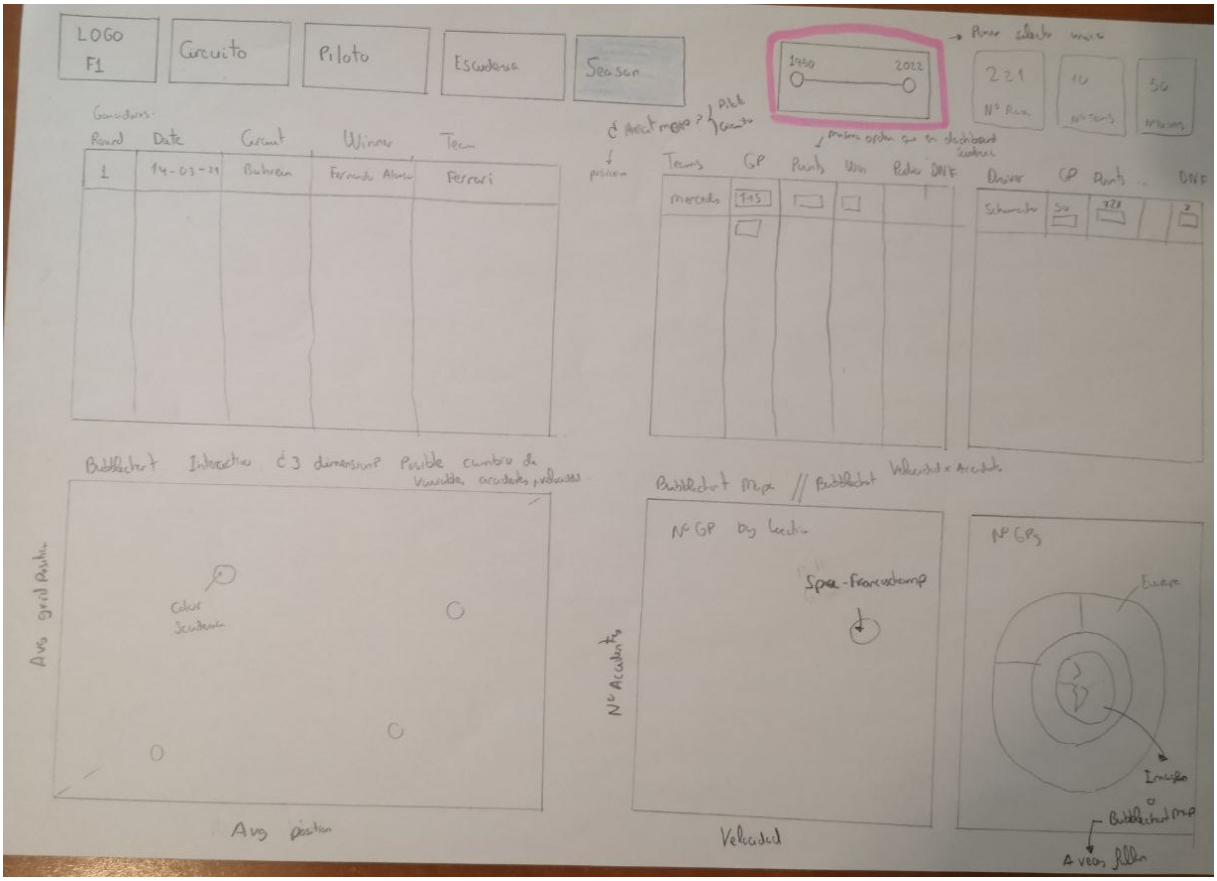


Figura 5.9: Mockup Temporada

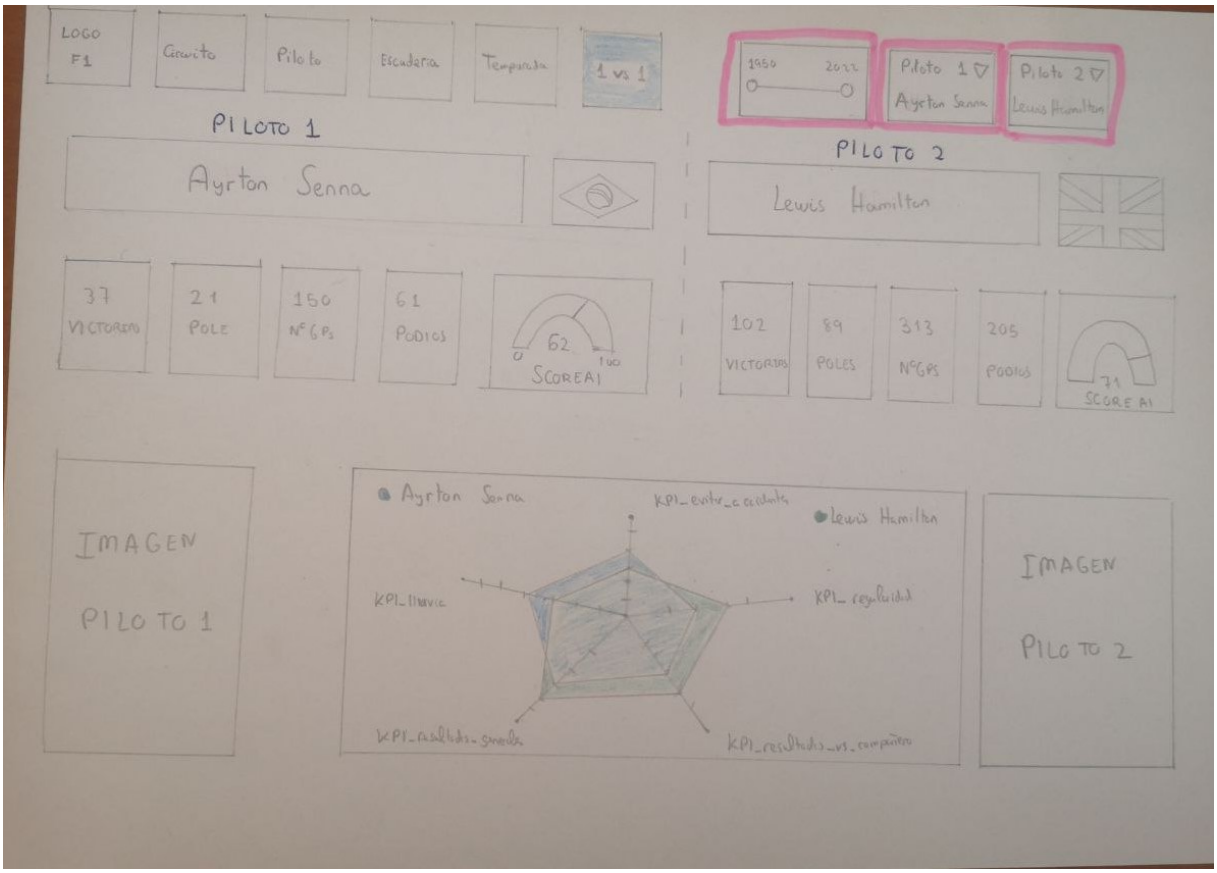


Figura 5.10: Mockup 1 vs 1

5.4. Entorno tecnológico

Estas han sido las tecnologías escogidas para el desarrollo de este Trabajo de Fin de Grado:

Figura 5.11: Tecnologías utilizadas

- **Python** 3.10.7 [74]

  Python, en su versión 3.10.7, ha sido utilizado como lenguaje de programación para implementar la ingesta y la transformación de datos en este proyecto. A través de Python se realiza la conexión con la API de Kaggle para acceder a la información de la Fórmula 1 y también mediante técnicas de web scraping se obtienen los datos de la Wikipedia. Estos son recogidos y posteriormente almacenados en el destino.

  Python es un lenguaje de programación interpretado que se encuentra enfocado hacia la legibilidad del código. La ventaja principal de Python sobre otros lenguajes de programación, Java o C++, es que es posible desarrollar código de una forma más rápida e intuitiva, mejorando la productividad y reduciendo el número total de líneas de código en la implementación. Otro aspecto a tener en cuenta es la portabilidad del lenguaje, permitiendo una correcta ejecución en máquinas Mac, Linux o Windows. Por último, una de las principales ventajas de Python es la comunidad que lleva consigo, lo que implica un alto grado de cuidado del lenguaje así como el gran número de sitios web que proporcionan ayuda sobre errores, configuración y guías para el desarrollo.

  Si nos fijamos en la evolución de este lenguaje a lo largo del tiempo, podemos ver que su popularidad ha ido siempre en aumento, siendo utilizado para el desarrollo de aplicaciones web, aplicaciones de escritorio, servidores de red, inteligencia artificial... Al principio, la popularidad de Python fue creciendo debido a su facilidad de aprendizaje y a la simplicidad del código, pero posteriormente este crecimiento aumentó considerablemente gracias a la explosión de campos como la inteligencia artificial, el aprendizaje automático o el análisis de datos.

  Otra de las principales ventajas de Python es la amplia cantidad de APIs y bibliotecas que podemos encontrar en la red. En este aspecto, la biblioteca que ha presentado un mayor crecimiento ha sido *Pandas*, utilizada en este proyecto para el tratamiento de datos.

  La extracción de la información del proyecto se realizará mediante acceso a la API de Kaggle y mediante técnicas de web scraping. Web scraping es una técnica utilizada para acceder a información de sitios web. Esta técnica consiste en simular la navegación de un usuario real por la web a través del código fuente de dichos sitios web. En lo que se refiere a las técnicas de web scraping que podemos encontrar actualmente, podemos destacar el framework *Scrapy* y las bibliotecas *Urllib, Requests, Selenium* y *BeautifulSoup*. Para el desarrollo de este proyecto la biblioteca que se ha utilizado para realizar el web scraping ha sido *BeautifulSoup*. [75]

- **Power BI**[76]

  El programa Power BI es una herramienta que forma parte del paquete Microsoft Office 365, la cual permite a las empresas recopilar, transformar, analizar y crear visualizaciones de grandes cantidades de datos de diversos orígenes, como bases de datos, archivos Excel, servicios en la nube y muchas más. Con Power BI, los usuarios crean informes interactivos y paneles de control personalizados que proporcionan una visualización clara y concisa de la información, lo que facilita la toma de decisiones basada en datos, el objetivo del Business Intelligence. Además, es una herramienta muy intuitiva y fácil de aprender que además posee múltiples cursos y tutoriales en Internet y dentro de la propia aplicación.

Este programa, a día de hoy, es uno de los más conocidos en el Business Intelligence ya que aporta una variedad de gráficos que utilizan todo tipo de empresas los cuales se segmentan de diferentes maneras y en diferentes pestañas. Power BI dispone de numerosos métodos de visualización, así como tarjetas, tablas, gráficos de barras o incluso mapas geográficos con diferentes diseños disponibles, ver Figura 5.12.

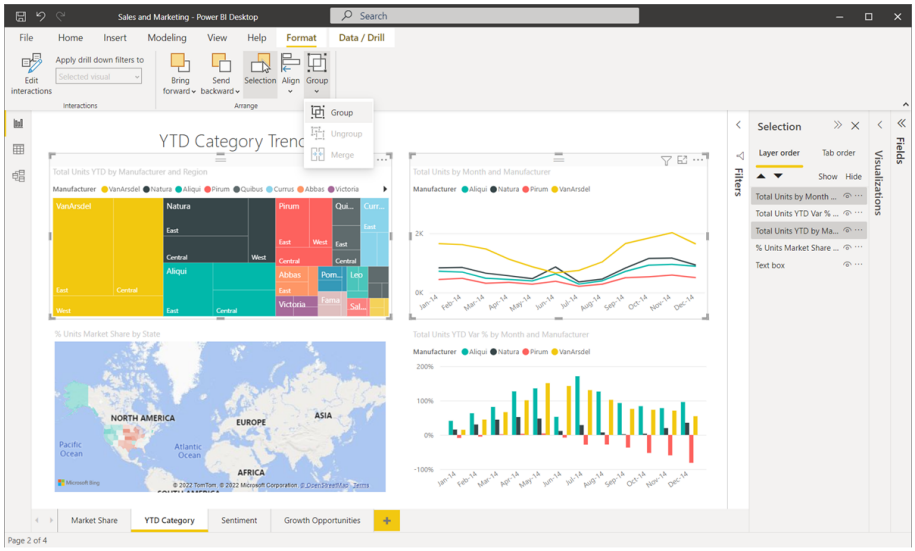


Figura 5.12: Ejemplo de diferentes gráficos de Power BI (extraído de [77])

Una de las grandes ventajas de esta aplicación es la plataforma de conectividad que contiene permitiendo crear y compartir visualizaciones de datos interactivas sin perder la calidad de los informes. Es decir, sin pixelación y de forma ordenada mediante diferentes páginas.

Otra de las ventajas de esta herramienta es que en las nuevas actualizaciones se ha creado una multiplataforma para que de este modo se puedan visualizar o crear informes en distintos dispositivos. Por este motivo, se han creado diferentes versiones a mayores de la versión de escritorio: la versión online que facilita la actualización de datos ya que se actualizan de manera automática y la versión para móviles o tablets que adapta las diferentes pestañas a las características de los dispositivos.

Figura 5.13: Power BI en distintos dispositivos (extraído de [78])

Como se observa en la Figura 5.14, mediante un diagrama de Venn, cada versión ofrece diferentes tareas que ayudan al completo desarrollo de los informes. La versión de escritorio permite transformar

y crear nuevas medidas con los datos obtenidos y elegir el diseño de cada página. Por el contrario, el Servicio Power BI, no ofrece tantos orígenes de datos y tanta variedad en las medidas, pero permite la creación de flujos de datos y el uso compartido. Finalmente, en el área central del diagrama se muestran las tareas que pueden realizar las dos versiones; por ejemplo, la creación de informes y visualizaciones segmentadas por filtros, la creación de marcadores o la seguridad en relación a los datos.[79]

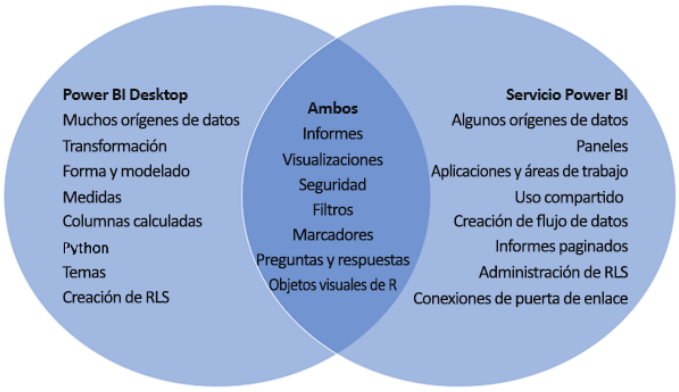


Figura 5.14: Comparación de Power BI Desktop y Online (extraído de [79])

Cabe destacar que ofrece un número abundante de fórmulas que se pueden combinar entre sí para realizar diversos cálculos y transformaciones de datos, como se ha comentado anteriormente. Para ello, dispone de dos lenguajes de programación, M y DAX. Los dos son lenguajes funcionales, pero se utilizan para diferentes propósitos.

- El lenguaje M se usa en Power Query, es un lenguaje de consulta que utiliza una multitud de fuentes de datos las cuales puede transformar y devolver los resultados de la consulta realizada.
- DAX son las siglas de Data Analysis eXpressions y se utiliza para trabajar con datos almacenados en tablas. La mayoría de las funciones se asemejan a las de las hojas de cálculo de Excel, pero DAX contiene muchas más funciones para resumir y dividir escenarios de datos complejos.

En resumen, Power Query (M) se utiliza para consultar fuentes de datos, limpiar y cargar datos. Por el contrario, DAX analiza los datos y crea tablas dinámicas.

Para finalizar, hay que mencionar que Power BI se posiciona líder en el informe Gartner Magic Quadrants, por parte de Gartner, sobre plataformas de análisis y Business Intelligence. [80]

Para este proyecto se ha elegido la versión de Power BI Desktop para la creación de la aplicación por las características de transformación, medidas y columnas calculadas que no dispone el Servicio de Power BI.

5.5. Guías en el diseño de cuadros de mando

Según Edward Tufte, un prestigioso estadístico de la universidad de Yale, el objetivo de un Dashboard es visualizar en un mismo cuadro la información más importante necesaria para lograr uno o más objetivos, pudiendo ser así monitorizada de un vistazo. El mismo Edward Tufte ha escrito un libro llamado "The visual display of quantitative information" en el cual explica las guías para realizar un cuadro de mando de manera correcta.[81]

Teniendo en cuenta las guías descritas en "The visual display of quantitative information", junto con las explicadas por el artículo "The complete guide to data visualization" escrito por la empresa *cluster* y el documento "A Guide to Creating Dashboards People Love to Use" creado por la empresa especilizada en BI llamada *Juice Analytics* se han seguido las siguientes guías para el desarrollo de la aplicación de Business Intelligence, *Fórmula 1 BI*.[82, 83].

5.5.1. Establecer la base de la aplicación

Antes de definir el diseño del cuadro de mando, primero es necesario responder a varias preguntas acerca del propósito de la aplicación, ya que según las respuestas de estas preguntas el diseño puede cambiar de una manera drástica.

- ¿Cómo va a aportar valor el cuadro de mando a la organización?
- ¿Qué tipo de cuadro de mando voy a crear?
- ¿A quién va dirigido el cuadro de mando y cuáles son sus necesidades?
- ¿Cuál es el hilo conductor de la historia de mi cuadro de mando?
- ¿Cuáles son las métricas clave que centrarán a los usuarios en información útil?

Las respuestas a esas preguntas están claras, se quiere crear un cuadro de mando profesional para una empresa de retransmisión de la Fórmula 1. Por tanto, los usuarios a los que está dirigido es a narradores de televisión expertos en Fórmula 1 pero que no tienen porqué ser expertos en análisis de datos o en el uso de aplicaciones informáticas de visualización de datos. Las necesidades fueron descritas en el apartado 4.1 en el que se realiza la definición conceptual del proyecto. La aplicación va a aportar valor respecto al resto de compañías de retransmisión ya que va a proporcionar mucha información de una manera rápida y sencilla con la que pueden amenizar la retransmisión. Las métricas clave son especialmente *scoreAI* y las cinco métricas parciales con las que se crea dicha métrica como se ha explicado en el capítulo 7. Por tanto, la interacción entre el usuario con la aplicación es muy simple. Esto es para facilitar la tarea a los usuarios que no tienen porqué ser expertos en utilizar este tipo de aplicaciones. Por el mismo motivo, no se utilizan gráficos complejos para el usuario que sean difíciles de comprender. Se da mucho peso a las métricas creadas en el capítulo 7 dentro de la aplicación.

5.5.2. Establecer la estructura de la aplicación

Esta etapa se divide en cuatro partes:

- Entregable: ¿En qué formato se entrega el cuadro de mandos?
- Estructura: ¿Cómo se presenta el cuadro de mando para ayudar a los usuarios a comprender el panorama general?
- Principios de diseño: ¿Cuáles son los objetivos fundamentales que guiarán sus decisiones de diseño?
- Funcionalidad: ¿Qué capacidades incluirá el cuadro de mando para ayudar a los usuarios a comprender e interactuar con la información?

Entregable

Para decidir en que formato entregar el cuadro de mando hay que pensar en ciertos factores que pueden influenciarlo. Estos factores pueden definir si elegir como entregable una aplicación online, un excel o una presentación.

1. Puntualidad (Timeliness): ¿Con qué frecuencia se actualizan los datos del cuadro de mando?
2. Valor estético (Aesthetic value): ¿Qué importancia tiene que el cuadro de mandos sea atractivo, o ¿puede ser puramente utilitario?
3. Movilidad (Mobility): ¿Necesita el público acceder a la información sobre la marcha?
4. Conectividad (Connectivity): ¿Necesita el cuadro de mando necesita conectarse a fuentes en tiempo real?

5. Detalle de los datos (Data detail): ¿Ofrecerá el cuadro de mando la posibilidad de desglosar los datos para ver más contexto?
6. Densidad de datos (Data density): ¿Qué nivel de información contendrán las vistas de los datos?
7. Interactividad (Interactivity): ¿Se beneficiará el usuario de la interacción con el cuadro de mandos?
8. Colaboración (Collaboration): ¿Es importante que los usuarios puedan compartir y colaborar en el cuadro de mando?

En la Figura 5.15, *Juice Analytics* especifica que entregables son eficaces ante los factores descritos previamente. Como se observa el único entregable que es eficaz ante la interactividad es la aplicación online. Por este mismo motivo se ha escogido este entregable ya que la interactividad va a ser muy importante para poder tener una aplicación mucho más completa y que muestre mucha más información sin sobrecargar los cuadros de mando. Power BI permite tanto exportar a pdf como publicar los cuadros de mando para poder generar un enlace y poder visualizarlo como una página web, esta segunda opción es la elegida como entregable.

	Paper One-pager	Paper Presentation	Excel	Online app	E-mail / text message	Large screen
Timeliness	-	-	+	+	+	+
Aesthetic	+	+		+	-	+
Mobility	+				+	-
Connectivity	-	-		+	+	+
Data detail	-	+	+	+	-	
Data density	+	+			-	
Interactivity	-	-		+	-	-
Collaboration					+	-

Figura 5.15: Tipos de entregables y su eficacia ante los diversos factores (extraído de [83])

Estructura

La estructura de un cuadro de mando es una de las partes más importantes en su diseño. Por ese motivo debe realizarse cuidadosamente y siguiendo una sería de guías y patrones.

Existen tres tipos generales de estructuras: De flujo, relacionales y agrupaciones.

Una estructura basada en el flujo enfatiza una secuencia de acontecimientos o acciones a lo largo del tiempo. Por otro lado la estructura relacional enfatiza las relaciones entre entidades o medidas. Estas relaciones o conexiones pueden ser matemáticas, geográficas, organizativas o funcionales. Por último, la estructura basada en agrupaciones consiste en agrupar la información relacionada en categorías o en una jerarquía. El simple hecho de agrupar cosas similares aporta cierta lógica y accesibilidad a un cuadro de mando que, de otro modo, sería desordenado.

La estructura que se ha escogido para los cinco cuadros de mando ha sido la basada en agrupaciones, las cuales se explican en detalle en el apartado 8.1.

Principios de diseño

Al igual que con las interfaces web existen principios de diseño centrados en los cuadros de mando.

- Moduralidad: Algunos cuadros de mando se hacen grandes y poco manejables en un esfuerzo por crear una única visión completa de todo un negocio o proceso.
- Revelado gradual: Revela información a medida que el usuario manifiesta interés. En otras palabras, no se debe mostrar al usuario toda la información a la vez.

- Guiar la atención: No basta con poner la información a disposición del usuario; hay que utilizar señales visuales y funcionalidad para atraer al usuario hacia lo más importante.
- Facilidad de aprendizaje: Reducir al mínimo la barrera de entrada para los nuevos usuarios evitando la sobrecarga de funciones, minimizando los clicks para cada tarea y proporcionando descripciones claras y concisas del significado de las cosas.
- Conducir a la acción: Permitir al usuario terminar su tarea rápidamente y comprender la acción que debe tomar en función de los resultados.
- Personalizable: Incorporar flexibilidad para que el cuadro de mando sea relevante para distintos usuarios. La forma más común de permitir a los usuarios personalizar el cuadro de mando, es definir el alcance de los datos mediante filtros.
- Explicación previa a la información: Dejar que los datos hablen por sí solos puede ser una receta para la mala interpretación y la confusión. Por este motivo se crea un menú de ayuda al usuario para explicar las métricas y los gráficos utilizados.

Funcionalidad

Existen diferentes características que deben tener los cuadros de mando. El primer grupo de características son las básicas que debe tener en cuenta cualquier cuadro de mando. El segundo grupo son las funciones avanzadas, si se quiere que el usuario posea mayor control sobre la aplicación.

- Características básicas:
  - Desglose: Capacidad de pasar de una métrica o vista resumida a detalles adicionales que proporciona más contexto o desglose de la información.
  - Filtros: Permiten a los usuarios definir el alcance de los datos en el cuadro de mando para reflejar sus necesidades. Los filtros pueden ser globales (refinar el alcance para todo el cuadro de mando) o locales (filtro para un único gráfico, métrica o vista específicos).
  - Comparación: Capacidad de ver dos o más subconjuntos de datos uno al lado del otro.
  - Alertas: Resaltar información en función de criterios predefinidos. La alerta puede activarse cuando una métrica supera un umbral determinado.
  - Exportar / imprimir: Ofrecer a los usuarios la posibilidad de extraer información de un cuadro de mando. Exportar a formatos que permitan a los usuarios hacer más con los datos, como Excel y CSV en lugar de PDF.

  Power BI ayuda en la tarea de añadir estas características básicas gracias a los filtros de segmentación de datos, que mediante la opción de interacciones pueden elegirse a que gráficos o medidas afecta y los filtros dentro de un mismo gráfico. También posee un botón en las tablas para descargar los datos en formato CSV y mediante tooltips o también llamado información sobre herramientas, permite realizar el desglose para mostrar más detalles sobre una métrica.
- Características avanzadas:
  - Resumen textual: Descripción textual generada automáticamente de la información clave del cuadro de mando. Esto es tan simple como una frase que incluya un par de datos importantes.
  - Etiquetas: Posibilidad de que los usuarios identifiquen en el cuadro de mandos aspectos que les resulten importantes.
  - Anotaciones: Permite a los usuarios añadir comentarios a cifras o gráficos concretos.
  - Guardar cambios: Cuanto más configura un usuario un cuadro de mando según sus necesidades particulares, más importante es permitirles guardar lo que ha creado.

- Visualizaciones avanzadas: Si resulta útil mostrar datos más complejos en el cuadro de mandos, una variedad de visualizaciones avanzadas puede ayudar a hacerlos digeribles. Algunos tipos de visualización a tener en cuenta son el mapa geográfico, el mapa de árbol, el diagrama de red, los gráficos de radar, los gráficos de dispersión y los gráficos de burbujas. Pero hay que tener cuidado, el uso incorrecto de visualizaciones complejas puede hacer que su audiencia se sienta perdida y confusa.

Power BI dispone de múltiples gráficas sencillas por defecto, para incluir ciertas visualizaciones avanzadas es necesario descargarlas para poder utilizarlas como es el caso del gráfico de radar muy útil para representar varias variables numéricas.

5.5.3. Diseñar el cuadro de mando

La última etapa consiste en diseñar el cuadro de mando la cual se divide en dos subfases, el diseño de la interfaz y la información mostrada.

Diseño de la interfaz

El diseño de la interfaz posee tres elementos clave, la organización de la página, los colores elegidos y la tipografía escogida.

En la organización de la página hay que tener en cuenta como las personas escanean una página web y en especial un cuadro de mando. Para ello hay diversos patrones de diseño como el patrón F, muy común en el diseño de páginas web y para cuadros de mando especialmente el patrón Z. Ambos patrones son similares y tienen en común que la parte superior izquierda es la que primero se observa y después se observa la fila superior de izquierda a derecha como se ve en la Figura 5.16. Los patrones F y Z también coinciden en que la zona inferior derecha es en la que menos se fijan los usuarios. Por esos motivos, los menús para cambiar entre cuadros de mando y los filtros generales de los cuadros de mando se han situado en la parte superior ya que es lo primero que observa el usuario y están en una zona de rápido reconocimiento. Siguiendo el patrón Z en los dashboards la información más importante se representa en la parte superior y a medida que se va bajando la información es más detallada utilizando tablas. [84]

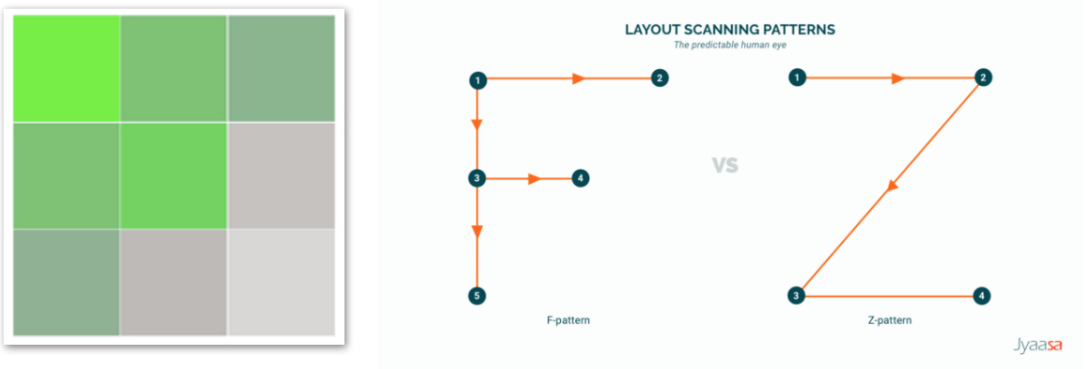


Figura 5.16: Posiciones que primero se observan en un dashboard junto con los patrones F y Z (extraído de [84])

Otro sistema que utilizan los diseñadores para crear cuadros de mando, es un sistema de cuadrícula, basado en columnas de la misma anchura, para que los elementos queden alineados. Esto se ha tenido en cuenta en los cuadros de mando Piloto, Escudería y Circuito creando un sistema de cuadrícula basado en tres columnas de la misma anchura y en el dashboard de 1 vs 1 con dos columnas. Este diseño genera coherencia y orden en los cuadros de mando y permite al usuario analizar de forma más clara los gráficos que si estuvieran descolocados.

Por último es necesario no sobrecargar los cuadros de mandos generando espacios en blanco. En *Fórmula 1 BI* se ha conseguido mediante la agrupación de los gráficos en recuadros, generando una sensación de aislamiento entre gráficos, que permite diferenciar cada sección y no sobrecargar los cuadros de mando.

El segundo apartado, después de la organización de página, es la elección del color, para cada cuadro de mando se ha escogido un fondo con un color diferente para notar más la diferencia de que se tratan de cuadros de mando diferentes. Eligiendo un color azul para el cuadro de mando Piloto, plateado para la escudería, marrón para Circuito, color dorado para las temporadas y azul y fucsia para el 1 vs 1. El color de los gráficos corresponde con el color del fondo de cada cuadro de mando para tener una armonía con los colores. La explicación detallada de la utilización del color en cada gráfico concreto es en el apartado 8.1.

En cuanto a la tipografía elegida, puede elegirse una fuente serif o sans-serif. Los tipos de letra serif, como Times New Roman, tienen pequeños remates decorativos en los extremos de las letras, lo que les da una apariencia más clásica y elegante. Por otro lado, las fuentes sans-serif, como Arial, no tienen estos remates y presentan líneas limpias y rectas, lo que les da un aspecto más moderno y minimalista. La fuente utilizada para toda la aplicación es Segoe UI siendo una fuente sans-serif siendo la utilizada por defecto por Power BI.

Información mostrada

Un cuadro de mando tiene que contar una historia con datos para ello es necesario crear gráficos y tablas que resalten información y sean fáciles de interpretar. Para ello es muy importante la elección de los gráficos que van a ser utilizados, hay múltiples gráficos cada uno para un propósito concreto, para elegir utilizar un gráfico u otro, primero es necesario conocer el tipo de los datos que van a ser representados, explicados en el apartado 6.2.2, pueden ser categóricos (ordinales o nominales) y numéricos (discretos y continuos).

Existen guías y esquemas de qué gráficos utilizar teniendo en cuenta el objetivo con el que se quiere mostrar y el tipo de datos que se van a representar como se indica en la Figura 5.17. El motivo de la elección de cada gráfico concreto se explica en el apartado 8.1.

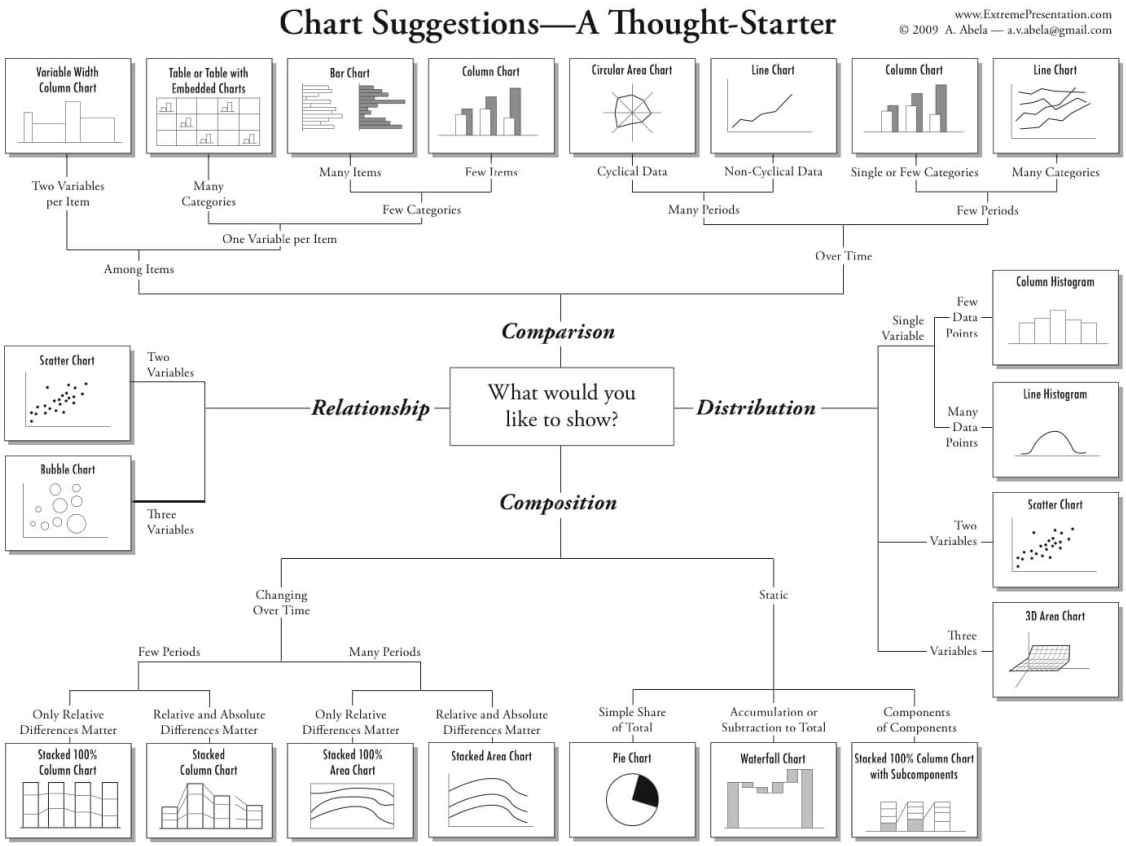


Figura 5.17: Tipos de gráficos y cuando deben utilizarse (extraído de [85])

Existen guías en el diseño de gráficos y son las siguientes:

1. Aumentar la relación datos/tinta: Es uno de los mandamientos de Edward Tufte el famoso estadístico de la universidad de Yale mencionado previamente. Reducir los elementos decorativos que no aporten información. Evitar los efectos tridimensionales ya que no añaden nada de valor al gráfico. Aumentar la relación datos/tinta haciendo que cada píxel cuente una historia sobre sus datos.

2. Maximizar el contraste: Maximizar el contraste entre los datos y el fondo. Por este motivo todos los gráficos de todos los cuadros de mando tienen un fondo blanco y los datos se representan con un color que genera contraste con el fondo.

3. Etiquetas legibles: Siempre que sea posible, evitar las etiquetas giradas porque son difíciles de leer y desvían la atención del usuario.

4. La repetición es mala: No es necesario tener una leyenda y un título para los gráficos de una sola serie.

5. Evitar el suavizado y el 3D. Evitar añadir una función de suavizado a las líneas, da la impresión de que hay puntos de datos que no existen. Del mismo modo, los efectos 3D no aportan ningún valor al gráfico. Por estos motivos no se realiza ni suavizado ni se ha incluido ningún elemento 3D.

6. Ordenar para comprender: Añadir estructura y claridad al gráfico ordenándolo por una métrica de interés. Esto se ha incluido en *Fórmula 1 BI* especialmente para las tablas.

Para el estilo de tabla existen estos consejos:

- Eliminar las líneas de la cuadrícula.
- Utilizar líneas o espacios en blanco para separar áreas conceptualmente diferentes.
- Mostrar la menor cantidad de números posible para satisfacer las necesidades de la tabla.
- Utilizar un espaciado coherente entre columnas y filas para crear un ritmo horizontal y vertical.

Capítulo 6

Proceso de ETL: Implementar el Almacén de datos

En este capítulo se realiza la implementación del proceso ETL. Para ello se dispone de una sección para cada una de las siguientes etapas:

- Extracción donde se explica el proceso de ingesta de los datos desde los orígenes.
- Transformación en la que se aplica una serie de reglas para garantizar la calidad de los datos.
- Carga de los datos en la que los datos ya transformados pasan al entorno de explotación para poder realizar análisis posteriores.

6.1. Extracción de datos desde los orígenes

El proceso de extracción de datos es la etapa que determina cuáles son las fuentes de datos que se van a utilizar, abarcando desde su descarga del sistema de origen a su almacenamiento en un formato preparado para ser transformado adecuadamente en la siguiente etapa.

6.1.1. Descripción de las fuentes de datos

Como ya se ha indicado anteriormente se va a utilizar como principal fuente los datos procesados y proporcionados por Rohan Rao y disponibles en Kaggle [71], constan de 14 ficheros CSV y pueden descargarse manualmente desde la web de Kaggle. Estos datos han sido tratados por Rohan Rao y extraídos originalmente de Ergast Developer API la cuál es un servicio web que proporciona un registro histórico de datos de carreras de motor con fines no comerciales. La API proporciona datos de Fórmula 1 desde el comienzo de los campeonatos mundiales en 1950. El uso de esta API puede utilizarse para aplicaciones y servicios personales, no comerciales, incluidos fines educativos y de investigación. [86]

Como fuentes adicionales para conseguir datos no disponibles como la imagen o la longitud del circuito se utiliza la Wikipedia y se accede a la información mediante web scraping con Python.

Por último, para obtener el país de origen a partir de la nacionalidad se obtiene mediante un dataset de GitHub.

El esquema que describe las fuentes y el modo de acceso a cada una de ellas se observa en la Figura 6.1.

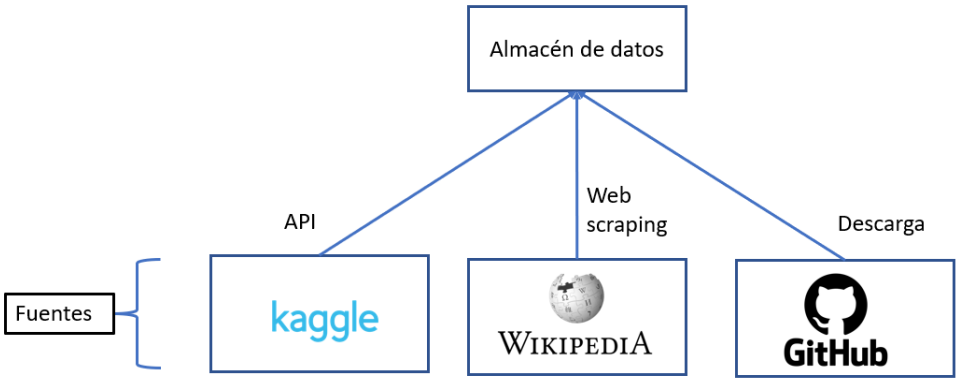


Figura 6.1: Fuentes y el modo de acceso a cada una de ellas (elaboración propia)

6.1.2. Conexión a los orígenes de datos

API de Kaggle

La tarea de descargar los datos desde los orígenes puede convertirse en algo tediosa si queremos tener los datos actualizados regularmente ya que Rohan Rao indica que se actualizan los ficheros cada semana. Por este motivo se debe intentar automatizar la tarea de extracción de datos. Para ello, se ha realizado un pequeño script de Python que descarga en local los archivos CSV.

```
import kaggle
kaggle.api.authenticate()
kaggle.api.dataset_download_files("rohanrao/formula-1-world-championship-1950-2020",
    path="./datos/", unzip=True, quiet=False)
```

Con la librería *kaggle* de Python se accede a la API de Kaggle, se debe proporcionar la URL donde se encuentran los datos y la carpeta destino donde quieren ser guardados. Ejecutando el anterior script se descargan todos los archivos CSV mencionandos anteriormente. Para poder utilizar la API de Kaggle es necesario descargarse una API-token desde el perfil de la web y situarlo en la carpeta "C:\Users\username\.kaggle"

Web scraping en la Wikipedia

Web scraping es una técnica utilizada para acceder a información de sitios web. Esta técnica consiste en simular la navegación de un usuario real por la web a través del código fuente de dichos sitios web. Para ello, se han utilizado las librerías *Beautiful Soup* [75] y *requests* [87] de Python con las cuáles se ha obtenido mediante web scraping el enlace a la imagen de los pilotos, escuderías y circuitos en la Wikipedia. Esto ha sido posible gracias a que en algunas dimensiones se dispone de una variable llamada `url` la cuál es el enlace a la Wikipedia de ese piloto, escudería o circuito. El siguiente script es un ejemplo de web scraping con Python para obtener el enlace de la imagen del piloto en la Wikipedia utilizando *requests* y *Beautiful Soup*.

```
def get_image_url(wiki_url,delete_word):
    '''
    Funcion que obtiene la url de la imagen principal de un piloto, escuderia o
    circuito
    en la Wikipedia a partir del enlace a la pagina de la Wikipedia de ese piloto,
    escuderia o circuito.
    Argumentos:
        wiki_url: Enlace a la pagina de la Wikipedia de la que queremos extraer su
    imagen.
        delete_word: Palabra que no puede contener el enlace de la imagen deseada.
    Devuelve:
        Enlace de la imagen de la pagina de la Wikipedia pasada por argumento

    '''
    print(wiki_url)
```

```
    response = requests.get(wiki_url)
    soup = BeautifulSoup(response.text, 'html.parser')
    infobox = soup.find('table', {'class': 'infobox'})
    if infobox:
        images = infobox.find_all('img')

        if (len(images)>=1) and (delete_word not in images[0]['src'].lower()):
            print('https:' + images[0]['src'])
            return 'https:' + images[0]['src']
        elif len(images)>=2:
            # Si el primer enlace tiene la palabra descrita en 'delete_word' se escoge la segunda imagen
            if (images[1]):
                print('https:' + images[1]['src'])
                return 'https:' + images[1]['src']
    return None
```

Descarga de dataset de nacionalidades de GitHub

Por último para obtener la información del país de origen a partir de la nacionalidad se ha obtenido mediante un dataset de GitHub el cuál relacionaba los países con sus nacionalidades. Con el siguiente script de Python gracias a la función *read_csv* de la librería *Pandas* [88] se accede de manera sencilla a la información del dataset: [89]

```
# Se cargan los datos de países de un dataset de GitHub para obtener el país apartir de la nacionalidad
url = 'https://raw.githubusercontent.com/knowitall/chunkedextractor/master/src/main/resources/edu/knowitall/chunkedextractor/demonyms.csv'
countriesDataset = pd.read_csv(url, on_bad_lines='skip')
# Se renombran de las columnas para que coincidan con los datos que disponemos
countriesDataset.columns = ['nationality', 'country']
```

6.1.3. Tipo e identificador de cada dato en origen

- **circuits.csv**
  - circuitId: Variable numérica discreta.
  - circuitRef: Variable categórica nominal.
  - name: Variable categórica nominal.
  - location: Variable categórica nominal.
  - country: Variable categórica nominal.
  - lat: Variable numérica continua.
  - lng: Variable numérica continua.
  - alt: Variable numérica continua.
  - url: Variable categórica nominal.
- **constructors.csv**:
  - constructorId: Variable numérica discreta.
  - constructorRef: Variable categórica nominal.
  - name: Variable categórica nominal.
  - nationality: Variable categórica nominal.
  - url: Variable categórica nominal.
- **drivers.csv**:

- driverId: Variable numérica discreta.
- driverRef: Variable categórica nominal.
- number: Variable numérica discreta.
- code: Variable categórica nominal.
- forename: Variable categórica nominal.
- surname: Variable categórica nominal.
- dob: Variable numérica continua de fecha.
- nationality: Variable categórica nominal.
- url: Variable categórica nominal.

- **lap_times.csv**

  - raceId: Variable numérica discreta.
  - driverId: Variable numérica discreta.
  - lap: Variable numérica discreta.
  - position: Variable numérica discreta.
  - time: Variable numérica continua de tiempo.
  - milliseconds: Variable numérica discreta.

- **pit_stops.csv**:

  - raceId: Variable numérica discreta.
  - driverId: Variable numérica discreta.
  - stop: Variable numérica discreta.
  - lap: Variable numérica discreta.
  - time: Variable numérica continua de tiempo.
  - duration: Variable numérica continua.
  - milliseconds:Variable numérica discreta.

- **races.csv**:

  - raceId: Variable numérica discreta.
  - year: Variable numérica discreta.
  - round: Variable numérica discreta.
  - circuitId: Variable numérica discreta.
  - name: Variable categórica nominal.
  - date: Variable numérica continua de fecha.
  - time: Variable numérica continua de tiempo.
  - url: Variable categórica nominal.
  - fp1_date: Variable numérica continua de fecha.
  - fp1_time: Variable numérica continua de tiempo.
  - fp2_date: Variable numérica continua de fecha.
  - fp2_time: Variable numérica continua de tiempo.
  - fp3_date: Variable numérica continua de fecha.
  - fp3_time: Variable numérica continua de tiempo.

- quali_date: Variable numérica continua de fecha.
- quali_time: Variable numérica continua de tiempo.
- sprint_date: Variable numérica continua de fecha.
- sprint_time: Variable numérica continua de tiempo.

- **results.csv**:

  - resultId: Variable numérica discreta.
  - raceId: Variable numérica discreta.
  - driverId: Variable numérica discreta.
  - constructorId: Variable numérica discreta.
  - number: Variable numérica discreta.
  - grid: Variable numérica discreta.
  - position: Variable numérica discreta.
  - positionText: Variable categórica ordinal.
  - positionOrder: Variable numérica discreta.
  - points: Variable numérica discreta.
  - laps: Variable numérica discreta.
  - time: Variable numérica continua de tiempo.
  - milliseconds: Variable numérica discreta.
  - fastestLap:Variable numérica discreta.
  - rank: Variable numérica discreta.
  - fastestLapTime: Variable numérica continua de tiempo.
  - fastestLapSpeed: Variable numérica continua.
  - statusId: Variable numérica discreta.

- **status.csv**:

  - statusId: Variable numérica discreta.
  - status: Variable categórica nominal.

6.1.4. Exploración de los datos

Esta etapa de exploración de datos se utiliza el lenguaje Python, la librería para el tratamiento de datos *Pandas* [88] y la librería de visualización de datos *matplotlib* [90].

Valores ausentes (missing values)

Como se observó en la etapa de Profiling de los datos 4.4 existían varias variables que a veces tomaban como valor el carácter especial "\N", el cuál lo utiliza para indicar un valor especial, Power BI lo detecta como un texto y provoca que transforme columnas numéricas que tienen "\N" como columnas de texto.

Muchos de ellos tienen que ver con abandonos de pilotos en un GP, cuando esto ocurre la columna que indica la posición de la carrera se indica como "\N" porque no ha llegado a finalizarla pero otras veces simplemente es porque no se dispone de la información y se trata de un valor nulo.

Es por ello que mediante Python al leer cada fichero se transforma a valor nulo si aparece ese carácter.

Tras ello se procede a contabilizar el número de valores ausentes en cada tabla destino con las columnas deseadas, antes de mostrar todos los valores ausentes por columnas primero se realiza por tablas enteras así se descartan las tablas con 0 o pocos valores ausentes y centrarse en los casos en los que tiene muchos valores ausentes, por tanto para obtener el número de valores ausentes por cada tabla se realiza este script:

```
for i in range(len(tables)):
    print("Valores ausentes en "
          +tablesNames[i]+": "+str(tables[i].isna().sum().sum()))
```

Obteniéndose el siguiente resultado:

```
Valores ausentes en circuits.csv: 2
Valores ausentes en constructors.csv: 0
Valores ausentes en drivers.csv: 1560
Valores ausentes en lapTimes.csv: 0
Valores ausentes en pitStops.csv: 0
Valores ausentes en races.csv: 11257
Valores ausentes en results.csv: 121796
Valores ausentes en status.csv: 0
```

Figura 6.2: Missing values por cada archivo de origen

En los archivos circuits.csv, constructors.csv, lapTimes.csv, pitStops.csv y status.csv no encontramos grandes cantidades de nulos en sus variables como se observa en la Figura 6.2

En el archivo drivers.csv disponemos con dos variables con altos porcentaje de nulos (number y code) ambas superan el 88 %, por otro lado en el archivo races.csv todas las variables de tiempo y fecha de clasificación superan el 95 % de nulos ya que han sido recientemente añadida a las ultimas carreras únicamente. Por último, en results.csv existen varias variables como el tiempo que ha tardado en realizar la carrera (time) o la máxima velocidad media por vuelta (fastestLapSpeed) que superan el 70 % de nulos.

Los datos que disponemos son datos desde 1950 hasta la actualidad, que los archivos tengan 0 % de nulos en las variables no significa que dispongamos de todos los datos completos. Por ello, es necesario realizar un análisis más a fondo de los archivos que disponemos.

Gracias a este profundo análisis, se ha encontrado que en el archivo pitStops.csv pese a tener todas las variables sin datos nulos los datos están incompletos ya que sólo disponemos información a partir de las carreras con identificador 841. Por lo tanto, existen 840 carreras en las que no se disponen los datos de pitStops y hay que tenerlo en cuenta en las posteriores visualizaciones.

Valores atípicos (outliers)

Para identificar posibles valores atípicos (outliers) se ha decidido realizar un diagrama de cajas (boxplot) para cada variable numérica para observar si existen valores atípicos.

Viendo la Figura 6.3 se observa los diagramas de caja de las variables `milliseconds` de los archivos pit_stops.csv y lap_times.csv. Se observan cientos de outliers por cada variable de tiempos de parada o de vueltas que superan el millón de milisegundos los cuáles son 16.67 minutos lo que claramente son errores de medición ya que los pitstops duran menos de un minuto y las vueltas suelen durar siempre menos de cuatro minutos.

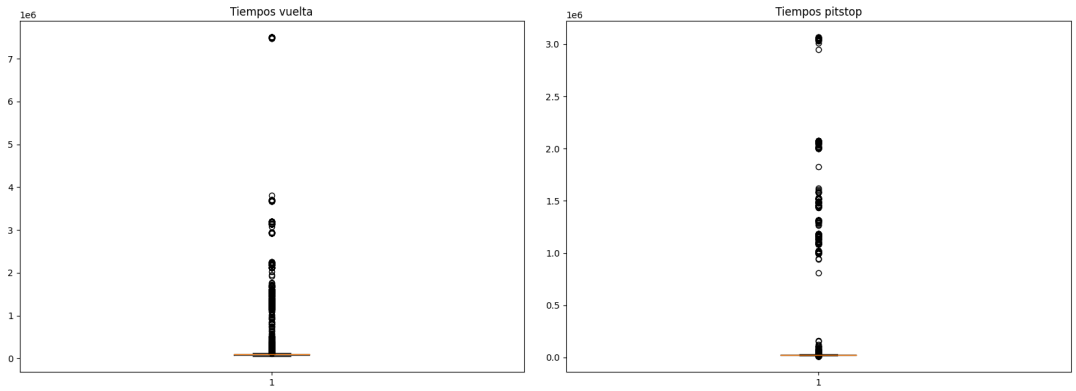


Figura 6.3: Boxplots de las variables milliseconds de los archivos de pitstops y de lapTimes (elaboración propia)

6.2. Transformación de los datos

La etapa de transformación de datos es la encargada de convertir los datos en crudo a un formato requerido en la aplicación de destino. Dentro de esta etapa se incluyen tareas como tratar los valores ausentes, los valores atípicos o outliers, discretizar ciertos atributos,crear nuevas variables...

6.2.1. Descripción del destino de los datos

El destino de los datos son archivos CSV actualizados periódicamente a los que se accederán fácilmente mediante la herramienta Power BI para realizar los posteriores cuadros de mando.

6.2.2. Tipo e identificador de cada dato en destino

- **Tabla Circuito**
  - circuitId: Variable numérica discreta.
  - circuitRef: Variable categórica nominal.
  - name: Variable categórica nominal.
  - location: Variable categórica nominal.
  - country: Variable categórica nominal.
  - lat: Variable numérica continua.
  - lng: Variable numérica continua.
  - alt: Variable numérica continua.
  - url: Variable categórica nominal.
  - image_url: Variable categórica nominal.
- **Tabla Escudería**:
  - constructorId: Variable numérica discreta.
  - constructorRef: Variable categórica nominal.
  - name: Variable categórica nominal.
  - nationality: Variable categórica nominal.
  - url: Variable categórica nominal.
  - image_url: Variable categórica nominal.
  - country: Variable categórica nominal.

- **Tabla Piloto**:
  - driverId: Variable numérica discreta.
  - driverRef: Variable categórica nominal.
  - number: Variable categórica ordinal.
  - code: Variable categórica nominal.
  - forename: Variable categórica nominal.
  - surname: Variable categórica nominal.
  - dob: Variable numérica continua de fecha.
  - nationality: Variable categórica nominal.
  - url: Variable categórica nominal.
  - country: Variable categórica nominal.
  - complete_name: Variable categórica nominal.
  - image_url: Variable categórica nominal.
- **Tabla CarreraDescripcion**:
  - raceId: Variable numérica discreta.
  - year: Variable numérica discreta.
  - round: Variable numérica discreta.
  - circuitId: Variable numérica discreta.
  - name: Variable categórica nominal.
  - date: Variable numérica continua de fecha.
  - time: Variable numérica continua de tiempo.
  - url: Variable categórica nominal.
  - typeCircuitCleaned: Variable categórica nominal.
  - weatherCleaned: Variable categórica nominal.
  - circuitLengthCleaned: Variable numérica continua.
- **CarreraResultado**:
  - resultId: Variable numérica discreta.
  - raceId: Variable numérica discreta.
  - driverId: Variable numérica discreta.
  - constructorId: Variable numérica discreta.
  - number: Variable numérica discreta.
  - grid: Variable numérica discreta.
  - positionOrder: Variable numérica discreta.
  - points: Variable numérica discreta.
  - laps: Variable numérica discreta.
  - milliseconds: Variable numérica discreta.
  - rankFastLap: Variable numérica discreta.
  - fastestLapTime: Variable numérica continua de tiempo.
  - fastestLapSpeed: Variable numérica continua.
  - status: Variable categórica nominal.

- status_fault: Variable categórica nominal.
- fastestLapTimeMilliseconds: Variable numérica continua.
- time: Variable numérica de tiempo.
- Victoria: Variable binaria.
- Podio: Variable binaria.
- Pole: Variable binaria.

- **TemporadaPiloto**:

  - year: Variable numérica discreta.
  - driverRef: Variable categórica nominal.
  - constructorRef: Variable categórica nominal.
  - points_sum: Variable numérica continua.
  - ranking: Variable numérica discreta.
  - *scoreAI*: Variable numérica continua con rango(0-100).
  - KPI_compañero_equipo: Variable numérica continua con rango(0-100).
  - KPI_lluvia: Variable numérica continua con rango(0-100).
  - KPI_evitar_accidentes: Variable numérica continua con rango(0-100).
  - KPI_resultados_generales: Variable numérica continua con rango(0-100).
  - KPI_regularidad: Variable numérica continua con rango(0-100).
  - CampeonatoPiloto: Variable binaria.

- **TemporadaEscudería**:

  - year: Variable numérica discreta.
  - constructorRef: Variable categórica nominal.
  - N_GPs: Variable numérica discreta.
  - points_sum: Variable numérica continua.
  - ranking: Variable numérica discreta.
  - CampeonatoConstructores: Variable binaria.

- **Year**:

  - year: Variable numérica discreta.

6.2.3. Elaboración del Mapa Lógico: Relación dato origen – dato destino

La relación entre los campos destino y los de origen (junto al fichero de origen del que provienen) en el Almacén de datos, aparece expuesta en las Tablas 6.1-6.5.

Tabla Circuito (dimensión)		
Nombre campo destino	Fichero origen	Nombre campo origen
circuitId	circuits.csv	circuitId
circuitRef	circuits.csv	circuitRef
name	circuits.csv	name
location	circuits.csv	location
country	circuits.csv	country
lat	circuits.csv	lat
lng	circuits.csv	lng
alt	circuits.csv	alt
url	circuits.csv	url
image_url	Calculado utilizando url de circuits.csv	

Tabla 6.1: Relación dato origen – dato destino de la tabla Circuito

Tabla Escudería (dimensión)		
Nombre campo destino	Fichero origen	Nombre campo origen
constructorId	constructors.csv	constructorsId
constructorRef	constructors.csv	constructorRef
name	constructors.csv	name
nationality	constructors.csv	nationality
url	constructors.csv	url
country	Calculado utilizando nationality de constructors.csv	
image_url	Calculado utilizando url de constructors.csv	

Tabla 6.2: Relación dato origen – dato destino de la tabla Escudería

Tabla Piloto (dimensión)		
Nombre campo destino	Fichero origen	Nombre campo origen
driverId	drivers.csv	driverId
driverRef	drivers.csv	driverRef
number	drivers.csv	number
code	drivers.csv	code
forename	drivers.csv	forename
surname	drivers.csv	surname
dob	drivers.csv	dob
nationality	drivers.csv	nationality
url	drivers.csv	url
country	Calculado utilizando nationality de drivers.csv	
complete_name	Calculado utilizando forename y surname de drivers.csv	
image_url	Calculado utilizando url de drivers.csv	

Tabla 6.3: Relación dato origen – dato destino de la tabla Piloto

Tabla CarreraDescripcion (dimensión)		
Nombre campo destino	Fichero origen	Nombre campo origen
raceId	races.csv	raceId
year	races.csv	year
round	races.csv	round
circuitId	races.csv	circuitId
name	races.csv	name
date	races.csv	date
time	races.csv	time
url	races.csv	url
typeCircuitCleaned	Calculado utilizando url de races.csv	
weatherCleaned	Calculado utilizando url de races.csv	
circuitLengthCleaned	Calculado utilizando url de races.csv	

Tabla 6.4: Relación dato origen – dato destino de la tabla CarreraDescripcion

Tabla CarreraResultado (hechos)		
Nombre campo destino	Fichero origen	Nombre campo origen
resultId	results.csv	resultId
raceId	results.csv	raceId
driverId	results.csv	driverId
constructorId	results.csv	constructorId
number	results.csv	number
grid	results.csv	grid
positionOrder	results.csv	positionOrder
points	results.csv	points
laps	results.csv	laps
time	results.csv	time
milliseconds	results.csv	milliseconds
rank	results.csv	rank
fastestLapTime	results.csv	fastestLapTime
fastestLapSpeed	results.csv	fastestLapTime
status	status.csv	status
status_fault	Calculado utilizando status de status.csv	
fastestLapTimeMilliseconds	Calculado utilizando fastestLapTime de results.csv	
Victoria	Calculado utilizando positionOrder de results.csv	
Podio	Calculado utilizando positionOrder de results.csv	
Pole	Calculado utilizando grid de results.csv	

Tabla 6.5: Relación dato origen – dato destino de la tabla CarreraResultado

Tabla TemporadaPiloto (hechos)		
Nombre campo destino	Fichero origen	Nombre campo origen
year	Calculado utilizando year de races.csv	
driverRef	Calculado utilizando driverRef de drivers.csv	
constructorRef	Calculado utilizando constructorRef de constructor.csv	
ranking	Calculado utilizando varias variables de results.csv	
points_sum	Calculado utilizando points de results.csv	
scoreAI	Calculado utilizando múltiples variables	
KPI_vs_compañero_equipo	Calculado utilizando múltiples variables	
KPI_lluvia	Calculado utilizando múltiples variables	
KPI_evitar_accidentes	Calculado utilizando múltiples variables	
KPI_resultados_generales	Calculado utilizando múltiples variables	
KPI_regularidad	Calculado utilizando múltiples variables	
campeonatoPiloto	Calculado utilizando múltiples variables	

Tabla 6.6: Relación dato origen – dato destino de la tabla TemporadaPiloto

Tabla TemporadaEscuderia (hechos)		
Nombre campo destino	Fichero origen	Nombre campo origen
year	Calculado utilizando year de races.csv	
constructorRef	Calculado utilizando constructorRef de constructor.csv	
N_GPS	Calculado utilizando variables de results.csv	
points_sum	Calculado utilizando points de results.csv	
ranking	Calculado utilizando varias variables de results.csv	
CampeonatoEscuderia	Calculado utilizando múltiples variables	

Tabla 6.7: Relación dato origen – dato destino de la tabla TemporadaEscuderia

6.2.4. Descripción Tareas de transformación

Tratamiento de valores ausentes(Missing values)

La primera transformación que se realiza es sustituir el carácter "\N" por valor nulo, esto se realiza en la lectura de los datos indicándolo con el argumento na_values = r'\N'como se observa en el siguiente script de lectura de drivers.csv.

```
drivers = pd.read_csv(f'{fpath}drivers.csv',  na_values=r'\N')
```

Para los valores ausentes no se realiza ningún tipo de imputación en el destino.

Selección de columnas

Después, lo siguiente que debe realizarse en la etapa de transformación, es seleccionar únicamente las columnas que se vayan a utilizar en el destino siguiendo el mapa lógico realizado en el apartado 6.2.3.

Tratamiento de valores atípicos(Outliers)

En la etapa de exploración del apartado 6.1.4 se observó que las variables `milliseconds` de los archivos de pitStops.csv y lapTimes.csv mostraban cientos de outliers provocados por errores de medición. Al haber tantos outliers en estas variables se ha tomado la decisión de eliminar las tablas de TiempoVuelta y TiempoPitStop del destino.

Creación nuevas variables

Se han creado nuevas variables a partir de los datos iniciales las cuáles enriquecen la información disponible entre las cuáles destaca:

- `complete_name`: Nombre completo de un piloto. Se crea juntando las variables `forename` (nombre) y `surname` (apellido). Corresponde a la tabla Piloto.

- `image_url`: Enlace a la imagen de Wikipedia. se obtiene mediante web scraping. Se ha creado en la tabla Piloto, Circuito y Escudería.

- `country`: País de origen. Se obtiene a partir de la variable `nationality` (nacionalidad) y el dataset que relaciona países y nacionalidades de GitHub. [89] Se ha creado en la tabla Piloto y Escudería.

- `status_fault`: Variable que indica quién fue el causante de un abandono de una manera resumida. Indica si fue un error humano o un error mecánico. Este información ha sido obtenida mediante la variable `status` la cuál posee información sobre el tipo de abandono del piloto y poseía 140 valores diferentes. Corresponde a la tabla CarreraResultado. Puede tomar tres valores "piloto", "escuderia" u "otro".

- `fastestLapTimeMilliseconds`: Tiempo de la vuelta rápida de cada piloto en milisegundos. Para ello se ha transformado la variable `fastestLapTime` que está en formato mm:ss.fff a milisegundos. Corresponde a la tabla CarreraResultado.

- `Victoria`: Variable que indica si un piloto ha finalizado en primera posición un carrera. Corresponde a la tabla CarreraResultado.

- `Podio`: Variable que indica si un piloto ha finalizado en el podio de una carrera, es decir, ha finalizado entre los tres primeros. Corresponde a la tabla CarreraResultado.

- `Pole`: Variable que indica si un piloto ha comenzado primero en una carrera, es decir, ha finalizado primero en la clasificación. Corresponde a la tabla CarreraResultado.

- `typeCircuitCleaned`: Tipo de circuito. Se obtiene mediante web scraping y puede tener tres valores diferentes: "Circuito urbano", "Circuito permanente" u "otro". Corresponde a la tabla CarreraDescripcion y no a la tabla Circuito por si el circuito ha cambiado de tipo durante los años.

- `weatherCleaned`: Meteorología de una carrera. Se obtiene mediante web scraping y puede tener cuatro valores diferentes: "Lluvia", "Nublado", "Soleado" u "Otro". Corresponde a la tabla CarreraDescripcion.

- `circuitLengthCleaned`: Longitud del circuito en kilómetros. Se obtiene mediante web scraping. Corresponde a la tabla CarreraDescripcion y no a la tabla Circuito por si el circuito ha cambiado de longitud durante los años algo que suele ser muy común.

6.3. Carga de datos

La etapa de carga es el proceso en el cual los datos procedentes de la transformación pasan al sistema de destino, donde son utilizados para visualización y análisis.

6.3.1. Tareas de creación del destino de los datos

En esta etapa se crea el almacén de datos, siguiendo el esquema planificado previamente, donde posteriormente se ubican los datos resultantes tras la ejecución de la ETL, descrita en el siguiente apartado.

En este proyecto, el almacén de datos está formado con ficheros CSV que se actualizan en cada ejecución de la ETL, estos respetan el esquema planificado. Cada fichero almacena una tabla y después en Power BI se terminan de ajustar las relaciones entre tablas para que tengan el mismo esquema que el diseñado previamente, (ver Figura 5.4).

6.3.2. Tareas de carga de datos en cada destino de los datos

Cada vez que se ejecuta la ETL se descargan los nuevos datos del origen y se leen los que ya están almacenados en destino. Se transforma los datos nuevos y se les junta con los datos previamente almacenados en destino. Finalmente con este script se sobrescriben todos los archivos CSV que forman el almacén de datos del destino.

```
# Script para guardar todos los archivos en el almacen de datos

tables = [Circuito,Escuderia,Piloto,CarreraDescripcion,CarreraResultado,Year,
    temporadaPiloto,temporadaEscuderia]
tablesNames=['Circuito','Escuderia','Piloto','CarreraDescripcion','CarreraResultado','
    Year','TemporadaPiloto','TemporadaEscuderia']
# Guardamos los datos en el destino
destinationPath = './datos/destination/'
for i in range(len(tables)):
    tables[i].to_csv(f'{destinationPath}{tablesNames[i]}.csv',index=False)
```

6.4. Ejecución ETL

El proceso ETL seguido se ha realizado con Python. Se comienza leyendo los datos del origen y del destino. Se filtran únicamente los nuevos registros que no aparecen en el destino. A los nuevos registros se les realizan las tareas transformación y uniones pertinentes. Finalmente, se guardan en ficheros CSVs que componen el almacén de datos.

El próximo paso consiste en añadir varias tareas de flujo de datos. Estas se encargan de todas las etapas del proceso ETL.

6.4.1. Descripción flujo de datos de la dimensión Piloto

El esquema del flujo de datos se visualiza en la Figura 6.4. En ella se explica que primero se descargan los datos de la API de Kaggle con el script que se indicó en el apartado 6.1. Se lee el fichero correspondiente a datos de pilotos (drivers.csv) y a los resultados de carreras (results.csv). También se accede a la tabla Piloto del almacén de datos y se seleccionan los nuevos registros de pilotos que no estén almacenados en el almacén. Esto se realiza para acelerar el proceso de la ETL y no volver a realizar las transformaciones a registros que ya tenemos almacenados en el almacén de datos.

Después se han filtrado los pilotos eliminando aquellos que han disputado menos de diez carreras en la historia de la Fórmula 1. Esto se ha realizado ya que en los comienzos de la F1 había muchos pilotos que únicamente participaban en una carrera. Mediante este filtro se reduce el número de pilotos de 857 a 373.

Tras ello a los nuevos registros mediante las librerías *Beautiful Soup* [75] y requests [87] de Python se ha obtenido mediante web scraping el enlace a la imagen de los pilotos en la Wikipedia ya que de cada piloto de drivers.csv se dispone de una variable llamada `url` la cuál es el enlace del piloto en la Wikipedia, gracias a este script con esa información podemos obtener el enlace de la imagen del piloto en la Wikipedia.

```
def get_image_url(wiki_url,delete_word):
    '''
    Funcion que obtiene la url de la imagen principal de un piloto, escuderia o
    circuito
    en la Wikipedia a partir del enlace a la pagina de la Wikipedia de ese piloto,
    escuderia o circuito.
    Argumentos:
        wiki_url: Enlace a la pagina de la Wikipedia de la que queremos extraer su
    imagen.
        delete_word: Palabra que no puede contener el enlace de la imagen deseada.
    Devuelve:
        Enlace de la imagen de la pagina de la Wikipedia pasada por argumento

    '''
    print(wiki_url)
```

```
    response = requests.get(wiki_url)
    soup = BeautifulSoup(response.text, 'html.parser')
    infobox = soup.find('table', {'class': 'infobox'})
    if infobox:
        images = infobox.find_all('img')

        if (len(images)>=1) and (delete_word not in images[0]['src'].lower()):
            print('https:' + images[0]['src'])
            return 'https:' + images[0]['src']
        elif len(images)>=2:
            # Si el primer enlace tiene la palabra descrita en 'delete_word' se escoge la segunda imagen
            if (images[1]):
                print('https:' + images[1]['src'])
                return 'https:' + images[1]['src']
    return None
```

Al realizar el web scraping se observó que no siempre lo realizaba correctamente ya que el script obtiene la primera imagen del recuadro principal del piloto en la Wikipedia y en algunos casos aparecía la bandera de la nacionalidad del piloto en vez de la imagen del piloto. Por ello, si se detecta en la url como "delete_word" = "flag" se descarta esa imagen y se selecciona la segunda que es la imagen del piloto.

En la información original de drivers.csv se dispone de la variable `nationality` la cuál es la nacionalidad del piloto, pero para nuestro destino también se requiere el país de origen. Se encontró un repositorio en GitHub que relacionaba los países con sus nacionalidades. Con el siguiente script gracias a la función *read_csv* se accede de manera sencilla a la información: [89]

```
# Cargar los datos de países de un dataset de GitHub para obtener el país apartir de la nacionalidad
url = 'https://raw.githubusercontent.com/knowitall/chunkedextractor/master/src/main/resources/edu/knowitall/chunkedextractor/demonyms.csv'
countriesDataset = pd.read_csv(url, on_bad_lines='skip')
# Renombre de las columnas para que coincidan con los datos que disponemos
countriesDataset.columns = ['nationality', 'country']
```

Por último, para actualizar la información de la tabla Piloto se ha creado una función la cuál recibe por parámetro los datos de origen (drivers.csv y results.csv), la tabla Piloto del almacén de datos y el dataset que relaciona países y nacionalidades.

Tras ello, se filtran los pilotos de los datos recién descargados del origen y se seleccionan únicamente los pilotos que no estuvieran almacenados en el destino. Después, se hace un segundo filtro eliminando a los pilotos con menos de 10 carreras en la Fórmula 1. Tras ello, se incluye el país de origen a cada piloto, se crea la variable `complete_name` y se realiza el web scraping para obtener la imagen del Piloto en la Wikipedia. Finalmente, se combinan los pilotos que teníamos previamente en la tabla Piloto del destino con los nuevos pilotos ya transformados.

```
def update_drivers_table(originDriver,destinationDriver,countriesDataset,results):
    '''
    Funcion que crea y devuelve la tabla Piloto. Se eliminan aquellos pilotos con menos de 10 carreras en la Formula 1.
    Argumentos:
        originDriver: Datos originales de pilotos.
        destinationDriver: Datos de pilotos almacenados en el data warehouse.
        countriesDataset: Dataset que enlaza paises y nacionalidades.
        results: Datos originales de resultados de carreras.
    Devuelve: Tabla Piloto
    '''
    # Se filtra los pilotos nuevos que no estuvieran ya guardados en el data warehouse
    new_drivers = originDriver[~originDriver['driverId'].isin(destinationDriver['driverId'])]
    # Se eliminan los pilotos con menos de 10 carreras en la Fórmula 1
    merged_driver_results = pd.merge(new_drivers, results, on='driverId')
    driver_count_races = merged_driver_results['driverId'].value_counts()
    driver_more_10_races = driver_count_races[driver_count_races >= 10].index
    new_drivers = new_drivers[new_drivers['driverId'].isin(driver_more_10_races)]

```

```
    # Se fusionan los nuevos pilotos para obtener el pais a partir de su nacionalidad
    new_drivers = new_drivers.merge(countriesDataset, how='left', left_on='nationality', right_on='nationality')
    # Se eliminan los pilotos duplicados
    new_drivers=new_drivers.drop_duplicates()
    # Se crea la variable de nombre completo
    new_drivers.loc[:,'complete_name']= new_drivers.loc[:,'forename']+' '+new_drivers.loc[:,'surname']
    # Se imputan los valores ausentes de las variables number y code
    new_drivers.loc[:,['number','code']] = new_drivers.loc[:,['number','code']].fillna('-')
    # Se realiza web scraping en la Wikipedia para obtener imagen del piloto
    new_drivers['image_url'] = new_drivers['url'].apply(get_image_url,delete_word='flag')
    destinationDriver = pd.concat([destinationDriver, new_drivers],ignore_index=True)
    return destinationDriver

Piloto = update_drivers_table(drivers,Piloto,countriesDataset,results)
```

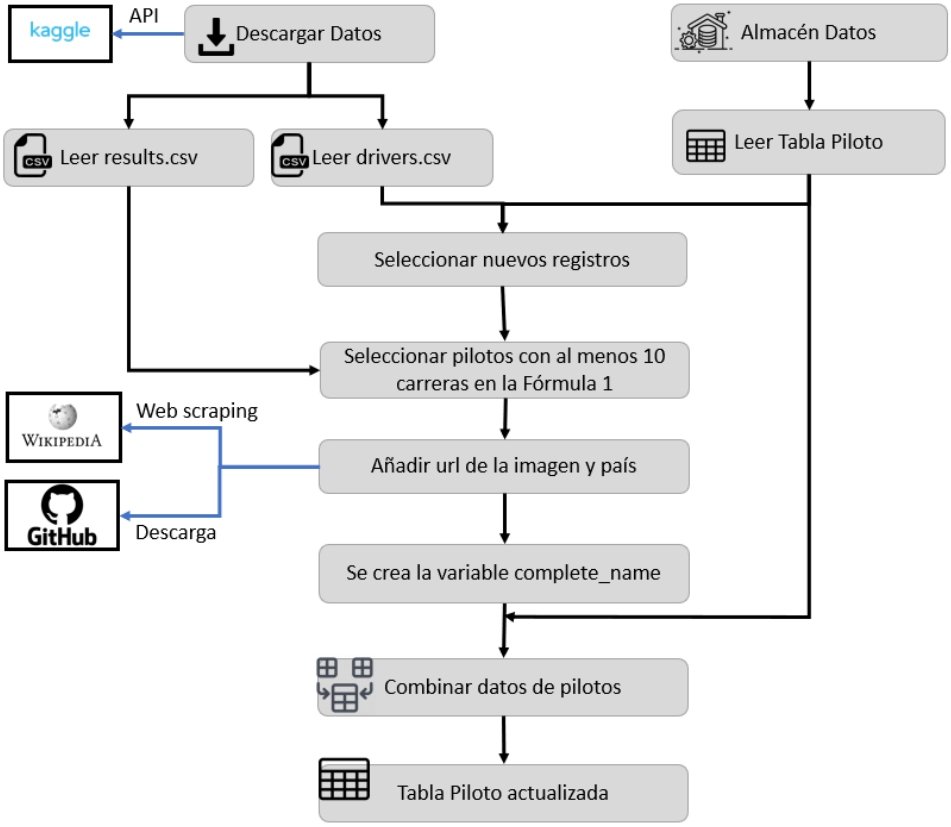


Figura 6.4: Flujo de datos de la dimensión Piloto (elaboración propia)

6.4.2. Descripción flujo de datos de la dimensión Escudería

El esquema del flujo de datos se visualiza en la Figura 6.5. En ella se explica que primero se descargan los datos de la API de Kaggle con el script que se indicó en el apartado 6.1, se lee el fichero correspondiente a datos de escuderías (constructors.csv) y a su vez se lee la tabla Escudería del almacén de datos y se seleccionan los nuevos registros de escuderías que no estén almacenados en el almacén. Esto se realiza para acelerar el proceso de la ETL y no volver a realizar las transformaciones a registros que ya tenemos almacenados en el almacén de datos.

Tras ello a los nuevos registros mediante las librerías *Beautiful Soup*[75] y requests [87] de Python se ha obtenido mediante web scraping el enlace a la imagen del logo de las escuderías en la Wikipedia ya que de cada escudería de constructors.csv se dispone de una variable llamada `url` la cuál es el enlace de la

escudería en la Wikipedia. Reutilizando el script que se utilizó para obtener la imagen de pilotos podemos obtener el enlace de la imagen del logo de la escudería en la Wikipedia, se vuelve a utilizar "delete_word" = "flag" para descartar las imágenes que aparece la bandera del país en vez del logo de la escudería y seleccionar la segunda imagen que es el logo.

En la información original de constructors.csv se dispone de la variable `nationality` la cuál es la nacionalidad de la escudería, pero para nuestro destino también se requiere el país de origen. Se vuelve a utilizar el mismo repositorio de GitHub que se utilizó para la tabla Piloto.

Por último, para actualizar la información de la tabla Escudería se ha creado una función la cuál recibe por parámetro los datos de origen (constructors.csv), la tabla Escudería del almacén de datos y el dataset que relaciona países y nacionalidades.

Tras ello, se filtran las escuderías de los datos recién descargados del origen y se seleccionan únicamente las escuderías que no estuvieran almacenados en el destino. Después, se incluye el país de origen a cada escudería y se realiza el web scraping para obtener la imagen del logo de la escudería en la Wikipedia. Finalmente, se combinan las escuderías que teníamos previamente en la tabla Escudería del destino con los nuevas escuderías ya transformadas.

```
def update_constructors_table(originConstructor,destinationConstructor,countriesDataset):
    '''
    Funcion que crea y devuelve la tabla Escuderia.
    Argumentos:
        originConstructor: Datos originales de escuderias.
        destinationConstructor: Datos de escuderias almacenados en el data warehouse.
        countriesDataset: Dataset que enlaza paises y nacionalidades.
    Devuelve: Tabla Escuderia

    '''
    # Se filtra las escuderia nuevas que no estuvieran ya guardados en el data warehouse
    new_constructors = originConstructor[~originConstructor['constructorId'].isin(destinationConstructor['constructorId'])]
    # Se fusionan las nuevas escuderia para obtener el pais a partir de su nacionalidad
    new_constructors = new_constructors.merge(countriesDataset, how='left', left_on='nationality', right_on='nationality')
    # Se eliminan las escuderias duplicadas
    new_constructors=new_constructors.drop_duplicates()
    # Se realiza web scraping en la Wikipedia para obtener imagen de la escuderia
    new_constructors['image_url'] = new_constructors['url'].apply(get_image_url, delete_word='flag')
    destinationConstructor = pd.concat([destinationConstructor, new_constructors], ignore_index=True)
    return destinationConstructor

Escuderia = update_constructors_table(constructors,Escuderia,countriesDataset)
```

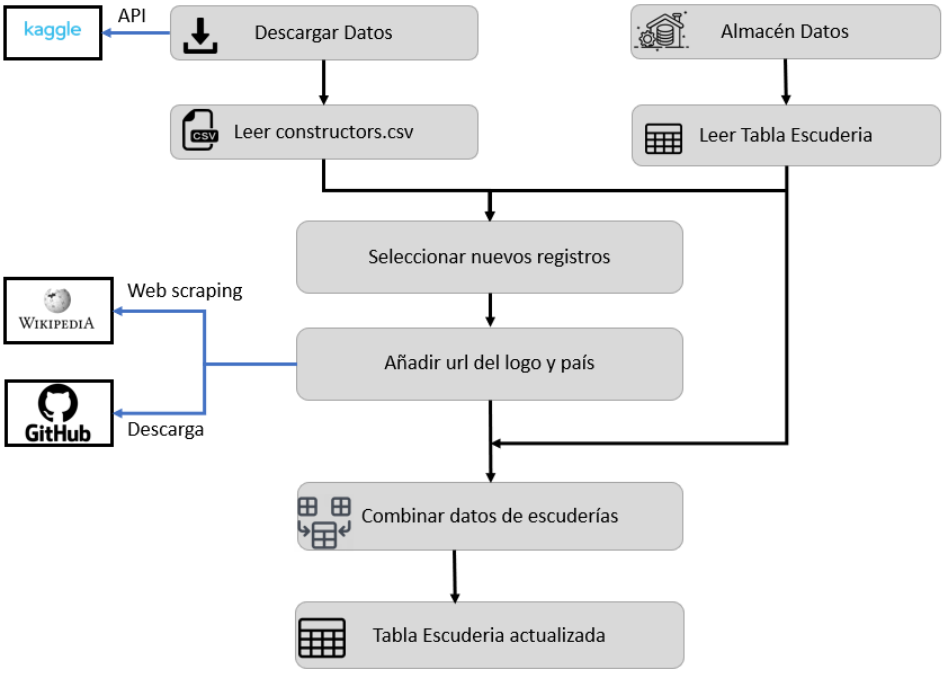


Figura 6.5: Flujo de datos de la dimensión Escudería (elaboración propia)

6.4.3. Descripción flujo de datos de la dimensión Circuito

El esquema del flujo de datos se visualiza en la Figura 6.6. En ella se explica que primero se descargan los datos de la API de Kaggle con el script que se indicó en el apartado 6.1, se lee el fichero correspondiente a datos de circuitos (circuits.csv) y a su vez se lee la tabla Circuito del almacén de datos y se seleccionan los nuevos registros de circuitos que no estén almacenados en el almacén. Esto se realiza para acelerar el proceso de la ETL y no volver a realizar las transformaciones a registros que ya tenemos almacenados en el almacén de datos.

Tras ello a los nuevos registros mediante las librerías *Beautiful Soup* [75] y requests [87] de Python se ha obtenido mediante web scraping el enlace a la imagen del trazado del circuito en la Wikipedia ya que de cada circuito en circuits.csv se dispone de una variable llamada `url` la cuál es el enlace del circuito en la Wikipedia, reutilizando el script que se utilizó para obtener la imagen de pilotos y escuderías podemos obtener el enlace de la imagen del circuito en la Wikipedia, esta vez se utiliza "delete_word" = "logo" para descartar las imágenes que aparece el logo del circuito en vez del trazado que es la imagen deseada.

Por último, para actualizar la información de la tabla Circuito se ha creado una función la cuál recibe por parámetro los datos de origen (circuits.csv) y la tabla Circuito del almacén de datos.

Tras ello, se filtran los circuitos de los datos recién descargados del origen y se seleccionan únicamente los circuitos que no estuvieran almacenados en el destino. Después, se realiza el web scraping para obtener la imagen del trazado del circuito en la Wikipedia. Finalmente, se combinan los circuitos que teníamos previamente en la tabla Circuito del destino con los nuevos circuitos ya transformados.

```
def update_circuits_table(originCircuit,destinationCircuit):
    '''
    Funcion que crea y devuelve la tabla Circuito.
    Argumentos:
        originCircuit: Datos originales de circuitos.
        destinationCircuit: Datos de circuitos almacenados en el data warehouse.
    Devuelve: Tabla Circuito
    '''
     # Se filtra los circuitos nuevos que no estuvieran ya guardados en el data warehouse
    new_circuits = originCircuit[~originCircuit['circuitId'].isin(destinationCircuit['circuitId'])]
     # Se realiza web scraping en la Wikipedia para obtener imagen del circuito
```

```
    new_circuits['image_url'] = new_circuits['url'].apply(get_image_url,delete_word='
    logo')
    destinationCircuit = pd.concat([destinationCircuit, new_circuits],ignore_index=True
    )
    return destinationCircuit

Circuito = update_circuits_table(circuits,Circuito)
```

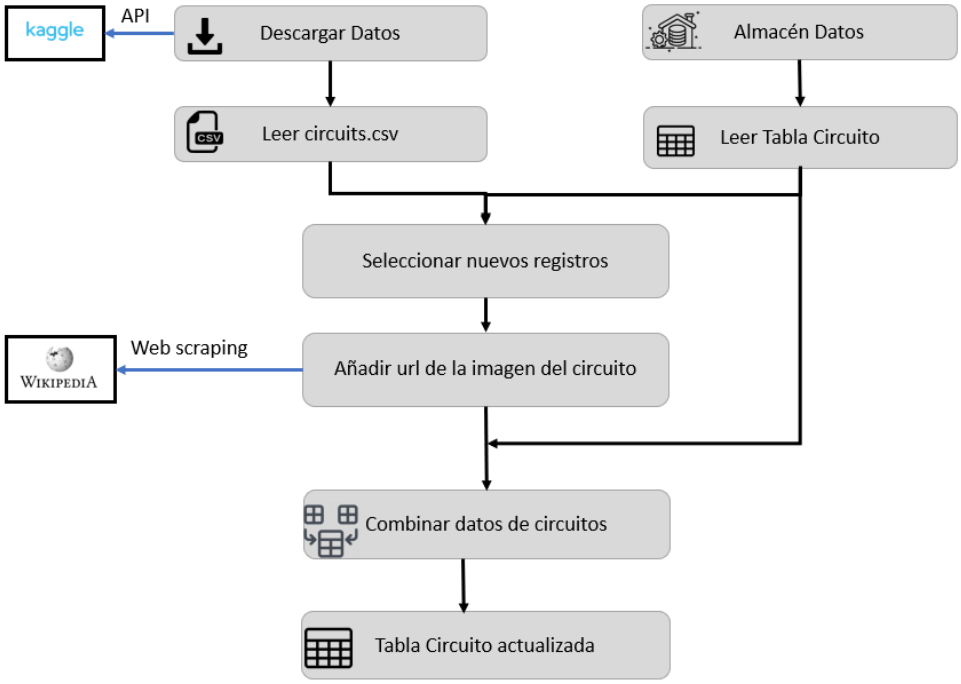


Figura 6.6: Flujo de datos de la dimensión Circuito (elaboración propia)

6.4.4. Descripción flujo de datos de la tabla de hechos CarreraResultado

El esquema del flujo de datos se visualiza en la Figura 6.7. En ella se explica que primero se descargan los datos de la API de Kaggle con el script que se indicó en el apartado 6.1. Después se lee el fichero correspondiente a datos de resultados de carreras (results.csv) y a su vez se lee la tabla CarreraResultado del almacén de datos y se seleccionan los nuevos registros de resultados que no estén almacenados en el almacén. Esto se realiza para acelerar el proceso de la ETL y no volver a realizar las transformaciones a registros que ya tenemos almacenados en el almacén de datos.

Tras ello, se descartan las columnas`position`, `positionText`, `fastestLap` y `statusId` ya que no forman parte de las columnas del destino cómo se describió en el mapa lógico del apartado 6.2.3. Cuando se han seleccionado las columnas que van a estar en el destino se crean las variables `Victoria`, `Podio` y `Pole` a partir de las variables de posición final (`positionOrder`) y posición inicial (`grid`) de los resultados de carrera, también se crea `fastestLapTimeMilliseconds` a partir de `fastestLapTime` y `status_fault` a partir de `status` (más detalles de las nuevas variables en el apartado 6.2.4).

Para crear la variable `status_fault`, se crea una función llamada *get_ status_ fault(status)* que clasifica los tipos de abandono en errores de piloto o de escudería.

```
HUMAN_FAULTS = ['Accident', 'Collision', 'Collision damage', 'Damage', 'Debris', 'Fatal
    accident', 'Illness','Injured', 'Injury', 'Physical', 'Retired', 'Safety', 'Spun
    off', 'Stalled']

MACHINE_FAULTS = [  'Alternator', 'Axle', 'Battery', 'Brake duct', 'Brakes', 'CV joint'
    , 'Chassis', 'Clutch', 'Crankshaft', 'Damage', 'Debris','Differential', 'Distributor
    ', 'Driver Seat', 'Driveshaft', 'Drivetrain', 'ERS', 'Electrical', 'Electronics', '
    Engine', 'Engine fire', 'Engine misfire','Exhaust', 'Excluded', 'Eye injury', 'Fire'
    , 'Front wing', 'Fuel leak', 'Fuel pipe', 'Fuel pressure', 'Fuel pump', 'Fuel rig',
```

```
    'Fuel system', 'Gearbox', 'Halfshaft', 'Handling','Heat shield fire', 'Hydraulics',
    'Ignition', 'Injection', 'Launch control', 'Magneto',
'Mechanical', 'Oil leak', 'Oil line', 'Oil pipe', 'Oil pressure', 'Oil pump', 'Out of
    fuel','Overheating', 'Pneumatics', 'Power Unit', 'Power loss', 'Puncture', 'Radiator
    ', 'Rear wing','Refuelling', 'Safety belt', 'Safety concerns', 'Seat', 'Spark plugs'
    , 'Steering', 'Supercharger',
'Suspension', 'Technical', 'Throttle', 'Track rod', 'Transmission', 'Turbo', 'Tyre', '
    Tyre puncture', 'Undertray', 'Underweight', 'Vibrations', 'Water leak', 'Water pipe'
    , 'Water pressure', 'Water pump', 'Wheel', 'Wheel bearing', 'Wheel nut', 'Wheel rim'
    ]


def get_status_fault(status):
    '''
    Funcion que indica el culpable de un abandono. Puede ser culpa del piloto o de la
    escuderia. Tambien puede tomar el valor 'otro' significa que no ha abandonado la
    carrera.
    '''
    if status in HUMAN_FAULTS:
        return 'piloto'
    elif status in MACHINE_FAULTS:
        return 'escuderia'
    else:
        return 'otro'
```

Por último, para actualizar la información de la tabla CarreraResultado se ha creado una función la cuál recibe por parámetro los datos de origen (results.csv) y la tabla CarreraResultado del almacén de datos.

Tras ello, se filtran los resultados de carreras de los datos recién descargados del origen y se seleccionan únicamente los resultados de carreras que no estuvieran almacenados en el destino. Después se seleccionan las columnas deseadas y se crean las nuevas variables. Finalmente, se combinan los resultados de carreras que teníamos previamente en la tabla CarreraResultado del destino con los nuevos resultados de carreras ya transformados.

```
def update_results_table(originResults,destinationResults,status):
    '''
    Funcion que crea y devuelve la tabla CarreraResultado.
    Argumentos:
        originResults: Datos originales de resultados.
        destinationResults: Datos de resultados almacenados en el data warehouse.
        status: Datos respecto al estado de como puede finalizar una carrera un piloto.
    Devuelve: Tabla CarreraResultado
    '''
    # Se filtra los resultados nuevos que no estuvieran ya guardados en el data
    warehouse
    new_results = originResults[~originResults['resultId'].isin(destinationResults['
    resultId'])]
    new_results = pd.merge(new_results, status[['statusId', 'status']], on='statusId',
    how='left')
    # Se eliminan las columnas no deseadas
    new_results = new_results.drop(['position','positionText','fastestLap','statusId'],
    axis=1)
    # Se crean las variables nuevas
    new_results['status_fault'] = new_results['status'].apply(get_status_fault)
    new_results['fastestLapTimeMilliseconds'] = pd.to_timedelta('00:'+new_results['
    fastestLapTime']).dt.total_seconds() * 1000
    new_results['Victoria'] = (new_results['positionOrder'] == 1).astype(int)
    new_results['Podio'] = (new_results['positionOrder'] <= 3).astype(int)
    new_results['Pole'] = (new_results['grid'] == 1).astype(int)
    destinationResults = pd.concat([destinationResults, new_results],ignore_index=True)
    return destinationResults

CarreraResultado = update_results_table(results,CarreraResultado,status)
```

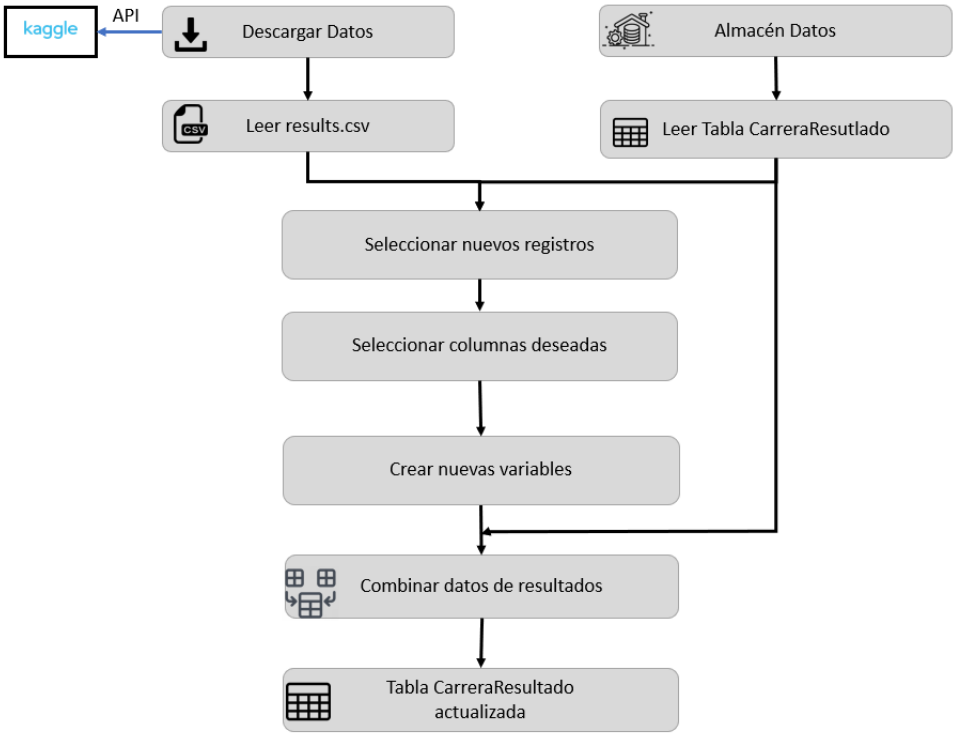


Figura 6.7: Flujo de datos de la dimensión CarreraResultado (elaboración propia)

6.4.5. Descripción flujo de datos de la dimensión CarreraDescripcion

El esquema del flujo de datos se visualizar en la Figura 6.8. En ella se explica que primero se descargan los datos de la API de Kaggle con el script que se indicó en el apartado 6.1, se lee el fichero correspondiente a datos de descripciones de carreras (races.csv) y de resultados de carreras (results.csv). También se lee la tabla CarreraDescripcion del almacén de datos y se seleccionan los nuevos registros de descripciones de carreras que no estén almacenados en el almacén. Esto se realiza para acelerar el proceso de la ETL y no volver a realizar las transformaciones a registros que ya tenemos almacenados en el almacén de datos. Después, se realiza un filtro eliminando aquellas descripciones de carreras que no tengan resultado ya que en el origen al principio de año añaden todas las descripciones de carreras del año y después a medida que suceden las carreras añaden los resultados.

Tras ello, se descartan las columnas: `fp1_date`, `fp1_time`, `fp2_date`, `fp2_time`, `fp3_date`, `fp3_time`, `quali_date`, `quali_time`, `sprint_date`, `sprint_time`, ya que no forman parte de las columnas del destino cómo se describió en el mapa lógico del apartado 6.2.3. Después, a los nuevos registros mediante las librerías *Beautiful Soup* [75] y requests [87] de Python se ha obtenido, mediante web scraping en la Wikipedia, la meteorología de la carrera, el tipo de circuito y la longitud del circuito en esa carrera. Esto ha sido posible porque en cada carrera de races.csv se dispone de una variable llamada `url` la cuál es el enlace de la carrera en la Wikipedia. Para ello, se han creado las funciones *get_course_race(table)*, *get_course_length_race(table)*, *get_weather_race(table)* y *get_info_race(wiki_url)*, (más detalles de las nuevas variables en el apartado 6.2.4).

```
def get_course_race(table):
    '''
    Funcion que obtiene y devuelve el tipo de circuito de una carrera concreta mediante
     web scraping en la Wikipedia.
    '''
    if(table.find('th', text='Course') is not None):
        course_row = table.find('th', text='Course').parent
        course_info = course_row.find_all('td')[0:2]
        course_name = course_info[0].get_text().strip()
        return course_name
    return None

```

```
def get_course_length_race(table):
    '''
    Funcion que obtiene y devuelve la longitud del trazado de una carrera concreta
    mediante web scraping en la Wikipedia
    '''
    if(table.find('th', text='Course length') is not None):
        courseLength_row = table.find('th', text='Course length').parent
        courseLength_info = courseLength_row.find_all('td')[0:2]
        course_length = courseLength_info[0].get_text().strip()
        return course_length
    return None
def get_weather_race(table):
    '''
    Funcion que obtiene y devuelve la meteorologia de una carrera concreta mediante web
     scraping en la Wikipedia
    '''
    if(table.find('th', text='Weather') is not None):
        weather_row = table.find('th', text='Weather').parent
        weather_info = weather_row.find_all('td')[0].get_text().strip()
        return weather_info
    return None

def get_info_race(wiki_url):
    '''
    Funcion que recopila y devuelve la informacion extra de una carrera realizando web
    scraping en la Wikipedia.
    Obtiene el tipo de circuito, la longitud del trazado y la meteorologia.
    '''
    print(wiki_url)
    response = requests.get(wiki_url)
    soup = BeautifulSoup(response.content, 'html.parser')
    race_summary_table = soup.find('table', class_='infobox vevent')
    course_name=get_course_race(race_summary_table)
    course_length=get_course_length_race(race_summary_table)
    weather_info=get_weather_race(race_summary_table)

    return [course_name, course_length, weather_info]
```

Después de realizar el scraping es necesario limpiar las variables para que tengan los valores deseados, esto se ha realizado con tres funciones llamadas clean_type_circuit (raw_type), clean_circuit_length (raw_length) y clean_weather (raw_weather).

```
def clean_type_circuit(raw_type):
    '''
    Funcion que clasifica el circuito en circuito urbano o permanente segun el texto de
     descripcion de la Wikipedia.
    '''
    if 'street' in str(raw_type).lower():
        return 'Circuito urbano'
    elif 'permanent' in str(raw_type).lower():
        return 'Circuito permanente'
    else:
        return 'otro'
def clean_weather(raw_weather):
    '''
    Funcion que clasifica la meteorologia de una carrera en 'Lluvia',''Nublado', '
    Soleado' u 'Otro'
    '''
    RAINY_WORDS=['rain','rainy','wet','showers','Drizzly']
    CLOUDY_WORDS=['cloudy','clouds','cloud','overcast']
    SUNNY_WORDS= ['sunny','dry','warm','clear','Fine']

    if any(word in str(raw_weather).lower() for word in RAINY_WORDS):
        return 'Lluvia'
    elif any(word in str(raw_weather).lower() for word in CLOUDY_WORDS):
        return 'Nublado'
```

```
    elif any(word in str(raw_weather).lower() for word in SUNNY_WORDS):
        return 'Soleado'
    else:
        return 'Otro'

def clean_circuit_length(raw_length):
    '''
    Funcion que transforma el texto con la longitud del circuito en un numero flotante.
    '''
    kilometers = raw_length.split(' ')[0]

    if '[' in kilometers:

        kilometers = kilometers.split('[')[0]
    elif '\xa0' in kilometers:

        kilometers = kilometers.split('\xa0')[0]

    return float(kilometers)
```

Por último, para actualizar la información de la tabla CarreraDescripcion se ha creado una función la cuál recibe por parámetro los datos de origen (races.csv), la tabla CarreraDescripcion del almacén de datos y los resultados de las carreras.

Tras ello, se filtran las escuderías de los datos recién descargados del origen y se seleccionan únicamente las carreras que no estuvieran almacenados en el destino y que tengan resultados. Esto es porque en el origen de datos al comienzo de año se añaden todas las descripciones de carreras del año aunque no estén disputadas, pero en el destino no se quiere tener descripciones de carreras no disputadas. Después, se seleccionan las columnas deseadas y se crean las nuevas variables y realiza el web scraping para obtener la longitud y tipo del circuito y la meteorología en la Wikipedia. Finalmente, se combinan las descripciones de carreras que teníamos previamente en la tabla CarreraDescripcion del destino con los nuevas descripciones de carreras ya transformadas.

```
def update_races_table(originRaces,destinationRaces,resultsRaces):
    '''
    Funcion que crea y devuelve la tabla CarreraDescripcion. Se almacenan descripciones
     de carreras que tengan resultados.

    Argumentos:
        originRaces: Datos originales de descripciones de carreras.
        destinationRaces: Datos de descripciones de carreras almacenados en el data
    warehouse.
        resultsRaces: Datos originales de resultados.
    Devuelve: Tabla CarreraDescripcion
    '''
    # Se filtra las descripciones de carreras nuevas que no estuvieran ya guardados en
    el data warehouse
    new_races = originRaces[~originRaces['raceId'].isin(destinationRaces['raceId'])]
    # se realiza la descripcion de carreras que esten en resultados
    new_races = new_races[new_races['raceId'].isin(resultsRaces['raceId'])]

    if(len(new_races.index)>0):
        # Se eliminan las columnas no deseadas
        new_races = new_races.drop(['fp1_date','fp1_time','fp2_date','fp2_time','
    fp3_date','fp3_time',
                                    'quali_date', 'quali_time', 'sprint_date', '
    sprint_time' ],axis=1)
        # Se incluyen  las variables obtenidas con web scraping
        raw_info=new_races.loc[:,'url'].apply(get_info_race)
        stacked_data = np.vstack(raw_info.values)
        race_info_raw = pd.DataFrame(stacked_data, columns=['typeCircuitRaw', '
    circuitLengthRaw', 'weatherRaw'])
        typeCircuitCleaned = race_info_raw.loc[:,'typeCircuitRaw'].apply(
    clean_type_circuit)
        weatherCleaned = race_info_raw.loc[:,'weatherRaw'].apply(clean_weather)
```

```
26          circuitLengthCleaned = race_info_raw.loc[:,'circuitLengthRaw'].apply(
    clean_circuit_length)
27          new_variables = pd.DataFrame({'typeCircuitCleaned': typeCircuitCleaned, '
    weatherCleaned': weatherCleaned, 'circuitLengthCleaned': circuitLengthCleaned})
28
29          new_variables.index=new_races.index
30          new_races = pd.concat([new_races, new_variables], axis=1)
31          destinationRaces = pd.concat([destinationRaces, new_races],ignore_index=True)
32      return destinationRaces
33
34  CarreraDescripcion = update_races_table(races,CarreraDescripcion,results)
```

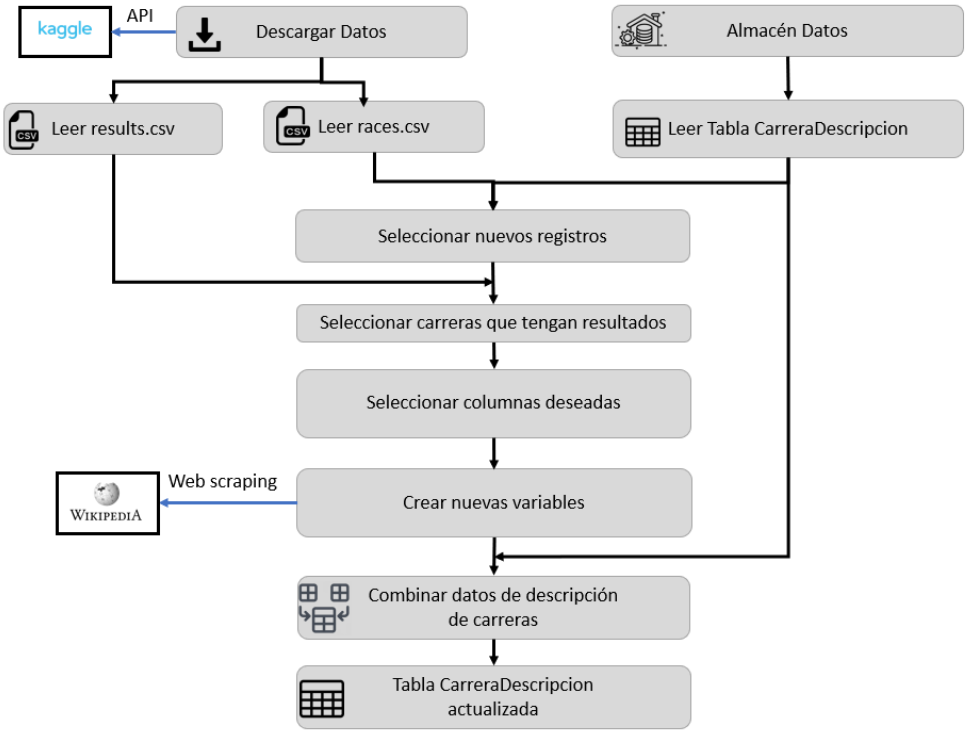


Figura 6.8: Flujo de datos de la dimensión CarreraDescripcion (elaboración propia)

6.5. Resultado de la ETL

6.5.1. Esquema del almacén de datos implementado

Tras realizar la ETL el esquema de almacén de datos con los campos contraídos se encuentra en la Figura 6.9 mientras que el esquema con campos expandidos se encuentra en la Figura 6.10.

El mayor cambio respecto al esquema realizado en análisis, (ver Figura 5.4) es la eliminación de las tablas de hechos de TiempoVuelta y TiempoPitStop a causa de la ingente cantidad de outliers que presentaban por errores de medición.

6.5.2. Verificación/test validación

Las pruebas referentes al flujo de la ETL se detallan en el capítulo 9 referente a las pruebas realizadas.

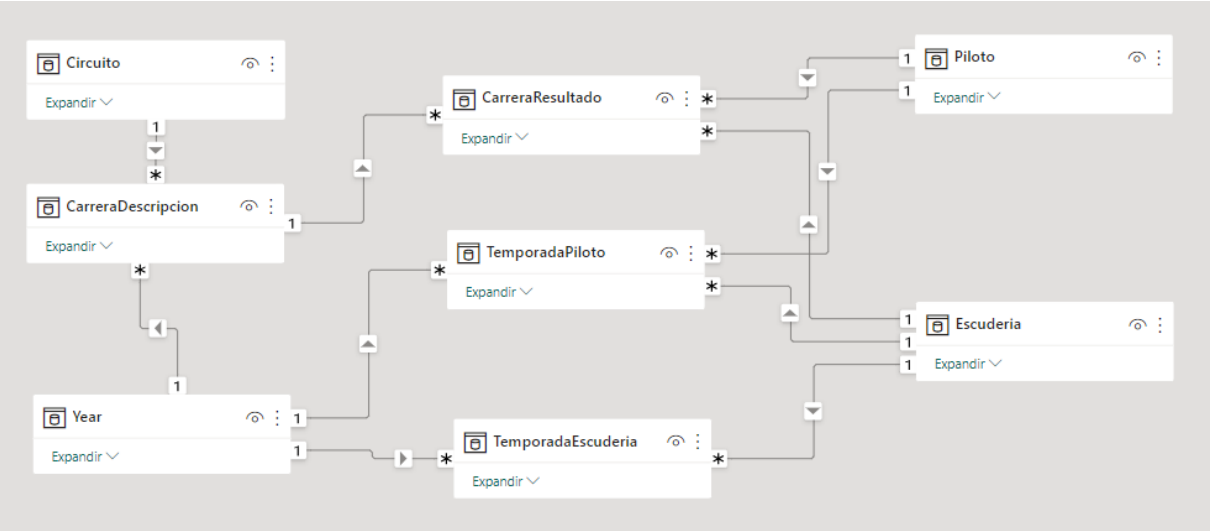


Figura 6.9: Esquema del almacén de datos implementado con los campos contraídos

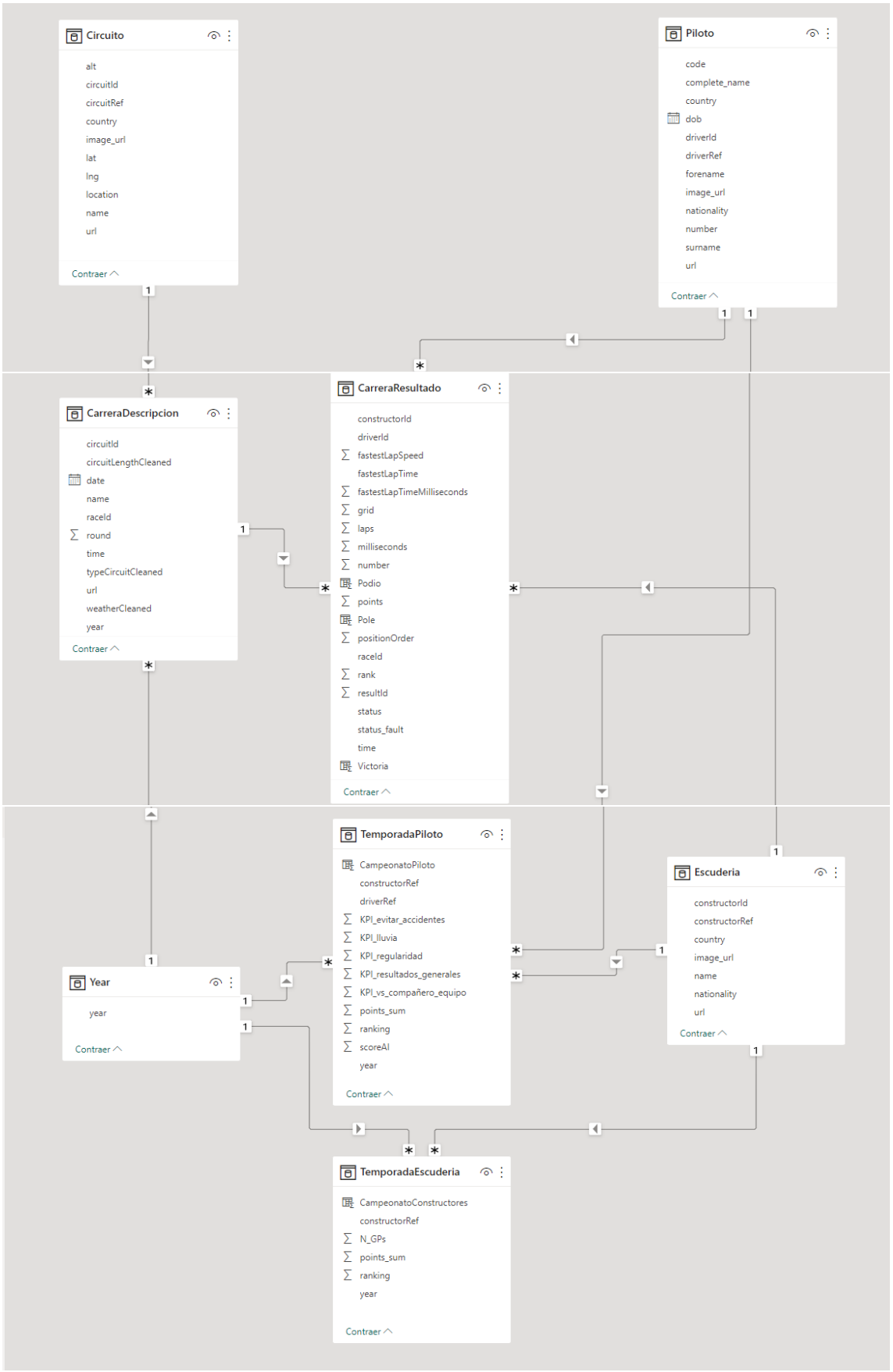


Figura 6.10: Esquema del almacén de datos implementado con los campos expandidos

Capítulo 7

Métrica *scoreAI*

La Fórmula 1 es un deporte en el cual la posición final de los pilotos en la clasificación está fuertemente correlacionada con la calidad del coche que conducen. Es por ello, que los puntos obtenidos al final de temporada no tiene porqué ser un reflejo del nivel real de un piloto ya que puede haberse influenciado de tener un buen o mal coche.

Por ese motivo, se ha creado una métrica que evalúe pilotos de Fórmula 1 reduciendo el efecto del monoplaza, llamada *scoreAI*. Esta medida se calcula midiendo múltiples variables de las diferentes carreras de un año.

7.1. Explicación métrica *scoreAI*

ScoreAI es una medida flotante acotada en el rango de 0-100, ya que es un rango muy común para establecer puntuaciones. Valores bajos de la variable significa que un piloto ha hecho mala temporada, por el contrario, valores altos en *scoreAI* significa que el piloto ha realizado buena temporada. Esta métrica corresponde a una medida por piloto y año.

Se han considerado varios factores clave para adjudicar buena medida a un piloto y son las siguientes:

- Buenos resultados respecto a su compañero de equipo: Esto es para mitigar el efecto del monoplaza. Ya que entre pilotos de la misma escudería los coches que conducen son similares.
- Buenos resultados en carrera con lluvia: Un buen piloto de Formula 1 debe ser bueno ante condiciones adversas como la lluvia. Gracias al web scraping se ha podido obtener la meteorología en cada carrera y saber si llovió en la carrera o no.
- Pocos accidentes: Un buen piloto debe destacar en minimizar tanto los accidentes como las colisiones entre múltiples monoplazas.
- Buenos resultados respecto al resto de pilotos: Se compara los resultados del piloto con el resto de pilotos de ese año.
- Regularidad en sus resultados: Otro factor para ser buen piloto es el de ser constante en sus resultados. Es decir, que no tenga altibajos en sus resultados ya sea en la carrera final, clasificación o en el ranking de la vuelta rápida.

El cliente pide que *scoreAI* sea una medida explicable. Para poder explicar el porqué de la puntuación creada a cada piloto por año se crean cinco métricas parciales que van a ayudar a explicar el valor de la métrica asignada. Cada métrica parcial está asociada a uno de los cinco factores mencionados previamente que se han considerado como importantes para ser un buen piloto de F1.

Estas son las cinco métricas parciales creadas:

- `KPI_vs_compañero_equipo`: Métrica que indica la puntuación de un piloto respecto a su compañero de equipo. Es una medida flotante acotada en el rango 0-100. Esta variable recoge las diferencias entre compañeros de equipo en múltiples variables como posición final en carrera, diferencias en el

número de accidentes... Esta variable apenas está influenciada por la calidad del monoplaza ya que entre pilotos de la misma escudería los coches son similares. Valores grandes de está variable indican grandes diferencias a favor del piloto en comparación con su compañero de equipo.

- `KPI_lluvia`: Métrica que indica la puntuación de un piloto respecto a los resultados en carreras con lluvia. Es una medida flotante acotada en el rango 0-100. Esta variable recoge información respecto a los resultados en lluvia de un piloto. Valores grandes de está variable indican buenos resultados del piloto en carreras con lluvia. Se define como carrera con lluvia si en la descripción del clima de la carrera en la Wikipedia aparece alguna de las siguientes palabras: "rain","rainy","wet","showers","Drizzly".

- `KPI_evitar_accidentes`: Métrica que mide la habilidad evitando accidentes de un piloto. Es una medida flotante acotada en el rango 0-100. Esta variable recoge información respecto a la cantidad de accidentes que ha provocado y número de colisiones múltiples en las que ha participado. Valores grandes de está variable indica que el piloto ha participado en pocos accidentes y colisiones múltiples.

- `KPI_resultados_generales`: Métrica que indica la puntuación de un piloto respecto al resto de pilotos de la Formula 1 en ese año. Es una medida flotante acotada en el rango 0-100. Esta variable recoge las diferencias entre todos los pilotos del año en múltiples variables como posición final en carrera, diferencias en el número de accidentes... Valores grandes de está variable indican grandes diferencias a favor del piloto en comparación con el resto de competidores. En esta variable sí influye la calidad del monoplaza pero se reduce la influencia de la calidad del compañero de equipo.

- `KPI_regularidad`: Métrica que indica la regularidad de un piloto. Es una medida flotante acotada en el rango 0-100. Esta variable recoge la variabilidad de un piloto en la posición obtenida en carrera, en la salida, en el ranking para alcanzar la vuelta rápida y en el número de posiciones ganadas en carrera. Valores grandes de está variable indica que un piloto es regular y consistente en sus resultados obtenidos.

Por tanto, la métrica global `scoreAI` es una combinación de las cinco métricas parciales dando más peso a los resultados contra el compañero de equipo (`KPI_vs_compañero_equipo`) ya que es una medida en la que apenas influye la calidad del monoplaza. Después en orden de importancia se considera la habilidad evitando accidentes (`KPI_evitar_accidentes`). A continuación, en un tercer nivel están con el mismo nivel de importancia los resultados con lluvia (`KPI_lluvia`) y el resultado frente el resto de pilotos (`KPI_resultados_generales`). Por último, en menor medida se tiene en cuenta la regularidad de un piloto (`KPI_regularidad`).

7.2. Etapas en la creación de *scoreAI*

Cuando ya se disponen los datos depurados del resto de tablas deben realizarse varios procesos a partir de datos de resultados de carreras para obtener la puntuación *scoreAI*.

7.2.1. Primera etapa: Crear histórico de carreras

El primer paso para calcular la métrica *scoreAI* para piloto y año es crear un histórico de carreras. Este histórico tiene que disponer de toda la información de cada carrera. Para crearlo es necesario fusionar las tablas de resultados (CarreraResultado) y de descripciones de carreras (CarreraDescripcion).

Para lograr está fusión se realiza un *left join* mediante la clave foránea `raceId` de CarreraResultado y la clave primaria `raceId` de CarreraDescripcion. También se ha realizado un *join* con la tabla Piloto para disponer del nombre del piloto mediante la clave `driverId`. En Python los *joins* entre tablas se realizan con la función *merge* y hay que indicarle mediante el argumento *how* que se desea realizar un *left join*. También se indica con *left_on* y *right_on* los campos con los que se realiza el *join*, este caso `raceId`.

```
# Script para crear el historico de carreras
historicoCarreras = CarreraResultado.merge(CarreraDescripcion, how='left', left_on='
    raceId', right_on='raceId')
historicoCarreras = historicoCarreras.merge(Piloto, how='left', left_on='driverId',
    right_on='driverId')
```

Cuando se tiene el histórico de resultados de carreras se crean nuevas variables a partir de la información original y así enriquecer la información disponible.

Se han creado cinco variables binarias y una variable entera mediante funciones *lambda*:

- `victory`: Variable binaria, indica si un piloto ha terminado primero la carrera (1 indica que ha quedado primero, 0 si no ha quedado primero).
- `podium`: Variable binaria, indica si un piloto ha terminado entre los tres primeros la carrera (1 ha quedado en el podio, 0 si no ha quedado en el podio).
- `pole`: Variable binaria, indica si un piloto ha empezado primero la carrera (1 indica que ha empezado primero, 0 si no ha empezado primero).
- `accident`: Variable binaria, indica si el piloto ha provocado un accidente (1 indica que ha provocado un accidente, 0 si no lo ha provocado).
- `collision`: Variable binaria, indica si el piloto ha estado involucrado en una colisión múltiple (1 indica que ha estado involucrado en una colisión múltiple, 0 si no ha estado involucrado).
- `gainedPosition`: Variable entera, indica el número de posiciones que ha ganado un piloto respecto a la salida. Se calcula así: `grid` - `positionOrder`.

Mediante la función *assign* y las funciones *lambda* permite crear las variables de una manera compacta.

```
# Script para crear variables nuevas apartir de las que ya se disponen
historicoCarreras = historicoCarreras.assign(
    victory = lambda x: (x['positionOrder'] == 1).astype(int),
    podium = lambda x: (x['positionOrder'] <= 3).astype(int),
    pole = lambda x: (x['rank'] == 1).astype(int),
    accident = lambda x: (x['status'] == 'Accident').astype(int),
    collision = lambda x: (x['status'].isin(['Collision','Collision damage'])).astype(
    int),
    gainedPosition = lambda x: (x['grid']- x['positionOrder']).astype(int)
)
```

7.2.2. Segunda etapa: Crear histórico de carreras agrupado

Como se ha mencionado previamente la métrica objetivo es por piloto y año. En cambio, la información disponible es por carrera por ese motivo se debe realizar agrupaciones por piloto y año de todas las variables. Estás agrupaciones pueden realizarse de diferente manera ya sea realizando la suma, la media, varianza, conteos...

Se crean tres tipos de agregaciones distintas:

- **Agregación anual de carreras terminadas**: Se realiza está agregación sobre el histórico de carreras pero seleccionando los resultados de carreras de pilotos que han terminado la carrera. Esto se indica mediante (`status_fault`== "otro"). Se realiza esta agrupación por año, piloto y escudería. Las variables agregadas son:
  - `points`: Se calcula la suma de puntos en un año por piloto y escudería. No se incluye en la métrica final pero sirve para calcular la posición final en el ranking.
  - `positionOrder`: Se realizan dos agregaciones para la posición final en la carrera. La primera es realizando la media para poder compararlo con el resto de pilotos y con su compañero de equipo. La segunda es calculando la desviación típica para observar si un piloto es consistente en las carreras durante el año.
  - `grid`: Se realizan dos agregaciones para la posición inicial en la carrera, la media y la desviación típica.
  - `rank`: Se realizan dos agregaciones para la posición del piloto en alcanzar la vuelta rápida en la carrera, la media y la desviación típica.

- `gainedPosition`: Se realizan dos agregaciones para el número de posiciones ganadas por un piloto en carrera, la media y la desviación típica.

Para agrupar se utiliza la función *groupby* y se indican los valores por los que se desea agrupar. Para realizar varias agregaciones simultáneamente de manera compacta se utiliza la función *agg* en la que hay que pasar como argumento un diccionario con las agregaciones deseadas.

```
# Script agregacion de carreras finalizadas por los pilotos
aggregationsFinishedRaces = {
    'points': 'sum',
    'positionOrder': ['mean','std'],
    'grid': ['mean','std'],
    'rank': ['mean','std'],
    'gainedPosition':['mean','std'],
 }
agg_finish_race = race_history.loc[race_history.status_fault=='otro',:].groupby(['
    year', 'driverId','constructorId']).agg(aggregationsFinishedRaces)
```

- **Agregación anual de carreras con lluvia**: Se realiza está agregación sobre el histórico de carreras pero seleccionando únicamente las carreras que ha llovido (`weatherCleaned`=="Lluvia"). Se realiza esta agrupación por año, piloto y escudería. Las variables agregadas son:

  - `positionOrder`: Se realiza una agregación para la posición final en la carrera. Para esta agregación se utiliza la media para comparar con el resto de pilotos y con su compañero de equipo.
  - `rank`: Se realiza una agregación para la posición del piloto en alcanzar la vuelta rápida. Para esta agregación se utiliza la media.
  - `gainedPosition`: Se realizan una agregación para el número de posiciones ganadas por un piloto en carrera.Para esta agregación se utiliza la media.

  No se ha incluido la información de la posición inicial en carrera (`grid`) ya que sólo conocemos que ha llovido el día de la carrera no el día que se ha disputado la clasificación.

```
# Script agregacion de carreras con lluvia
aggregationsRainyRaces = {
    'positionOrder': 'mean',
    'rank':'mean',
    'gainedPosition':'mean'
}
agg_rainy_race = race_history.loc[race_history.weatherCleaned=='Lluvia',:].groupby
    (['year', 'driverId','constructorId']).agg(aggregationsRainyRaces)
```

- **Agregación anual de todas las carreras**: Se realiza está agregación sobre el histórico de carreras pero sin filtrar. Se realiza esta agrupación por año, piloto y escudería. Las variables agregadas son:

  - `victory`: Se calcula la suma de victorias en un año por piloto y escudería.
  - `podium`: Se calcula la suma de podios en un año por piloto y escudería.
  - `pole`: Se calcula la suma de poles en un año por piloto y escudería.
  - `accident`: Se calcula la suma de accidentes que ha provocado en un año por piloto y escudería.
  - `collision`: Se calcula la suma de colisiones múltiples en las que ha estado involucrado en un año por piloto y escudería.

```
# Script agregacion de todas las carreras
aggregationsAllRaces = {
    'victory': 'sum',
    'podium': 'sum',
    'pole': 'sum',
    'accident': 'sum',
    'collision': 'sum',
}
agg_all_race= race_history.groupby(['year', 'driverId','constructorId']).agg(
    aggregationsAllRaces)
```

Tras realizar las tres agregaciones es necesario fusionar los tres resultados de las agregaciones en un único dataframe, esto se logra con la función *merge* de la librería *pandas*. El objeto resultado de la fusión se transforma en un dataframe mediante la función *reset_index*. Cada agregación de cada variable es una columna del dataframe. Cada fila representa los resultados de un piloto en una escudería en un año concreto.

```
# Script para crear un dataframe de un historico agrupado que fusione las tres agregaciones
grouped_race_history = pd.merge(agg_finish_race,agg_all_race,on=['year', 'driverId','constructorId'])
# Importante hacer el join por la izquierda porque existen temporadas en las que no llueve
grouped_race_history = pd.merge(grouped_race_history,agg_rainy_race,on=['year', 'driverId','constructorId'],how='left')

# Se convierte el objeto creado por groupby a un dataframe
grouped_race_history=grouped_race_history.reset_index()
# Se renombran las variables

grouped_race_history.columns = ['year','driverId','constructorId','points_sum','positionOrder_mean','positionOrder_std','grid_mean','grid_std','rank_mean','rank_std','gainedPosition_mean','gainedPosition_std','victory_sum', 'podium_sum','pole_sum', 'accident_sum','collision_sum','positionOrder_rainy_mean','rank_rainy_mean','gainedPosition_rainy_mean']
```

Para finalizar la segunda etapa, con el histórico agrupado se crea una variable nueva llamada `ranking` la cual es un ranking final de un piloto por año. Para crear el ranking se tiene en cuenta primero la suma total de puntos `points_sum` y si hay empate a puntos es la suma de victorias `victory_sum` como indican en el reglamento oficial de la Formula 1 en el apartado 7.2 establecido por la FIA. [72]

```
# Script para la creacion del ranking basado en el reglamento de la FIA por piloto y temporada
grouped_race_history['ranking'] = grouped_race_history.groupby('year')['points_sum'].rank(method='min', ascending=False)
grouped_race_history['ranking'] += grouped_race_history.groupby(['year', 'points_sum'])['victory_sum'].rank(method='min', ascending=False)
grouped_race_history['ranking'] =grouped_race_history.groupby('year')['ranking'].rank(method='min', ascending=True)
grouped_race_history = grouped_race_history.sort_values(['year', 'ranking'])
```

7.2.3. Tercera etapa: Creación variables en las que no influya el monoplaza

Comparar a pilotos de Formula 1 es complejo ya que cada piloto conduce un monoplaza diferente y el tener un mejor o peor coche influye mucho en los resultados de un piloto. Para poder reducir el efecto que provoca el coche lo mejor es comparar los pilotos entre compañeros de equipo ya que aunque no son iguales, los monoplazas que conducen son muy parecidos.

Por esta razón se crean 13 nuevas variables agregadas para el compañero de equipo de cada piloto. Por cada escudería suele haber dos pilotos pero cuando un piloto está lesionado o enfermo las escuderías cuentan con un piloto reserva que le sustituye. Esto se ha tenido en cuenta y para solventarlo se realiza la media de la escudería sin contar al propio piloto. Se realiza una media entre todos los pilotos compañeros de equipo ya sea principal o reserva.

Las variables utilizadas fueron las siguientes: `positionOrder_mean`, `grid_mean`, `rank_mean`, `gainedPosition_mean`, `victory_sum`, `podium_sum`, `pole_sum`, `accident_sum`, `collision_sum`, `ranking`, `positionOrder_rainy_mean`, `rank_rainy_mean`, `gainedPosition_rainy_mean`. Estas variables fueron creadas en la segunda etapa.

Se han escogido las variables que recogen la media o suma de resultados incluso de los accidentes y colisiones por el siguiente motivo, si el compañero de equipo tiene muchos accidentes y colisiones puede ser debido a que tenga un monoplaza inestable y difícil de conducir.

Para ello se ha creado una función en Python llamada *calc_teammate_mean* a la que se llama con la función *apply* y tras haber agrupado por año, piloto y escudería. La función *calc_teammate_mean*

selecciona las variables deseadas en el mismo año pero de los otros pilotos de la escudería. Tras ello, se renombran las variables y se realiza la media por escudería como se mencionó previamente para evitar el problema de cuando existen varios compañeros de equipo en una misma temporada. Por último se fusiona con el resto de variables en el histórico agrupado.

```
# Script para la creacion variables en las que no influya el monoplaza

# Crear variables del teammate

COLUMNS_TEAMMATE = ['positionOrder_mean','grid_mean','rank_mean','gainedPosition_mean',
    'victory_sum','podium_sum','pole_sum','accident_sum','collision_sum','ranking','
    positionOrder_rainy_mean','rank_rainy_mean','gainedPosition_rainy_mean']

def calc_teammate_mean(driver,columns_teammate,grouped_race_history):
    driver_year  = driver.year.values[0]
    driver_constructorId  = driver.constructorId.values[0]
    driver_driverId  = driver.driverId.values[0]

    aux=grouped_race_history.loc[ (grouped_race_history.year==driver_year) &
                    (grouped_race_history.constructorId==driver_constructorId) & (
    grouped_race_history.driverId!=driver_driverId),columns_teammate]

    return aux.reset_index()

teammate_mean = grouped_race_history.groupby(['year','driverId','constructorId']).apply
    (calc_teammate_mean,['year', 'driverId', 'constructorId'] +COLUMNS_TEAMMATE,
    grouped_race_history)
# Se elimina porque ya tengo como multiindice el 'year', 'driverId', 'constructorId'
    por ello elimino las columnas de 'year', 'driverId', 'constructorId'
teammate_mean= teammate_mean.drop(columns=['index','year', 'driverId', 'constructorId'
    ])
teammate_mean.columns =  [col + '_teammate' for col in COLUMNS_TEAMMATE]

# Es necesario calcular la media porque hay veces que por lesion entra un tercer piloto
     a sustituir al piloto lesionado entonces no son 2 compañeros sino 3.
teammate_mean=teammate_mean.groupby(['year','driverId','constructorId'])[teammate_mean.
    columns].mean()
teammate_mean= teammate_mean.reset_index()
grouped_race_history = pd.merge(grouped_race_history,teammate_mean,on=['year', '
    driverId','constructorId'])
```

Para facilitar la adición de estas variables a la métrica final se ha optado por crear variables diferencia entre las variables de piloto y las de su compañero de equipo. Estas variables son resultado de restar las variables calculadas de un piloto menos las de su compañero de equipo. Esto facilita la tarea de comparar si un piloto lo ha hecho mejor o peor a igualdad de monoplaza.

Con un simple script por cada columna de compañero de equipo se crea la variable diferencia (variable_diff = variable piloto - variable compañero de equipo). Se añade al histórico a la vez que se elimina la variable de compañero de equipo.

```
# Script Creacion variables diferencia

def select_variables_scoreAI(data,columns_teammate):

    for col in columns_teammate:
        diff_col = col + '_diff'
        data[diff_col] = data[col] - data[col + '_teammate']
        data=data.drop(columns=[col + '_teammate'])

    return data

grouped_race_history=select_variables_scoreAI(grouped_race_history,COLUMNS_TEAMMATE)
```

7.2.4. Cuarta etapa: Escalar y depurar histórico de carreras agrupado

Todas las variables que se disponen son variables numéricas pero para realizar una puntuación final es más sencillo si estas variables están acotadas. El objetivo de escalar todas las variables es acotar las variables en el intervalo [0,1] siendo 0 un valor malo en esa variable y 1 un valor bueno en la variable.

Uno de los problemas que hay que tener en cuenta es que el número de carreras y el número de pilotos en una temporada es variable a lo largo de los años. Por ello variables como número de victorias (`victory_sum`) o número de accidentes (`accident_sum`) se ven afectadas por el número de carreras. Otras variables como la posición final media (`positionOrder_mean`) o el número de posiciones media ganadas en una carrera (`gainedPosition_mean`) están afectadas por el número de pilotos que había en ese año concreto.

Por esa razón se ha decidido escalar los datos por año en el rango 0-1. Al escalar por año se consigue solventar el problema mencionado de la variabilidad en el número de carreras y pilotos durante cada temporada. Siendo la fórmula del escalado por año:

$$x_{esc_i} = \frac{x - x_{min_i}}{x_{max_i} - x_{min_i}}, i = temporada(1950, ..., Actualidad)$$

A cada observación se resta entre el mínimo de ese año y se divide entre la resta entre el máximo y el mínimo de ese año.

Antes de escalar se realizó una exploración de los datos y se observó varios valores ausentes. Por este motivo por cada valor ausente que encontremos se imputa con la media en ese año. Tras ello, mediante la función *MinMaxScaler* de la librería *sklearn* [91] se realiza el escalado (0-1) pero agrupando por año mediante la función *groupby* para lograr que el escalado sea por año.

Con este escalado por año se logra que las métricas sean invariantes en el tiempo ya que si se hubiera realizado un escalado sin agrupar por año por cada nuevos datos las métricas antiguas se modificarían.

Después de realizar el escalado se observó que seguían habiendo valores ausentes. Esto era provocado porque en temporadas antiguas había variables como la posición en alcanzar la vuelta rápida (`rank`) que no se medían entonces al realizar la imputación con la media del año seguía siendo un valor nulo. Por tanto, esta vez se ha optado por imputar por una constante que va a ser 0.5 ya que como está escalado en el rango 0-1, 0.5 se trata del valor medio del rango.

```
# Script para escalar y depurar el historico de carreras agrupado
from sklearn.preprocessing import MinMaxScaler
# Los valores ausentes se imputan con la media del  año en esa variable
grouped_race_history_scaled =grouped_race_history.groupby('year').transform(lambda x: x
    .fillna(x.mean()))
grouped_race_history_scaled['year']=grouped_race_history['year']
# Se escalan los datos por temporada
scaler = MinMaxScaler()
grouped_race_history_scaled = grouped_race_history_scaled.groupby('year').apply(lambda
    x: pd.DataFrame(scaler.fit_transform(x), columns=x.columns, index=x.index))
grouped_race_history_scaled = grouped_race_history_scaled.reset_index(level=[0], drop=
    True)
grouped_race_history_scaled = grouped_race_history_scaled.fillna(0.5)
```

Después de tener todas las variables acotadas en el intervalo [0,1] surge el problema de que hay variables como la posición media en carrera (`positionOrder_mean`) o el número de accidentes (`accident_sum`) que cuanto mayor valor de las variables peor piloto es. Al contrario ocurre con la variable número de victorias (`victory_sum`) que cuanto mayor valor de la variable mejor piloto es.

Por ese motivo es necesario transformar las variables que son inversamente proporcionales a la calidad del piloto, como la posición media en carrera (`positionOrder_mean`) o el número de accidentes (`accident_sum`). Esto se hace para que todas las variables escaladas el 0 indique un valor malo y 1 un valor bueno en la variable. Simplemente con seleccionar las columnas inversamente proporcionales y realizar la operación 1 - variable ya se obtiene la variable en el rango 0-1 y que es directamente proporcional a la calidad del piloto.

```
# Script para convertir las variables inversamente proporcionales a directamente proporcionales
inverse_columns = [ 'positionOrder_mean', 'positionOrder_std','grid_mean', 'grid_std', 'rank_mean', 'rank_std', 'accident_sum','collision_sum', 'ranking',
'positionOrder_rainy_mean', 'rank_rainy_mean', 'positionOrder_mean_diff', 'grid_mean_diff', 'rank_mean_diff', 'accident_sum_diff', 'collision_sum_diff','ranking_diff','positionOrder_rainy_mean_diff','rank_rainy_mean_diff']
grouped_race_history_scaled[inverse_columns] = 1 - grouped_race_history_scaled[inverse_columns]
grouped_race_history_scaled=grouped_race_history_scaled.drop('year',axis=1)
```

7.2.5. Quinta etapa: Creación de métricas parciales

En esta etapa se crean las métricas parciales definidas en el apartado 7.1, estas métricas consisten en una media ponderada con pesos escogidos en función de la importancia de cada variable en cada métrica parcial.

Por lo tanto, lo primero es definir que variables influyen en cada métrica y en que cantidad para asignarles un peso determinado.

- `KPI_vs_compañero_equipo`: Se han escogido para esta métrica todas las variables de diferencia con el compañero de equipo ya sean de resultados, accidentes o en lluvia.

  - `positionOrder_mean_diff`: Diferencia media anual de la posición final en cada carrera respecto a su compañero de equipo. Peso: 10.
  - `grid_mean_diff`: Diferencia media anual de la posición inicial en cada carrera respecto a su compañero de equipo. Peso: 10.
  - `rank_mean_diff`: Diferencia media anual del ranking de alcanzar la vuelta rápida en carrera respecto a su compañero de equipo. Peso: 7.
  - `gainedPosition_mean_diff`: Diferencia media anual de las posiciones ganadas en carrera respecto a su compañero de equipo. Peso: 10.
  - `victory_sum_diff`: Diferencia en la suma anual de victorias respecto a su compañero de equipo. Peso: 10.
  - `podium_sum_diff`: Diferencia en la suma anual de podios respecto a su compañero de equipo. Peso: 10.
  - `pole_sum_diff`: Diferencia en la suma anual de poles respecto a su compañero de equipo. Peso: 10.
  - `accident_sum_diff`: Diferencia en la suma anual de accidentes provocados respecto a su compañero de equipo. Peso: 5.
  - `collision_sum_diff`: Diferencia en la suma anual de colisiones múltiples en las que ha estado involucrado respecto a su compañero de equipo. Peso: 5.
  - `ranking_diff`: Diferencia de la posición final del ranking de pilotos respecto a su compañero de equipo. Peso: 10.
  - `positionOrder_rainy_mean_diff`: Diferencia media anual de la posición final en cada carrera con lluvia respecto a su compañero de equipo. Peso: 5.
  - `rank_rainy_mean_diff`: Diferencia media anual del ranking en alcanzar la vuelta rápida en cada carrera con lluvia respecto a su compañero de equipo. Peso: 5.
  - `gainedPosition_rainy_mean_diff`: Diferencia media anual de las posiciones ganadas en carrera con lluvia respecto a su compañero de equipo. Peso: 5.

  La fórmula para crear `KPI_vs_compañero_equipo` es la siguiente:

$$\begin{aligned}\texttt{KPI\_vs\_compañero\_equipo} = \frac{1}{\sum_{i=1}^{p}\beta_i} \cdot (&\beta_1 \cdot \texttt{positionOrder\_mean\_diff} \\ &+ \beta_2 \cdot \texttt{grid\_mean\_diff} \\ &+ \beta_3 \cdot \texttt{rank\_mean\_diff} \\ &+ \beta_4 \cdot \texttt{gainedPosition\_mean\_diff} \\ &+ \beta_5 \cdot \texttt{victory\_sum\_diff} \\ &+ \beta_6 \cdot \texttt{podium\_sum\_diff} \\ &+ \beta_7 \cdot \texttt{pole\_sum\_diff} \\ &+ \beta_8 \cdot \texttt{accident\_sum\_diff} \\ &+ \beta_9 \cdot \texttt{collision\_sum\_diff} \\ &+ \beta_{10} \cdot \texttt{ranking\_diff} \\ &+ \beta_{11} \cdot \texttt{positionOrder\_rainy\_mean\_diff} \\ &+ \beta_{12} \cdot \texttt{rank\_rainy\_mean\_diff} \\ &+ \beta_{13} \cdot \texttt{gainedPosition\_rainy\_mean\_diff})\end{aligned} \tag{7.1}$$

```
# Script para crear Resultados respecto al piloto de su misma escuderia
results_vs_teammate_weights_dict = {'positionOrder_mean_diff':10,'grid_mean_diff':10, 'rank_mean_diff':7, 'gainedPosition_mean_diff':10,'victory_sum_diff':10, 'podium_sum_diff':10, 'pole_sum_diff':10,'accident_sum_diff':5, 'collision_sum_diff':5, 'ranking_diff':10,'positionOrder_rainy_mean_diff':5,'rank_rainy_mean_diff':5,'gainedPosition_rainy_mean_diff':5}
results_vs_teammate_weights_list = [results_vs_teammate_weights_dict[col] for col in results_vs_teammate_weights_dict.keys()]

partial_metrics['KPI_vs_compañero_equipo'] = 100*np.average(grouped_race_history_scaled[results_vs_teammate_weights_dict.keys()], axis=1, weights=results_vs_teammate_weights_list)
```

- `KPI_lluvia`: Se han escogido para esta métrica todas las variables que afecta la lluvia tanto únicamente del piloto como las de diferencia con el compañero de equipo.

  - `positionOrder_rainy_mean`: Media anual de la posición final en cada carrera con lluvia. Peso: 5.
  - `rank_rainy_mean`: Media anual del ranking de alcanzar la vuelta rápida en carrera con lluvia. Peso: 5.
  - `gainedPosition_rainy_mean`: Media anual de las posiciones ganadas en carrera con lluvia. Peso: 5.
  - `positionOrder_rainy_mean_diff`: Diferencia media anual de la posición final en cada carrera con lluvia respecto a su compañero de equipo. Peso: 5.
  - `rank_rainy_mean_diff`: Diferencia media anual del ranking en alcanzar la vuelta rápida en cada carrera con lluvia respecto a su compañero de equipo. Peso: 5.
  - `gainedPosition_rainy_mean_diff`: Diferencia media anual de las posiciones ganadas en carrera con lluvia respecto a su compañero de equipo. Peso: 5.

  La fórmula para crear `KPI_lluvia` es la siguiente:

$$\begin{aligned}\texttt{KPI\_lluvia} = \frac{1}{\sum_{i=1}^{p}\beta_i} \cdot (&\beta_1 \cdot \texttt{positionOrder\_rainy\_mean} \\ &+ \beta_2 \cdot \texttt{rank\_rainy\_mean} \\ &+ \beta_3 \cdot \texttt{gainedPosition\_rainy\_mean} \\ &+ \beta_4 \cdot \texttt{positionOrder\_rainy\_mean\_diff} \\ &+ \beta_5 \cdot \texttt{rank\_rainy\_mean\_diff} \\ &+ \beta_6 \cdot \texttt{gainedPosition\_rainy\_mean\_diff})\end{aligned} \tag{7.2}$$

```
# Script para crear Resultados respecto a carreras con lluvia
results_with_rain_weights_dict = {'positionOrder_rainy_mean':5,'rank_rainy_mean':5,
    'gainedPosition_rainy_mean':5, 'positionOrder_rainy_mean_diff':5,'
    rank_rainy_mean_diff':5,'gainedPosition_rainy_mean_diff':5}

rainy_results_weights_list = [rainy_results_weights_dict[col] for col in
    rainy_results_weights_dict.keys()]
partial_metrics['KPI_lluvia'] = 100*np.average(grouped_race_history_scaled[
    rainy_results_weights_dict.keys()], axis=1, weights=rainy_results_weights_list)
```

- `KPI_evitar_accidentes`: Se han escogido para esta métrica todas las variables relacionadas con accidentes y colisiones tanto únicamente del piloto como las de diferencia con el compañero de equipo.

  - `accident_sum`: Suma anual de accidentes provocados. Peso: 5.
  - `collision_sum` : Suma anual de colisiones múltiples en las que ha estado involucrado. Peso: 5.
  - `accident_sum_diff`: Diferencia en la suma anual de accidentes provocados respecto a su compañero de equipo. Peso: 5.
  - `collision_sum_diff`: Diferencia en la suma anual de colisiones múltiples en las que ha estado involucrado respecto a su compañero de equipo. Peso: 5.

  La fórmula para crear `KPI_evitar_accidentes` es la siguiente:

$$\begin{aligned}\texttt{KPI\_evitar\_accidentes} = \frac{1}{\sum_{i=1}^{p}\beta_i} \cdot (&\beta_1 \cdot \texttt{accident\_sum} \\ &+ \beta_2 \cdot \texttt{collision\_sum} \\ &+ \beta_3 \cdot \texttt{accident\_sum\_diff} \\ &+ \beta_4 \cdot \texttt{collision\_sum\_diff})\end{aligned} \tag{7.3}$$

```
# Script para crear Resultados respecto a evitar accidentes
skills_avoid_accidents_weights_dict = { 'accident_sum':5, 'collision_sum':5,'
    accident_sum_diff':5,'collision_sum_diff':5}

skills_avoid_accidents_weights_list = [skills_avoid_accidents_weights_dict[col] for
     col in skills_avoid_accidents_weights_dict.keys()]
partial_metrics['KPI_evitar_accidentes'] = 100*np.average(
    grouped_race_history_scaled[skills_avoid_accidents_weights_dict.keys()], axis
    =1, weights=skills_avoid_accidents_weights_list)
```

- `KPI_resultados_generales`: Se han escogido para esta métrica todas las variables de piloto ya sean de resultados, accidentes o en lluvia.

  - `positionOrder_mean`: Media anual de la posición final en cada carrera respecto a su compañero de equipo. Peso: 10.
  - `positionOrder_std`: Desviación típica anual de la posición final en cada carrera. Peso: 3.

- `grid_mean`: Media anual de la posición inicial en cada carrera respecto a su compañero de equipo. Peso: 10.
- `grid_std`: Desviación típica anual de la posición inicial en cada carrera. Peso: 3.
- `rank_mean`: Media anual del ranking de alcanzar la vuelta rápida en carrera respecto a su compañero de equipo. Peso: 7.
- `rank_std`: Desviación típica anual del ranking de alcanzar la vuelta rápida en carrera. Peso: 3.
- `gainedPosition_mean`: Media anual de las posiciones ganadas en carrera respecto a su compañero de equipo. Peso: 10.
- `gainedPosition_std`: Desviación típica anual de las posiciones ganadas en carrera. Peso: 3.
- `victory_sum`: Suma anual de victorias respecto a su compañero de equipo. Peso: 10.
- `podium_sum`: Suma anual de podios respecto a su compañero de equipo. Peso: 10.
- `pole_sum`: Suma anual de poles respecto a su compañero de equipo. Peso: 10.
- `accident_sum`: Suma anual de accidentes provocados respecto a su compañero de equipo. Peso: 5.
- `collision_sum`: Suma anual de colisiones múltiples en las que ha estado involucrado respecto a su compañero de equipo. Peso: 5.
- `ranking`: Posición final del ranking de pilotos respecto a su compañero de equipo. Peso: 10.
- `positionOrder_rainy_mean`: Media anual de la posición final en cada carrera con lluvia respecto a su compañero de equipo. Peso: 5.
- `rank_rainy_mean`: Media anual del ranking en alcanzar la vuelta rápida en cada carrera con lluvia respecto a su compañero de equipo. Peso: 5.
- `gainedPosition_rainy_mean`: Media anual de las posiciones ganadas en carrera con lluvia respecto a su compañero de equipo. Peso: 5.

La fórmula para crear `KPI_resultados_generales` es la siguiente:

$$\begin{aligned}\texttt{KPI\_resultados\_generales} = \frac{1}{\sum_{i=1}^{p}\beta_i} \cdot (&\beta_1 \cdot \texttt{positionOrder\_mean} \\ &+ \beta_2 \cdot \texttt{positionOrder\_std} \\ &+ \beta_3 \cdot \texttt{grid\_mean} \\ &+ \beta_4 \cdot \texttt{grid\_std} \\ &+ \beta_5 \cdot \texttt{rank\_mean} \\ &+ \beta_6 \cdot \texttt{rank\_std} \\ &+ \beta_7 \cdot \texttt{gainedPosition\_mean} \\ &+ \beta_8 \cdot \texttt{gainedPosition\_std} \\ &+ \beta_9 \cdot \texttt{victory\_sum} \\ &+ \beta_{10} \cdot \texttt{podium\_sum} \\ &+ \beta_{11} \cdot \texttt{pole\_sum} \\ &+ \beta_{12} \cdot \texttt{accident\_sum} \\ &+ \beta_{13} \cdot \texttt{collision\_sum} \\ &+ \beta_{14} \cdot \texttt{ranking} \\ &+ \beta_{15} \cdot \texttt{positionOrder\_rainy\_mean} \\ &+ \beta_{16} \cdot \texttt{rank\_rainy\_mean} \\ &+ \beta_{17} \cdot \texttt{gainedPosition\_rainy\_mean})\end{aligned} \tag{7.4}$$

```
# Script para crear Resultados respecto al resto de pilotos de la temporada
results_vs_all_drivers_weights_dict = {'positionOrder_mean':10, 'positionOrder_std'
    :3, 'grid_mean':10, 'grid_std':3,'rank_mean':7, 'rank_std':3, '
    gainedPosition_mean':10, 'gainedPosition_std':3,'victory_sum':10,'podium_sum'
    :10, 'pole_sum':10,'accident_sum':5, 'collision_sum':5,'ranking':10,'
    positionOrder_rainy_mean':5,'rank_rainy_mean':5,'gainedPosition_rainy_mean':5}
results_vs_all_drivers_weights_list = [results_vs_all_drivers_weights_dict[col] for
     col in results_vs_all_drivers_weights_dict.keys()]
partial_metrics['KPI_resultados_generales'] = 100*np.average(
    grouped_race_history_scaled[results_vs_all_drivers_weights_dict.keys()], axis
    =1, weights=results_vs_all_drivers_weights_list)
```

- `KPI_regularidad`: Se han escogido para esta métrica todas las variables relacionadas con la regularidad y consistencia de un piloto en sus resultados.

  - `positionOrder_std`: Desviación típica anual de la posición final en cada carrera. Peso: 5.
  - `grid_std`: Desviación típica anual de la posición inicial en cada carrera. Peso: 5.
  - `rank_std`: Desviación típica anual del ranking de alcanzar la vuelta rápida en carrera. Peso: 5.
  - `gainedPosition_std`: Desviación típica anual de las posiciones ganadas en carrera. Peso: 5.

  La fórmula para crear `KPI_regularidad` es la siguiente:

$$\begin{aligned}\texttt{KPI\_regularidad} = \frac{1}{\sum_{i=1}^{p}\beta_i} \cdot (&\beta_1 \cdot \texttt{positionOrder\_std} \\ &+ \beta_2 \cdot \texttt{grid\_std} \\ &+ \beta_3 \cdot \texttt{rank\_std} \\ &+ \beta_4 \cdot \texttt{gainedPosition\_std})\end{aligned} \tag{7.5}$$

```
# Script para crear Resultados respecto a la regularidad de un piloto
consistency_weights_dict = {'positionOrder_std':5, 'grid_std':5, 'rank_std':5,'
    gainedPosition_std':5}

consistency_weights_list = [consistency_weights_dict[col] for col in
    consistency_weights_dict.keys()]
partial_metrics['KPI_regularidad'] = 100*np.average(grouped_race_history_scaled[
    consistency_weights_dict.keys()], axis=1, weights=consistency_weights_list)
```

Existen varias variables que afectan a varias métricas parciales como son `accident_sum` que forma parte de la métrica de resultados del piloto `KPI_resultados_generales` y en la métrica de habilidad evitando accidentes `KPI_evitar_accidentes`. Para que estas variables no afecten en mayor medida simplemente por pertenecer a dos medidas parciales lo que se ha realizado es reducir el peso de las variables que se repitan en varias métricas parciales como en este caso.

7.2.6. Sexta etapa: Creación de métrica global *scoreAI*

Por último tras crear las métricas parciales que componen la métrica global de `scoreAI` es necesario asignar pesos para realizar la media ponderada de métricas parciales. Como se definió en el apartado 7.1 se debe dar mayor peso a la métrica parcial de resultados respecto al compañero de equipo (`KPI_vs_compañero_equipo` peso 4.5), seguido de habilidades evitando accidentes (`KPI_evitar_accidentes` peso 2), después las métricas de resultados con lluvia (`KPI_lluvia` peso 1.5), y resultados frente al resto de pilotos (`KPI_resultados_generales` peso 1.5) y por último un menor peso a la consistencia del piloto (`KPI_regularidad` peso 0.5).

Por tanto la fórmula para crear *scoreAI* es la siguiente:

$$
\begin{aligned}
ScoreAI = \frac{1}{\sum_{i=1}^{p} \beta_i} \cdot (&\beta_1 \cdot \texttt{KPI\_vs\_compañero\_equipo} \\
&+ \beta_2 \cdot \texttt{KPI\_lluvia} \\
&+ \beta_3 \cdot \texttt{KPI\_evitar\_accidentes} \\
&+ \beta_4 \cdot \texttt{KPI\_resultados\_generales} \\
&+ \beta_5 \cdot \texttt{KPI\_regularidad})
\end{aligned}
\tag{7.6}
$$

Siendo β_i los pesos de cada variable, el script de Python para calcular la métrica *scoreAI* se realiza con la función *average* de la librería *numpy*.

```
# Sexta etapa: Script para crear Metrica global scoreAI
scoreAI = partial_metrics.copy()
scoreAI_weights_dict = {'KPI_vs_compañero_equipo':4.5,  'KPI_lluvia':1.5,'
    KPI_evitar_accidentes':2,
'KPI_resultados_generales':1.5, 'KPI_regularidad':0.5 }
scoreAI_weights_list = [scoreAI_weights_dict[col] for col in scoreAI_weights_dict.keys
    ()]
scoreAI['scoreAI'] = np.average(partial_metrics, axis=1, weights=scoreAI_weights_list)
```

7.2.7. Séptima etapa: Creación de la Tabla TemporadaPiloto

Tras crear la métrica *scoreAI* y las métricas parciales se deben agregar los datos por piloto y año. Esto se realiza porque existen pilotos que han competido en varias escuderías en un mismo año. El caso más reciente es el de George Russell en 2020 pilotando para Williams y para Mercedes. Actualmente no suele ser muy común pero en la década de los 50 solía ser bastante habitual. En estos casos se ha agregado la información de la siguiente manera para las variables de suma de puntos (`points_sum`), número de victorias (`victory_sum`) se agregan mediante la suma. Para la métrica *scoreAI* y el resto de métricas parciales se agregan con la media entre las escuderías que ha pilotado en ese año.

```
#Script para agregar los datos de pilotos que en un misma temporada compitieron en
    varias escuderias
temporadaPiloto = scoreAI.merge(Piloto, how='left', left_on='driverId', right_on='
    driverId')
temporadaPiloto = temporadaPiloto.merge(Escuderia, how='left', left_on='constructorId',
     right_on='constructorId')
temporadaPiloto = temporadaPiloto.loc[:,['year','driverRef','constructorRef','
    victory_sum','points_sum','scoreAI','KPI_vs_compañero_equipo','KPI_lluvia','
    KPI_evitar_accidentes','KPI_resultados_generales','KPI_regularidad']]
temporadaPiloto = temporadaPiloto.groupby(['year', 'driverRef']).agg({
    'constructorRef': lambda x: ', '.join(x),
    'points_sum': 'sum',
    'victory_sum': 'sum',
    'scoreAI': 'mean',
    'KPI_vs_compañero_equipo': 'mean',
    'KPI_lluvia': 'mean',
    'KPI_evitar_accidentes': 'mean',
    'KPI_resultados_generales': 'mean',
    'KPI_regularidad': 'mean'
}).reset_index()
```

Después de tener los datos en el formato deseado: una fila para cada piloto y año, se vuelve a crear el ranking anual de pilotos. Para crear el ranking se tiene en cuenta primero la suma total de puntos `points_sum` y si hay empate a puntos es la suma de victorias `victory_sum` como indican en el reglamento oficial de la Formula 1 en el apartado 7.2 establecido por la FIA. [72]

Por último, después de calcular el ranking se crea la variable binaria `CampeonatoPiloto`. Esta variable vale 1 si el piloto ha obtenido la primera posición en el ranking, 0 en caso contrario. Hasta que no empiece una nueva temporada no se otorga el mundial de pilotos así se evita dar el título antes de que finalice la temporada.

```
temporadaPiloto.columns = ['year', 'driverRef', 'constructorRef', 'points_sum','victory_sum', 'scoreAI',
                    'KPI_vs_compañero_equipo', 'KPI_lluvia', 'KPI_evitar_accidentes',
                    'KPI_resultados_generales', 'KPI_regularidad']
# Se crea el ranking anual de pilotos
temporadaPiloto['ranking'] = temporadaPiloto.groupby('year')['points_sum'].rank(method='min', ascending=False)
temporadaPiloto['ranking'] += temporadaPiloto.groupby(['year', 'points_sum'])['victory_sum'].rank(method='min', ascending=False)
temporadaPiloto['ranking'] =temporadaPiloto.groupby('year')['ranking'].rank(method='min', ascending=True)
# Hasta que no empiece una nueva temporada no se otorga el campeonato de pilotos, esto evita dar el titulo antes de que finalice una temporada
currentYear = max(Year.year)
temporadaPiloto['CampeonatoPiloto'] = ((temporadaPiloto['ranking'] == 1) & (temporadaPiloto['year']<currentYear)).astype(int)
temporadaPiloto =temporadaPiloto.drop('victory_sum',axis=1)
```

7.2.8. Octava etapa: Creación de la Tabla TemporadaEscuderia

Para crear la tabla de TemporadaEscuderia se agregan por año y escudería las variables de suma de puntos (`points_sum`) y número de victorias (`victory_sum`). Se agregan mediante la suma.

Después se crea el ranking anual de escuderías. Este ranking se calcula sumando los puntos de todos los pilotos de la escudería en un año. Tras crear el ranking anual de escuderías se crea la variable `CampeonatoConstructores`. Esta variable vale 1 si la escudería ha obtenido la primera posición en el ranking, 0 en caso contrario. Se ha tenido en cuenta que el mundial de constructores no se empezó a otorgar hasta el año 1958. Por último se ha creado la variable `N_GPs`, la cuál indica el número de grandes premios disputados por una escudería en un año.

```
# Octava etapa: Creacion de la Tabla TemporadaEscuderia

temporadaEscuderia = scoreAI.merge(Piloto, how='left', left_on='driverId', right_on='driverId')
temporadaEscuderia = temporadaEscuderia.merge(Escuderia, how='left', left_on='constructorId', right_on='constructorId')
temporadaEscuderia = temporadaEscuderia.loc[:,['year','driverRef','constructorRef','victory_sum','points_sum']]
temporadaEscuderia = temporadaEscuderia.groupby(['year', 'constructorRef']).agg({
    'points_sum': 'sum',
    'victory_sum': 'sum',
}).reset_index()
temporadaEscuderia.columns = ['year', 'constructorRef', 'points_sum','victory_sum']
# Se crea el ranking anual de escuderias
temporadaEscuderia['ranking'] = temporadaEscuderia.groupby('year')['points_sum'].rank(method='min', ascending=False)
temporadaEscuderia['ranking'] += temporadaEscuderia.groupby(['year', 'points_sum'])['victory_sum'].rank(method='min', ascending=False)
temporadaEscuderia['ranking'] =temporadaEscuderia.groupby('year')['ranking'].rank(method='min', ascending=True)
currentYear = max(Year.year)
# El mundial de contructores no comenzó a otorgarse hasta el año 1958
temporadaEscuderia['CampeonatoConstructores'] = ((temporadaEscuderia['ranking'] == 1) & (temporadaEscuderia['year']<currentYear)
& (temporadaEscuderia['year']>=1958)).astype(int)

temporadaEscuderia =temporadaEscuderia.drop('victory_sum',axis=1)
contructorGPs =race_history.loc[:,['year','raceId','constructorRef']]
contructorGPs =contructorGPs.drop_duplicates()
contructorGPs = contructorGPs.groupby(['year','constructorRef']).count().reset_index()
contructorGPs.columns=['year','constructorRef','N_GPs']
temporadaEscuderia = pd.merge(temporadaEscuderia,contructorGPs, on = ['year','constructorRef'])
```

Capítulo 8

Fórmula 1 BI: Aplicación para la visualización de datos de F1

En este capítulo se presenta *Fórmula 1 BI* [1], una aplicación de Business Intelligence para la visualización de datos de la Fórmula 1. *Fórmula 1 BI* consta de cinco "Dashboards" realizados con la herramienta Power BI. Se han seguido las guías de diseño de cuadros de mando descritas en el apartado 5.5. Adicionalmente, la aplicación cuenta con un manual de ayuda al usuario integrado dentro de la propia aplicación.

8.1. La aplicación *Fórmula 1 BI*

Se ha creado una aplicación con seis páginas diferentes:

- **Página de Inicio**: Es el menú inicial de la aplicación, dispone de cinco botones con imágenes que llevan a cada uno de los cinco dashboards realizados. Más detalles en el apartado 8.1.1.
- **Dashboard Piloto**: Muestra información gráfica de pilotos. El usuario selecciona en qué circuitos y en qué temporadas desea visualizar la información de un único piloto. Más detalles en el apartado 8.1.2.
- **Dashboard Escudería**: Muestra información gráfica de escuderías. El usuario selecciona en qué circuitos y en qué temporadas desea visualizar la información de una única escudería. Más detalles en el apartado 8.1.3.
- **Dashboard Circuito**: Muestra información gráfica de circuitos. El usuario selecciona en qué temporadas desea visualizar la información de un único circuito. Más detalles en el apartado 8.1.4.
- **Dashboard Temporada**: Muestra información gráfica resumida de pilotos, escuderías y circuitos de todas las temporadas . El usuario selecciona en qué temporadas desea visualizar la información de pilotos, circuitos y escuderías. Más detalles en el apartado 8.1.5.
- **Dashboard 1 vs 1**: Muestra información gráfica para comparar dos pilotos. El usuario selecciona en qué temporadas y de qué pilotos desea visualizar la información. Más detalles en el apartado 8.1.6.

8.1.1. Página de Inicio

En la Figura 8.1 se muestra la página de presentación de la aplicación. En ella se observa el nombre de la aplicación junto con cinco botones con imágenes que le llevan a cada uno de los cinco cuadro de mandos creados, Piloto, Escudería, Circuito, Temporada y 1 vs 1.

Los botones han sido creados con navegadores de página también llamados "Page Bookmarks".

En la esquina superior se sitúa el botón de más detalles (ℹ) que permite acceder al manual de ayuda del usuario, el cuál es detallado en el apartado 8.2.

[1] Enlace a la visualización de Power BI

Figura 8.1: Página de Inicio de *Fórmula 1 BI*

8.1.2. Dashboard Piloto

Este dashboard se encarga de visualizar datos de un piloto concreto, en uno o más circuitos escogidos y en un periodo de años seleccionados.

Teniendo en cuenta el mockup con el diseño para el cuadro de mando Piloto (ver Figura 5.7) y las guías de diseño de cuadros de mando explicadas en el apartado 5.5 junto con la disponibilidad y calidad de los datos analizados en el Capítulo 6 de la ETL, se ha implementado el Dashboard de Piloto, (ver Figura 8.2).

Figura 8.2: Dashboard Piloto

Siguiendo la estructura basada en agrupaciones explicada en el aparatado 5.5.2, se han creado ocho recuadros para representar información similar en cada recuadro y así ordenar el dashboard y que la

información se represente de una manera más clara y organizada.

Encabezado del cuadro de mando Piloto

En la parte superior junto al logo de la Fórmula 1 se encuentra el menú para navegar entre los diferentes cuadros de mando, siempre en un color más claro está resaltado el cuadro de mando que está seleccionado. A la derecha del menú se encuentran los filtros específicos para el dashboard de Piloto. Estos filtros son el filtro de piloto que permite seleccionar de manera única un piloto, el filtro de circuito que permite seleccionar de una manera múltiple los circuitos deseados y por último un slider para seleccionar el rango de años deseado.

Todos los filtros junto con el menú de selección de dashboard se encuentran en la parte superior en todos los cuadros de mando para que la aplicación tenga consistencia interna entre dashboards y que sea de fácil aprendizaje para el usuario. En la esquina superior derecha se sitúa el botón de más detalles (ℹ) que permite acceder al manual de ayuda del usuario, el cuál es detallado en el apartado 8.2.

Recuadro de información general del piloto

Siguiendo con el patrón Z mencionado en el apartado 5.5, en la zona dónde primero se visualiza del cuadro de mando se sitúa un recuadro con la información general del piloto, con la imagen (obtenida mediante web scraping en la Wikipedia), bandera de la nacionalidad del piloto, abreviatura del piloto, número del monoplaza y nombre.

Recuadro de resultados destacados

A la derecha de la información general del piloto se muestran con etiquetas los resultados destacados de ese piloto en los circuitos seleccionados en el rango de años elegido. Los resultados destacados mostrados son los siguientes:

- Campeonatos: Número de campeonatos de pilotos de la Fórmula 1 obtenido por el piloto en el rango de años elegido.
- Nº GPs: Número de Grandes Premios disputados por el piloto en los circuitos seleccionados y en el rango de años elegido.
- Victorias: Número de victorias del piloto en los circuitos seleccionados y en el rango de años elegido.
- Podios: Número de podios del piloto en los circuitos seleccionados y en el rango de años elegido. Se ha creado un tooltip si se pasa el ratón por encima de la etiqueta. Este tooltip muestra la distribución de los podios mediante un **diagrama de barras (bar chart)** indicando las veces que ha quedado en cada posición del podio. Como color de las barras se ha escogido el color oro (primer puesto), plata (segundo puesto) y bronce (tercer puesto),(ver Figura 8.3).
- Poles: Número de poles del piloto en los circuitos seleccionados y en el rango de años elegido.

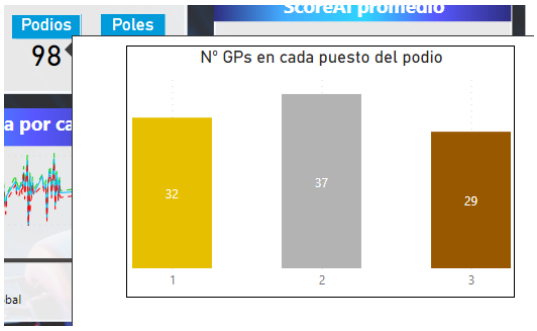


Figura 8.3: Tooltip en la etiqueta de Podios

Recuadro de *scoreAI* promedio

A la derecha de los resultados destacados se muestra una de las métricas a las que el cliente le ha dado mucha importancia, la métrica *scoreAI* detallada en el Capítulo 7. Se ha creado mediante un **gráfico de gauge (gauge plot)** también llamado **medidor radial** en el cuál para crear el gráfico hay que indicar cual es el valor mínimo y máximo que puede alcanzar. Para ello se crearon mediante DAX dos medidas muy simples `minScoreAI` y `maxScoreAI` siendo la primera 0 y la segunda 100. El valor mostrado es el promedio de la métrica *scoreAI* en el rango de años elegido. Como color del gráfico se ha optado seguir la metáfora del semáforo según el valor de la variable, si el valor de la métrica es bajo (indicador de mal rendimiento del piloto) el color del gráfico es cercano al rojo, valores altos de la variable *scoreAI* (indicador de buen rendimiento del piloto) corresponden con colores similares al verde, para rendimientos medios corresponde con colores similares a amarillo. También se indica con una marca azul el valor de la mejor temporada contando todos los pilotos en los años seleccionados.

Se ha creado un tooltip si se pasa el ratón por encima del gráfico. Este tooltip muestra las métricas parciales mediante un **gráfico de radar (radar chart / spider chart)** indicando el valor promedio de estas métricas del piloto en los años seleccionados, (ver Figura 8.4).

Figura 8.4: Tooltip en el gráfico de *scoreAI* promedio

Recuadro de puntos por año

Debajo de estos tres recuadros superiores aparece el recuadro de puntos por año y de velocidad máxima por carrera. En el recuadro de puntos por año se representa mediante un **gráfico de barras (bar chart)** el total de puntos obtenido por el piloto en los circuitos seleccionados y en el rango de años elegido. En el eje X se sitúa el año mientras que en el eje Y la suma de puntos. Para mostrar más información, como leyenda se incluye la escudería donde pilotaba en ese año. Como información adicional mediante un tooltip cuando se pasa el ratón por encima de una barra de un año se indica a mayores el ranking final en el que quedó el piloto ese año junto con el número de puntos obtenidos en ese año, la escudería donde pilotó y el año concreto de la barra, (ver Figura 8.5). Se ha utilizado un diagrama de barras en vez de un gráfico de líneas porque así se incluye con el color de las barras la escudería donde pilotó en ese año.

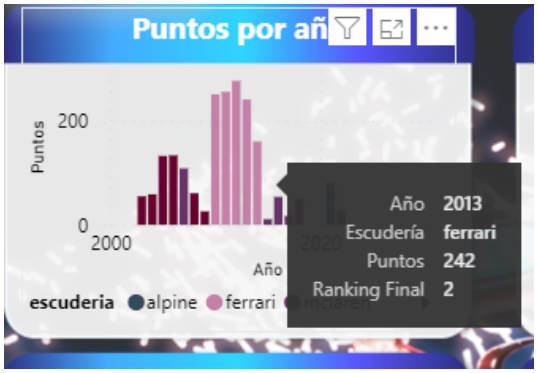


Figura 8.5: Tooltip en el gráfico de barras de puntos por año

Recuadro de máxima velocidad media por vuelta por carrera

A la derecha del gráfico de puntos por año se sitúa el gráfico de máxima velocidad media por vuelta por carrera, para ello se utiliza un **gráfico de líneas (line plot)** ya que se representa en el Eje Y la variable máxima de la velocidad media por vuelta alcanzada por el piloto en cada carrera de los circuitos seleccionados en el rango de años seleccionado. En el X se representa la fecha. Como color se ha escogido el mismo que el color de fondo del dashboard para crear armonía. En el tooltip se indica el valor de la velocidad máxima, la fecha, el nombre del circuito y la meteorología de esa carrera.

Recuadro de culpable de los abandonos

Por último, en la fila inferior se representa información más específica y más detallada. En el recuadro inferior izquierdo está relacionado con los abandonos del piloto en los circuitos y años seleccionados. Se han creados dos gráficos relacionados con los abandonos, el primero para representar quien fue el culpable del abandono (error humano o error mecánico) y el segundo para ver la evolución en los años de esos abandonos. Para no sobrecargar el dashboard e introducir en tamaño pequeño los gráficos se ha decidido introducir en el recuadro dos botones (en Power BI llamados navegadores de marcadores) para intercambiar entre cada uno de los gráficos como se observa en la Figura 8.6. Para el gráfico de culpable de abandonos se ha elegido un **gráfico de donut (donut chart)**, no es un tipo de gráfico muy recomendado cuando se tienen muchas categorías pero en este caso solo se disponen de dos categorías (culpable=escuderia y culpable = piloto). Como colores se han elegido un color gris (igual que el fondo de dashboard Escudería) para errores de escudería (errores mecánicos) y el color azul (igual que el fondo de dashboard Piloto) para errores de piloto (errores humanos). Dentro del gráfico de donut se ha incluido la suma de abandonos entre los dos posibles culpables. Para el gráfico de evolución por años de los abandonos se ha realizado un **gráfico de barras apiladas (stacked bar chart)** apilando por año los abandonos según el culpable del accidente (piloto o escudería) y siguiendo el mismo código de colores que para el gráfico de donut por consistencia interna.

Figura 8.6: Gráficos de culpable de abandonos y de evolución de abandonos

Recuadro de resumen por año

En la fila inferior de la columna central del sistema de cuadrícula se sitúa el resumen por año del piloto en los años seleccionados. Esta información está resumida en formato tabla y al ser muy detallada se ha optado por situarla en la parte inferior del cuadro de mando. Para no sobrecargar el cuadro de mando se ha introducido dos botones para intercambiar entre los resultados obtenidos del piloto y las métricas calculadas del piloto, (ver Figura 8.7). Para ambos resúmenes se ha optado por una **tabla** ya que se muestran muchas variables.

Para el apartado de resultados se representa el año, escudería donde pilotó, el número de puntos, número de victorias, podios, poles y ranking final obtenido. Para los puntos se ha incluido un gráfico de barras dentro de la columna. Para el ranking final se ha incluido un formato condicional que colorea el fondo de color oro, plata o bronce si el piloto ha finalizado en el ranking final primero, segundo o tercero respectivamente.

Para las métricas se ha representado junto al año y la escudería donde pilotó, la métrica *scoreAI* y las métricas parciales `KPI_vs_compañero_equipo`, `KPI_evitar_accidentes`, `KPI_resultados_generales`, `KPI_lluvia` y `KPI_regularidad`. Para todas las métricas se ha creado un formato condicional que indica de color rojo un valor bajo de la métrica (malo) y en color verde un valor alto (bueno). Al fin y al cabo se ha creado un **mapa de calor (heatmap)** en el que las columnas son las métricas y las filas son los años.

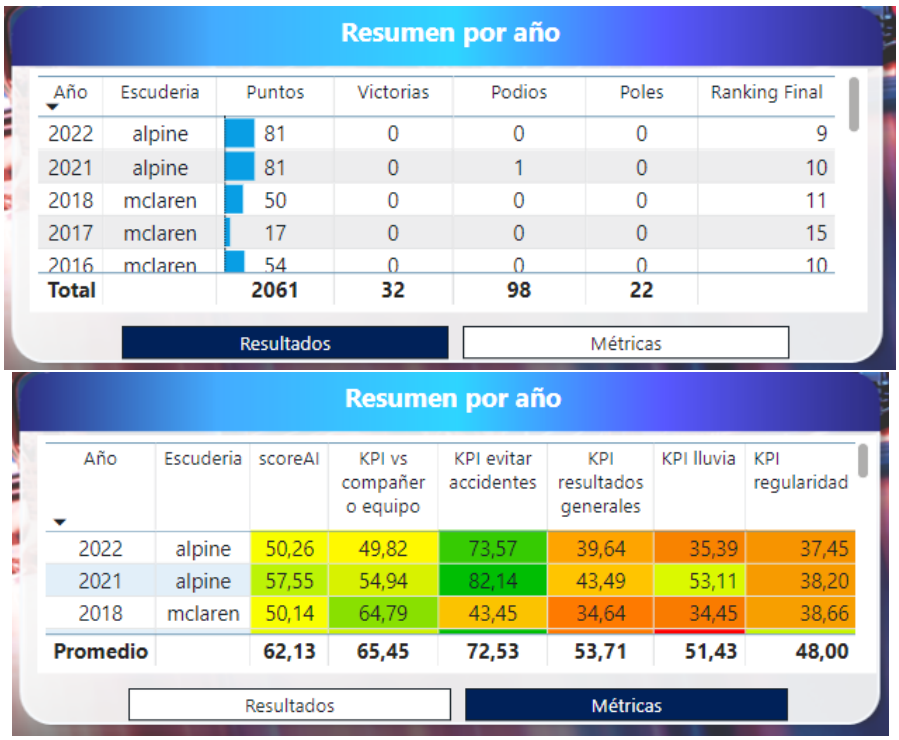


Figura 8.7: Resumen por año de un piloto

Recuadro de Max velocidad media por vuelta

Por último en la esquina inferior izquierda se sitúa el apartado de máxima velocidad media por vuelta con tres gráficos. El primero es un **gráfico de gauge (gauge plot)** que muestra la máxima velocidad media por vuelta alcanzada por el piloto en los circuitos y años seleccionados. Para crear el gráfico es necesario introducir un valor mínimo y un valor máximo para ello se han creado dos medidas DAX:

- `MaxFastSpeedPilotos`: Máximo de la máxima velocidad media por vuelta de todos piloto en los circuitos y años seleccionados. Para ello se indica con una medida DAX que se quiere calcular el máximo de `fastLapSpeed` sin que afecte el filtro de piloto.

```
MaxFastSpeedPilotos =
CALCULATE(
    MAX('CarreraResultado'[fastestLapSpeed]),
    ALL('Piloto'[complete_name]))
```

- `MinFastSpeedPilotos`: Mínimo de la máxima velocidad media por vuelta de todos los pilotos en los circuitos y años seleccionados. Para ello se indica con una medida DAX que se quiere calcular el mínimo de `fastLapSpeed` sin que afecte el filtro de piloto.

```
MinFastSpeedPilotos =
CALCULATE(
    MIN('CarreraResultado'[fastestLapSpeed]),
    ALL('Piloto'[complete_name]))
```

Los valores de estas dos medidas se indican en los extremos del gráfico de gauge con dos etiquetas una roja para `MinFastSpeedPilotos` y otra verde para `MaxFastSpeedPilotos`. El gráfico de gauge también hay que indicar tres rangos de colores de forma manual lo cuál es una limitación del gráfico de Power BI ya que sería mejor utilizar una paleta de colores con degradado. Se ha optado la zona roja desde el valor mínimo hasta 130 kilómetros por hora (velocidad lenta), después la zona amarilla hasta 220 kilómetros por hora (velocidad media) y de 220 hasta el máximo zona verde (velocidad rápida). Debajo del gráfico de gauge se han situado dos **tablas**. La primera indica cuando y dónde ha alcanzado el piloto la máxima velocidad media por vuelta de los circuitos y años seleccionados. La segunda tabla indica cuando y dónde ha alcanzado el piloto la vuelta más rápida de los circuitos y años seleccionados.

8.1.3. Dashboard Escudería

Este dashboard se encarga de visualizar datos de una escudería concreta, en uno o más circuitos escogidos y en un periodo de años seleccionados.

Teniendo en cuenta el mockup con el diseño para el cuadro de mando Escudería (ver Figura 5.8) y las guías de diseño de cuadros de mando explicadas en el apartado 5.5 junto con la disponibilidad y calidad de los datos analizados en el Capítulo 6 de la ETL se ha implementado el Dashboard de Escudería, (ver Figura 8.8).

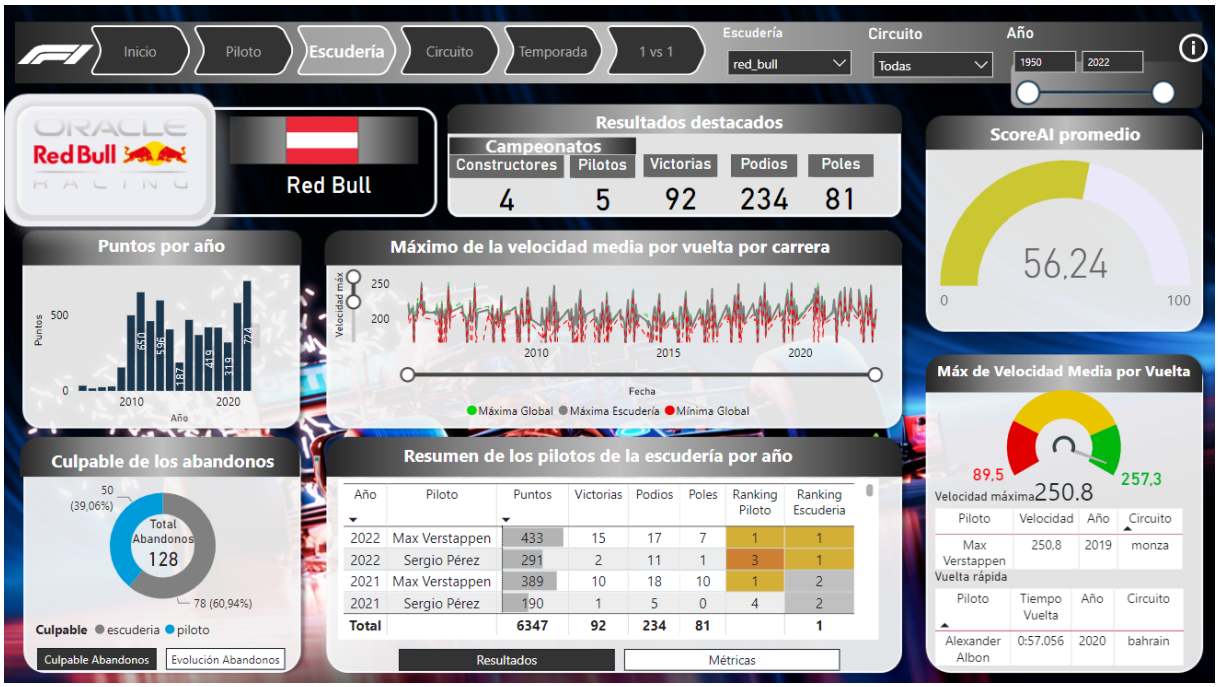


Figura 8.8: Dashboard Escudería

Siguiendo con la estructura basada en agrupaciones se han creado los mismos ocho recuadros situados en los mismos lugares que para el Dashboard Piloto, esto se ha realizado por consistencia interna de la aplicación y para reducir el tiempo de aprendizaje de la aplicación.

El color de fondo se ha cambiado a color gris (igual que cuando el culpable es la escudería en el gráfico de culpable de los abandonos) para indicar al usuario que se ha cambiado de cuadro de mando.

Encabezado del cuadro de mando Escudería

En la parte superior junto al logo de la Fórmula 1 se encuentra el menú para navegar entre los diferentes cuadros de mando, siempre en un color más claro está resaltado el cuadro de mando que está seleccionado. A la derecha del menú se encuentran los filtros específicos para el dashboard de Escudería los cuáles son el filtro de Escudería que permite seleccionar de manera única una escudería, el filtro de circuito que permite seleccionar de una manera múltiple los circuitos deseados y por último un slider para seleccionar el rango de años deseado. En la esquina superior se sitúa el botón de más detalles (ℹ) que permite acceder al manual de ayuda del usuario, el cuál es detallado en el apartado 8.2.

Recuadro de información general de la escudería

Siguiendo con el patrón Z mencionado en el apartado 5.5, en la zona dónde primero se visualiza del cuadro de mando se sitúa un recuadro con la información general de la escudería, con la imagen (obtenida mediante web scraping en la Wikipedia), bandera de la nacionalidad de la escudería y nombre de la escudería.

Recuadro de resultados destacados

A la derecha de la información general de la escudería se muestran con etiquetas los resultados destacados de esa escudería en los circuitos seleccionados en el rango de años elegido. Los resultados destacados mostrados son los siguientes:

- Campeonatos de constructores: Número de campeonatos de constructores de la Fórmula 1 obtenido por la escudería en el rango de años elegido.
- Campeonatos de pilotos: Número de campeonatos de pilotos de la Fórmula 1 obtenido por pilotos de la escudería en el rango de años elegido.
- Victorias: Número de victorias de la escudería en los circuitos seleccionados y en el rango de años elegido.
- Podios: Número de podios de la escudería en los circuitos seleccionados y en el rango de años elegido. Se vuelve a incluir un tooltip con la distribución de esos podios.
- Poles: Número de poles de la escudería en los circuitos seleccionados y en el rango de años elegido.

Recuadro de *scoreAI* promedio

A la derecha de los resultados destacados al igual que en el cuadro de mando Piloto se encuentra la métrica *scoreAI* pero esta vez se realiza el promedio de las métricas de los pilotos de la escudería en los años seleccionados. El gráfico y la paleta de colores es igual que en el dashboard Piloto.

Recuadro de puntos por año

Debajo de estos tres recuadros superiores aparece el recuadro de puntos por año y de velocidad máxima por carrera. En el recuadro de puntos por año se representa mediante un **gráfico de barras (bar chart)** el total de puntos obtenido por el piloto en los circuitos seleccionados y en el rango de años elegido. En el eje X se sitúa el año mientras que en el eje Y la suma de puntos. Para indicar más información mediante un tooltip cuando se pasa el ratón por encima de una barra de un año se indica a mayores el ranking final obtenido por la escudería, el ranking final del mejor y peor piloto de la escudería. También se muestra en el tooltip el número de puntos obtenidos por la escudería y el año concreto de la barra.

Recuadro de máxima velocidad media por vuelta por carrera

A la derecha del gráfico de puntos por año se sitúa el gráfico de máxima velocidad media por vuelta por carrera, para ello se utiliza un **gráfico de líneas (line plot)** ya que se representa en el Eje Y la variable Máximo de la velocidad media por vuelta alcanzada por la escudería en cada carrera de los circuitos seleccionados en el rango de años seleccionado, la cuál es una variable numérica continua y en el X se representa la fecha. Como color se ha escogido el mismo que el color de fondo del dashboard para crear armonía y en el tooltip se indica el valor de la velocidad máxima, la fecha, el nombre del circuito y la meteorología de esa carrera.

Recuadro de culpable de los abandonos

Por último en la fila inferior se representa información más específica y más detallada. En el recuadro inferior izquierdo está relacionado con los abandonos de pilotos de la escudería en los circuitos y años seleccionados. Se han creados dos gráficos relacionados con los abandonos, el primero para representar quien fue el culpable del abandono (error humano o error mecánico) y el segundo para ver la evolución en los años de esos abandonos. Para no sobrecargar el dashboard e introducir en tamaño pequeño los gráficos se ha vuelto a introducir en el recuadro dos botones para intercambiar entre cada uno de los gráficos. Para el gráfico de culpable de abandonos se ha elegido un **gráfico de donut (donut chart)**, como colores se han elegido un color gris (igual que el fondo de dashboard Escudería) para errores de escudería (errores mecánicos) y el color azul (igual que el fondo de dashboard Piloto) para errores de piloto (errores humanos). Dentro del gráfico de donut se ha incluido la suma de abandonos entre los dos posibles culpables. Para el gráfico de evolución por años de los abandonos se ha realizado un **gráfico de barras apiladas (stacked bar chart)** apilando los abandonos según el culpable del accidente (piloto o escudería) y siguiendo el mismo código de colores que para el gráfico de donut por consistencia interna.

Recuadro de resumen de los pilotos de la escudería por año

En la fila inferior de la columna central del sistema de cuadrícula se sitúa el resumen por año de los pilotos de la escudería en los años seleccionados. Esta información está resumida en formato tabla y al ser muy detallada se ha optado por situarla en la parte inferior del cuadro de mando. Para no sobrecargar el cuadro de mando se ha introducido dos botones para intercambiar entre los resultados obtenidos de pilotos de la escudería y las métricas calculadas a los pilotos de la escudería. Para ambos resúmenes se ha optado por una **tabla** ya que se muestran muchas variables.

Para el apartado de resultados se representa el año, nombre del piloto, el número de puntos, número de victorias, podios, poles y ranking final obtenido. Para los puntos se ha incluido un gráfico de barras dentro de la columna. Para el ranking final se ha incluido un formato condicional que colorea el fondo de color oro,plata o bronce si el piloto ha finalizado en el ranking final primero, segundo o tercero respectivamente.

Para las métricas se ha representado junto al año y la escudería donde pilotó, la métrica *scoreAI* y las métricas parciales `KPI_vs_compañero_equipo`, `KPI_evitar_accidentes`, `KPI_resultados_generales`, `KPI_lluvia` y `KPI_regularidad`. Para todas las métricas se ha creado un formato condicional que indica de color rojo un valor bajo de la métrica (malo) y en color verde un valor alto (bueno). Al fin y al cabo se ha creado un **mapa de calor (heatmap)** en el que las columnas son las métricas y las filas son los años.

Recuadro de Max velocidad media por vuelta

Por último en la esquina inferior izquierda se situa el apartado de máxima velocidad media por vuelta con tres gráficos. El primero es un **gráfico de gauge (gauge plot)** que muestra la velocidad máxima alcanzada por la escudería en los circuitos y años seleccionados. Para crear el gráfico es necesario introducir un valor mínimo y un valor máximo para ello se han creado dos medidas DAX:

- `MaxFastSpeedEscuderia`: Máximo de la máxima velocidad media por vuelta de una escudería en los circuitos y años seleccionados incluyendo los registros de todas las escuderías. Para ello se indica

con una medida DAX que quieres calcular el máximo de `fastLapSpeed` sin que afecte el filtro de escudería.

```
MaxFastSpeedEscuderia =
CALCULATE(
    MAX('CarreraResultado'[fastestLapSpeed]),
    ALL('Escuderia'[constructorRef]))
```

- `MinFastSpeedEscuderia`: Mínimo de la máxima velocidad media por vuelta en los circuitos y años seleccionados incluyendo los registros de todas las escuderías. Para ello se indica con una medida DAX que quieres calcular el mínimo de `fastLapSpeed` sin que afecte el filtro de escudería.

```
MinFastSpeedEscuderia =
CALCULATE(
    MIN('CarreraResultado'[fastestLapSpeed]),
    ALL('Escuderia'[constructorRef]))
```

Los valores de estas dos medidas se indican en los extremos del gráfico de gauge con dos etiquetas una roja para `MinFastSpeedEscuderia` y otra verde para `MaxFastSpeedEscuderia`. El rango de colores para el gráfico de gauge es el mismo que el elegido para el dashboard Piloto. Debajo del gráfico de gauge se han situado dos **tablas**, la primera indicando cuando y qué piloto ha alcanzado la velocidad máxima de la escudería en los circuitos y años seleccionados. La segunda tabla indica cuando y qué piloto de la escudería ha realizado la vuelta más rápida de los circuitos y años seleccionados.

8.1.4. Dashboard Circuito

Este dashboard se encarga de visualizar datos de un circuito concreto en un periodo de años seleccionados.

Teniendo en cuenta el mockup con el diseño para el cuadro de mando Circuito, (ver Figura 5.6) y las guías de diseño de cuadros de mando explicadas en el apartado 5.5 junto con la disponibilidad y calidad de los datos analizados en el Capítulo 6 de la ETL se ha implementado el Dashboard de Circuito, ver Figura 8.9.

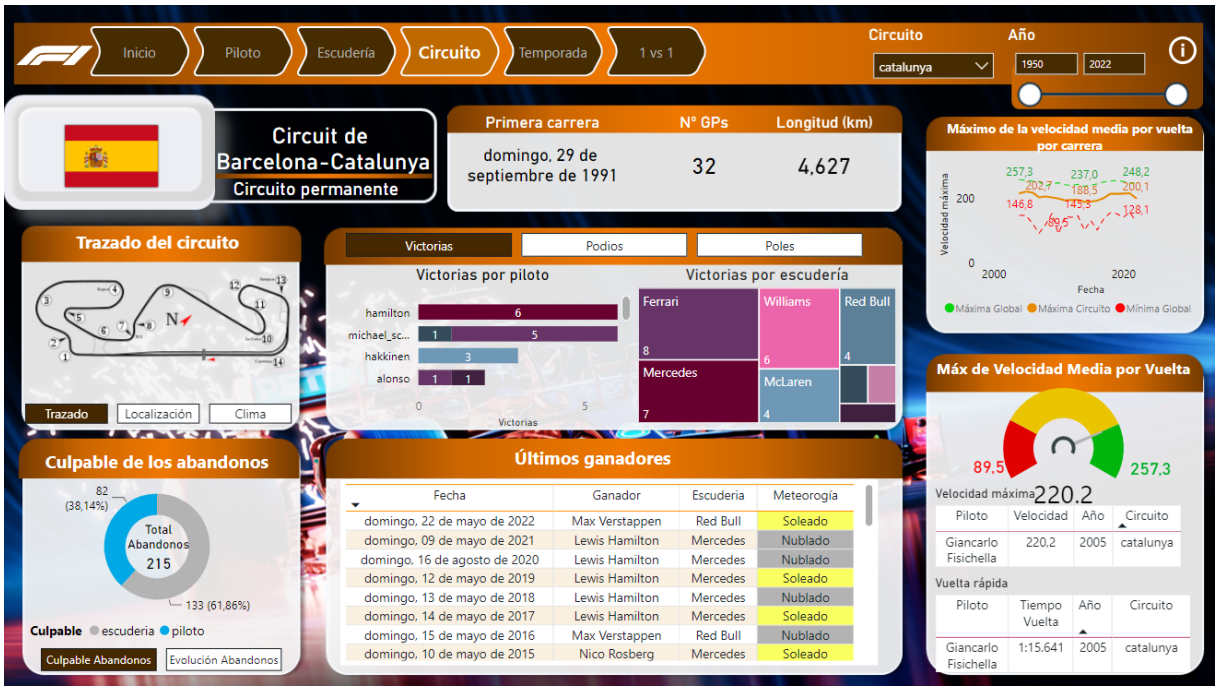


Figura 8.9: Dashboard Circuito

Siguiendo la estructura basada en agrupaciones se han creado ocho recuadros situados en los mismos lugares que para el Dashboard Piloto y Dashboard Escudería, esto se ha realizado por consistencia interna de la aplicación y para reducir el tiempo de aprendizaje de la aplicación.

El color de fondo se ha cambiado a color marrón para indicar al usuario que se ha cambiado de cuadro de mando.

Encabezado del cuadro de mando Circuito

En la parte superior junto al logo de la Fórmula 1 se encuentra el menú para navegar entre los diferentes cuadros de mando, siempre en un color más claro está resaltado el cuadro de mando que está seleccionado. A la derecha del menú se encuentran los filtros específicos para el dashboard de Circuito los cuáles son el filtro de circuito que permite seleccionar de manera única un circuito y un slider para seleccionar el rango de años deseado. En la esquina superior se sitúa el botón de más detalles (ℹ) que permite acceder al manual de ayuda del usuario, el cuál es detallado en el apartado 8.2.

Recuadro de información general del circuito

Siguiendo con el patrón Z mencionado en el apartado 5.5, en la zona dónde primero se visualiza del cuadro de mando se sitúa un recuadro con la información general del circuito, con la bandera del país donde se sitúa el circuito, el nombre completo del circuito y el tipo de circuito que es (circuito permanente, urbano u otro).

Recuadro de resultados destacados

A la derecha de la información general del circuito se muestran con etiquetas los resultados destacados de ese circuito. Los resultados destacados mostrados son los siguientes:

- Primera carrera: Fecha de la primera carrera celebrada en ese circuito. Este dato no está afectado por el slider de año, pese a que se escoja un rango de años en las que no está la fecha de la primera carrera seguirá apareciendo la fecha de la primera de forma correcta. Para que el filtro de año no afecte se realiza siguiendo estos pasos:

  Formato → Editar interacciones, después se selecciona el filtro de año y en el apartado de filtros en la esquina superior derecha de la etiqueta se selecciona ninguno.

- Nº GPs: Número total de Grandes Premios disputados en ese circuito, no afecta el slider de año al igual que con la primera carrera.

- Longitud (km): Longitud del trazado del circuito más actual dentro de los años elegidos por el slider, se mide en kilómetros.

Recuadro de máxima velocidad media por vuelta por carrera

A la derecha de los resultados destacados en vez de la métrica *scoreAI* que carece de sentido en este cuadro de mando, se sitúa el **gráfico de líneas (lineplot)** para representar la máxima velocidad media por vuelta por carrera. Mostrando en el eje X la fecha y en el eje Y el máximo de la máxima velocidad media por vuelta en ese año. Se muestra mediante un tooltip la fecha exacta y la meteorología de la carrera. Se ha cambiado de lugar del gráfico de máxima velocidad media por vuelta por carrera ya que en el lugar donde se sitúa en otros cuadros de mando se va representar dos gráficos más importantes y que necesitan un recuadro más grande. Las dos opciones que había era dejar el gráfico de velocidad máxima como estaba en otros cuadros de mando pero cambiar toda la estructura de recuadros o únicamente cambiar el recuadro donde se representa la velocidad máxima por carrera, finalmente se eligió la segunda opción ya que era la que menos desorganizaba el cuadro de mando.

Recuadro de trazado, localización y meteorología

Debajo de estos tres recuadros superiores aparece el recuadro de trazado del circuito, localización y meteorología. Como el recuadro es pequeño para no sobrecargar el cuadro de mando sólo se representa un gráfico a la vez y se puede cambiar los gráficos mediante tres marcadores.

El primer gráfico corresponde a cuando se selecciona el botón "Trazado" y se muestra una imagen del trazado del circuito (imagen obtenida mediante web scraping en la Wikipedia).

La segunda visualización correspondiente a cuando se selecciona el marcador "Localización" es un **mapa de burbujas (bubble chart map)** con una única burbuja del tamaño de número de Grandes Premios en la ubicación exacta del circuito. Este mapa es interactivo y permite al usuario navegar para hacer zoom y moverse por el mapa y observar a qué ciudad o región pertenece el circuito, de que ciudades se encuentra cerca o si se sitúa próxima al mar. Encima del mapa se sitúa mediante una etiqueta la localización del circuito.

Por último el tercer marcador titulado "Clima" mediante un **gráfico de donut (donut chart)** se representa la proporción de veces que ha llovido, ha estado nublado, soleado u otro. Como colores de la leyenda se ha escogido amarillo para soleado, azul para lluvia, gris para nublado y azul oscuro para otros, este código de colores se respeta para el resto de cuadros de mando. Se ha escogido un gráfico de donut porque hay pocas categorías y permite observar tanto el valor absoluto como la proporción de cada categoría. Este recuadro se observa en la Figura 8.10.

Figura 8.10: Recuadro de trazado, localización y meteorología

Recuadro de victorias, podios y poles

A la derecha del gráfico de puntos por año se sitúa los gráficos de victorias, podios y poles tanto de pilotos como de escudería, como son seis gráficos se ha optado por crear tres marcadores uno para victorias, otro para podios y otro para poles. Para los tres marcadores se representan los dos mismos gráficos en los que cambian la variable que se representa. Para mostrar las victorias/podios/poles por piloto se ha escogido un **gráfico de barras apiladas (stacked bar chart)**. El eje Y es el apellido del piloto y el eje X es el recuento de la variable seleccionada en las diferentes escuderías. Este diagrama de barras apilado aparece ordenado mostrando en la parte superior al piloto con mayor recuento de la variable. Como leyenda se ha escogido la escudería.

Como no existe mucho espacio no se representa la leyenda indicando que escudería es cada color, pero esto se rectifica con el gráfico que se ha elegido para representar las victorias/podios/poles por escudería la cual se ha escogido un **tree map**. Este gráfico representa con un área determinado el total de victorias/podios/poles de cada escudería, indicando con un mayor área las escuderías con mayor valor en la variable indicada. El color de cada escudería corresponde con el mismo del diagrama de barras de victorias/podios/poles por piloto, por ese motivo no se incluyó explicación de la leyenda.

Recuadro de culpable de los abandonos

Por último en la fila inferior se representa información más específica y más detallada. En el recuadro inferior izquierdo está relacionado con los abandonos de pilotos en el circuito y años seleccionados. Se han creados dos gráficos relacionados con los abandonos. El primero para representar quien fue el culpable del abandono (error humano o error mecánico). El segundo para ver la evolución en los años de esos abandonos. Para no sobrecargar el dashboard e introducir en tamaño pequeño los gráficos se ha vuelto a introducir en el recuadro dos botones para intercambiar entre cada uno de los gráficos. Para el gráfico de culpable de abandonos se ha elegido un **gráfico de donut (donut chart)**, como colores se han elegido un color gris (igual que el fondo de dashboard Escudería) para errores de escudería (errores mecánicos) y el color azul (igual que el fondo de dashboard Piloto) para errores de piloto (errores humanos). Dentro

del gráfico de donut se ha incluido la suma de abandonos entre los dos posibles culpables. Para el gráfico de evolución por años de los abandonos se ha realizado un **gráfico de barras apiladas (stacked bar chart)** apilando los abandonos según el culpable del accidente (piloto o escudería) y siguiendo el mismo código de colores que para el gráfico de donut por consistencia interna.

Recuadro de últimos ganadores

En la fila inferior de la columna central del sistema de cuadrícula se sitúa la información de los últimos ganadores en los años seleccionados. Esta información está resumida en formato tabla y al ser muy detallada se ha optado por situarla en la parte inferior del cuadro de mando. Para este resumen se ha optado por una **tabla** ya que se muestran muchas variables como son la fecha, el nombre del ganador, la escudería del ganador y la meteorología de la carrera. Se ha incluido un formato condicional para la variable de meteorología que colorea de color azul si hubo lluvia, gris si estuvo nublado y amarillo si estuvo soleado.

Recuadro de Max velocidad media por vuelta

Por último en la esquina inferior izquierda se sitúa el apartado de máxima velocidad media por vuelta con tres gráficos. El primero es un **gráfico de gauge (gauge plot)** que muestra la máxima velocidad media por vuelta alcanzada en el circuito en los años seleccionados. Para crear el gráfico es necesario introducir un valor mínimo y un valor máximo para ello se han creado dos medidas DAX:

- `MaxFastSpeedCircuito`: Máximo de la máxima velocidad media por vuelta en todos circuitos y en los años seleccionados incluyendo los registros de todos los circuitos. Para ello se indica con una medida DAX que quieres calcular el máximo de `fastLapSpeed` sin que afecte el filtro de circuito.

```
MaxFastSpeedCircuito =
CALCULATE(
    MAX('CarreraResultado'[fastestLapSpeed]),
    ALL('Circuito'[circuitRef]))
```

- `MinFastSpeedCircuito`: Mínimo de la máxima velocidad media por vuelta en todos los circuitos y en los años seleccionados incluyendo los registros de todas las escuderías. Para ello se indica con una medida DAX que quieres calcular el mínimo de `fastLapSpeed` sin que afecte el filtro de circuito.

```
MinFastSpeedCircuito =
CALCULATE(
    MIN('CarreraResultado'[fastestLapSpeed]),
    ALL('Circuito'[circuitRef]))
```

Los valores de estas dos medidas se indican en los extremos del gráfico de gauge con dos etiquetas una roja para `MinFastSpeedCircuito` y otra verde para `MaxFastSpeedCircuito`. El gráfico de gauge también hay que indicar los rangos de colores de forma manual y se ha optado la zona roja desde el valor mínimo hasta 130 kilómetros por hora (velocidad lenta), después la zona amarilla hasta 220 kilómetros por hora (velocidad media) y de 220 hasta el máximo zona verde (velocidad rápida). Debajo del gráfico de gauge se han situado dos **tablas**, la primera indicando cuando y qué piloto ha alcanzado la velocidad máxima del circuito y en los años seleccionados. La segunda tabla indica cuando y qué piloto ha realizado la vuelta más rápida del circuito en los años seleccionados.

8.1.5. Dashboard Temporada

Este dashboard se encarga de visualizar datos de una o varias temporadas de la Fórmula 1.

Teniendo en cuenta el mockup con el diseño para el cuadro de mando Temporada (ver Figura 5.9) y las guías de diseño de cuadros de mando explicadas en el apartado 5.5 junto con la disponibilidad y calidad de los datos analizados en el capítulo 6 de la ETL se ha implementado el Dashboard de Temporada, ver Figura 8.11.

Siguiendo la estructura realizada para el resto de cuadros de mandos basado en agrupaciones se han creado cinco recuadros.

El color de fondo se ha cambiado a color dorado para indicar al usuario que se ha cambiado de cuadro de mando.

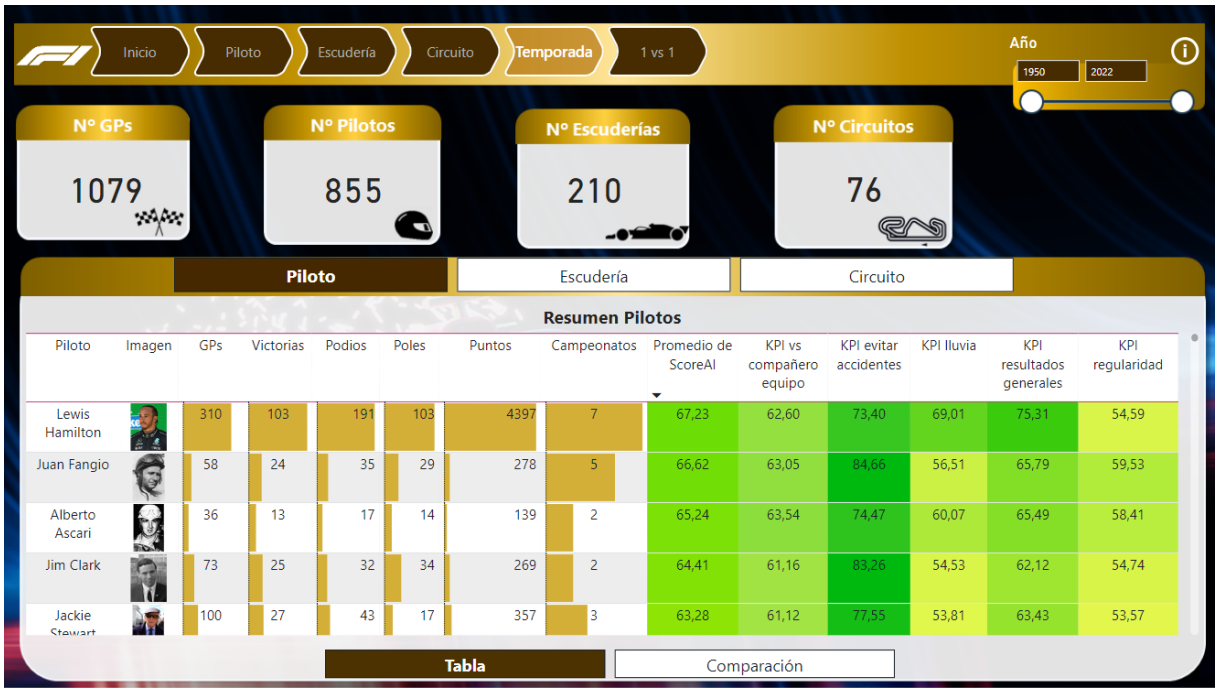


Figura 8.11: Dashboard Temporada

Encabezado del cuadro de mando Temporada

En la parte superior junto al logo de la Fórmula 1 se encuentra el menú para navegar entre los diferentes cuadros de mando, siempre en un color más claro está resaltado el cuadro de mando que está seleccionado. A la derecha del menú se encuentran los filtros específicos para el dashboard de Temporada los cuáles es únicamente un slider para seleccionar el rango de años deseado. En la esquina superior se sitúa el botón de más detalles (ℹ) que permite acceder al manual de ayuda del usuario, el cuál es detallado en el apartado 8.2.

Recuadros de NºGPs, Nº Pilotos, Nº Escuderías y Nº Circuitos

Siguiendo con el patrón Z mencionado en el apartado 5.5, en la zona dónde primero se visualiza del cuadro de mando se sitúan cuatro recuadros indicando el número total de Grandes Premios, pilotos, escuderías y circuitos en el rango de años seleccionados. Se representa mediante una etiqueta y debajo del valor se indica un icono representativo de la variable en cuestión (un casco para un piloto, un monoplaza para una escudería ...).

Recuadros de Piloto, Escudería y Circuito

Debajo de los recuadros con las etiquetas anteriores se representa un recuadro con un resumen general de pilotos , escuderías y circuitos. Para elegir entre cada uno de los resúmenes se han creado tres marcadores para elegir entre "Piloto", "Escudería" y "Circuito". Para cada uno de los tres apartados se muestran dos tipos de información la primera en formato tabla y la segunda con un gráfico para realizar una comparación de pilotos. Para elegir uno u otro se ha creado otros dos marcadores: "Tabla" y "Comparación".

Para la combinación de marcadores "Piloto" y "Tabla" se representa mediante una **tabla** las siguientes variables: nombre completo de piloto, número de Grandes Premios, victorias, podios, poles, títulos, puntos y luego promedio de la métrica *scoreAI* y promedio de las métricas parciales `KPI_vs_compañero_equipo`, `KPI_evitar_accidentes`, `KPI_resultados_generales`, `KPI_lluvia`

y `KPI_regularidad`. Para las variables que no son métricas se han incluido un gráfico de barras dentro de la columna. Para las métricas se han creado un formato condicional que indica de color rojo un valor bajo de la métrica (malo) y en color verde un valor alto (bueno).

Si estando elegido el marcador de Piloto se cambia el marcador de "Tabla" por el de "Comparación" aparece un **gráfico de burbujas (bubble chart)** en el que se representa el ranking final frente al *scoreAI* de un piloto en un año concreto.

En la parte inferior hay un slider para seleccionar un año concreto y a la izquierda del slider un botón para iniciar la animación desde el año seleccionado hasta el final. El tamaño de las burbujas representa el número de puntos de un piloto en un año. Si se pulsa en una burbuja realiza la animación de la trayectoria de ese piloto concreto. También se han incluido dos sliders en ambos ejes por si el usuario solo quiere visualizar ciertos rangos concretos. Se indica mediante un fondo personalizado qué pilotos han sido mejores y peores en un año y quienes han tenido un buen coche.

Ambos gráficos ("Tabla" y "Comparación") para Piloto se observan en la Figura 8.12

Figura 8.12: Tabla y gráfico de comparación de Piloto

Para la combinación de marcadores "Escudería" y "Tabla" se representa mediante una **tabla (table)** las siguientes variables: nombre escudería,logo, número de victorias, podios, poles, títulos de pilotos y puntos. Para todas las variables se ha incluido un gráfico de barras dentro de la columna.

Si estando elegido el marcador de Escudería se cambia el marcador de "Tabla" por el de "Comparación" aparece un **gráfico de burbujas (bubble chart)** en el que se representa el ranking final promedio de los pilotos de cada escudería frente al *scoreAI* promedio de los pilotos de cada escudería.

En la parte inferior hay un slider para seleccionar un año concreto y a la izquierda del slider un botón para iniciar la animación desde el año seleccionado hasta el final. El tamaño de las burbujas representa el número de puntos de la escudería en un año. Si se pulsa en una burbuja realiza la animación de la trayectoria de esa escudería concreta. También se han incluido dos sliders en ambos ejes por si el usuario solo quiere visualizar ciertos rangos concretos. Se indica mediante un fondo personalizado qué escuderías han tenido mejores pilotos y cuáles mejores coches.

Ambos gráficos ("Tabla" y "Comparación") para Escudería se observan en la Figura 8.13

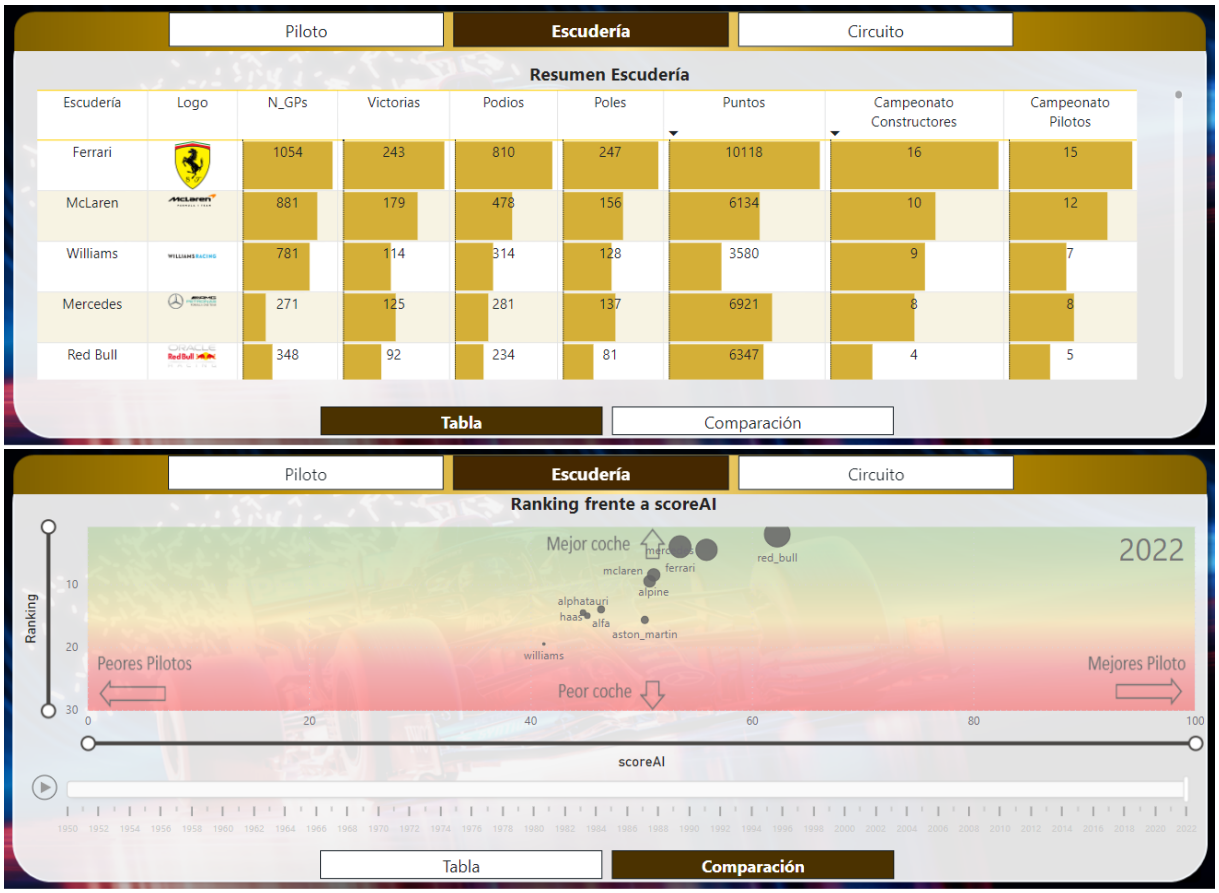


Figura 8.13: Tabla y gráfico de comparación de Escudería

Por último para la combinación de marcadores "Circuito" y "Tabla" se representa mediante una **tabla (table)** las siguientes variables: año,ronda, nombre del circuito, piloto ganador, su escudería, la meteorología y el tipo de circuito. Se ha incluido un formato condicional para la columna de meteorología que colorea de color azul si hubo lluvia, gris si fue nublado y amarillo si soleado.

Si estando elegido el marcador de Escudería se cambia el marcador de "Tabla" por el de "Comparación" aparece un **gráfico de burbujas (bubble chart)** en el que se representa la velocidad máxima de cada circuito frente al número de accidentes.

En la parte inferior hay un slider para seleccionar un año concreto y a la izquierda del slider un botón para iniciar la animación desde el año seleccionado hasta el final. El color de la burbuja indica la meteorología en ese circuito en un año concreto. Si se pulsa en una burbuja realiza la animación de la trayectoria de esa circuito concreto. También se han incluido dos sliders en ambos ejes por si el usuario solo quiere visualizar ciertos rangos concretos. Se indica mediante un fondo personalizado qué circuitos son más rápidos y cuáles son más peligrosos.

Ambos gráficos ("Tabla" y "Comparación") para Escudería se observan en la Figura 8.14

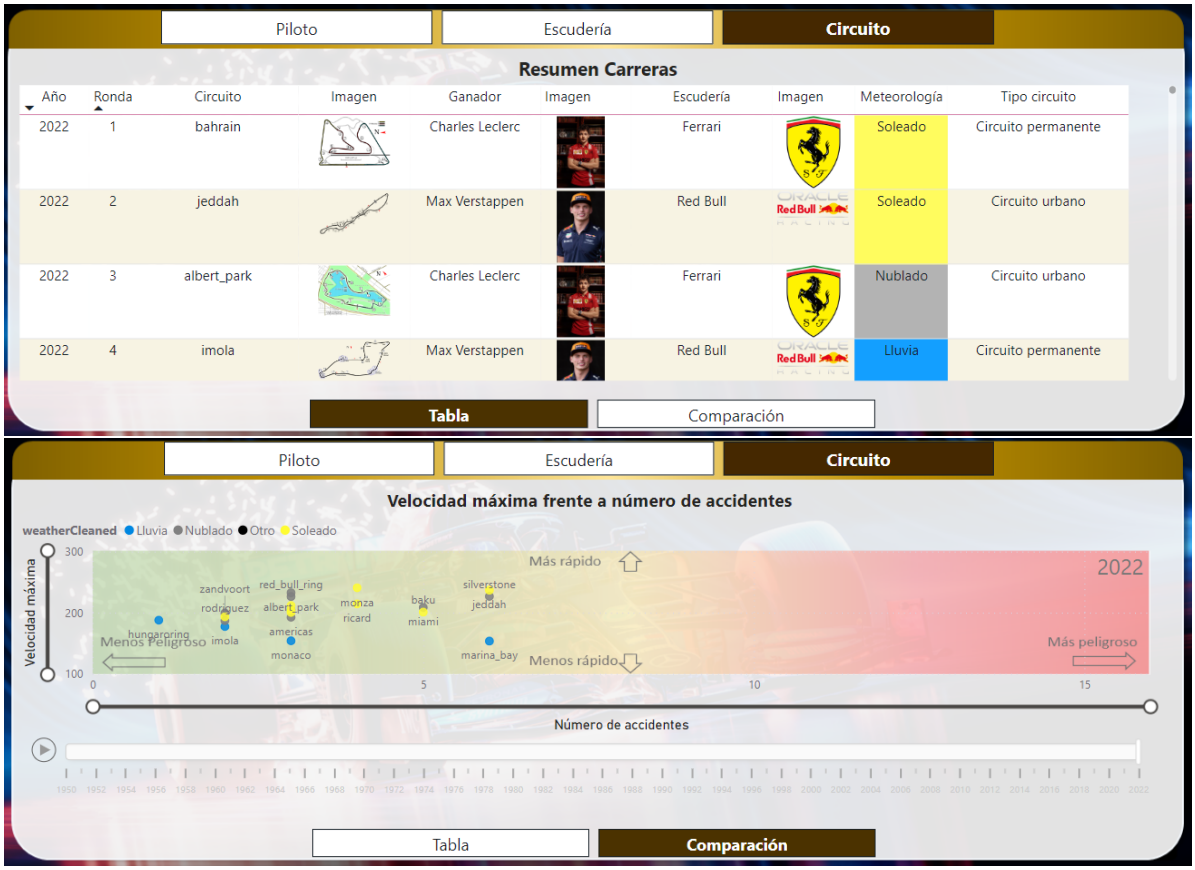


Figura 8.14: Tabla y gráfico de comparación de Circuito

8.1.6. Dashboard 1 vs 1

Este dashboard se encarga de visualizar datos para comparar dos pilotos de Fórmula 1, (ver Figura 8.15).

Figura 8.15: Dashboard 1 vs 1

Teniendo en cuenta el mockup con el diseño del Dashboard 1 vs 1, (ver Figura 5.10) y las guías de diseño de cuadros de mando explicadas en el apartado 5.5 junto con la disponibilidad y calidad de los datos analizados en el capítulo de la ETL 6 se ha implementado el Dashboard de 1 vs 1.

Siguiendo la estructura realizada para el resto de cuadros de mandos basado en agrupaciones se han creado diez recuadros, cinco recuadros para cada piloto.

El color de fondo se ha cambiado a color azul para el piloto 1 (situado a la izquierda) y color fucsia para el piloto 2 (situado a la derecha). Se ha escogido dos colores diferentes para que se diferencien claramente que recuadros corresponden a cada piloto.

Encabezado del cuadro de mando 1 vs 1

En la parte superior junto al logo de la Fórmula 1 se encuentra el menú para navegar entre los diferentes cuadros de mando, siempre en un color más claro está resaltado el cuadro de mando que está seleccionado. A la derecha del menú se encuentran los filtros específicos para el dashboard de 1 vs 1 los cuáles son dos selectores de pilotos (uno para piloto 1 y otro para piloto 2) y un slider para seleccionar el rango de años deseado. En la esquina superior se sitúa el botón de más detalles (ℹ) que permite acceder al manual de ayuda del usuario, el cuál es detallado en el apartado 8.2.

Recuadro de información general

Siguiendo la estructura del sistema de cuadrículas se ha creado una columna para cada piloto. En la parte superior de cada columna se ha creado un recuadro con la información general del piloto en el se incluye el nombre del piloto y una bandera indicando el país de origen de cada piloto.

Recuadros de imagen, de *scoreAI* y de resultados generales

Debajo del recuadro de información general se sitúan tres recuadros. El primero con la imagen del piloto (obtenida mediante web scraping de la Wikipedia). A su lado se sitúa el recuadro de Resultados Generales el cual incluye cinco etiquetas para representar el número de títulos, de Grandes Premios, victorias, podios y poles de cada piloto en el rango de años seleccionados.

En la parte inferior del recuadro de Resultados Generales se sitúa un recuadro con dos gráficos relacionados con *scoreAI* los cuales se ven seleccionando el marcador de "*scoreAI* promedio" y "*scoreAI* por año" respectivamente, cuando se selecciona un marcador afecta a los recuadros de ambos pilotos.

Si se selecciona "*scoreAI* promedio" se representa mediante un **gráfico de gauge (gauge plot)** el promedio de la métrica *scoreAI* en el rango de años elegido, (ver Figura 8.16). Como color del gráfico se ha optado realizar la metáfora del semáforo según el valor de la variable. Ei el valor de la métrica es bajo (indicador de mal rendimiento del piloto) el color del gráfico es cercano al rojo, valores altos de la variable *scoreAI* (indicador de buen rendimiento del piloto) corresponden con colores similares al verde, para rendimientos medios corresponde con colores similares a amarillo.

Figura 8.16: Recuadro de *scoreAI*

Por otro lado, si se selecciona el marcador "*scoreAI* por año" se muestra una **tabla** con todos los años que ha participado en la Fórmula 1 (dentro del rango de años seleccionado), en qué escudería pilotó, el *scoreAI* de ese año y el puesto del ranking de pilotos en el que finalizó, (ver Figura 8.16). Para el ranking final se ha incluido un formato condicional que colorea el fondo de color oro, plata o bronce si el piloto ha finalizado en el ranking final primero, segundo o tercero respectivamente. Para la métrica *scoreAI* se ha creado un formato condicional que indica de color rojo un valor bajo de la métrica (malo) y en color verde un valor alto (bueno).

Recuadro de métricas

En la parte inferior del cuadro de mando se sitúa un recuadro para cada piloto para representar las métricas parciales `KPI_vs_compañero_equipo`, `KPI_evitar_accidentes`, `KPI_resultados_generales`, `KPI_lluvia` y `KPI_regularidad`.

Para visualizar las métricas se ha optado por un **gráfico de radar (radar chart/spider chart)** el cuál es un gráfico que permite representar varias variables numéricas que tengan el mismo rango, como todas las métricas parciales están acotadas en el rango 0-100 permite que se pueda realizar este gráfico.

Este gráfico no viene por defecto en Power BI y es necesario descargarlo para poder utilizarlo.

Visualizaciones → Obtener más objetos visuales → Buscar radar chart en el buscador y agregar el de la empresa xViz LLC.[92]

Este gráfico verificado permite visualizar tanto el gráfico de radar como en formato tabla seleccionando en la esquina superior derecha el icono de la tabla, (ver Figura 8.17).

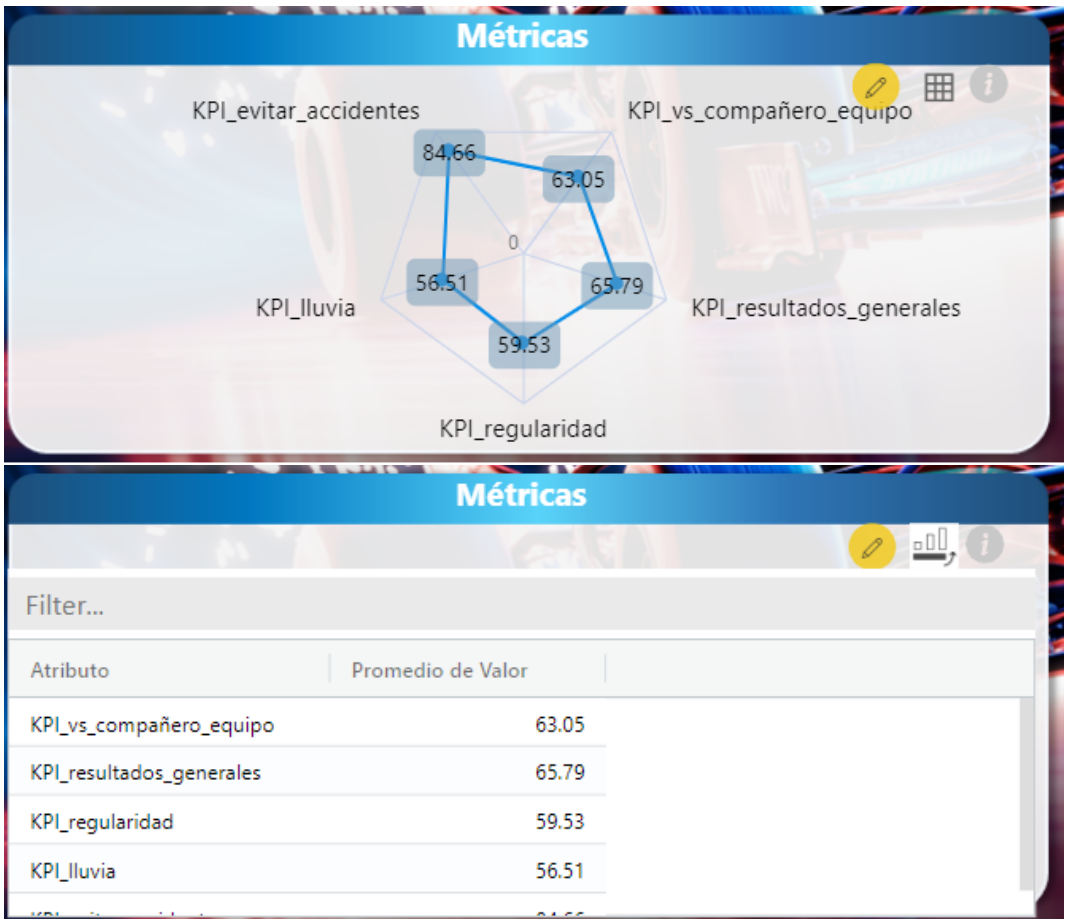


Figura 8.17: Recuadro de métricas

8.2. Manual de usuario de *Fórmula 1 BI*

En el apartado anterior se ha realizado una amplia descripción de todos los gráficos que se han representado en cada dashboard de la aplicación. En el presente apartado se realiza una descripción del manual de ayuda para el usuario que se ha integrado dentro de la propia aplicación con Power BI.

Para poder crear este manual primero hay que tener claro quién van a ser los usuarios de la aplicación. En nuestro caso los usuarios van a ser narradores de Fórmula 1 los cuáles son expertos en Fórmula 1 pero puede que no tengan experiencia utilizando aplicaciones de visualización de datos. Por ello además de crear una aplicación fácil de utilizar se ha creado un manual de ayuda. Este manual no se ha centrado en explicar conceptos de la Fórmula 1 como qué es una pole o un "GP" ya que los narradores ya están familiarizados con estos términos. El manual se ha centrado en explicar a los usuarios qué información pueden encontrar si pasan el ratón por encima del gráfico (tooltips) o indicar una descripción de las variables utilizadas en los gráficos.

Se ha creado un manual por cada dashboard y a todos se acceden desde el mismo lugar ya que en el encabezado se sitúa un botón para ver más detalles (ℹ) que muestra la ayuda. Para quitar la ayuda hay que volver a pulsar el mismo botón.

En el manual de ayuda para la página de inicio, (ver Figura 8.18) se muestra una breve descripción de cada dashboard al que puede navegar para evitar errores en la navegación y reduzca el tiempo en alcanzar su objetivo.

Figura 8.18: Manual de ayuda de la página de Inicio

En los dashboards de Piloto, (ver Figura 8.19), Escudería, (ver Figura 8.20), Circuito, (ver Figura 8.21), Temporada, (ver Figura 8.22) y 1 vs 1, (ver Figura 8.23). Se crea un mensaje de explicación de cada gráfico los cuáles aparecen todos a la vez cuando se pulsa el botón de más detalles (ℹ). En los mensajes de ayuda se indican que tooltips aparecen si pasas el ratón por encima, también se realiza una breve descripción de las métricas utilizadas en cada gráfico para explicar como han sido calculadas. Esta ayuda permite reducir la carga de información de los dashboards ya que permite sobrecargar menos los gráficos y lograr que todos los gráficos se puedan interpretar fácilmente.

Figura 8.19: Manual de ayuda del dashboard Piloto

Figura 8.20: Manual de ayuda del dashboard Escudería

Figura 8.21: Manual de ayuda del dashboard Circuito

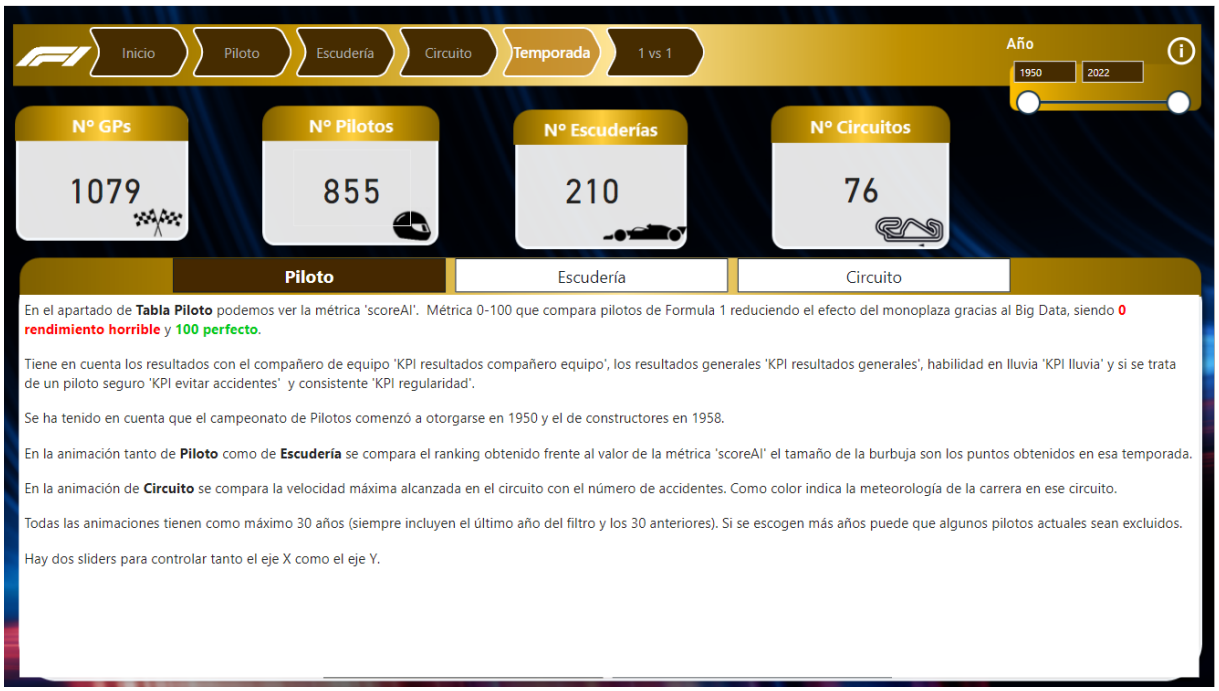


Figura 8.22: Manual de ayuda del dashboard Temporada

Figura 8.23: Manual de ayuda del dashboard 1vs1

Capítulo 9

Pruebas

El testing juega un papel fundamental en un proyecto de Business Intelligence. A medida que las organizaciones dependen cada vez más de la inteligencia empresarial para tomar decisiones, es esencial garantizar la calidad y confiabilidad de los datos y los análisis generados. El testing nos permite evaluar la precisión y consistencia de los resultados obtenidos, identificar posibles errores, verificar la integridad de los datos y validar la funcionalidad del sistema en su conjunto.

Estas pruebas se dividen en dos etapas:

1. Pruebas de la ETL: Se realizan para garantizar la calidad y confiabilidad de los datos y para garantizar que se ha realizado de una manera correcta el proceso de la ETL.
2. Pruebas de usabilidad: Las pruebas de usabilidad sobre un dashboard se refieren al proceso de evaluar la facilidad de uso, eficiencia y satisfacción de los usuarios al interactuar con un panel de control de Business Intelligence.

9.1. Pruebas de la ETL

Este proceso de test ETL se enfoca en garantizar la integridad, precisión y confiabilidad de los datos durante el flujo de extracción, transformación y carga. Estas pruebas del flujo ETL abarcan multitud de pruebas diferentes como las pruebas de validación, integración o transformación de datos. Para cada tabla se han verificado las condiciones más comunes dentro de las pruebas de ETL de un proceso de BI. [93, 94]

Las pruebas se han programado en un script de Python utilizando *asserts*. Se han dividido las pruebas en cuatro grupos:

- Tablas en las que no se han filtrado los datos, por tanto debe haber el mismo número de filas que en el origen. Corresponden a las Tablas Circuito, Escudería y CarreraResultado. Por cada tabla se verifican las siguientes condiciones:

  1. Se comprueba que no hay filas duplicadas en destino.
  2. Se comprueba que el origen y el destino tienen el mismo número de filas.
  3. Se comprueba que los tipos de las columnas del destino son iguales que en el origen. Sólo se tienen en cuenta las columnas del origen que permanecen en el destino.
  4. Se comprueba que los valores de las columnas del destino son iguales que en el origen. Sólo se tienen en cuenta las columnas del origen que permanecen en el destino.
  5. Se comprueba que existen las nuevas columnas que se han añadido al destino.
  6. Se comprueba que los tipos de las nuevas variables del destino son los esperados.
  7. Se comprueba que el número de columnas en destino es igual al número de columnas que permanecen del origen más las columnas nuevas.

  Para verificar estas condiciones en las Tablas Circuito, Escudería y CarreraResultado se ha creado una función genérica llamada *general_ etl_ test(origin,destiny,wanted_ columns,new_ columns)*.

```python
def general_etl_test(origin,destiny,wanted_columns,new_columns):
    """
    Funcion que realiza los test para las tablas donde no se filtran
    los datos y hay las mismas filas en origen y destino. Tablas Escuderia,
    Circuito y CarreraResultado.

    Argumentos:
            origin: Tabla origen
            destiny: Tabla destino
            wanted_columns: Diccionario con las columnas del origen que permanecen
    en destino como clave y su tipo como valor
            new_columns: Diccionario con las columnas nuevas en destino que no
    aparecen en origen como clave y su tipo como valor
    """

    # Se comprueba que no hay filas duplicadas en destino
    assert not destiny.duplicated().any(), 'Filas duplicadas en el destino'

    # Se comprueba que origen y destino tienen el mismo numero de filas
    assert len(origin.index) == len(destiny.index), 'Origen y destino No tienen el
    mismo número de filas'

    # wanted_columns: Mismas columnas en origen y destino
    for column in wanted_columns:
        # Se comprueba que los tipos de las columnas del destino son iguales que en
     el origen.
        assert wanted_columns[column] == destiny[column].dtype, f'Tipo de la
    columna {column} no concuerdan'
        # Se comprueba que los valores de las columnas del destino son iguales que
    en el origen.
        assert origin[column].equals(destiny[column]), f'Valores de la columna {
    column} no concuerdan'

    unwanted_columns = set(origin.columns).difference(set(wanted_columns))

    for column in unwanted_columns:
        assert column not in destiny.columns, f'Columna no deseada {column} está
    presente en el destino'


    # Columnas nuevas en destino
    for column, dtype in new_columns.items():
        # Se comprueba que existen las nuevas columnas
        assert column in destiny.columns, f'Columna {column} no aparece en el
    destino'
        # Se comprueba que los tipos de las variables son los esperados
        assert destiny[column].dtype == dtype, f'Tipo de la columna {column} no
    tiene el tipo esperado {dtype}'

    # Se comprueba que el numero de columnas en destino es igual a
    #  numero de columnas iguales que en origen mas las nuevas
    assert len(destiny.columns) == len(wanted_columns)+len(new_columns), f'
    Inesperado número de columnas en destino. Esperado: {len(wanted_columns)+len(
    new_columns)}, Real: {len(destiny.columns)}'


```

Para cada una de las tres tablas se llama a la función *general_ etl_ test* de la siguiente manera:

```python
 # Tabla Circuito
wanted_columns ={column: str(dtype) for column, dtype in circuits.dtypes.items()}
new_columns = {'image_url':'object'}
general_etl_test(circuits,Circuito,wanted_columns,new_columns)

# Tabla Escuderia
wanted_columns ={column: str(dtype) for column, dtype in constructors.dtypes.items
```

```
    ()}
new_columns = {'image_url':'object','country':'object'}
general_etl_test(constructors,Escuderia,wanted_columns,new_columns)

# Tabla CarreraResultado
wanted_columns ={column: str(dtype) for column, dtype in results.drop(['position','
    positionText','fastestLap','statusId'],axis=1).dtypes.items()}
new_columns = {'status':'object', 'status_fault':'object','
    fastestLapTimeMilliseconds':'float64', 'Victoria':'float64', 'Podio':'float64',
     'Pole':'float64'}
general_etl_test(results,CarreraResultado,wanted_columns,new_columns)


```

- Tablas en las que se han filtrado los datos, por tanto no tiene porque tener el mismo número de filas que en el origen. Corresponden a las Tablas Piloto y CarreraDescripcion. Por cada tabla se verifican las siguientes condiciones:

  1. Se comprueba que no hay filas duplicadas en destino.
  2. Se comprueba que los tipos de las columnas del destino son iguales que en el origen. Sólo se tienen en cuenta las columnas del origen que permanecen en el destino.
  3. Se comprueba que existen las nuevas columnas que se han añadido al destino.
  4. Se comprueba que los tipos de las nuevas variables del destino son los esperados.
  5. Se comprueba que el número de columnas en destino es igual a número de columnas que permanecen del origen más las columnas nuevas.
  6. Se comprueba que se ha realizado correctamente el filtro.

  Para verificar estas condiciones en las Tablas Piloto y CarreraDescripcion se ha creado una función genérica llamada *filtered_etl_test(origin,destiny,wanted_columns,new_columns,results)*.

```
def filtered_etl_test(origin,destiny,wanted_columns,new_columns,results):
    """
    Funcion que realiza los test para las tablas donde se filtran
    los datos y no tiene porque haber las mismas filas en origen y destino. Tablas
    Escuderia, Circuito y CarreraResultado.

    Argumentos:
            origin: Tabla origen
            destiny: Tabla destino
            wanted_columns: Diccionario con las columnas del origen que permanecen
    en destino como clave y su tipo como valor
            new_columns: Diccionario con las columnas nuevas en destino que no
    aparecen en origen como clave y su tipo como valor
            results: Tabla origen de resultados de carreras
    """
    # Se comprueba que no hay filas duplicadas en destino
    assert not destiny.duplicated().any(), 'Filas duplicadas en el destino'

    # wanted_columns: Mismas columnas en origen y destino
    for column in wanted_columns:
        # Se comprueba que los tipos de las columnas del destino son iguales que en
     el origen.
        assert wanted_columns[column] == destiny[column].dtype, f'Tipo de la
    columna {column} no concuerdan'

    unwanted_columns = set(origin.columns).difference(set(wanted_columns))

    for column in unwanted_columns:
        assert column not in destiny.columns, f'Columna no deseada {column} está
    presente en el destino'

```

```
    # Columnas nuevas en destino
    for column, dtype in new_columns.items():
        # Se comprueba que existen las nuevas columnas
        assert column in destiny.columns, f'Columna {column} no aparece en el destino'
        # Se comprueba que los tipos de las variables son los esperados
        assert destiny[column].dtype == dtype, f'Tipo de la columna {column} no tiene el tipo esperado {dtype}'

    # Se comprueba que el numero de columnas en destino es igual a
    #  numero de columnas iguales que en origen mas las nuevas
    assert len(destiny.columns) == len(wanted_columns)+len(new_columns), f'Inesperado número de columnas en destino. Esperado: {len(wanted_columns)+len(new_columns)}, Real: {len(destiny.columns)}'


```

Para cada una de las dos tablas se llama a la función *filtered_ etl_ test* de la siguiente manera:

```
# Tabla Piloto
def driver_test(origin,destiny,wanted_columns,new_columns,results):
    filtered_etl_test(origin,destiny,wanted_columns,new_columns,results)

    # Se comprueba que se ha realizado bien el filtro (pilotos deben tener al menos 10 carreras en Formula 1)
    driver_races = results['driverId'].value_counts()
    for driver in destiny['driverId'].tolist():
        assert driver in driver_races and driver_races[driver] >= 10, f'Piloto con id: {driver} no ha realizado al menos 10 carreras en la F1'

wanted_columns ={column: str(dtype) for column, dtype in drivers.dtypes.items()}
wanted_columns['number'] = 'object'
new_columns = {'image_url':'object','country':'object','complete_name':'object'}
driver_test(drivers,Piloto,wanted_columns,new_columns,results)

# Tabla CarreraDescripcion
def race_test(origin,destiny,wanted_columns,new_columns,results):
    filtered_etl_test(origin,destiny,wanted_columns,new_columns,results)
    # Se comprueba que se ha realizado bien el filtro (la descripcion de carreras tiene que aparecer en resultados)
    assert set(destiny['raceId']).issubset(set(results['raceId'])), f'Existen Carreras que no tienen resultados; por tanto no debe estar en destino'


wanted_columns ={column: str(dtype) for column, dtype in races.drop(['fp1_date','fp1_time','fp2_date','fp2_time','fp3_date','fp3_time',
                                    'quali_date', 'quali_time', 'sprint_date', 'sprint_time'],axis=1).dtypes.items()}
new_columns = {'typeCircuitCleaned':'object','weatherCleaned':'object', 'circuitLengthCleaned':'float64'}
race_test(races,CarreraDescripcion,wanted_columns,new_columns,results)
```

- Tabla Year. Para la tabla Year al ser una tabla creada se comprueba que la variable `year` ha sido creada correctamente:

    1. Se comprueba que hay los mismos años que en la Tabla de CarreraDescripcion.
    2. Se comprueba que no hay filas duplicadas en destino.

    Para la tabla Year se llama a la función *year_ etl_ test* de la siguiente manera:

```
     def year_etl_test(destiny,racesUpdated):
    # Se comprueba que el conjunto de años de Year es subconjunto de años de las carreras
    assert set(destiny['year']).issubset(set(racesUpdated['year'])), 'Existen más años en Year que los que hay en CarreraDescripcion'
```

```
    # Se comprueba que el conjunto de años de las carreras  es subconjunto de años de Year
    assert set(racesUpdated['year']).issubset(set(destiny['year'])),'Existen menos años en Year que los que hay en CarreraDescripcion'
    # Se comprueba que no hay filas duplicadas en destino
    assert not destiny.duplicated().any(), 'Filas duplicadas en el destino'
year_etl_test(Year,CarreraDescripcion)
```

- Tablas de resultados por año. Corresponden a las tablas TemporadaPiloto y TemporadaEscuderia. Por cada tabla se verifican las siguientes condiciones:

  1. Se comprueba que no hay filas duplicadas en destino.
  2. Se comprueba que existen las nuevas columnas.
  3. Se comprueba que los tipos de las variables son los esperados.

  Para verificar estas condiciones en las Tablas TemporadaPiloto y TemporadaEscuderia se ha creado una función genérica llamada *season_etl_test(destiny,new_columns)*

```
    """
    Funcion que realiza los test para las tablas de resultados por año. Tablas TemporadaPiloto y TemporadaEscuderia.
    Argumentos:
            destiny: Tabla destino
            new_columns: Diccionario con las columnas nuevas en destino que no aparecen en origen como clave y su tipo como valor
    """
    # Se comprueba que no hay filas duplicadas en destino
    assert not destiny.duplicated().any(), 'Filas duplicadas en el destino'
    # Columnas nuevas en destino
    for column, dtype in new_columns.items():
        # Se comprueba que existen las nuevas columnas
        assert column in destiny.columns, f'Columna {column} no aparece en el destino'
        # Se comprueba que los tipos de las variables son los esperados
        assert destiny[column].dtype == dtype, f'Tipo de la columna {column} no tiene el tipo esperado {dtype}'


```

Para cada una de las dos tablas se llama a la función *season_etl_test* de la siguiente manera:

```
      # TemporadaPiloto
new_columns = {'year':'int64', 'driverRef':'object', 'constructorRef':'object', 'points_sum':'float64', 'scoreAI':'float64',
       'KPI_vs_compañero_equipo':'float64', 'KPI_lluvia':'float64', 'KPI_evitar_accidentes':'float64',
       'KPI_resultados_generales':'float64', 'KPI_regularidad':'float64', 'ranking':'float64',
       'CampeonatoPiloto':'int64'}

season_etl_test(TemporadaPiloto,new_columns)

# TemporadaEscuderia
new_columns = {'year':'int64', 'constructorRef':'object', 'points_sum':'float64', 'ranking':'float64',
       'CampeonatoConstructores':'int64', 'N_GPs':'int64'}

season_etl_test(TemporadaEscuderia,new_columns)


```

9.2. Pruebas de usabilidad

El objetivo principal de las pruebas de usabilidad es identificar y resolver posibles problemas o barreras que puedan dificultar la experiencia del usuario con la aplicación. Esto implica analizar la navegación, la disposición de los elementos, la claridad de la información presentada, la capacidad de filtrar y explorar datos, la comprensión de los gráficos y visualizaciones, entre otros aspectos relevantes. Durante estas pruebas, se suelen recopilar datos cualitativos y cuantitativos a través de la observación directa, cuestionarios, entrevistas o registros de interacción. Estos datos permiten a los diseñadores y desarrolladores del proyecto de BI obtener información valiosa sobre los puntos fuertes y débiles del diseño, así como identificar posibles mejoras.

9.3. Planificación de la evaluación de usabilidad

En esta sección se debe indicar qué se quiere evaluar de los dashboards y cómo se quiere evaluar.

Por cada cuadro de mando se muestra mucha información como se observa en el capítulo 8 de descripción de la aplicación *Fórmula 1 BI*. Por ello, a cada usuario no se le puede preguntar sobre todos los gráficos de la aplicación ya que sería una tarea muy tediosa para los usuarios. Por este motivo se ha creado un test que simula cinco contextos diferentes dentro de una carrera de Fórmula 1 y simula que los usuarios son narradores de carreras Fórmula 1 con la necesidad de realizar una tarea con la aplicación. A cada usuario se le asignan cinco tareas, una por cada dashboard realizado. Para poder probar la usabilidad de todos los gráficos de los cuadros de mando a cada usuario se le realiza un test diferente para que entre todos los usuarios se pueda evaluar la aplicación entera.

Para evaluarlo por cada se ha utilizado varias medidas de usabilidad: [95]

- **Tasa de finalización**: Se registra si la tarea ha resultado exitosa o fracasa.
- **Problemas de usabilidad (problemas de UI) encontrados**: Describe el problema que se ha encontrado con la aplicación.
- **Tiempo de la tarea**: Registra, en segundos o minutos, cuánto tiempo le toma a un usuario completar la tarea.
- **Nivel de satisfacción con la tarea**: Después de intentar una tarea, se le pide al usuario que responda sobre lo difícil que le fue la tarea.
- **Errores**: Registra cualquier acción no intencional, error u omisión que realice un usuario al intentar una tarea. Registra cada error junto con una descripción.
- **Expectativa**: Se registra si una tarea fue más o menos difícil que lo esperado.

Antes de explicar las tareas a desarrollar a cada usuario se le pregunta primero sí da su permiso para recoger sus respuestas que serán anonimizadas. Después se pregunta su grado de familiaridad con la Fórmula 1 y con aplicaciones de BI o de visualización de datos.

Finalmente tras intentar las tareas se recogen comentarios del usuario respecto a la aplicación.

9.4. Formulario de usabilidad

Para cada usuario se ha rellenado un formulario dónde se han registrado las cinco tareas qué debía completar y el resultado obtenido de la evaluación.

Se han realizado cinco pruebas de usabilidad a cinco usuarios diferentes con distinto grado de familiaridad con la Fórmula 1 y con aplicaciones de BI, (ver Figura 9.1 hasta la Figura 9.10).

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

Descripción usuario	
Id:	1
¿Da el usuario permiso para realizar este test de usabilidad y recoger sus repuestas las cuales serán totalmente anonimizadas?	Sí
Familiaridad con la Fórmula 1	Nula
Familiaridad con aplicaciones de BI o visualización de datos	Alta

Descripción del Test de Usabilidad de la aplicación Fórmula 1 BI

A continuación se les designará varias tareas concretas para realizar con la aplicación Fórmula 1 BI. Habrá únicamente una tarea por cada Dashboard de la aplicación.

Se intentará crear una narrativa simulando una retransmisión de Fórmula 1 real para dar contexto a las tareas.

Descripción de las tareas

CONTEXTO 1: "Bienvenidos hoy domingo comienza el Gran Premio de Australia, también llamado circuito de Albert Park, uno de los Gran Premios con más historia de la Fórmula 1**"**.

TAREA 1: " El usuario debe indicar el número de Grandes Premios (GPs) disputados en la historia en el circuito de Albert Park y la fecha de la primera carrera".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 25 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: No encontró la fecha en un principio y lo miró en la tabla, después se dio cuenta de que aparecía en la parte superior.
- **Expectativa de la tarea**: Menos difícil que lo esperado.

CONTEXTO 2: "Abandono del piloto británico Lando Norris de la escudería McLaren, parece que vuelve a ser por un problema mecánico, los cuales son muy comunes en McLaren estos últimos años**"**.

TAREA 2: "El usuario debe indicar el número de errores mecánicos respecto al número de errores por culpa del piloto en la escudería McLaren entre los años 2018 y 2022 (últimos 5 años)".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 15 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

Figura 9.1: Test de usabilidad para el usuario 1 (parte 1)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

CONTEXTO 3: "Está habiendo una pelea muy disputada por la tercera plaza entre los pilotos Sergio Pérez y George Russell pero ¿quién está pilotando mejor esta temporada según el Big Data?**"**.

TAREA 3: "El usuario debe comparar el scoreAI de ambos pilotos de la temporada de 2022 y decir en que habilidades es mejor George Russell con Sergio Pérez. Siendo scoreAI una métrica que evalúa pilotos en una temporada reduciendo el efecto del monoplaza que conducen, es una métrica 0-100 siendo 0 una mala puntuación de un piloto y un 100 una puntuación perfecta.

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 60 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

CONTEXTO 4: "Finalmente se alzó con la ansiada victoria el piloto asturiano Fernando Alonso tras una magistral salida y buen uso de los neumáticos".

TAREA 4: "El usuario debe indicar cuantas victorias podios y poles llevaba antes del Gran Premio Fernando Alonso".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 15 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

CONTEXTO 5: "Esta victoria de Alonso le consagra como uno de los mejores pilotos del siglo junto con otros pilotos como…".

TAREA 5: "El usuario debe indicar quien son los 5 mejores pilotos por scoreAI de la temporada del siglo XXI".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 15 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

COMENTARIOS FINALES SOBRE LA APLICACIÓN

- En el menú de Inicio que el área de selección sea el de la imagen no sólo el del texto
- Aumentar contraste imagen-texto en el menú de inicio
- Poner en mayúscula títulos de gráfico y títulos de leyendas
- Eliminar interacciones si pulsas gráficos como la bandera

Supervisado por:	Alberto González de la Plaza	Fecha:	05/06/2023

2

Figura 9.2: Test de usabilidad para el usuario 1 (parte 2)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

Descripción usuario	
Id	2
¿Da el usuario permiso para realizar este test de usabilidad y recoger sus repuestas las cuales serán totalmente anonimizadas?	Sí
Familiaridad con la Fórmula 1	Baja
Familiaridad con aplicaciones de BI o visualización de datos	Alta

Descripción del Test de Usabilidad de la aplicación Fórmula 1 BI

A continuación se les designará varias tareas concretas para realizar con la aplicación Fórmula 1 BI. Habrá únicamente una tarea por cada Dashboard de la aplicación.
Se intentará crear una narrativa simulando una retransmisión de Fórmula 1 real para dar contexto a las tareas.

Descripción de las tareas

CONTEXTO 1: "Bienvenidos hoy domingo comienza el Gran Premio de México en el circuito Hermanos Rodríguez**"**.

TAREA 1: " El usuario debe indicar en qué ciudad se encuentra el circuito rodríguez ".
Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 75 segundos.
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: No encontraba el botón de localización de manera intuitiva.
- **Expectativa de la tarea**: Más difícil que lo esperado.

CONTEXTO 2: "Esta temporada han aparecido nuevos pilotos como Sergeant, Nick de Vries o Piastri"
TAREA 2: "El usuario debe indicar cuantos pilotos ha habido en todas las temporadas de la Fórmula 1"
Métricas de usabilidad

- **Tasa de finalización**: Tarea fracasada.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: X .
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: No encontraba el dashboard correcto de Temporada, pensó que era en el de piloto, ha sido problema de cómo estaba formulada la pregunta.
- **Expectativa de la tarea**: Más difícil que lo esperado.

Figura 9.3: Test de usabilidad para el usuario 2 (parte 1)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

CONTEXTO 3: "Increíble la maniobra de Fernando Alonso y Lewis Hamilton para evitar el accidente con Romain Grosjean"

TAREA 3: "El usuario debe comparar la habilidad de evitar accidentes promedio de los pilotos Fernando Alonso y Lewis Hamilton a lo largo de toda la historia".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**:60 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: de difícil que lo esperado.

CONTEXTO 4: "Parece que con esta nueva victoria de Max Verstappen Red Bull podría lograr su récord de puntos en su historia y consiguiendo así su mundial de constructores número… ".

TAREA 4: "El usuario debe obtener el récord de puntos de la escudería Red Bull y el número de mundiales de constructores".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**:
- **Tiempo tarea**: 36 segundos.
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: no encontraba las métricas por culpa del error en la UI.
- **Expectativa de la tarea**: Más difícil que lo esperado.

CONTEXTO 5: "Esta temporada está siendo perfecta para el piloto holandés Max Verstappen que podría considerarse como una de las mejores temporadas de su historia ".

TAREA 5: "El usuario debe indicar la mejor temporada de Max Verstappen respecto a puntos y luego la mejor temporada respecto a scoreAI". Siendo scoreAI una métrica que evalúa pilotos en una temporada reduciendo el efecto del monoplaza que conducen, es una métrica 0-100 siendo 0 una mala puntuación de un piloto y un 100 una puntuación perfecta.

Métricas de usabilidad

- **Tasa de finalización**: Tarea fracasada.
- **Problemas encontrados de UI y su severidad**: Error en el Bookmark Métricas que estaba seleccionado por defecto cuando debía estar seleccionado Resultados.
- **Tiempo tarea**: X segundos.
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: No encontraba las métricas por culpa del error en la UI.
- **Expectativa de la tarea**: Más de difícil que lo esperado.

COMENTARIOS FINALES SOBRE LA APLICACIÓN

- Explicar que las métricas van de 0-100 y que 100 es lo mejor
- Cambiar números en blanco por guion.
- Llamar campeonato de constructores al campeonato de escuderías.
- Si no tiene velocidad que no aparezca.

Supervisado por:	Alberto González de la Plaza	Fecha:	05/06/2023

Figura 9.4: Test de usabilidad para el usuario 2 (parte 2)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

Descripción usuario	
Id	3
¿Da el usuario permiso para realizar este test de usabilidad y recoger sus repuestas las cuales serán totalmente anonimizadas?	Sí
Familiaridad con la Fórmula 1	Alta
Familiaridad con aplicaciones de BI o visualización de datos	Media

Descripción del Test de Usabilidad de la aplicación Fórmula 1 BI

A continuación se les designará varias tareas concretas para realizar con la aplicación Fórmula 1 BI. Habrá únicamente una tarea por cada Dashboard de la aplicación.

Se intentará crear una narrativa simulando una retransmisión de Fórmula 1 real para dar contexto a las tareas.

Descripción de las tareas

CONTEXTO 1: "Bienvenidos hoy domingo comienza el Gran Premio de Mónaco uno de los circuitos más lentos de la F1".

TAREA 1: " El usuario debe indicar la velocidad máxima alcanzada en el Gran Premio de Mónaco y decir quién fue el que alcanzó esa velocidad y compararlo con la máxima de la historia de la F1 en cualquier circuito".

Métricas de usabilidad

- **Tasa de finalización**: Tarea fracasada.
- **Problemas encontrados de UI y su severidad**: Si la velocidad Max o es la velocidad Max.
- **Tiempo tarea**: X.
- **Nivel satisfacción con la tarea**: Media.
- **Errores del usuario**: No encontrar la velocidad máxima.
- **Expectativa de la tarea**: Más difícil que lo esperado.

CONTEXTO 2: "Terrible accidente en la primera vuelta de Romain Grojean realizando una maniobra muy peligrosa, no es la primera vez que provoca un accidente de este calibre".

TAREA 2: "El usuario debe indicar cuantos abandonos por causa error del piloto ha estado involucrado y si el piloto posee buena métrica en los últimos años evitando accidentes".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 20 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Menos difícil que lo esperado.

Figura 9.5: Test de usabilidad para el usuario 3 (parte 1)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

CONTEXTO 3: " Vaya espectáculo de Ferrari esta carrera en el que de momento van primero Charles Leclerc y segundo Carlos Sainz dando una lección de pilotaje, Ferrari a lo largo de la historia ha tenido a muy buenos pilotos como Fangio, Schumacher o Fernando Alonso ¿pero quién es el piloto con la mejor temporada suya respecto a resultados y respecto a las métricas de Ferrari Big Data?**"**.

TAREA 3: "El usuario debe indicar la mejor temporada de un piloto de Ferrari según scoreAI". Siendo scoreAI una métrica que evalúa pilotos en una temporada reduciendo el efecto del monoplaza que conducen, es una métrica 0-100 siendo 0 una mala puntuación de un piloto y un 100 una puntuación perfecta.

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Año por defecto.
- **Tiempo tarea**: 70 segundos.
- **Nivel satisfacción con la tarea**: Media.
- **Errores del usuario**: Ha supuesto que estaban todos los años seleccionados.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

CONTEXTO 4: "Comienza a llover en Mónaco la carrera está muy interesante por ver quién entra al podio entre los pilotos británicos Lewis Hamilton y Lando Norris ¿A cuál de los dos pilotos le favorece más la lluvia?"

TAREA 4: "El usuario debe comparar la habilidad en lluvia de los pilotos Lando Norris y Lewis Hamilton en los últimos 3 años (2020-2022).

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 50 segundos.
- **Nivel satisfacción con la tarea**: Media.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Menos difícil que lo esperado.

CONTEXTO 5: "La fórmula 1 es un deporte en el que influye mucho el coche y los resultados globales no indican fielmente la calidad de un piloto"

TAREA 5: "El usuario debe indicar dos pilotos sobrevalorados hubo en 2021 (mal scoreAI pero buen coche)".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 80 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

COMENTARIOS FINALES SOBRE LA APLICACIÓN

- Cambiar nombre animación por comparación
- Escribir por teclado nombre piloto
- Indicar con texto mínimo o máximo en el medidor radial
- Indicar que la velocidad máxima es la velocidad media por vuelta, no el máximo alcanzado

Supervisado por:	Alberto González de la Plaza	Fecha:	07/06/2023

Figura 9.6: Test de usabilidad para el usuario 3 (parte 2)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

Descripción usuario	
Id	4
¿Da el usuario permiso para realizar este test de usabilidad y recoger sus repuestas las cuales serán totalmente anonimizadas?	Sí
Familiaridad con la Fórmula 1	Media
Familiaridad con aplicaciones de BI o visualización de datos	Media

Descripción del Test de Usabilidad de la aplicación Fórmula 1 BI

A continuación se les designará varias tareas concretas para realizar con la aplicación Fórmula 1 BI. Habrá únicamente una tarea por cada Dashboard de la aplicación.

Se intentará crear una narrativa simulando una retransmisión de Fórmula 1 real para dar contexto a las tareas.

Descripción de las tareas

Contexto 1: "Bienvenidos hoy domingo comienza el Gran Premio de Spa y parece que va a llover durante toda la carrera, es muy común la lluvia en este circuito**"**.

TAREA 1: " El usuario debe indicar el número de veces de que ha llovido en el circuito de Spa en la historia e indicar cuando fue la última carrera donde hubo lluvia".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 75 segundos.
- **Nivel satisfacción con la tarea**: Medio.
- **Errores del usuario**: Ha tardado en encontrar la última carrera.
- **Expectativa de la tarea**: Más difícil que lo esperado.

CONTEXTO 2: "Increíble vuelta de Charles Leclerc con el Ferrari en el circuito de Spa con 1:47:212 superando su mejor vuelta en Spa con una marca de … en el año…".

TAREA 2: "El usuario debe indicar el tiempo de la vuelta rápida de Charles Leclerc en el circuito de Spa y cuándo fue".

Métricas de usabilidad

- **Tasa de finalización**: Tarea fracasada.
- **Problemas encontrados de UI y su severidad**: La bandera de Mónaco no aparece en la bandera del dashboard, investigar si es problema del gráfico o de los datos. Problema de interactividad al pulsar en la tabla de métricas de piloto
- **Tiempo tarea**: X.
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: No ha encontrado el dato de vuelta rápida.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

Figura 9.7: Test de usabilidad para el usuario 4 (parte 1)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

CONTEXTO 3: **"**Otra carrera en lluvia durante esta temporada que gana Red Bull increíble lo de esta escudería, ¿Cuántas carreras con lluvia ha habido este año y que escuderías las han ganado?**"**.

TAREA 3: "El usuario debe indicar cuantas carreras ha habido con lluvia y qué escuderías las han ganado en toda la temporada de 2022".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**:85 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

CONTEXTO 4: **"**Parece que en el público se encuentra el legendario Alain Prost una leyenda de la Fórmula 1 que realizó uno de los duelos más bonitos con el fallecido Ayrton Senna"

TAREA 4: "El usuario debe comparar la mejor temporada de cada piloto según scoreAI no el promedio"

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 75 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: de difícil que lo esperado.

CONTEXTO 5: "Este año el monoplaza de Red Bull es muy rápido en recta gracias al nuevo motor Honda que han diseñado"

TAREA 5: "El usuario debe indicar en que carreras ha sido dónde Red Bull ha alcanzado más y menos velocidad en la temporada 2022".

Métricas de usabilidad

- **Tasa de finalización**: Tarea fracasada.
- **Problemas encontrados de UI y su severidad**: .
- **Tiempo tarea**: 15 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ha confundido la medida mínima del velocímetro con la mínima en esa temporada.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

COMENTARIOS FINALES SOBRE LA APLICACIÓN

- Necesita poner una etiqueta o tooltip donde el velocímetro
- Que por defecto aparezcan todos los años

Supervisado por:	Alberto González de la Plaza	Fecha:	07/06/2023

Figura 9.8: Test de usabilidad para el usuario 4 (parte 2)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

Descripción usuario	
Id	5
¿Da el usuario permiso para realizar este test de usabilidad y recoger sus repuestas las cuales serán totalmente anonimizadas?	Sí
Familiaridad con la Fórmula 1	Media
Familiaridad con aplicaciones de BI o visualización de datos	Alta

Descripción del Test de Usabilidad de la aplicación Fórmula 1 BI

A continuación se les designará varias tareas concretas para realizar con la aplicación Fórmula 1 BI. Habrá únicamente una tarea por cada Dashboard de la aplicación.
Se intentará crear una narrativa simulando una retransmisión de Fórmula 1 real para dar contexto a las tareas.

Descripción de las tareas

Contexto 1: "Bienvenidos hoy domingo comienza el Gran Premio de Silverstone, hoy parte desde la pole Lewis Hamilton con el Mercedes**"**.

TAREA 1: "El usuario debe indicar quién han sido los pilotos con más poles en la historia del Gran Premio de Silverstone y con qué escuderías lo han logrado".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 45 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: .
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

CONTEXTO 2: "Que mala suerte de Fernando Alonso que tiene que abandonar la carrera por un error de escudería, esta temporada parece que va a superar en abandonos por culpa de la escudería a la temporada de 2017 con McLaren".

TAREA 2: "El usuario debe comparar el número de abandonos en 2017 y compararlo con los de 2022".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**:
- **Tiempo tarea**: 100 segundos.
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: No ha intuido que estaba en el dashboard de Piloto, ha buscado en el de Temporada.
- **Expectativa de la tarea**: Más de difícil que lo esperado.

Figura 9.9: Test de usabilidad para el usuario 5 (parte 1)

TEST USABILIDAD FÓRMULA 1 BI – BI ANALYTICS

CONTEXTO 3: "Hoy se ha visto una carrera muy accidentada y rápida en Silverstone, ¿cuáles han sido los circuitos más rápidos y peligrosos de la temporada?"

TAREA 3: "El usuario debe indicar cuales han sido los circuitos más rápidos y peligrosos de la temporada 2022".

Métricas de usabilidad

- **Tasa de finalización**: Tarea fracasada.
- **Problemas encontrados de UI y su severidad**:
- **Tiempo tarea**: X.
- **Nivel satisfacción con la tarea**: Difícil.
- **Errores del usuario**: No entendía que podía acceder con el botón "animación"..
- **Expectativa de la tarea**: Más de difícil que lo esperado.

CONTEXTO 4: "Finalmente victoria para Max Verstappen seguido de Fernando Alonso, ¿ha superado el piloto holandés al piloto español en victorias y en podios?"

TAREA 4: "El usuario debe comparar el número de victorias y de podios entre ambos pilotos, también se desea observar la distribución de los podios de cada uno para ver quien tiene más segundos puestos y terceros"

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**:90 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Menos difícil que lo esperado.

CONTEXTO 5: "Finalizamos la carrera muy accidentada con dos accidentes provocados por los pilotos de Williams "

TAREA 5: "El usuario debe indicar los peores pilotos de Williams evitando accidentes en los últimos cinco años (2018-2022)".

Métricas de usabilidad

- **Tasa de finalización**: Tarea exitosa.
- **Problemas encontrados de UI y su severidad**: Ninguno.
- **Tiempo tarea**: 75 segundos.
- **Nivel satisfacción con la tarea**: Fácil.
- **Errores del usuario**: Ninguno.
- **Expectativa de la tarea**: Igual de difícil que lo esperado.

COMENTARIOS FINALES SOBRE LA APLICACIÓN

- Cambiar nombre del botón de la animación en el dashboard Temporada
- Cambiar los buscadores en formato lista por formato texto
- Que por defecto aparezcan todos los años

Supervisado por:	Alberto González de la Plaza	Fecha:	07/06/2023

Figura 9.10: Test de usabilidad para el usuario 5 (parte 2)

9.5. Análisis de resultados

Analizando los resultados de la prueba de usabilidad descrita en el apartado anterior se han podido obtener varios resultados interesantes ya que ha permitido arreglar algunos errores concretos cómo los siguientes:

- Por defecto deben aparecer seleccionado todos los años desde 1950 hasta la actualidad.
- Cambiar el nombre de ciertas etiquetas como animación porque no son intuitivas para el usuario.
- Necesidad de explicar mediante etiqueta o mediante tooltip algún dato del dashboard como el de velocidad máxima de todos los circuitos.
- Indicar al usuario que velocidad máxima no es el pico de velocidad alcanzada por un piloto sino la máxima de la velocidad media de una vuelta.
- Filtrar el gráfico de velocidad por carrera que si no ha corrido entre los años 2005 hasta la actualidad (años en los que se dispone la máxima de la velocidad media por vuelta) no aparezcan los datos.

Otras sugerencias como la de incluir entrada por texto en los filtros de piloto o circuito se han descartado porque el filtro de selección única en Power BI no permite entrada de texto y el filtro especial para entrada de texto no permite autocompletado. También se ha descartado otras recomendaciones que por limitación de la herramienta de Power BI no se puede realizar ya que hay que adaptarse a las restricciones de cada gráfico que no dan tanta libertad al desarrollador. Algunas de las recomendaciones descartadas han sido: que salten mensajes de aviso en ciertos momentos según los datos o realizar varios tooltips para un único gráfico para explicar todos los elementos del gráfico.

Capítulo 10

Conclusiones

En esta última sección se exponen las conclusiones derivadas de la realización del proyecto y se presentan las líneas de trabajo a futuro.

10.1. Grado de consecución de los objetivos

Los objetivos planteados en el apartado 1.3 se han alcanzado totalmente. Tras meses de trabajo, se ha logrado crear una aplicación de visualización de datos siguiendo una planificación de un proyecto de Business Intelligence con una interfaz moderna y usable. De esta manera se cumplen los requisitos propuestos por el cliente, asemejándose a los mockups creados previamente. Esta aplicación utiliza múltiples gráficos sin excesiva complejidad para el usuario que ayudan a una interpretación rápida de las visualizaciones.

Se ha conseguido realizar un proceso automático de ETL, desde la extracción de datos de una API mediante Python a todas las tareas de transformación y limpieza de los datos.

También se ha definido una métrica para evaluar pilotos en un año solicitada por el cliente y además se ha logrado que sea una métrica explicable mediante la creación de las métricas parciales las cuales explican el porqué de la puntuación de la métrica general.

10.2. Dificultades encontradas

En el transcurso del proyecto han aparecido ciertos obstáculos que se han debido solventar. Algunos de ellos han estado relacionados con la calidad de los datos. Pese a que los datos originales están bastante tratados y limpios se han encontrado varias variables con una cantidad ingente de outliers, como eran las variables de tiempos de vuelta y de pit stops. Esto ha provocado que se eliminen de los cuadros de mando ya que no se podía saber cuando había sido un mal pit stop o cuando era un dato erróneo. Este fue el riesgo más peligroso que se analizó en la etapa de análisis de riesgos del apartado 3.4.

Otros de los dos riesgos identificados con los que había que tener precaución y que han provocado ciertas dificultades, fueron el de realizar una planificación poco realista ya que hubo tareas como la documentación o la etapa de la ETL, que se han demorado más de lo previsto. Además los requisitos del proyecto fueron cambiando con el tiempo lo que provocó en cambios en la versión final de la aplicación respecto a los mockups iniciales.

Estos tres riesgos fueron identificados previamente en la etapa de análisis de riesgos del apartado 3.4. Gracias a las respuestas definidas no causaron graves problemas en el proyecto, aún así han producido ligeras desviaciones en la planificación inicial las cuales se detallarán en el siguiente apartado.

Por último una dificultad no tenida en cuenta en los riesgos ha sido acostumbrarse a la poca libertad de la herramienta utilizada para la creación de visualizaciones, Power BI, la cuál tiene las visualizaciones definidas y hay que adaptarse a los requisitos de cada gráfico. Power BI no proporciona la misma libertad que otras librerías como D3 en Javascript o Shiny en R, en las cuáles la libertad de personalización es máxima. Esto ha provocado que hayan tenido que rehacerse gráficos y cambiar la estructura de aplicación final respecto a algunos mockups.

10.3. Desviaciones sobre la planificación

El proyecto comenzó el 16 de enero de 2023 y su fecha final estimada era el 9 de junio del mismo año pero finalmente con las dificultades encontradas en el apartado anterior se ha retrasado 10 días hasta el 19 de junio. Este retraso no ha sido considerado un grave problema ya que se estimó terminar dos semanas antes de la fecha límite por si surgía algún retraso en el proyecto.

En la Figura 10.1 se muestra la diferencia en los diagramas de Gantt previsto frente al diagrama de Gantt real con los retrasos.

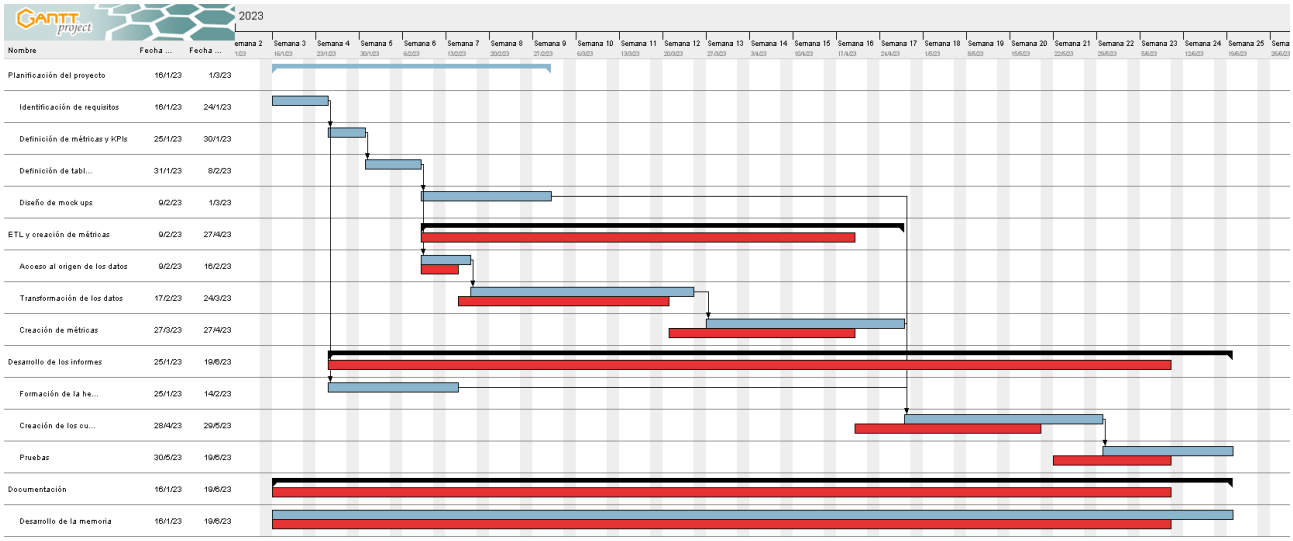


Figura 10.1: Diagrama de Gantt real con retrasos (azul) frente al diagrama Gantt de la planificación (rojo)

Algunas de las dificultades encontradas descritas en el apartado anterior como la de realizar una previsión poco realista especialmente de la etapa de documentación y de ETL no sólo han llevado a retrasar unos días la duración del proyecto sino que el proyecto ha durado más de las 300 horas planificadas. Durando en torno a las 400 horas lo que ha llevado a un aumento del presupuesto total, ya que en las últimas semanas se han aumentado las 15 horas semanales previstas en planificación.

Por tanto los recursos humanos han aumentado de los 5954.4 € estimados en la etapa de planificación, (ver Tabla 3.2) hasta los 7977.7 €, (ver Tabla 10.1).

Rol	**Tiempo (h)**	**Coste (€/h)**	**Coste total (€)**
Business Analyst	150	19.23	2884.5
Data Scientist	130	23.08	3000.4
Business Intelligence Analyst	120	17.44	2092.8
Total	400		7977.7

Tabla 10.1: Desglose de los recursos humanos real

Por tanto obtenido el coste total de los recursos humanos se presenta el presupuesto total para el desarrollo del proyecto el cuál es de 8512.45 €, (ver Tabla 10.2), frente a los 6355.48 € que fueron estimados en la etapa de planificación, (ver Tabla 3.3).

Recurso	Unidades equivalentes	€Unidad	€Total
Recurso humano	400 horas	Ver Tabla 10.1	7977.7
Oficina coworking	2.5[1] meses	125€/mes [67]	312.5
Ordenador renting	2.5[1] meses	50 €/mes	125
Monitor renting	2.5[1] meses	24 €/mes	60
Licencia Windows 10	1	260 €	10.83[2]
Licencia Power BI Pro	2.5[1] meses	9.40 €/mes	23.5
Licencia Office	1	70 €	2.92[2]
Total			8512.45

Tabla 10.2: Presupuesto total del proyecto

[1] Si fuera un proyecto real en el que se trabaje 40 horas semanales (160 horas mensuales) habría durado el proyecto $\frac{400}{160} = 2{,}5$ meses en vez de 6 como es lo que está planeado. Por ese motivo no se utiliza el ordenador o el monitor el 100 % del tiempo y hay que tenerlo en cuenta en los costes.

[2] Al ser un precio fijo hay que tener en cuenta los años de amortización (el software se estima que se amortiza en 5 años = 60 meses). Como se van a utilizar 2.5 meses el precio final será igual a $\frac{Preciofijo \cdot 2{,}5}{60}$.

10.4. Conocimientos aplicados y aprendizajes obtenidos

El desarrollo de este Trabajo de Fin de Grado ha supuesto una puesta en práctica mediante un caso práctico de una gran cantidad de contenidos vistos en diferentes asignaturas, en particular:

- Fundamentos de Programación: Básico para cualquier proyecto en el que haya que programar.
- Interacción Persona Computadora y Diseño y Evaluación de Sistemas Interactivos: Para la gestión de sistemas con interacción y aplicaciones. También para conocer los patrones y diseños para crear aplicaciones con usabilidad.
- Sistemas de Integración de la información: Para realizar correctamente el proceso de la ETL desde la obtención de los datos en los orígenes, siguiendo por la transformación para garantizar la calidad de los datos y por último la carga de los datos en la que los datos ya transformados pasan al entorno de explotación.
- Planificación y Diseño de Sistemas Computacionales y Fundamentos de Ingeniería de Software: Para la planificación y gestión del proyecto.
- Fundamentos de Inteligencia Artificial, Ingeniería del Conocimiento, Técnicas de Aprendizaje Automático y Minería de Datos: Para las técnicas de limpieza y la posterior creación de métricas a partir de los datos.
- También ciertas asignaturas del Grado en Estadística que también estoy cursando como pueden ser Estadística Descriptiva o computación estadística: Para el correcto uso de los gráficos para cada momento concreto.

En el presente apartado se detallan las competencias adquiridas o reforzadas conseguidas a lo largo del Trabajo de Fin de Grado:

- Se ha aprendido a desarrollar un proyecto de Business Intelligence de principio a fin, comenzando con las reuniones para la toma de requisitos con los clientes hasta finalizar creando la aplicación de visualización.
- Se han aumentado los conocimientos de la herramienta de Business Intelligence Power BI, especialmente el uso de medidas DAX para la creación de métricas que no les afecten los filtros y la navegación de marcadores (Bookmarks) para intercambiar entre varios gráficos y que los cuadros de mando sean más completos sin estar sobrecargados.

- Se ha aprendido a crear cuadros de mando usables, siguiendo patrones y guías de diseño específicas para cuadros de mando.
- Se han aumentado los conocimientos de programación en Python especialmente realizar web scraping con la librería *Beautiful Soup*.

10.5. Trabajo Futuro

El proyecto expuesto en este documento cuenta con diversas ampliaciones futuras, sobre todo en el ámbito de la creación de nuevas métricas y visualizaciones, ya que la información generada por la Fórmula 1 es inmensa, sólo hace falta analizarla profundamente. Las principales líneas a seguir son:

- Enriquecer la métrica *scoreAI* de pilotos mediante la adicción de nuevas variables como pueden ser "habilidad cuidando neumáticos" o "adaptabilidad frente a cambios de escudería".
- Crear una métrica similar a *scoreAI* pero de escuderías, no se ha realizado porque se consideraba más interesante para pilotos ya que la métrica para escuderías con los datos disponibles estaría muy correlacionada con la puntuación final. Para poder crearla correctamente habría que ir más allá y disponer de información sin outliers sobre los pit stops, presupuesto de la escudería, calidad del motor...
- Acceso a información de la telemetría en tiempo real, esto permitiría observar información de la carrera en directo y crear nuevas métricas.
- Ampliar la información de la aplicación añadiendo Fórmula 2 y categorías inferiores.
- Crear un algoritmo de similaridad entre pilotos y observar a quién se parecen las futuras promesas de la Fórmula 2 y categorías inferiores, tanto históricamente como pilotos actuales.

Anexo A

Contenido digital

A.1. Contenido del repositorio

El repositorio se encuentra en la cuenta personal del alumno en el GitLab de la Escuela [1]. En el repositorio, se puede encontrar:

- datos: Directorio con los datos utilizados, tiene dos subdirectorios.
  - origin: Contiene los 14 archivos CSV descargados de kaggle sin tratar.
  - destination: Destino de los datos. Contiene 8 archivos CSV que forman el destino.
- app.pbix: Archivo de Power BI con la aplicación *Fórmula 1 BI* [2].
- ETL.ipynb: Archivo Jupyter Notebook con las órdenes necesarias para realizar el flujo ETL.
- testETL.ipynb: Archivo Jupyter Notebook con las órdenes llevadas a cabo para realizar las pruebas de la ETL.

A.2. Manual de instalación

Para visualizar la aplicación *Fórmula 1 BI* simplemente, hay que entrar en el enlace generado con Power BI y disponible en el pie de página. No es necesario la instalación de ningún programa ni aplicación para su ejecución.

Los datos del repositorio están actualizados en mayo de 2023 aunque todavía no están añadidas las carreras de 2023 en el origen. Pero si se desea ejecutar la ETL o las pruebas ETL realizadas en Python 3.10.7 para comprobar que se han realizado correctamente, primero se deben descargar los archivos del repositorio, basta con acceder al repositorio y descargar todos los directorios.

Los siguientes paquetes se han utilizado para el desarrollo de la aplicación. Para instalarlos se debe escribir la siguiente orden dentro de una celda de Jupyter Notebook y después ejecutarla:

```
!pip install *paquete*
```

- *Beautiful Soup*: 4.11.1
- *numpy*: 1.23.5
- *Pandas*: 1.5.3
- *requests*: 2.28.1

Para ejecutar los archivos de ETL.ipynb y testETL.ipynb al ser ficheros Jupyter Notebook simplemente hay que ejecutar las celdas en orden.

[1] `https://gitlab.inf.uva.es/albegon/tfg_inf_alberto_gonzalez`
[2] Enlace a la visualización de Power BI

Bibliografía

[1] FIA. *Formula 1 announces TV, race attendance and digital audience figures for 2021*. 2022. URL: https://www.formula1.com/en/latest/article.formula-1-announces-tv-race-attendance-and-digital-audience-figures-for-2021.1YDpVJIOHGNuok907sWcKW.html (Último acceso 10-02-2023).

[2] Maury Brown. «Inside The Numbers That Show Formula 1's Popularity And Financial Growth». En: *Forbes* (mar. de 2023). URL: https://www.forbes.com/sites/maurybrown/2023/03/29/inside-the-numbers-that-show-formula-1s-popularity-and-financial-growth/?sh=7b2301a14df6 (Último acceso 10-02-2023).

[3] Media. «What is Formula 1?» En: *f1 Chronicle* (abr. de 2021). URL: https://f1chronicle.com/what-is-formula-1/?utm_content=cmp-true (Último acceso 24-05-2023).

[4] Glenn B. Manishin. «F1 Origins». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=222 (Último acceso 26-05-2023).

[5] Glenn B. Manishin. «The Early Years». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=272 (Último acceso 26-05-2023).

[6] motorsport. *Juan Manuel Fangio, el primer multicampeón de F1*. 2022. URL: https://es.motorsport.com/f1/news/estadisticas-fangio-muerte-nacimiento-791118/791118/#gal-791118-m0-juan-manuel-fangio-4 (Último acceso 26-05-2023).

[7] Glenn B. Manishin. «The British Era». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=923 (Último acceso 26-05-2023).

[8] motorsport. *GALERÍA: a 54 años de la muerte de Jim Clark*. 2022. URL: https://lat.motorsport.com/f1/news/galeria-a-54-anos-de-la-muerte-de-jim-clark-890796/1622765/ (Último acceso 26-05-2023).

[9] Glenn B. Manishin. «Wings and Ground Effects». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=1643 (Último acceso 26-05-2023).

[10] Glenn B. Manishin. «The turbo era». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=1706 (Último acceso 26-05-2023).

[11] Glenn B. Manishin. «The Active Cars». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=1742 (Último acceso 26-05-2023).

[12] TyC Sports. *Senna-Prost, el duelo que renace en el juego de la F1*. 2019. URL: https://www.tycsports.com/nota/player-one/2019/05/07/senna-prost-el-duelo-que-renace-en-el-juego-de-la-f1.html (Último acceso 26-05-2023).

[13] Glenn B. Manishin. «After Tamburello». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=1796 (Último acceso 26-05-2023).

[14] Glenn B. Manishin. «Scuderia Resurgent». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=1887 (Último acceso 26-05-2023).

[15] motorsport. *Alonso vuelve a Turquía con el espíritu de aquella lucha con Schumacher*. 2021. URL: https://es.motorsport.com/f1/news/alonso-previo-turquia-declaraciones-schumacher/6681539/ (Último acceso 26-05-2023).

[16] Glenn B. Manishin. «Into the Void». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=30235 (Último acceso 26-05-2023).

[17] Glenn B. Manishin. «The Hybrid Era». En: *Formula One Art and Genius, The magazine* (2018). URL: http://www.f1-grandprix.com/?page_id=48636 (Último acceso 26-05-2023).

[18] Jack Leslie. *The F1 Evolution*. 2013. URL: http://www.jackleslief1.co.uk/2013/01/the-f1-evolution.html (Último acceso 26-05-2023).

[19] FIA. *Governance*. 2023. URL: https://www.fia.com/governance (Último acceso 25-05-2023).

[20] Steven De Groote. «Driver equipment». En: *F1 Technical* (jun. de 2006). URL: https://www.f1technical.net/articles/17 (Último acceso 24-05-2023).

[21] Massimiliano Vitelli. «Cómo se fabrican los trajes de los pilotos de Fórmula 1». En: *Revista CQ* (2022). URL: https://www.revistagq.com/deporte/articulo/como-se-fabrican-los-trajes-de-los-pilotos-de-formula-1 (Último acceso 26-05-2023).

[22] FormulaF1. *Circuito de Silverstone (Gran Bretaña)*. 2010. URL: https://www.formulaf1.es/6594/circuito-de-silverstone-gran-bretana/ (Último acceso 26-05-2023).

[23] FormulaF1. *Circuito de Monaco*. 2008. URL: https://www.formulaf1.es/520/circuito-de-monaco/ (Último acceso 24-05-2023).

[24] motorsport. *FIA track grades: Requirements to hold an F1 race, potential tracks*. 2021. URL: https://www.motorsport.com/f1/news/fia-track-grades-requirements-f1-potential/6508330/ (Último acceso 24-05-2023).

[25] Media. «How Does Formula 1 Qualifying Work?» En: *f1 Chronicle* (dic. de 2022). URL: https://f1chronicle.com/how-does-formula-1-qualifying-work/#How-long-does-F1-qualifying-last (Último acceso 24-05-2023).

[26] Nathan Evans. «What is F1 Sprint? Schedule, Shootout changes, points system and how format works in 2023». En: *The Sporting News* (abr. de 2023). URL: https://www.sportingnews.com/us/formula-1/news/f1-sprint-schedule-shootout-changes-points-system-format-2023/qwjsrxpokfhzgegecmyiowto (Último acceso 24-05-2023).

[27] Nate Saunders. *What is the Sprint Shootout? F1's new sprint race weekend format explained*. Abr. de 2021. URL: https://www.espn.com/f1/story/_/id/36288097/what-sprint-shootout-f1-new-sprint-race-weekend-format-explained (Último acceso 28-04-2023).

[28] Wikipedia. *Sistemas de puntuación de Fórmula 1*. URL: https://es.wikipedia.org/wiki/Sistemas_de_puntuaci%C3%B3n_de_F%C3%B3rmula_1#:~:text=Otorga%2025%20puntos%20al%20primer,y%201%20al%20d%C3%A9cimo%20clasificado. (Último acceso 25-05-2023).

[29] laformula1esmipasion. *Como se han repartido los puntos en la F1 a traves de las temporadas?* 2014. URL: https://laformula1esmipasion.wordpress.com/2014/03/29/como-se-han-repartido-los-puntos-en-la-f1-a-traves-de-los-anos/ (Último acceso 25-05-2023).

[30] James W Roberts. *Aston Martin Red Bull Racing tiene el equipo de F1 más rápido de la Tierra*. 2019. URL: https://www.redbull.com/es-es/red-bull-record-pit-stop (Último acceso 25-05-2023).

[31] motorsport. *Red Bull hizo con Sergio Pérez el pitstop más rápido de la F1 2022*. 2022. URL: https://espanol.motorsport.com/f1/news/red-bull-pitstop-mas-rapido-perez-formula1-2022/10347380/ (Último acceso 25-05-2023).

[32] Brian Silvestro. «Here's What Every Button on a Modern F1 Steering Wheel Does». En: *Road and track* (2019). URL: https://www.roadandtrack.com/motorsports/a26827434/2019-mercedes-f1-steering-wheel-explained/ (Último acceso 25-05-2023).

[33] Olivia Le Poidevin. «What is halo and how does it save lives?» En: *BBC* (jul. de 2022). URL: https://www.bbc.com/news/newsbeat-62037334 (Último acceso 25-05-2023).

[34] tutorials point. *FIA track grades: Requirements to hold an F1 race, potential tracks*. URL: https://www.tutorialspoint.com/formula_one/car_design_specs_rules.htm (Último acceso 25-05-2023).

[35] motorsport. *Insider's guide: How is an F1 car made?* 2022. URL: https://us.motorsport.com/f1/news/how-is-an-f1-car-made/7626324/ (Último acceso 25-05-2023).

[36] filmaffinity. *Rush*. 2013. URL: https://www.filmaffinity.com/es/film374668.html (Último acceso 25-05-2023).

[37] Ben Tippett. «The Complete Guide To Understanding Formula 1». En: *Defector* (abr. de 2021). URL: https://defector.com/the-complete-guide-to-understanding-formula-1 (Último acceso 25-05-2023).

[38] Ben Miller. «F1 flags explained: What different colours mean in Formula One 2023 season». En: *The Sporting News* (abr. de 2023). URL: https://www.sportingnews.com/us/formula-1/news/f1-flags-explained-colours-formula-one/enkyw7v7rgpfaacav5yhrkek (Último acceso 25-05-2023).

[39] Wikipedia. *Banderas formula 1*. URL: https://es.wikipedia.org/wiki/Archivo:Banderas_formula_1.jpg (Último acceso 25-05-2023).

[40] Mauro Mariani. «¿Qué es el safety car y qué función cumple en las carreras de Fórmula 1?» En: *The Sporting News* (jun. de 2022). URL: https://www.sportingnews.com/es/formula-1/news/que-es-el-safety-car-f1-que-funcion-cumple/giuekzb9kcyagwh0qva3bwlt (Último acceso 25-05-2023).

[41] automundo. *La FIA les aclara a los pilotos de la Fórmula 1 cuál es la función del safety car*. 2022. URL: https://automundo.com.ar/la-fia-les-aclara-a-los-pilotos-de-la-formula-1-cual-es-la-funcion-del-safety-car/ (Último acceso 25-05-2023).

[42] Karim Ahmad. «7 Times F1 Tech Ended Up In Your Road Car». En: *Make Use Of* (jun. de 2022). URL: https://www.makeuseof.com/f1-tech-in-road-car/ (Último acceso 26-05-2023).

[43] Joel Shapiro. «Data Driven at 200 MPH: How Analytics Transforms Formula One Racing». En: *Forbes* (ene. de 2023). URL: https://www.forbes.com/sites/joelshapiro/2023/01/26/data-driven-at-200-mph-how-analytics-transforms-formula-one-racing/?sh=4b2b94bf39db (Último acceso 26-05-2023).

[44] FIA. *Formula 1 Partners*. 2023. URL: https://www.formula1.com/en/toolbar/partners.html (Último acceso 26-05-2023).

[45] Carlo Revelli. *Intelligence stratégique sur Internet*. DUNOD, 1998.

[46] Gartner. *Analytics and Business Intelligence (ABI)*. URL: https://www.gartner.com/en/information-technology/glossary/business-intelligence-bi (Último acceso 01-06-2023).

[47] Nedim Dedić y Clare Stanier. *Measuring the success of changes to existing business intelligence solutions to improve business intelligence reporting*. Springer, 2016.

[48] Cristina Lago. «150 years of business intelligence: A brief history». En: *CIO* (jul. de 2018). URL: https://www.cio.com/article/221963/history-of-business-intelligence.html (Último acceso 01-06-2023).

[49] Paulo Limpio. *Exploring the History of Business Intelligence*. URL: https://www.toptal.com/project-managers/it/history-of-business-intelligence (Último acceso 01-06-2023).

[50] Richard Miller Devens. *Cyclopaedia of Commercial and Business Anecdotes*. Springer, 1865.

[51] Derek Rodner. *The History of Business Intelligence Infographic*. 2019. URL: https://blog.agilenceinc.com/the-history-of-business-intelligence-infographic (Último acceso 01-06-2023).

[52] Thomas Davenport y Laurence Prusak. *Working Knowledge: How Organizations Manage what They Know*. Harvard Business Review Press, 1998.

[53] Sinnexus. *Datos, información, conocimiento.* URL: https://www.sinnexus.com/business_intelligence/piramide_negocio.aspx (Último acceso 01-06-2023).

[54] Alberto Rodríguez Rodríguez y Elizabeth Bernal Gamboa. *Gestión de la información cuantitativa en las universidades.* Universidad Nacional de Colombia, 2019. ISBN: 978-958-783-805-3. URL: https://estadisticaun.github.io/L_Conceptual/2-4-inteligencia-de-negocios.html.

[55] IMF. *Tipos de datos: datos estructurados, semiestructurados y no estructurados.* URL: https://blogs.imf-formacion.com/blog/tecnologia/tipos-de-datos-datos-estructurados-semiestructurados-y-no-estructurados-202006/#Que_son_los_datos_estructurados (Último acceso 01-06-2023).

[56] PowerData. *Diferencias entre data mart, data lake, data warehouse y data cube.* 2017. URL: https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/diferencias-entre-data-mart-data-lake-data-warehouse-y-data-cube (Último acceso 01-06-2023).

[57] Signaturit. *¿Qué es Business Intelligence (BI) y qué herramientas existen?* 2021. URL: https://blog.signaturit.com/es/que-es-business-intelligence-bi-y-que-herramientas-existen (Último acceso 01-06-2023).

[58] HEAVY.AI. *BI Tools - A Complete Introduction.* URL: https://www.heavy.ai/learn/bi-tools (Último acceso 08-06-2023).

[59] skypoint cloud. *Business Intelligence Tools: A Pros and Cons Comparison Chart.* URL: https://skypointcloud.com/blog/business-intelligence-tools-comparison-chart/ (Último acceso 08-06-2023).

[60] FineReport. *Top 8 BI tools of 2023: Comparison and How to decide.* 2023. URL: https://www.finereport.com/en/bi-tools/top-5-bi-tools-of-2019-comparison-and-how-to-decide.html#PowerBI_In-depth_Review (Último acceso 08-06-2023).

[61] Beservices. *Las fases de un proyecto de Business Intelligence.* 2022. URL: https://blog.beservices.es/blog/las-fases-de-un-proyecto-de-business-intelligence (Último acceso 08-06-2023).

[62] ExistBI. *Phases of Business Intelligence Systems Development.* URL: https://www.existbi.com/articles/phases-of-business-intelligence-systems-development/ (Último acceso 08-06-2023).

[63] Beatriz Canales Inocencio. *Procesos de la inteligencia de negocios.* URL: https://www.gestiopolis.com/procesos-inteligencia-negocios/ (Último acceso 08-06-2023).

[64] felipe anderson. *Business Intelligence Value Chain – The Process of Powerful Business Decision-making.* 2022. URL: https://www.analytixlabs.co.in/blog/business-intelligence-value-chain/#Business_Intelligence_Lifecycle (Último acceso 08-06-2023).

[65] Michelle Heath. «Does EPM/BI Project Really Matter? Waterfall vs. Agile vs. Blended». En: *US-Analytics* (). URL: https://www.us-analytics.com/hyperionblog/does-epm-and-bi-project-methodology-really-matter (Último acceso 29-05-2023).

[66] Talent. *Salario en España 2023.* URL: https://es.talent.com/salary?job (Último acceso 22-01-2023).

[67] Leburo. *Tarifas de Coworking en Valladolid.* 2023. URL: https://leburocowork.es/precio-coworking-valladolid (Último acceso 29-05-2023).

[68] Bruce Harpham. «9 ways you're failing at business intelligence». En: *CIO* (nov. de 2017). URL: https://www.cio.com/article/227988/9-ways-youre-failing-at-business-intelligence.html (Último acceso 23-01-2023).

[69] YellowFin. *Top 10 Business Intelligence risks (and their solutions).* URL: https://www.yellowfinbi.com/blog/top-10-business-intelligence-risks-and-their-solutions-part-one (Último acceso 23-01-2023).

[70] Mark Tossell. *6 Reasons Why BI and Analytics Projects Fail – And How to Avoid It.* Nov. de 2021. URL: https://www.salesforceben.com/6-reasons-why-bi-and-analytics-projects-fail-and-how-to-avoid-it/ (Último acceso 14-02-2023).

[71] Rohan Rao. *Formula 1 World Championship (1950 - 2022) F1 race data from 1950 to 2022*. Updated 2022, July 15. URL: https://www.kaggle.com/datasets/rohanrao/formula-1-world-championship-1950-2020 (Último acceso 26-03-2023).

[72] FIA. *2022 FORMULA ONE SPORTING REGULATIONS*. Feb. de 2022. URL: https://www.fia.com/sites/default/files/formula_1_-_sporting_regulations_-_2022_-_iss_4_-_2022-02-18.pdf (Último acceso 29-04-2023).

[73] David Taylor. *Star Schema vs Snowflake Schema – Difference Between Them*. 2023. URL: https://www.guru99.com/star-snowflake-data-warehousing.html (Último acceso 11-06-2023).

[74] python. *Welcome to Python.org*. URL: https://www.python.org/ (Último acceso 27-05-2023).

[75] Crummy. *Beautiful Soup Documentation*. URL: https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (Último acceso 27-05-2023).

[76] Microsoft. *Visualización de datos | Microsoft Power BI*. URL: https://powerbi.microsoft.com/es-es/ (Último acceso 27-05-2023).

[77] Microsoft. *Agrupación de objetos visuales en informes de Power BI Desktop*. 2023. URL: https://learn.microsoft.com/es-es/power-bi/create-reports/desktop-grouping-visuals (Último acceso 27-05-2023).

[78] Microsoft. *¿Qué es Power BI?* URL: https://powerbi.microsoft.com/es-es/what-is-power-bi/ (Último acceso 27-05-2023).

[79] Manuel Peralta. *Comparativa Power BI Desktop y Servicio Power BI*. 2020. URL: https://www.abd.es/2020/09/comparativa-power-bi-desktop-y-servicio-power-bi/ (Último acceso 27-05-2023).

[80] Microsoft. *2022 Gartner Magic Quadrant for Analytics and Business Intelligence Platforms*. 2022. URL: https://info.microsoft.com/ww-landing-2022-gartner-mq-report-on-bi-and-analytics-platforms.html?lcid=es-es (Último acceso 27-05-2023).

[81] EdwardTufte. *The visual display of quantitative information*. Cheshire, Conn. :Graphics Press, 2001.

[82] cluster. *The complete guide to data visualization*. 2023. URL: https://clusterdesign.io/data-visualization-guide/ (Último acceso 18-05-2023).

[83] Juice Analytics. *A Guide to Creating Dashboards People Love to Use*. 2009. URL: https://static1.squarespace.com/static/52f42657e4b0b3416ff6b831/t/5310292ce4b08d35a87c9426/1393568044420/Guide_to_Dashboard_Design.pdf (Último acceso 18-05-2023).

[84] LINE. *Visual Hierarchy, Gutenberg Diagram, F and Z Pattern*. 2019. URL: https://lineindesign.medium.com/be-a-designer-who-can-also-help-with-writing-copy-2f4ea02a5646 (Último acceso 18-05-2023).

[85] My market research methods. *Types of Charts: Choose the Best Chart to Convey Your Message*. 2020. URL: https://www.mymarketresearchmethods.com/types-of-charts-choose/ (Último acceso 27-05-2023).

[86] Ergast. *API Documentation*. URL: http://ergast.com/mrd/ (Último acceso 27-04-2023).

[87] Kenneth Reitz. *requests 2.31.0*. URL: https://pypi.org/project/requests/ (Último acceso 27-05-2023).

[88] Wes McKinney. *pandas*. URL: https://pandas.pydata.org/ (Último acceso 27-05-2023).

[89] Michael Schmitz y Harinder Sethi. *Chunked Extractors*. URL: https://raw.githubusercontent.com/knowitall/chunkedextractor/master/src/main/resources/edu/knowitall/chunkedextractor/demonyms.csv (Último acceso 27-05-2023).

[90] John D. Hunter. *Matplotlib: Visualization with Python*. URL: https://matplotlib.org/ (Último acceso 27-05-2023).

[91] scikit-learn. *sklearn.preprocessing.MinMaxScaler*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html (Último acceso 29-04-2023).

[92] xViz Enterprise. *Radar/Polar Chart*. URL: https://xviz.com/visuals/radar-polar-chart/ (Último acceso 21-05-2023).

[93] Thomas Hamilton. *BI Testing: Business Intelligence Test Cases*. URL: https://www.guru99.com/business-intelligence-testing-sample-test-cases.html (Último acceso 08-06-2023).

[94] David Taylor. *ETL Testing Tutorial*. 2023. URL: https://www.guru99.com/utlimate-guide-etl-datawarehouse-testing.html (Último acceso 08-06-2023).

[95] TuDashboard. *10 métricas usabilidad esenciales de medir*. 2020. URL: https://tudashboard.com/metricas-usabilidad/ (Último acceso 09-06-2023).