UNIVERSIDAD DE VALLADOLID

APLICACIÓN DE LA TÉCNICA DE K-ANONIMIZACIÓN SOBRE BASES DE DATOS RELACIONALES

ESCUELA DE INGENIERÍA INFORMÁTICA

MENCIÓN TECNOLOGÍAS DE LA INFORMACIÓN

TRABAJO DE FIN DE GRADO

Alumna:
Silvia Olmos Lara

Tutora:
Mercedes Martínez González

Agradecimientos

Me gustaría dedicarle unas palabras a todas las personas que me han ayudado durante estos cuatro años de carrera. A mi tutora Mercedes, por su confianza, dedicación y apoyo durante estos meses. A Amador Aparicio, profesor y a David Sanz, responsable de privacidad de la UVa, por prestar su ayuda en todo lo posible. A todos los profesores que han formado parte de mi formación académica. A mis amigos y compañeros de carrera por amenizarme estos años de estudio. A mi familia, por su cariño y apoyo siempre. A mi pareja Tomás por animarme diariamente.

Resumen

La K-anonimización es una técnica de anonimización que sirve para proteger la privacidad de fuentes de datos previniendo la reidentificación de los individuos a los que corresponden dichos datos. En este trabajo se ha desarrollado una aplicación de escritorio que permite a sus usuarios anonimizar conjuntos de datos utilizando el algoritmo Incógnito, un algoritmo que implementa esta técnica y que promete una gran eficiencia al anonimizar fuentes de datos. La aplicación se apoya en bases de datos relacionales, y se ha desarrollado aplicando la privacidad desde el diseño y teniendo en cuenta los requisitos habituales de usabilidad, ya que está pensada para que sea fácil de utilizar para usuarios no expertos en K-anonimización.

Abstract

K-anonymization is an anonymization technique that protect data privacy preventing the re-identification of the individuals to which such data corresponds. In this project, has been developed a desktop application that allows its users to anonymize data sets using Incognito algorithm, an algorithm that implements this technique and that promises great efficiency when anonymizing data sources. The application is supported by relational databases, and has been developed applying privacy by design and taking into account the usual usability requirements, since it is designed to be easy to use for non-experts in K-anonymization users.

Índice general

Capítulo 1

Introducción

1.1. Contexto

Estamos en una época de revolución de los datos, impulsado tanto por la gran cantidad de datos que circulan en la actualidad, como por los múltiples análisis que se realizan sobre ellos. El análisis de los datos puede ser muy beneficioso para múltiples campos. En las empresas, por ejemplo, se analizan los hábitos de compra y los productos que prefiere el cliente, el rendimiento del empleado, los paquetes óptimos de ventas, la efectividad de los socios y muchos otros factores que afectan a las ventas [7].

Por todas las ventajas que proporciona el análisis de datos se hace muy importante compartir y publicar datos. Sin embargo, cierta información que se comparte contiene aspectos personales con datos sensibles que hay que proteger para mantener la privacidad de los individuos.

La privacidad y en concreto la protección de los datos personales es un derecho de todos amparado por la ley. En el caso de la unión europea entró en vigor el RGPD, Reglamento de Protección de Datos, en mayo del año 2018 [11]. El reglamento recoge todo lo relativo al tratamiento de datos personales y a la libre circulación de los mismos.

Para anonimizar datos existen varias técnicas, algunas de ellas son [24]:

1. **Algoritmos de hash**, es una técnica que aplicada a un dato proporciona una clave única o casi única, y a partir de la cual es imposible reconstruir el dato original.

2. **Algoritmos de cifrado homomórfico**, permiten realizar operaciones sobre datos cifrados de manera que el resultado es el mismo que si se hubiese realizado sobre los datos originales, y los resultados se pueden descifrar si se dispone de la clave adecuada.

3. **Perturbación de datos**, en esta técnica se realiza variaciones o supresiones de los datos.

En este TFG se ha desarrollado la técnica de K-anonimización, la cual pertenece a la tercera categoría, ya que utiliza la técnica de generalización, mediante la cual se realizan variaciones en los datos, y la técnica de supresión, que elimina algunos valores.

1.2. Motivación

Un gran problema respecto a las fuentes de datos anonimizados es que se interconectan con otras con las que pueden compartir atributos comunes. Esto provoca que pueda ser posible interconectar datos de varias fuentes, y aún habiendo eliminado los datos que identifican a las personas, establecer ciertos vínculos que pueden llegar a amenazar la privacidad de los individuos, esto es, reidentificar a los individuos. Actualmente, esto se considera un problema grave y así está recogido en la normativa de protección de datos, tanto a nivel europeo en el Reglamento General de Protección de Datos (RGPD) [11], como en la regulación española que las desarrolla y amplía la Ley Orgánica 3/2018, del 5 de diciembre, de Protección de Datos Personales y Garantía de los Derechos Digitales (LOPDGDD) [4].

La K-anonimización nace como una medida para gestionar el riesgo de reidentificación manteniendo la utilidad de los datos, a partir de una nueva versión de los mismos [23].

Uno de los algoritmos más populares que implementan la técnica de K-anonimización es Incógnito, el cual fue creado por Kristen LeFevre, David J. DeWitt y Raghu Ramakrishnan en el año 2005. Este algoritmo promete hasta un orden de magnitud más rápido que los algoritmos anteriores [17].

Dado que este algoritmo se está utilizando habitualmente y que la propuesta del artículo en el que se explica este algoritmo se aplica a bases de datos relacionales, una materia con la que se trabaja en los estudios del Grado de Ingeniería Informática, se trataba de una buena oportunidad de combinar los conocimientos adquiridos durante la carrera con aspectos más novedosos, como la privacidad desde el diseño.

1.3. Objetivos

El objetivo principal de este trabajo es la obtención de una aplicación que implemente la técnica de K-anonimización a través del algoritmo Incógnito sobre bases de datos relacionales y que sea capaz de transformar datos para que cumplan con un cierto nivel de k-anonimidad. Además, la aplicación debe garantizar la seguridad y la privacidad de los datos manipulados, para lo que se tendrá que seguir el principio de privacidad desde el diseño [22].

Otro objetivo de este proyecto es que la aplicación implementada sea capaz de realizar propuestas de algunos parámetros para facilitar el uso de la herramienta a usuarios que no estén familiarizados con la técnica de K-anonimización. Esto es, ofrecer una aplicación accesible y fácil de usar para usuarios no expertos. Se pretende proponer parámetros como el conjunto de atributos cuasi-identificadores, el valor de K y las jerarquías de generalización de los distintos atributos.

1.4. Organización de la memoria

Este documento se organiza en nueve capítulos de la siguiente manera:

- **Capítulo 1. Introducción**: Una breve presentación del proyecto que se va a realizar.
- **Capítulo 2. Planificación**: Se describe la planificación inicial del proyecto, incluyendo la planificación de las tareas a realizar, el plan de riesgos del proyecto y una estimación del presupuesto.
- **Capítulo 3. Fundamentos teóricos**: Se explican los conceptos teóricos necesarios para la comprensión del proyecto.
- **Capítulo 4. Herramientas que aplican la K-anonimización**: Se hace una investigación de cómo funcionan algunas herramientas que implementan esta técnica para asentar los conceptos teóricos.
- **Capítulo 5. Análisis**: Se realiza el proceso de análisis del proyecto, especificando los requisitos que debe cumplir y los casos de uso que debe llevar a cabo.
- **Capítulo 6. Diseño**: Se detalla el diseño de las diferentes partes de la aplicación.
- **Capítulo 7. Implementación**: Especificación de las tecnologías utilizadas y los métodos para la implementación de la aplicación.
- **Capítulo 8. Validación**: Comprobación de que los resultados dados por la aplicación son correctos.
- **Capítulo 9. Conclusiones y trabajo futuro**: Reflexión sobre el trabajo realizado y el trabajo que se podría realizar en el futuro.
- **Anexo. Guía de usuario**: Un manual de como funciona la aplicación.
- **Anexo. Manual de instalación**: Una guía para la instalación de la aplicación.
- **Anexo. RGPD**: Los artículos detallados del Reglamento General de Protección de Datos mencionados en el documento.

Capítulo 2

Planificación del proyecto

En este capítulo se realiza la planificación del proyecto necesaria antes de comenzar a trabajar en el mismo. Primero se detalla una planificación inicial en la cual se indican las distintas tareas y el tiempo que llevará cada una de ellas. Después, se analizan los distintos riesgos que pueden aparecer a lo largo del trabajo y hacer peligrar la planificación inicial y por tanto el proyecto. Por último, se indica una estimación del presupuesto necesario para llevar a cabo este proyecto.

2.1. Planificación inicial

Para la planificación del proyecto se va a seguir una metodología de cascada, en la cual se desarrolla el proyecto de forma secuencial, dividiéndolo en distintas fases y desarrollándolas en orden. Es decir, cada fase no podrá comenzar hasta que la anterior no haya finalizado completamente. En cada una de las fases entrará no solo el desarrollo de las tareas indicadas, si no también la realización de las secciones de la memoria correspondientes. Las fases que se seguirán en este proyecto serán:

1. **Iniciación**: Durante esta fase se estudiarán los conceptos teóricos necesarios y se realizará la planificación completa necesarios para la realización del proyecto.

2. **Análisis**: En esta fase se realizará el estudio de los requisitos que tiene que debería cumplir el proyecto y se detallarán todos los posibles casos de uso.

3. **Diseño**: Para esta fase se realizarán varios diagramas que detallarán como estará diseñada la aplicación. Se realizará un modelo de dominio, un modelo conceptual y lógico para la base de datos, un diagrama de paquetes, uno de despliegue y algún diagrama de secuencia que se considere necesario para los procedimientos más complejos. También se hará un estudio de los patrones que tendrá que cumplir la aplicación.

4. **Implementación**: Durante esta fase se desarrolla la aplicación. Será la fase más larga ya que es una tarea que consta de varias partes, en primer lugar un período de aprendizaje de la tecnología a utilizar, posteriormente la realización de la interfaz con su funcionalidad, y para finalizar la implementación de los algoritmos.

5. **Pruebas**: Durante esta fase se comprobará que la aplicación funciona correctamente y que los resultados obtenidos son los deseados. Para esto se utilizarán otras herramientas similares validadas por la AEPD (Agencia Española de Protección de Datos) y se comprobarán los

resultados. Al finalizar este período de validación, se realizará unas comprobaciones finales de todas las fases del proyecto.

Las fechas de inicio y finalización del proyecto se detallan a continuación, así como las distintas tareas a realizar con sus respectivas estimaciones de duración (Tabla 2.1). La duración total del proyecto es de 21 semanas, sin embargo la planificación de las tareas se realiza en 20 semanas, dejando así una semana de margen para posibles imprevistos.

- **Fecha de inicio:** 1 de Febrero
- **Fecha de finalización:** 27 de Junio
- **Duración del proyecto:** 21 semanas

El diagrama de Gantt correspondiente a la planificación inicial del proyecto se detalla en la figura 2.1. En este diagrama se puede observar que todas las tareas tienen una relación de dependencia de fin a inicio con sus predecesoras, es decir, cada tarea no podrá comenzar hasta que la anterior haya finalizado.

	Tarea	**Duración**	**Fecha inicio**	**Fecha fin**
1	Lectura y asimilación de conceptos teóricos	1 semana	1/2/21	7/2/21
2	Planificación	1 semana	8/2/21	14/2/21
3	Fundamentos teóricos	1 semana	15/2/21	21/2/21
4	Recolección de requisitos	1 semana	22/2/21	28/2/21
5	Realización de casos de uso	1 semana	1/3/21	7/3/21
6	Diseño	2 semanas	8/3/21	21/3/21
7	Implementación	10 semanas	22/3/21	23/5/21
8	Pruebas	2 semanas	31/5/21	13/6/21
9	Introducción, conclusiones y resumen	1 semana	14/6/21	20/6/21

Tabla 2.1: Planificación del proyecto

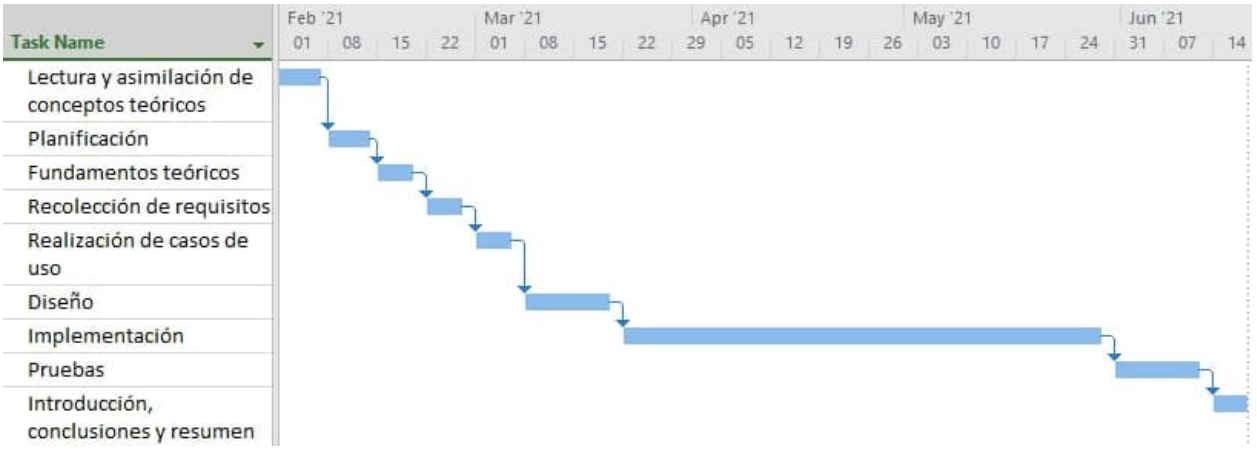

Figura 2.1: Diagrama de Gantt inicial

Como se puede ver en el diagrama de la figura 2.1 la tarea que más tiempo requiere es la de implementación que consta de unas 10 semanas. Además de por los motivos explicados anteriormente, el período de tiempo de esta tarea coincide con la realización de las prácticas externas en empresa por parte del trabajador, por lo que este tendrá menos tiempo disponible para la realización del trabajo.

2.2. Plan de riesgos

En esta sección se realiza un análisis de los distintos riesgos que nos podremos encontrar a la hora de llevar a cabo la planificación. En primer lugar se identificarán los distintos riesgos que podrían surgir. Después, se analizará su exposición utilizando una matriz de riesgos por frecuencia e impacto. Por último, se realizarán los distintos planes de contingencia para aquellos riesgos que tengan una exposición alta.

Para realizar el análisis de los riesgos y calcular su exposición utilizaremos la matriz de riesgos que se muestra en la tabla 2.2, valorando la frecuencia en una escala del 1 al 5, de menor a mayor probabilidad de ocurrencia y el impacto con la misma escala dependiendo del daño que provocaría al proyecto si el riesgo llegara a producirse.

Frecuencia						
	5	Alta	Alta	Muy alta	Muy alta	Muy alta
	4	Media	Media	Alta	Muy alta	Muy alta
	3	Baja	Media	Media	Alta	Muy alta
	2	Baja	Baja	Media	Media	Alta
	1	Baja	Baja	Baja	Media	Alta
		1	2	3	4	5
		Impacto				

Tabla 2.2: Matriz de riesgos

Los riesgos identificados y su análisis se muestran en la tabla 2.3. Una vez realizado este análisis y en base a la exposición de cada riesgo se pueden tomar distintas decisiones. Si la exposición es baja se tiene la opción de simplemente aceptar el riesgo. Si la exposición es más alta, hay dos posibilidades, o realizar un plan de prevención previo a su aparición, evitando el riesgo o reduciéndolo, o un plan de contingencia posterior a su aparición transfiriendo este riesgo a otra persona o mitigándolo buscando una solución para que el riesgo provoque los menores daños posibles. Si se considera necesario se puede llevar a cabo ambos planes para un mismo riesgo. El plan de riesgos se muestra en la tabla 2.4.

Riesgo	**Frecuencia**	**Impacto**	**Exposición**
Ordenador de trabajo estropeado	1	5	Alta
Errores de tiempo de planificación	5	3	Muy alta
Enfermedad del trabajador	2	5	Alta
Modificación de requisitos	2	3	Media
Uso de tecnologías desconocidas	4	2	Media
Encontrar un fallo de diseño tardío	2	5	Alta
Recursos necesarios inaccesibles	2	4	Media
Cambios en la carga de otros trabajos	4	5	Muy alta

Tabla 2.3: Análisis de riesgos

Riesgo	**Plan**
Ordenador de trabajo estropeado	Reducir el riesgo realizando copias de seguridad y mitigar el riesgo utilizando otro ordenador
Errores de tiempo de planificación	Evitar el riesgo introduciendo un margen de tiempo en la planificación
Enfermedad del trabajador	Reducir el riesgo introduciendo un margen de tiempo en la planificación
Modificación de requisitos	Reducir el riesgo dedicando más tiempo al análisis de requisitos
Uso de tecnologías desconocidas	Evitar el riesgo aumentando el tiempo de aprendizaje sobre las tecnologías
Encontrar un fallo de diseño tardío	Reducir el riesgo realizando revisiones junto al tutor frecuentemente
Recursos necesarios inaccesibles	Buscar otros que los sustituyan
Cambios en la carga de otros trabajos	Replanificar las tareas

Tabla 2.4: Plan de riesgos

2.3. Presupuesto

En esta sección se calculará el presupuesto necesario para realizar el proyecto teniendo en cuenta los gastos del personal y los gastos de los productos que se utilizarán.

El gasto de productos será de 0€, ya que el hardware necesario para realizar el proyecto es una computadora normal, sin ninguna característica necesaria, por lo tanto, se utilizará el equipo personal del desarrollador. En cuanto a los gastos de software se utilizará únicamente servicios gratuitos.

El gasto de personal también será de 0€ ya que el desarrollador no cobrará por la realización del proyecto. Sin embargo, podemos estimar el costo que supondría si el desarrollador cobrara por su trabajo. El salario medio de un programador junior en España es de 18.000 € al año [29], en una jornada de 40 horas a la semana, es decir 160 horas al mes, 1920 horas al año, por lo tanto, el sueldo de un programador es de aproximadamente 10€/hora. La estimación de tiempo que llevará el proyecto será de 300 horas. Entonces el gasto de personal y también el gasto total del proyecto sería de 3000€.

Capítulo 3

Fundamentos teóricos

En este capítulo se detallarán los principales conceptos que hay que conocer para entender como funciona la técnica de K-anonimización. Además, también se explicará como funciona el algoritmo Incógnito, uno de los algoritmos de K-anonimización más populares y el que se implementará en este proyecto. [17]

Para la mejor comprensión de esta sección se pondrán ejemplos que partirán de la tabla 3.1. Esta tabla contiene datos personales en los cuales se ha eliminado cualquier atributo que pueda identificar a la persona. Son datos médicos en los que se indica el código postal en el que vive un individuo, la edad que tiene y si tiene problemas de colesterol alto o no. Esta tabla contiene el dato sensible del colesterol el cual querremos proteger para que no se pueda identificar si cierta persona tiene este problema o no a partir de los otros atributos de la tabla.

Código postal	**Edad**	**Colesterol**
37003	40	S
28108	44	S
24700	37	N
24700	37	N
37003	44	S
28108	40	S

Tabla 3.1: Tabla de ejemplo

3.1. La privacidad y la anonimización

Según la RAE la **privacidad** es el "ámbito de la vida privada que se tiene derecho a proteger de cualquier intromisión". Es un derecho fundamental de las personas y por tanto es muy importante mantenerlo en cualquier ámbito. Actualmente, en la era digital, la privacidad es una gran preocupación para las personas, ya que gracias a internet la circulación de datos personales ha aumentado de manera abismal.

Para mantener esta privacidad existen gran cantidad de técnicas de anonimización o seudonimización de los datos.

Según la AEPD la **anonimización** es "el proceso de eliminar o reducir al mínimo los riesgos de reidentificación de los datos pero manteniendo la veracidad de los resultados de su tratamiento", es decir, además de evitar la identificación de las personas por cualquier medio, los datos anonimizados deben garantizar que cualquier operación o tratamiento que pueda ser realizado con posterioridad a la anonimización no conlleva una distorsión de los datos reales [24].

Según el RGPD la **seudonimización** es "el tratamiento de datos personales de manera tal que ya no puedan atribuirse a un interesado sin utilizar información adicional". Normalmente se realiza modificando un atributo en un registro de datos por otro (un seudónimo). Sin embargo, sigue existiendo una alta probabilidad de identificar a la persona de manera indirecta [11].

3.2. Tipos de atributos

En primer lugar, tenemos que distinguir los distintos tipos de atributos que puede tener una base de datos dependiendo de la información que contengan. Pueden ser [23]:

- **Identificadores**: Son los atributos que por si solos identifican al sujeto. P.ej: DNI, teléfono, número de la seguridad social, etc.
- **Cuasi-identificadores**: Son los atributos que si se agrupan con otros cuasi-identificadores pueden identificar a un sujeto. P.ej: edad, código postal, fecha de nacimiento, etc.
- **Atributos sensibles**: Son los atributos cuyos datos podrían tener un gran impacto en la privacidad del sujeto. P.ej: Enfermedades, nivel de renta, tratamientos médicos, etc.

3.3. K-anonimización

La K-anonimización es una técnica que se utiliza para prevenir ataques donde la agregación de datos permita la reidentificación de las personas a partir de los datos recolectados de diferentes fuentes de los cuales se han eliminado los atributos que permiten la identificación directa [14].

Fue propuesta por Latanya Sweeney en el año 2002, con el objetivo de resolver el problema en el que dados unos datos estructurados personales se quiera garantizar de manera científica que en una nueva versión de estos no se pueda reidentificar a las personas, y además, los datos sigan siendo útiles en la práctica [31].

La K-anonimidad es la propiedad de los datos anonimizados que permite cuantificar hasta qué punto se preserva la anonimidad de los sujetos presentes en un conjunto de datos en el que se han eliminado los identificadores. Dicho de otro modo, es una medida del riesgo de que agentes externos puedan obtener información de carácter personal a partir de datos anonimizados [23].

Toda relación tiene un **conjunto de frecuencias**, esto es, la cantidad de registros que toman el valor de cada combinación de atributos cuasi-identificadores. Por ejemplo, a partir de la tabla 3.1, la frecuencia para código postal = 24700, edad = 37, y colesterol = N es 2, ya que hay dos registros que toman estos valores. La frecuencia para código postal = 37003 y edad = 40 es 1.

Cuando decimos que una cierta relación es K-anónima, nos referimos a que cumple con la **propiedad de k-anonimato**, es decir, cada valor en el conjunto de frecuencias de esta relación es mayor o igual que K. La tabla 3.1 es 1-anónima para el conjunto de atributos cuasi-identificadores compuesto por código postal y edad. Un conjunto de datos 1-anónimo significa que algún conjunto de datos de los atributos cuasi-identificadores aparece únicamente en una tupla, por lo tanto, es posible identificar al individuo. Es decir, estos datos no son anónimos [23].

3.4. Métodos de k-anonimización

Para implementar la K-anonimización se pueden utilizar dos métodos que no introducen perturbación en los datos, es decir, no introducen información errónea en los datos originales si no que logran la protección mediante la sustitución de los valores originales por otros más generales. Estos métodos son [23]:

- **Generalización**: consiste en modificar los valores de los atributos cuasi-identificadores para que sean menos precisos, mediante rangos para valores numéricos o mediante jerarquías para valores nominales.

  La generalización puede ser **global** si siempre se realiza de la misma manera la transformación para los distintos registros de un mismo tipo de atributo, o **local** si se utilizan diferentes criterios para cada registro.

- **Supresión**: consiste en eliminar algunos registros de los datos cuyos valores se encuentran en un rango muy distinto al de los valores de otros registros. De esta manera no se distorsionan los resultados al realizar la generalización. Sin embargo, este método se utiliza con poca frecuencia ya que genera una gran pérdida de información y por lo tanto sólamente se aplican a valores que no son relevantes para la finalidad del tratamiento.

  Por ejemplo, si tenemos unos datos con un registro de edades de personas, y la mayoría de ellas se encuentran en un rango entre 30-35 podríamos generalizar los valores con este rango. Sin embargo, si un registro tuviese un valor de 60 no podríamos separar la generalización de los datos en dos rangos como 30-35 y 60-65 ya que para el segundo rango únicamente tenemos un valor y se podría identificar al individuo. Si quisiéramos incluir todos los valores dentro de la misma generalización quedaría un rango de 30-60, lo que crea una pérdida de información bastante elevada. Por lo tanto, en este caso podríamos suprimir el valor de la persona de 60 años y dejar el resto de valores dentro de la generalización 30-35.

En las tablas que se muestran a continuación se pueden ver distintos ejemplos de las técnicas explicadas. En la tabla 3.2 vemos un ejemplo de generalización global sobre los atributos código postal y edad, en los cuales todos sus valores están generalizados. Sin embargo, en la tabla 3.3 se muestra un ejemplo de generalización local sobre el atributo código postal. En este caso algunos valores están generalizados y otros no, ya que si se quiere que los datos cumplan una 2-anonimidad es suficiente con generalizar los datos de esta manera. De esta manera, sin generalizar todos los datos se pierde una menor cantidad de información.

Por último en la tabla 3.4 se muestra un ejemplo de supresión en el cual las dos últimas tuplas serán suprimidas ya que el valor del atributo edad en ambos casos y el valor del código postal en la última tupla tienen valores que se escapan de los rangos que se siguen en resto de tuplas. Por lo tanto, se suprimirán para no perder la precisión.

Código postal	**Edad**	**Colesterol**
37***	40-49	S
28***	40-49	S
24***	30-39	N
24***	30-39	N
37***	40-49	S
28***	40-49	S

Tabla 3.2: Ejemplo de generalización global

Código postal	**Edad**	**Colesterol**
37***	40-49	S
28***	40-49	S
24700	30-39	N
24700	30-39	N
37***	40-49	S
28***	40-49	S

Tabla 3.3: Ejemplo de generalización local

Código postal	**Edad**	**Colesterol**
37003	40	S
28108	44	S
24700	37	N
24700	37	N
37003	44	S
28108	40	S
37891	33	N
50011	13	S

Tabla 3.4: Ejemplo de supresión

3.5. Propiedades de generalización

Antes de explicar el algoritmo que se va a implementar en este proyecto, hay que mencionar tres propiedades que juegan un papel imprescindible en este. Estas son [17]:

- **Propiedad de generalización**: Tenemos dos conjuntos de atributos P y Q de una misma relación, siendo Q una generalización de P. Si la relación es k-anónima con respecto a P, entonces también lo es con respecto a Q.

  Ejemplo: Teniendo los conjuntos de atributos, P y Q (tablas 3.6 y 3.7), de una misma relación (tabla 3.5), siendo Q una generalización de P. Sabemos que la relación es 1-anónima con respecto a P, por lo tanto también será 1-anónima con respecto a Q.

- **Propiedad de resumen**: Tenemos dos conjuntos de atributos P y Q de una misma relación, siendo Q el mismo conjunto que P o una generalización de P. Si tenemos f1, el conjunto de frecuencias de la relación con respecto a P, podemos generar f2, el conjunto de frecuencias de la relación con respecto a Q, sumando f1 con cada conjunto de valores de Q.

  Ejemplo: Teniendo los conjuntos de atributos, P y Q (tablas 3.6 y 3.7), de una misma relación (tabla 3.5), siendo Q una generalización de P. Sabemos que el conjunto de frecuencias de la relación con respecto a P, f1, es [1,1,1,1,1]. Si unimos Q con f1 (tabla 3.8) y sumamos los valores de f1 agrupando por cada conjunto de valores distinto de Sexo y Z1, tenemos que f2, el conjunto de frecuencias de la relación con respecto a Q es [2,1,1,1].

- **Propiedad del subconjunto**: Teniendo P un conjunto de atributos de la relación. Si la relación es K-anónima con respecto a P, entonces también lo es con respecto a cualquier conjunto de atributos T, tal que T sea un subconjunto de P.

  Ejemplo: Teniendo P (tabla 3.6), un conjunto de atributos de la relación. Sabemos que la relación es 1-anónima con respecto a P, por lo tanto también lo será con respecto al conjunto de atributos de [Sexo] y [Código postal].

Sexo	Código postal	Colesterol
M	53715	S
M	53710	N
F	90210	S
M	02174	S
F	02237	N

Tabla 3.5: Ejemplo de relación

Sexo	Código postal
M	53715
M	53710
F	90210
M	02174
F	02237

Tabla 3.6: Conjunto de atributos P

Sexo	Z1
M	5371*
M	5371*
F	9021*
M	0217*
F	0223*

Tabla 3.7: Conjunto de atributos Q

Sexo	Z1	f1
M	5371*	1
M	5371*	1
F	9021*	1
M	0217*	1
F	0223*	1

Tabla 3.8: Unión de Q con f1

3.6. Incógnito

Incógnito es uno de los algoritmos de K-anonimización más populares, y será el algoritmo que se utilice en este proyecto. El objetivo de este algoritmo es la búsqueda de todas las generalizaciones candidatas que cumplen con un cierto nivel de k-anonimidad. Además, promete hasta un orden de magnitud más rápido que otros algoritmos utilizados para el mismo fin, es decir, promete que el tiempo de ejecución crecerá en una menor proporción en relación a otros algoritmos, a medida que aumentan los datos de entrada [10].

El primer paso de este algoritmo es construir las **jerarquías de generalización** para los distintos atributos de la relación. Esta jerarquía se representa en forma de grafo, donde cada nodo representa una de las opciones que se puede utilizar para generalizar los valores del atributo correspondiente, disminuyendo la precisión del valor a medida que avanzamos en el grafo.

Figura 3.1: Jerarquía de generalización para el atributo sexo

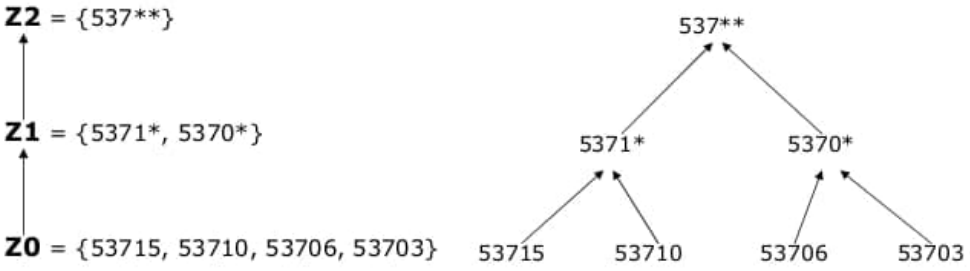

Figura 3.2: Jerarquía de generalización para el atributo Código postal

Las jerarquías de generalización de los distintos atributos de una relación se pueden unir para formar la **red de generalización**. En la figura 3.3 se muestra un ejemplo.

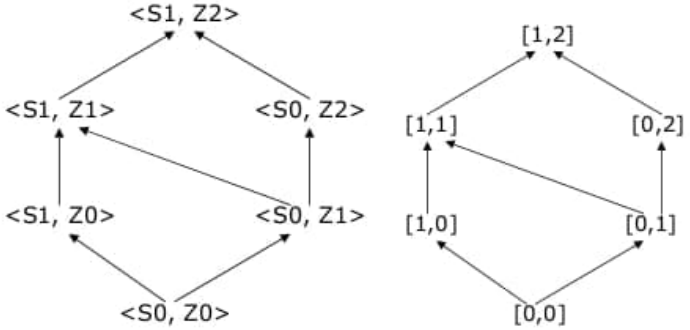

Figura 3.3: Red de generalización para los atributos Sexo y Código postal

El paso más complejo de este algoritmo es el de la búsqueda de las **posibles generalizaciones k-anónimas**. Para ello, se verifica en primer lugar la k-anonimidad para conjuntos de un solo atributo cuasi-identificador, quedándonos con las generalizaciones que cumplan la condición. Después, se va verificando esta k-anonimidad, en conjuntos cada vez más grandes a partir de los conjuntos que ya hemos verificado que son k-anónimos, hasta conseguir todas las generalizaciones k-anónimas posibles.

En cada conjunto de atributos, para buscar los nodos que cumplen la K-anonimidad se realiza una búsqueda primero en amplitud, para la cual se va recorriendo el grafo correspondiente desde el nivel inferior hacia arriba comprobando la K-anonimidad de cada nodo, y cuando nos encontramos con un nodo que cumple esta condición, se marcan todos sus hijos ya que por la propiedad de generalización también cumplen la condición. De esta manera nos ahorramos comprobar la K-anonimidad en todos los nodos.

En la figura 3.4 se muestra un ejemplo de búsqueda de candidatos k-anónimos a partir de los conjuntos k-anónimos [B0], [S0] y [Z0], utilizando el algoritmo de búsqueda primero en amplitud y la propiedad de generalización. Cuando encontramos los candidatos de 2 atributos, comenzamos a buscar a partir de estos los candidatos k-anónimos de 3 atributos. Si realizásemos directamente la búsqueda de los candidatos k-anónimos con 3 atributos tendríamos un grafo a analizar como el que se muestra en la figura 3.6. Sin embargo, al realizar la búsqueda a partir de los conjuntos con 2 atributos que ya sabemos que cumplen la propiedad, el grafo de candidatos se reduce al que se muestra en la figura 3.5.

Una vez que tenemos las distintas posibilidades de generalización, se realiza la búsqueda de la **combinación óptima**. Normalmente la combinación óptima es aquella que cumple con la propiedad de K-anonimización deseada llevando a cabo las mínimas generalizaciones necesarias. Sin embargo, se pueden definir otras nociones de combinación óptima dependiendo de la aplicación en la que se vaya a utilizar la relación anonimizada resultante. Por ejemplo, en alguna aplicación podría ser importante que el atributo sexo no se generalice incluso si esto supone generalizaciones adicionales en el código postal.

Cuando se haya decidido la combinación óptima, se aplica esta generalización a los datos para anonimizarlos [17].

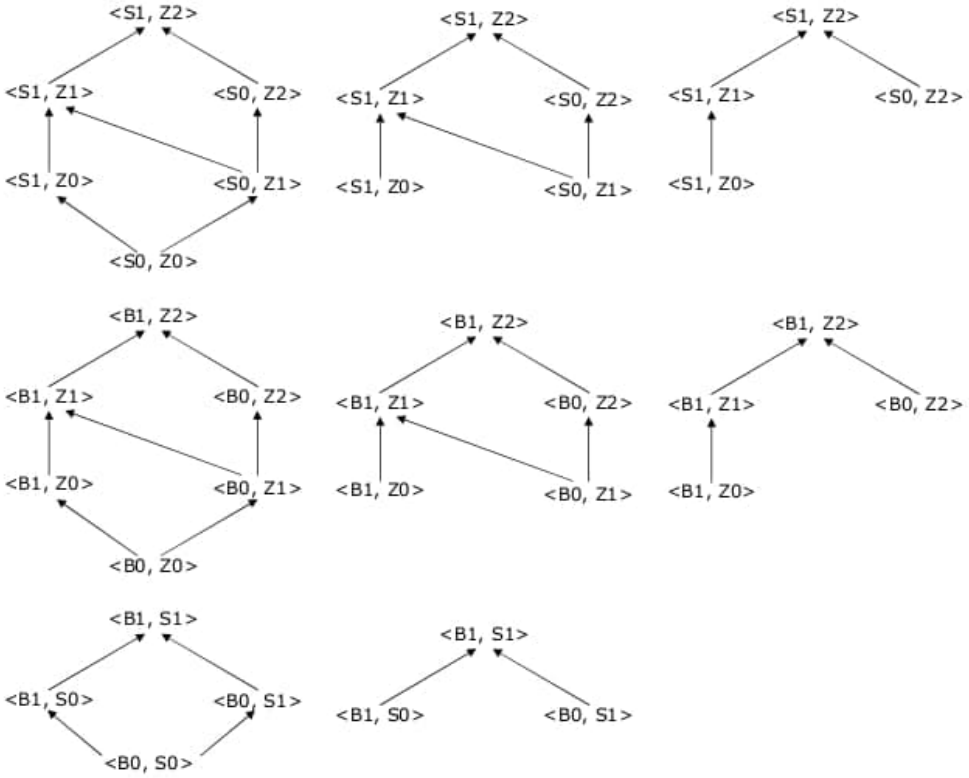

Figura 3.4: Búsqueda de los candidatos k-anónimos de 2 atributos

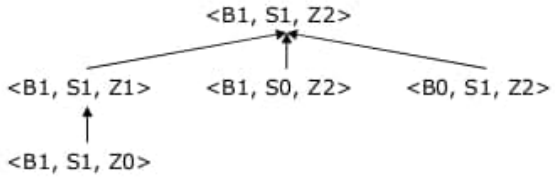

Figura 3.5: Búsqueda de los candidatos k-anónimos de 3 atributos con poda

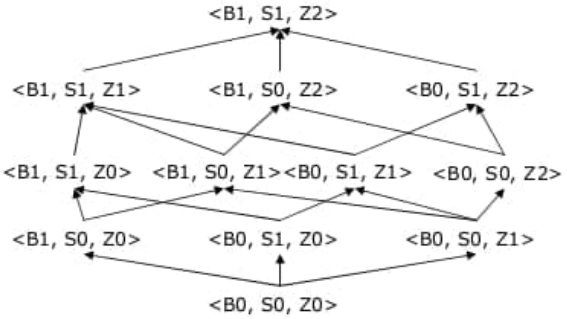

Figura 3.6: Búsqueda de los candidatos k-anónimos de 3 atributos sin poda

3.7. Debilidades de la K-anonimización

Por desgracia, la k-anonimización no es perfecta y puede ser vulnerable a algunos ataques. Estos se indican a continuación [31]:

3.7.1. Ataque de coindidencias sin ordenar

Cuando los datos de una tabla están ordenados se pueden deducir ciertos valores aunque la tabla esté anonimizada.

Por ejemplo, fijándonos en la figura 3.7, tenemos una tabla inicial como PT. Publicamos una versión anonimizada de esta tabla (GT1) la cual utiliza una generalización en el atributo 'Race' y ninguna en el atributo ZIP. Si después se publica otra tabla también anonimizada (GT2) con una generalización aplicada al atributo ZIP pero ninguna al atributo 'Race' y ambas tablas están ordenadas, se podrán deducir todos los valores de la tabla inicial juntando GT1 y GT2.

Este problema se resuelve simplemente ordenando aleatoriamente los datos de las tablas.

Race	ZIP
Asian	02138
Asian	02139
Asian	02141
Asian	02142
Black	02138
Black	02139
Black	02141
Black	02142
White	02138
White	02139
White	02141
White	02142

PT

Race	ZIP
Person	02138
Person	02139
Person	02141
Person	02142
Person	02138
Person	02139
Person	02141
Person	02142
Person	02138
Person	02139
Person	02141
Person	02142

GT1

Race	ZIP
Asian	02130
Asian	02130
Asian	02140
Asian	02140
Black	02130
Black	02130
Black	02140
Black	02140
White	02130
White	02130
White	02140
White	02140

GT2

Figura 3.7: Ejemplo de ataque de coincidencias sin ordenar.

3.7.2. Ataque de publicación complementaria

Por lo general, no todos los atributos de una tabla se incluyen en el conjunto de atributos cuasi-identificadores. Un problema puede ser que se publiquen dos versiones k-anónimas de una misma tabla que tengan en cuenta distintos conjuntos de cuasi-identificadores, por lo que se puedan mezclar ambas tablas y perder el k-anonimato.

Por ejemplo, supongamos que tenemos dos conjuntos de datos como los mostrados en la figura 3.8. Ambos son 2-anónimos y tienen sus datos ordenados aleatoriamente, sin embargo, los conjuntos de atributos cuasi-identificados no son los mismos, por lo que se pueden unir ambas tablas y obtener la tabla mostrada en la figura 3.9 para la cual los conjuntos [white, 1964, male, 02138] y [white, 1965, female, 02139] son únicos y por lo tanto, estos datos pierden el k-anonimato.

Para resolver esta vulnerabilidad hay que tener cuidado y todas las publicaciones de la misma información deben compartir los mismos atributos cuasi-identificadores.

3.7.3. Ataque temporal

Los datos son dinámicos, es decir, varían con el tiempo, las tuplas se agregan, cambian o eliminan constantemente. Como resultado de esto las publicaciones de datos k-anonimizados en el tiempo pueden estar sujetos a ataques.

Por ejemplo, supongamos que inicialmente se publican los datos mostrados en la tabla GT1 de la figura 3.8 y más tarde se añaden nuevas tuplas a los datos y se vuelven a publicar como GT3 con las tuplas añadidas: [black, 1965, male, 02139, headache] y [black, 1965, male, 02139, rash]. En este caso al igual que en el anterior se pueden unir ambas tablas para revelar las tuplas [white, 1964, male, 02138] y [white, 1965, female, 02139].

Una solución a este problema sería publicar siempre los nuevos datos a partir de los datos k-anonimizados publicados inicialmente. De esta manera si se publica la tabla GT1 con las tuplas añadidas [black, 1965, male, 02139, headache] y [black, 1965, male, 02139, rash] no se comprometerá ninguna información.

Race	BirthDate	Gender	ZIP	Problem
black	1965	male	02141	short of breath
black	1965	male	02141	chest pain
person	1965	female	0213*	painful eye
person	1965	female	0213*	wheezing
black	1964	female	02138	obesity
black	1964	female	02138	chest pain
white	1964	male	0213*	short of breath
person	1965	female	0213*	hypertension
white	1964	male	0213*	obesity
white	1964	male	0213*	fever
white	1967	male	02138	vomiting
white	1967	male	02138	back pain

GT1

Race	BirthDate	Gender	ZIP	Problem
black	1965	male	02141	short of breath
black	1965	male	02141	chest pain
black	1965	female	02138	painful eye
black	1965	female	02138	wheezing
black	1964	female	02138	obesity
black	1964	female	02138	chest pain
white	1960-69	male	02138	short of breath
white	1960-69	human	02139	hypertension
white	1960-69	human	02139	obesity
white	1960-69	human	02139	fever
white	1960-69	male	02138	vomiting
white	1960-69	male	02138	back pain

GT3

Figura 3.8: Ejemplo de ataque de publicación complementaria y ataque temporal

Race	BirthDate	Gender	ZIP	Problem
black	1965	male	02141	short of breath
black	1965	male	02141	chest pain
black	1965	female	02138	painful eye
black	1965	female	02138	wheezing
black	1964	female	02138	obesity
black	1964	female	02138	chest pain
white	1964	male	02138	short of breath
white	1965	female	02139	hypertension
white	1964	male	02139	obesity
white	1964	male	02139	fever
white	1967	male	02138	vomiting
white	1967	male	02138	back pain

LT

Figura 3.9: Ejemplo de ataque de publicación complementaria

Capítulo 4

Herramientas que aplican la K-anonimización

Con el fin de asentar los conocimientos teóricos adquiridos, se investigan diferentes herramientas de anonimización que utilizan el método de K-anonimización. También se utilizarán para validar los resultados del algoritmo una vez este implementado. Las herramientas investigadas han sido ARX [3] y la versión online de Amnesia [2].

4.1. ARX

ARX es un software de código abierto para anonimizar datos. Admite una amplia variedad de modelos de privacidad y riesgo, métodos para transformar datos y métodos para analizar la utilidad de los datos de salida. La aplicación cuenta con una interfaz gráfica de usuario multiplataforma, la cual se muestra en la figura 4.1.

ARX permite importar datos en varios formatos: CSV, XLS y XLSX. También permite importar distintas jerarquías de generalización desde ficheros CSV. El formato 'CSV' son documentos para representar datos en forma de tabla, donde las columnas se separan por comas y las filas por saltos de línea. El formato 'XLS' es el formato nativo de Excel, desarrollado por Microsoft para Microsoft Office. Es un formato pensado para almacenar hojas de cálculo. El formato 'XLSX' es una variación del XLS, que se estrenó con Excel 2007 y es el formato que se usa actualmente por defecto.

Una vez importados los datos, se muestran en una tabla en la parte izquierda de la interfaz. A la derecha de la misma se pueden configurar los distintos atributos de la tabla e indicar para cada atributo su tipo (identificador, cuasi-identificador, sensible o no sensible) y su jerarquía de generalización. También a la derecha se indica el tipo de anonimización que queremos aplicar a los datos y con que características. En este caso indicaríamos k-anonimización y el valor de k deseado.

Cuando todos los valores están configurados correctamente, se puede pulsar el botón de anonimizar. Esto muestra el grafo de las posibles generalizaciones que cumplirían con el valor de k indicado, y se muestra en color amarillo cual sería la generalización óptima como se puede ver en la figura 4.2.

Por último se puede seleccionar una generalización concreta con el botón derecho del ratón e indicar 'Apply transformation' para anonimizar los datos con la generalización seleccionada (figura 4.3). Una vez que tenemos estos datos anonimizados, se pueden exportar a un fichero CSV.

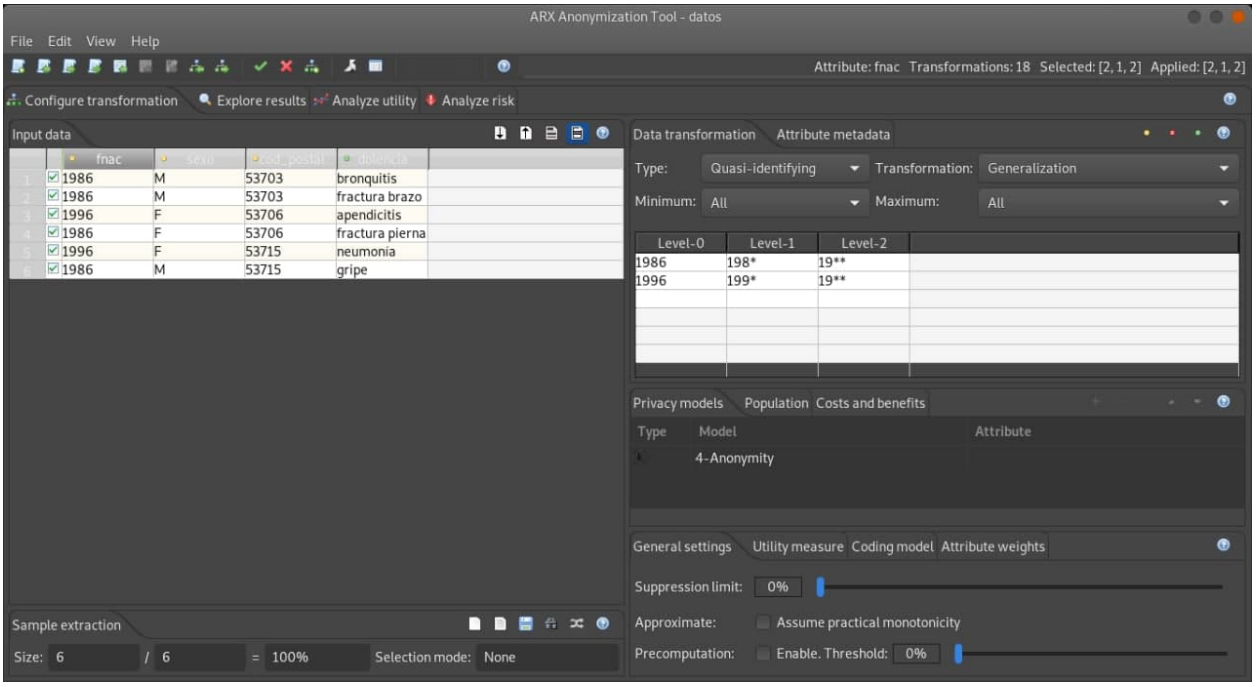

Figura 4.1: Interfaz gráfica de usuario de ARX

Figura 4.2: Grafo de las posibles generalizaciones de ARX

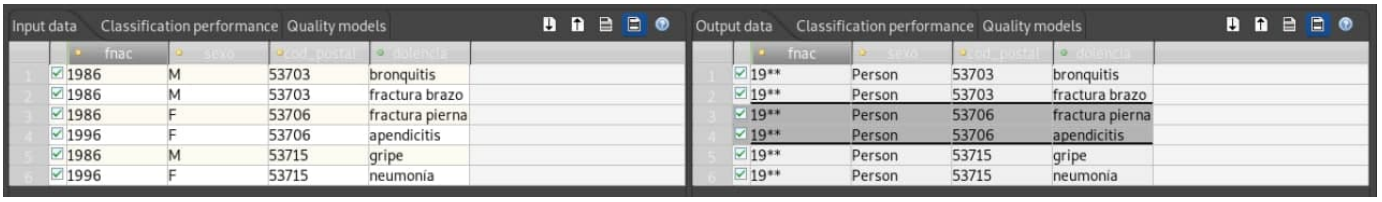

Figura 4.3: Datos anonimizados con ARX

4.2. Amnesia

Amnesia es una herramienta para anonimizar datos personales y confidenciales. Esta herramienta tiene tanto versión de escritorio como versión online. Para este proyecto únicamente se probó la versión online.

Anmesia permite importar datos desde un fichero de texto (figura 4.4). Cuando se selecciona el fichero desde el que se quieren importar los datos la herramienta solicita indicar cual es el delimitador que separa los datos y cuales son los tipos de estos datos (string, int, decimal...). También permite marcar los atributos que queremos mantener para la anonimización y desmarcar aquellos que queremos desechar. Para aplicar el algoritmo de K-anonimización se dejarían sin marcar los atributos identificadores. (figura 4.5)

Para indicar las distintas jerarquías, Amnesia no permite incluir explícitamente cual es la jerarquía que se quiere utilizar, si no que únicamente permite autogenerarlas desde la propia aplicación, solicitando algunos detalles. Para atributos de tipo string permite utilizar una jerarquía basada en máscara, es decir, añadiendo asteriscos a los últimos caracteres de la cadena, o basada en grupo, sustituyendo los diferentes valores por otro string. Para atributos de tipo numérico permite utilizar jerarquías basadas en rango, agrupando los números en rangos de x valores, o sustituyendo ciertos valores por otro número. En la figura 4.6 se puede ver una jerarquía basada en rango generada por Amnesia para datos numéricos.

Una vez indicados todos estos parámetros, la aplicación solicita que se enlacen los atributos con sus jerarquías correspondientes y que se indique el valor de K deseado, lo que genera el grafo de posibles generalizaciones, como se puede ver en la figura 4.7.

Por último, para anonimizar los datos con la generalización deseada solamente hay que seleccionar esta en el grafo y se genera la tabla con los datos anonimizados con la generalización indicada. Los datos anonimizados se puede exportar también a un fichero de texto. En la figura 4.8 se puede ver el resultado que proporciona Amnesia al finalizar el procedimiento de anonimización. En la parte izquierda de la pantalla se muestran los datos iniciales sin anonimizar y en la parte izquierda los datos anonimizados.

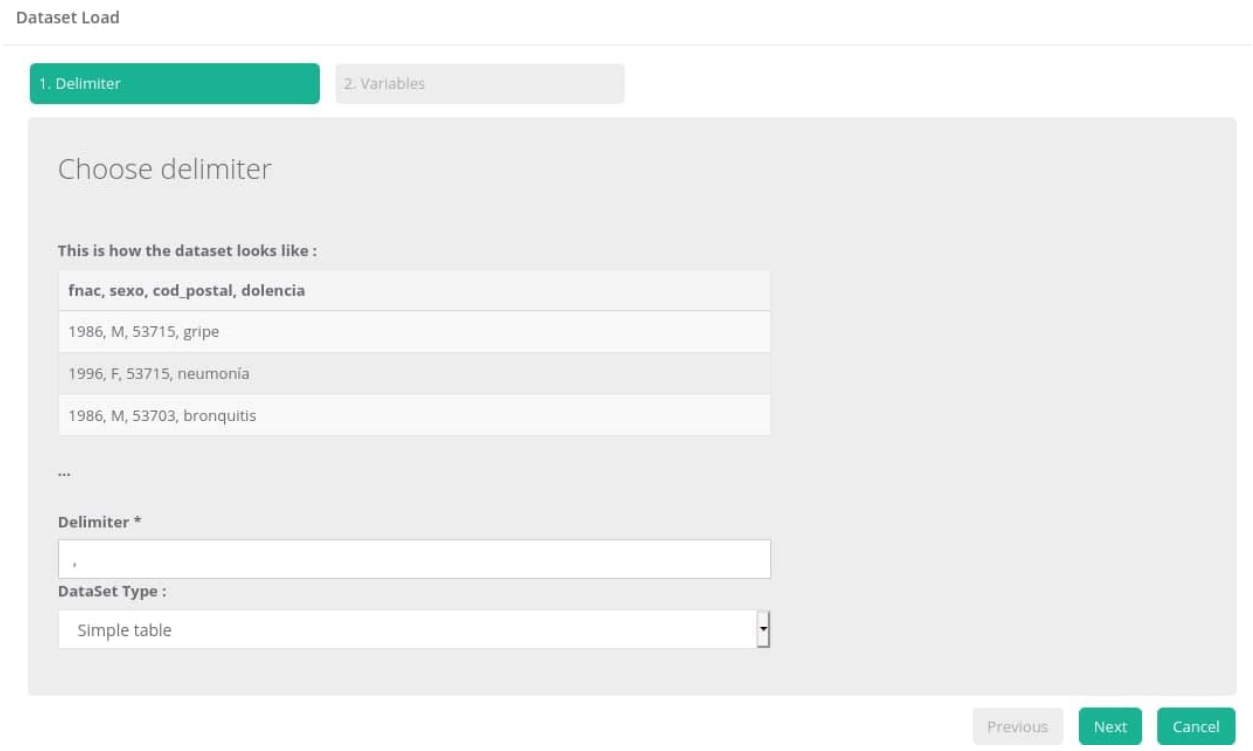

Figura 4.4: Importar datos desde Amnesia

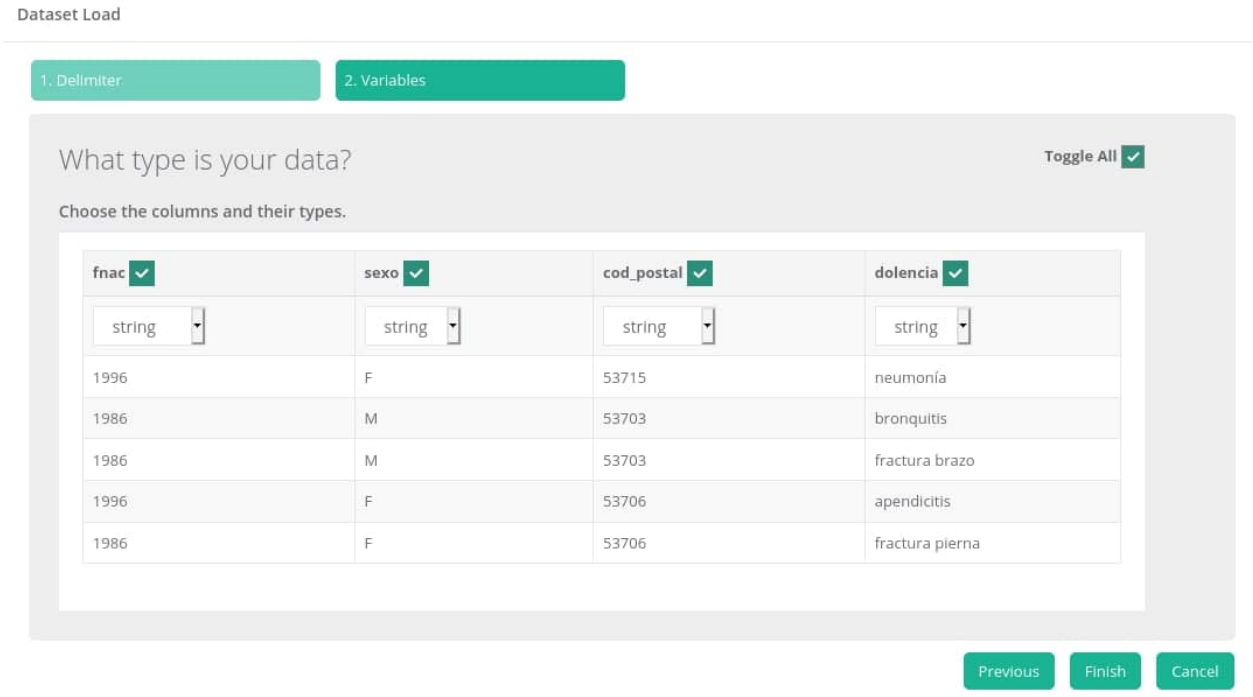

Figura 4.5: Indicar tipo de datos Amnesia

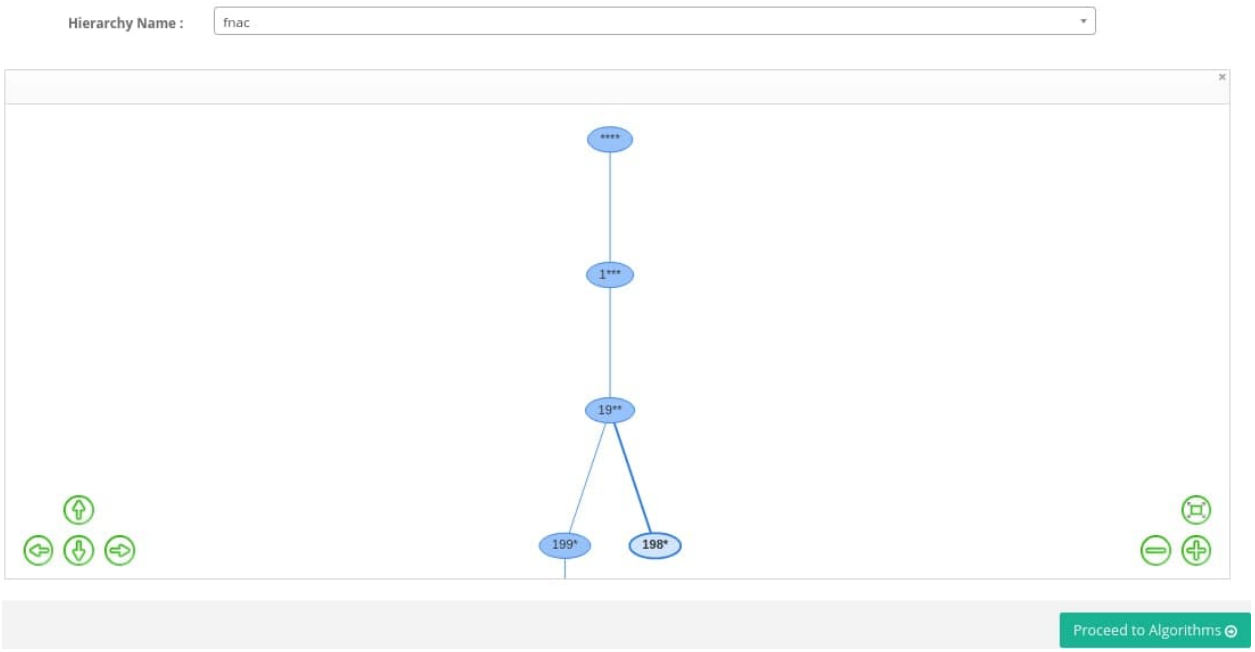

Figura 4.6: Jerarquía de generalización con Amnesia

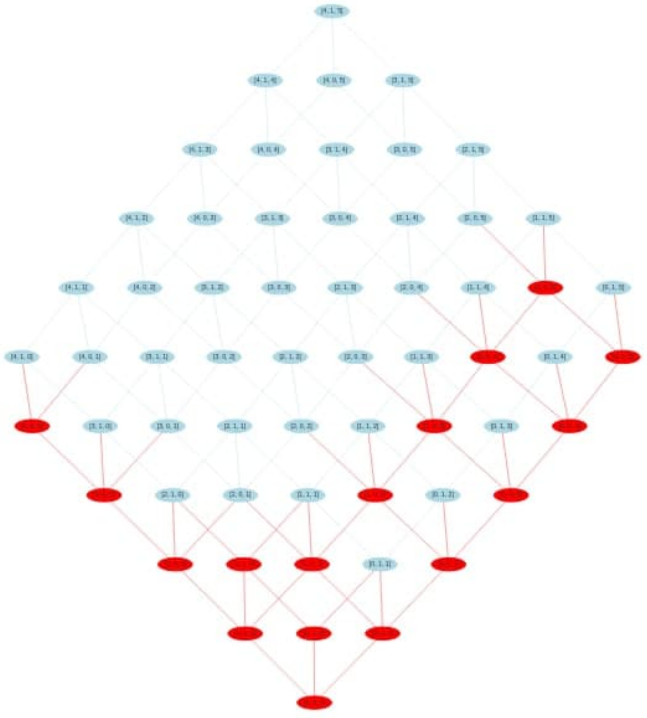

Figura 4.7: Grafo de resultados de Amnesia

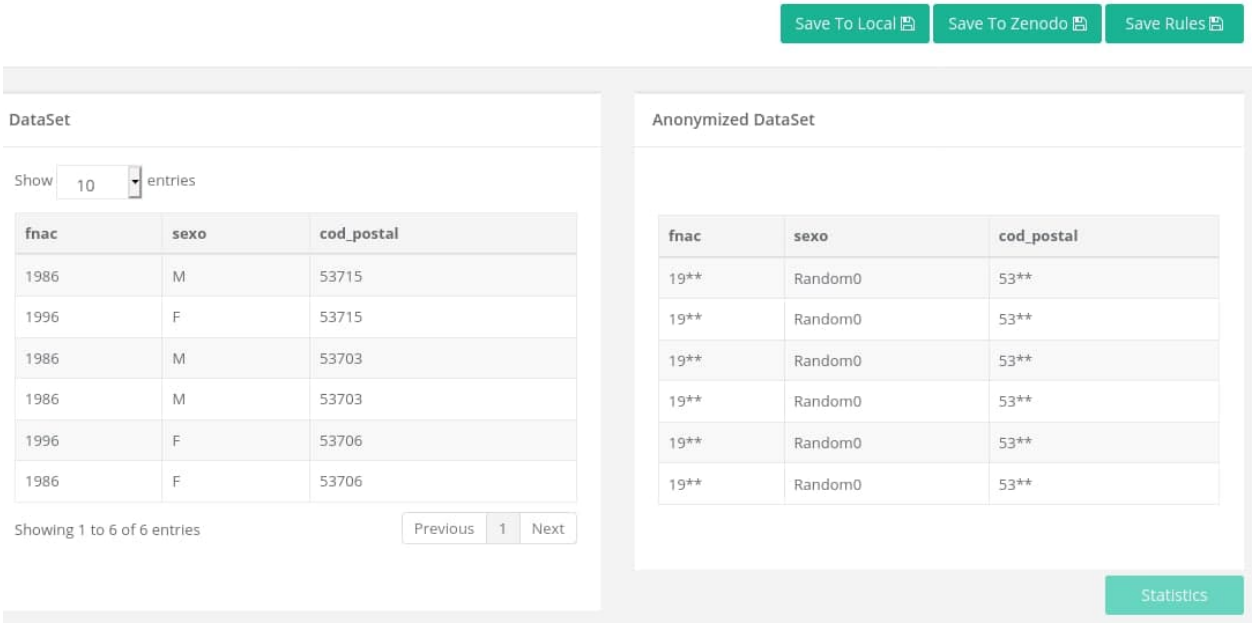

Figura 4.8: Datos anonimizados con Amnesia

Capítulo 5

Análisis

Durante la fase de análisis se especifica detalladamente todas las funcionalidades que debe tener el proyecto. En primer lugar recogen todos los requisitos que se deben cumplir necesariamente. Según la IEEE un requisito es:

"Una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado" [15].

Después, se realizará un estudio de los diferentes casos de uso que el sistema debería llevar a cabo.

5.1. Requisitos funcionales

Los requisitos funcionales son aquellas tareas que el sistema debe ser capaz de realizar. Los servicios que el sistema debe proporcionar, cómo debe reaccionar a una entrada particular y cómo debe comportarse ante situaciones particulares.

- El sistema debe ser capaz de anonimizar los datos de tal manera que no sea posible la reidentificación de un sujeto.
- El sistema debe ser capaz de importar datos de un fichero a una base de datos relacional.
- El sistema debe permitir al usuario indicar los separadores de datos que usan los ficheros a importar.
- El sistema debe permitir que el usuario indique el tipo y la longitud de los datos (int, char, varchar...)
- El sistema debe realizar una propuesta de los tipos de los atributos cuasi-identificadores.
- El sistema debe permitir que el usuario indique los tipos de los atributos. (cuasi-identificadores y identificadores o sensibles)
- El sistema debe realizar una propuesta de jerarquía de generalización para los atributos de tipo entero.
- El sistema debe permitir al usuario indicar la jerarquía de generalización de cada atributo.

- El sistema debe permitir al usuario que indique el grado de k-anonimato deseado.
- El sistema debe permitir que el usuario indique en que atributos quiere aplicar supresión.
- El sistema debe ser capaz de calcular las posibles combinaciones de generalización k-anónimas con un valor de k determinado.
- El sistema debe ser capaz de calcular la combinación de generalización óptima para cada valor de k.
- El sistema debe permitir que el usuario indique la combinación de generalización que desea que se aplique para anonimizar los datos.
- El sistema debe ser capaz de K-anonimizar los datos para cualquier generalización posible.
- El sistema debe permitir exportar los datos anonimizados.
- El sistema será capaz de crear archivos de configuración de las jerarquías de los atributos.
- El sistema será capaz de importar los archivos de configuración de las jerarquías de los atributos.
- El sistema será capaz de crear archivos de configuración para los tipos de los atributos (int, char, varchar, boolean...) y su longitud.
- El sistema será capaz de cargar archivos de configuración para los tipos de los atributos (int, char, varchar, boolean...) y su longitud.
- El sistema deberá tener una interfaz fácil de usar para usuarios no expertos en K-anonimización.

5.2. Requisitos no funcionales

Los requisitos no funcionales describen restricciones que afectan a los servicios o funciones del sistema. Es decir, describen cómo debe realizar el sistema las diferentes tareas.

- El sistema deberá soportar la codificación UTF-8.
- El sistema deberá usar bases de datos relacionales.
- El sistema deberá importar ficheros en formato .txt y .csv
- El sistema deberá exportar los datos anonimizados a ficheros en formato .txt y .csv
- El sistema debe permitir indicar al usuario el tipo (int, char,varchar...) y longitud de los datos tanto manualmente como utilizando un fichero de configuración.
- El sistema debe permitir al usuario indicar la jerarquía de generalización de cada atributo tanto manualmente como utilizando un fichero de configuración.

5.3. Requisitos de información

Los requisitos de información indican el tipo de información que debe guardar el sistema.

- El sistema deberá borrar todos los datos al finalizar la anonimización.
- El sistema almacenará durante el proceso de anonimización información del conjunto de datos sin anonimizar.
- El sistema almacenará durante el proceso de anonimización información de la jerarquía de generalización asociada a cada atributo.
- El sistema almacenará durante el proceso de anonimización los tipos de los atributos del conjunto de datos (sensibles, identificadores o no sensibles ni identificadores).
- El sistema almacenará durante el proceso de anonimización los atributos cuasi-identificadores.

5.4. Requisitos de seguridad y privacidad

- El sistema no guardará más datos de los necesarios.
- El sistema borrará todos los datos cuando el usuario termine el proceso de anonimización.
- El sistema comprobará que los datos realmente han sido eliminados tras borrarlos.
- El sistema realizará el borrado de los datos deberá ser una operación atómica.
- El tratamiento deberá estar registrado en el Registro de Actividades de Tratamiento de Datos Personales previsto en el artículo 30 del RGPD.

5.5. Casos de uso

En esta sección se indicará en primer lugar el diagrama de casos de uso del sistema propuesto. En este diagrama se representan las acciones que podrá realizar el usuario en el sistema, las cuales ya han sido indicadas en el apartado de requisitos funcionales.

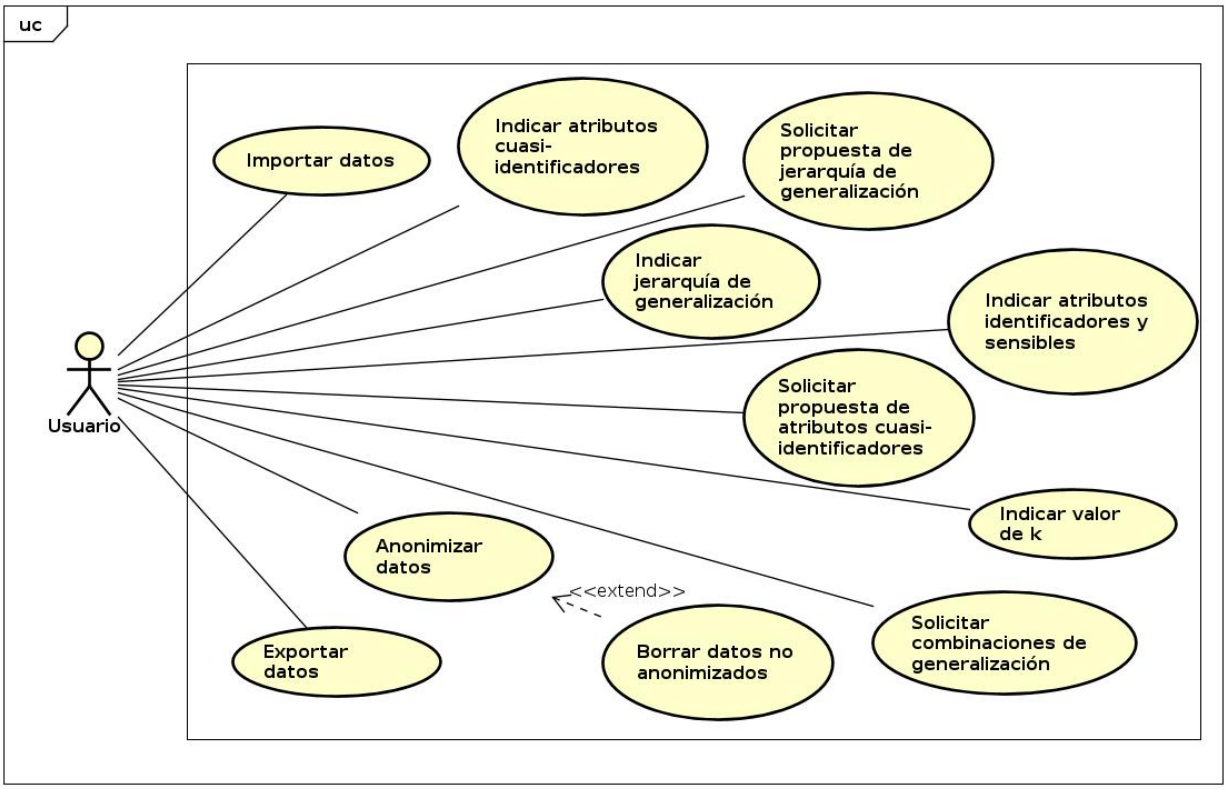

Figura 5.1: Diagrama de casos de uso

A continuación se detallan los distintos casos de uso, describiendo la secuencia de actividades que el sistema, con la intervención del usuario, debe realizar.

UC-01	Importar datos
Descripción	El usuario selecciona un fichero para importar sus datos.
Precondición	El usuario debe disponer de un fichero .txt o .csv con los datos que quiere anonimizar.
Secuencia normal	1. El usuario indica que quiere importar un fichero de datos. 2. El sistema le solicita al usuario que seleccione el fichero. 3. El usuario indica el fichero deseado. 4. El sistema solicita a usuario cual es el separador de los datos. 5. El usuario indica el separador. 6. El sistema solicita al usuario que elija si quiere indicar los tipos y tamaño de los atributos mediante un fichero de configuración o manualmente. 7. El usuario indica que quiere indicarlo manualmente. 8. El sistema muestra los nombres de los atributos. 9. El sistema solicita al usuario que indique los tipos y el tamaño máximo de cada atributo de los datos a importar. 10. El usuario indica el tipo y el tamaño máximo de cada atributo. 11. El usuario indica que quiere importar los datos con los parámetros indicados 12. El sistema guarda los datos en la base de datos. 13. El sistema muestra una tabla con los datos importados.
Flujos alternativos	7a. El usuario indica que quiere indicar los tipos y tamaño máximo de los atributos mediante un fichero de configuración. 7b. El sistema solicita al usuario que indique la ruta al fichero de configuración. 7c. El usuario indica la ruta al fichero de configuración y el caso de uso continúa en el paso 9.
	10a. El usuario no esta conforme con algún parámetro indicado y lo cambia. El caso de uso continua en el paso 11.
Postcondición	Los datos del fichero se han importad a la base de datos y se muestra la tabla al usuario.

Tabla 5.1: Caso de uso: Importar fichero de datos

UC-02	Indicar atributos identificadores y sensibles
Descripción	Se indican los atributos identificadores y sensibles
Precondición	Los datos a anonimizar están importados
Secuencia normal	1. El sistema muestra los atributos de los datos importados. 2. El sistema solicita al usuario que marque los atributos que sean identificadores y los sensibles. 3. El usuario marca los atributos que identificadores y los sensibles. 4. El sistema actualiza los tipos de los atributos marcados. 5. El sistema actualiza el conjunto de posibles atributos cuasi-identificadores.
Postcondición	El sistema tiene guardados los atributos que son sensibles y los identificadores.

Tabla 5.2: Caso de uso: Indicar atributos sensibles o identificadores

UC-03	Solicitar propuesta de atributos cuasi-identificadores.
Descripción	El sistema muestra una propuesta de los atributos cuasi-identificadores más óptimos.
Precondición	Los datos a anonimizar están importados y se han marcado los atributos identificadores y los sensibles
Secuencia normal	1. El usuario solicita una propuesta de atributos cuasi-identificadores. 2. El sistema calcula los atributos cuasi-identificadores óptimos para el conjunto de datos. 3. El sistema muestra los atributos cuasi-identificadores óptimos. 4. El usuario acepta la propuesta de atributos cuasi-identificadores. 5. El sistema guarda los atibutos cuasi-identificadores.
Flujos alternativos	3a. El usuario cancela la propuesta de atributos cuasi-identificadores y el caso de uso finaliza
Postcondición	El sistema ha calculado los atributos cuasi-identificadores óptimos y se los muestra al usuario.

Tabla 5.3: Caso de uso: Solicitar propuesta de atributos cuasi-identificadores

UC-04	Indicar atributos cuasi-identificadores
Descripción	Se indican los atributos cuasi-identificadores, los cuales sufrirán las modificación para anonimizar los datos.
Precondición	Los datos a anonimizar están importados y se han marcado los atributos identificadores o sensibles.
Secuencia normal	1. El sistema muestra los atributos que no son identificadores ni sensibles. 2. El sistema solicita al usuario que marque los atributos que desee que sean cuasi-identificadores. 3. El usuario marca los atributos que quiera como cuasi-identificadores. 4. El sistema guarda los atributos cuasi-identificadores.
Postcondición	El sistema tiene guardados los atributos que son cuasi-identificadores.

Tabla 5.4: Caso de uso: Indicar atributos cuasi-identificadores

UC-05	Solicitar propuesta de la jerarquía de generalización de un atributo entero.
Descripción	El sistema muestra una propuesta de jerarquía de generalización para el atributo indicado por el usuario.
Precondición	Los datos están importados y han sido indicados los atributos cuasi-identificadores.
Secuencia normal	1. El usuario selecciona añadir una jerarquía de generalización. 2. El sistema solicita al usuario que indique el atributo al que le quiere añadir la jerarquía de generalización. 3. El usuario indica el atributo. 4. El sistema muestra la jerarquía de generalización actual del atributo. 5. El usuario solicita una propuesta de jerarquía de generalización para ese atributo. 6. El sistema muestra una propuesta de jerarquía de generalización para el atributo. 7. El usuario acepta la jerarquía propuesta. 8. El sistema actualiza la nueva jerarquía de generalización para el atributo.
Flujos alternativos	6a. El usuario rechaza la jerarquía propuesta y el caso de uso finaliza.
Postcondición	El sistema ha calculado una propuesta de jerarquía de generalización para un atributo y se lo ha mostrado al usuario.

Tabla 5.5: Caso de uso: Solicitar propuesta de la jerarquía de generalización de un atributo

UC-06	Indicar la jerarquía de generalización de un atributo
Descripción	El usuario indica la propuesta que quiere utilizar para un cierto atributo.
Precondición	Los datos están importados y han sido indicados los atributos cuasi-identificadores.
Secuencia normal	1. El usuario solicita incluir una jerarquía de generalización. 2. El sistema solicita al usuario el atributo al que quiere asignarle la jerarquía. 3. El usuario indica el atributo. 4. El sistema solicita al usuario que indique la jerarquía. 5. El usuario indica la jerarquía. 6. El usuario confirma la jerarquía. 7. El sistema guarda la jerarquía.
Flujos alternativos	4a. El usuario indica que quiere importar la jerarquía desde un fichero de configuración. 4b. El sistema solicita al usuario que indique el fichero que contiene la jerarquía de generalización deseada para el atributo indicado. 4c. El usuario indica el fichero que contiene la jerarquía de generalización deseada para el atributo indicado. 4d. El sistema muestra la jerarquía de generalización escogida y el caso de uso continua en el paso 6.
	7a. El usuario indica que quiere exportar la jerarquía a un fichero de configuración. 7b. El sistema solicita al usuario que indique la ruta y el nombre del fichero en el que quiere guardar la jerarquía. 7c. El usuario indica la ruta y el nombre del fichero.
	5a. El usuario indica que quiere añadir supresión al atributo 5b. El sistema añade un último nivel en la jerarquía compuesto por asteriscos y el caso de uso continua en el paso 6.
Postcondición	La jerarquía de generalización está guardada.

Tabla 5.6: Caso de uso: Incluir jerarquía de generalización

UC-07	Solicitar propuesta de valor de k
Descripción	El sistema muestra una propuesta del valor de k óptimo para los datos a anonimizar.
Precondición	Los datos están importados y los tipos de los atributos indicados.
Secuencia normal	1. El usuario solicita una propuesta del valor de k. 2. El sistema calcula el valor de k óptimo para el conjunto de datos y los cuasi-identificadores indicados. 3. El sistema muestra el valor de k óptimo. 4. El usuario acepta el valor de k. 5. El sistema marca el valor de k elegido.
Flujos alternativos	3a. El usuario rechaza el valor de k y el caso de uso finaliza.
Postcondición	El sistema ha calculado el valor de k óptimo y se lo muestra al usuario.

Tabla 5.7: Caso de uso: Solicitar propuesta de valor de k

UC-08	Solicitar posibles generalizaciones candidatas k-anónimas.
Descripción	Se calculan y se muestran las posibles generalizaciones candidatas k-anónimas
Precondición	Los datos están importado, los tipos de los atributos indicados y las jerarquías de generalización de cada atributo están guardadas
Secuencia normal	1. El usuario solicita las posibles combinaciones de generalización k-anónimas. 2. El sistema solicita un valor de k. 3. El usuario indica el valor de k deseado. 4. El sistema calcula las generalizaciones candidatas k-anónimas posibles. 5. El sistema muestra las generalizaciones candidatas k-anónimas y la generalización óptima.
Postcondición	El sistema muestra las generalizaciones candidatas k-anónimas y la generalización óptima

Tabla 5.8: Caso de uso: Solicitar posibles combinaciones de generalización k-anónimas

UC-09	Anonimizar datos
Descripción	El usuario solicita anonimizar los datos mediante el método de k-anonimización
Precondición	Las generalizaciones candidatas k-anónimas están calculadas y se muestran en pantalla.
Secuencia normal	1. El usuario solicita anonimizar los datos. 2. El sistema solicita al usuario que indique que generalización candidata desea usar. 3. El usuario indica la generalización candidata que quiere usar. 4. El sistema anonimiza los datos. 5. El sistema muestra los datos anonimizados con los atributos identificadores eliminados.
Postcondición	Se muestran los datos anonimizados

Tabla 5.9: Caso de uso: Anonimizar datos

UC-10	Borrar datos no anonimizados
Descripción	Cuando el usuario finaliza el procedimiento con los datos no anonimizados estos se borran de la base de datos.
Precondición	La aplicación tiene un conjunto de datos cargado.
Secuencia normal	1. El usuario indica que desea finalizar el procedimiento de anonimización de datos. 2. El sistema informa que se borrarán todos los datos cargados. 3. El usuario acepta la condición. 4. El sistema borra todos los datos de la base de datos.
Postcondición	La base de datos está vacía.

Tabla 5.10: Caso de uso: Borrar datos no anonimizados

UC-11	Exportar datos
Descripción	El usuario solicita exportar los datos con los que está trabajando.
Precondición	El sistema ha anonimizado los datos con los parámetros deseados.
Secuencia normal	1. El usuario solicita extraer los datos que han sido anonimizados. 2. El sistema solicita que indique el directorio en el que quiere guardar el fichero y el nombre del mismo. 3. El usuario indica el directorio y el nombre del archivo. 4. El sistema exporta los datos al fichero indicado.
Postcondición	Se exportan los datos a un fichero con el nombre indicado y en el directorio indicado.

Tabla 5.11: Caso de uso: Exportar datos

Capítulo 6

Diseño

En esta sección se detallará el estudio realizado para el diseño del proyecto. Se incluirán diferentes diagramas para explicarlo. Un modelo de dominio para expresar de manera visual los diferentes conceptos (entidades) que van a formar parte del proyecto. Los modelo conceptual y lógico de la base de datos, donde se detallará claramente las diferentes tablas que se van a crear y las relaciones existentes entre estas. Un diagrama para indicar los paquetes que formarán parte de la aplicación. Un diagrama de secuencia para detallar más profundamente el proceso de anonimización de los datos. Un diagrama de despliegue para explicar los distintos componentes que forman parte de la aplicación y como se relacionan entre ellos. Y por último una explicación de los distintos patrones que se van a utilizar para la realización del proyecto.

6.1. Modelo de dominio

En el modelo de dominio mostrado en la figura 6.1 tenemos una clase 'Datos originales'. Cada instancia de esta clase corresponde a cada una de las filas de los datos importados por el usuario. Está compuesta por distintos 'Atributos', los cuales tienen un nombre, un valor y un tipo. El nombre corresponde al nombre de la columna . Cada uno de estos atributos tienen una jerarquía de generalización, que es única para cada atributo.

Para todos los datos que ha importado el usuario, hay varias generalizaciones candidatas, las cuales, cada una de ellas cumple con un valor de k-anonimato. Para cada valor de k existe una generalización óptima, que a su vez es una generalización candidata.

Cada instancia de 'Datos anonimizados' corresponde con una instancia de 'Datos originales'. 'Datos anonimizados' contiene los atributos cuasi-identificadores generalizados y los atributos sensibles. Los atributos contenidos en la misma instancia de 'Datos anonimizados' son anonimizados mediante una generalización candidata concreta.

Tanto los datos originales como los datos anonimizados están compuestos por atributos.

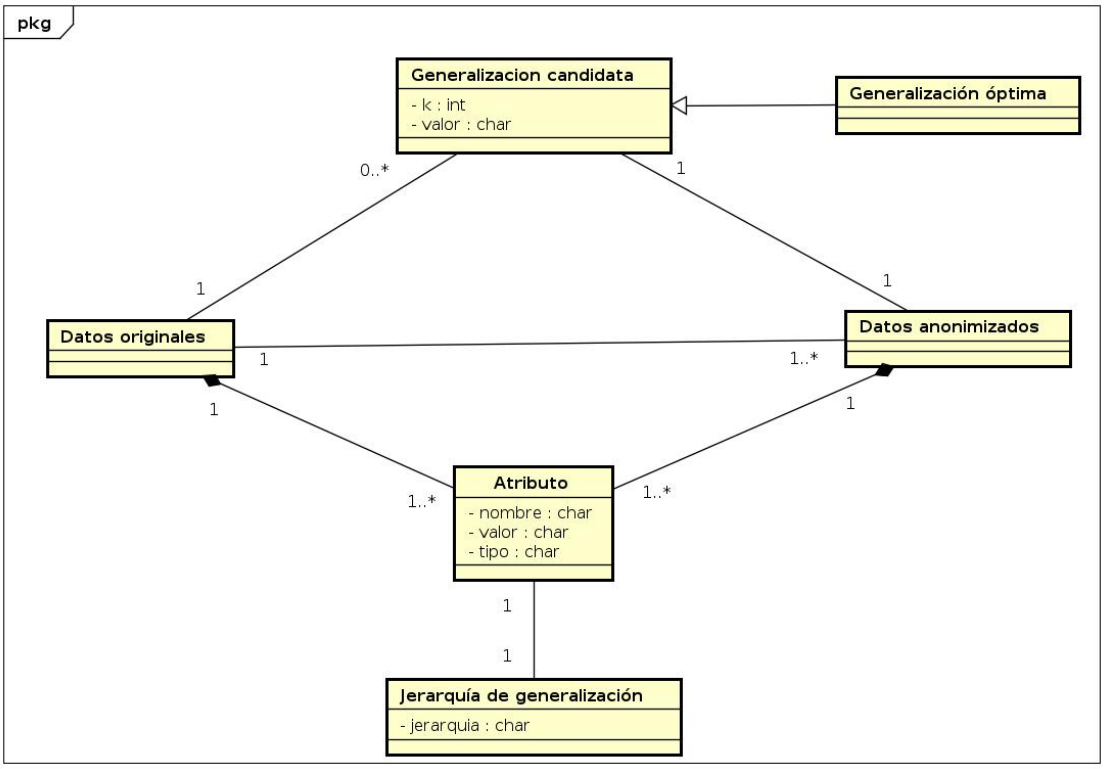

Figura 6.1: Modelo de dominio

6.2. Metamodelo de la base de datos

En este caso en vez de realizar un modelo de la base de datos, se va a necesitar un metamodelo el cual debe cubrir cualquier caso, ya que se necesita guarda la información sobre el esquema de los datos además de los propios datos.

La base de datos que se va a tener va a contener distintas tablas que se irán creando cada vez que se importen unos datos nuevos, durante la configuración de los diferentes parámetros y mientras se procese el algoritmo. Estas tablas son necesarias únicamente para el funcionamiento del algoritmo, por lo que una vez este termine y el usuario importe otros datos o cierre la aplicación se borrarán todas las tablas de la base de datos. De esta manera nos aseguramos de no almacenar de manera persistente ningún tipo de información sensible del usuario.

Se necesitarán distintas tablas que se crearán dinámicamente dependiendo del número de atributos que contengan los datos que introduzca el usuario y cuantos de estos se marquen como cuasi-identificadores. Por lo tanto, en está sección se describirán las tablas genéricamente, describiendo un esquema general, pero teniendo en cuenta que este esquema variará dependiendo de los datos iniciales.

6.2.1. Metamodelo conceptual

Para realizar el metamodelo conceptual de la base de datos en primer lugar se identifican los tipos de entidades, es decir, los datos que hay que almacenar. Estos son:

- **DATOS(atributo1, atributo2, atributo3...)**: Son los datos iniciales que el usuario quiere anonimizar, tiene N atributos dependiendo del fichero de datos que se importe. Los nombres de los atributos se modificarán dependiendo de los datos especificados por el usuario.
- **ATRIBUTOS(atributo, tipo**): Son los atributos de los datos, estos pueden ser identificadores, cuasi-identificadores, sensibles. Si los atributos no son ninguno de los tipos anteriores se marcarán como NSoI (no sensible o identificador), que se tomará como el tipo de atributos por defecto.
- **JERARQUÍA(nivel0, nivel1...)**: Guarda las jerarquías de generalización de los atributos. Habrá tantas tablas de jerarquías como atributos tengan los datos a anonimizar y cada una de ellas contendrá tantos niveles como posibles generalizaciones tenga el atributo. Llamaremos a cada una de las tablas Jerarquia_atributo siendo atributo el nombre del atributo del que se almacena su jerarquía.
- **NODO(ID, dim1, index1, dim2, index2...)**: Esta tabla representa los nodos de un grafo formado por las posibles generalizaciones que se pueden aplicar a los datos. El atributo 'ID' es un identificador único para cada nodo. 'Dim_i' representa el nombre del atributo al que se refiere '$index_i$', el cual almacena el nivel de generalización del atributo.
  Existirán tantas tablas nodo como atributos cuasi-identificadores tengan los datos, ya que se comenzará creando un grafo con 1 único atributo y se continuarán añadiendo de manera recursiva otros grafos con un atributo más cada vez, hasta tener el grafo con tantos atributos como cuasi-identificadores tengan los datos. A cada una de estas tablas las llamaremos Nodo_x siendo x el número de atributos que contiene ese nodo.

En segundo lugar se analizarían las relaciones que tienen estas entidades entre si. Las relaciones son:

- **Contiene (Datos, Atributos)**: Los datos están compuestos por atributos.
- **Pertenece (Jerarquía, Atributos)**: Una jerarquía pertenece a un atributo.
- **Enlace (Nodo, Nodo)**: Los nodos se enlazan entre ellos formando un grafo.

En siguiente lugar estudiamos las cardinalidades que tienen estas relaciones.

- card_min(Datos, Contiene) = 1
- card_max(Datos, Contiene) = N

Unos datos contienen 1 o más atributos. Mínimo siempre tiene que tener un atributo, si no no habría nada que anonimizar.

- card_min(Atributos, Contiene) = 1
- card_max(Atributos, Contiene) = 1

Un atributo está contenido en unos únicos datos que son los datos originales sin anonimizar, ya que son los únicos que se guardan.

- card_min(Jerarquía, Pertenece) = 1
- card_max(Jerarquía, Pertenece) = 1

Una jerarquía pertenece a un único atributo.

- card_min(Atributos, Pertenece) = 1
- card_max(Atributos, Pertenece) = 1

Un atributo tiene una única jerarquía de generalización.

- card_min(Nodo, Enlace) = 1
- card_max(Nodo, Enlace) = N

Un nodo se enlaza con varios nodos. Mínimo siempre tendrá un enlace ya que todo nodo siempre tendrá alguna generalización o será generalización de otro. Si no, no habría generalizaciones y no se podrían anonimizar los datos.

El modelo conceptual queda representado en la figura 6.2.

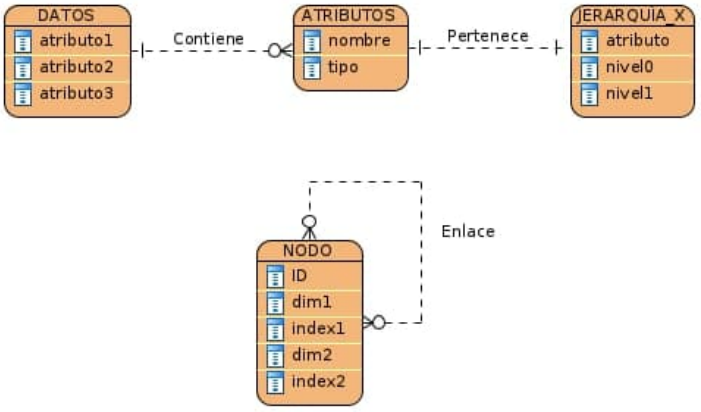

Figura 6.2: Modelo conceptual de la base de datos

6.2.2. Metamodelo lógico

Para pasar del modelo conceptual al modelo lógico transformamos las entidades indicadas en el primero creando una tabla por cada entidad. Estas quedarían de la siguiente forma:

- **DATOS(atributo1, atributo2, atributo3...)**
- **ATRIBUTOS(atributo, tipo)**
- **JERARQUÍA(nivel0, nivel1...)**
- **NODO(ID, dim1, index1, dim2, index2...)**

Después procesamos las relaciones entre tablas indicadas. Para las relaciones uno a varios se absorbe el identificador hacia la tabla con cardinalidad máxima uno.

- Para la relación Contiene, tendríamos que absorber el nombre del atributo hacia la tabla Datos, sin embargo, el nombre de los atributos será sustituido en esta tabla por 'atributo1','atributo2'... Por lo tanto, no es necesario añadir esto.

Para las relaciones varios a varios generamos una tabla con el nombre de la relación y los identificadores de las tablas que relacionan.

- **ENLACE (start, end)**: Siendo 'start' el id del nodo desde el que parte el enlace y 'end' el id del nodo al que llega. Es decir, 'end' es el id del nodo que representa una generalización del nodo cuyo id es 'start'.

Las relaciones uno a uno las transformaríamos incluyendo todo en una misma tabla la cual quedaría de la siguiente manera:

- **ATRIBUTO(nombre, tipo, nivel0, nivel1...)**

Sin embargo, no todos los atributos tienen los mismos niveles de jerarquía, por lo tanto al no tener la misma estructura de la tabla no podremos incluir todos en la misma y habría que crear varias tablas. Para mayor facilidad para la aplicación y ya que los tipos de los atributos se guardarán en momentos distintos a las jerarquías. Mantendremos en una misma tabla todos los tipos de los atributos y crearemos varias tablas de Jerarquías para cada atributo nombrando a cada una de ellas con el atributo al que se refiere como 'jerarquia_atributo'.

Por lo tanto las tablas que nos quedan son las que se muestran en la figura 6.3.

Figura 6.3: Modelo lógico de la base de datos

Ejemplo: Para una mayor comprensión de este modelo se va a incluir a continuación un ejemplo de una posible instancia de la base de datos.

Supongamos unos datos iniciales como los que se muestran en la tabla 6.1. En estos datos tenemos cuatro atributos: fnac, sexo, cod_postal y dolencia. Si marcamos dolencia como atributo sensible y el resto como no sensible ni identificador (NSoI) la tabla de atributos quedaría como se muestra en la tabla 6.2.

Fnac	**Sexo**	**Cod_postal**	**Dolencia**
1986	M	53715	gripe
1996	F	53715	neumonía
1986	M	53703	bronquitis
1986	M	53703	fractura brazo
1996	F	53706	apendicitis
1986	F	53706	fractura pierna

Tabla 6.1: Instancia de datos

atributo	**tipo**
fnac	NSoI
sexo	NSoI
cod_postal	NSoI
dolencia	sensible

Tabla 6.2: Instancia de atributos

Las jerarquías de generalización de los distintos atributos se muestran en las tablas 6.3, 6.4, 6.5. Como decíamos en el apartado anterior, cada jerarquía puede tener una cantidad diferente de niveles. En este caso se muestran jerarquías de 2 o 3 niveles, pero una jerarquía puede tener cualquier número de niveles.

level_0	**level_1**	**level_2**
53706	5370*	537**
53715	5371*	537**
53703	5370*	537**

Tabla 6.3: Instancia de Jerarquía_cod_postal

level_0	**level_1**	**level_2**
1986	198*	19**
1996	199*	19**

Tabla 6.4: Instancia de Jerarquía_fnac

level_0	**level_1**
M	Persona
F	Persona

Tabla 6.5: Instancia de Jerarquía_sexo

Si marcamos como atributos cuasi-identificadores a los atributos fnac y cod_postal e indicamos un valor de k=3. Según el algoritmo Incógnito se crearía en primer lugar la tabla de nodos_1 compuesta por 1 atributo y sus posibles generalizaciones (tabla 6.6). Y la tabla de enlaces_1 se crearía de manera que se genere un grafo lineal en el que cada nodo tiene como máximo un padre y un hijo (tabla 6.8). Para los distintos nodos en esta tabla se comprobaría si se cumple la k-anonimidad, que en este caso es 3-anonimidad. Para el atributo fnac vemos que sólamente se cumple esta propiedad cuando generalizamos al segundo nivel de la jerarquía. Por lo tanto, se crea la tabla de nodos_2 combinando las generalizaciones de fnac que cumplen la k-anonimidad con todas las posibles generalizaciones del siguiente atributo. De esta manera la tabla nodos_2 quedaría como se muestra en la tabla 6.7 y sus enlaces son los mostrados en la tabla 6.9.

Como únicamente tenemos dos atributos cuasi-identificadores sólo se crearán dos tablas de nodos y enlaces. A continuación habría que comprobar que nodos de la tabla de Nodos_2 cumple la k-anonimidad y estos serían las generalizaciones candidatas.

id	dim_0	index_0
1	fnac	0
2	fnac	1
3	fnac	2

Tabla 6.6: Instancia de Nodos_1

id	dim_0	index_0	dim_1	index_1
1	fnac	2	cod_postal	0
2	fnac	2	cod_postal	1
3	fnac	2	cod_postal	2

Tabla 6.7: Instancia de Nodos_2

start	end
1	2
2	3

Tabla 6.8: Instancia de Enlaces_1

start	end
1	2
2	3

Tabla 6.9: Instancia de Enlaces_2

6.3. Diagrama de paquetes

En este diagrama se definen los distintos paquetes a nivel lógico, es decir las clases e interfaces que forman parte de la aplicación y la dependencia entre ellos. Además de las librerías que se importarán y utilizarán para el desarrollo de la aplicación [8].

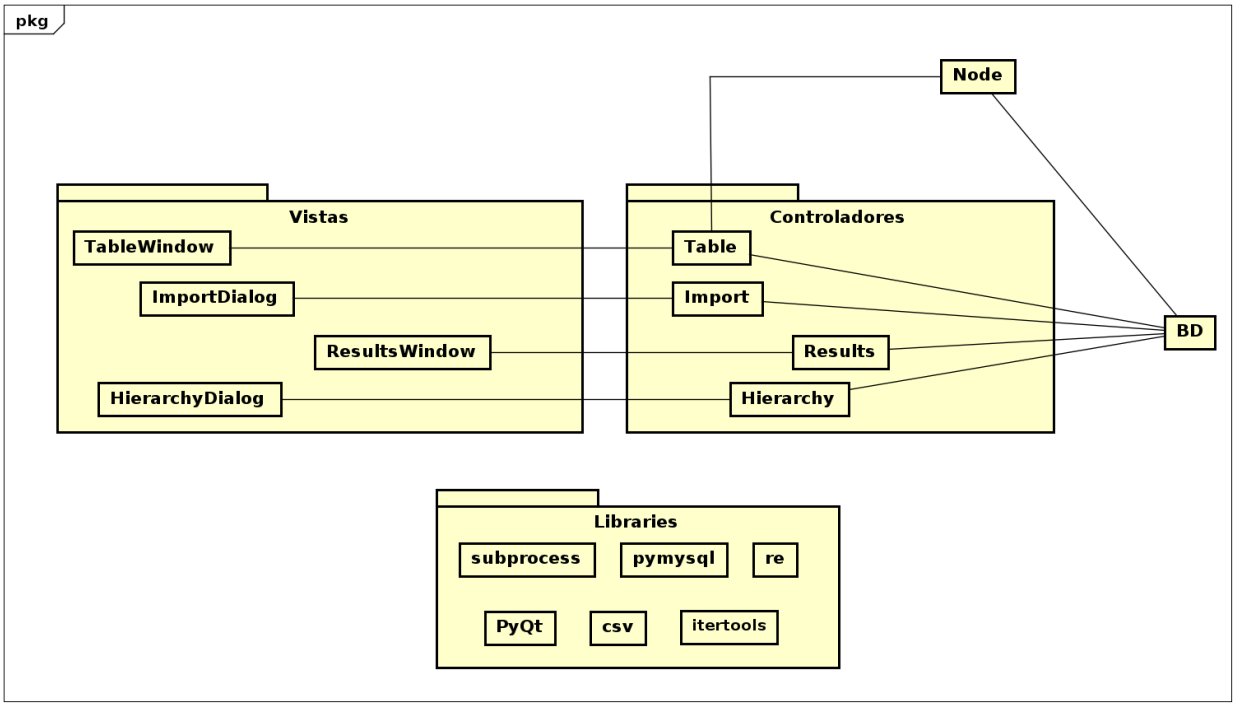

Figura 6.4: Diagrama de paquetes

Las vistas de las que se compone la aplicación son las siguientes:

- **TableWindow:** La vista principal, donde se mostrarán los datos iniciales sin anonimizar y los distintos menús para indicar los parámetros necesarios para anonimizar estos datos.
- **ImportDialog:** El diálogo mediante el cual se indican el fichero de datos que se quiere anonimizar y los parámetros necesarios para importar estos datos a una base de datos relacional (el separador de los datos y los tipos y tamaños de los atributos que los componen).

- **HierarchyDialog:** El diálogo en el cual se indica las jerarquías de generalización de los atributos cuasi-identificadores.
- **ResultsWindow:** La ventana de resultados, donde se mostrarán las generalizaciones candidatas para los cuasi-identificadores y valor de k escogidos. También se podrá aplicar la generalización deseada a los datos y se podrán exportar los datos anonimizados.

La aplicación dispone de cuatro controladores, uno asociado a cada una de las vistas y todos ellos en constante comunicación con la base de datos. Para comunicarse con la base de datos disponemos de una clase con nombre 'BD' la cual contiene todos los métodos necesarios para insertar, actualizar, borrar o consultar información de la base de datos. También tenemos la case **Node**, la cual es esencial para la implementación del algoritmo Incógnito. Cada instancia de esta clase representa cada uno de los nodos del árbol de generalizaciones candidatas del algoritmo. Contiene todos los atributos necesarios para la implementación del algoritmo (el valor de la generalización que representa, su conjunto de frecuencias, referencias a sus nodos padres...).

Por último, se utilizarán las siguientes librerías:

- **subprocess**: Esta librería permite ejecutar nuevos programas o procesos y obtener sus códigos de retorno. En este caso se utilizará para la ejecución del bash script para importar los datos desde un fichero a la base de datos de mysql [30].
- **pymysql**: Este paquete permite conectar la aplicación con la base de datos y ejecutar sentencias mysql desde código Python [25].
- **PyQt**: Esta librería es un conjunto de enlaces de Python para el marco de aplicación Qt. Qt es un conjunto de bibliotecas y herramientas de desarrollo C++ para la implementación de interfaces gráficas de usuario. Será la librería utilizada para realizar la interfaz de la aplicación [27].
- **csv**: El módulo csv implementa clases para leer y escribir datos tabulares en formato CSV. Se utilizará para escribir y leer todos los ficheros de configuración necesarios [6].
- **re**: Este módulo proporciona operaciones de coincidencia de expresiones regulares [28].
- **itertools**: Este módulo de python implementa distintas herramientas de iteradores, como por ejemplo un acumulador, una permutación, una combinación... En este caso se utiliza para la propuesta de atributos cuasi-identificadores explicada posteriormente, para la cual se necesita calcular todas las combinaciones posibles de varios números [16].

6.4. Diagrama de secuencia

Para entender más fácilmente los pasos realizados en el proceso de anonimización se incluye a continuación un diagrama de secuencia.

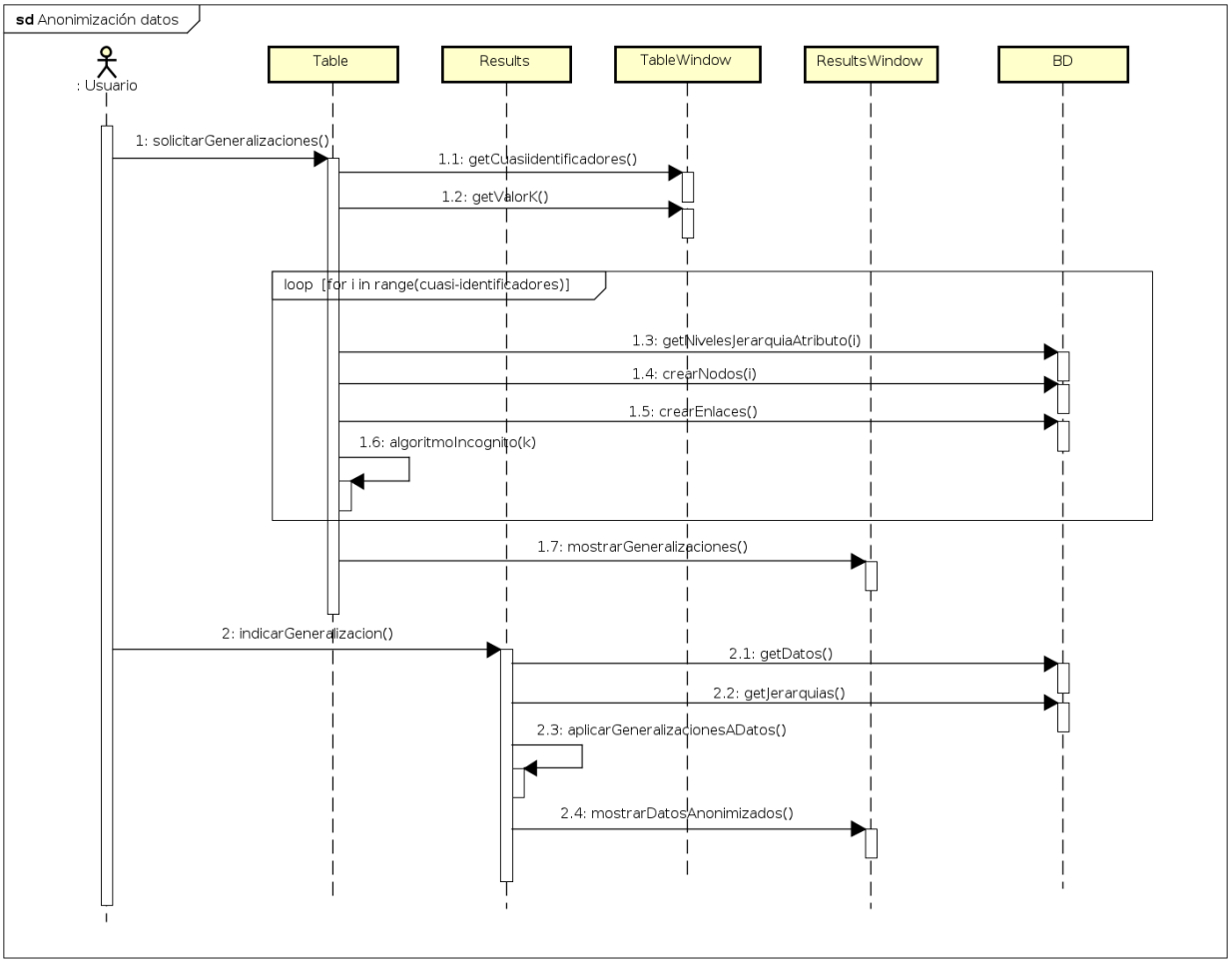

Figura 6.5: Diagrama de secuencia Anonimizar

La clase Table es el controlador de la vista principal 'TableWindow'. El proceso de anonimización comienza cuando se acciona esta orden en la vista principal. El controlador recoge de su vista todos los datos necesarios para la anonimización, es decir, los cuasi-identificadores y el valor de k, y realiza el algoritmo de incógnito, en el cual se recorre un bucle tantas veces como atributos cuasi-identificadores haya. En cada iteración del bucle se comprueba los niveles de generalización que tiene el atributo correspondiente y se crea una tabla nodos con las posibles generalizaciones y una tabla con los enlaces que unen estos nodos. Se comprueban los nodos que cumplen con la propiedad de k-anonimato y se marcan como posibles generalizaciones candidatas. Cuando el algoritmo ha terminado se muestran las posibles generalizaciones candidatas en la vista 'ResultsWindow', cuyo controlador 'Results' queda a la espera de que el usuario indique la generalización que quiere aplicar a los datos. Cuando este lo hace, se aplica la generalización correspondiente según las jerarquías de cada atributo y se muestran los datos anonimizados.

6.5. Diagrama de despliegue

En la figura 6.6 se muestra el diagrama de despliegue del sistema. Este muestra la disposición de las particiones físicas del sistema de información y la asignación de los componentes software a estas particiones. Es decir, las relaciones físicas entre los componentes software y hardware [5].

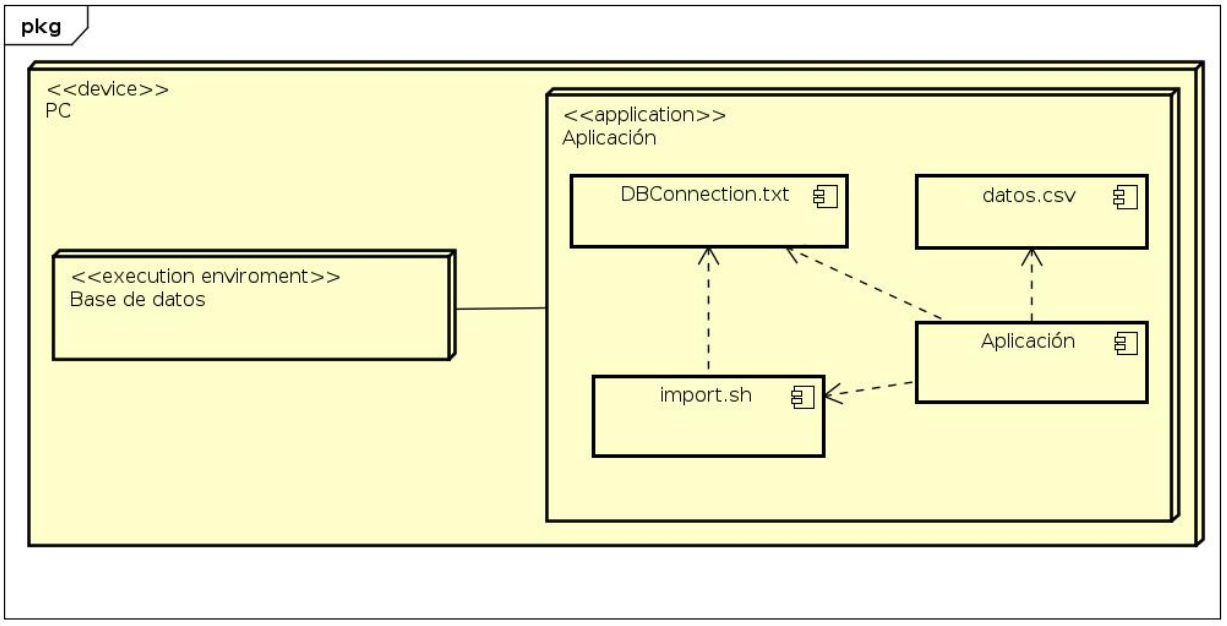

Figura 6.6: Diagrama de despliegue

Todo el software necesario está desplegado en el mismo dispositivo, que en este caso es un PC. Dentro de este, existen dos nodos los cuales estarán en continuo contacto para el correcto funcionamiento de la aplicación.

- **Base de datos:** será un servidor mysql instalado en la misma máquina en la que se ejecute la aplicación. En este se guardarán los datos especificados en el apartado anterior. [6.2]
- **Aplicación:** la propia aplicación, la cual estará compuesta por el código fuente de la misma, un archivo '.csv' o '.txt' el cual contendrá los datos a anonimizar y un fichero denominado 'import.sh' que será necesario para importar los datos a la base de datos. Los datos para conectarse a la base de datos se guardan en el archivo 'DBConnection.txt'. Este será necesario tanto para importar los datos, como para conectarse a estos desde la aplicación.

6.6. Patrones

Los patrones de diseño son soluciones habituales a problemas comunes en el diseño de software. En nuestro caso, al disponer de una interfaz y una base de datos, el patrón que más encaja con el proyecto es el patrón Modelo-Vista-Controlador.

6.6.1. MVC

Este patrón expresa como organizar y estructurar los componentes de la aplicación, sus responsabilidades y las relaciones existentes entre cada uno de ellos.

La aplicación se divide en tres capas o grupos de componentes, el **modelo**, en el que se indica una representación de los datos del dominio, es decir, se crean entidades que nos servirán para almacenar la información necesaria. La siguiente capa que describe este patrón es la capa de la **vista** que es la responsable de generar la interfaz de la aplicación, es aquella que interacciona directamente con el usuario. Por último, el **controlador**, es el intermediario entre el modelo y la vista. Es el que se encarga de interpretar las acciones del usuario sobre la vista, realizar los cambios correspondientes en el modelo, y adaptar la vista en correlación a esto. Por tanto, el controlador se podría definir como un coordinador general del sistema.

El patrón también define las relaciones existentes entre los tres componentes del MVC y con el usuario. Como se muestra en la figura 6.7, las acciones realizadas por el usuario las recogerá y procesará el controlador, el cual será también el responsable de modificar tanto la vista como el modelo. Por otra parte, el modelo únicamente informará sobre sus datos al controlador. Y por último, la vista serán los datos mostrados al usuario [1].

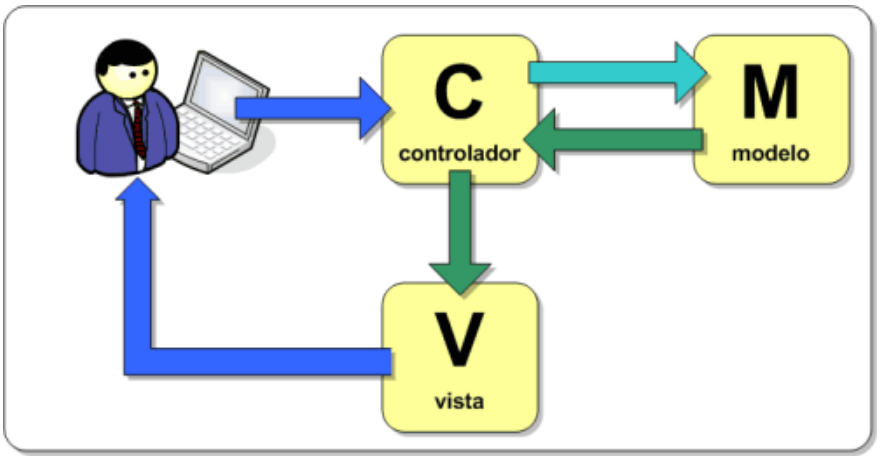

Figura 6.7: Relaciones en el MVC

Hay ciertas interacciones que el patrón MVC en sí no indica como se realizan, por ello, este tiene diferentes versiones o diferentes patrones que indican como se gestiona por ejemplo las consultas a la base de datos, como está organizado el modelo en relación con la base de datos, si tenemos un solo controlador para toda la aplicación o si vamos a disponer de varios... Para explicar como se van a realizar todas estas acciones se definen los siguientes patrones [12]:

Transaction Script

Para estructurar el modelo vamos a utilizar el patrón Transaction Script, ya que la lógica de la aplicación es bastante simple. Este patrón organiza toda la lógica como un procedimiento, realizando llamadas directamente a la base de datos para realizar cambios en esta o simplemente mostrar información. Cada transacción tendrá su propio script de transacción, aunque las subtareas comunes se pueden dividir en subprocedimientos.

Table Data Gateway

Para interaccionar con la base de datos se utilizará el patrón Table Data Gateway, el cual es el idóneo para compaginar con el Transaction Script elegido para la lógica.

Se basa en que un Table Data Gateway contiene todo el SQL para acceder a una sola tabla o vista: selecciona, inserta, actualiza y elimina. Otro código llama a sus métodos para toda interacción con la base de datos.

Page Controller

Para este proyecto vamos a tener un controlador por cada una de las vistas, ya que aunque el número de vistas que tenga la aplicación será bastante bajo, la gestión de estas vistas constará de una gran cantidad de código, en especial la que tenga que realizar los procedimientos correspondientes para anonimizar los datos. Entonces, para mayor claridad se utilizará un controlador por cada vista.

Capítulo 7

Implementación

7.1. Tecnología utilizada

Para la implementación de este proyecto se ha realizado una aplicación de escritorio, con una interfaz para interaccionar con el usuario y con una base de datos para guardar los datos necesarios.

Se ha utilizado el lenguaje **Python**, ya que es un lenguaje muy limpio y legible en el que con pocas líneas de código se puede conseguir lo mismo que en otros lenguajes con una extensión mucho mayor. Además, es un lenguaje interpretado, por lo que no necesita ser compilado y esto hace que sea más rápido que otros que si lo necesitan. Por último, Python es un lenguaje de alto nivel, es decir, es perfectamente entendible por humanos, sin embargo, también nos permite trabajar con bits y bytes como si fuera un lenguaje de medio nivel [19].

Para la realización de la interfaz se ha decidido utilizar una librería gráfica de Python, ya que se considera el método más sencillo para la realización de esta, y además la interfaz no es una parte primordial en la realización de este proyecto, por lo que se realizará una interfaz sencilla. Entre las diferentes librerías gráficas orientadas al lenguaje Python se ha elegido utilizar la librería **PyQt** ya que es una librería fácil de utilizar que dispone de la herramienta Qt designer con la que se pueden realizar las interfaces de manera muy sencilla. Esta herramienta proporciona una serie de widgets con los que se puede construir la interfaz simplemente arrastrándolos a un panel. Además, los distintos elementos son controlados a través de eventos. Cualquier acción del usuario como hacer clic en un botón o seleccionar un elemento de una lista lanzará un evento que hará muy sencillo el control de esta interfaz [26].

Para la base de datos relacional de la aplicación se ha elegido utilizar **MariaDB**, ya que es la alternativa libre de MySQL a Oracle [18].

Por último, se va a utilizar **Github** como repositorio para almacenar las distintas versiones de código de la aplicación y **Overleaf** como editor de LaTex para la realización de la memoria del proyecto [13] [21].

7.2. Importación de los datos

Para realizar la importación de los datos nos ajustamos a un modelo de ETL (Extract/Transform/Load), en el cual se transforman los datos originales en un formato apto para almacenarse en bases de datos relacionales.

Para importar los datos se necesita el fichero con los datos en formato TXT o CSV, el delimitador que separa estos datos y el tipo (varchar, char, int...) y longitud máxima de los distintos atributos.

Mediante un script de bash se lee la primera línea del fichero de datos que contiene los nombres de los atributos separados con el delimitador indicado, y se unen estos nombres con su tipo y longitud correspondiente para crear el esquema de la tabla MySQL correspondiente.

Cuando tenemos el esquema de la tabla creado, para introducir los datos en esta tabla es suficiente con utilizar el comando 'mysqlimport' que proporciona MySQL, en el cual se le indica el fichero con los datos, el delimitador que los separa, el carácter de final de línea y que ignore la primera línea la cual está compuesta por el nombre de los atributos.

7.3. Exportación de los datos

Al igual que la importación, la aplicación permite exportar los datos a ficheros CSV o TXT.

Para exportar los datos anonimizados se utiliza la biblioteca de Python 'csv'. Se utiliza la función 'csv.writer(file)' para abrir en modo escritura el archivo indicado por el usuario. Después se utiliza la función 'writerows()', a la cual se le puede pasar como parámetro una lista compuesta de tuplas o listas con los datos a escribir en el fichero.

Para componer el fichero en primer lugar se llama a la función writerows() con una lista compuesta por una sola lista con los nombres de los atributos para escribir la primera línea del fichero. En segundo lugar se le pasa a la misma función otra lista compuesta por varias tuplas con los distintos datos anonimizados. De esta manera ya quedarían exportados los datos anonimizados.

7.4. Incógnito

El algoritmo implementado para encontrar todas las generalizaciones candidatas es el algoritmo Incógnito explicado en los fundamentos teóricos de este proyecto [3].

El pseudocódigo de este algoritmo se muestra a continuación:

```
INPUT: A table T to be k-anonymized, a set Q of n quasi-identifier attributes,
and a set of dimension tables (one for each quasi-identifier in Q).
OUTPUT: The set of k-anonymous full-domain generalizations of T.
```

```
C1 = {Nodes in the domain generalization hierarchies of attributes in Q}
E1 = {Edges in the domain generalization hierarchies of attributes in Q}
queue = an empty queue
FOR i = 1 to n DO:
    Si = copy of Ci
    {roots} = {all nodes in Ci with no edge in Ei directed to them}
    Insert {roots} into queue, keepeng queue sorted by height
    WHILE queue is not empty DO:
        node = Remove first item from queue
        IF node is not marked THEN:
            IF node is a root THEN
                frequencySet = Compute frequency set of T with respect to
                attributes of node using T.
            ELSE
                frequencySet = Compute frequency set of T with respect to
                attributes of node using parent's frequency set.
            END IF
            Use frequencySet to check k-anonymity with respect to attributes
            of node
            IF T is k-anonymous with respect to attributes of node THEN:
                Mark all direct generalizations of node
            ELSE
                Delete node from Si
                Insert direct generalizations of node into queue, keeping
                queue ordered by height
            END IF
        END IF
    END WHILE
    Ci+1, Ei+1 = GraphGeneration(Si, Ei)
END FOR
RETURN Projection of attributes of Sn into T and dimension tables
```

El algoritmo consta de un bucle que se repite tantas veces como atributos cuasi-identificadores haya. Lo primero que se hace en el bucle es calcular Ci y Ei, los cuales en este proyecto hemos llamado como Nodes_x y Edges_x. A continuación se hace una copia de los Nodes_x y se guarda en Si, variable la cual va a contener los nodos candidatos. Luego se inserta en una cola los nodos root de Si, es decir, aquellos que no son generalización directa de ningún otro o que no tienen padres.

Después se comienzan a procesar los nodos de la cola, hasta que esta quede totalmente vacía. Por cada nodo de la cola si está marcado simplemente se saca de la cola y no se procesa, ya que sabemos que cumple la propiedad de k-anonimato. Si el nodo no está marcado se calcula su conjunto de frecuencias (directamente si el nodo es 'root', o si no lo es, a través del conjunto de frecuencias del padre).

Si el nodo cumple la propiedad de k-anonimato, según la propiedad de generalización todas sus generalizaciones también cumplirán esta propiedad, por lo tanto se marcan estos nodos para no comprobar sus conjuntos de frecuencia.

Si el nodo no cumple con la propiedad de k-anonimato se elimina del conjunto de nodos candidatos (Si) y se añaden sus generalizaciones directas a la cola.

Al finalizar cada iteración se crean las tablas Nodes_x+1 y Edges_x+1 a partir de los nodos candidatos contenidos en la variable 'Si', tal y como se explica en el algoritmo Incógnito.

Al terminar este algoritmo, la variable 'Si' contendrá las generalizaciones candidatas según el conjunto de atributos cuasi-identificadores que cumplen con la propiedad de k-anonimato.

7.4.1. Conjuntos de frecuencias

Los conjuntos de frecuencias de las posibles generalizaciones se pueden calcular de dos formas distintas. Si la generalización candidata se encuentra en un nodo hoja del árbol de búsqueda se calcula el conjunto de frecuencias directamente. Pero si la generalización candidata no es un nodo hoja y tiene un padre, se utiliza el conjunto de frecuencias del padre para calcular su conjunto de frecuencias.

- **getFrequencySet**: Se calcula el conjunto de frecuencias recogiendo todas las tuplas constituidas por únicamente los atributos cuasi-identificadores de los datos y contando cuantas veces se repite cada una de ellas.
- **getFrequencySetByFather**: Se calcula el conjunto de frecuencias tomando de partida el conjunto de frecuencias del padre, aplicándole correspondiente generalización y utilizando la propiedad de resumen para calcular su conjunto de frecuencias.

7.5. Propuesta de atributos cuasi-identificadores

Para realizar una propuesta de atributos cuasi-identificadores al usuario nos guiamos por el artículo de Mahagoub y Murtadha titulado 'Simple and effective method for selecting quasi-identifier' [20].

Para limitar el problema de identificación de un sujeto los métodos de anonimización suelen utilizar técnicas de control de divulgación, las cuales normalmente conducen a la pérdida de una gran cantidad de información. Sin embargo, este método mantiene un equilibrio entre la pérdida de información y la privacidad de los datos.

El primer objetivo del algoritmo propuesto es minimizar la pérdida de datos, para ello se eliminan todos los datos claves de la tabla. Es decir los atributos identificadores.

El segundo objetivo de este algoritmo, es el que nos interesa en esta sección, es la identificación de atributos cuasi-identificadores. Buscamos encontrar el subconjunto de atributos mínimo significativo y evaluar su importancia en términos de identidad personal. Para ello, se siguen los siguientes pasos:

- El primer paso es escoger un conjunto de atributos con total dependencia de la persona que se pueden encontrar en diferentes recursos de datos. (Nominated set)

- El segundo paso es encontrar el 'Power Set' (PS), esto es, el conjunto de todas las combinaciones posibles de los atributos escogidos en el primer paso.
- En el tercer paso del algoritmo se genera una tabla para cada elemento del PS que contenga todos los valores del conjunto de atributos acompañados del número de valores distintos en la tabla.
- El último paso consta de elegir el elemento del PS con el número máximo de tuplas distintas, este elemento será el conjunto de cuasi-identificadores. Si existe más de un elemento con el número máximo de tuplas se selecciona el elemento con el número mínimo de atributos.

De esta manera conseguimos que el conjunto de atributos cuasi-identificadores sea aquel que consigue abarcar un mayor número de tuplas distintas, lo que es igual a un menor número de tuplas iguales, y lo que equivale a una mayor probabilidad de reidentificación de la persona.

Sin embargo, para perder la menor cantidad de información posible escogemos el elemento con el menor número de atributos. De esta manera aplicaremos la generalización a menos atributos, perdiendo menos información pero manteniendo la privacidad.

Este es el algoritmo que se explica en el artículo mencionado. Sin embargo, en la práctica creemos que no es necesario escoger el elemento con absolutamente el mayor número de valores distintos. Es decir, si tenemos dos elementos, uno con 2 atributos y 2000 valores distintos y otro elemento con 4 atributos y 2010 valores distintos, escogiendo como conjunto de cuasi-identificadores el primero la privacidad es prácticamente la misma. Sin embargo, si que hay diferencia en la pérdida de información escogiendo el elemento con 2 atributos o el de 4 atributos.

Para implementar este algoritmo se ha fijado un porcentaje de un 5 % para aceptar la variación de elementos distintos. Por ejemplo, si fijamos el porcentaje en un 5 % el valor máximo de valores distintos es 2010, pero si existe un elemento con un menor número de atributos y con número de valores distintos de 1909 (2010-(2010*0.05)) o mayor, se escogerá este elemento como conjunto de atributos cuasi-identificadores.

7.5.1. Ejemplos

Para entender mejor este algoritmo se van a explicar dos ejemplos a continuación.

Para el primer ejemplo vamos a suponer que tenemos como 'Nominated Set' los atributos Edad, Sexo y Raza. Por lo tanto el 'Power Set' será Edad, Sexo, Raza, (Edad, Sexo), (Edad, Raza), (Sexo, Raza), (Edad, Sexo, Raza).

El número de valores distintos de cada elemento del PS se muestra en la figura 7.1. Como se puede ver en esta tabla el elemento con un máximo número de tuplas distintas es el elemento (Edad, Sexo, Raza). Por lo tanto este sería el conjunto de atributos cuasi-identificadores.

Por otra parte, en el segundo ejemplo, tenemos como 'Nominated Set' a los atributos Código postal, Sexo y Raza. El PS sería por tanto Código postal, Sexo, Raza, (Código postal, Sexo), (Código postal, Raza), (Sexo, Raza), (Código postal, Sexo, Raza) y el número de valores distintos

Elemento	Número de tuplas
Edad	91
Sexo	2
Raza	5
Edad, Sexo	182
Edad, Raza	449
Sexo, Raza	10
Edad, Sexo, Raza	881

Tabla 7.1: Elementos del Power Set frente al número de tuplas distintas

de cada elemento del PS serían los mostrados en la tabla 7.2. En este caso aunque el elemento con un máximo número de tuplas distintas es el elemento (Código postal, Sexo, Raza), la diferencia entre los los valores de los elementos, (Código postal), (Código postal, Sexo), (Código postal, raza) y (Código postal, Sexo, Raza) es mínima. Por lo tanto, el elemento escogido para formar el conjunto de atributos cuasi-identificadores sería el que contenga un menor número de atributos, esto es, el elemento (Código postal).

Elemento	Número de tuplas
Código postal	21648
Sexo	2
Raza	5
Código postal, Sexo	22019
Código postal, Raza	21942
Sexo, Raza	10
Código postal, Sexo, Raza	22188

Tabla 7.2: Elementos del Power Set acompañados del número de tuplas distintas

7.6. Propuesta de jerarquía de generalización

Las jerarquías de generalización de los atributos enteros suelen ser en forma de rango, es decir, en cada nivel se sustituye un dígito del número a procesar por un asterisco. Tal y como se muestra en la figura 7.3. Por esto y para ofrecer una mayor comodidad al usuario, se da la posibilidad de generar de manera automática una jerarquía de este tipo.

level_0	level_1	level_2
1986	198*	19**
1996	199*	19**

Tabla 7.3: Ejemplo de jerarquía para números enteros

Capítulo 8

Validación

Para validar el algoritmo implementado se han hecho varias pruebas con distintos conjuntos de datos y se han contrastado nuestros resultados con los obtenidos a partir de la herramienta ARX, explicada en el capítulo 4 y la cual ha sido validada por la AEPD.

8.1. Prueba 1

En la primera prueba se utilizó un conjunto de datos pequeño y sencillo. El conjunto de datos es el mostrado en la figura 8.1.

fnac	sexo	cod_postal	dolencia
1986	M	53715	gripe
1996	F	53715	neumonía
1986	M	53703	bronquitis
1986	M	53703	fractura brazo
1996	F	53706	apendicitis
1986	F	53706	fractura pierna

Figura 8.1: Conjunto de datos de la prueba 1

Marcamos el atributo "dolencia" como atributo sensible, y "fnac" y "cod_postal" como cuasi-identificadores con las siguientes jerarquías:

Nivel 0	Nivel 1	Nivel 2
53706	5370*	537**
53715	5371*	537**
53703	5370*	537**

Figura 8.2: Jerarquía de generalización de cod_postal para la prueba 1

Nivel 0	Nivel 1	Nivel 2
1986	198*	19**
1996	199*	19**

Figura 8.3: Jerarquia de generalización de fnac para la prueba 1

Indicamos un valor de K igual a dos y generamos las generalizaciones candidatas desde ambas herramientas. La solución obtenida por ARX se muestra en la figura 8.4. La solución obtenida por

nuestra aplicación se muestra en la figura 8.5. Como se puede observar, ambas herramientas generan los mismos resultados. La única diferencia es que ARX solo da una opción como generalización óptima mientras que nuestra aplicación da todas las opciones óptimas.

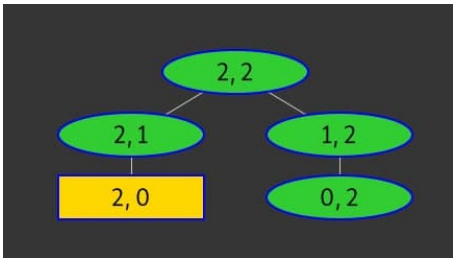

Figura 8.4: Generalizaciones candidatas dadas por ARX para la prueba 1

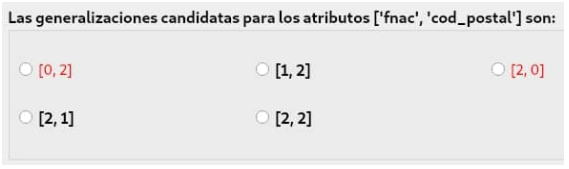

Figura 8.5: Generalizaciones candidatas dadas por la herramienta de este proyecto para la prueba 1

8.2. Prueba 2

Para la segunda prueba se utilizará unos datos más complejos, con jerarquías de generalización de más niveles. Los datos de entrada se muestran en la figura 8.6.

residencia	sexo	campo	ingresos
Navalcarnero	M	ENFERMERIA	medio
Torrelaguna	F	ENFERMERIA	medio
Ampudia	M	MEDICINA	alto
Palenzuela	M	MECANICA	bajo
Navalcarnero	F	QUIMICA	bajo
Navalcarnero	F	BIOLOGIA	bajo
Palenzuela	M	ENFERMERIA	medio
Palenzuela	M	MEDICINA	alto
Gama	F	INFORMATICA	alto
Mogarraz	F	TELECOMUNICACIONES	alto
Palenzuela	M	ELECTRONICA	alto
Medina del Campo	M	BIOLOGIA	medio
Medina de Rioseco	F	ELECTRONICA	alto
Candelario	M	BIOLOGIA	bajo
Ampudia	M	BIOLOGIA	alto
Palenzuela	M	MEDICINA	medio
Medina del Campo	M	ENFERMERIA	medio
Medina de Rioseco	F	ENFERMERIA	alto
Gama	M	MEDICINA	alto

Figura 8.6: Conjunto de datos de la prueba 2

Indicamos el atributo "ingresos" como atributo sensible, y el resto como cuasi-identificadores. Las jerarquías de generalización de los cuasi-identificadores se muestran en las figuras 8.7, 8.8 y 8.9:

Nivel 0	Nivel 1	Nivel 2	Nivel 3
Gama	Salamanca	CyL	España
Torrelaguna	Palencia	CyL	España
Candelario	Salamanca	CyL	España
Ampudia	Valladolid	CyL	España
Navalcarnero	Madrid	Madrid	España
Medina del ...	Palencia	CyL	España
Medina de ...	Palencia	CyL	España
Palenzuela	Valladolid	CyL	España
Mogarraz	Madrid	Madrid	España

Figura 8.7: Jerarquía de generalización de residencia para la prueba 2

Nivel 0	Nivel 1
F	Persona
M	Persona

Figura 8.8: Jerarquía de generalización de sexo para la prueba 2

Nivel 0	Nivel 1	Nivel 2	Nivel 3
MEDICINA	ciencias	natural	general
INFORMATICA	salud	natural	general
BIOLOGIA	TIC	ingenieria	general
TELECOMUNICACIONES	TIC	ingenieria	general
QUIMICA	ciencias	natural	general
MECANICA	salud	natural	general
ENFERMERIA	industrial	ingenieria	general
ELECTRONICA	industrial	ingenieria	general

Figura 8.9: Jerarquía de generalización de campo para la prueba 2

Indicamos un valor de K igual a 3 y generamos las generalizaciones candidatas desde ambas herramientas. La solución obtenida por ARX se muestra en la figura 8.10 y la obtenida por la aplicación propia en la figura 8.11. Como se puede observar las generalizaciones candidatas obtenidas desde ambas herramientas son las mismas. Sin embargo ARX muestra una generalización óptima distinta, ya que se utilizan técnicas distintas para calcular esta. En el caso de ARX se utilizan ciertas fórmulas para calcular la pérdida de información y la probabilidad de reidentificación, y a partir de esto se calcula la generalización óptima. En el caso de la herramienta de este proyecto se considera la generalización óptima aquella con unos niveles de generalización más bajos.

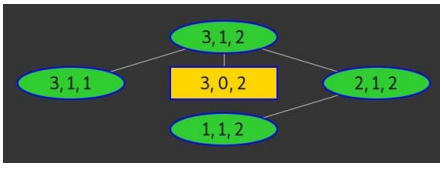

Figura 8.10: Generalizaciones candidatas dadas por ARX para la prueba 2

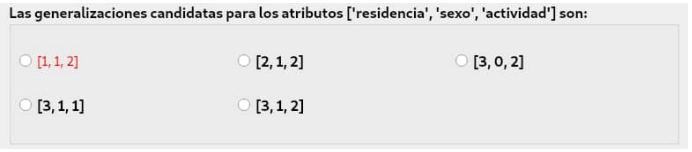

Figura 8.11: Generalizaciones candidatas dadas por la herramienta de este proyecto para la prueba 2

8.3. Prueba 3

Para la última prueba, se utilizó un conjunto de datos sencillo pero con una cantidad de datos elevada. El conjunto de datos ocupa 5 KB de memoria, y se generó aleatoriamente. Este se muestra en la figura 8.12.

fnac	sexo	cod_postal	dolencia
1972	M	53710	enfermedad
1984	M	53703	enfermedad
1982	F	53710	enfermedad
1979	F	53715	enfermedad
1984	F	53706	enfermedad
1983	M	53706	enfermedad
1981	M	53703	enfermedad
1970	F	53706	enfermedad
1983	F	53703	enfermedad
1985	F	53715	enfermedad
1989	F	53706	enfermedad
1972	F	53710	enfermedad
1975	F	53703	enfermedad
1986	F	53703	enfermedad
1981	F	53715	enfermedad
1988	M	53706	enfermedad
1979	M	53703	enfermedad
1982	F	53715	enfermedad
1974	F	53706	enfermedad

Figura 8.12: Conjunto de datos de la prueba 3

Se marcan como atributos cuasi-identificadores "fnac", "sexo" y "cod_postal", y "docencia" como atributo sensible. Las jerarquías de generalización se muestran en las figuras 8.13, 8.14 y 8.15:

Como la cantidad de datos es muy grande, se marca un valor de K bastante alto, en este caso se marca K=200. Generamos las generalizaciones candidatas desde ambas herramientas y obtenemos las siguientes soluciones. ARX muestra únicamente 6 generalizaciones candidatas, como se puede ver en la figura 8.16. Sin embargo hay muchas más generalizaciones entre medias de estas que no se muestran. En el conjunto de candidatos de nuestra herramienta se muestran las mismas generalizaciones que indica ARX, e incluye también las intermedias que son generalización directa de otras generalizaciones que si cumplen la k-anonimidad. Las generalizaciones candidatas dadas por nuestra herramienta se muestran en la figura 8.17. En este caso, la generalización óptima coincide en ambos casos.

Nivel 0	Nivel 1
F	Persona
M	Persona

Figura 8.13: Jerarquía de generalización de sexo para la prueba 3

Nivel 0	Nivel 1	Nivel 2
1970	197*	19**
1971	197*	19**
1972	197*	19**
1973	197*	19**
1974	197*	19**
1975	197*	19**
1976	197*	19**
1977	197*	19**
1978	197*	19**
1979	197*	19**
1980	198*	19**
1981	198*	19**
1982	198*	19**
1983	198*	19**
1984	198*	19**
1985	198*	19**
1986	198*	19**

Figura 8.14: Jerarquía de generalización de fnac para la prueba 3

Nivel 0	Nivel 1	Nivel 2
53706	5370*	537**
53715	5371*	537**
53710	5371*	537**
53703	5370*	537**

Figura 8.15: Jerarquía de generalización de cod_postal para la prueba 3

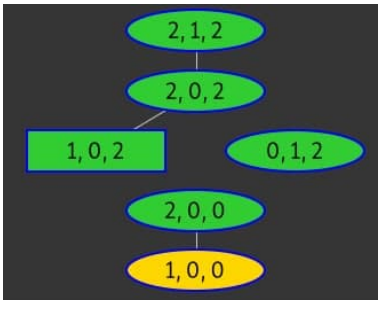

Figura 8.16: Generalizaciones candidatas dadas por ARX para la prueba 3

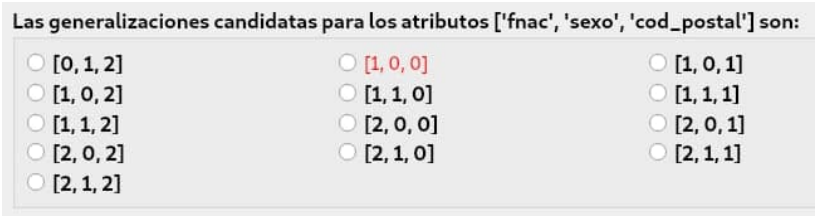

Figura 8.17: Generalizaciones candidatas dadas por la herramienta de este proyecto para la prueba 3

Capítulo 9

Conclusiones y trabajo futuro

9.1. Conclusiones

Los principales objetivos marcados inicialmente se han cumplido. Se ha realizado una investigación profunda sobre anonimización y más en concreto sobre K-anonimización, llegando a comprender totalmente esta técnica. Se aprendieron los métodos de generalización y de supresión utilizados para anonimizar los datos y se descubrió que el método de supresión prácticamente no se utiliza en la actualidad, ya que la pérdida total de algunos datos no suele ser de utilidad.

Por otro lado también se cumplieron algunos de los objetivos secundarios, como la generación de una propuesta de atributos cuasi-identificadores y también la propuesta de una jerarquía de generalización para atributos numéricos. Sin embargo, no se llegó a implementar la propuesta del valor de K óptimo, ya que para esto eran necesarias variables estadísticas complejas, con las cuales había que encontrar un valor de K que maximizara la privacidad, minimizando a su vez la pérdida de información, lo cual se escapaba del alcance de este trabajo.

No se utilizaron datos reales para las pruebas para evitar cuestiones adicionales como la declaración del tratamiento de datos personales para que se incluyera en el Registro de Actividades de Tratamiento de Datos Personales (RAT) de la UVa [9]. No obstante, la aplicación se diseño siguiendo requisitos de seguridad para poder ser usada con datos reales. Por ejemplo, se evitó almacenar los datos de entrada, lo cual minimiza el riesgo de fuga de información y brechas de seguridad, esto es, minimizando los riesgos de privacidad.

La realización de una aplicación que implemente el algoritmo Incógnito sobre bases de datos relacionales sirvió para asentar los conocimientos adquiridos en la investigación previa ya que se pusieron todos estos en práctica. También ayudó a comprender correctamente algunos conceptos que inicialmente se entendieron de manera errónea. Además, se aprendió a utilizar tecnologías nuevas que no se habían usado anteriormente, como la librería PyQT, la cual fue un gran descubrimiento, pues es una librería muy intuitiva y fácil de usar. Por último, se pudo mejorar la destreza del lenguaje de programación Python con el cual se había trabajado muy poco.

La planificación se cumplió durante las primeras semanas, sin embargo, la fase de implementación llevó más tiempo de lo pensado inicialmente. Además durante esta fase hubo que modificar algunos

aspectos de las fases de análisis y diseño. La fase de pruebas sin embargo se llevó a cabo en menos tiempo del planificado.

9.2. Trabajo futuro

Se pueden realizar mejoras y avances sobre la aplicación realizada. Una de estas es la propuesta del valor de K óptimo calculando la pérdida de información y el nivel de privacidad que se tiene con cada posible valor, escogiendo aquel que maximice la privacidad y minimice la pérdida de información.

Ya que la aplicación procesa datos privados no anonimizados, sería conveniente avanzar con unos requisitos de privacidad más exigentes si se trata con datos reales, como es asegurarse que los datos del usuario se borran realmente al terminar el procedimiento de anonimización. Actualmente se borran todos los datos de la base de datos pero no se comprueba que están borrados de disco y de memoria. Para esto se debería realizar una operación atómica reescribiendo sobre los mismos bloques donde se almacenaron los datos del usuario. Además, el proceso debería estar securizado. Una posible solución es que las máquinas donde se ejecuta estén situadas en entornos de ejecución seguros para prevenir posibles ataques.

Por último, como se trata con datos personales, este tratamiento debería registrarse en el Registro de Actividades de Tratamiento de Datos Personales previsto en el artículo 30 del Reglamento General de Protección de Datos (RGPD) [11].

Bibliografía

[1] José María Aguilar. «¿Qué es el patrón MVC en programación y por qué es úti?» En: *Campus MVP* (2019). URL: https://www.campusmvp.es/recursos/post/que-es-el-patron-mvc-en-programacion-y-por-que-es-util.aspx.

[2] *Amnesia Anonymization Tool*. URL: https://amnesia.openaire.eu/.

[3] *ARX - Data Anonymization Tool*. URL: https://arx.deidentifier.org/.

[4] BOE. «Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales». En: (2018). URL: https://www.boe.es/boe/dias/2018/12/06/pdfs/BOE-A-2018-16673.pdf.

[5] Manuel Cillero. «Diagrama de Despliegue». En: *manuel.cillero.es* (). URL: https://manuel.cillero.es/doc/metodologia/metrica-3/tecnicas/diagrama-de-despliegue/.

[6] *csv — Lectura y escritura de archivos CSV*. 3.9.6. Python. URL: https://docs.python.org/es/3/library/csv.html.

[7] Jenny Dearborn. «La importancia de los datos: claves para su análisis». En: *Conecta software* (2020). URL: https://conectasoftware.com/libros/area/marketing-y-ventas/la-importancia-de-los-datos/.

[8] «Diagrama de paquetes». En: *DiagramasUML* (). URL: https://diagramasuml.com/paquetes/.

[9] David Sanz Esteban. *Registro de actividades de tratamiento*. Universidad de Valladolid. 2021. URL: https://secretariageneral.uva.es/competencias/proteccion-de-datos/registro-actividades-tratamiento/.

[10] etomas. «El orden de los algoritmos... esa gran O». En: *Plain Concepts* (2012). URL: https://geeks.ms/etomas/2012/11/28/el-orden-de-los-algoritmos-esa-gran-o/.

[11] Unión Europea. «Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo de 27 de abril de 2016 relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE (Reglamento General de Protección de Datos)». En: (2016). URL: https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A32016R0679.

[12] Martin Fowler. «Catalog of Patterns of Enterprise Application Architecture». En: (January 2003). URL: https://martinfowler.com/eaaCatalog/.

[13] *GitHub: Where the world builds software*. URL: https://github.com.

[14] M. Mercedes Martínez González. «Introducción práctica a la K-anonimización». En: *Diario La Ley* n.º 2 (2019).

[15] «IEEE Standard Glossary of Software Engineering Terminology». En: *IEEE Std 610.12-1990* (1990), págs. 1-84. DOI: 10.1109/IEEESTD.1990.101064.

[16] *itertools — Functions creating iterators for efficient looping*. 3.9.6. Python. URL: https://docs.python.org/3/library/itertools.html.

[17] Kristen LeFevre, David J. DeWitt y Raghu Ramakrishnan. «Incognito: Efficient Full-Domain K-Anonymity». En: *Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005*. Ed. por Fatma Özcan. ACM, 2005, págs. 49-60. DOI: 10.1145/1066157.1066164. URL: https://doi.org/10.1145/1066157.1066164.

[18] *MariaDB Foundation*. URL: https://mariadb.org.

[19] José Domingo Muñoz. «Por qué aprender a programar Python». En: *OpenWebinars* (2017). URL: https://openwebinars.net/blog/por-que-aprender-a-programar-python/.

[20] Amani Mahagoub Omer y Mohd Murtadha Bin Mohamad. «Simple and efective method for selecting quasi-identifier». En: vol. 89. No.2. Journal of Theoretical y Applied Information Technology, 2016. URL: http://www.jatit.org/volumes/Vol89No2/25Vol89No2.pdf.

[21] *Overleaf, Editor de LaTeX online*. URL: https://es.overleaf.com.

[22] Agencia Española de Protección de Datos (AEPD). «Guía de Privacidad desde el Diseño». En: (2019). URL: https://www.aepd.es/sites/default/files/2019-11/guia-privacidad-desde-diseno.pdf.

[23] Agencia Española de Protección de Datos (AEPD). «La K-anonimidad como medida de la privacidad». En: (2019). URL: https://www.aepd.es/sites/default/files/2019-09/nota-tecnica-kanonimidad.pdf.

[24] Agencia Española de Protección de Datos (AEPD). «Orientaciones y garantias en los procedimientos de anonimización de datos personales». En: (2019). URL: https://www.aepd.es/sites/default/files/2019-09/guia-orientaciones-procedimientos-anonimizacion.pdf.

[25] *PyMySQL*. 1.0.2. Python Package Index. URL: https://pypi.org/project/PyMySQL/#documentation.

[26] «PyQt vs Tkinter». En: *Dev* (2019). URL: https://dev.to/amigosmaker/pyqt-vs-tkinter-spanish-2n4k.

[27] *PyQt5 Reference Guide*. 5.15. Riverbank Computing. URL: https://www.riverbankcomputing.com/static/Docs/PyQt5/introduction.html.

[28] *re — Operaciones con expresiones regulares*. 3.9.6. Python. URL: https://docs.python.org/es/3/library/re.html?highlight=re#module-re.

[29] «Salario de programador en España». En: *Hack a boss* (2021). URL: https://hackaboss.com/blog/salario-programador-espana-2018-2019/.

[30] *subprocess — Subprocess management*. 3.9.6. Python. URL: https://docs.python.org/3/library/subprocess.html.

[31] Latanya Sweeney. «k-Anonymity: A Model for Protecting Privacy». En: *Int. J. Uncertain. Fuzziness Knowl. Based Syst.* 10.5 (2002), págs. 557-570. DOI: 10.1142/S0218488502001648. URL: https://doi.org/10.1142/S0218488502001648.

Capítulo 10

Anexo: Guía de usuario

10.1. Inicio de la aplicación e importación de datos

Inicialmente la aplicación se muestra como se ve en la figura 10.1, con un cuadro en blanco a la izquierda donde posteriormente se incorporarán los datos sin anonimizar. En la parte derecha de la interfaz tenemos las diferentes opciones. Cuando aún no hay datos para anonimizar, la única opción a la que podremos acceder es la de insertar datos. Cuando pulsemos esta opción se nos mostrará un diálogo con las diferentes opciones para poder abrir los datos que queremos anonimizar.

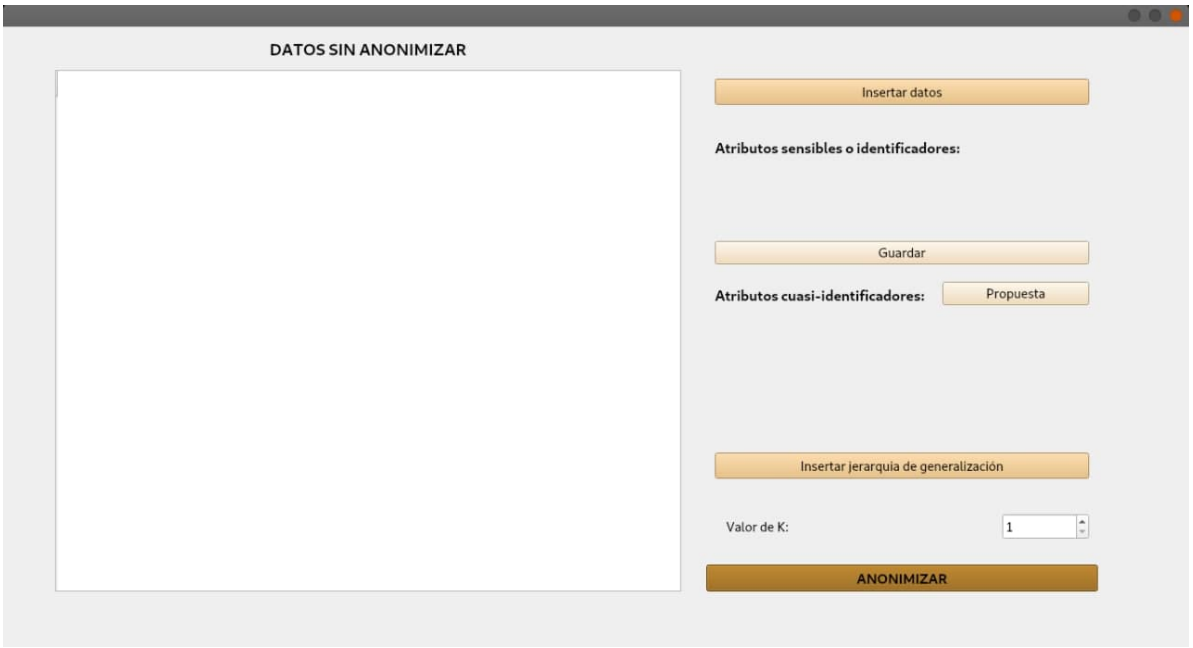

Figura 10.1: Aplicación al iniciarse

El diálogo para insertar los datos se muestra en la figura 10.2. En este menú se piden los siguientes datos:

- La ruta hacia los datos, es decir, el archivo con la tabla de datos para anonimizar. Esta ruta se puede insertar manualmente o automáticamente mediante un gestor de archivos el cual se abre al pulsando en el botón abrir.

- El separador de los datos, es decir, el carácter o caracteres que separa una columna de otra. En archivos con formato .csv suele ser una coma.
- Por último, es necesario indicar el tipo y la longitud máxima de los distintos atributos de los datos. Este paso se puede realizar con un archivo de configuración, al cual se podrá acceder de la misma manera que al archivo de datos, indicando la ruta explícitamente o mediante un gestor de archivos. Sin embargo, como este archivo puede ser un poco engorroso crearlo manualmente, se ofrece al usuario la posibilidad de crearlo a partir de la interfaz, pulsando en el botón de añadir manualmente. Cuando se acciona este botón aparecen en la parte inferior de la interfaz distintas opciones, como se muestra en la figura 10.3. Se indican los nombres de los distintos atributos de los datos indicados en la primera opción de este diálogo, un comboBox con las distintas opciones de tipos de datos de MySQL (varchar, char, int, double, boolean...), y finalmente un spinBox para indicar el tamaño máximo de cada atributo.

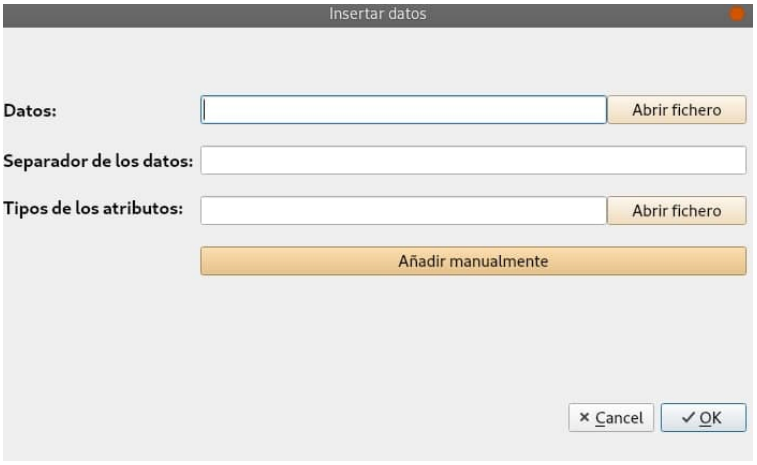

Figura 10.2: Vista para importar los datos

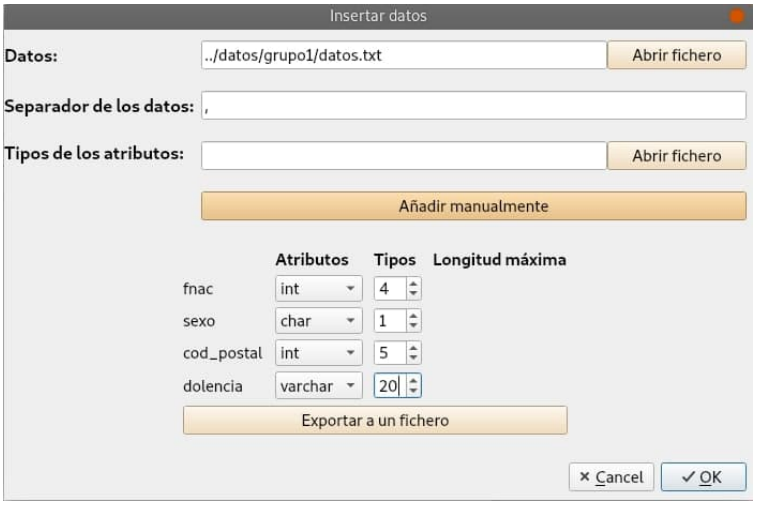

Figura 10.3: Vista para importar los datos añadiendo los tipos de los atributos manualmente

Rellenando todos estos campos ya tendremos importados los datos en la aplicación. Se volverá a la ventana principal, la cual ya tendrá rellena en la parte izquierda la vista de los datos con la tabla que se acaba de importar, y en la parte derecha los distintos atributos que contiene esta tabla. Esta vista se muestra en la figura 10.4.

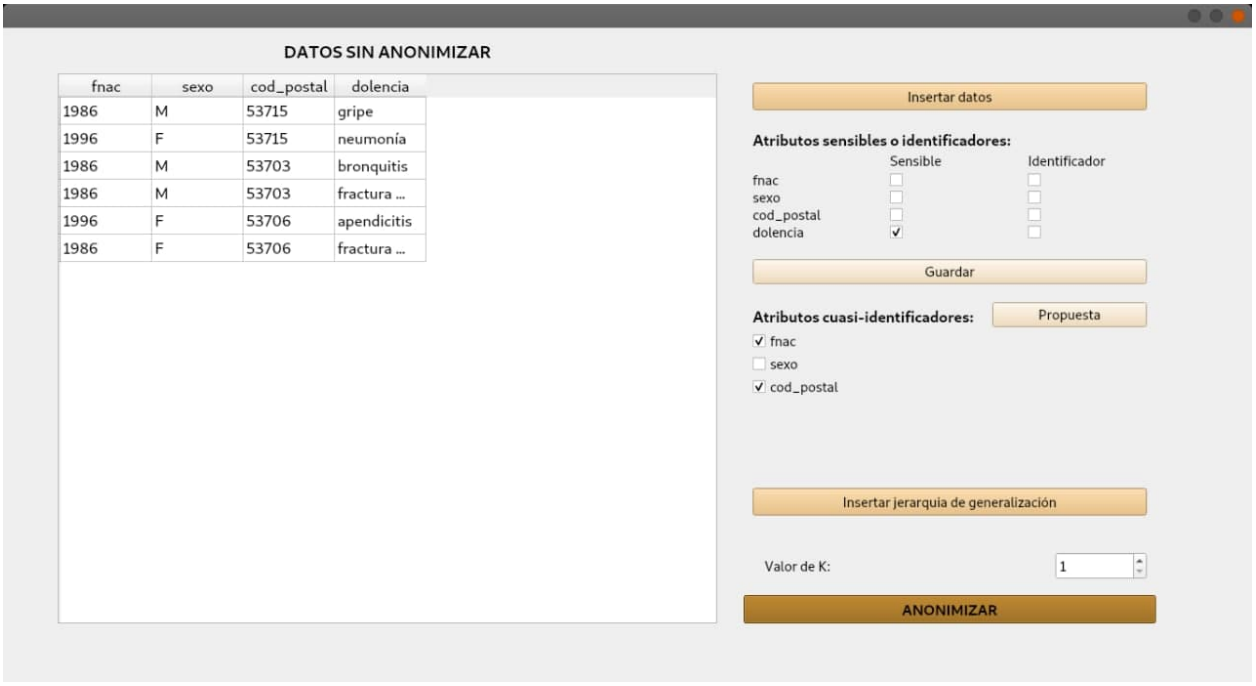

Figura 10.4: Vista principal de la aplicación

10.2. Selección de parámetros para la K-anonimización

En la parte derecha de la interfaz, donde nos encontramos el menú de opciones, tenemos inicialmente una lista con todos los atributos de los datos. En esta lista tendremos que indicar cuales son los atributos sensibles o identificadores. Los atributos sensibles son aquellos que queremos proteger y los atributos identificadores se eliminarán del conjunto de datos anonimizado. Una vez hayamos indicado correctamente estos atributos y pulsemos en el botón "guardar" se mostrará en la parte inferior el resto de atributos. De estos atributos podremos marcar aquellos que queramos que sean los cuasi-identificadores, es decir, a los que queremos aplicar los métodos de anonimización.

Debajo de la lista de atributos, se muestra un botón para insertar las distintas jerarquías de generalización de los atributos cuasi-identificadores. Cuando se pulsa este botón se abre el diálogo que se muestra en la figura 10.5. En este diálogo podremos indicar jerarquías para todos los atributos que no sean identificadores o sensibles, aunque no sean cuasi-identificadores. Sin embargo, para anonimizar los datos tendremos que tener jerarquías para todos los atributos que hayamos marcado como cuasi-identificadores.

Figura 10.5: Vista para indicar las jerarquías de generalización

Figura 10.6: Vista de jerarquía de generalización completa

10.3. Jerarquías de generalización

En el diálogo para indicar las jerarquías de generalización nos encontramos con dos opciones en la parte superior para importar o exportar las distintas jerarquías. Ambas opciones abren un gestor de archivos para indicar el archivo del que queremos importar la jerarquía o al que queremos exportarlo. Al importar una jerarquía se rellena la tabla automáticamente con los datos importa-

dos. Para exportar la jerarquía primeramente tendremos que rellenar la tabla con la jerarquía de generalización deseada.

Para rellenar la tabla con la jerarquía de generalización tenemos los botones de borrar nivel o añadir nivel, dependiendo de los niveles de generalización que queramos que tenga cada jerarquía. En la parte izquierda de la interfaz tenemos un comboBox para elegir el atributo cuasi-identificador del que queremos rellenar su jerarquía.

Para las jerarquías de atributos numéricos se da la opción de generarla automáticamente pulsando en el botón "generar propuesta numérica", lo que rellenará una jerarquía por rango, sustituyendo un dígito por un asterisco en cada nivel, tal y como se muestra en la figura 10.6. Para poder suprimir el atributo pulsamos en el botón "añadir supresión", lo que añadirá un último nivel donde todo son asteriscos, que significará la supresión del atributo.

Para guardar la jerarquía una vez rellenada en la aplicación se deberá pulsar en el botón de guardar datos que se muestra en la parte inferior de la vista.

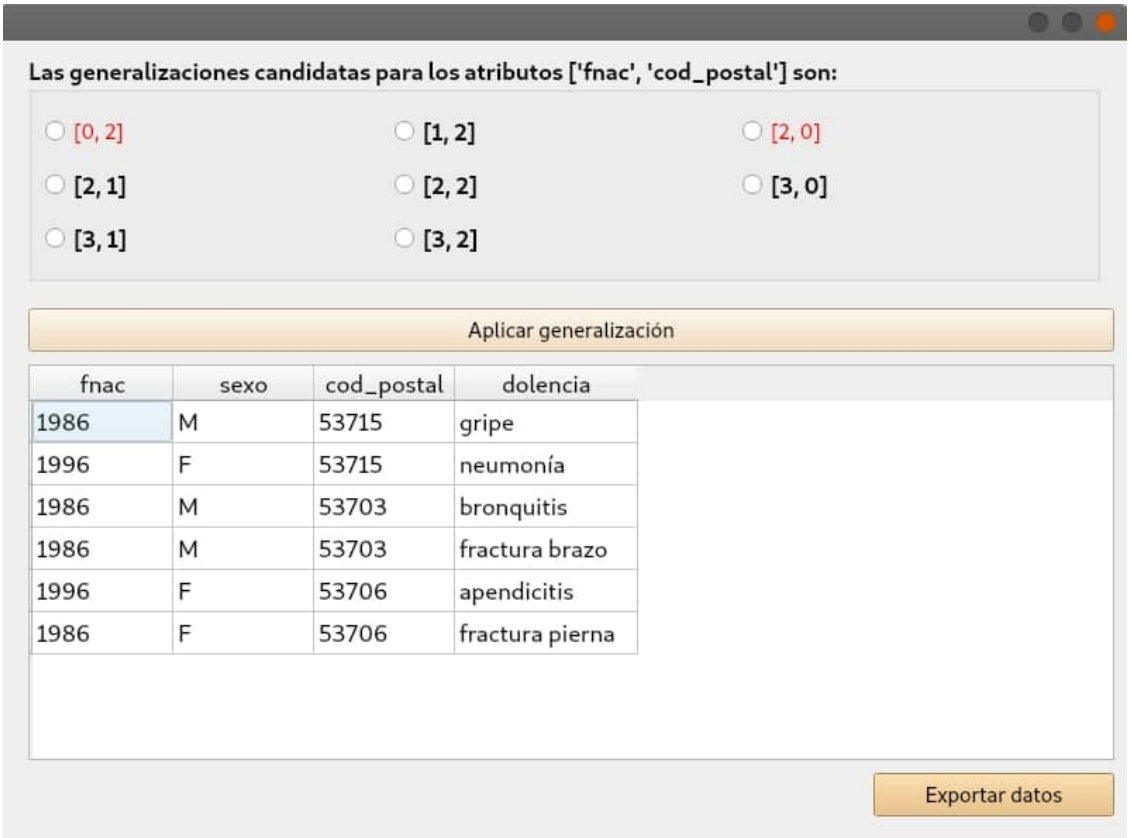

Figura 10.7: Vista de resultados para anonimizar los datos

10.4. Selección del valor de 'K'

Por último cuando tengamos todos los datos necesarios para aplicar el algoritmo de k-anonimización. Indicamos el valor de k que queremos que cumpla la anonimización y pulsamos el botón de anonimizar, el cual nos abrirá una ventana nueva como la que se muestra en la figura 10.7.

10.5. Anonimización

En la vista de resultados se muestran en la parte superior las generalizaciones candidatas que cumplen en valor de K indicado. Se muestran en rojo las generalizaciones óptimas. Y en la parte inferior se muestran los datos sin los atributos identificadores.

Para aplicar una generalización se selecciona la generalización deseada y se pulsa el botón 'Aplicar generalización', el cual modifica los datos de la tabla inferior con la generalización indicada.

10.6. Exportación de resultados

Por último, una vez que hemos aplicado la generalización deseada podemos exportar los datos anonimizados a un fichero .csv o .txt, con la opción del botón inferior de la aplicación.

Capítulo 11

Anexo: Manual de instalación

Para instalar la aplicación se requiere contar con:

- Base de datos MySQL o MariaDB
- Python

Se deben instalar los siguientes paquetes:

- **Qt5**: *sudo apt install qt5-default*
- **PyQt5**: *pip install PyQt5*
- **PyMySQL**: *pip install PyMySQL*

Es necesario tener una base de datos en el servidor MySQL o MariaDB. Se recomienda también crear un usuario específico para la aplicación que únicamente tenga acceso a esta base de datos. Para ello accedemos al servidor con el usuario 'root' y realizamos los siguientes pasos:

1. Creamos un usuario: *CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';*
2. Creamos la base de datos: *CREATE DATABASE 'database';*
3. Otorgamos al usuario todos los privilegios hacia esa base de datos: *GRANT ALL PRIVILEGES ON 'database'.* TO 'user'@'localhost';*

Por último modificamos el fichero DBconnection.txt con los datos correspondientes para conectarse a la base de datos que acabamos de crear. Este fichero contiene el siguiente formato:

```
host='localhost'
user='annon'
passwd='password'
database='TFG'
```

Para iniciar la aplicación sólamente habrá que ejecutar el archivo *start.sh*.

Capítulo 12

Anexo: RGPD

12.1. Considerando 26

Los principios de la protección de datos deben aplicarse a toda la información relativa a una persona física identificada o identificable. Los datos personales seudonimizados, que cabría atribuir a una persona física mediante la utilización de información adicional, deben considerarse información sobre una persona física identificable. Para determinar si una persona física es identificable, deben tenerse en cuenta todos los medios, como la singularización, que razonablemente pueda utilizar el responsable del tratamiento o cualquier otra persona para identificar directa o indirectamente a la persona física. Para determinar si existe una probabilidad razonable de que se utilicen medios para identificar a una persona física, deben tenerse en cuenta todos los factores objetivos, como los costes y el tiempo necesarios para la identificación, teniendo en cuenta tanto la tecnología disponible en el momento del tratamiento como los avances tecnológicos. Por lo tanto los principios de protección de datos no deben aplicarse a la información anónima, es decir información que no guarda relación con una persona física identificada o identificable, ni a los datos convertidos en anónimos de forma que el interesado no sea identificable, o deje de serlo. En consecuencia, el presente Reglamento no afecta al tratamiento de dicha información anónima, inclusive con fines estadísticos o de investigación.

12.2. Artículo 25: Protección de datos desde el diseño y por defecto

1. Teniendo en cuenta el estado de la técnica, el coste de la aplicación y la naturaleza, ámbito, contexto y fines del tratamiento, así como los riesgos de diversa probabilidad y gravedad que entraña el tratamiento para los derechos y libertades de las personas físicas, el responsable del tratamiento aplicará, tanto en el momento de determinar los medios de tratamiento como en el momento del propio tratamiento, medidas técnicas y organizativas apropiadas, como la seudonimización, concebidas para aplicar de forma efectiva los principios de protección de datos, como la minimización de datos, e integrar las garantías necesarias en el tratamiento, a fin de cumplir los requisitos del presente Reglamento y proteger los derechos de los interesados.

2. El responsable del tratamiento aplicará las medidas técnicas y organizativas apropiadas con miras a garantizar que, por defecto, solo sean objeto de tratamiento los datos personales que sean necesarios para cada uno de los fines específicos del tratamiento. Esta obligación se aplicará a la

cantidad de datos personales recogidos, a la extensión de su tratamiento, a su plazo de conservación y a su accesibilidad. Tales medidas garantizarán en particular que, por defecto, los datos personales no sean accesibles, sin la intervención de la persona, a un número indeterminado de personas físicas.

3. Podrá utilizarse un mecanismo de certificación aprobado con arreglo al artículo 42 como elemento que acredite el cumplimiento de las obligaciones establecidas en los apartados 1 y 2 del presente artículo.