Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Bases de Datos Orientadas a Grafos en la integración de datos para fines estadísticos, el caso del Sistema de Datos Integrados del ISTAC

Graph-Oriented Databases in the integration of data for statistical purposes, the case of the Integrated Data System of ISTAC

Jaime Rodríguez Pérez

La Laguna, 08 de septiembre de 2021

Dña. **Isabel Sánchez Berriel**, con N.I.F. 42.885.838-S profesora Contratada Doctora tipo 1 adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora

Dña. **Luz Marina Moreno de Antonio**, con N.I.F. 45.457.492-Q profesora Contratada Doctora tipo 1 adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como co-tutora

C E R T I F I C A N

Que la presente memoria titulada:

"Bases de Datos Orientadas a Grafos en la integración de datos para fines estadísticos, el caso del Sistema de Datos Integrados del ISTAC"

ha sido realizada bajo su dirección por D.Jaime

Rodríguez Pérez, con N.I.F.78646909-L.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 08 de Septiembre de 2021

Agradecimientos

A mis padres, por apoyarme siempre y darme fuerzas cuando yo no las tenía. Y sobre todo por el sobreesfuerzo que hicieron para que sus tres hijos estudien y se labren el futuro que desearon.

A mis hermanos, por no agobiarme nunca con mis estudios y ayudarme a despejar la mente cuando lo necesitaba.

A mi familia en general, por creer siempre en mí y animarme a llegar a donde estoy hoy.

A mis amigos por entender siempre como me sentía y no dudar nunca de mi.

A mi perro Mat, por ser el único que no me preguntaba “¿Cómo llevas el TFG?”, y con sus paseos relajarme.

Y a mis dos tutoras del trabajo de fin de grado, Isabel Sánchez y Luz Marina Moreno de Antonio, por ayudarme en todo, siempre con buena cara, por animarme, y por estar ante cualquier circunstancia con las fuerzas y la capacidad para hacer lo necesario para ayudarme y conseguir el objetivo.

Licencia

.

* Si quiere permitir que se compartan las adaptaciones de tu obra y quieres permitir usos comerciales de tu obra (licencia de Cultura Libre) indica:

Resumen

El objetivo de este Trabajo de Fin de Grado es el estudio de la viabilidad y beneficios del uso de un modelo de datos orientado a grafos para el Sistema de Datos Integrados (*iDatos*) utilizado en el banco de datos del ISTAC. Se analizan los registros y relaciones existentes entre fuentes administrativas así como otras posibles externas. Se pretende obtener el modelo de datos basado en grafos y la implementación del mismo en una prueba de concepto con un conjunto de datos reales o simulados en los casos en que estos no sean publicables. En concreto, la construcción de esta BDD se hará sobre Neo4j, producto open-source implementado en java y con una amplia comunidad.

Palabras clave: Gestión de datos maestros, estadística pública, Bases de Datos Orientadas a Grafos

Abstract

The objective of this Final Degree Project is to study the feasibility and benefits of using a graph-oriented data model for the Integrated Data System (iDatos) used in the ISTAC database. The records and existing relationships between administrative sources as well as other possible external sources are analyzed. It is intended to obtain the data model based on graphs and its implementation in a proof of concept with a set of real or simulated data in cases where these are not publishable. The possibility of obtaining relationships that are not explicit in the tables in the ISTAC iDatos system, but that can be inferred thanks to the use of the graph-oriented database, is also studied. Specifically, the construction of this BDD will be done on Neo4j, an open-source product implemented in Java and with a large community.

Keywords: Master Data Management, public statistics, graph-oriented database

Índice general

Índice de figuras

Capítulo 1 Introducción

El Instituto Canario de Estadística (ISTAC), es el órgano central del sistema estadístico autonómico y centro oficial de investigación del Gobierno de Canarias, cuyas funciones principales son: proveer información estadística y coordinar la actividad estadística pública. Entre sus directrices está especificado que se constituirá un banco de datos administrativos para fines estadísticos provenientes fundamentalmente de los ficheros administrativos de la Comunidad Autónoma de Canarias.

En la actualidad, en el marco del Plan Estadístico 2018-2022 se pretende impulsar el Sistema de Datos Integrados (*iDatos*) con el fin de producir estadísticas *multifuente* apoyándose en una gestión eficiente de datos maestros compartidos en múltiples registros, de forma que faciliten el enlazamiento de los diferentes orígenes de datos. Dentro de este plan se han marcado objetivos que potencien tanto el uso de registros administrativos y fuentes de datos con el fin de mejorar la eficacia, disminuir los costes, reducir progresivamente la carga de encuestas a los usuarios y aumentar la oportunidad del dato, disponiendo en muchas ocasiones de indicadores en un tiempo menor. El gran volumen de datos manejado y su continuo crecimiento exige el uso de tecnologías Big Data que garanticen la eficacia y el rendimiento de la solución que se proponga.

1.1 Gestión de Datos Maestros (MDM)

Se trata de la tecnología que permite gestionar conjuntos de datos maestros. Los datos maestros permiten determinar un único elemento de referencia y facilita la construcción de las tablas que registran las relaciones entre los diferentes registros.

El conjunto de datos maestros que se contemplan en el iDatos son, direcciones, edificios y viviendas, población y hogares, y por último empresas y establecimientos.

1.2 Estado del arte

La solución actual almacena las relaciones en una base de datos relacional, sin embargo, en los últimos años han proliferado en diferentes contextos las bases de datos orientadas a grafos en las que las relaciones constituyen el elemento crucial en el modelo de datos. El beneficio del almacenamiento nativo de grafos viene dado por la infraestructura de distribución de los datos que se diseña y construye especialmente para tener un buen rendimiento y una alta escalabilidad en el tratamiento de los modelos de grafos, idóneos para la representación de las relaciones. Frente a las bases de datos relacionales y otras soluciones

NoSQL, cuando se pretende explotar las relaciones entre datos masivos relacionados hay un aumento evidente de rendimiento. En las bases de datos relacionales el rendimiento de las consultas en el que intervienen operaciones JOIN decae a medida que el conjunto de datos crece, sin embargo, en bases de datos orientadas a grafos tiende a ser constante, ya que se limitan al subgrafo alcanzable desde el nodo.

Este Trabajo de Fin de Grado se ha ocupado del análisis del problema y diseño del esquema de una Base de Datos Orientada a Grafos que de soporte al sistema iDatos.

Capítulo 2 ORGANIZACIÓN Y GESTIÓN DE LA INFORMACIÓN DEL SISTEMA

2.1 LOS ESQUEMAS-TIPO DEL ENTORNO REPOSITORIO DEL BANCO DE DATOS

El procesamiento supervisado por lotes es la arquitectura que el ISTAC usa para el banco de datos, el procesamiento por lotes es un método de ejecución de tareas de datos repetitivas y de gran volumen. Este método permite a los usuarios procesar datos cuando se disponga de recursos informáticos y con poca o nula interacción del usuario.

En el procesamiento por lotes, existen varios tipos de entornos, en iDatos se trabaja con el entorno repositorio. El entorno repositorio se organiza en función de los diferentes tipos de archivos que contiene, en este caso se cuenta con tres tipos de niveles diferentes, cartografías, microdatos o macrodatos.

Dentro de cada nivel podemos observar diferentes tipos de datos.
En las *cartografías* se encuentran los tipos de datos, Raw cartography (RC) , Support Cartography (IGS) y Geography information reference (IGR).

En los *microdatos*, se encuentran los Raw Data(RD), los Master Data(ID), los Statistical Data(SD), los Scientific Data(CD), y los Public Data(PD).

Finalmente, en los *macrodatos* los MacroDataSet (MDS), DataSetCube(DSC) y los IndicatorsCube (DSI).

El Sistema de Datos Integrados trabajará con los siguientes tipos de datos:

- Raw Cartography, “Esquemas de cartografía en crudo, con sólo tratamiento de normalización”. [2]

- Support Cartography, “Esquemas de cartografía de soporte. En estos esquemas se almacenan cartografías normalizadas y depuradas, utilizadas para el procesamiento y análisis espacial.” [2]

- Geography Information, “Esquemas de cartografía de referencia. En estos esquemas se almacenan cartografías normalizadas, transformadas y depuradas, para su uso como entidades geográficas de difusión estadística. En este caso se almacenan distintas geometrías e información auxiliar”. [2]

- Raw Data, “Esquemas de microdatos en crudo, con solo tratamiento de normalización. Todo fichero se georreferencia desde su incorporación al Banco de Datos.”[2]

- Master Data, “Esquemas de microdatos maestros, donde se almacenan los cuatro grandes directorios (direcciones, edificios y viviendas, población y hogares, empresas y establecimientos). En el caso que nos ocupa es especialmente relevante el Directorio de Calles y Portales, donde se almacena la información base para la georreferenciación.” [2][1]

2.2. ORGANIZACIÓN DE LOS MICRODATOS PARA FACILITAR LA INTEGRACIÓN

El artículo 32 de la Ley 1/1991 indica que “El banco de datos administrativos para fines estadísticos debe facilitar la fusión de los ficheros para fines estadísticos”. Para cumplir con el artículo 32 de esta ley el ISTAC organiza los datos de la siguiente forma. Organiza los microdatos dentro del Banco de Datos de la Infraestructura de datos y metadatos en diferentes tipos de tablas, que a su vez se dividen en función de si son: *datos, metadatos o relaciones.*

[1] *2019, ISTAC, estadística de población activa registrada(EPA-reg), p.7*

Dentro del grupo de los *datos*, tenemos las tablas tipo Datos o DAT, que almacenan microdatos, y las tablas tipo Georreferencias o GEO que almacenan georreferencias.
El Banco de Datos organiza las tablas tipo DAT de la siguiente manera:
Para cada fila u observación se crea un identificador único universal que llamaremos *uuid*, un identificador de tabla o *stid*, un identificador único local o *luid* y la fecha en la que ha sido creado, que la denominaremos *marcatiempo*. Por tanto estos 4 identificadores los encontraremos en todas las observaciones de las tablas tipo que existan. Aparte de estos identificadores, cada tabla cuenta con identificadores propios únicos de ese tipo de tabla en concreto, pudiendo así acceder a datos

Las tablas DAT con las que contamos en el Banco de Datos son las siguientes y se dividen en 3 grupos, que llamaremos las tablas DAT, las tablas IDT y las tablas IDF:

- Tablas DAT de Afiliados: Almacenan los datos de los registros de afiliación de una persona física en una fecha específica del Banco de Datos del ISTAC.
- Tablas DAT de Cotización: Almacenan los datos de los registros de cotización de una empresa en una fecha específica del Banco de Datos del ISTAC.
- Tablas DAT de Demandantes: Almacenan los datos obtenidos en una fecha específica de la

demanda de empleo de una persona física del Banco de Datos del ISTAC.

- Tablas DAT del Padrón: Almacenan los datos obtenidos en una fecha específica del padrón de una persona física del Banco de Datos del ISTAC.
- Tablas IDT e IDF de cada uno de los directorios que forman parte del sistema de iDatos: El fichero IDT del directorio contiene las variables nucleares para poder identificar al representante único de la entidad, mientras que el IDF contiene cada una de las diferentes versiones de la entidad en todos los registros administrativos que intervienen en una estadística.

Dentro del grupo de los metadatos:

- Diseño de registro, almacena diseños de los registros del conjunto de las tablas. Estas tablas nos ayudan a identificar el valor de los códigos utilizados en las tablas tipo DAT. En la figura 1 se muestra un ejemplo de tabla DSD, en este caso de la tabla Alifiados, donde nos muestran los códigos de una variable y su correspondiente valor o nombre.

code	parent	variable_eleme	name#es
01			Ingenieros y licenciados
02			Ingenieros técnicos, peritos y ayudantes titulados
03			Jefes administrativos y de taller
04			Ayudantes no titulados
05			Oficiales administrativos
06			Subalternos
07			Auxiliares administrativos
08			Oficiales de primera y segunda
09			Oficiales de tercera y especialistas
10			Peones
11			Trabajadores menores de dieciocho años
13			Sin cotización
15			Oficiales de primera y segunda. Opción mensual
16			Oficiales de tercera y especialistas. Opción mensual
17			Mayores 18 años no cualificados. Opción mensual
18			Trabajadores menores de 18 años. Opción mensual
_N			No responde
_O			Otros
_U			No sabe / dato desconocido
_X			Dato fuera de rango
_Z			No aplicable / no procede

figura 1. ejemplo de tabla DSD

- Registro de datos, son tablas que contienen los registro de los datos y la relación que los une con el diseño de registro
- Extensiones de códigos, almacenan los codelist de las tablas de microdatos.

Y por último en el grupo de las relaciones tenemos las relaciones entre unidades de información o URD, que almacenan relaciones de microdatos con otros microdatos.

Las tablas de relaciones que se almacenan en las tablas URD, nos permiten elaborar estadísticas multifuentes,

almacenando para cada fichero de datos el conjunto de relaciones que se establezcan con otros ficheros de datos.

En las tablas URD, se almacenan las relaciones entre dos ficheros de datos, uno origen (DAT_A) y otro destino (DAT_B). La tabla URD por lo tanto necesitará de identificadores que diferencie los ficheros de origen y los de destino, por ello precisa de:

Un UUID_A y un UUID_B, así como un STID_A y un STID_B que son tanto al identificador único universal o uuid como el identificador de esquema de tabla o stid de los correspondientes DAT_A y DAT_B.

A parte la tabla URD, al igual que estaban organizadas las tablas DAT, cuenta con su propio identificador único universal *uuid* un identificador de tabla o *stid* y la fecha en la que ha sido creado *marcatiempo.*

UUID	STID	Marcatiempo	UUID_A	STID_A	UUID_B	STID_B
URD.uuid	URD.stid	URD.marcatiempo	DAT_A.uuid	DAT_A.stid	DAT_B.uuid	DAT_B.stid

Figura 2.Ejemplo tabla URD

La figura 2, es un ejemplo de cómo está organizado un archivo URD, siendo la primera fila la cabecera del archivo y la segunda fila el registro de las relaciones existentes entre dos ficheros de datos, en la cabecera también se encuentran otras variables como el tipo de relación y la descripción de ésta

Los siguientes identificadores de la tabla URD nos hacen una descripción de las relaciones.

El primer identificador que encontramos se trata del tipo de relación que existe(REL_TYPE). En las tablas URD existen dos tipos de relaciones, las relaciones de tipo SOURCE y las relaciones de tipo TRACESTOCK. Los siguientes identificadores serían REL_PERIOD, REL_EVENT, REL_DESCRIPTION, LINK_QUALITY, LINK_QUALITY_RANK, LINK_ACTIVE, LINK_ACTIVE_IN, LINK_ACTIVE_OUT. Estos identificadores se añadirán como propiedades de las relaciones.

Las relaciones de tipo SOURCE se dan cuando se relacionan tablas de tipo IDT con tablas tipo IDF. Mientras que las tablas TRACESTOCK se dan cuando se relacionan tablas IDT con tablas DAT de cualquiera de las existentes.

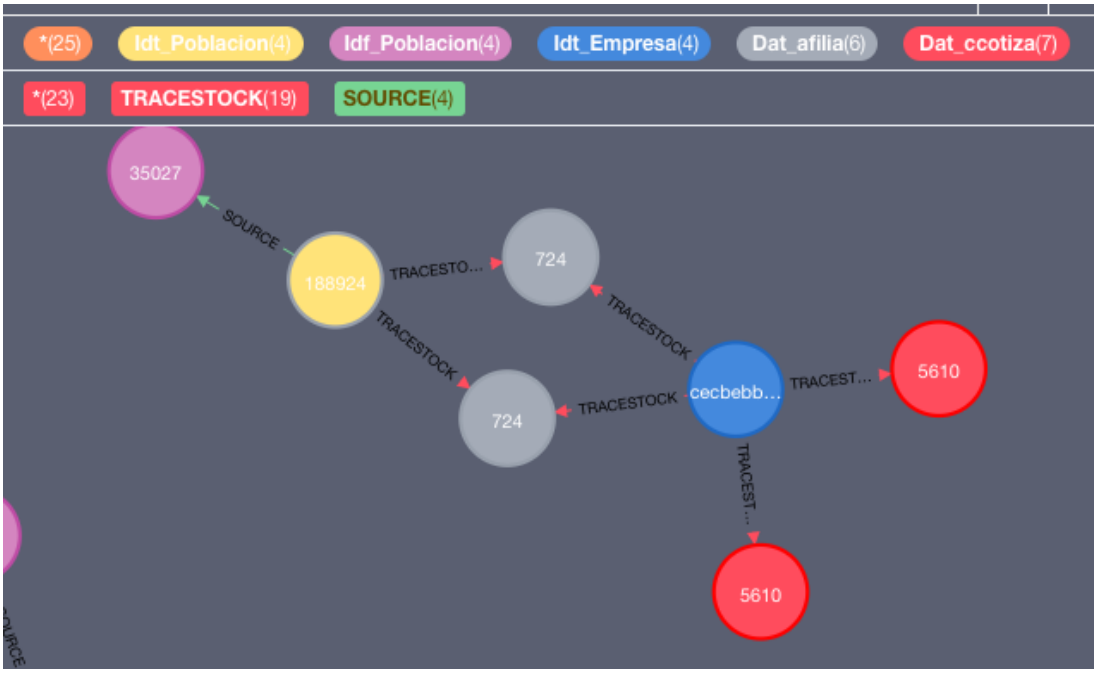

Figura 3 Ejemplo en Neo4j de relaciones.

En la figura 3, se puede observar un ejemplo de relaciones existentes, en primer lugar en color rosado tenemos un IDF que tiene una relación SOURCE con un IDT del mismo directorio, hablaremos en el capítulo siguiente de los tipos de directorios que existen. Éste directorio a su vez tiene relación TRACE-STOCK con dos DAT de Afiliados. Aparte podemos observar como otro directorio diferente al anterior, también tiene relación TRACE-STOCK con los dos nodos DAT de Afiliados, esto es lo que permite que exista en el Banco de Datos una conexión entre los diferentes IDT de cada directorio, añadiendo profundidad al grafo.

2.3 DIRECTORIOS EN IDATOS

El esquema de datos maestros o ID está formado por un conjunto de 4 directorios que forman parte del Sistema de Datos Integrados.

1. Lugares
2. Edificios, viviendas y locales(DEV)
3. Población y hogares(DPH)
4. Empresas y establecimientos(DUE)

Para producir estos directorios el ISTAC se basa en varios conceptos.

1. Directorio: Colección de registros conceptualmente relacionados
2. Registro: Colección de ficheros relacionados con una de unidad de análisis

3. Capas: Colección de ficheros de un registro según funcionalidad de integración
4. Variables nucleares: Variables clave de un registro
5. Variables normalizadas: Variables que no siendo claves se quieren normalizar en iDatos
6. Variables de entidades relacionadas: Identificadores que vinculan unidades de un registro con las de otro registro

Los directorios se dividen en 4 bloques, y cada uno consta de un directorio principal y varios subdirectorios relacionados.

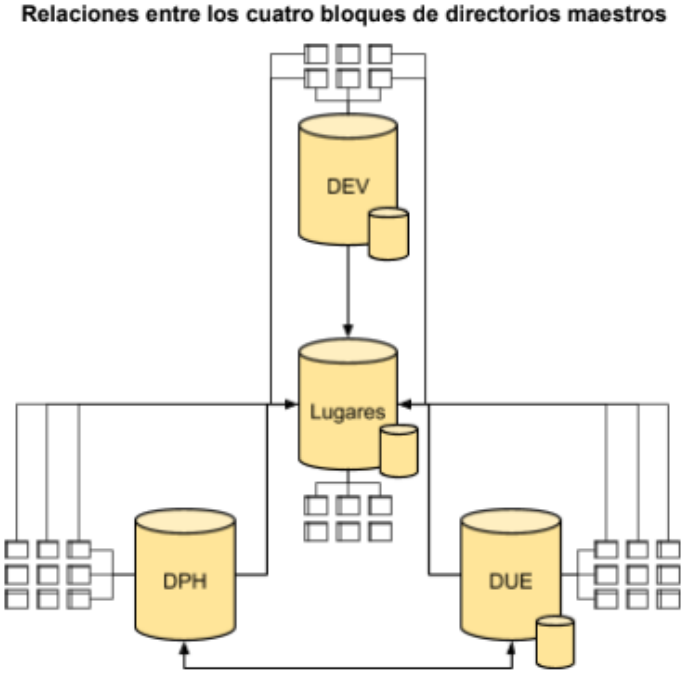

Figura 4. (2019, ISTAC, estadística de población activa registrada(EPA-reg), p.17) [1]

Y dentro de cada directorio nos encontramos los registros

comentados anteriormente. En este TFG se ha trabajado con los siguientes registros:

1. Población (Población y Hogares, DPH)
2. Portales (Lugares)
3. Empresa (Empresa y establecimientos, DUE)

2.3.1 DIRECTORIOS DE UNIDADES ECONÓMICAS: REGISTRO EMPRESAS

El directorio de unidades económicas contiene el registro de referencia tanto de las empresas como los autónomos. En el IDT del registro empresa podemos encontrarnos las variables nucleares, siendo además variables de entidad relacionadas, en cuanto a los ficheros IDF, obtenemos diferentes versiones de todos los registros distintos de cada empresa única.

A continuación, podemos observar el IDF de una empresa con algunas de las variables comunes de ese fichero.

UUID	STID	MARCATIEMPO	EMPRESA_PERSONA_FISICA
003f7f92-2265-4de5-9be4-b30c4b3faf45	c00021a_id.idf_empresas	20FEB2020	6
007ee1dd-ccca-4e01-9ff7-58e6a1bc7672	c00021a_id.idf_empresas	20FEB2020	6
00928afe-e133-4c52-8362-639a89fccec5	c00021a_id.idf_empresas	20FEB2020	6
00a8c26b-a0f7-4688-a2a3-0bbc84f1d8c9	c00021a_id.idf_empresas	26FEB2020	6
01375e02-b8ff-4ff7-a940-b26e54647192	c00021a_id.idf_empresas	21FEB2020	6
014b2687-3fd0-4eea-8405-e0ed464f46eb	c00021a_id.idf_empresas	26FEB2020	6
022c803c-b4ae-4d92-b08a-73d02168b7c7	c00021a_id.idf_empresas	20FEB2020	6
0259230a-e013-4aff-834e-cd3d146babac	c00021a_id.idf_empresas	20FEB2020	6
026267be-8377-41d7-bef3-cdf8a3717ff4	c00021a_id.idf_empresas	21FEB2020	6
026a6138-9bdc-4a13-9203-8065a6745355	c00021a_id.idf_empresas	20FEB2020	6
0283609f-ba71-49ac-8e24-41c2b327d736	c00021a_id.idf_empresas	20FEB2020	1
02b780f1-e3ec-4cc1-8770-8455818e2071	c00021a_id.idf_empresas	20FEB2020	1
031eefa2-b48e-4ed4-bff1-f85fb807f97d	c00021a_id.idf_empresas	26FEB2020	1

Figura 5. IDF Empresa

2.3.2 DIRECTORIO DE CALLES Y PORTALES: REGISTRO PORTALES

El directorio de calles y portales registra puntos en el espacio que representan los portales únicos existentes. De guardar estos puntos se encarga el fichero IDT, mientras que el IDF del registro de portales guarda las diferentes versiones de cada portal único, es decir, las diferentes formas de llamar a una dirección en concreto, o por ejemplo el cambio de nombre de ese punto a lo largo del tiempo.

CONCEPT_ID	TECH_TYPE	TECH_SIZE	LABEL	CODELIST
uuid	varchar	36	Identificador Único Universal	
luid	serial		Identificador Único Local	
stid	varchar	61	Identificador de esquema y tabla	
marcatiempo	date		Sello de tiempo de creación de la observación	
tvia_nn	varchar	13	Tipo de vía no normalizado	
nvia_nn	varchar	50	Nombre de vía no normalizado	
numer_nn	varchar	7	Número de portal no normalizado	
tvia	varchar	13	Tipo de vía	
cvia	varchar	5	Código de vía	
nvia	varchar	50	Nombre de vía	
numer	varchar	4	Número de portal	
kmt	**varchar**	**3**	**Punto kilométrico**	
nomedif	**varchar**	**50**	**Nombre del edificio**	
codmun	varchar	5	Código de municipio	ISTAC:CL_AREA_ES(02.000)
nommun	varchar	35	Nombre del municipio	ISTAC:CL_AREA_ES(02:000)
direccion	varchar	255	Dirección (tvia+nvia+numer+nommun)	

Figura 6. IDF portal (2019, ISTAC, estadística de población activa registrada(EPA-reg), p.9)

2.3.3 DIRECTORIO DE POBLACIÓN Y HOGARES: REGISTRO POBLACIÓN

UUID	STID	MARCATIEMPO	NOMEPER_TIPO_IPF	NOMEPER_FNAC	NOMEPER_SEXO	NOMEPER_CODMUNNAC	NOMEPER_PAISNAC
000811e2-fd35-485a-82c9-4e42a25438e7	c00063a_id.idf_poblacion	19JUN2019	1	07OCT1957	2	_U	_U
000b7f3f-bdc2-4a29-af3b-8b8f3fbe4220	c00063a_id.idf_poblacion	19JUN2019	1	08AUG1983	2	35009	724
0011a03f-4dbf-4f1e-98ce-4507df6632c2	c00063a_id.idf_poblacion	17JAN2020	1	26MAR1966	2	35016	724
0017c7ff-6591-4079-8279-9d407deead40	c00063a_id.idf_poblacion	06NOV2019	1	02MAY1968	2	35034	724
00301239-d1aa-4e60-aa56-4b8c60c6690f	c00063a_id.idf_poblacion	08AUG2019	1	03MAY1985	2	_U	_U
00401700-a8fb-4de7-b9bc-54f96aeaa17c	c00063a_id.idf_poblacion	19JUN2019	1	12MAY1979	2	28079	724
0059b1f0-ad10-4825-9532-62f8390b6497	c00063a_id.idf_poblacion	19JUN2019	1	27JAN1970	2	35006	724
0064d334-61ea-41db-bd8a-6e587063d14f	c00063a_id.idf_poblacion	19JUN2019	1	08AUG1968	1	35016	724
0070a1f4-5c10-4e1a-a4a4-6b5329b47768	c00063a_id.idf_poblacion	19JUN2019	1	21APR1965	2	35016	724
00816b41-3b7d-4259-abf2-8ec5e0ec57ea	c00063a_id.idf_poblacion	19JUN2019	1	01AUG1989	2	35009	724
0081f0b3-fca2-40d4-925f-ff73c98b10c4	c00063a_id.idf_poblacion	19JUN2019	1	08NOV1992	2	_Z	170
008a2007-8e4b-4d07-839b-3a5791e71af4	c00063a_id.idf_poblacion	19JUN2019	1	18SEP1965	2	35023	724
00a55d3f-a9f8-47ba-9b54-db9a0f9a315b	c00063a_id.idf_poblacion	19JUN2019	1	08FEB1985	2	35019	724

Figura 7. IDF Población

El directorio de población registra a una persona física, en el archivo IDT, las variables nucleares que identifican a esa persona física como un individuo único. El archivo IDF se encarga de evitar que ocurran problemas de duplicidad, creando las diferentes versiones existentes de la persona. Los problemas de duplicidad surgen cuando un individuo tiene varios registros, pero con diferentes valores de las propiedades, creando así una duplicidad de ese individuo, por ejemplo, un problema muy común son los apellidos que contienen un ‘del’ o ‘de’ o nombres compuestos.

Capítulo 3 BASES DE DATOS ORIENTADA A GRAFOS

3.1 ¿Qué es una base de datos orientada a grafos?

Las bases de datos orientadas a grafos se caracterizan como su nombre indica por el uso de grafos que nos permiten representar los datos interconectados de forma visual y comprensible. El grafo está formado por un conjunto de vértices o nodos, que son nuestros datos u objetos que contienen la información, y arcos o aristas que nos permiten comprender las relaciones existentes entre los diferentes nodos.

Los principales aspectos positivos de las bases de datos orientadas a grafos se pueden resumir en los siguientes puntos:

- Buen rendimiento cuando existe un crecimiento exponencial del volumen de datos, por lo que las bases de grafos son capaces de adaptarse a las exigencias estructurales que van surgiendo
- Flexibilidad, ofreciendo diferentes métodos analíticos

3.2 TECNOLOGÍAS USADAS

3.2.1 NEO4j

Neo4J [1] es una de las bases de datos orientada a grafos (BDOG) más conocidas del mercado, se trata de una tecnología implementada en java.

El desarrollador de eBay Volker Pacher dijo lo siguiente sobre el rendimiento de uso de neo4j en su plataforma, “Nuestra solución Neo4j es literalmente mil veces más rápida que la solución anterior MySQL, con búsquedas que requieren entre 10 y 100 veces menos código”.

Una de sus otras ventajas más importantes es su agilidad a la hora de gestionar los datos, si llegáramos al límite de capacidad de la aplicación, el volumen total de datos tendría que superar los 34.000 millones de nodos y relaciones. Y por último su flexibilidad, debido a su alta capacidad de añadir nodos y relaciones a grafos creados con anterioridad.

Los principales casos de uso de Neo4J, son detección de fraudes, recomendaciones en tiempo real y redes sociales, gestión de centros de datos, y por último y la que nosotros hemos utilizado para este tipo de proyecto, la gestión de sistemas de datos maestros.

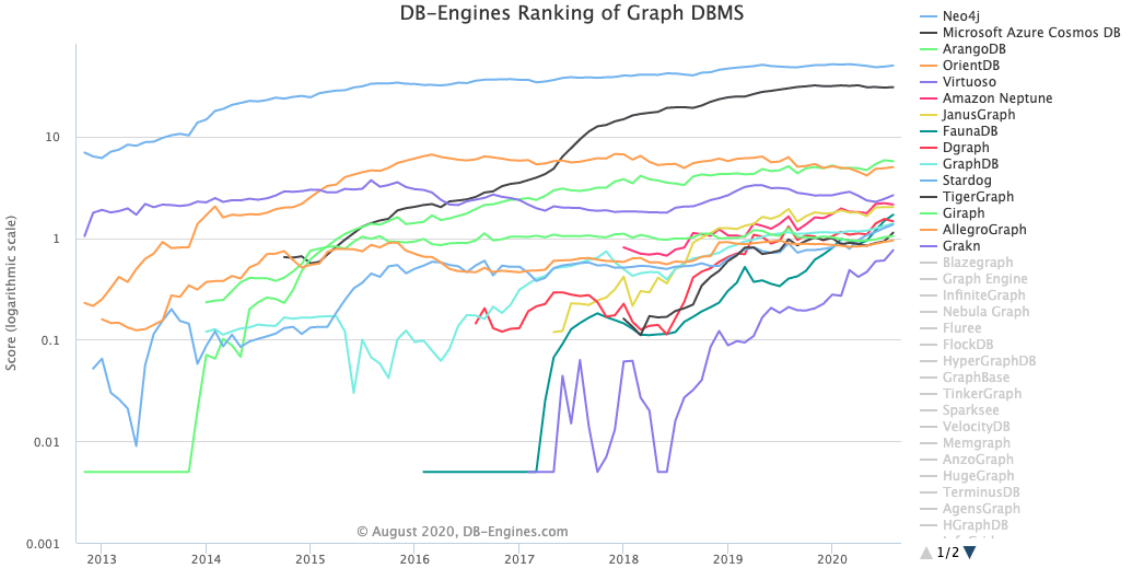

Figura 8. Gráficos en base al tiempo de base de datos orientada a grafos.[2] *[4]*

Por qué elegimos NEO4j frente a otras tecnologías tales como Amazon Neptune, SAP hana graph o Orient DB que son las bases de datos más conocidas. En primer lugar, porque Neo4J es una de las principales bases de datos a nivel mundial, empresas importantes como eBay, Walmart, Lufthansa, CISCO o UBS confían en sus servicios, además se trata de una tecnología open source.

[2] https://db-engines.com/en/ranking_trend/graph+dbms

☐ include secondary database models — 32 systems in ranking, August 2020

Rank Aug 2020	Rank Jul 2020	Rank Aug 2019	DBMS	Database Model	Score Aug 2020	Score Jul 2020	Score Aug 2019
1.	1.	1.	Neo4j	Graph	50.18	+1.26	+1.79
2.	2.	2.	Microsoft Azure Cosmos DB	Multi-model	30.73	+0.32	+0.79
3.	3.	↑ 4.	ArangoDB	Multi-model	5.73	-0.11	+0.61
4.	4.	↓ 3.	OrientDB	Multi-model	5.02	+0.14	-1.27
5.	5.	5.	Virtuoso	Multi-model	2.65	+0.21	-0.41
6.	6.	↑ 7.	Amazon Neptune	Multi-model	2.15	-0.06	+0.51
7.	7.	↓ 6.	JanusGraph	Graph	2.02	+0.00	+0.08
8.	↑ 9.	↑ 18.	FaunaDB	Multi-model	1.70	+0.22	+1.30
9.	↓ 8.	↓ 8.	Dgraph	Graph	1.47	-0.08	+0.16
10.	10.	10.	GraphDB	Multi-model	1.39	+0.07	+0.25
11.	11.	↑ 12.	Stardog	Multi-model	1.36	+0.10	+0.47
12.	↑ 13.	↓ 11.	TigerGraph	Graph	1.13	+0.20	+0.16
13.	↓ 12.	↓ 9.	Giraph	Graph	1.05	+0.03	-0.21
14.	14.	↓ 13.	AllegroGraph	Multi-model	0.95	+0.03	+0.06
15.	↑ 17.	↑ 22.	Grakn	Multi-model	0.76	+0.16	+0.55
16.	↓ 15.	↓ 14.	Blazegraph	Multi-model	0.74	+0.05	+0.05
17.	↓ 16.	↓ 15.	Graph Engine	Multi-model	0.61	0.00	+0.06
18.	18.	↓ 17.	InfiniteGraph	Graph	0.43	+0.01	+0.02
19.	↑ 20.		Nebula Graph	Graph	0.35	+0.04	
20.	↓ 19.	↑ 32.	Fluree	Graph	0.33	+0.01	+0.33
21.	21.	↓ 19.	FlockDB	Graph	0.29	+0.01	+0.02

Figura 9. Ranking base de datos orientada a grafos.[4] [3]

Se trata también de un programa muy intuitivo a la hora de empezar a trabajar con bases de datos orientada a grafos.

3.2.2 CYPHER

Neo4j usa su propio lenguaje, cypher, es un tipo de lenguaje declarativo, fue diseñado con la mente en el lenguaje SQL, pero teniendo en cuenta los componentes y las necesidades de una base de datos orientada grafos, para así simplificar el trabajo y que sea un lenguaje más intuitivo. [3]

[3] https://db-engines.com/en/ranking_trend/graph+dbms

Estructura

Cypher usa cláusulas, al igual que SQL, las query (sentencias) se realizan construyendo con varias cláusulas, las cláusulas son condiciones de modificación, que realizan funciones con los datos que se desean manipular.

Por ejemplo si quisiéramos para crear un grafo simple donde hay personas que tienen amigos, se realizará con las siguientes sentencias:

```
CREATE (john:Person {name: 'John'})
CREATE (joe:Person {name: 'Joe'})
CREATE (steve:Person {name: 'Steve'})
CREATE (sara:Person {name: 'Sara'})
CREATE (maria:Person {name: 'Maria'})
CREATE (john)-[:FRIEND]->(joe)-[:FRIEND]->(steve)
CREATE (john)-[:FRIEND]->(sara)-[:FRIEND]->(maria)
```

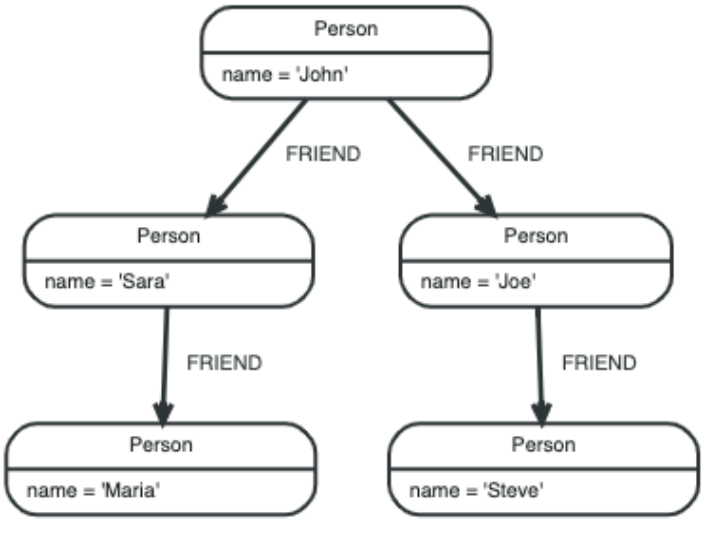

Figura 10. Ejemplo de estructura. [5] [4]

En esta consulta encuentra un usuario llamado 'John' y amigos de 'John' (aunque no sus amigos directos) antes de devolver tanto a 'John' como a los amigos de amigos que se encuentren.

```
MATCH (john {name: 'John'})-[:FRIEND]->()-[:FRIEND]->(fof)
RETURN john.name, fof.name
```

Resulting in:

```
+----------------------+
| john.name | fof.name |
+----------------------+
| "John"    | "Maria"  |
| "John"    | "Steve"  |
+----------------------+
2 rows
```

[4] https://neo4j.com/developer/cypher/intro-cypher/

Capítulo 4 IMPLEMENTACIÓN

4.1 DISEÑO DE LA BASE DE DATOS ORIENTADA A GRAFOS

Como observamos en el capítulo 2 las tablas URD relacionan tablas tipo DAT con otras tablas tipo DAT, para realizar el diseño estudiamos las relaciones existentes en estas tablas URD de cada directorio, es decir la tabla URD del directorio empresa, del directorio portales y directorio población. El ISTAC clasifica las relaciones en dos tipos, las relaciones de tipo SOURCE que se dan cuando se relaciona una tabla IDT de un directorio y una tabla IDF del mismo directorio, y las relaciones TRACE-STOCK que se dan únicamente cuando un IDT es relacionado con un DAT de los existente ya sea afiliados, demandantes, padrón o cotización.

Tipos de relaciones TRACE-STOCK que encontramos en el Banco de Datos:

- IDT del directorio población, tiene relación con las tablas DAT de afiliado, padrón y de demandante.
- IDT del directorio empresa, tiene relación con las tablas DAT de afiliado y cotización
- IDT del directorio portales, tiene relación con la tabla

DAT padrón.

Esto hace que se nos quede un grafo como el de la figura 11 que veremos a continuación.

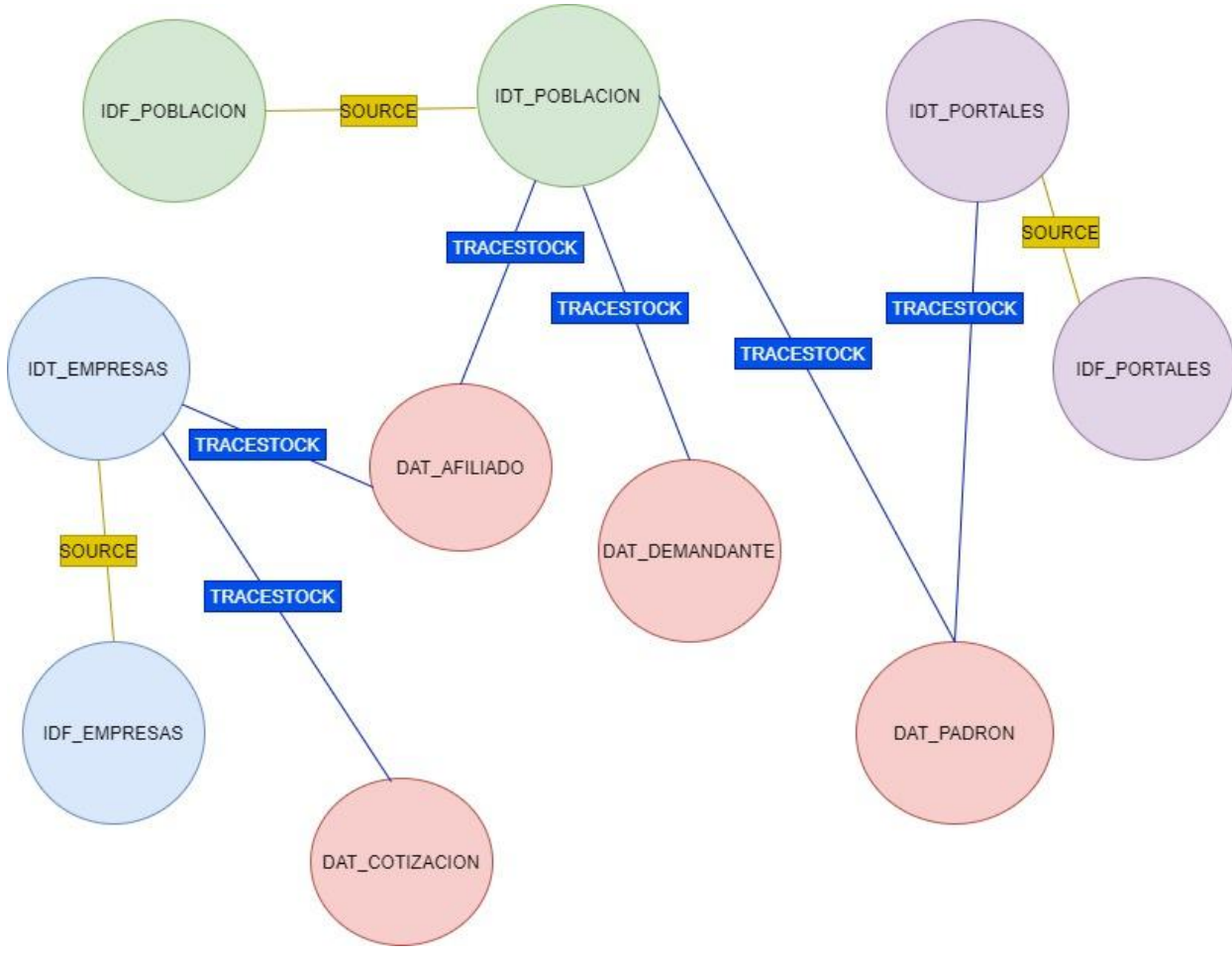

Figura 11. Organización de tipos de nodos.

Por lo tanto, hemos terminado creando un grafo de 10 tipos de nodos, 3 IDT, 3 IDF, uno por cada directorio, más los 4 nodos DAT existentes.

Una vez diseñada la base de datos, nos encontramos con un problema, a la hora de trabajar con el DAT de demandantes. El DAT de demandantes cuando se empezó a estudiar, se observó que se trataba de un archivo con un volumen de atributos muy grande, ya que la base de datos con la que se trabaja no está normalizada, provocando que diferentes valores de un atributo para un mismo registro estén incluidos como atributos. Este volumen de ocurrencias de un atributo en el DAT de demandantes hace que sea un archivo complejo y de difícil manejo a la hora de trabajar con él. Por ello se diseñaron conjuntos de datos como subnodos del DAT de demandantes, donde contarán con el uuid, stid y marcatiempo del nodo padre, y además con todas las ocurrencias encontradas de ese atributo. Los subnodos creados a partir del DAT de demandantes son los siguientes:

- Discapacidad, engloba los códigos de discapacidad con tres ocurrencias en el archivo.
- Formación ocupacional, este dato se repite cinco veces, con distintos códigos, que van desde el identificador único del registro de la formación ocupacional, hasta por ejemplo el tipo de formación.
- Idiomas, se repite hasta en ocho ocasiones, contiene códigos con los idiomas que habla la persona física, el nivel de lectura, nivel de escritura, etc…
- Formación Complementaria, obtenemos hasta 10 ocurrencias de este dato, tales como nivel formativo, áreas de conocimiento, etc…

- Colectivo Especial, también obtenemos diez ocurrencias de este dato, se tratan de los posibles códigos del colectivo especial, así como la fecha de adjudicación a ese colectivo, etc…
- Ámbito de búsqueda, uno de los datos con mayor índice de ocurrencia, ya que encontramos hasta veinticinco, los códigos tratan sobre la localización de esa búsqueda, país, ciudad, municipio, etc…

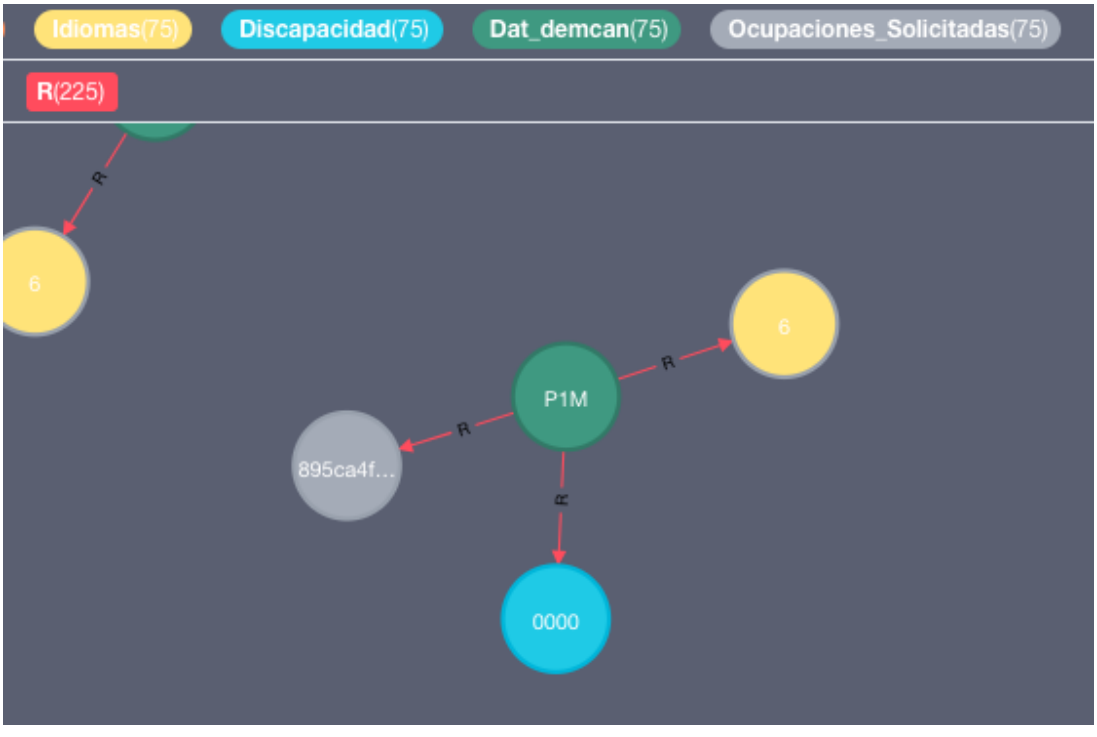

Figura 12 Ejemplo en neo4j de relaciones R entre subnodos y nodo padre DAT de Demandantes.

En la figura 12 podemos observar un ejemplo en Neo4j de cómo el nodo DAT de Demandantes (Dat_demcan) en este

caso en color verde tiene, tres relaciones R, con el subnodo de tipo Idiomas de color amarillo, el de tipo Discapacidad de color azul y el de tipo Ocupaciones Solicitadas de color gris, esta nueva relación de tipo R, es creada para poder unir el nodo padre con los seis subnodos que existen.

4.2 ESTRUCTURA INICIAL DE LOS DATOS

Los datos se nos entregaron en forma de ficheros tipo ***CSV***, para así poder realizar una importación sencilla, nos entregaron una carpeta con un subconjunto para pruebas de ficheros de tipo *DAT con datos de* afiliados, demandantes, padrón y cotización, en concreto, datos de afiliados, del padrón y de demandantes de distintos años, y otra carpeta con las muestras de cada registro (empresa, población y portales), con tres archivos distintos dentro de esta carpeta, el archivo *IDT*, el *IDF* y el *URD*.

Todos estos archivos vienen con una cabecera, que representa los atributos de cada uno de los valores separados por “;”, y cada fila, a partir de la cabecera, se trata de las observaciones.

4.3 MIGRACIÓN DE DATOS (QUERYS)

Para aumentar el rendimiento de la base de datos se recomienda crear índices y declarar las claves únicas de los nodos. Por ello en primer lugar creamos la clave única que

automáticamente creará un índice sobre esa clave.

La sentencia utilizada es las siguiente:

```
CREATE CONSTRAINT ON (n:Idt_Poblacion) ASSERT
n.uuid IS UNIQUE;
```

En la sentencia anterior, se crea un índice para el fichero IDT del directorio población donde la clave única, es el identificador único universal uuid.

Una vez cargado los índices y las claves únicas, nuestro siguiente paso, será cargar la base de datos de nodos, para ello, importamos de nuestra carpeta '*import*' en la aplicación de neo4j el archivo CSV, que queremos importar. Usamos el comando MERGE, que comprueba si ese tipo de nodo que se quiere crear ya existe, en cuyo caso no lo vuelve a crear.

```
LOAD CSV WITH HEADERS FROM
"file:///data/poblacion/IDT_SELECT_TFG.csv" AS line
FIELDTERMINATOR ";"
MERGE(:Idt_Poblacion{uuid:line.UUID_IDT,
stid:line.STID_IDT})
```

Con la primera línea indicamos que vamos a cargar un archivo CSV con cabecera y seguidamente le indicamos la ruta donde se encuentra ese archivo, una vez determinada la ruta, indicamos el tipo de delimitador que existe en la cabecera CSV.

Finalmente, le indicamos que en base a ese archivo cree los nodos con el comando MERGE comentado anteriormente, en este ejemplo en concreto del archivo IDT del directorio población, junto a las propiedades del nodo, en este caso UUID y STID.

Estas sentencias se realizan de igual forma para los directorios de población, empresa y portales. Para poder acceder a todos los registros con las sentencias si fuera necesario, se ha incluido un anexo con todas ellas para que cualquier persona pueda si le interesara usarlas o verla

4.2 CONSULTAS

A continuación se muestran algunas consultas realizadas en neo4j usando cypher.

1- Personas discapacitadas que saben al menos dos idioma

```
MATCH
(ID:Idiomas)-[:R]-(DATDEMCAN:Dat_demcan)-[:R]-(DISC:Discapacidad)
MATCH (IDTPB:Idt_Poblacion)-[:TRACESTOCK]-(DATDEMCAN)
WHERE DISC.dem029_disc01<>'0000' OR DISC.dem029_disc02<>'0000'
OR DISC.dem029_disc03<>'0000' AND ID.dem018_idiom01<>''
RETURN ID, DISC,IDTPB;
```

En este ejemplo primero realizamos la cláusula MATCH [7], que nos permite especificar el patrón para buscar en la base de datos, en este caso le especificamos las relaciones de tipo R entre el nodo DATDEMCAN, es decir la tabla tipo DAT del directorio de demandantes, juntos a los dos subnodos IDIOMAS y DISCAPACIDAD, una vez especificado ese patrón realizamos otro buscando las relaciones TRACESTOCK entre ese nodo DATDEMCAN anterior y el nodo IDT del directorio población, obteniendo una lista de las personas físicas que están demandando empleo. Para determinar dentro de esa lista las personas que tienen una discapacidad y que saben al menos dos idiomas, es decir el idioma nativo y al menos un idioma más usamos la cláusula WHERE [6] para especificar lo que buscamos en el patrón escrito anteriormente en las

cláusulas MATCH. En este caso indicamos que los códigos de los idiomas han de ser distintos de ‘’ ya que indicarían que esa persona demandante sabe algún idioma y además que tenga código de discapacidad distinto de ‘0000’ ya que indicaría que esa persona cuenta con algún tipo de discapacidad

2- Listado de empresas que han aumentado su número de empleados entre junio y septiembre de 2017.

```
MATCH
(IdtEm:Idt_Empresa)-[:TRACESTOCK]-(Datcc:Dat_ccotiza{stid:'rd_cotizacion_segsoc.dat_ccotiza20170630_v01'}),
(IdtEm)-[:TRACESTOCK]-(Datcc09:Dat_ccotiza{stid:'rd_cotizacion_segsoc.dat_ccotiza20170930_v01'}) WHERE toInteger(Datcc.ccotiza_asalariados)< toInteger(Datcc09.ccotiza_asalariados) RETURN IdtEm, Datcc, Datcc09
```

En este segundo ejemplo, volvemos a usar la cláusula MATCH para obtener un patrón, en este caso para buscar los IDT del directorio empresa con relación TRACESTOCK con el directorio DAT de cotización en una fecha, en este caso en junio de 2017, y otro patrón pero con fecha en septiembre de 2017, una vez realizado la búsqueda de este patrón especificamos que busque solo donde el atributo Datcc_ccotiza_asalariados que determina el número de empleados que hay, sea menor en junio que en septiembre.

Capítulo 5
Conclusiones y líneas futuras

La bases de datos orientada a grafos diseñada a partir de los datos obtenidos a través del ISTAC nos ha ayudado a la hora de obtener una base de datos más visible y de fácil comprensión que la que tendríamos con una base de datos convencional, ayudándonos así a poder analizar de forma más clara los problemas que nos iban surgiendo y cómo encararlos.

A nivel social trabajar con grandes cantidades de datos es importante, con esta base de datos se pretende obtener datos del mercado del trabajo, evitando el uso de encuestas tediosas para la población, usando la información estrictamente necesaria, de forma más ágil y con una muestra de datos mucho más grande de la actual.

En cuanto al uso de las tecnologías, Neo4j se trata de una tecnología fácil e intuitiva para alguien que trabaja por primera vez con ella. Su sistema gráfico muestra de una forma clara los datos obtenidos y nos permite trabajar con ellos de forma sencilla.

En unas posibles líneas futuras se pretende :

- Estudiar el grafo resultante desarrollado en este trabajo y diseñar consultas que permitan comparar el rendimiento en una base de datos similar a la utilizada actualmente por el ISTAC.
- Aumentar el volumen de datos añadidos al Banco de Datos actual.
- Crear algoritmos de búsquedas.

Capítulo 6 Summary and Conclusions

The graph-oriented database designed from the data obtained through the ISTAC has helped us to obtain a more visible and easy-to-understand database than the one we would have with a conventional database, thus helping us to be able to analyze in a clearer way the problems that were arising and how to face them.

At the social level, working with large amounts of data is important, with this database the aim is to obtain labor market data, avoiding the use of tedious surveys for the population, using the information strictly necessary, in a more agile way and with a sample much larger data than today.

Regarding the use of technologies, Neo4j is an easy and intuitive technology for someone who is working with it for the first time. Its graphic system clearly shows the data obtained and allows us to work with them in a simple way.

In some possible future lines it is intended:

Study the resulting graph developed in this work and design queries that allow the performance to be compared in a database similar to the one currently used by ISTAC.

Increase the volume of data added to the current Database.

Create search algorithms.

Capítulo 7 Presupuesto

7.1 Sección Uno

Tipos	Descripción
Mano de obra	300 horas (2100€)
Material necesario	Pc windows 10 ram 8GB (700€)
Tecnologías usadas	neo4j, cypher (open source 0€) windows 10 (100 €)
Total	2900 €

Tabla 7.1: Resumen de tipos

ANEXO: CREACIÓN DE LA BASE DE DATOS ORIENTADA A GRAFOS EN NEO4J

1.CREACIÓN DE ÍNDICES

```
CREATE INDEX ON :Idf_Poblacion(uuid);

CREATE INDEX ON :Idt_Poblacion(uuid);

CREATE INDEX ON :Idf_Portales(uuid);

CREATE INDEX ON :Idt_Portales(uuid);

CREATE INDEX ON :Idf_Empresa(uuid);

CREATE INDEX ON :Idt_Empresa(uuid);

CREATE INDEX ON :Dat_afilia(uuid);

CREATE INDEX ON :Dat_pmh(uuid);

CREATE INDEX ON :Dat_ccotiza(uuid);

CREATE INDEX ON :Dat_demcan(uuid);
```

2.CARGANDO DATOS DAT

2.1. DATOS AFILIA

```
LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_AFILIA20170630_V01.csv" AS line
```

FIELDTERMINATOR ";"

CREATE(:Dat_afilia{uuid:line.uuid,luid:line.luid,stid:line.stid,marcatiempo:line.marcatiempo,afilia_sexo:line.afilia_sexo,afilia_fnac:line.afilia_fnac,afilia_naci:line.afilia_naci,afilia_domires_codmun:line.afilia_domires_codmun,afilia_domitrab_codmun:line.afilia_domitrab_codmun,afilia_rcotiza:line.afilia_rcotiza,afilia_falta:line.afilia_falta,afilia_asal_cccotiza:line.afilia_asal_ccotiza,afilia_asal_gcotiza:line.afilia_asal_gcotiza,afilia_asal_tcontrat:line.afilia_asal_tcontrat,afilia_asal_coeftpar:line.afilia_asal_coeftpar,afilia_asal_relaclab:line.afilia_asal_relaclab,afilia_auto_cnae09:line.afilia_auto_cnae09,afilia_auto_cobertat:line.afilia_auto_cobertat,afilia_auto_seta:line.afilia_auto_seta,afilia_auto_relacent:line.afilia_auto_relacent,afilia_auto_colegpro:l ine.afilia_auto_colegpro});

2.2. DATOS DEMANDANTES

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_DEMCAN20170630_V01.csv" AS line FIELDTERMINATOR ";"

CREATE(:Dat_demcan{uuid:line.uuid,stid:line.stid,luid:line.luid,dem_ref_period:line.dem_ref_period,dem_time_period:line.dem_time_period,dem_time_format:line.dem_time_format,dem004_sexo:line.dem004_sexo,dem005_fnac:line.dem005_fnac,dem006_edad:line.dem006_edad,dem007_naci:line.dem007_naci,dem008_demext:line.dem008_demext,dem084_extau

t:line.dem084_extaut,dem085_fextaut:line.dem085_fextaut,dem009_
domires_codmun:line.d
em009_domires_codmun,dem082_ocupabilidad:line.dem082_ocupab
ilidad,dem045_cnae:li
ne.dem045_cnae,dem030_situalab:line.dem030_situalab,dem031_sit
ualab_fini:line.dem031
_situalab_fini,dem032_situalab_time:line.dem032_situalab_time,dem
038_ere:line.dem038_
ere,dem039_ere_fini:line.dem039_ere_fini,dem040_ere_ffin:line.dem
040_ere_ffin,dem048_
presta:line.dem048_presta,dem049_presta_caualt:line.dem049_prest
a_caualt,dem050_pres
ta_caubaj:line.dem050_presta_caubaj,dem051_presta_fini:line.dem0
51_presta_fini,dem052
_presta_ffin:line.dem052_presta_ffin,dem053_presta_basereg:line.de
m053_presta_basereg,
dem059_alta_uag:line.dem059_alta_uag,dem060_alta_fini:line.dem0
60_alta_fini,dem075_al
ta_ccaa:line.dem075_alta_ccaa,dem076_alta_nueva:line.dem076_alt
a_nueva,dem067_alta
_ftras:line.dem067_alta_ftras,dem087_alta_diaaltm:line.dem087_alta
_diaaltm,dem061_situa
adm:line.dem061_situaadm,dem062_situaadm_cau:line.dem062_situ
aadm_cau,dem063_sit
uaadm_fini:line.dem063_situaadm_fini,dem064_situaadm_fcau:line.d
em064_situaadm_fcau
,dem065_situaadm_frcau:line.dem065_situaadm_frcau,dem066_situ
aadm_uag:line.dem066
_situaadm_uag,dem068_demanda_tiempo:line.dem068_demanda_ti
empo,dem069_deman
da_larga:line.dem069_demanda_larga,dem070_demanda_noparo:lin
e.dem070_demanda_n

oparo,dem071_demanda_inscrita:line.dem071_demanda_inscrita,de

m072_desempl_06m:li
ne.dem072_desempl_06m,dem073_desempl_12m:line.dem073_des
empl_12m,dem088_de
sempl_18m:line.dem088_desempl_18m,dem074_desempl_24m:line.
dem074_desempl_24m
,dem077_situadem01:line.dem077_situadem01,dem077_situadem02
:line.dem077_situadem
02,dem077_situadem03:line.dem077_situadem03,dem078_desempl
_di01:line.dem078_des
empl_di01,dem078_desempl_di02:line.dem078_desempl_di02,dem0
78_desempl_di03:line.
dem078_desempl_di03,dem079_desempl_rec:line.dem079_desempl
_rec,dem036_dispo_fin
i:line.dem036_dispo_fini,dem037_dispo_ffin:line.dem037_dispo_ffin,d
em033_dispo_tipojor:li
ne.dem033_dispo_tipojor,dem034_dispo_tpperiodo:line.dem034_disp
o_tpperiodo,dem035_
dispo_tphoras:line.dem035_dispo_tphoras,dem041_dispo_tradom:lin
e.dem041_dispo_trado
m,dem042_dispo_autoemp:line.dem042_dispo_autoemp,dem043_di
spo_fijodisc:line.dem04
3_dispo_fijodisc,dem044_dispo_teletrab:line.dem044_dispo_teletrab,
dem054_geobusca:line
.dem054_geobusca,dem055_geobusca_fccaa:line.dem055_geobusc
a_fccaa,dem056_geob
usca_sfccaa:line.dem056_geobusca_sfccaa,dem057_geobusca_ue:li
ne.dem057_geobusca
_ue,dem058_geobusca_fue:line.dem058_geobusca_fue})

2.2.1. CREACIÓN DE SUBNODOS

DISCAPACIDAD

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_DEMCAN20170630_V01.csv" AS line FIELDTERMINATOR ";"

CREATE(Discapacidad:Discapacidad{uuid:line.uuid,stid:line.stid,luid:line.luid,dem029_disc01:line.dem029_disc01,dem029_disc02:line.dem029_disc02,dem029_disc03:line.dem029_disc03})

COLECTIVO ESPECIAL

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_DEMCAN20170630_V01.csv" AS line FIELDTERMINATOR ";"

CREATE(Colectivo_Especial:Colectivo_especia{uuid:line.uuid,stid:line.stid,luid:line.luid,dem046_colesp01:line.dem046_colesp01,dem047_colesp01_fini:line.dem047_colesp01_fini})

OCUPACIONES SOLICITADAS

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_DEMCAN20170630_V01.csv" AS line FIELDTERMINATOR ";"

CREATE(Ocupaciones_Solicitadas:Ocupaciones_Solicitadas{uuid:line.uuid,stid:line.stid,luid:line.luid,dem012_ocu01:line.dem012_ocu01,dem013_ocu01_nivel:line.dem013_ocu01_nivel,dem014_ocu01_exp:line.dem014_ocu01_exp,dem015_ocu01_fini:line.dem015_ocu01_fini,dem012_ocu02:line.dem012_ocu02,dem013_ocu02_nivel:line.dem013_ocu02_nivel,dem014_ocu02_exp:line.dem014_ocu02_exp,dem015_ocu02_fini:line.dem015_ocu02_fini,dem012_ocu03:line.dem012_ocu03,dem013_ocu03_nivel:line.dem013_ocu03_nivel,dem014_ocu03_exp:line.dem014_ocu03_exp,dem015_ocu03_fini:line.dem015_

ocu03_fini,dem012_ocu04:line.dem012_ocu04,dem013_ocu04_nivel:line.dem013_ocu04_nivel,dem014_ocu04_exp:line.dem014_ocu04_exp,dem015_ocu04_fini:line.dem015_ocu04_fini,dem012_ocu05:line.dem012_ocu05,dem013_ocu05_nivel:line.dem013_ocu05_nivel,dem014_ocu05_exp:line.dem014_ocu05_exp,dem015_ocu05_fini:line.dem015_ocu05_fini,dem012_ocu06:line.dem012_ocu06,dem013_ocu06_nivel:line.dem013_ocu06_nivel,dem014_ocu06_exp:line.dem014_ocu06_exp,dem015_ocu06_fini:line.dem015_ocu06_fini})

IDIOMAS

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_DEMCAN20170630_V01.csv" AS line FIELDTERMINATOR ";" CREATE(Idiomas:Idiomas{uuid:line.uuid,dem018_idiom01:line.dem018_idiom01,dem019_idiom01_lect:line.dem019_idiom01_lect,dem020_idiom01_escr:line.dem020_idiom1_escr,dem022_idiom01_inte:line.dem022_idiom01_inte,dem023_idiom01_trad:line.dem023_idiom01_trad,dem024_idiom01_doce:line.dem024_idiom01_doce,dem018_idiom02:line.dem018_idiom02,dem019_idiom02_lect:line.dem019_idiom02_lect,dem020_idiom02_escr:line.dem020_idiom1_escr,dem022_idiom02_inte:line.dem022_idiom02_inte,dem023_idiom02_trad:line.dem023_idiom02_trad,dem024_idiom02_doce:line.dem024_idiom02_doce,dem018_idiom03:line.dem018_idiom03,dem019_idiom03_lect:line.dem019_idiom03_lect,dem020_idiom03_escr:line.dem020_idiom1_escr,dem022_idiom03_inte:line.dem022_idiom03_inte,dem023_idiom03_trad:line.dem023_idiom03_trad,dem024_idiom03_doce:line.dem024_idiom03_doce,dem018_idiom04:line.dem018_idiom04,dem019_idiom04_lect:line.dem019_idiom04_lect,dem020_idiom04_escr:line.dem020_idiom1_escr,dem022_idiom04_inte:line.dem022_idiom04_inte,dem023_idiom04_trad:line.dem023_idiom04_trad,dem024_idiom04_doce:line.dem024

_idiom04_doce,dem018_idiom05:line.dem01
8_idiom05,dem019_idiom05_lect:line.dem019_idiom05_lect,dem020
_idiom05_escr:line.dem020_idiom1_escr,dem022_idiom05_inte:line.d
em022_idiom05_inte,dem023_idiom05_trad:line.dem023_idiom05_tr
ad,dem024_idiom05_doce:line.dem024_idiom05_doce,dem018_idio
m06:line.dem018_idiom06,dem019_idiom06_lect:line.dem019_idiom
06_lect,dem020_idiom06_escr:line.dem020_idiom1_escr,dem022_idi
om06_inte:line.dem022_idiom06_inte,dem02
3_idiom06_trad:line.dem023_idiom06_trad,dem024_idiom06_doce:lin
e.dem024_idiom06_doce,dem018_idiom07:line.dem018_idiom07,de
m019_idiom07_lect:line.dem019_idiom07_lect,dem020_idiom07_esc
r:line.dem020_idiom1_escr,dem022_idiom07_inte:line.dem022_idio
m07_inte,dem023_idiom07_trad:line.dem023_idiom07_trad,dem024_
idiom07_doce:line.dem024_idiom07_doce,dem018_idiom08:line.dem
018_idiom08,dem019_idiom08_lect:line.dem019_idiom08_lect,dem0
20_idiom08_escr:line.dem020_idiom1_escr,dem022_idiom08_inte:lin
e.dem022_idiom08_inte,dem023_idiom08_trad:line.dem023_idiom08
_trad,dem024_idio m08_doce:line.dem024_idiom08_doce})

FORMACIÓN COMPLEMENTARIA

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_DEMCAN20170630_V01.csv" AS line FIELDTERMINATOR ";"
CREATE(Formacion_Complementaria:Formacion_Complementaria{u
uid:line.uuid,dem025_forcom01_nivel:line.dem025_forcom01_nivel,d
em026_forcom01_area:line.dem026_forcom01_area,dem027_forcom
01_fam:line.dem027_forcom01_fam,dem028_forcom01_ho:line.dem
028_forcom01_ho,dem025_forcom02_nivel:line.dem025_forcom02_n
ivel,dem026_forcom02_area:line.dem026_forcom02_area,dem027_f
orcom02_fam:line.dem027_forcom02_fam,de
m028_forcom02:line.dem028_forcom02_ho,dem025_forcom03_nivel:

line.dem025_forcom03_nivel,dem026_forcom03_area:line.dem026_forcom03_area,dem027_forcom03_fam:line.dem027_forcom03_fam,dem028_forcom03:line.dem028_forcom03_ho,dem025_forcom04_nivel:line.dem025_forcom04_nivel,dem026_forcom04_area:line.dem026_forcom04_area,dem027_forcom04_fam:line.dem027_forcom04_fam,dem028_forcom04:line.dem028_forcom04_ho,dem025_forcom05_nivel:line.dem025_forcom05_nivel,dem026_forcom05_area:line.dem026_forcom05_area,dem027_forcom05_fam:line.dem027_forcom05_fam,dem028_forcom05:line.dem028_forcom05_ho,dem025_forcom06_nivel:line.dem025_forcom06_nivel,dem026_forcom06_area:line.dem026_forcom06_area,dem027_forcom06_fam:line.dem027_forcom06_fam,dem028_forcom06:line.dem028_forcom06_ho,dem025_forcom07_nivel:line.dem025_forcom07_nivel,dem026_forcom07_area:line.dem026_forcom07_area,dem027_forcom07_fam:line.dem027_forcom07_fam,dem028_forcom07:line.dem028_forcom07_ho,dem025_forcom08_nivel:line.dem025_forcom08_nivel,dem026_forcom08_area:line.dem026_forcom08_area,dem027_forcom08_fam:line.dem027_forcom08_fam,dem028_forcom08:line.dem028_forcom08_ho,dem025_forcom09_nivel:line.dem025_forcom09_nivel,dem026_forcom09_area:line.dem026_forcom09_area,dem027_forcom09_fam:line.dem027_forcom09_fam,dem028_forcom08:line.dem028_forcom09_ho,dem025_forcom10_nivel:line.dem025_forcom10_nivel,dem026_forcom10_area:line.dem026_forcom10_area,dem027_forcom10_fam:line.dem027_forcom10_fam,dem028_forcom10:line.dem028_forcom10_ho})

2.2.2. CREACIÓN DE RELACIONES ENTRE SUBNODOS Y DATOS_DEMCAN

MATCH(a:Dat_demcan), (b:Discapacidad) WHERE a.uuid=b.uuid CREATE (a)-[r:R]->(b) MATCH(a:Dat_demcan), (b:Idiomas) WHERE a.uuid=b.uuid CREATE (a)-[r:R]->(b)

MATCH(a:Dat_demcan), (b:Formacion_Complementaria) WHERE

a.uuid=b.uuid CREATE (a)-[r:R]->(b)

MATCH(a:Dat_demcan), (b:Ocupaciones_Solicitadas) WHERE a.uuid=b.uiud CREATE (a)-[r:R]->(b)

MATCH(a:Dat_demcan), (b:Colectivo_Especial) WHERE a.uuid=b.uuid CREATE (a)-[r:R]->(b)

2.3. DATOS COTIZACIÓN

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_CCOTIZA_20170630_V01.csv" AS line FIELDTERMINATOR ";"

CREATE(:Dat_ccotiza{uuid:line.uuid,luid:line.luid,stid:line.stid,marcatiempo:line.marcatiempo
,ccotiza_domisoc_cmun:line.ccotiza_domisoc_cmun,ccotiza_domisoc_cpost:line.ccotiza_do
misoc_cpost,ccotiza_falta:line.ccotiza_falta,ccotiza_cnae:line.ccotiza_cnae,ccotiza_asalaria
dos:line.ccotiza_asalariados,ccotiza_situacion_admin:line.ccotiza_situacion_admin,ccotiza_f situacion:line.ccotiza_fsituacion});

2.4. DATOS PADRÓN

LOAD CSV WITH HEADERS FROM "file:///GrafosIstac/DAT_PMH20170701_V01.csv" AS line FIELDTERMINATOR ";"

CREATE(Dat:Dat_pmh{uui:line.uuid,luid:line.luid,stid:line.stid,marcatiempo:line.marcatiempo,pmh_domires_codmun_rc:line.pmh_domires_codmun_rc,pmh_id_codmunnac_rc:line.pmh_id_codmunnac_rc,pmh_id_paisnac_rc:line.pmh_id_paisnac_rc,pmh_id_fnac:line.pmh_id_fnac,pmh_id_cdev:line.pmh_id_cdev,pmh_id_fvar:line.pmh_id_fvar,pmh_

id_cvar:line.pmh_id_cvar,pmh_id_cauv:line.pmh_id_cauv,pmh_dv_sexo:line.pmh_dv_sexo,pmh_dv_codmunac_rc:line.pmh_dv_codmunac_rc,pmh_dv_paisnac_rc:line.pmh_dv_paisnac_rc,pmh_dv_fnac:line.pmh_dv_fnac,pmh_dv_cnes:line.pmh_dv_cnes,pmh_dv_naci:line.pmh_dv_naci,pmh_dv_naci_rc:line.pmh_dv_naci_rc,pmh_dv_prdp:line.pmh_dv_prdp,pmh_dv_mudp:line.pmh_dv_mud,p,pmh_dv_codp:line.pmh_dv_codp})

3.CARGANDO DATOS DEL REGISTRO POBLACIÓN

3.1 CARGANDO FICHERO IDT

LOAD CSV WITH HEADERS FROM "file:///data/poblacion/IDT_SELECT_TFG.csv" AS line FIELDTERMINATOR ";" MERGE(:Idt_Poblacion{uuid:line.UUID_IDT, stid:line.STID_IDT})

3.2 CARGANDO FICHERO IDF

LOAD CSV WITH HEADERS FROM "file:///data/poblacion/IDF_POBLACION_TFG.csv" AS line FIELDTERMINATOR ";" CREATE(:Idf_Poblacion{uuid:line.UUID,

stid:line.STID,marca_tiempo:line.MARCATIEMPO,

NOMEPER_TIPO_IDF:line.NOMEPER_TIPO_IDF,NOMEPER_FNAC:line.NOMEPER_FNA C, NOMEPER_SEXO:line.NOMEPER_SEXO,

NOMEPER_CODMUNNAC:line.NOMEPER_CODMUNNAC,NOMEPER_PAISANAC:line.NO MEPER_PAISNAC});

3.3. CREANDO RELACIONES

3.3.1 RELACIONES SOURCE(IDT-IDF)

```
LOAD CSV WITH HEADERS FROM "file:///data/poblacion/URD_POBLACION_TFG.csv" AS line FIELDTERMINATOR ";"
MATCH(Idt_Poblacion:Idt_Poblacion{uuid:line.UUID_A})
MATCH(Idf_Poblacion:Idf_Poblacion{uuid:line.UUID_B})

CREATE(Idt_Poblacion)-[r:SOURCE{REL_TYPE:line.REL_TYPE,REL_PERIOD:line.REL_P
ERIOD,REL_EVENT:line.REL_EVENT,REL_DESCRIPTION:line.REL_DESCRIPTION}]->(Idf _Poblacion);
```

3.3.2 RELACIONES TRACESTOCK(IDT-DAT)

DAT_afilia

```
LOAD CSV WITH HEADERS FROM "file:///poblacion/URD_POBLACION_TFG.csv" AS line FIELDTERMINATOR ";"
MATCH(Idt_Poblacion:Idt_Poblacion{uuid:line.UUID_A}) MATCH (Dat:Dat_afilia{uuid:line.UUID_B})

CREATE(Idt_Poblacion)-[r:TRACESTOCK{LINK_TYPE:line.LINK_TYPE,LINK_DESCRIPTI
ON:line.LINK_DESCRIPTION,LINK_QUALITY:line.LINK_QUALITY,LINK_QUALITY_RANK:li
ne.LINK_QUALITY_RANK,LINK_ACTIVE:line.LINK_ACTIVE,LINK_ACTIVE_IN:line.LINK_A
CTIVE_IN,LINK_ACTIVE_OUT:line.LINK_ACTIVE_OUT}]->(Dat);
```

DAT_demcan

LOAD CSV WITH HEADERS FROM "file:///poblacion/URD_POBLACION_TFG.csv" AS line FIELDTERMINATOR ";" MATCH(Idt_Poblacion:Idt_Poblacion{uuid:line.UUID_A}) MATCH (Dat:Dat_demcan{uuid:line.UUID_B})

CREATE(Idt_Poblacion)-[r:TRACESTOCK{LINK_TYPE:line.LINK_TYPE,LINK_DESCRIPTION:line.LINK_DESCRIPTION,LINK_QUALITY:line.LINK_QUALITY,LINK_QUALITY_RANK:line.LINK_QUALITY_RANK,LINK_ACTIVE:line.LINK_ACTIVE,LINK_ACTIVE_IN:line.LINK_ACTIVE_IN,LINK_ACTIVE_OUT:line.LINK_ACTIVE_OUT}]->(Dat);

DAT_pmh

LOAD CSV WITH HEADERS FROM "file:///poblacion/URD_POBLACION_TFG.csv" AS line FIELDTERMINATOR ";" MATCH(Idt_Poblacion:Idt_Poblacion{uuid:line.UUID_A}) MATCH (Dat:Dat_pmh{uuid:line.UUID_B})

CREATE(Idt_Poblacion)-[r:TRACESTOCK{LINK_TYPE:line.LINK_TYPE,LINK_DESCRIPTION:line.LINK_DESCRIPTION,LINK_QUALITY:line.LINK_QUALITY,LINK_QUALITY_RANK:line.LINK_QUALITY_RANK,LINK_ACTIVE:line.LINK_ACTIVE,LINK_ACTIVE_IN:line.LINK_ACTIVE_IN,LINK_ACTIVE_OUT:line.LINK_ACTIVE_OUT}]->(Dat);

4.CARGANDO DATOS DEL REGISTRO PORTALES

4.1 CARGANDO DATOS IDT

```
LOAD CSV WITH HEADERS FROM "file:///data/portales/IDT_PORTALES_TFG.csv" AS line FIELDTERMINATOR ";" MERGE(:Idt_Portales{uuid:line.uuid,stid:line.stid});
```

4.2 CARGANDO DATOS IDF

```
LOAD CSV WITH HEADERS FROM "file:///data/portales/IDF_PORTALES_TFG.csv" AS line FIELDTERMINATOR ";"

MERGE(:Idf_Portales{uuid:line.uuid,stid:line.stid,codmun:line.codmun });
```

4.3 . CREANDO RELACIONES

4.1.1 RELACIONES SOURCE

```
LOAD CSV WITH HEADERS FROM "file:///data/portales/URD_PORTALES_TFG.csv" AS line FIELDTERMINATOR ";"
MATCH(Idt_Poblacion:Idt_Portales{uuid:line.uuid_a})
MATCH(Idf_Poblacion:Idf_Portales{uuid:line.uuid_b})

CREATE(Idt_Portales)-[r:SOURCE{REL_TYPE:line.rel_type,REL_PERIOD:line.rel_period,R
EL_EVENT:line.rel_event,REL_DESCRIPTION:line.rel_description}]->(Idf_Portales);
```

4.1.2 RELACIONES TRACESTOCK

DAT_pmh

```
LOAD CSV WITH HEADERS FROM
```

"file:///portales/URD_PORTALES_TFG.csv" AS line FIELDTERMINATOR ";" MATCH(Idt_Portales:Idt_Portales{uuid:line.uuid_a}) MATCH (Dat:Dat_pmh{uuid:line.uuid_b})

CREATE(Idt_Portales)-[r:TRACESTOCK{LINK_TYPE:line.link_type,LINK_DESCRIPTION:lin
e.link_description,LINK_QUALITY:line.link_quality,LINK_QUALITY_RANK:line.link_quality_r
ank,LINK_ACTIVE:line.link_active,LINK_ACTIVE_IN:line.link_active_in,LINK_ACTIVE_OUT: line.link_active_out}]->(Dat);

5.CARGANDO DATOS DEL REGISTRO EMPRESA

5.1 CREACIÓN FICHEROS IDT

LOAD CSV WITH HEADERS FROM "file:///empresa/IDT_EMPRESAS_TFG.csv" AS line FIELDTERMINATOR ";" MERGE(:Idt_Empresa{uuid:line.uuid,stid:line.stid});

5.2 CREACIÓN FICHEROS IDF

LOAD CSV WITH HEADERS FROM "file:///data/empresa/IDT_EMPRESAS_TFG.csv" AS line FIELDTERMINATOR ";"

MERGE(:Idf_Empresa{uuid:line.UUID;stid:line.STID;marca_tiempo:line.MARCATIEMPO;em
presa_persona_fisica:line.EMPRESA_PERSONA_FISICA});

5.3 CREANDO RELACIONES

5.3.1 RELACIONES SOURCE

LOAD CSV WITH HEADERS FROM

"file:///empresa/URD_EMPRESAS_TFG.csv" AS line FIELDTERMINATOR ";"
MATCH(Idt_Empresa:Idt_Empresa{uuid:line.UUID_A})
MATCH(Idf_Empresa:Idf_Empresa{uuid:line.UUID_B})

CREATE(Idt_Empresa)-[r:SOURCE{rel_type:line.REL_TYPE,rel_period:line.REL_PERIOD,r
el_event:line.REL_EVENT,rel_description:line.REL_DESCRIPTION}]->(Idf_Empresa);

5.3.2 RELACIONES TRACESTOCK

DAT_afilia

LOAD CSV WITH HEADERS FROM "file:///empresa/URD_EMPRESAS_TFG.csv" AS line FIELDTERMINATOR ";"
MATCH(Idt_Empresa:Idt_Empresa{uuid:line.UUID_A})
MATCH (Dat:Dat_afilia{uuid:line.UUID_B})

CREATE(Idt_Empresa)-[r:TRACESTOCK{link_type:line.LINK_TYPE,link_description:line.LINK_DESCRIPTION,link_quality:line.LINK_QUALITY,link_quality_rank:line.LINK_QUALITY_R
ANK,link_active:line.LINK_ACTIVE,link_active_in:line.LINK_ACTIVE_IN,link_active_out:line. LINK_ACTIVE_OUT}]->(Dat);

DAT_ccotiza

LOAD CSV WITH HEADERS FROM "file:///empresa/URD_EMPRESAS_TFG.csv" AS line FIELDTERMINATOR ";"
MATCH(Idt_Empresa:Idt_Empresa{uuid:line.UUID_A}) MATCH (Dat:Dat_ccotiza{uuid:line.UUID_B})

CREATE(Idt_Empresa)-[r:TRACESTOCK{link_type:line.LINK_TYPE,link_description:line.LIN
K_DESCRIPTION,link_quality:line.LINK_QUALITY,link_quality_rank:li

```
ne.LINK_QUALITY_R
ANK,link_active:line.LINK_ACTIVE,link_active_in:line.LINK_ACTIVE_
IN,link_active_out:line. LINK_ACTIVE_OUT}]->(Dat);


********************************************************************************* *
* JAIME RODRIGUEZ PEREZ
*
*
* 01-09-2021
*
*
* DISEÑO DE LA BASE DE DATOS ORIENTADA A GRAFOS, SENTENCIAS
PARA CREARLA, USO DE CLÁUSULAS CYPHER [3]
*
*
*********************************************************************************/
```

Bibliografía

[1] Página oficial de la tecnología neo4j. Fecha de última consulta: 01-09-2021 https://neo4j.com/

[2] 2019, ISTAC, estadística de población activa registrada(EPA-reg), p.7

[3] Manual de cláusulas de Cypher. Fecha de última consulta:01-09-2021 https://neo4j.com/docs/cypher-manual/current/clauses/

[4] Gráficos de popularidad de bases de datos orientadas a grafos. Fecha de última consulta: 01-09-2021 https://db-engines.com/en/ranking_trend/graph+dbms

[5] Tutorial para empezar a usar Cypher. Fecha de última consulta: 01-09-2021 https://neo4j.com/developer/cypher/intro-cyphe

[6] Claúsula WHERE del manual de cypher Fecha de última consulta: 06-09-2021 https://neo4j.com/docs/cypher-manual/current/clauses/where/

[7] Claúsula MATCH del manual de cypher Fecha de última consulta: 06-09-2021 https://neo4j.com/docs/cypher-manual/current/clauses/match/