Alejandro Jaime Cabrera (NIU: 1636745) Asiel López Serrano (NIU: 1636405)
Joel Tapia Salvador (NIU: 1638962) David Montaña Tseitlin (NIU: 1564129)
GITHUB: https://github.com/NIU1638962/Projecte-Neo4j/invitations

Ejercicio 1

"Importa les dades en la BD de Neo4j del projecte. Genera un script en cypher que carregui totes les dades, generi tots els nodes, relacions i afegeixi les característiques allà on toqui."
(Asiel López) - Fitxer: veure.cypher

Para hacer este ejercicio primero lo que hacemos es cargar los nodos:

- Para los de INDIVIDUAL utilizamos conversores de tipo toInteger para el año y la id del nodo, justo después de cargar este tipo de nodos creamos la primera constraint, llamada UniqueIndividual, con el objetivo de que no se dupliquen nodos que tienen la misma id.
- Lo siguiente es cargar los nodos de HABITATGE donde utilizamos otros tres conversores a entero para el número, el año del padrón y para la id del lugar, y seguidamente creamos la segunda constraint llamada UniqueHabitatge para que no se dupliquen nodos con la misma id de lugar.

Ahora toca crear las relaciones:

- Comenzando por las de FAMILIA, utilizamos dos conversores de enteros para las id de los nodos, hacemos un match para encontrarlos y para que no se pueda duplicar la relación, hacemos un MERGE del primer nodo con el segundo sin flecha para que sea bidireccional de una relación con la etiqueta FAMILIA.
- Para hacer la siguiente relación de VIU, la creamos de una forma ligeramente diferente, primero utilizamos dos toInteger para el año, la id de la vivienda y para el id del individuo y en el MERGE lo que hacemos es una relación unidireccional del individuo hacia la vivienda con la etiqueta VIU.
- Para la última relación de SAME_AS, utilizamos dos toInteger para las id de los individuos que resultan ser la misma persona, y hacemos un MERGE con la relación etiquetada como SAME_AS.

Por último, un extra que planteamos es que si se ejecuta varias veces el archivo "veure.cypher" nos daría error al estar intentando crear duplicados de CONSTRAINTS y simplemente hacemos dos DROPS para las mismas.

Ejercicio 2

"Resoleu les següents consultes:"

(Alejandro Jaime & David Montaña) - Fitxer: consultes.cypher

Para la elaboración de este ejercicio hemos aplicado las diferentes técnicas aprendidas durante el curso.

	Alejandro	**David**
Reparto de consultas	4, 5, 6, 7, 8, 9 y 10	1, 2, 3, 11, 12 y 13

A continuación se detallan su funcionamiento:

1.- Miramos cada individuo (INDIVIDUAL) que vive en una vivienda (HABITATGE) y filtramos el año del censo y el municipio con atributos de HABITATGE.

2.- Es la misma que la consulta anterior pero ahora utilizamos la cláusula DISTINCT en COLLECT apellidos.

3.- Miramos todos los individuos del padrón, filtramos por año. Usando COLLECT agrupamos las viviendas por año con su id y ordenamos por año.

4.- Buscamos un Individual específico, 'rafel marti' en el año 1838, y luego buscamos otros Individual que vivían en el mismo Habitatge en 'SFLL'. Los nombres y apellidos de estos individuos se devuelven como resultado, eliminando los duplicados.

5.- Utilizamos la relación SAME_AS para encontrar todas las apariciones de 'miguel estape bofill' independientemente de las variaciones en la forma en que se escribía su nombre. Los nodos correspondientes se devuelven como resultado.

6.- Similar a la consulta anterior, pero en lugar de devolver los nodos, devolvemos el nombre, los apellidos y los segundos apellidos de 'miguel estape bofill', eliminando los duplicados.

7.- Buscamos todos los individuos relacionados con 'benito julivert' y devolvemos el nombre, los apellidos y el tipo de relación.

8.- Extensión de la consulta anterior, pero limitada a los hijos de 'benito julivert' y ordenada alfabéticamente por nombre.

9.- Buscamos todas las distintas relaciones FAMILIA entre cualquier pareja de nodos del grafo y las listamos.

10.- Identificamos los nodos que representan el mismo Habitatge (carrer y número) a lo largo de los padrones de Sant Feliu del Llobregat (SFLL), seleccionando solo los Habitatge que tenían ambas informaciones (carrer y número). Para cada Habitatge, devolvemos el carrer y número, el número total de padrones en los que aparece, la lista de años de los padrones y la lista de las IDs de las llars. Los resultados se ordenan por el total de padrones y se limitan a los 15 primeros.

11.- Como los hijos tienen dos padres tenemos que eliminar a las madres por eso buscamos a los individuos P que sean familia con ellos mismos y luego sus hijos que viven en una casa. Utilizamos el mapa WITH para guardar el número de hijos. Posteriormente retornamos lo que se pide pero ordenamos por el número de hijos y usamos LIMIT 20.

12.- Al igual que en la consulta anterior necesitamos el número de hijos así que el principio es el mismo. Como dicta el enunciado usamos una subquery CALL para obtener el número de casas. Para ello se cojen todas las casas de *Sant Feliu del Llobregat* y realizamos una "agrupación de tipo count". Dividimos el total de hijos entre el número de viviendas.

13.- La consulta dentro de CALL mira todas las viviendas de SFLL y las agrupa por calle y año en el WITH luego devolvemos el número de habitantes de las casas y el año. La query principal se encarga de encontrar en qué calles está este número mínimo de habitantes de aquel año. Esto se hace contando el número de habitantes y comparándolo con MIN_HABS. Ordenamos los resultados de manera ascendente por año.

Ejercicio 3

"En aquest exercici analitzarem les dades del graf per entendre millor l'estructura de les dades. Els següents apartats tenen com objectiu orientar-vos respecte a com utilitzar algunes de les eines que ofereix Neo4J."
(Joel Tapia) - Fitxer: exercici3.cypher

a) Estudi de les components connexes (cc) i de l'estructura de les component en funció de la seva mida.

Ejecutamos la comanda 1, esta utilizando el algoritmo de Louvain devuelve las comunidades de nodos detectados (considerando la dirección natural de los nodos), ordenadas por tamaño de más grande a más pequeño, devuelve todas las comunidades ordenadas de mayor a menor en orden de cantidad de nodos. También ejecutamos la comanda 2 que nos devolverá el número de comunidades utilizando este algoritmo sabemos cuántas comunidades aparecen con el algoritmo de louvain, repetimos lo mismo con los algoritmos labelPropagation (comanda 3 y comanda 4) y Weakly Connected Components (comanda 5 y comanda 6).

	Louvain	**labelPropagation**	**Weakly Connected Components**
Tamaño comunidades más grandes	361 nodos	51 nodos	14891 nodos
Número comunidades	2576 comunidades	4847 comunidades	2500 comunidades

De esta manera podemos hacernos a la idea de cómo están distribuidas las comunidades. Podemos deducir que hay una gran comunidad de tal manera que se puede llegar al 81% de los nodos saltando de una vivienda a otra y personas que están conectadas y hay 2500 comunidades diferentes (viviendas o conjuntos de viviendas) completamente aïsladas (información extraïda de wcc).
También podemos ver que las comunidades de grandes influencias entre nodos no ocupan más de 50 nodos de tamaño, comunidades bastantes pequeñas, y esto tiene sentido ya que en viviendas no muchas viven a lo largo de la historia en un mismo sitio y se conocen (información extraída de label propagation).

b) Semblança entre els nodes. Ens interessa saber quins nodes són semblants com a pas previ a identificar els individus que són el mateix (i unirem amb una aresta de tipus SAME_AS).

La ejecución de la comanda 7 nos da 11 nodos INDIVIDUOS con 181 aristas, a partir de este cálculo de similitud podríamos interpretar que en una vivienda pueden llegar a haber 11 tipos de roles y conexiones dentro de una vivienda.