Time Machine: Disseny i implementació d'un portal basat en grafs de coneixement per a representació d'informació històrica

Roger Casanova Sevilla

Resum— Aquest treball que s'emmarca dins del projecte Europeu Time Machine i té com a objectiu el desenvolupament d'una plataforma per a l'estudi i explotació de dades de censos històrics i manuscrits poblacionals de municipis del Baix Llobregat de principis del segle passat. La gestió i emmagatzematge de les dades s'ha realitzat a través de sistemes gestors de BBDD orientades a graf amb la finalitat de poder analitzar i explotar les dades a través d'una estructura d'aquestes característiques. Paral·lelament s'ha treballat en el desenvolupament d'un portal web amb connexió directa a l'estructura de graf, on poder realitzar consultes, visualitzar i interactuar amb el contingut i fent ús d'un formulari que s'encarregarà d'interpretar els imputs per a generar consultes sobre aquest graf permetent cert grau d'automatització i reduint la dificultat en la realització de consultes.

Paraules clau— Projecte Time Machine, Graph DB, Node.js, Knowledge graph, Cypher, Neo4j, Mongo DB, Api Rest, Funcions algoritmiques, responsive design, procés ETL, npm, neovis.js.

Abstract— This work is part of the European Time Machine project and aims to develop a platform for the study and exploitation of data from historical censuses and population manuscripts of municipalities in the Baix Llobregat of the early last century. Data management and storage has been done through graph-oriented DB management systems in order to be able to analyze and exploit the data through a structure of these characteristics. At the same time, has been developed a web portal with a direct connection to the graph structure, allowing to query, view and interact with the content and using a form that will interpret the imput information to generate queries on the graph structure giving a certain degree of automation and reducing the difficulty in making queries.

Index Terms— Time Machine Project , Graph DB, Node.js, Knowledge graph, Cypher, Neo4j, Mongo DB, Api Rest, Algorithmic functions, Responsive design, ETL process, npm neovis.js.

1 Introducció

AQUEST projecte s'emmarca dins del projecte europeu *Time Machine*, en el qual la UAB participa. El principal objectiu de *Time Machine* és la digitalització de documents, registres històrics, imatges, etc. dels últims 2000 anys i l'ús de la Intel·ligència Artificial per l'anàlisi d'aquestes dades. L'anomenat *Big Data* del pasat. Els censos i padrons i altres documents demogràfics conservats a arxius històrics són un reflex de les societats del passat i de l'evolució de les comunitats.

La finalitat d'aquest TFG és la de desenvolupar una plataforma per a l'estudi i explotació de dades de censos històrics manuscrits poblacionals de municipis del Baix Llobregat de principis del segle passat a través d'un procés on les dades, prèviament digitalitzades, han estat migrades a una base de dades orientada a graf. Per les seves característiques, les dades demogràfiques es poden representar i emmagatzemant en una estructura de graf o xarxa. Paral·lelament a la creació del graf, s'ha desenvolupat una pàgina web amb connexió directa al graf que permet realitzar consultes amb cert grau de complexitat de manera intuïtiva. Per tant els objectius principals del projecte es poden enunciar com segueix:

- Construcció d'una *BBDD* orientada a grafs utilitzant eines d'emmagatzematge i gestió d'aquestes estructures
- Creació d'una pàgina web amb connexió directa al graf per navegar, visualitzar i formular consultes sobre aquestes dades.
- Validar el software amb la importació de dades reals transcrites de padrons històrics de la comarca del Baix Llobregat.

E-mail de contacte: roger.casanovase@e-campus.uab.cat
Menció realitzada: Enginyeria del Software
Treball tutoritzat per: Josep Lladós Canet Departament de Ciències de la Computació
Curs 2019/20

L'ús d'una *BBDD* orientada a grafs ofereix un ventall de possibilitats sobre una font de dades d'aquestes característiques respecte a una *BBDD* relacional típica. En lloc de registres ara en tenim nodes, i en lloc de columnes cada node té associat una sèrie d'atributs. El veritable potencial però són les relacions establertes sobre aquests nodes a partir d'atributs coincidents. Per exemple els membres d'una unitat familiar poden tenir en comú el fet que viuen en un mateix domicili, tenen certs vincles de sang, etc. Es poden extreure multitud de dades i relacions en funció de la qualitat de la font de dades. Altres relacions més complexes com veïnatges, o professions permeten relacionar a persones formant clusters de nodes dels quals per exemple en podem fer anàlisis sociològiques fàcilment amb més profunditat. Com per exemple, detectar tendències per sexes, edats, classes socials, les professions minoritàries de l'època... També, si es compta amb el cens poblacional de diferents anys sobre una mateixa població, pot aportar la dimensió temporal i veure l'evolució dels seus habitants. En la Figura 1 es pot veure un domicili (node en verd) interconnectat amb les persones que hi resideixen en ell. Si fem clic sobre qualsevol dels nodes, podrem observar els atributs que el componen com per exemple el nom, on treballaven o el carrer on vivien. Aquesta eina de visualització és *Neo4j* i permet modelar, consultar i visualitzar una *BBDD* orientada a graf. És l'eina escollida per a realitzar la construcció del graf per a aquest projecte i a l'apartat 2.1 Eines i tecnologies es descriuran en detall les seves característiques.

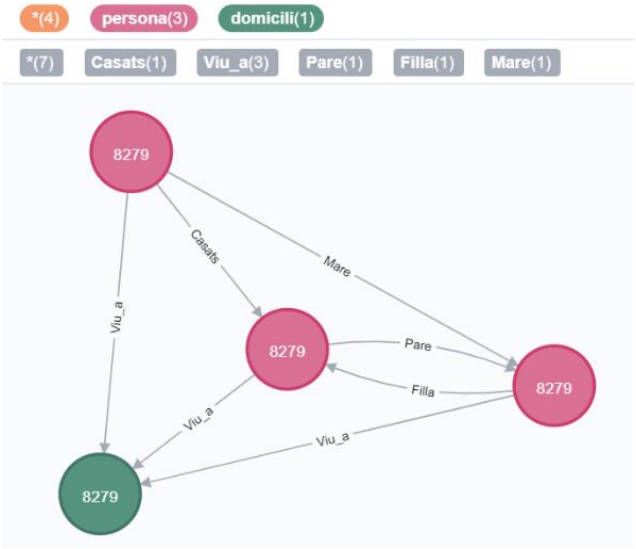

Figura 1. Exemple de consulta sobre un graf

La construcció d'un portal web on consultar el graf té la intenció de promoure aquest model d'accés a les dades. De popularitzar-se es poden extreure aquest tipus de conclusions a gran escala i assegurar la preservació de la memòria de moltes persones. Un portal d'aquestes característiques pot facilitar enormement la cerca de fa-miliars i avantpassats. Evidentment tot això pot portar a conflictes legals amb relació a la privacitat. Es poden aplicar tècniques d'emmascarament, per exemple esborrant atributs identificadors i aplicant distorsió sobre els atributs semi identificadors de les dades per a preservar aquesta privacitat o limitar les possibles consultes.

La resta d'aquesta memòria s'organitza de la següent manera. A la secció 2 es fa un breu repàs de l'estat de l'art. La secció 3 descriu la metodologia utilitzada per al desenvolupament del projecte. La secció 4 se centra a explicar la creació del graf, els algoritmes que s'han explorat durant el desenvolupament del projecte i final-ment una explicació detallada de les funcionalitats de la pàgina web. La secció 5 està dividida en dos apartats, en el primer conté els requisits funcionals i no funcionals i a continuació es fa una explicació detallada de l'arquitectura del sistema. A la secció 6 s'expliquen els mètodes de testing i bones pràctiques que s'han seguit al llarg de desenvolupament i finalment a l'últim apartat s'extrauen conclusions del treball realitzat.

2 METODOLOGÍA

2.1 Eines i tecnologies

Al llarg del desenvolupament del projecte han estat necessàries diverses tecnologies i llenguatges de programació. Una tasca fonamental a l'inici del desenvolupament del sistema, més concretament en la planificació, ha estat l'estudi i elecció de les eines i mecanismes amb els quals s'havia d'obtenir un equilibri pel que fa a complexitat, fiabilitat i modernitat i sobretot tenint en compte quines s'adaptaven millor a les característiques del sistema. A continuació es descriuen les tecnologies que han estat utilitzades en el desenvolupament del projecte:

Neo4j: És el gestor de bases de dades escollit per a la creació del graf. Ofereix moltes facilitats tant per a una importació de dades eficient com per a manegar grans volums de dades sense caigudes de rendiment. Un altre punt fort és que compta amb una gran comunitat que ha treballat en eines pensades per a permetre encastar i visualitzar grafs en entorns web sense haver d'interactuar directament amb el software de *Neo4j*. A més és una eina completament gratuïta.

Cypher: És el llenguatge de consultes que utilitza per *Neo4j* (no és propi de *Neo4j*, també és utilitzat en altres tecnologies similars). Està basat en *SQL*, en cas d'haver treballat amb bases de dades relacionals prèviament fa que sigui relativament fàcil aprendre a utilitzar-ho.

Node.js: És un software de codi obert utilitzat per a poder desenvolupar el back end d'una web, és a dir el seu servidor, en *JavaScript*. A més és asíncron el que vol dir que pot executar el codi en diferents fils d'execució evitant que aquest es quedi bloquejat. Compta amb una arquitectura orientada a esdeveniments.

Npm: És el sistema de gestió de paquets amb el que treballa Node.js. Alguns dels paquets utilitzats per a desenvolupar el back end de l'aplicació són Express per a configurar

el servidor, Cors per a possibilitar les comunicacions *http* desde el navegador i *Mongoose* per a crear i realitzar la connexió del servidor amb la base de dades no relacional *Mongo DB* utilitzada. Més endavant s'explicarà el propòsit d'aquesta base de dades.

Neovis.js: Aquesta llibreria ha estat desenvolupada per a poder donar resposta a les necessitats de poder visualitzar i consultar de manera externa un graf desenvolupat en *Neo4j*.

Webpack: És una eina útil per a gestionar els múltiples arxius que intervenen en un projecte. Utilitzant l'arxiu de configuració *webpack.config.js* es gestionen tots els arxius del front end perquè siguin carregats a la carpeta publica del projecte. Es configura d'aquesta manera per a mantenir una organització clara dels arxius que han de ser públics per als usuaris i quins no han de poder ser editats. És a dir la programació del front end no pot ser editada sense permisos, el que es visualitza per pantalla és la conversió a codi estàndard (*CSS, HTML i JS*) que fa *webpack.*

Llenguatges de programació: Per al desenvolupament del *front end* de l'aplicació s'ha utilitzat principalment *HTML5 CSS3* i *Javascript* en combinació amb el framework *Bootstrap4.*

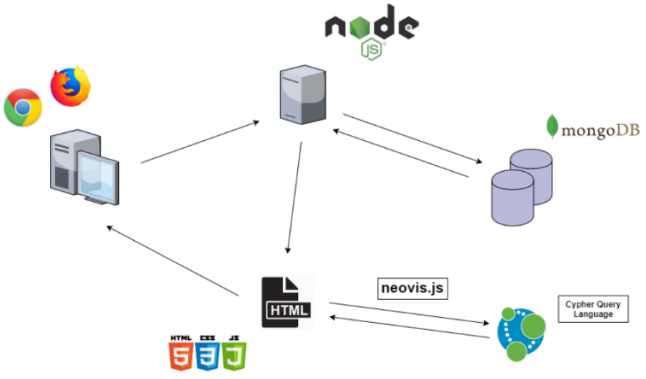

Figura 2. Principals elements que composen el sistema

La figura 2 mostra els principals components del sistema i les eines descrites que han estat utilitzades per al desenvolupament.

2.1 Cronograma

Donat el desconeixement inicial de moltes de les eines utilitzades en el projecte, una de les tasques fonamentals ha estat l'autoaprenentatge. Per aquest motiu una part important del temps dedicat ha estat en la formació per a poder abordar els objectius que més endavant es presenten. Un altre punt important a tenir en compte i més en un projecte d'enginyeria del software és la documentació i planificació al llarg d'aquests mesos.

A continuació es farà un repàs als diferents objectius que es van definir en un primer moment i es farà una anàlisi del punt en el qual es troben actualment.

O1 Implementar la BBDD orientada a graf a partir de les fonts de dades de censos municipals que han estat proporcionades.

Aquest punt es pot considerar com el bàsic per a la resta d'aspectes a tractar. És a dir, la resta d'objectius tenen una dependència directa. S'han definit les tècniques necessàries per a poder realitzar la migració de dades i per tant amb la documentació actual i els coneixements adquirits amb aquest objectiu es pot treballar un projecte de característiques similars però major complexitat com per exemple un escenari amb major quantitat de dades, una font de dades de menys qualitat i més difícil d'analitzar, major quantitat d'etiquetes de nodes i per tant un increment en el nombre de relacions resultants, etc.

O2 Desenvolupament de la pàgina web.

Si bé el punt anterior podria ser considerat com l'objectiu bàsic, el desenvolupament de la pàgina web es tracta del punt de major importància. L'objectiu del projecte és desenvolupar amb èxit un portal web on interactuar amb una base de dades orientada a graf. Les funcionalitats crítiques han estat implementades. Pel que fa a l'arquitectura de la web es va plantejar des d'un principi que era necessari un disseny modern que fos adaptatiu per a poder ser utilitzat sense problemes en diferents resolucions (pc, smartphones, tablets...). Això s'ha aconseguit com ja ha estat comentat en l'apartat d'eines i tecnologies, gra-cies a la utilització de frameworks per al front end que faciliten el disseny adaptatiu i un back end implementat amb Node.js que ha permès gestionar l'intercanvi de dades amb el servidor de manera asíncrona facilitant una resposta de temps optima del sistema.

O3 Aplicar algorismes per a generar noves relacions i millorar la qualitat del graf.

Des d'un principi es va plantejar aquest objectiu com un mecanisme de millora de l'estructura del graf i com a eina necessària per a establir relacions fiables però no ha estat el cas. Pot ser que estructures de graf més complexes sigui necessari aplicar certs tipus d'algorismes per a detectar clústers de nodes i establir unes determinades relacions. Per tant la importància d'aquest punt a estat relegada a un aspecte més experimental on tractar d'aplicar determinades funcions algorítmiques sobre el graf del projecte. Una possible implementació futura pot ser afegir l'opció d'aplicar alguns d'aquests algorismes en els resultats obtinguts amb la llibreria neovis.js ,utilitzada per a visualitzar el graf a la pàgina web, que permet la integració d'algues d'aquestes tècniques com per exemple pageRank (explicat a l'apartat 4.2).

O4 Visualitzar el graf a la pàgina web.

Aquest punt podria anar de la mà amb el propi desenvolupament de la pàgina, però val la pena diferenciar-ho, ja que no és una característica típica com pot ser el disseny del front end o la connexió amb el servidor i l'intercanvi de dades. El procés d'encastar el graf a la pàgina web es pot portar a terme amb multitud de llibreries pensades per a aquest propòsit. Com ja s'ha comentat prèviament, s'ha utilitzat Neovis.js, una llibreria desenvolupada per a establir de manera directa una connexió amb el servidor on es troba allotjat el graf. Existeixen altres tècniques com la llibreria DS3.js però tenen propòsits més generals i una pitjor consonància amb Neo4j. Una millora futura en aquest aspecte que a més aprofitaria la possibilitat de poder emmagatzemar consultes a la base de dades del servidor, és la d'associar imatges a IDs de nodes o consultes per a mostrar per exemple, en cas de buscar una determinada persona, mostrar una fotografia relacionada. O petites descripcions que podrien anar associades a personatges il·lustres d'un municipi (polítics, artistes, etc.)

3 Estat de l'art

L'ús de bases de dades orientades a graf s'ha popularitzat en l'última dècada. Un exemple de referència (si no el més important) és el graf de coneixement amb el qual Google complementa les cerques al seu navegador. Aquest Knowledge Graph té la capacitat d'oferir resultats alternatius que estan lligats directament a la cerca realitzada. En cas de buscar un artista del barroc com per exemple Lorenzo Bernini [figura 3] es mostrarà a altres artistes del barroc com a alternatives, o altres escultors, els seus familiars... Existeixen moltes aplicacions actualment en ús que moltes persones ja han normalitzat però que no eren tan comuns fa un lustre. Per exemple en buscar un parell de sabates d'un model concret en poden sortir d'altres relacionats. En tractar-se d'una aplicació comercial es pot per exemple posicionar preferentment determinats productes.

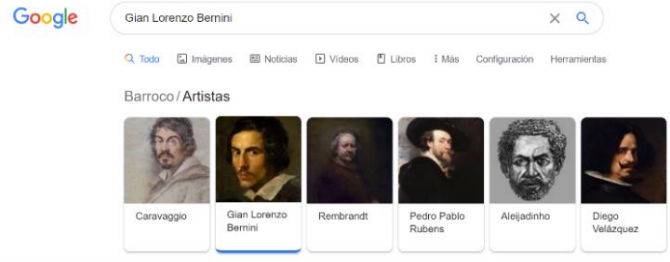

Figura 3. Google Knowledge Graph

Un altre punt interessant a destacar és *Neo4j*, aquest sistema gestor de bases de dades orientades a graf porta més de deu anys es tracta d'una de les empreses líders en el sector, algunes marques com *Volvo* o *Ebay*, *Adobe*, *Cisco* i moltes més. A més, hi ha construida una gran comunitat que combina estudiants i professionals de diferents sectors per a facilitar l'accés a material gratuït com webinars o un llibre[3] gratuït explicant en profunditat l'ús d'algorismes.

4 Implementació del sistema

4.1 Implementació del graf

L'origen de les dades en aquest projecte sempre ha estat en format digital, l'objectiu no ha estat migrar les dades de la font física, que ja havia estat realitzat en un projecte previ amb metodologia de crowdsourcing[4] gràcies a diversos transcriptors voluntaris. El procés per a desenvolupar el graf el podem definir en 5 etapes.

Primer s'ha de realitzar una anàlisis de dades i adaptació al format desitjat (en cas que sigui convenient). Si les dades es presenten en format de base de dades relacional clàssica, pot ser convenient realitzar modificacions sobre les taules com la fusió o la divisió d'aquestes per a afavorir el procés d'importació al software de Neo4j. Sempre serà més fàcil realitzar aquests canvis estructurals a la font de les dades que un cop les hem migrat. Pot ser que sigui necessari realitzar la transformació de les dades a un format que sigui compatible per a la migració. Per exemple JSON o CSV són compatibles amb la llibreria de Neo4j APOC, que incorpora algunes funcions com load.json [figura 4]. Un cop les dades es troben en el format correcte el següent pas és realitzar la importació de les dades amb les llibreries esmentades. En la metodologia aplicada en aquest cas és necessari:

- Indicar on es troba la font de les dades
- Definir un nom per al label que ha d'estar associat a un grup de nodes. En aquest cas Persones.
- Indicar quin serà l'atribut a ser l'identificador únic del node
- Establir una relació 1 a 1 amb tots els atributs que volem que hi siguin al node.

```
//Import de les dades

CALL apoc.load.json("file:///padrons_municipals.json") YIELD value as persona
CREATE (p:persona {id: p.id_padro_individu})
SET p.id_padro_individu = persona.id_padro_individu,
    p.Municipi = persona.Municipi,
    p.id_padro_llar = persona.id_padro_llar,
    p.any_padro = persona.any_padro,
    p.id_document = persona.id_document,
    p.id_individu = persona.id_individu,
    p.id_llar = persona.id_llar,
    p.id_llar_general = persona.id_llar_general,
    p.casa_carrer = persona.casa_carrer,
    p.casa_num = persona.casa_num,
    p.Noms_harmo = persona.Noms_harmo,
    p.cognom1_harmo = persona.cognom1_harmo,
    p.cognom2_harmo = persona.cognom2_harmo,
    p.nom = persona.nom,
    p.cognom_1 = persona.cognom_1,
    p.cognom_2 = persona.cognom_2,
    p.data_naix = persona.data_naix,
    p.cohort = persona.cohort,
    p.edat = persona.edat,
    p.estat_civil = persona.estat_civil,
    p.parentesc = persona.parentesc,
    p.parentesc_har = persona.parentesc_har,
    p.ocupacio = persona.ocupacio,
    p.ocupacio_hisco = persona.ocupacio_hisco,
    p.nou_dni = persona.nou_dni,
    p.tipus_l = persona.tipus_llar
```

Figura 4. Procés de creació del label Persona

El següent pas un cop realitzada la importació de totes les dades ha de ser el d'establir les relacions entre aquests nodes. La metodologia és la següent. Es realitza una consulta sobre el graf amb unes determinades condicions com que els cognoms siguin coincidents en el cas dels germans i en el cas que un parell o més de nodes siguin retornats es defineix una relació mitjançant *MERGE* o *CREATE* (*MERGE* evita crear relacions duplicades en cas que es puguin produir). Un cop hem definit les relacions més bàsiques en podem generar amb més facilitat de noves més complexes. Per exemple, un cop sabem que dos individus viuen al mateix domicili, podem deduir que són germans a partir dels seus cognoms [*Figura 5*]. I a la mateixa vegada aquesta relació de germans facilita trobar els pares i així successivament.

```
//Relació Mateixa_persona

MATCH (p1:persona),(p2:persona)
WHERE p1.nom = p2.nom
AND p1.cognom_1 = p2.cognom_1
AND p1.cognom_2 = p2.cognom_2
AND p1.any_padro < p2.any_padro
AND p1.data_naix = p2.data_naix
MERGE (p1)-[r:Mateix_Individu]->(p2)

//Relació germans // nomes els que resideixen a la maeixa casa de moment

MATCH (p1:persona)-[:Viu_a]->()<-[:Viu_a]-(p2:persona)
WHERE p1.cognom_1 = p2.cognom_1
AND p1.cognom_2 = p2.cognom_2
AND p1.any_padro = p2.any_padro
AND p1.nom <> p2.nom
MERGE (p1)-[r:Germans]->(p2)
```

Figura 5. Exemples d'establiment de relacions entre nodes

En aquest projecte s'ha definit dos tipus de nodes, 8 relacions i fins a 26 atributs.

Nodes:

- Persona
- Domicili

Es pot dir que el node persona conté el node domicili, ja que els dos labels han estat generats a partir de la mateixa Font de dades, en canvi el node domicili només conté els atributs relacionats amb el lloc de residència, municipi i carrers a part dels atributs identificadors. El propòsit d'haver-ho dissenyat així i no únicament amb el node persona és a causa del fet que un label domicili facilita la identificació de veïns i permet fer agrupacions que visualment són molt més descriptives.

Relacions:

- Casats
- Fill/Filla
- Germans
- Mare/Pare
- Mateix individu
- Veïns
- Viu a

Atributs:

Municipi, Noms_harmo, any_padro, casa_carrer, casa_num, cognom1_harmo, cognom2_harmo, co, nom_1, cognom_2, cohort, data_naix, edat, estat_civil, id, id_document, id_general, id_individu, id_llar, id_llar_general, id_padro_individu, id_padro_llar, nom, nou_dni, ocupacio, ocupacio_hisco, parentesc, parentesc_har tipus_llar.

4.2 Ús de funcions algorítmiques

En aquest apartat veurem les diferents tipologies de funcions funcions algorítmiques en estructures de graf i aplicabilitat en el context del projecte.

Community Detection: Els algorismes de detecció de comunitats agrupen el graf en funció de les relacions per trobar comunitats on els membres tenen interaccions més significatives. Les connexions revelen cúmuls estrets, grups aïllats i diverses estructures. Aquesta informació ajuda a preveure comportaments o tendències similars, a estimar la resiliència, a trobar entitats duplicades o simplement a preparar dades per a altres anàlisis.

Centrality(importància): Els algorismes de centralitat revelen quins nodes són importants basant-se en determinats aspectes de la tipologia del graf, per a descobrir els rols dels nodes individuals i el seu impacte. Dins d'aquest grup s'inclou el famós algorisme *PageRank*. Aquest algorisme s'utilitza per inferir dinàmiques de grups com ara la credibilitat, i els ponts entre grups.

Similarity: Algorismes de similitud fan ús de comparacions per obtenir una puntuació sobre cada nodes individuals en funció dels seus veïns o propietats. S'utilitza en aplicacions com ara recomanacions personalitzades i en el desenvolupament de jerarquies categòriques.

Heuristic Link Predictions: Tenen com a finalitat trobar camins més eficients o els més curts per recórrer entre nodes. En aquesta categoria s'inclouen els algorismes d'A * i Dijkstra, que s'utilitzen per avaluar rutes de molts tipus com per exemple l'encaminament de trucades o IP de menys cost.

Pathfinding & Search: Preveure la posibilitat que hi ha en que dos nodes estiguin relacionats fent ús de la proximitat dels nodes, així com d'altres elements estructurals.

4.3 Funcionalitat de la pàgina web

La finalitat principal de la pàgina web és la de poder realitzar una consulta sobre el graf amb una resposta immediata a través d'un formulari intuïtiu i evitant haver d'utilitzar llenguatge de consultes perquè persones sense coneixements de programació en puguin fer ús. A partir d'aquest punt es pot pensar funcionalitats extra per a més valor al sistema. Actualment a part del sistema de cerca, l'usuari pot enregistrar a una base de dades la consulta que ha realitzat en format *JSON*.

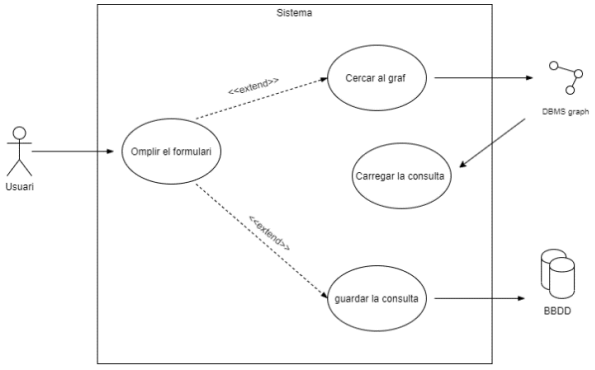

Figura 6. Diagrama de casos d'ús del sistema.

En un cas d'ús típic [*Figura 6*], l'usuari accedeix al sistema que li presentarà una interfície senzilla on hi apareix un formulari i un requadre per a la visualització del resultat de la consulta. Aquest formulari està dividit en 3 apartats. El primer i últim permeten definir dos nodes persona respectivament i l'altre permet especificar el tipus de relació que els interconnecta mitjançant un desplegable.

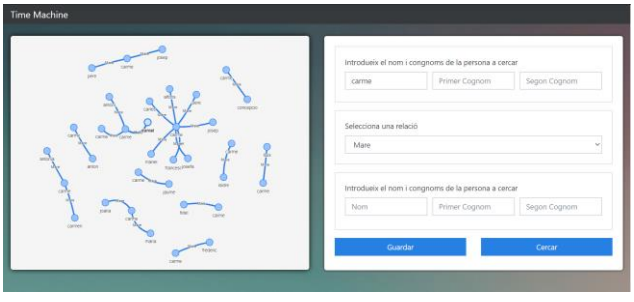

Figura 7. Exemple de consulta sobre el graf a la pàgina web.

Com podem observar a la *figura 7*, els atributs disponibles per a associar a cada node Persona són el nom i els cognoms. Aquest formulari ofereix fins a una combinatòria de 8 possibles outupts de diferents característiques. Podem enviar el formulari buit i el que ens retornarà el graf és una selecció aleatòria de 25 nodes. D'aquesta manera s'evita donar a resposta com error una consulta que tècnicament és possible, es pot prendre com un resultat aleatori. Aquesta combinatòria fa cobertura a totes les possibilitats d'imput al formulari. Podem obtenir com a retorn únicament un node, o els dos nodes amb relació, sense relació i buscar únicament una relació entre dos nodes no especificats. També és possible omplir el segon node i obviar el primer requadre encara que el resultat és el mateix a la inversa. Donar aquesta possibilitat evita errors o bé missatges d'advertència innecessaris.

Es poden realitzar tantes consultes com es vulgui sobre el graf, en cap moment la pàgina ha de fer refresc per a mostrar el graf. En un moment donat l'usuari pot decidir, per exemple, si el resultat de la consulta li ha semblat interessant i el vol guardar a la base de dades del servidor pot clicar sobre guardar en lloc de cercar.

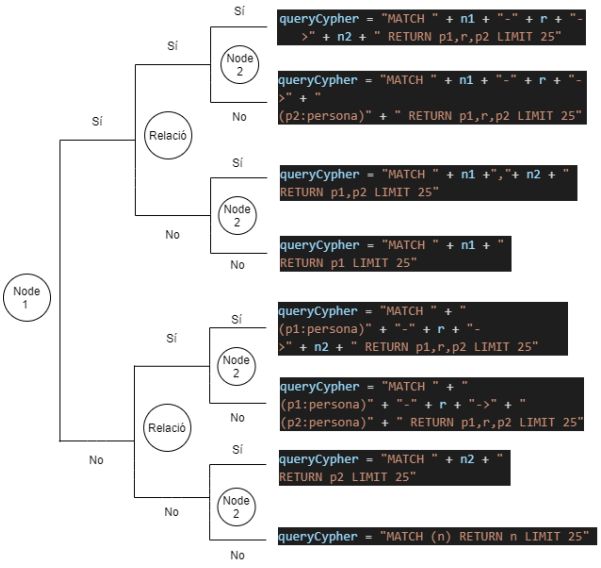

Figura 8. Combinatòria de possibles sortides en format Cypher

5 5 ANÀLISI I DISSENY ARQUITECTÒNIC

5.1 Requisits funcionals i no funcionals

En les següents taules es mostren els requisits funcionals i no funcionals que es van proposar en les primeres fases del projecte i que han guiat el desenvolupament.

ID	Descripció
RF1	El sistema ha de permetre realitzar consultes sobre una base de dades orientada a graf.
RF2	El sistema ha de permetre realitzar cerques sobre nodes.
RF3	El sistema ha de permetre realitzar cerques sobre relacions.
RF4	El sistema ha de facilitar una llista dels tipus de nodes i atributs.
RF5	El sistema ha de proporcionar una interfície gràfica d'alt nivell per a realitzar les consultes sobre el graf.

RF6	El sistema ha de proporcionar l'opció de realitzar consultes directament en el llenguatge de queries Cypher.
RF7	El sistema ha de permetre als usuaris la interacció amb el graf. Permetent l'expansió dels nodes, i la mostra dels atributs, etc.
RF8	El sistema ha de proporcionar un manual d'usuari introductori.
RF9	El sistema ha de permetre la recerca de com a mínim persones i domicilis relacionats.

ID	Descripció
RNF1	El sistema ha de retornar qualsevol consulta en un temps raonable (2s – 30s)
RNF2	La interfície del sistema ha de ser intuïtiva i permetre el seu ús a qualsevol subjecte independentment de la seva formació.
RNF3	El sistema ha de comptar amb una alta fiabilitat davant possibles pèrdues de dades.
RNF4	El sistema ha de garantir la integritat i fiabilitat de les dades.
RNF5	El sistema ha de ser capaç de suportat diverses connexions simultànies a la base de dades.
RNF6	El sistema ha de ser escalable i permetre la ingesta de gran quantitat de dades.
RNF7	El sistema ha d'estar disponible les 24 hores del dia els set dies de la setmana.
RNF8	El sistema ha de ser responsive i permetre l'adaptabilitat a diferents dispositius i resolucions de pantalla
RNF9	El sistema ha de ser completament funcional en navegadors basats en chromium i mozilla.

5.2 Arquitectura del sistema

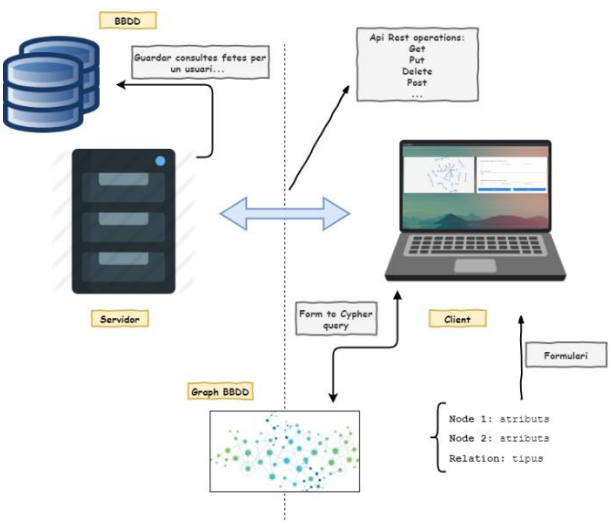

Figura 9. Diagrama del sistema

Com es pot observar a la figura 9, el servidor del sistema està desenvolupat amb *Node.js* fent ús del framework Express, aquest es comunica amb la base de dades implementada amb *Mongo DB* per a guardar les consultes realitzades fent ús de l'opció *Guardar* del formulari. És a dir, un cop l'usuari ha configurat la consulta (com s'ha comentat abans, e pot deixar buida)

Té dues opcions. Pot clicar l'opció *Cercar* el, que traduirà la consulta a *Cypher* i posteriorment serà enviada al servidor *Neo4j* on es troba allotjat el graf amb el qual el sistema s'ha connectat fent ús de la llibreria *Neovis.js*, el qual retornarà el resultat de la consulta per a ser visualitzat per pantalla. Un cop retornat el resultat el formulari no farà reset, per a poder donar opció a guardar la consulta feta al servidor mitjançant el mètode *POST* de *l'API* rest implementada al sistema, que enregistrarà la consulta en format *JSON*. La utilitat d'aquesta opció ara mateix és reduïda. Però pot donar peu a futures funcionalitats com un historial de consultes o la possibilitat d'associar imatges a retornar des de el servidor associat a un *ID*. Aquest *ID* està implementat i està format a partir d'un número aleatori en combinació amb l'hora del sistema. Com es pot veure a la figura 10 a la perició *GET* realitzada amb el programa *Postman* part de l'*ID* per defecte i les dades del formulari, també tenim un identificador únic que a més ens indica quan es va fer la consulta. Una possible utilitat d'aquest identificador és la d'esborrar l'historial de consultes d'una data concreta amb el mètode *DELETE*.

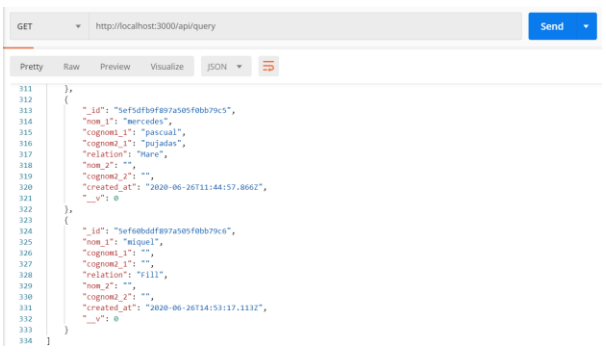

Figura 10. Petició Get que retorna les consultes emmagatzemades en format JSON a la base de dades.

6 TESTING DEL SISTEMA

Font de les dades

Les fonts de dades no han estat revisades, ja que han estat prèviament auditades. També s'ha de tenir en compte que en tractar-se en molts casos de dades de censos molt antics, ja des de la font física on es trobava documentada la informació, aquesta pot contenir errors gramaticals a l'hora descriure els noms per exemple entre altres imprecisions.

Graf

Les comprovacions realitzades en la construcció del graf s'han fet a l'hora de fer la importació de dades i en provar que en establir les relacions entre nodes no és produïssin errors, per exemple, més de dues persones amb la relació de casats. Aquestes comprovacions no han estat fetes seguint cap mena de tècnica concreta més enllà de fixar-se cada cop que es feia algun canvi en l'estructura del graf.

Pagina web

Exploratory testing - aquesta tècnica de testing és ideal en productes ja funcionals per a avaluar la resposta del sistema a determinats "atacs" o dit d'una altra manera, inputs que forcen al sistema a situacions que poden por-tar a error. Alguns dels atacs realitzats:

- Inputs amb caràcters no valids (!,4.$3"...)
- Tractar de produir overflow al formulari amb cadenes de caràcters molt llargues.
- Executar el formulari sense que hagi rebut ningun imput.
- Cercar repetides vegades sense refrescar la pàgina esperant que al repetir el mateix input en múltiples ocasions pugui ocasionar alguna mena d'error.

L'altre test realitzat ha consistit a provar totes les combinacions possibles de valors en el formulari. Per a validar que els 8 tipus possibles de quèries resultants funcionin de forma adequada (apartat 4.3). En cas que la combinatòria d'opcions hagués estat molt elevada es podria haver fet *pairwise testing*, que redueix el nombre de casos a provar únicament fent test per a cada parella possible de valors. Per exemple en una combinatòria de 3 variables amb cadascuna tres possibles valors, fent *pairwise testing* es passa de 27 a 9 casos de prova.

No s'ha realitzat test unitari sobre els mòduls programats, ja que per les característiques del projecte, com que no és massa gran ni comptar amb un gran nombre de classes s'ha vist prou amb el test exploratori i la comprovació de combinacions possibles al formulari. S'utilitzen molts mòduls predefinits que aporten fiabilitat al sistema. Les funcions que poden fallar més són els *JavaScript* del front end per a la connexió amb el graf i per a traduir el contingut del formulari a *Cypher*, la seva fiabilitat s'ha provat mitjançant testing d'atacs.

7 Conclusió

S'ha vist el gran potencial de les bases de dades orientades a graf i la gran aplicabilitat que tenen a l'hora d'analitzar un conjunt de dades. La cerca de noves relacions que puguin aportar una informació que en models relacionals clàssics i més si es fa ús de les llibreries que proporciona el gestor de bases de dades *Neo4j* que en permet anar un pas més enllà i identificar patrons en grans volums de dades. També és important remarcar la importància de les actuals alternatives en el desenvolupament de portals web com l'ús de Node.js sumat a la gran quantitat de frameworks i llibreries disponibles que faciliten en gran manera la integració de diferents tecnologies de manera modular. Aquesta tasca hauria estat més difícil sens dubte amb un servidor implementat en *PHP* per exemple. El projecte té el potencial per a seguir creixent. Millorar l'apartat de cerca, incorporar nous sets de dades, establir noves relacions són alguns exemples.

8 Agraïments

Al meu tutor Josep Lladós, per a donarme la possibilitat de realitzar aquest projecte i ajudarme en tot moment. També agrair a l'Arxiu comarcal del Baix Llobregat de la Generalitat de Catalunya, per proporcionar les fonts originals i al centre d'Estudis demogràfics Catalans per haver validat les dades.

Bibliografia

[1] Time Machine Europe, https://www.timemachine.eu , © 2020 Time Machine Organisation Visitat el 29.06.2020
[2] The Neo4j Operations Manual v4.1, © 2020 Neo4j, Inc.
[3] Graph Algorithms: Practical Examples in Apache Spark and Neo4j, By Mark Needham & Amy E. Hodler, © 2020 Neo4j, Inc
[4] Crowdsourcing, ¿que es?, https://www.ge-desco.es/blog/crowdsourcing-que-es-y-como-funciona/, Visitat el 29.06.2020
[5] Census Knowledge Graph, Oriol Ramos Terrades
[6] APOC User Guide 4.0, https://neo4j.com/docs/labs/apoc/current/, Visitat el 29.06.2020
[7] Comprehensive Guide to Graph Algorithms in Neo4j, © 2018 Neo4j. All rights reserved.
[8] Import: RDBMS to Graph, https://neo4j.com/developer/relational-to-graph-import/, Visitat el 29.06.2020
[9] The Neo4j Cypher Manual v4.1, © 2020 Neo4j, Inc.
[10] cypher-refcard-4.1, © 2020 Neo4j, Inc.
[11] Npm, https://www.npmjs.com, Visitat el 29.06.2020
[12] Tecnologia i innovació ciutadana en la construcció de xarxes socials històriques per a la comprensió del llegat demogràfic, Alicia Fornés, Josep Lladós, Centre de Visió per Computador – UAB, Joana M. Pujades, Anna Cabré, Centre d'Estudis Demogràfics – UAB, 08.06.2017
[13] Node.js Guides, https://nodejs.org/en/docs/guides/, © OpenJS Foundation, Visitat el 29.06.2020
[14] Webpack documentation, https://webpack.js.org/concepts/, Visitat el 29.06.2020
[15] Introducing the Knowledge Graph: things, not strings, https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html, Visitat el 29.06.2020
[16] Bootstrap4 guide, https://www.w3schools.com/bootstrap4/default.asp, w3schools 1999-2020, Visitat el 29.06.2020

APÈNDIX

A1. EXEMPLES DE CONSULTES REALITZADES MITJANÇANT EL FORMULARI DEL PORTAL WEB

Les dades en detall de les cerques realitzades no es mostraran per a garantir la protecció de dades.

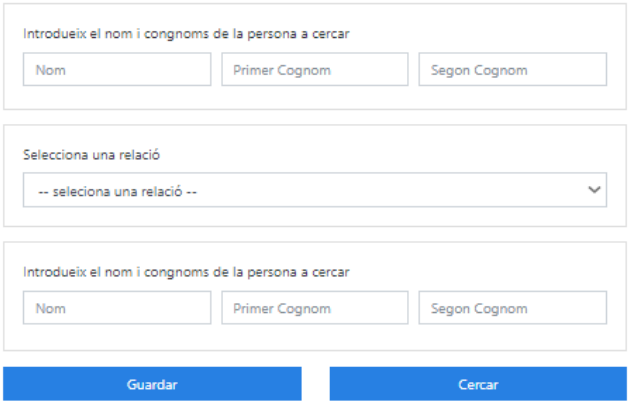

Figura Ap 1. Formulari vist en detall.

Figura Ap 2. Cerca amb el formulari buit formulari.

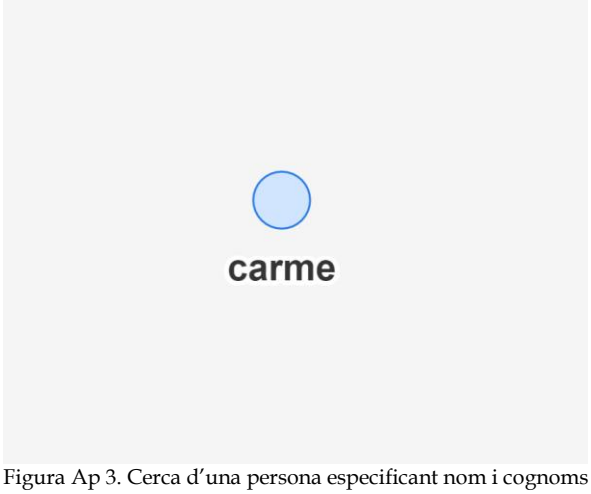

Figura Ap 3. Cerca d'una persona especificant nom i cognoms

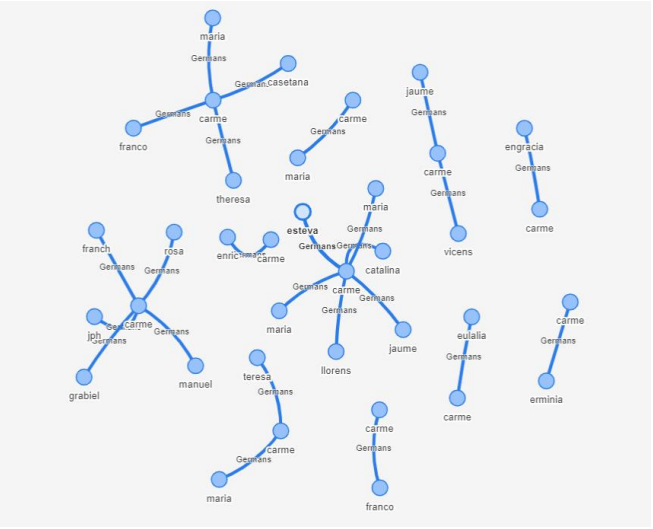

Figura Ap 4. Cerca de persones amb nom *Carme* que tenen germans.

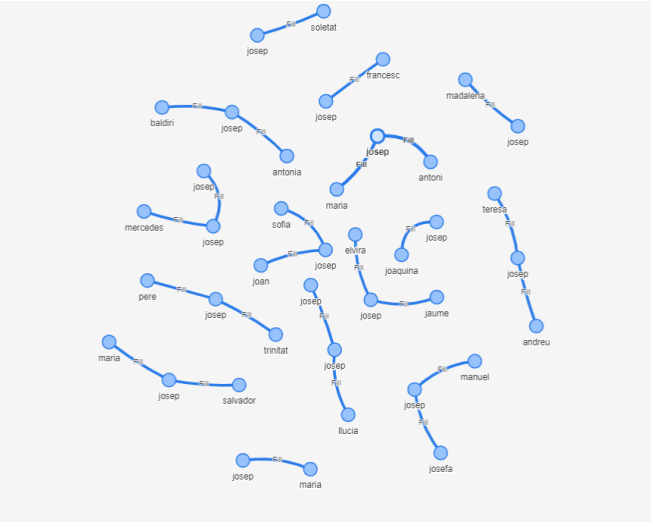

Figura Ap 5. Cerca dels pares de persones amb nom *Josep* .

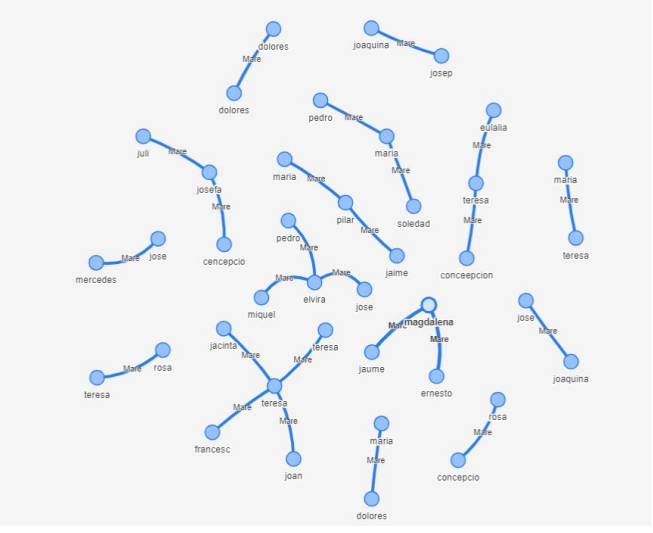

Figura Ap 6. Cerca de persones que son mares sense especificar nom o cognoms.