Magatzems de dades en el context de la web semàntica

Mònica Mañá Aragay
Enginyeria Tècnica Informàtica de Gestió

Consultor
Joan Anton Perez Braña

08/01/2013

FITXA DEL TREBALL FINAL

Títol del treball:	**Magatzems de dades en el context de la web semàntica**
Nom de l'autor:	Mònica Mañá Aragay
Nom del consultor:	Joan Anton Perez Braña
Data de lliurament (mm/aaaa):	12/2012
Àrea del Treball Final:	XML i Web Semàntica
Titulació:	Enginyeria Tècnica Informàtica de Gestió

Resum del Treball (màxim 250 paraules):

L'objectiu d'aquest treball és estudiar els sistemes de gestió de bases de dades (SGBD) utilitzats en el context de la web semàntica en general i analitzar-ne un en particular.

Per una banda es realitza un estat de l'art rigorós dels SGBD utilitzats en la web semàntica (VirtuosoDB, OWLIM, eXist...), s'identifica què els diferencia d'altres SGBD tradicionals, perquè serveixen, com utilitzar-los, i les limitacions i avantatges que presenten en vers els SGBD tradicionals.

Per altra banda, es realitza un anàlisi a fons de Virtuoso DB, descrivint les seves funcionalitats, com emmagatzema la informació, com afegir, consultar i modificar informació semàntica, com importar-hi ontologies, relacionar dades semàntiques, i se'n realitza proves de rendiment.

Al finalitzar el treball s'han assolit els objectius principals: descripció exhaustiva dels components del web semàntic, anàlisi dels sistemes SGBD utilitzats en web semàntica, descripció dels mètodes de consulta i modificació de les dades, i anàlisi de Virtuoso Opensource, instal·lació i configuració i indexat de camps de text lliure.

Abstract (in English, 250 words or less):

The aim of this work is to study the database management systems used in the context of the Semantic Web, and to analyze one of them.

Firstly, state of art of the DBMS used in the semantic web (like VirtuosoDB, OWLIM, eXist...) is realized, identifying what differentiates them from traditional DMBS, how to use it, what are their functionalities, and which are its limitations and advantages in front on the traditional ones.

Secondly, an analysis of VirtuosoDB is realized, describing its functionalities, how it stores information, how to add, consult and modify semantic information, how to use ontology, describe semantic data, and finally there are reported some performance tests and its results.

On having finished this work the principal objectives have been reached: exhaustive description of the components of the semantic web, analysis of the systems SGBD used in semantic web, description of the methods of consultation and modification of the information and analysis of Virtuoso Opensource: installation and configuration and free text indexation.

Paraules clau (entre 4 i 8):

Web Semàntica
SGBD Semàntic
Virtuoso Opensource
RDF

Índex

Llista de figures

1. Introducció

1.1 Context i justificació del Treball

Aquest Treball es du a terme com a treball de fi de carrera amb l'objectiu de posar en pràctica els coneixements adquirits en diferents assignatures.

El TFC té com a objectiu principal mostrar l'assoliment d'aprenentatge que s'ha dut a terme al llarg dels estudis de l'Enginyeria Tècnica en Informàtica de Gestió.

Aquests objectius es concreten en:

- Analitzar un problema complex de tipus pràctic transformant-lo en un projecte informàtic.

- Planificar i estructurar el desenvolupament del projecte mitjançant l'elaboració d'un pla de treball aplicant una metodologia adient.

- Treballar a fons els aspectes formals del desenvolupament de projectes.

- Sintetitzar una solució viable i realista al problema proposat.

- Elaborar una memòria del projecte segons una estructura prefixada.

- Elaborar una presentació del desenvolupament i resultats finals del projecte.

1.2 Objectius del Treball

L'objectiu d'aquest treball és estudiar els magatzems de dades utilitzats en el context de la web semàntica en general i analitzar-ne un en particular.

Per una banda es realitzarà un estat de l'art rigorós dels SGBD utilitzats en la web semàntica (VirtuosoDB, OWLIM, eXist...), s'identificarà què els diferencia d'altres SGBD tradicionals, perquè serveixen, com utilitzar-los, limitacions i avantatges en vers de SGBD tradicionals...

D'altra banda, es seleccionarà un dels SGBD estudiats (en aquest cas Virtuoso Opensource) i se'n farà un anàlisi a fons, descrivint les seves funcionalitats, com emmagatzema la informació, com afegir, consultar i modificar informació semàntica, com importar-hi ontologies, relacionar dades semàntiques, realitzar proves de rendiment, indexat de text lliure, etc.

El treball es realitzarà en les següents fases:

- Estudi inicial i elecció de la base de dades a estudiar (Virtuoso Opensource o OWLIM) – En el nostre cas la opció escollida ha estat Virtuoso Opensource, descartant OWLIM de l'estudi.

- Instal·lació del programari necessari

- Anàlisi funcional dels magatzems de dades utilitzats en la web semàntica

- Selecció dels jocs de dades i elaboració de les proves de rendiment

- Obtenció de les conclusions

- Elaboració de la memòria i de la presentació

1.3 Enfocament i mètode seguit

Aquest treball es realitza a partir d'un estudi exhaustiu dels components del web semàntic. Partint d'aquesta base, s'estudien les diferents estratègies per a l'emmagatzematge d'aquests components i també les alternatives per a l'explotació de les dades en el context semàntic.

Finalment, s'estudien els diferents SGBD semàntics que actualment hi ha al mercat, i se'n analitza un en concret. Aquest anàlisi comporta la tria d'un joc de dades, la càrrega d'aquest a la base de dades i mitjançant l'aprofundiment en els llenguatges de consulta específics, la realització de consultes i l'anàlisi de rendiment del sistema.

1.4 Planificació del projecte

En aquest apartat es descriu la planificació temporal d'aquest treball, incloent una periodificació de les tasques a realitzar, un graf de precedències indicant l'ordre d'aquestes en el temps, una descripció de les fites i el diagrama de Gantt.

1.4.1 Temporització i fites

En aquest punt es fa un estudi de cadascuna de les tasques que s'han de portar a terme, el temps disponible i, en funció d'aquestes dades i de les limitacions temporals que presenta aquest, s'estableix la planificació i es marquen les fites que s'hauran de complir.

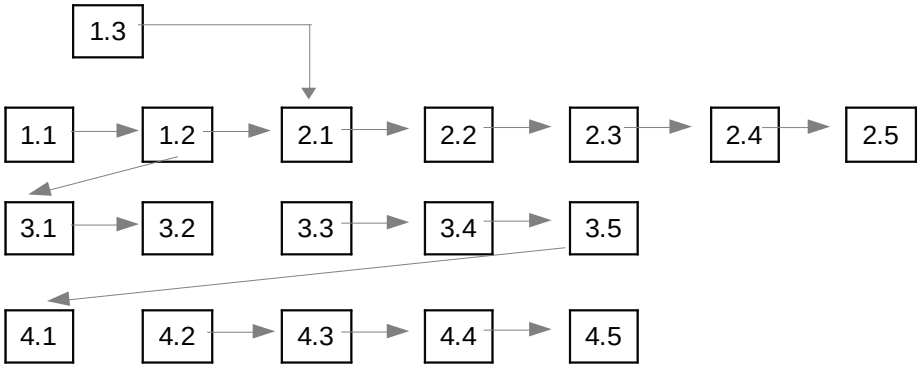
Les tasques es defineixen en funció dels diferents lliuraments que marca el pla d'estudis de l'assignatura. Dins de cada lliurament, s'avalua la càrrega de feina que serà possible desenvolupar, amb el que queden definits els sub apartats.

1.4.2.1 Tasques

Tasca	**Precedents**
1. Definició del projecte	
1.1 Descàrrega informació de l'aula: enunciat del projecte, pla de treball d'exemple, TFC exemple	
1.2 Lectura de la documentació descarregada i lectures recomanades pel consultor	1.1
1.3 Reunió d'inici	
2. Pla de treball	
2.1 Recerca informació, elecció del tema del TFC	1.3
2.2 Elaboració del pla de treball, avaluació de les tasques a realitzar	2.1
2.3 Temporització de les tasques al pla de treball	2.2
2.4 Lliurament PAC1 – Pla de treball	2.3
2.5 Correcció PAC1 amb les correccions del consultor	2.4
3 Inici PAC2	
3.1 Lectura acurada de tota la informació sobre els SGBD utilitzats en el context de la web semàntica (VirtuosoDB, OWLIM, eXist...)	1.2
3.2 Estudi de les SGBD anteriors, diferències amb les tradicionals, usos, limitacions i avantatges	3.1
3.3 Instal·lació del SGBD escollit	
3.4 Elecció del conjunt de metadades i les seves instàncies, obtenció dels XML, del conjunt de dades amb què es realitzaran les proves	3.3
3.5 Elaboració i entrega PAC2	3.4
4. Inici PAC3	

4.1 Refinament de la memòria amb la retroacció rebuda de la PAC2, elaboració de les conclusions finals	3.5
4.2 Elaboració del vídeo de presentació	
4.3 Lliurament primera versió de la memòria	4.2
4.4 Refinament de la memòria amb la retroacció rebuda, revisió continguts, redacció, ortografia	4.3
4.5 Lliurament final i presentació	4.4
5. Debat	

1.4.2.2 Gràfic de precedències



1.4.2.3 Fites

Durant el desenvolupament d'aquest projecte hi ha una sèrie de lliuraments parcials prèviament definits.
A banda d'aquestes fites, cal afegir d'altres accions que tot i no estar planificades formalment ni tenir una data fixada, cal tenir presents en aquest document.

Aquestes fites són:

- Assistència a les reunions que programi el consultor
- Lliuraments d'esborranys de les PAC per tal de garantir que els lliuraments oficials compleixen tots els requisits necessaris, obtenint del consultor els comentaris i correccions pertinents. Es procurarà realitzar aquests lliuraments com a mínim una setmana abans de la data del lliurament oficial.
- Lliuraments oficials de les PAC definides al pla d'estudis
- Debat final

Les dates previstes per a aquests lliuraments son:

Fita	**Data**
Reunió d'inici	25/09/12
Lliurament PAC1	01/10/12
Esborrany PAC2	29/10/12
Lliurament PAC2	05/11/12
Esborrany PAC3	10/12/12
Lliurament PAC3	17/12/12
Lliurament presentació i memòria	08/01/13
Inici debat	Sense data
Fi debat	Sense data

1.4.2.4 Calendari

Tenint en compte la disponibilitat horària condicionada a obligacions laborals i familiars, es dedicarà a aquest treball una mitja de 2h30 hores diàries de dilluns a divendres a diumenge. Aquesta dedicació es podrà ampliar en cas que sigui necessari i sempre que sigui possible.

Això suposarà per mesos aquesta dedicació:

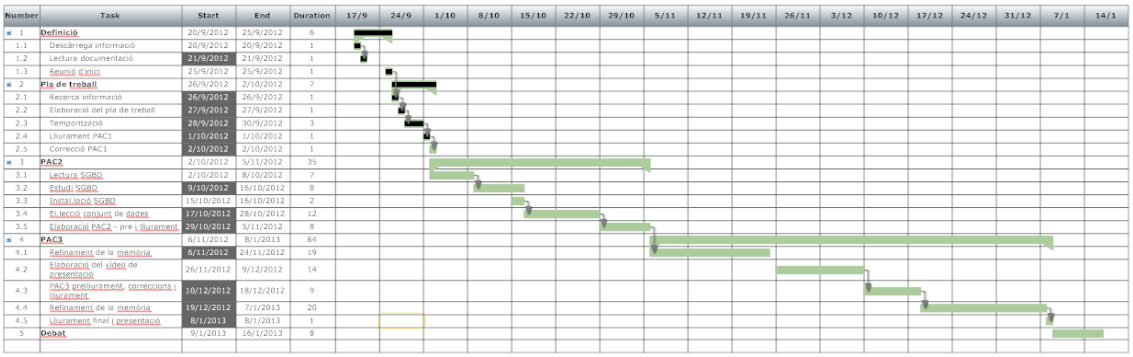
Setembre (19 a 30)	18 hores
Octubre	77 hores
Novembre	75 hores
Desembre	80 hores
Gener	40 hores
Total	290 hores

1.4.2.5 Planificació

Tasca	**Dates**	**Hores**
1. Definició	19/09	
1.1 Descàrrega informació	20/09	1
1.2 Lectura de la documentació	21/09	5
1.3 Reunió d'inici	25/09	2
2. Pla de treball	26/09-01/10	
2.1 Recerca informació	26/09	4
2.2 Elaboració del pla de treball	27/09	5

2.3 Temporització	28/09-30/09	5
2.4 Lliurament PAC1	01/10	
2.5 Correcció PAC1	02/10-10/10	3
3 PAC2	02/10-5/11	
3.1 Lectura SGBD	03/10-08/10	20
3.2 Estudi de les SGBD	08/10-15/10	20
3.3 Instal·lació del SGBD	15/10-16/10	5
3.4 Elecció del conjunt dades	17/10-28/10	42
3.5 Elaboració i prelliurament PAC2 – lliurament PAC2	28/10-05/11	30
4. Inici PAC3	06/11	
4.1 Refinament de la memòria	07/11-25/11	44
4.2 Elaboració del vídeo de presentació	26/11-09/12	40
4.3 Prelliurament PAC3, correccions i lliurament definitiu PAC3	09/12-17/12	24
4.4 Refinament de la memòria	18/12-07/01	40
4.5 Lliurament final i presentació	8/01	
5. Debat	-	

1.4.2.6 Diagrama de Gannt



Number	Task	Start	End	Duration
1	**Definició**	20/9/2012	25/9/2012	6
1.1	Descàrrega informació	20/9/2012	20/9/2012	1
1.2	Lectura documentació	21/9/2012	21/9/2012	1
1.3	Reunió d'inici	25/9/2012	25/9/2012	1
2	**Pla de treball**	26/9/2012	2/10/2012	7
2.1	Recerca informació	26/9/2012	26/9/2012	1
2.2	Elaboració del pla de treball	27/9/2012	27/9/2012	1
2.3	Temporització	28/9/2012	30/9/2012	3
2.4	Lliurament PAC1	1/10/2012	1/10/2012	1
2.5	Correcció PAC1	2/10/2012	2/10/2012	1
3	**PAC2**	2/10/2012	5/11/2012	35
3.1	Lectura SGBD	2/10/2012	8/10/2012	7
3.2	Estudi SGBD	9/10/2012	16/10/2012	8
3.3	Instal.lació SGBD	15/10/2012	16/10/2012	2
3.4	Elecció conjunt de dades	17/10/2012	28/10/2012	12
3.5	Elaboració PAC2 - pre i lliurament	29/10/2012	5/11/2012	8
4	**PAC3**	6/11/2012	8/1/2013	64
4.1	Refinament de la memòria	6/11/2012	24/11/2012	19
4.2	Elaboració del vídeo de presentació	26/11/2012	9/12/2012	14
4.3	PAC3 prelliurament, correccions i lliurament	10/12/2012	18/12/2012	9
4.4	Refinament de la memòria	19/12/2012	7/1/2013	20
4.5	Lliurament final i presentació	8/1/2013	8/1/2013	1
5	**Debat**	9/1/2013	16/1/2013	8

1.5 Breu sumari de productes obtinguts

Al final de la realització d'aquest TFC s'obtindrà la memòria i el vídeo explicatiu del treball realitzat. A la finalització d'aquest treball no s'obtindrà cap aplicatiu, ja que l'objectiu d'aquest és realitzar proves de rendiment amb un magatzem de dades semàntic. Per tant, es realitzarà la instal·lació de Virtuoso DB i es faran proves de rendiment sobre l'indexat de camps de text lliure d'un conjunt de metadades de gran volum.

1.6 Breu descripció dels altres capítols de la memòria

En aquest capítol es detalla la resta d'apartats que contindrà aquesta memòria, en funció dels objectius plantejats en el capítol anterior, i la previsió de planes que tindrà cadascun dels punts considerats.
Aquests apartats són:

1. Introducció: Es realitza una breu introducció dels conceptes i els components de la web semàntica. Dins d'aquests components s'analitza el llenguatge de descripció RDF, la sintaxi RDF/XML i el llenguatge de marcat OWL.

2. Estat de l'art dels SGBD utilitzats en la web semàntica: S'analitzen les diferents alternatives en SGBD utilitzats en la web semàntica. S'estudien diferents sistemes i filosofies d'emmagatzematge i recuperació d'aquest tipus de dades. En concret, s'analitza com maneguen XML els diferents SGBD, perquè parlem de magatzems de dades, com operen amb aquestes dades els diferents sistemes (orientats a objectes, relacionals, nadius...).S'estudien les tècniques d'emmagatzematge, manipulació i obtenció de dades dels magatzems semàntics.

3. Diferències amb els SGBD tradicionals: S'enumeren les diferències entre els sistemes tradicionals i els sistemes d'emmagatzematge utilitzats en la web semàntica.

4. Ús dels SGBD: Es descriuen els usos més habituals d'aquest tipus de bases de dades i s'analitzen diferents tècniques per a realitzar-los.

5. Limitacions i avantatges: En aquest apartat s'enumeren les limitacions i els avantatges dels diferents models de dades relacionant-los amb els diferents SGBD.

6. Anàlisi: En aquest apartat de la memòria s'analitza un SGBD semàntic concret: Virtuoso, es descriuen els jocs de proves utilitzats

(metadades) i les operacions realitzades i s'analitzen els resultats obtinguts.

7. Conclusions: Per a finalitzar aquesta memòria, s'extreuen una sèrie de conclusions derivades de tot l'estudi anterior

2. Introducció a la web semàntica

El World Wide Web (WWW) ha revolucionat en els últims anys la manera de comunicar-nos entre nosaltres, la manera com es propaga la informació, la forma com arriba als seus destinataris i també les formes de negoci.

Dins el WWW, la majoria de documents i continguts que s'hi poden trobar són destinats al consumidor "humà", ja que el ús majoritari del Web és actualment fet per persones que cerquen, comparteixen i fan ús de la informació, fent compres, revisant catàlegs *on-line*, comunicant-se amb altres persones...

La majoria d'aquestes activitats es desenvolupen a través d'enginys de cerca (Google, Yahoo, Altavista...) que actualment tenen les següents limitacions:[1]

- Baixa precisió en la recerca, gran volum de respostes: tot i que es retornen els documents més rellevants que compleixen els criteris de la cerca, també se'n retornen a l'usuari milers de menys rellevants o totalment irrellevants

- Baix volum de respostes: Sovint no hi ha resposta a la petició de l'usuari, o aquestes pàgines rellevants no es retornen, tot i que aquest problema és menys comú amb els cercadors actuals

- Els resultats són poc sensibles al vocabulari: Sovint la cerca inicial no retorna els resultats que esperava l'usuari donat que els documents usen diferent terminologia que la consulta. Consultes semànticament semblants haurien de retornar resultats similars

- Els resultats retornen una única pàgina Web: si l'usuari necessita informació que és distribuïda en varis documents, ha d'iniciar vàries consultes per a recopilar els diferents documents, extreure'n la informació parcial de cadascun d'ells i fer-ne una composició.

El principal obstacle per a donar un millor suport als usuaris Web és que el contingut d'aquest no és accessible pel mateix maquinari (*machine-accessible*), en termes d'interpretació de sentències, extracció d'informació útil pels usuaris, etc.

Una de les alternatives per a millorar aquests aspectes és utilitzar la Web Semàntica, que neix com una ampliació de la web amb l'objectiu de que els sistemes d'organització del coneixement permetin:

- Una organització d'aquest coneixement en espais conceptuals d'acord amb el seu significat

[1] Grigoris Antoniou and Frank van Harmelen, *A Semantic Web Primer* (London: The MIT Press, 2004)

- Unes eines que suportin el manteniment evitant inconsistències i capaces d'extreure nou coneixement

- Cerques basades en preguntes gramaticalment correctes: el coneixement ha de ser cercat, extret i presentat d'una manera amigable a l'usuari

- Cerques basades en múltiples documents

- Definir permisos sobre parts de l' informació , enlloc de sobre quins documents.

En aquest treball es descriu quin és l'estat actual dels Sistemes Gestors de Bases de dades que implementen el model semàntic, és a dir, quins dels aspectes enumerats anteriorment s'han consolidat en aquests sistemes i com funcionen. A més, s'analitza un SGBD concret, Datalink VirtuosoDB, i també es realitza una aproximació al llenguatge de consulta SPARQL que s'utilitzarà per a obtenir informació del magatzem de dades.

2.1 Components de la web semàntica

Els principals components de la web semàntica són els metallenguatges i els estàndards de representació XML, XML Schema, RDF, RDF Schema, OWL i SPQARQL.

- XML (Extensible Markup Language): és un llenguatge de marques desenvolupat per el World Wide Web Consortium (W3C). Proporciona una sintaxi bàsica per a l'estructura del contingut dins els documents sense associar restriccions semàntiques sobre el significat del contingut. XML té una relació amb les meta dades les quals són dades estructurades que descriuen el significat dels recursos com el contingut, la qualitat i la condició dels mateixos, per exemple: aplicant meta dades a una pàgina web, permet que aquest recurs tingui atributs com <Tema>, <Autor>, <Data de publicació> la qual cosa facilita la formulació de preguntes més concretes als motors de cerca. Per a aplicar aquest tipus de meta dades és necessari aplicar RDF, ja que XML aporta la sintaxi superficial pels documents estructurats, però no els dota de cap restricció sobre el significat.

- XML Schema: és una extensió del XML que permet proporcionar i dictar certes restriccions sobre el contingut dels recursos o continguts disponibles definint la sintaxi respecte l'ordre, tipus de dades i formats dins d'un document XML. Defineix l'estructura dels documents XML.

- RDF (Resource Description Framework): és un model de dades que

permet la definició d'ontologíes i metadades que es refereixen als objectes i les seves relacions. Es basa en una estructura bàsica, construïda per dos nodes (un subjecte i un objecte) els quals són units per un arc que és el predicat. La sintaxi d'un model desenvolupat en RDF. La sintaxi d'un model desenvolupat em RDF es pot representar mitjançant XML. RDF proporciona informació descriptiva simple sobre els recursos que hi ha a la Web i que s'utilitzarà per exemple, en catàlegs de llibres, directoris, col·leccions de fotos, esdeveniments...

- RDF Schema: és una extensió semàntica de RDF que proporciona un vocabulari per a descriure les propietats i les classes dels recursos RDF, permet la definició jeràrquica de classes, objectes, relacions entre classes i propietats, restriccions de domini i rang sobre les propietats.

- OWL (Web Ontology Language): és una extensió del RDF, amb totes les característiques d'aquest i d'altres d'addicionals, com l'aplicació d'operacions lògiques, la descripció detallada de les propietats i les classes com la cardinalitat, la simetria , transitivitat o relacions inverses. Proporciona un llenguatge per a definir ontologies estructurades que poden ser utilitzades a través de diferents sistemes. Les ontologies que s'encarreguen de definir els termes utilitzats per a descriure i representar una àrea de coneixement són utilitzades pels usuaris, les bases de dades i les aplicacions que necessiten compartir informació específica dins un camp de coneixement determinat (finances, medicina, esports...). Les ontologies inclouen definicions de conceptes bàsics en un camp determinat i la relació entre ells. Les ontologies es basen en un domini per a enumerar-ne les entitats i les relacions per a formular un esquema conceptual amb la finalitat de facilitar la comunicació i el intercanvi d'informació entre els diferents sistemes i entitats.

- SPARQL: és un protocol i un llenguatge de consulta que permet fer cerques sobre els recursos de la web semàntica utilitzant diferents fonts de dades. SPARQL és un llenguatge estandarditzat per a consultes de grafs RDF.

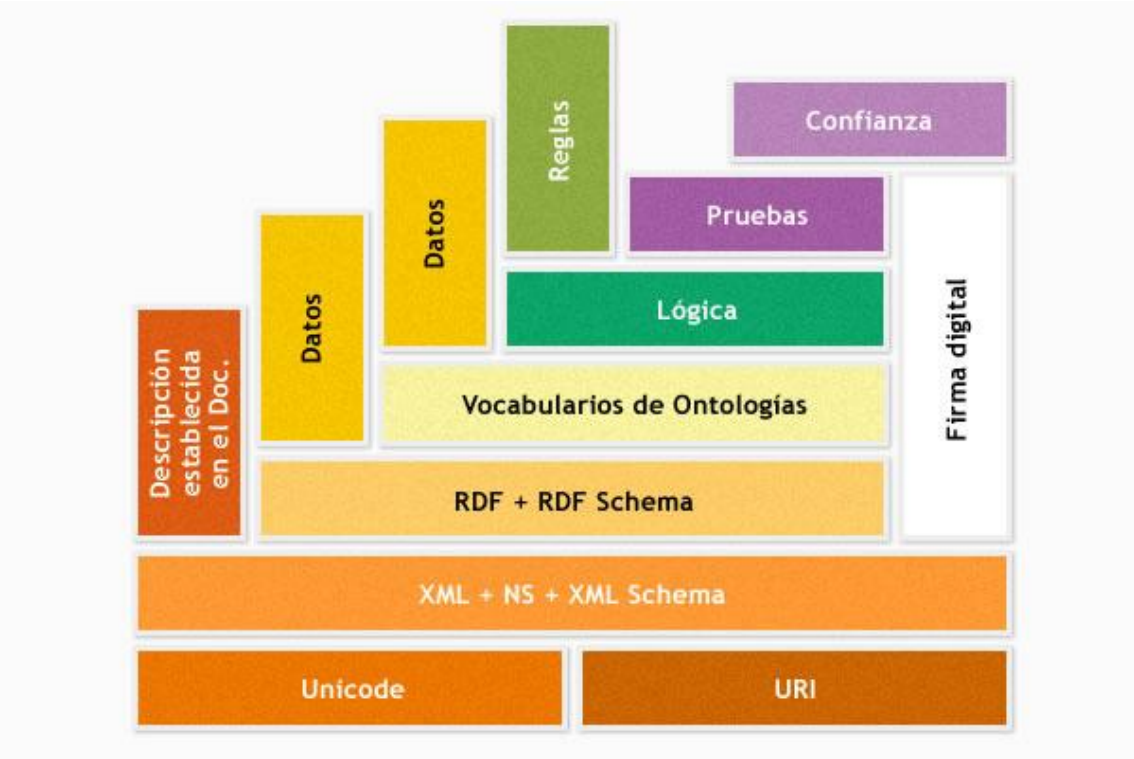


Figura 1 : Components de la Web Semàntica
Font: http://blog.evoit.com/2011/08/8-preguntas-sobre-la-web-semantica/

Les capes primàries d'aquesta tecnologia estan elaborades amb els codis URI (que tenen la funció d'identificar els recursos XML), Unicode y Namespace. Mitjançant RDF i RDF Schema s'afegeix la informació descriptiva, mentre les ontologies defineixen la relació entre les meta dades. Les capes superiors les formen les regles i la lògica que permet que els agents automàtics processin informació extremadament complexa (Cacheiro y Lago, 2008).

2.1.1 XML

Cada element dins un document XML és assenyalat amb una etiqueta. Aquesta etiqueta és una marca feta al document, que assenyala aquesta part d'informació i la dota d'un sentit clar i definit.

Els documents XML per tal de poder ésser analitzats per a qualsevol *parser* cal que siguin vàlids, és a dir, ben conformats. Han de seguir doncs una sèrie de convencions com són:

- Han de tenir una estructura jeràrquica.
- Només poden tenir una arrel o element inicial
- Els valors dels atributs han d'estar tancats entre cometes simples o dobles.
- Han de respectar majúscules i minúscules.

- Les construccions com etiquetes, referències d'entitat i declaracions s'anomenen marques: són parts del document que el processador XML ha d'entendre.

En aquest exemple s'observen les diferents etiquetes i l'estructura d'un document XML [35]:

```
<?xml version="1.0"?>
<PELICULA nombre="El Padrino" año=1985>
<PERSONAL>
</DIRECTOR nombre="Georgie Lucar">
</INTERPRETE nombre="Marlon Brando" interpreta-a="Don Corleone">
</INTERPRETE nombre="Al Pacino" interpreta-a="Michael Corleone">
</PERSONAL>
</ARGUMENTO descripción="Pelicula de mafias sicilianas en Estados Unidos">
</PELICULA>
```

XPath i XQuery

XPath i XQuery són estàndards per a accedir obtenir dades d'un document XML.

XPath és un llenguatge per a referir-se a camins dins un XML sempre tenint en compte que la informació a XML és semi estructurada en forma d'arbre. XQuery utilitza expresions XPath per a accedir a determinades parts del document XML.

XQuery és un llenguatge més complert, amb funcionalitats similars a SQL, tot i que inclou algunes capacitats de programació. Mitjançant XQuery es pot extreure i manipular informació de documents XML o de qualsevol font de dades que sigui representada mitjançant XML, com poden ser bases de dades relacionals o bé documents ofimàtics. XQuery no permet actualitzar els documents XML ni permet realitzar cerques textuals.

2.1.2 XML Schema

El propòsit del XML Schema és definir l'estructura dels documents XML que estiguin assignats a tal esquema i els tipus de dades vàlids per a cada element i atribut.

A partir del moment que es restringeix el contingut dels fitxers XML, es fa possible l'intercanvi d'informació entre aplicacions amb seguretat.

Els tipus de dades tenen al XML Schema la funció que tenien les classes en la POO. L'usuari por construir tipus de dades a partir de tipus predefinit, mitjançant mecanismes d'extensió semblants a l'herència. Els tipus de dades es classifiquen segons els atributs i elements que contenen entre tipus simples (tipus predefinits en XML, tipus List i Union) i complexes (tipus que tenen elements i/o atributs).

El XML Schema Definition Language (XSLD) és un estàndar del W3C que permet descriure l'estructura i el contingut d'un document XML. Es tracta d'una tecnología similar a Document Type Definition (DTD) però més potent i versàtil.

Una DTD és una definició en un document XML que especifica restriccions a la seva estructura i sintaxi i que descriu els elements (quines etiquetes són permeses i el seu contingut), la seva estructura (l'ordre de les etiquetes dins el document) i l'aniuament (quines etiquetes van dins d'altres).

Els avantatges de XML Schema vers DTD són [43]:

- Tipatge fort: XML Schema permet una definició de tipus més concreta que DTD
- XML Schema permet utilitzar tipus de dades similars als utilitzats en els llenguatges de programació i bases de dades, i permet també tipus definits per l'usuari a partir d'aquests.
- Permet controlar el nombre de repeticions o ocurrències d'un element en un valor concret.
- Un XML Schema és un document XML amb el que es poden emprar totes les eines desenvolupades per a treballar en XML, mentre que un DTD s'escriu en una notació sense sintaxi XML.
- XML Schema permet la representació de claus primàries i forànies.

Exemple [36]:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="Libro">
      <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="Título" type="xsd:string"/>
            <xsd:element name="Autores" type="xsd:string" maxOccurs="10"/>
            <xsd:element name="Editorial" type="xsd:string"/>
        </xsd:sequence>
        <xsd:attribute name="precio" type="xsd:double"/>
      </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

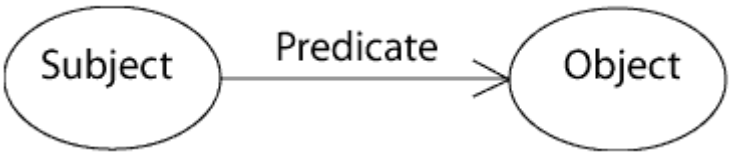
2.1.3 RDF, triplets de coneixement

Tal i com s'ha descrit en l'apartat anterior (2.2 Components de la web semàntica), RDF és un model estàndard amb l'objectiu general de representar la informació de la web.

El desenvolupament d'aquest model neix motivat per al seu ús en metadades web, aplicacions que requereixen informació en models oberts, informació *machine accessible*, interrelació entre aplicacions i procés automatitzat de la informació.

El disseny RDF pretén obtenir un model senzill amb una semàntica formal i inferència demostrable, que utilitzi una sintaxi basada en XML amb suport a tipus de dades XML Schema i que permeti que qualsevol pugui fer declaracions sobre qualsevol recurs.

L'estructura d'una expressió RDF és un conjunt de triples, cadascuna consistent en un subjecte, un predicat i un objecte (triplets). Un conjunt de triplets és un graf RDF i es pot il·lustrar de la següent manera:



Els subjectes, predicats i objectes en RDF són simples noms, o maneres d'anomenar les coses. Poden ser coses concretes (per exemple, casa_meva) o conceptes abstractes com (és_a). Aquests noms no tenen estructura interna o significat per sí mateixos, són com noms propis o variables, que es poden escollir lliurement sempre es que d'ençà es facin servir constantment. Els predicats són sempre relacions entre dues coses i les coses denotades per noms s'anomenen recursos, entitats o nodes. Això permet representar declaracions senzilles sobre recursos com un graf de nodes i arcs que representen els recursos, les seves propietats i valors.

Un node pot ser un URI, un literal o un blanc. Les propietats són referències URI.

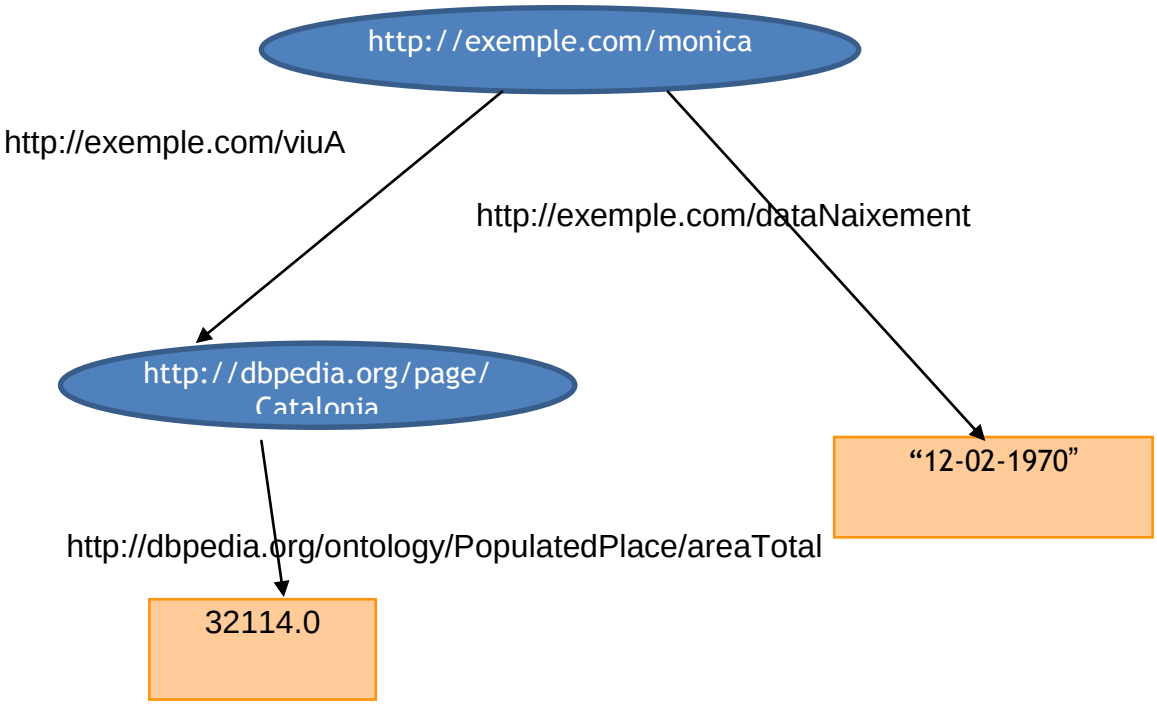
Les eines de procés no necessiten saber què significa el predicat *és_a*. Les eines RDF no coneixen el significat d'aquests noms, però tot i així són capaces de processar útilment la informació.

RDF està destinat a ser publicat a Internet, per tant els noms que utilitzem han de referenciar inequívocament una instància del món real. Els noms han de ser globals, és a dir, no s'hauria d'escollir un nom que algú més hagi pogut concebre per a referir-se a alguna cosa diferent. Formalment, els noms per a subjectes, predicats i objectes han de ser Identificadors Uniformes de Recursos (URIs).

Els recursos URI poden tenir la mateixa sintaxi que una direcció web, tot i que pot ser que hi hagi o no un lloc web en aquella direcció, i això no té importància.

La definició donada per un URI és única ja que està formada per un prefix que consta del URL on es troba la definició. Per convenció, si hi ha un propietari clar d'una URI, és aquest qui té el control d'aquest i ningú més que ell podrà inventar un nou recurs usant aquesta direcció.

Les declaracions es poden representar com un graf dirigit:



I en RDF (N3):

<http://exemple.com/monica> <http://exemple.com/dataNaixement> "12-01-1970" .
<http://exemple.com/monica><http://exemple.com/viuA>
<http://dbpedia.org/page/Catalonia> .
<http://dbpedia.org/page/Catalonia><http://dbpedia.org/ontology/PopulatedPlace/areaTotal> "32114.0" .

Donat que les URIs poden ser bastant extenses i repetitives, en les notacions RDF es solen abreviar utilitzant "espais de nom" de XML. Igual que en XML, l'espai de nom es declara a l 'inici del document RDF i a partir d'aquell moment es fa servir en la seva forma abreviada.

Els espais de noms no tenen cap significat en RDF, només és una eina per a abreviar URIs extenses. Aquests espais de nom o *namespaces* permeten definir elements amb el mateix nom dins el mateix context sempre i quant l'element porti un prefix davant. Aquests espais proporcionen un mètode simple

per a qualificar noms d'elements i atributs usats en el XML associant-los a espais de noms identificats per referències URI. L'ús de *namespaces* també evita confusions en la reutilització del codi.

En el nostre exemple podríem declarar els prefixes **ex** i **dbp** i fer-ne ús de la següent manera:

@prefix ex:, namespace URI: <http://www.exemple.com/> .
@prefix dbp:, namespace URI:http://dbpedia.org/page/

ex.monica.dataNaixement “12-02-1970”;
ex:viuA: dbp.Catalonia .
dbp.Catalonia http://dbpedia.org/ontology/PopulatedPlace/areaTotal “32114.0” .

Aquests conceptes formen la majoria del model abstracte de RDF per a codificar coneixement i és anàleg a la API comú que la majoria de llibreries XML disposen. Necessitem però un format d’arxiu per a intercanviar dades i de fet n’hi ha dos per a RDF.

Es podria pensar en RDF com a un model relacional, però en cap cas seria el mateix que el model de dades relacional basat en bases de dades SQL. Donat que les declaracions individuals de RDF s’expressen en triplets subjecte, predicat, objecte, els conjunts d’aquests amb un predicat comú es podrien assignar a les relacions binàries en el model relacional (en dues columnes de la taula)

foaf: Name	
subjecte	**objecte**
_: Persona	"John"
_: PersonB	"Jane"
_: PersonC	"Fred"

O també

subjecte	**foaf: Name**
_: Persona	"John"
_: PersonB	"Jane"
_: PersonC	"Fred"

On foaf:Name és una abreviatura del nom globalment únic http://xmlns.com/foaf/0.1/name

En altres paraules, a través de vistes relacionals la web (semàntica) com un tot, pot ser considerada una sola base de dades. En aquesta visió un fitxer RDF individual és només una memòria cau d'una part de les dades en la Web Semàntica. La visió en graf de RDF és més que una analogia amb l'estructura interrelacionada del web, és una extensió de la mateixa.

En el model relacional, una fila d'una taula és en realitat una afirmació que la relació s'aplica als valors de la fila. Una consulta SELECT és un filtre en les afirmacions que són certes per les condicions donades. Un RDBMS mantindrà la coherència lògica en totes les dades que conté. En aquest sentit, una base de dades relacional és un motor de raonament.

Però una altra diferència significativa entre la base de dades relacional i RDF és que en la primera, per a un determinat conjunt de valors d'una relació o bé es considera com CERT(hi ha una fila corresponent a la taula) o FALS (no n'hi ha). En el model RDF en el cas general, si un conjunt de valors no està en la "fila" (és a dir, no té una declaració particular), llavors no és fals, simplement es considera desconegut. A la pràctica, quan es consulta mitjançant programació o amb SPARQL, només es busca en un determinat conjunt de dades, de manera que això es tracta com l'univers (tot el graf) i per tant tancat.

Quan els predicats apareixen en taules com:

rdf: type	
subjecte	**objecte**
foaf: Nom	rdf: Property

El més senzill és deixar de pensar en un model relacional i pensar en el model orientat a objectes a objectes – en termes d'herència com a mínim - , que tot i seguir sent molt diferent és probablement més proper conceptualment.

La notació més difosa per a RDF és la serialització RDF/XML que és a més la sintaxi normativa (W3C) per a expressar RDF.

Per tal d'indicar quin tipus de document conté el XML, s'afegeix el node arrel:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

    < ... >

</rdf:RDF>
```

Aquest espai de noms (rdf) i el node RDF formen l'arrel per a tots els documents RDF. En el nostre exemple:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmnls: ex="http://www.exemple.com#">
```

```
    <rdf:Description rdf:about="http://www.exemple.com/monica">
        <ex:dataNaixement>"12-02-1970"</ex:dataNaixement>
        <ex:viuA rdf:resource="http://www.dbpedia.org/page/Catalonia" />
    </rdf:Description>

</rdf:RDF>
```

2.1.4 OWL

El llenguatge OWL va ser dissenyat per a ser usat en aplicacions que necessitessin processar el contingut de la informació enlloc de presentar-la simplement als humans.

Aquest llenguatge facilita una major interpretació dels continguts Web que XML, RDF i RDF-Schema, ja que permet representar explícitament el significat de termes en vocabularis i les relacions entre aquests termes.

Els avantatges de OWL vers RDF Schema són que el primer comprova la validesa de les dades i que permet la comprensió i integració de les informacions procedents de diferents fonts en dotar les dades de semàntica comprensible per les màquines. OWL incorpora més llenguatge per a descriure propietats o classes, per exemple: relacions entre classes, cardinalitat, igualtat, més tipus de propietats, característiques de propietats i classes numerades.

OWL proporciona tres subllenguatges:

- OWL Lite: proporciona classificació jeràrquica i restriccions simples.
- OWL DL: proporciona màxima expressivitat conservant una completesa computacional (garanteix que totes les conclusions son computables) i resolutivitat (tots els càlculs es completen en temps finit)
- OWL Full: proporciona màxima expressivitat i llibertat sintàctica de RDF sense garanties computacionals.

3. Tractament XML en els SGBD

Un sistema gestor de base de dades (SGBD) és una col·lecció d'aplicacions informàtiques que fan d'interfície entre la base de dades, l'usuari i d'altres aplicacions.

Aquest sistema es composa d'un llenguatge de definició de dades, un de manipulació de dades i un llenguatge de consultes.

Un SGBD permet definir les dades a diferents nivells d'abstracció i manipular-les garantint-ne la seguretat i la integritat.

Un SGDB ha de permetre:

- Definir una base de dades especificant tipus, estructures i restriccions de dades
- Construir la base de dades: emmagatzemar les dades en algun mitjà controlat pel mateix SGBD
- Manipular la base de dades: realitzar consultes, actualitzar-la, generar informes...

La majoria dels SGBD representen la informació en simples tuples en format registre.

Els SGBD tal i com els coneixem, han estat funcionant durant dues dècades, originalment en models jeràrquics o en xarxa. A partir de 1970, van aparèixer dues línies de recerca, per una banda la introducció del model relacional o per altra banda el desenvolupament de bases de dades semàntiques.

El model relacional va revolucionar el camp separant les dades lògiques de l' implementació física. La simplicitat del model va permetre el desenvolupament de potents llenguatges de consulta no procedimentals.

El model semàntic es va introduir inicialment com una eina de disseny d'esquemes: l'esquema es dissenyava en un model semàntic d'alt nivell i després es traduïa a un dels models tradicionals per a fer-ne l' implementació. L'objectiu d'aquests models semàntics inicials era modelar les relacions entre dades que es donaven freqüentment en les aplicacions de bases de dades típiques.

Els SGBD orientats a objectes, XML, etc.. neixen a la dècada dels 90 proporcionant capacitats de gestió de dades, gestió d'objectes (permetent la definició de tipus de dades més complexes i encapsulament de la semàntica de les dades) i gestió del coneixement (ORACLE, DB2, SYBASE,etc).

Les bases de dades XML neixen la de necessitat d'emmagatzemar i recuperar

dades poc estructurades. XML. N'hi ha de dos tipus:

- XML-Enabled: són bases de dades relacionals que suporten XML, però segueixen emmagatzemant la informació de manera relacional, és a dir, de forma tabular (taules, registres, camps) o bé emmagatzemant tot el document en format BLOB. Permeten obtenir els resultats en format XML. Aquestes bases de dades descomponen un document XML en el seu corresponent model relacional o d'objectes

- XML Nadius: aquestes bases de dades defineixen un model lògic pel document XML, respecten l'estructura del document, permeten fer consultes sobre aquesta estructura i recuperen el document tal i com va ser inserit a la base de dades originalment. La unitat d'emmagatzemament lògica d'aquestes bases de dades és el document.

Els avantatges dels SGBD nadius XML són:

- Permeten un procés més ràpid de les dades

- Emmagatzemen la informació en format natiu, sense que existeixi cap traducció del XML a estructures relacionals o objectes,

- Els seus esquemes permeten regles d'emmagatzemament i indexació,

- Mantenen intacte el model XML, conservant-ne la integritat

- Suporten llenguatges de consulta XML i les operacions es realitzen en XML

- No necessiten mapatges addicionals

- Permeten emmagatzemar documents heterogenis en la mateixa col·lecció (les col·leccions a les bases de dades XML natives són equiparables a les taules a les bases de dades relacionals)

Alguns sistemes XML nadius són:

- dbXML (comercial, emmagatzematge propietari)

- Virtuoso (comercial , emmagatzematge propietari)

- eXist (opensource, emmagatzematge relacional)

- BirdStep RDM XML (comercial, emmagatzematge OO)

- Tamino (comercial, emmagatzematge propietari)

A banda dels sistemes nadius XML, s'està treballant en la integració del XML en altres SGBD, mitjançant SQL/XML (ISO/IEC 9075-14: XML-Related Specifications).

SQL/XML és un estàndard ANSI, ISO que incorpora XML dins el llenguatge SQL de bases de dades objecte - relacionals. Permet emmagatzemar documents XML a aquestes bases de dades per a consultar-les mitjançant “XPath”, “XQuery” i per a publicar les seves dades SQL en un format de documents XML (mapatge de taules - XML i viceversa)..

Actualment suporten i implementen SQL/XML: IBM, Oracle, Microsoft (similars característiques però amb sintaxi propietària). Tots suporten XQuery dins el SQL tot i existir diferències en la implementació física d'emmagatzematge: Oracle 10g és basat en CLOB o taules OR, Microsoft 2005 i 2008 emmagatzemat com BLOB en format intern propietari i DB2 V9 basat en CLOB.

3.1 Magatzems de dades o Data Warehouses

Un magatzem de dades és una col·lecció de dades que forma el conjunt d'informació de que disposa una organització. El volum d'aquesta informació sol ser elevat, i mitjançant l'anàlisi i l'ús de les eines adequades per a explotar-ne les dades, es pot aprofitar en la presa de decisions empresarials.

Les eines que es poden utilitzar per a l'anàlisi d'aquestes dades són:

- Aplicacions analítiques (OLAP)
- Informes i consultes (*reporting and querying)*
- Mineria de dades (*data mining*)
- Tècniques de visualització de dades

Aquestes dades es poden trobar un o varis dels sistemes d'informació de l'organització donant suport als processos transaccionals d'aquesta (vendes, recursos humans, CRM...) i podrien estar fins i tot en formats diferents, però per a la seva explotació és necessari que estiguin integrats.

Aquests sistemes estan orientats a processos de negoci, a registrar transaccions, no a la presa de decisions. Per tal d'obtenir els indicadors o mesures de rendiment per a la presa de decisions, cal respondre a consultes complexes, càlcul d'agregats, etc als que l'estructura de les dades no està preparada, ja que el seu disseny probablement es va fer pensant en l'optimització d'aquests processos transaccionals.

Per tot això, es requereix un sistema d'informació específic que s'alimenti de les fonts de dades operacionals de la organització i que no en penalitzi el rendiment, per la qual cosa s'evita utilitzar les dades d'us actual.

Aquest sistema d'informació, conté la informació estratègica per a la presa de decisions, així doncs s'utilitzarà per a analitzar dades, detectar tendències i dissenyar estratègies

3.2 SGBD Semàntic

Partint d'aquest escenari, amb els requeriments que presenta la web semàntica i els components que la formen, es planteja la qüestió de l'emmagatzematge, manipulació i obtenció de resultats a partit d'aquestes dades aprofitant tota la potència que els atorga el model.

Hi ha diferents vessants quan a la implementació dels SGBD semàntics. Per una banda, hi ha els sistemes nadius i per altra banda els sistemes que adapten SGBD per a suportar el model (no nadius).

Dins els sistemes no nadius, s'han creat complexes teories i patrons per a encaixar objectes o estructures jerarquitzades com XML dins les bases de dades relacionals. Existeix un gran nombre de programari intermedi o *middleware* encarregat de la transferència entre estructures XML i bases de dades relacionals que permet,

En SGBD relacionals:

- Que els SGBD relacionals tinguin una estructura regular davant el caràcter heterogeni dels documents XML

- Els documents XML poden contenir molts nivells d'aniuament mentre que les dades relacionals són planes

- Els documents XML tenen un ordre intrínsec mentre les dades relacionals són no ordenades

- Les dades relacionals són normalment densos (cada columna té un valor) mentre que les dades XML són disperses, poden representar la carència d'informació mitjançant l'absència de l'element

En SGBD Orientats a Objectes:

- Utilitzen una nova classe per a cada tipus d'informació. Es modelen les dades en el document XML com un arbre d'objectes però que són específics per a les dades en el document

- Aquesta opció és sobretot apropiada per a documents XML validats i per a aplicacions centrades en les dades.

- Hi ha un gran nombre de productes que implementen aquesta associació de documents XML a objectes (XML Data Binding).

- Un cop transformat el document XML en objectes, de forma específica o genèrica, els objectes ja són gestionats directament pel SGBDOO.

- La informació es consulta mitjançant el llenguatge de consulta OQL (Object Query Language)

- Els mecanismes d'indexació, optimització, procés de consultes, etc són les del propi SGBDOO, i pel general no són específics per al model XML

SGBD Nadius:

Són sistemes dissenyats específicament per a emmagatzemar documents XML
Les seves característiques son:

- La unitat lògica d'emmagatzematge és el document XML

- Preserven el document

- Suporten llenguatges de consulta XML

- No tenen cap model d'emmagatzematge concret, ja que poden ser construïdes sobre bases de dades relacionals, jeràrquiques, orientades a objectes o be mitjançant formats d'emmagatzematge propietaris

Hi ha diferents tipus depenent de l'emmagatzematge dels documents:

- Basat en text: emmagatzema el document XML sencer en forma de text i proporciona funcionalitat a la base de dades per tal d'accedir-hi. Això es pot fer de diverses maneres: mitjançant l'emmagatzematge del document com un BLOB en una base de dades relacional, mitjançant un fitxer, i proporcionar índexs sobre el document per tal d'accelerar l'accés a la informació, o mitjançant l'emmagatzematge del document en un magatzem de dades amb índex, suport per a transaccions, etc.

- Basat en el model: emmagatzema un model binari del document (per exemple DOM, el model d'objectes del document, una API que proporciona un conjunt estàndard d'objectes per a representar documents HTML i XML) en un magatzem existent o bé un d'específic. Hi ha vàries possibilitats: traduir el DOM a taules relacionals, traduir el DOM a objectes en una base de dades orientada a objectes o utilitzar un magatzem creat per a aquesta finalitat.

Les característiques d'aquestes bases de dades són:

- Emmagatzematge de documents en col·leccions
    - Les col·leccions serien en les bases de dades natives, les taules en les relacionals.
    - Els documents es solen agrupar ,en funció de la informació que contenen, en col·leccions que a la vegada poden contenir altres col·leccions.

- Validació de documents.

- Consultes: la majoria de SGBD XML suporten un o més llenguatges de consulta.

- Indexació XML: permeten la creació d'índexs que acceleren les consultes que es realitzen més freqüentment.

- Creació d'identificadors únics: A cada document XML se li associa un identificador únic pel que serà reconegut dins el repositori.

- Actualitzacions i esborrats: els SGBD nadius tenen gran varietat d'estratègies per a actualitzar i esborrar documents. Molts sistemes suporten XUpdate per a aquesta funcionalitat.

3.3 Emmagatzematge, manipulació i obtenció de dades

Sobre la manipulació de la informació, cal ser conscients que aquestes dades es modificaran amb freqüència per par de les fonts: agregació de contingut equivalent o traduccions, inclusió de recursos relacionats o inserció o modificació d'esquemes de dades...

En al cas de l'emmagatzematge, existeixen algorismes per a transformar el contingut web en informació vàlida per a la web semàntica i ja sigui mitjançant aquest procés o a través de la inserció directa de sentències RDF, la gran quantitat de dades existent exigeix un sistema d'emmagatzematge persistent molt eficient i fiable en quant a la integritat de dades i relacions.

En principi, pel que respecta als camps literals, la seva modificació no ha de suposar un problema ja que aparentment no hauria de tenir repercussió en dades relacionades, però si la dada a modificar influeix sobre altra informació, el problema es complica en funció de la filosofia d'emmagatzematge escollit:

- si les dades s'infereixen en temps de cerca (*backward-chainging*) la modificació podria se immediata. En aquest cas s'emmagatzema un índex mínim amb el qual les respostes poden ser calculades sobre demanda. Aquest tipus de raonament és adequat quan:

  - hi ha poc marge per a la reutilització de respostes calculades
  - les respostes són dinàmiques
  - les respostes poden ser calculades en temps real d'una manera eficient
  - l'espai que ocupa la resposta és massa gran per a emmagatzemar-la

- si el raonament es fa abans d'incloure la informació a la base de dades (*forward-chainging*) la modificació d'una relació requerirà una nova inferència sobre la informació original i successivament, tots els elements del càlcul abans i després del canvi. En aquest cas s'inicia amb les dades disponibles i s'utilitzen regles d'inferència per a extreure'n més fins a assolir l'objectiu. Aquest tipus de raonament és apropiat per a computació i emmagatzematge de:

  - dades d'accés freqüent
  - dades relativament estàtiques
  - dades costoses de calcular
  - prou petites per a emmagatzemar eficientment

Si el criteri d'indexació inclou els camps literals, caldrà recalcular els índexs cosa que en estructures amb milions de registres pot ser molt costós. El estàndard d'obtenció de dades, si suposem que l'estàndard actual és RDF/RDF Schema /OWL, seria SPARQL.

SQARQL es pot mapejar eficientment a consultes SQL, però cal tenir en compte que les bases de dades relacionals treballen en universos tancats, mentre que en el context del model RDF /RDF Schema /OWL es fa en universos oberts. Això implica:

- En un univers obert, diferents persones poden utilitzar termes diferents per a referir-se a la mateixa cosa, o un terme es pot utilitzar per a referir-se a coses diferents.
- En un univers tancat, si la condició de cerca és falsa vol dir que l'element no existeix, mentre que en un univers obert, si no es troba l'element vol dir que és desconegut.
- Dins l'univers tancat totes les proposicions són certes o falses, no és així en l'obert.
- Un univers tancat té un sol model o implementació consistent, mentre a l'obert hi ha varis models.

Aquests aspectes són preocupants en el sentit que la cerca i el raonament sovint es realitzen en magatzems persistents, i així dispara els costos temporals.
També cal tenir el compte que les consultes es poden fer sobre varis models que poden correspondre o no amb magatzems físics diferents, precisant la composició dels resultats finals obtinguts en cada cerca parcial.

Actualment podríem pensar que aquests criteris d'emmagatzematge, manipulació i cerca ja els resolen els motors de cerca actuals, com Google, però la diferència és que aquests motors de cerca treballen amb sintaxi i no amb semàntica. La diferència entre un i altre sistema és:

- El emmagatzematge semàntic treballa amb models als que s'han d'ajustar les dades terminològica i en forma d'assercions (com a

components d’afirmació, per exemple “A és una instància de B”, “Joan és una persona”).

- La quantitat de models pot ser múltiple

- Les relacions entre dades semàntiques és explícita (o pot ser calculada) mentre que és la similitud lèxic/semàntica la que determina la relació en els buscadors convencionals

- La web semàntica exigeix càlculs complexes per a obtenir els elements que s'ajusten a un criteri de cerca

- Als cercadors actuals hi ha informació obsoleta (caducada o que no existeix realment) mentre que a la web semàntica no ha d'existir

- El raonament, l'ajust als models i qualitat de resultats fa que la informació no sigui tan fàcil de manipular als magatzems semàntics

3.4 Diferències SGBD tradicionals i semàntics, limitacions i avantatges

Si considerem com a models tradicionals el model relacional i el orientat a objectes, les diferències dels models són les que marquen a grans trets les diferències entre els diferents sistemes.[3]

En els models semàntics la informació és modelada en termes d'unitats atòmiques anomenades entitats o objectes. Casa entitat ha de tenir una identificació que la distingeixi de qualsevol altre sense ambigüitats. Diferents entitats del món real comparteixen unes propietats comuns, i un tipus d'entitat és una representació conceptual dels objectes que tenen propietats comuns. Mentre que els objectes o entitats representen instàncies de la base de dades, el tipus d'entitat correspon a la definició d'esquema. Entre entitats es poden establir jerarquies, i les propietats de les entitats són definides per atributs.

En canvi, en el model relacional, l'estructura la formen taules i tuples. El conjunt d'atributs són les propietats que caracteritzen una tupla. Cada tupla del model representa una entitat o objecte i l'esquema de la relació correspon a la definició del tipus d'objecte.

En el model d'objectes com a esquema de la base de dades, la estructura està formada per objectes que representen instàncies de la realitat, amb una estructura (atributs), un estat (valor dels atributs) i un comportament (mètodes). La base de dades orientada a objectes incorpora els conceptes importants del paradigma: encapsulació , herència i polimorfisme.

En el model semàntic, les regles es divideixen en regles de restricció i regles d'inferència. Algunes regles de restricció són inherents a la definició de l'estructura del model. Mentre altres han de ser declarades de forma explícita. Les restriccions estàtiques es refereixen a condicions que han de complir les dades en l'estat actual de la base de dades, i les dinàmiques descriuen

condicions que involucren més d'un estat de la base de dades. Les regles d'inferència són les que permeten obtenir dades derivades a partir de la informació de la base de dades. Aquestes regles estenen la semàntica de les dades capturada de la seva estructura. La informació que proporcionen aquestes regles rep el nom de coneixement abstracte, mentre que la informació capturada per l'estructura rep el nom de coneixement concret.

En el model relacional les regles poden ser d'integritat o de derivació. Les regles d'integritat definides sobre l'esquema permeten definir la clau primària, la integritat referencial mitjançant la declaració de les claus foranies, amb la corresponent funció de bijecció dels atributs que formen part del mateix domini en ambdues relacions, i la integritat de valor no nul. Les regles de derivació apareixen a través de les vistes. Una vista és una relació que es fa visible a l'usuari sense que aquesta relació formi part de l'esquema conceptual.

Al model d'objectes, com al relacional, hi ha regles d'integritat i de derivació. Les regles d'integritat són la identificació única de cada objecte, integritat referencial mitjançant la conformació de tipus, integritat de jerarquies amb herència.. d'altres regles d'integritat es declaren dins el mètode formant part de la implementació. Les regles derivació poden ser definides com mètodes del tipus d'objecte corresponent.

Les operacions de la estructura al model semàntic es descriuen mitjançant els llenguatges de manipulació de dades (DML). Aquests llenguatges poden ser orientats a consultes o permetre la transformació de les dades. En un principi, donat l'abast dels models semàntics de dades, només es limitaven a les operacions de consulta.

Les operacions al model relacional es realitzen a partir del llenguatge de consulta estàndard SQL orientat a la manipulació de tuples. Les dades resideixen a la base de dades i els processos es troben a les aplicacions desenvolupades mitjançant el llenguatge SQL immers en un llenguatge de programació. El desenvolupament es fa sota sistemes relacionals (model conceptual de dades, model lògic).

Les operacions al model d'objectes es descriuen mitjançant el llenguatge de consulta OQL que disposa d'una forma declarativa per a expressar l'accés als objectes i navegar entre els seus atributs complexes per a recuperar conjunts de valors, crear nous objectes o crear conjunts d'objectes. Els SGBDOO gestionen objectes en els quals estan encapsulades les dades i les operacions que actuen sobre aquestes.[5]

3.4.1 Limitacions del model semàntic

El propòsit de les relacions semàntiques que pretenem descriure en el model és específic: la construcció d'ontologies per a dominis ben definits i aplicacions particulars. El nostre objectiu no és el modelat de i representació de relacions "reals" sinó el modelat i representació de relacions per a la resolució de problemes. Això implica que el conjunt de relacions necessari per a una

aplicació dins un domini particular, ha de ser identificat prèviament. No obstant, la representació de relacions per a aquest propòsit presenta varis problemes.

Respecte a les ontologies, tot i que les relacions semàntiques mantenen junts els elements d'una ontologia i determinen la seva estructura, l'atenció s'ha centrat en els conceptes i les operacions que es poden fer amb ells. Això redueix les capacitats de modelat, i també les de reutilització i integració automàtica dels recursos lingüístics.

El modelat semàntic comporta major complexitat lògica i problemes d'interpretació de les consultes, degut al seu apropament al llenguatge natural, produeix ambigüitats.

La gran limitació del model és la impossibilitat d'abastar tot el coneixement

> "Insistir en que las ontologías permitirán razonar a los ordenadores y realizar inferencias sobre los contenidos, no de un dominio del conocimiento bien delimitado y para unas tareas específicas (como ya sucede), sino de la web en su conjunto y en todos los ámbitos del conocimiento y de la realidad, es, hoy por hoy, no querer ver precisamente la simple realidad. No existe ninguna evidencia empírica ni base conocida alguna para que esto pueda funcionar, ni a corto ni a medio plazo". (Lluís Codina, 2007)

Es considera que per a un ús adequat i funcional de les bases semàntiques s'hauria de restringir l'aplicació d'aquestes a un domini concret on els conceptes i relacions que entressin en joc fossin limitats i així, ser representables més fàcilment.

3.4.2 Avantatges del model semàntic

El modelat semàntic resumeix la complexitat lògica representativa de la base de dades, això permet més usabilitat per part de més usuaris, que no són conscients de l'estructura lògica d'una base de dades remota.

La base de dades semàntica permet l'adaptació de dades derivades, no existents a la base de dades però que per a l'usuari és obvia la seva existència.

La semàntica és una representació del món real, pel que la tendència és que es converteixi en un dels principals components a les aplicacions de base de dades en l'actualitat.

Les ontologies que permeten donar sentit semàntic relacionant varis conceptes, oferint un millor temps de resposta a la cerca sobre grans volums d'informació.

Permet la reusabilitat i interoperatibilitat entre aplicacions web i la millora del emmagatzematge i la recuperació de dades, la realització de certes tasques administratives i la comunicació entre usuaris en temps i espai.

Millora la localització d'informació al web d'integració de dades entre aplicacions i la col·laboració amb els altres a la xarxa.[2]

4. Llenguatges de consulta en magatzems de dades semàntiques

L'ús principal d'aquests SGBD és permetre la recuperació de documents o recursos, i això s'aconsegueix mitjançant sentències que són organitzades segons normes lògiques definides pel llenguatge. Existeixen diferents llenguatges de recuperació per a XML i per a RDF.

4.1 Llenguatges de consulta XML

A les bases de dades relacionals, es poden seleccionar i retornat parts de la base de dades utilitzant llenguatges de consulta tipus SQL.

El mateix succeeix per als documents XML, pels quals existeix un nombre de propostes de llenguatges de consulta, que permeten una sèrie d'operacions bàsiques com són:

- Filtrar, seleccionar valors des d'un document (navegació, selecció , extracció)
- Integrar valors des de múltiples fonts (agregació, composició...)
- Transformar valors d'un esquema a un altre (construcció de documents XML)

Alguns dels llenguatges de consulta per a XML són:

- XPath: és un dels llenguatges més populars per a navegar, seleccionar i extreure valors de documents XML. Per exemple, un camí (*path*) indicat amb una expressió com **/AAA/DDD/BBB** , selecciona tota els elements BBB que són fills de DDD i a l'hora fills de l'element arrel AAA:

```
<AAA>
   <BBB/>
   <CCC/>
   <BBB/>
   <BBB/>
   <DDD>
      <BBB/>
   </DDD>
   <CCC/>
</AAA>
```

- XQuery: Inclou *join* i agregació. Genera documents XML com a resposta:

```
For $x in doc("books.xml")/bookstore/book
where $x/price>30
order by $x/title
return $x/title
```

El concepte principal en els llenguatges de consulta XML és l'expressió “encaminadora” que especifica com un node o un conjunt de nodes, en la representació em arbre del document XML pot ser obtingut. S'introdueixen expressions del tipus XPath que es poden utilitzar per a altres objectius que les consultes, com per exemple per a transformar documents XML.

4.2 Llenguatges de consulta RDF

Per tal de poder accedir als continguts del web, és important l'ús dels llenguatges de recuperació en la recuperació i organització de la informació.

Tal i com s'ha descrit en els apartats anteriors, un dels components de la web semàntica és RDF. RDF és un format per a grafs dirigits i etiquetats per a representar la informació a la Web.

Els llenguatges de recuperació són llenguatges informàtics utilitzats per a recuperar informació dels magatzems de dades, en el nostre cas cal que permetin executar cerques complexes sobre grafs RDF amb una sintaxi senzilla.

Existeixen varis llenguatges de recuperació per a RDF com poden ser:

- SPAQRL
- SeRQL (Aidministrator)
- RDQL (Hewlett Packard)
- RQL (ICS-FORTH)...

En aquest treball, s'analitzarà el llenguatge de consulta SPARQL.

4.3 SPARQL

SPARQL és un llenguatge de recuperació basat en RDF. El seu nom és un acrònim recursiu de l'anglès *SPARQL Protocol and RDF Query Language*.

Es tracta d'una estandardització per a la consulta de grafs RDF normalitzat pel RDF Data Access Working Group (DAWG) del W3C.

SPARQL es pot utilitzar per a expressar consultes que permeten integrar diverses fonts de dades, si les dades s'emmagatzemen de forma nativa com RDF o són definits mitjançant vistes RDF a través d'algun sistema *middleware*.

SPARQL conté les capacitats per a la consulta dels patrons obligatoris i opcionals de graf, i de les seves conjuncions i disjuncions, i també suporta l'ampliació o restricció de l'àmbit de les consultes indicant els grafs sobre els que opera. Els resultats de les consultes SPARQL poden ser conjunts de resultats o grafs RDF.

Aquest llenguatge de consulta permet extreure informació en forma de URI, nodes blancs i literals. Permet també extreure subgrafs RDF i construir-ne de nous basats en els grafs inclosos a la consulta.

SPARQL consisteix en tres especificacions separades que contenen part de la seva funcionalitat. En total, el llenguatge consisteix en una *query,* un format per a les respostes i un mitjà per al transport de consultes i respostes:

- SPARQL Query Language: És el nucli de SPARQL, explica la sintaxi per a la composició de sentències i la seva concordança
- SPARQL Protocol: Format utilitzat per al retorn dels resultats de les cerques (queries SELECT o ASK) a partir d'un esquema XML.
- SPARQL Query XML Results Format: Descriu l'accés remot de dades i la transmissió de consultes dels clients als processadors. Utilitza WSDL per a definir protocols remots per a la consulta de base de dades basades en RDF.

Triplets en SPARQL

Igual que RDF, SPARQL també es composa de triplets consistents en subjecte – predicat - objecte finalitzades en un punt.

Les triplets en SPARQL representen patrons mitjançant els quals es cercaran triplets coincidents al magatzem de dades.

El llenguatge SPARQL té quatre tipus de consultes: SELECT, ASK , DESCRIBE i CONSULT.

La consulta SELECT és molt semblant a les SELECT de SQL, entenent que es farà una consulta amb una sèrie de filtres i condicions i es rebrà una resposta tabular. La diferència és que en SPARQL el que estem cercant són recursos, cada fila, és un recurs.

SPARQL també té la clàusula WHERE, però a diferència que en SQL, aquí aquesta clàusula és un patró de graf.[4][6][31][32]

Per a veure amb un exemple l'ús de SPARQL per a les cerques sobre RDF, utilitzarem un conjunt de dades inicial. Les dades escollides són del vocabulari FOAF que permet descriure persones i les relacions que s'estableixen entre elles.

```
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:foaf="http://xmlns.com/foaf/0.1/"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<foaf:Person>
```

```
<foaf:name>Monica Mañá</foaf:name>
<foaf:mbox rdf:resource="mailto:monicamana@gmail.com" />
<foaf:homepage rdf:resource="http://www.monicamana.com/" />
<foaf:nick>Jimbo</foaf:nick>
<foaf:gender>female </foaf:gender>
<foaf:knows>
<foaf:Person>
<foaf:name>Jose Manuel Lorca</foaf:name>
</foaf:Person>
</foaf:knows>
</foaf:Person>
/rdf:RDF>
```

Les URIs s'escriuen dins dels símbols < i >, i les cadenes entre cometes simples o dobles. Cal especificar sempre els *namespaces*, mitjançant URIs o amb la notació namespace:element.

Tant el subjecte, predicat i objecte d'un triplet poden ser substituïts per variables.

Una variable es pot associar a qualsevol valor concret. Amb SPARQL totes les associacions possibles es consideren, així doncs si un recurs té vàries instàncies d'una propietat, es retornaran varis resultats per a aquell recurs.

Un exemple de *query* en SPARQL seria:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?email
WHERE {
        ?person a foaf:Person.
        ?person foaf:name ?name.
        ?person foaf:mbox ?email.
}
```

La paraula clau PREFIX equival a la declaració de *namespaces* en XML, s'associa una URI a una etiqueta que en endavant s'utilitzarà per a descriure el *namespace.* Es poden declarar diversos prefixes en una mateixa *query.*

La paraula clau FROM identifica les dades sobre les que s'executa la consulta. En aquest exemple, es limitarà al RDF foaf. Una consulta podria incloure vàries clàusules FROM.

La paraula clau WHERE indica el patró sobre el que es filtrarà els triplets del RDF. Dins el WHERE s'especifica una llista de patrons de grafs (una llista de subjecte, predicat i objecte que finalitza amb un punt) que els possibles resultats han de complir. Així doncs en la consulta anterior es cercaria qualsevol entitat que sigui del tipus Persona i que tingui un nom i una adreça de correu.[33]

El tipus de consulta ASK retorna un booleà indicant si un patró de consulta coincideix o no.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

ASK WHERE {
```

```
        ?person a foaf:Person
          ; foaf:mbox <mailto:monicamana@gmail.com>
}
```

El tipus de consulta CONSTRUCT retorna un graf RDF construït substituint variables en un conjunt de triplets.

El tipus de consulta DESCRIBE retorna un graf RDF que descriu els recursos trobats.[34]

5. Anàlisi d’un SGBD semàntic concret: Virtuoso

Virtuoso és una plataforma creada per Openlink Software, que va néixer el 1998 i és considerat un motor de dades híbrid que combina les funcionalitats dels RDBMS9, ORDBMS10, bases de dades virtuals, RDF, XML i aplicacions web en un únic producte.

L'arquitectura híbrida del servidor permet comunicar diferents funcionalitats dins un únic producte a través de les següents característiques:

- S' integra de forma transparent en entorns de temps d'execució per a informàtica distribuïda, com Microsoft .NET y J2EE, facilitant d'aquesta forma la creació i l'allotjament de Web Services XML compatibles con WSDL, procediments emmagatzemats, funcions, desencadenants i tipus d'usuari definits escrits en Java o en qualsevol llenguatge compatible amb .NET.
- Gestió i integració de processos (BPEL)
- Treballa amb els ports HTTP y SOAP.
- Va ser desenvolupat en llenguatge de programació C.
- Es adequat per a sistemes operatius que permetin treballar en múltiples ordinadors, ja que consisteix en un sol procés amb varis fils que poden ser compartits pels clients
- Permet integritat de les entitats i integritat referencial
- Conté un diccionari de dades on s'emmagatzema tota la informació dels objectes dels usuaris
- El seu motor de base de dades proveeix connexions a fonts XML, ODBC11, JDBC12, ADO.NET13 i OLE DB14. A l'hora, la biblioteca Virtuoso PL proporciona funcions per a treballar amb diferents protocols d’Internet per a clients i servidors (POP3, FTP, NNTP, client LDAP o client SMTP)
- Amb el llenguatge ISQL (Interactive SQL) permet als administradors de la base de dades eliminar, actualitzar o consultar les dades per a les proves, a més permet realitzar anàlisi de problemes i manteniment de la base de dades
- Suporta SPARQL
- Les dades i les transaccions són assegurats mitjançant combinacions de la base de dades i del sistema operatiu quant a privilegis, rols i jerarquies dels mateixos. També s'inclou dins el motor de la base de dades el xifrat de dades per a protegir les dades transmeses.

5.1 Arquitectura de Virtuoso

L'arquitectura de Virtuoso es pot observar a la figura següent:

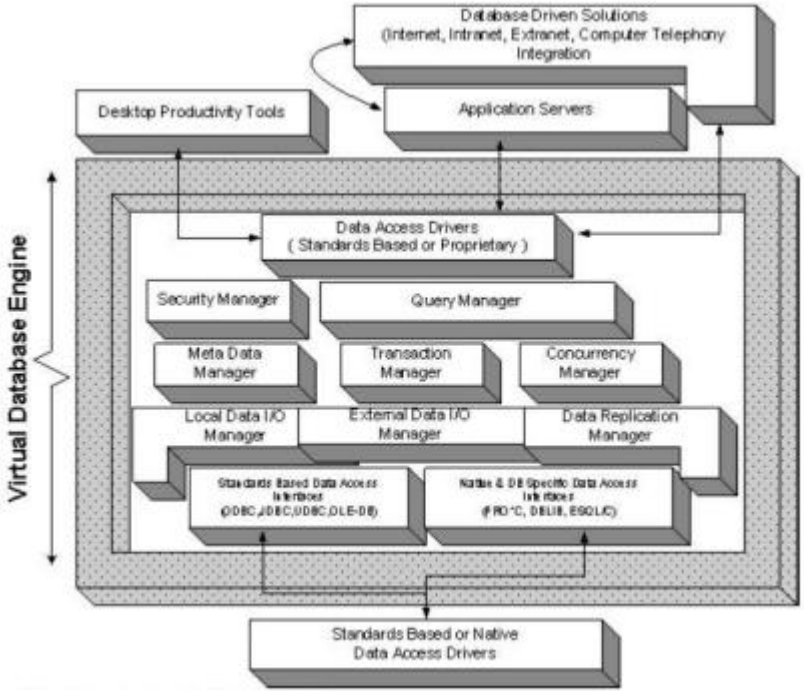


Figura 2: Arquitectura Virtuoso DB

En aquesta arquitectura es pot observar que el nucli del servidor és el motor de base de dades virtuals. A continuació se'n descriu l'arquitectura i els seus components:

Data Access Drivers: punt d'entrada als serveis del motor. Els serveis i aplicacions han de tenir les seves capes d'accés a dades descrites per a les mateixes API implementades per aquest component, i que ja cobreix els estàndards del mercat.

Security Manager: component responsable de protegir les dades i la seva transmissió (usant codificació) dins del domini del motor. També és responsable de mantenir els privilegis d'accés a nivell d'aplicació, usuari, rol i domini, en tot allò que fa referència a la creació, manipulació i destrucció de les dades del motor.

Query Manager: aporta anàlisi lèxic – sintàctic de les consultes, pla d'execució i compilació i servei de recerca.

Meta Data Manager: proporciona a Query Manager informació respecte les entitats sobre les que treballarà el pla d'execució. També es responsable d'enllaçar fonts de dades externes i dirigir al processador de consultes als gestors de dades d'entrada / sortida apropiats.

Transaction Manager: garanteix atomicitat, consistència, aïllament i durabilitat en les transaccions. Assegura que el motor és capaç de suportar Processament de Transaccions en Línia (OLTP) i distribuïdes (implementades seguint un protocol de confirmació en dos fases).

Concurrency Manager: permet que els clients i els serveis puguin obrir múltiples sessions concurrentment que executin insercions, modificacions i eliminacions sense que es comprometi el temps de resposta ni la integritat de les dades. El control de concurrència s'executa de manera optimista o pessimista en funció del que indiqui el client.

Local Data I/O Manager: proporciona un emmagatzematge local per a llegir i escriure a disc.

External Data I/O Manager: manega la lectura i escriptura des de fonts de dades externes. Pot ser implementat usant interfícies d'accés a dades com ODBC, JDBC, UDBC, OLEDB on natives.

Replication Manager: controla la migració de dades, la seva transformació i la sincronització a través d'un o més servidors Virtuoso dins un entorn d'execució distribuït. Coordina les activitats entre els controladors d'entrada i sortida.[9]

5.2 Modes d'emmagatzematge

Virtuoso permet emmagatzematge propietari al sistema d'arxius, així com en gestors de bases de dades, donades les seves característiques com a servidor multi protocol que proveu diferents API per a accedir a una base de dades relacional.

La solució que proposa Virtuoso per a organitzar la informació dins el magatzem de dades és la convencional del emmagatzematge genèric desnormalitzat.

Es basa en una taula amb quatre columnes: subjecte, predicat, objecte i graf (context o model) i amb índexs per defecte gspo i opgs.

El codi de creació de la taula principal és:

```
create table DB.DBA.RDF_QUAD (
        G IRI_ID,
        S IRI_ID,
        P IRI_ID,
        O any,
        primary key (G,S,P,O) );
create bitmap index RDF_QUAD_OGPS on DB.DBA.RDF_QUAD (O, G, P, S)
;
```

Les columnes G, S, P són de tipus enter, que internament s'anomena IRI (equivalent a URI) i que apunta una altra taula.

Quant a l'objecte, de tipus any, si la seva mida és petita (20 o menys caràcters), s'emmagatzema directament a la taula genèrica, en cas contrari apuntarà la

taula RDF_OBJ:

```
create table DB.DBA.RDF_OBJ (
        RO_ID integer primary key,
        RO_VAL varchar,
        RO_LONG long varchar,
        _DIGEST any
)
ceate index RO_VAL on DB.DBA.RDF_OBJ (RO_VAL)
ceate index RO_DIGEST on DB.DBA.RDF_OBJ (RO_DIGEST)
;
```

També s'emmagatzema en aquesta taula si l'objecte hagués de ser indexat per a cerques de text lliure.

En quant al tipus de dada XML Schema d'un objecte, es representa com 2 bytes a la taula RDF_DATATYPE, de la mateixa forma que succeeix amb l' idioma a la taula RDF_LANGUAGE:

```
create table DB.DBA.RDF_DATATYPE (
        RT_IID IRI_ID not null primary key,
        RT_TWOBYTE integer not null unique,
        RDT_QNAME varchar )
;
create table DB.DBA.RDF_LANGUAGE (
        RL_ID varchar not null primary key,
        RL_TWOBYTE integer not null unique )
;
```

També s'implementa la compressió de dades de dues maneres: s'eliminen els prefixos i s'inclouen (igualment comprimits) en una altra taula, o bé es comprimeix directament la pàgina de la base de dades.

Virtuoso també proporciona RDF Views per a mostrar dades relacionals com si fossin sentències RDF.[9]

5.3 Tractament de la informació

Es permet la càrrega massiva de dades a través del procediment emmagatzemat RDF_LOAD_RDFXML així com la càrrega de sentències individuals a través de RDF_QUAD_URI o d'altres.

Per a sentències individuals s'utilitza una extensió de SPARQL pròpia de Virtuoso (SPARUL) que permet inserció, actualització i eliminació.

Totes aquestes sentències es poden executar dins de transaccions i amb suport per a concurrència (gràcies als components Transaction Manager i Concurrency Manager del motor de base de dades).

Es podrien utilitzar també esdeveniments de modificació, donat que tot es realitza a través de SQL, seria possible programar *triggers* si s'accedís directament al esquema de la base de dades i aquesta els permetés[9].

Virtuoso infereix sentències que no han estat emmagatzemades físicament, és a dir, utilitza el mecanisme *backward-chaining*.

A banda de SPARQL i l'extensió SPARUL, Virtuoso suporta altres extensions que li permeten suportar *count, max, min, avg, sum, distinct i group by*, tot integrat dins de consultes SQL tradicionals.

De la mateixa manera, proporciona la capacitat d'indexació per a texte i XML, permetent crear índexs de text lliure en qualsevol columna de caràcters a través de la instrucció create_free_index.[9]

5.4 Instal·lació de Virtuoso Opensource

La instal·lació de Virtuoso Opensource en un sistema operatiu Windows i utilitzant l'emmagatzematge propietari en el sistema d'arxius, s'ha realitzat seguint els següents passos[37]:

1. Descàrrega de Virtuoso Opensource 6.1.6, del lloc web http://sourceforge.net/projects/virtuoso/files/virtuoso/6.1.6/virtuoso-opensource-win32-20120802.zip/download

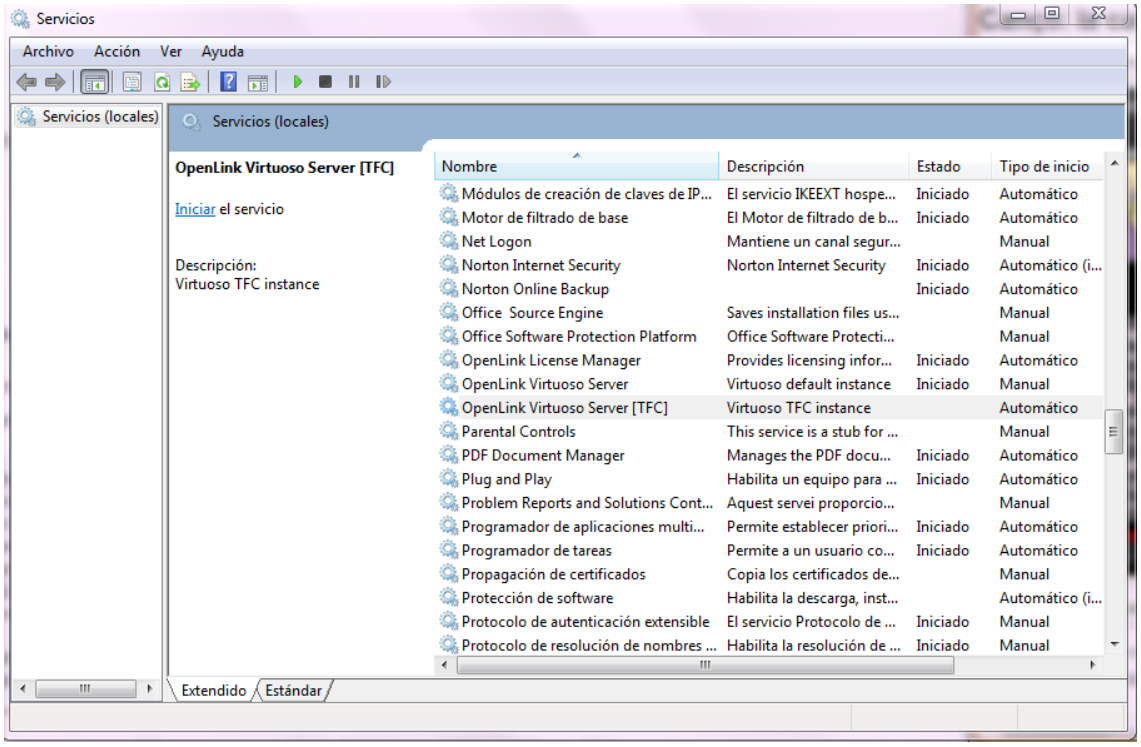
2. Un cop descarregat el fitxer, i sempre amb permisos d'Administrador, cal descomprimir els fitxers i establir les variables d'entorn:

```
SETPATH=%PATH%;%VIRTUOSO_HOME%\bin;%VIRTUOSO_HOME%\lib
```

3. Creem una nova instància sobreescrivint el servei si ja existís:

```
virtuoso-t +service screate +instance "Instance Name" +configfile virtuoso.ini
```

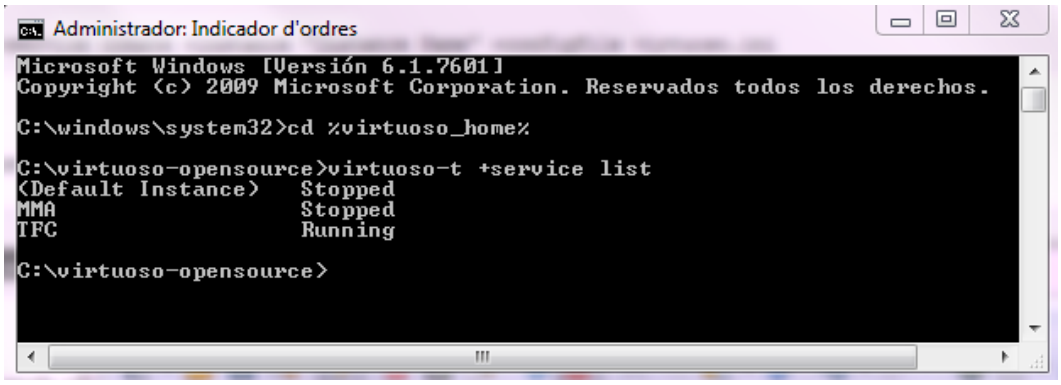
En el nostre exemple la instància rep el nom TFC. Un cop creat el servei el veurem al Windows Service Manager com Openlink Virtuoso Server [Instance Name] :



També podem arrencar i aturar el servei des de l'indicador d'ordres de Windows així com consultar quins serveis hi ha creats i registrats:

Veure els serveis que hi ha instal·lats a la màquina i el seu estat:

virtuoso-t +service list



Arrencar un servei:

virtuoso-t -I "Instance Name" +service start

Aturar un servei:

virtuoso-t -I "Instance Name" +service stop

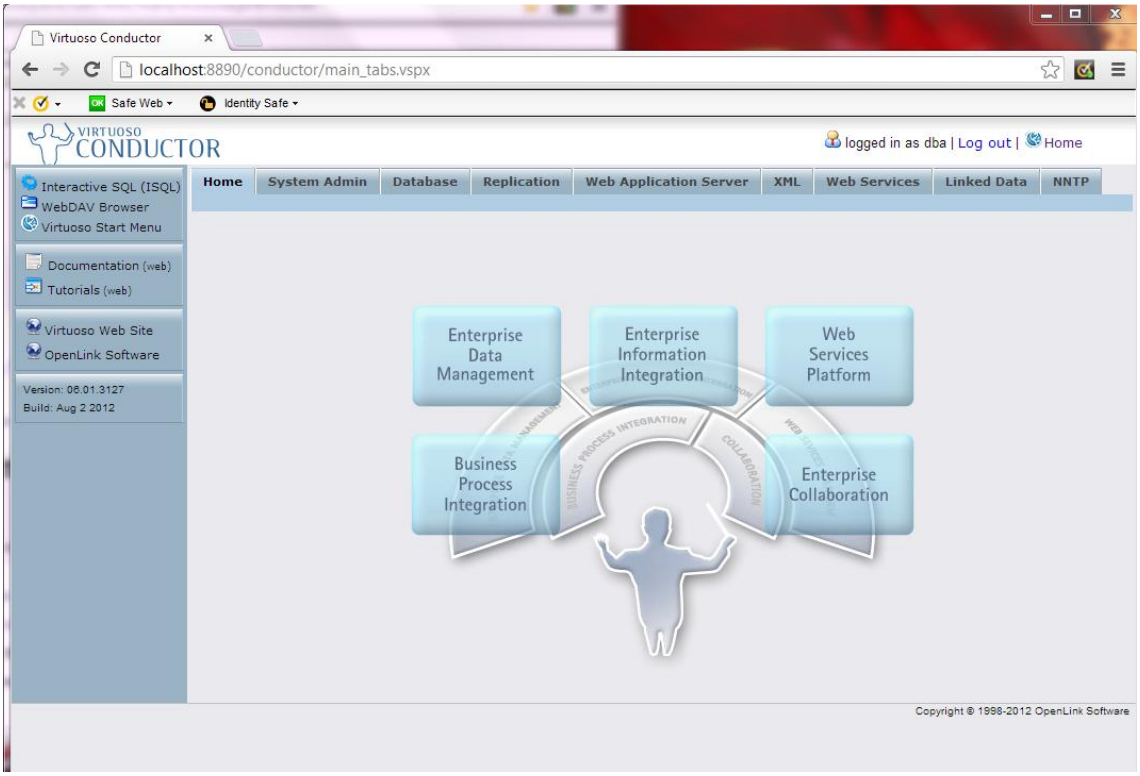
Un cop arrencat el servei podem accedir a les Web Admin Tools de Virtuoso (Virtuoso Conductor) mitjançant la URL:

http://localhost:8890/conductor

o bé

http://<hostname>:8890/conductor

Per defecte, la instal·lació proporciona l'usuari dba amb *password* dba. Des del Virtuoso Conductor es permet l'administració de comptes d'usuari i les configuracions dels perfils de seguretat:

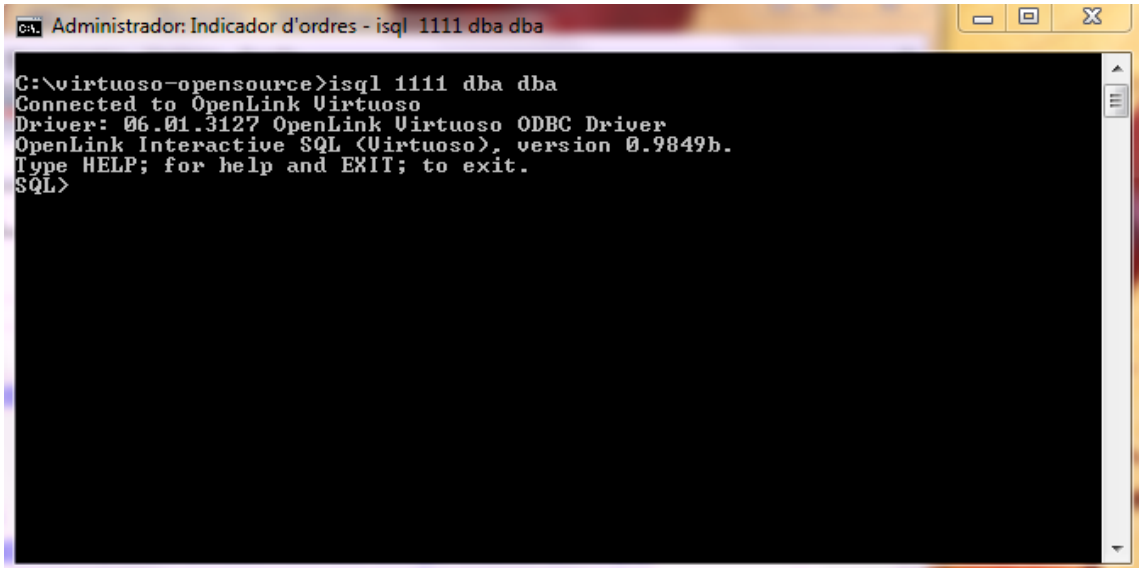


5.5 Càrrega massiva de dades a Virtuoso

Un cop instal·lat Virtuoso i amb una instància arrencada, podem realitzar la càrrega massiva de triplets RDF.

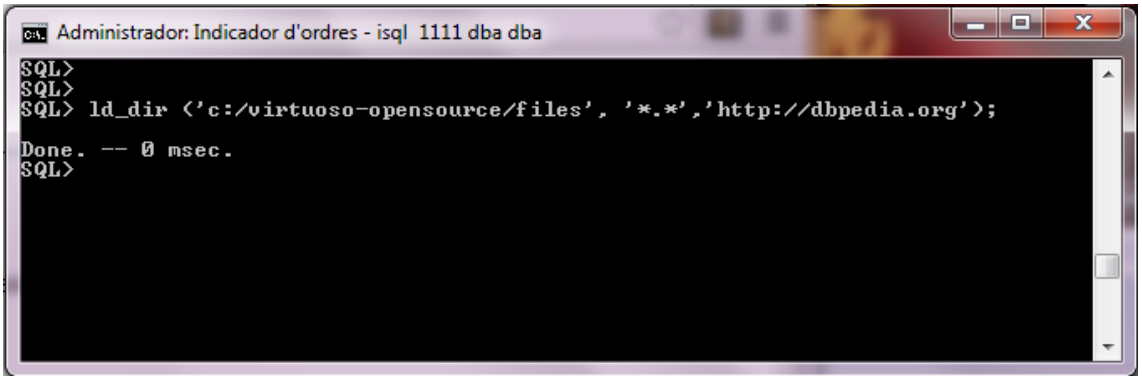
En el nostre exemple, utilitzarem dades descarregades de Dbpedia (http://wiki.dbpedia.org/Downloads38). L'objectiu d'aquest apartat és descriure com es fa la importació de dades a Virtuoso.

Primerament, accedirem a isql amb l'usuari dba i la clau per defecte “dba”.



Registrem els fitxers que seran carregats mitjançant la sentència ld_dir (càrrega des d'un directori especificat). En aquest exemple, es va descarregar el fitxer dbtunes_links.nt del lloc de descàrregues de Dbpedia (wiki.dbpedia.org/downloads38).

Els fitxers descarregats, amb extensió nt o .gz s'han d'ubicar a un directori amb permisos, és a dir, que estigui contemplat al paràmetre DirsAllowed del fitxer virtuoso.ini (en aquest cas el directori c:/virtuoso-opensource/files es va haver d'afegir al paràmetre DirsAllowed):



Creem un fitxer global.graph que contingui una línia amb la URI http://dbpedia.org

Executem rdf_loader_run:

```
Administrador: Indicador d'ordres - isql 1111 dba dba
SQL> rdf_loader_run ();

Done. -- 94 msec.
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

Comprovem si s'han carregat les dades correctament:

```
SQL>
SQL> sparql select count(*) from <http://dbpedia.org> where { ?s ?p ?o };
callret-0
INTEGER
_____________________________________________________________________________

839

1 Rows. -- 0 msec.
SQL>
SQL>
SQL>
SQL>
SQL>
SQL>
```

S'instal·len els packages Dbpedia i Rdf Mappers des de Virtuoso Conductor – Packages.

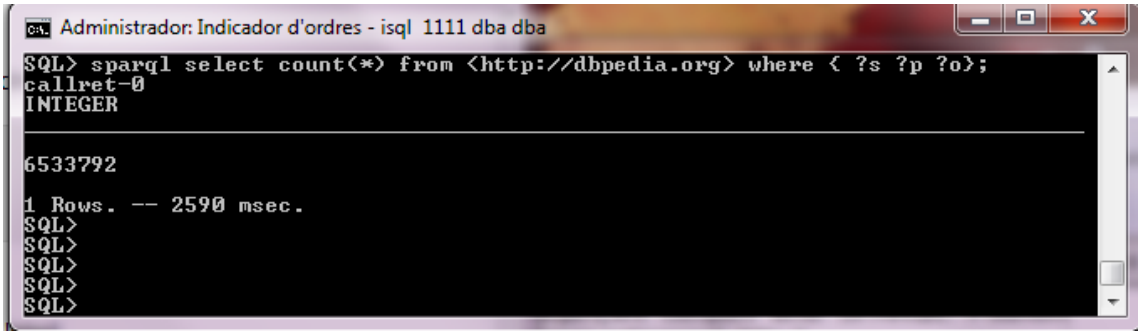
A partir d'aquest moment les dades allotjades a Virtuoso poden ser accedides usant un *browser* HTML o bé mitjançant cerques amb SPARQL. Podríem accedir també a un recurs concret a través del *browser* mitjançant: http://localhost:8890/resource/nom_recurs.

Per a comprovar que el mètode és vàlid també per a fitxers amb gran nombre de triplets, es carreguen els fitxers persondata_en.nq, persondata_en.nt i persondata_en.ttl (en total 2,63Gb).

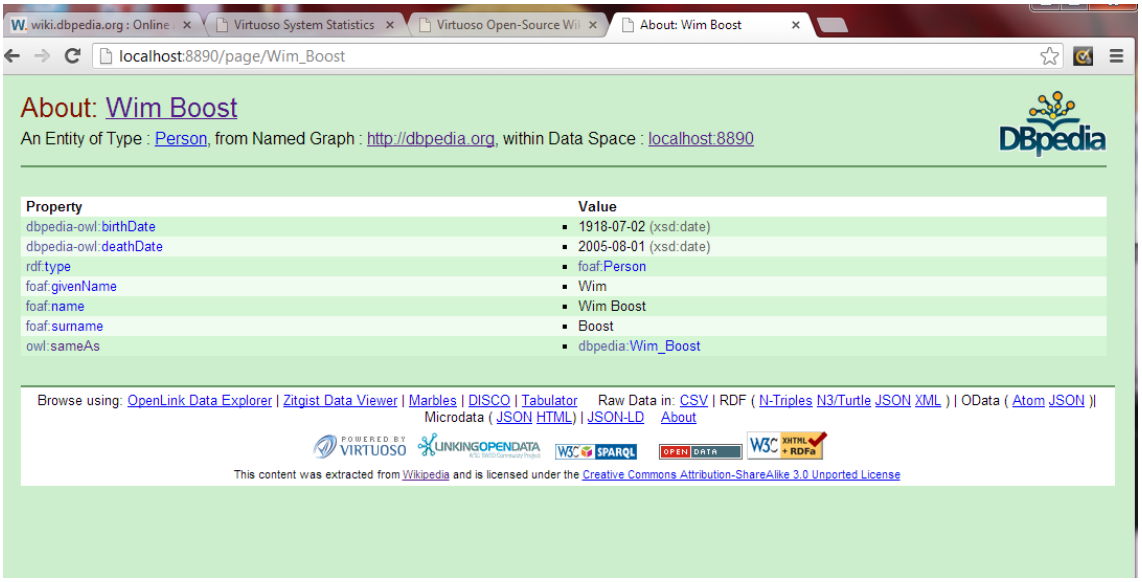
Càrrega dels tres fitxers persondata:

```
Administrador: Indicador d'ordres - isql 1111 dba dba
ld_dir ('/files','*.*','http://dbpedia.org')
SQL> ld_dir ('C:/virtuoso-opensource/files','*.*','http://dbpedia.org');

Done. -- 15 msec.
SQL> rdf_loader_run ();

Done. -- 4775643 msec.
SQL>
```

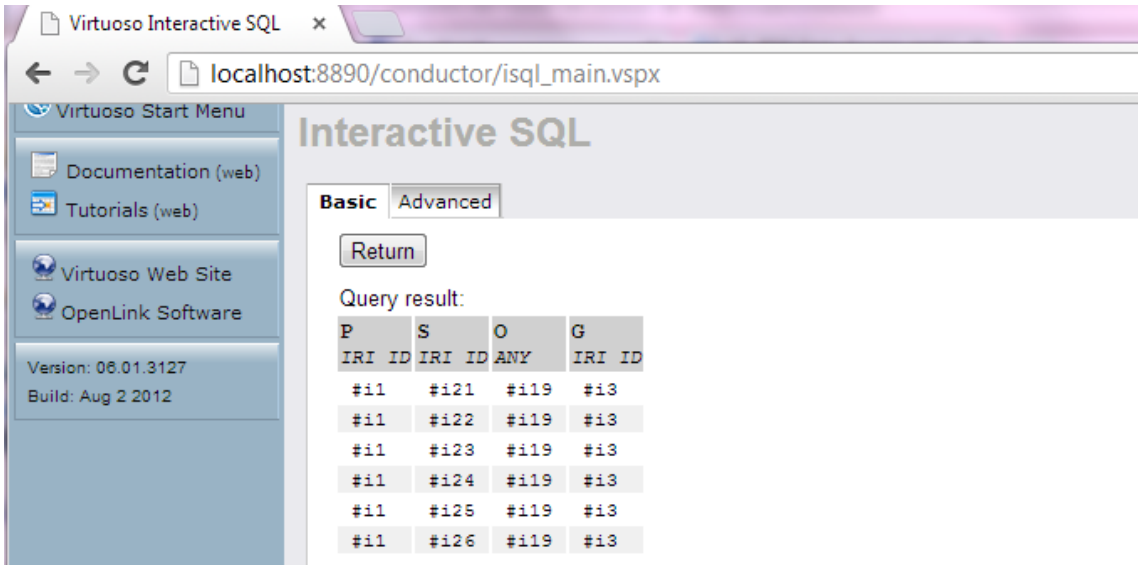
Comprovem que s'han carregat correctament:



Consulta d'un recurs de persondata mitjançant *browser:*



Podem comprovar que Virtuoso ha emmagatzemat a la taula DBA.DBA.RDF.QUAD els identificadors del graf, subjecte, objecte i predicat. Tret de l'objecte, la resta de dades són identificadors IRI que apunten una altra taula:



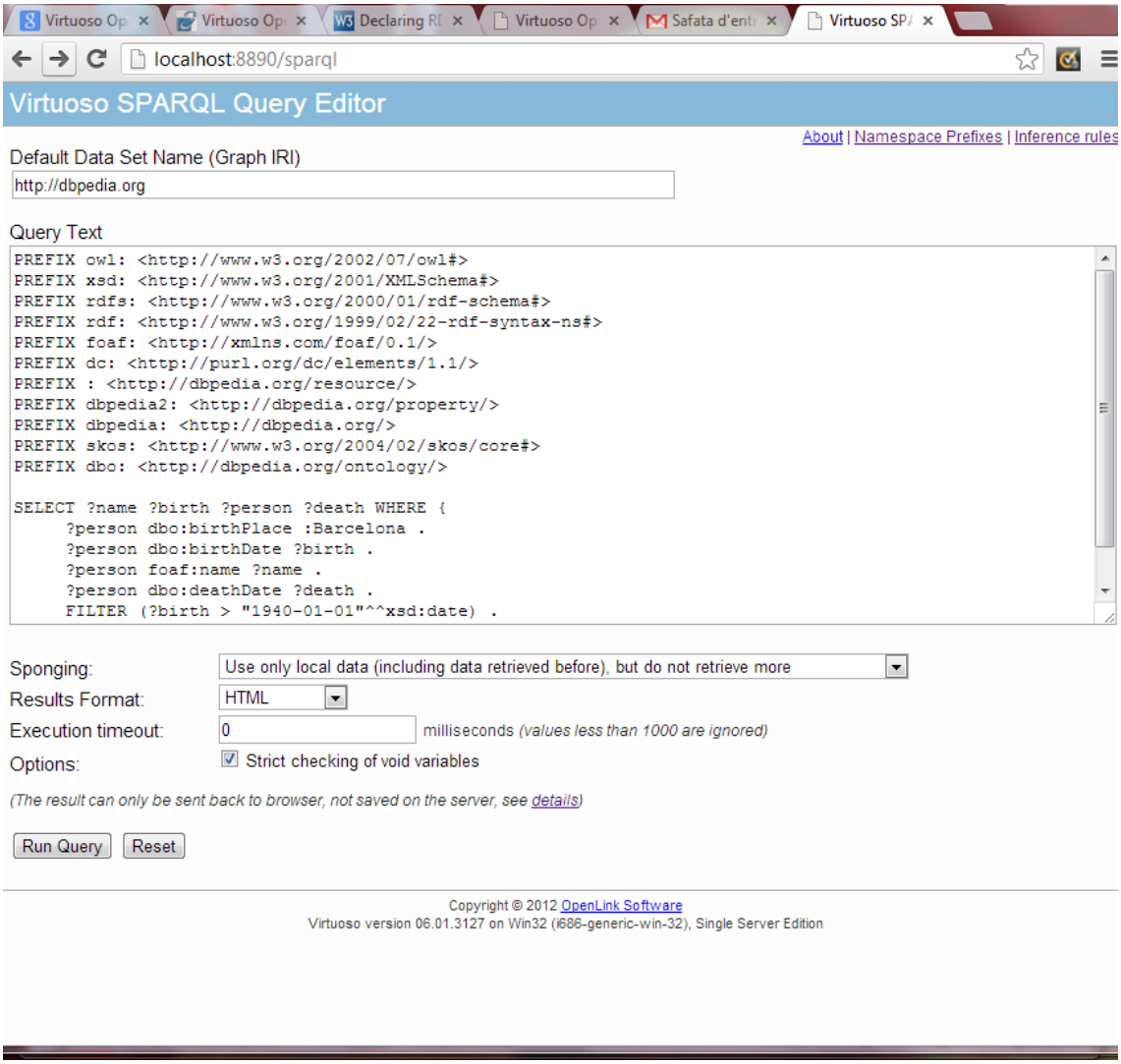
Les columnes són G pel graf, P pel predicat, S pel subjecte i O per l'objecte.

P, S i G són de tipus IRI_ID (la seva sintaxi és #i<NNN> on <NNN> permet fins a 20 dígits decimals. La columna O és del tipus ANY (qualsevol objecte serialitzable de SQL). La indexació suporta l'ordre lexicogràfic del tipus ANY, és a dir, entre dos elements d'un tipus compatible, l'ordre és aquell que té el tipus de dades per defecte.

Els objectes de més de 20 caràcters i aquells que s'han indexat per a cerques de text lliure s'emmagatzemen a la taula RDF_OBJ, podem consultar les seves dades mitjançant una consulta SQL:

SELECT RO_ID,RO_VAL,RO_LONG,RO_FLAGS,RO_DT_AND_LANG FROM DB.DBA.RDF_OBJ

A partir d'aquest moment podem fer consultes sobre el graf IRI des de SPARQL:



El resultat d'aquesta consulta (persones nascudes a Barcelona després de 01/01/1940 amb data de defunció especificada):

Virtuoso Op × Virtuoso Op × Declaring R × Virtuoso Op × Safata d'ent × localhost:88 ×

localhost:8890/sparql?default-graph-uri=http%3A%2F%2Fdbpedia.org&query=PREFIX+owl%3A+<h

name	birth	person	death
"Carles Sabater"@en	1962-09-21	http://dbpedia.org/resource/Carles_Sabater	1999-02-13
"Daniel Jarque"@en	1983-01-01	http://dbpedia.org/resource/Daniel_Jarque	2009-08-08
"Fernando Fernandez"@en	1940-02-07	http://dbpedia.org/resource/Fernando_Fernandez_(comics)	2010-08-09
"Helios Fernández"@en	1940-08-16	http://dbpedia.org/resource/Helios_Fern%C3%A1ndez	2004-10-03
"Helios Fernández"@en	1940-08-16	http://dbpedia.org/resource/Helios_Fernández	2004-10-03
"Lina Romay"@en	1954-06-25	http://dbpedia.org/resource/Lina_Romay	2012-02-15
"Maria Merce Marcal I Serra"@en	1952-11-13	http://dbpedia.org/resource/Maria_Merc%C3%A8_Mar%C3%A7al	1998-07-05
"Maria Merce Marcal I Serra"@en	1952-11-13	http://dbpedia.org/resource/Maria_Mercè_Marçal	1998-07-05
"Montserrat Figueras Garcia"@en	1942-03-15	http://dbpedia.org/resource/Montserrat_Figueras	2011-11-23
"Montserrat Roig Fransitorra"@en	1946-06-13	http://dbpedia.org/resource/Montserrat_Roig	1991-11-10
"Ramon Montesinos"@en	1943-05-31	http://dbpedia.org/resource/Ram%C3%B3n_Montesinos	2010-12-29
"Ramon Montesinos"@en	1943-05-31	http://dbpedia.org/resource/Ramón_Montesinos	2010-12-29

Per a analitzar el rendiment RDF de Virtuoso cal llençar les consultes amb la opció *Explain*. D'aquesta manera obtenim un informe que ens detalla l'estimació de la cardinalitat d'aquesta. Per a aprofundir en l'anàlisi, caldria dividir la consulta en fragments més petits i veure'n l'estimació fins a arribar al nivell de triplet - patró:

Interactive SQL - Google Chrome

localhost:8890/conductor/isql.vspx

Interactive SQL

Basic | Advanced

Local script [Choose File] No file chosen [Load] [Save] [SQL Builder]

☐ WebDAV source ☑ Local file

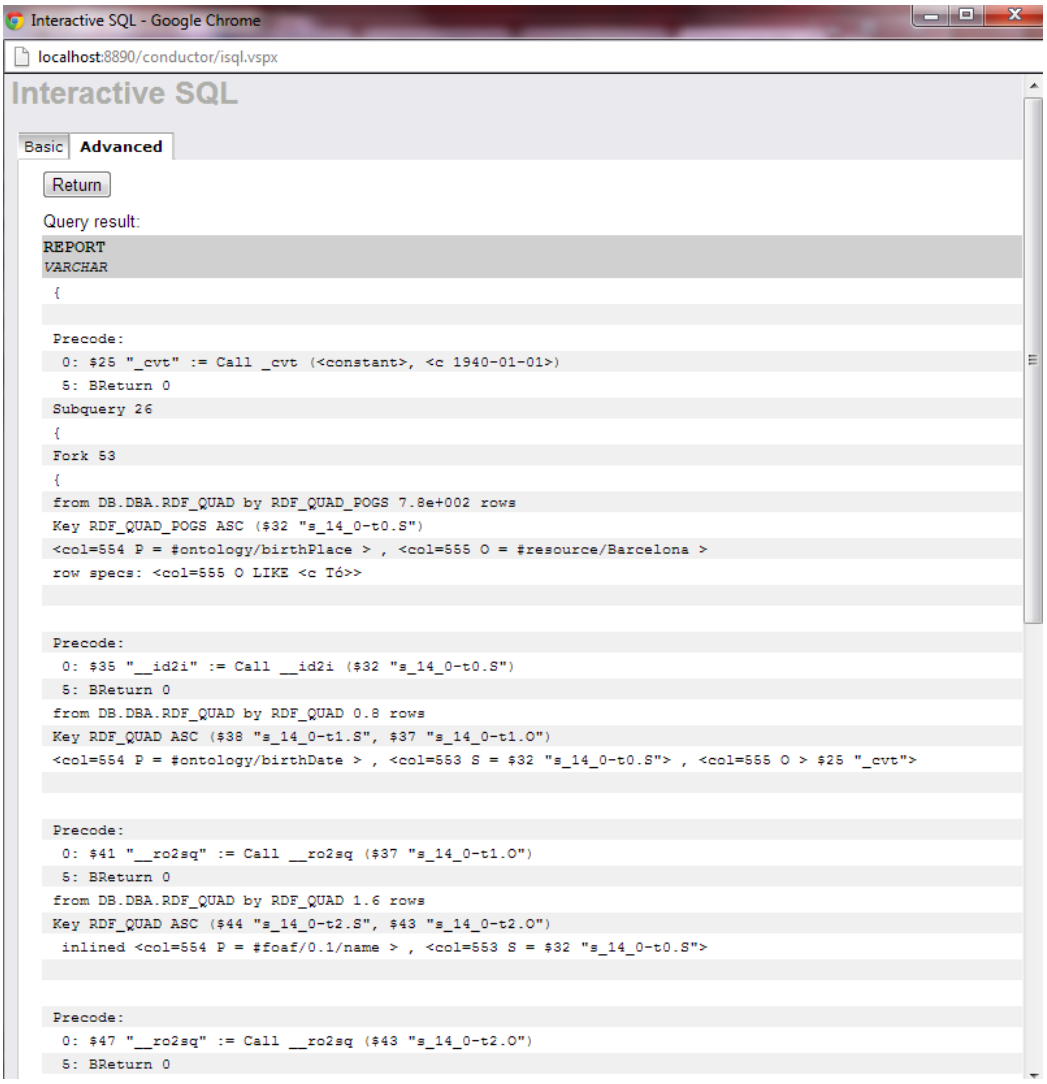
Run As dba Pwd ••• Transaction isolation Committed

[Execute] [Show Syntax] [Clear] [Close]

```
SPARQL
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?name ?birth ?person ?death WHERE {
     ?person dbo:birthPlace :Barcelona .
     ?person dbo:birthDate ?birth .
     ?person foaf:name ?name .
     ?person dbo:deathDate ?death .
     FILTER (?birth > "1940-01-01"^^xsd:date) .
}
ORDER BY ?name
```

☐ Explain ☑ Show no more than 100 rows

[Execute] [Clear] [Close]



5.6 Índexs a Virtuoso

L'esquema d'índex RDF a Virtuoso, està composat per dos índexs complerts més 3 índexs parcials. Aquest esquema generalment s'adapta a tot tipus de càrrega de treball, i tot i que normalment no és necessari, es podrien establir esquemes alternatius d'indexació.

L'esquema d'índexs es composa de:

- * PSOG ***:** Clau primària
- * POGS *: índex de mapa de bits per a cerques sobre el valor de l'objecte

- * SP *: índex parcial per als casos en que només s'especifica s
- * OP *: índex parcial per als casos en que només s'especifica o
- * GS *: índex parcial per als casos en que només s'especifica g

Aquest esquema es crea mitjançant aquestes declaracions:

```
CREATE TABLE DB.DBA.RDF_QUAD (
  G IRI_ID_8,
  S IRI_ID_8,
  P IRI_ID_8,
  O ANY,
  PRIMARY KEY (P, S, O, G)
  )
ALTER INDEX RDF_QUAD ON DB.DBA.RDF_QUAD
  PARTITION (S INT (0hexffff00));

CREATE DISTINCT NO PRIMARY KEY REF BITMAP INDEX RDF_QUAD_SP
  ON RDF_QUAD (S, P)
  PARTITION (S INT (0hexffff00));

CREATE BITMAP INDEX RDF_QUAD_POGS
  ON RDF_QUAD (P, O, G, S)
  PARTITION (O VARCHAR (-1, 0hexffff));

CREATE DISTINCT NO PRIMARY KEY REF BITMAP INDEX RDF_QUAD_GS
  ON RDF_QUAD (G, S)
  PARTITION (S INT (0hexffff00));

CREATE DISTINCT NO PRIMARY KEY REF INDEX RDF_QUAD_OP
  ON RDF_QUAD (O, P)
  PARTITION (O VARCHAR (-1, 0hexffff));
```

La idea és afavorir les consultes on s'especifica el predicat, així doncs el quad sencer es pot accedir de manera eficient quan P i S o P (i com a mínim un dels dos) són coneguts. L'agrupament de dades pel predicat millora el conjunt de treball, una pàgina que es llegeix de disc només tindrà entrades d'aquest predicat.

Per els casos menys freqüents en que només es coneix S, els diferents P dels S es troben al SP índex. Aquests parells SP s'utilitzen per a accedir a l'índex PSOG per a obtenir el O i el G. Els P dels S es troben després al SP índex i després d'això el quad es troba a l'índex PSOG.

Els índexs SP, OP I GS no emmagatzemen duplicats. Si hi ha un S amb varies valors per a P, només hi ha una sola entrada, i les entrades no s'eliminen de SP, OP o GS. Això no representa cap problema ja que sempre es consulta un índex complert (POSG o PSOG) amb la finalitat de saber si un quad en realitat

existeix. Quan s'actualitzen dades, normalment hi ha un graf que s'esborra completament i se'n insereix un de nou similar al seu lloc. Els índexs SP, OP i GD romanen després d'aquesta modificació pràcticament intactes.

No obstant això, és recomanable eliminar i tornar a crear els índexs SP, OP i GS periòdicament ja que quan hi ha actualitzacions freqüents els índexs es poden corrompre i això pot afectar el rendiment. Si això no fos possible, podríem tenir només índexs complerts a l'esquema (índex son cada clau conté totes les columnes de la clau principal del quad). Per a fer-ho, caldria especificar en el CREATE INDEX les opcions DISTINCT NO PRIMARY KEY REF, i d'aquesta manera els índexs seguirien sincronitzats.

La majoria de càrregues RDF tenen patrons de càrrega massiva i lectura intensiva amb molt pocs esborrats de dades. L'esquema d'índex per defecte està optimitzat per a aquests casos, oferint un estalvi considerable d'espai, ja que precisa entre un 60 i un 70% de l'espai d'un disseny amb 4 índexs complerts.

5.6.1 Especificar què cal indexar

L'objecte d'un determinat triplet és indexat depenent de les regles d'indexació. Si com a mínim una de les regles d'indexació coincideix amb el triplet, l'objecte s'indexa si és un *string*. Una regla d'indexació especifica un graf i un predicat, o bé pot ser un IRI o NULL, i en aquest cas equivaldria a tots els IRI.

Les regles tenen també una "raó" que pot ser utilitzada per les regles del grup en conjunts de dades d'aplicacions. Un triple deixarà d'estar indexat quan totes les regles que obliguen la seva indexació s'eliminin. Quan una aplicació requereix la indexació d'un conjunt de triplets, s'afegeixen regles amb aquesta finalitat. Aquestes regles estan etiquetades amb el nom de l'aplicació i la seva raó. Quan una aplicació ja no requereix la indexació, les normes que pertanyen a aquesta es poden eliminar.

La indexació s'habilita o es deshabilita per a una combinació graf/predicat específica amb:

```
create function DB.DBA.RDF_OBJ_FT_RULE_ADD
  (in rule_g varchar, in rule_p varchar, in reason varchar) returns integer
```

```
create function DB.DBA.RDF_OBJ_FT_RULE_DEL
  (in rule_g varchar, in rule_p varchar, in reason varchar) returns integer
```

La primera funció afegeix una regla. Els dos primers arguments son la representació en text dels IRI del graf i predicat. Si s'especifica NULL, tots els grafs o predicats coincidiran. Si en ambdós casos especifiquem NULL, tots els objectes amb valors *string* s'afegiran a l'índex de text.

Exemple:

```
DB.DBA.RDF_OBJ_FT_RULE_ADD (null, null, 'All');
```

```
DB.DBA.RDF_OBJ_FT_RULE_DEL (null, null, 'All');
```

La segona funció inverteix l'efecte de la primera. Només es pot esborrar una regla que estigui prèviament afegida.

L'argument "raó" és un *string* arbitrari que identifica l'aplicació que necessita la regla d'indexació. Dues aplicacions podrien afegir la mateixa regla. Si un objecte és indexat per més d'una regla, com que les dades de l'índex romanen lliures de duplicats, ni la mida de l'índex ni la velocitat es veuran afectades.

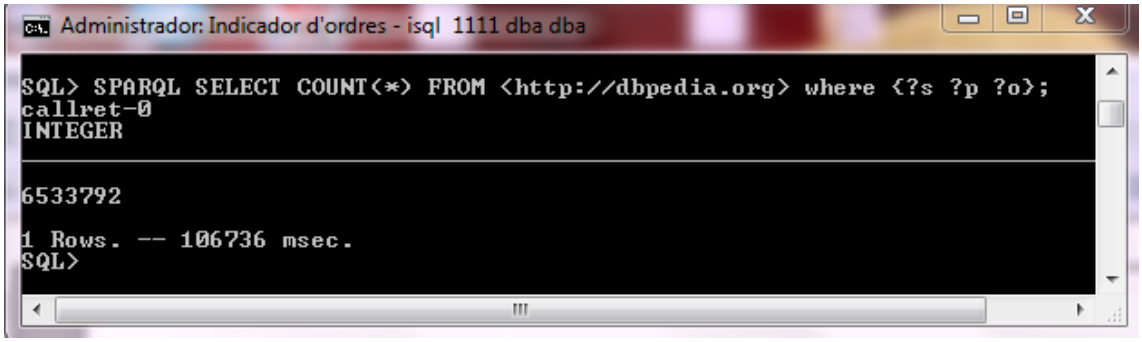
Si DB.DBA.RDF_OBJ_FT_RULE_ADD detecta que DB.DBA.RDF_QUAD conté quads les gràfiques i / o predicats els quals coincideixen amb la nova norma, però que no han estat indexats abans, aquests quads s'indexen automàticament.

No obstant això, la funció DB.DBA.RDF_OBJ_FT_RULE_DEL no elimina les dades de catalogació d'objectes relacionats .Així, la presència de dades d'indexació sobre un objecte no implica necessàriament que s'utilitzi en alguns quad coincidents amb alguna regla.

Les funcions anteriors retornen ú si la regla s'agrega o s'elimina i zero si la crida era redundant (la regla ja s'havia afegit abans o no hi havia cap regla per a esborrar).

Exemple:

La consulta es realitza sobre un conjunt de dades gran, amb 6.533.792 de triples:



Inicialment només es disposa dels índexs per defecte. S'executa una consulta per a obtenir tots els triples on l'objecte conté la cadena "Mon":

La consulta retorna 17.679 files, en 19’578 segons.

```
Administrador: Indicador d'ordres - isql 1111 dba dba
http://dbpedia.org/resource/Nicol%C3%A1s_Monckeberg
  http://xmlns.com/foaf/0.1/surname
    Monckeberg
http://dbpedia.org/resource/Nicol├ís_Monckeberg
  http://xmlns.com/foaf/0.1/surname
    Monckeberg
http://dbpedia.org/resource/Paul_O'Montis
  http://xmlns.com/foaf/0.1/surname
    OMontis
http://dbpedia.org/resource/Pierre_Monbeig
  http://xmlns.com/foaf/0.1/surname
    Monbeig
http://dbpedia.org/resource/Fernando_Monteiro_de_Castro_Soromenho
  http://xmlns.com/foaf/0.1/surname
    Monteiro de Castro Soromenho
http://dbpedia.org/resource/Adolfo_Casais_Monteiro
  http://xmlns.com/foaf/0.1/surname
    Casais Monteiro
http://dbpedia.org/resource/Guy_Montserret
  http://xmlns.com/foaf/0.1/surname
    Montserret
http://dbpedia.org/resource/Kael-Florent_Montout
  http://xmlns.com/foaf/0.1/surname
    Montout

17679 Rows. -- 19578 msec.
SQL>
```

S’afegeix una regla d’indexació sobre el graf IRI, per a tots els predicats (segón paràmetre NULL):

```
Administrador: Indicador d'ordres - isql 1111 dba dba
SQL>
SQL>
SQL>
SQL> DB.DBA.RDF_OBJ_FT_RULE_ADD('http://dbpedia.org', NULL, 'RESOURCE');

Done. -- 74178 msec.
SQL>
```

Per defecte, la indexació en aquest moment quedaria pendent de realitzar-se en un procés *batch* del servidor. Si volem forçar que l’índex sigui visible immediatament, s’ha de forçar amb la comanda DB.DBA.VT_INC_INDEX_DB_DBA_RDF_OBJ(); :

```
Administrador: Indicador d'ordres - isql 1111 dba dba
SQL>
SQL>
SQL>
SQL> DB.DBA.RDF_OBJ_FT_RULE_ADD('http://dbpedia.org', NULL, 'RESOURCE');

Done. -- 74178 msec.
SQL> DB.DBA.VT_INC_INDEX_DB_DBA_RDF_OBJ();

Done. -- 15 msec.
SQL>
```

Posteriorment, llencem de nou la consulta per a veure que l'índex ha optimitzat aquesta en 1295 mseg.

```
http://dbpedia.org/resource/Nicol├ís_Monckeberg
  http://xmlns.com/foaf/0.1/surname
    Monckeberg
http://dbpedia.org/resource/Paul_O'Montis
  http://xmlns.com/foaf/0.1/surname
    OMontis
http://dbpedia.org/resource/Pierre_Monbeig
  http://xmlns.com/foaf/0.1/surname
    Monbeig
http://dbpedia.org/resource/Fernando_Monteiro_de_Castro_Soromenho
  http://xmlns.com/foaf/0.1/surname
    Monteiro de Castro Soromenho
http://dbpedia.org/resource/Adolfo_Casais_Monteiro
  http://xmlns.com/foaf/0.1/surname
    Casais Monteiro
http://dbpedia.org/resource/Guy_Montserret
  http://xmlns.com/foaf/0.1/surname
    Montserret
http://dbpedia.org/resource/Kael-Florent_Montout
  http://xmlns.com/foaf/0.1/surname
    Montout

17679 Rows. -- 18283 msec.
SQL>
```

6. Conclusions

- Donat que els meus coneixements sobre el tema en el moment d'iniciar el projecte eren nuls, l'elaboració d'aquest ha estat un aprenentatge continu i ha suposat una gran tasca d'investigació.

- A banda de l'aprenentatge en el tema del projecte en sí mateix, ha representat un gran repte el fet de planificar amb realisme primerament, i posteriorment intentar ajustar-se al màxim a aquesta planificació. Tant per diferents factors externs com per problemes que han anat sorgint vinculats a la manca de coneixements sobre el tema, ha estat complicat seguir aquesta planificació amb exactitud. No obstant, per a mi ha resultat una gran lliçó haver pogut superar aquests obstacles, ja que en moments crítics s'ha intentat anar fent minúsculs avanços que permetien mantenir la motivació alta i que finalment han permès avançar fins a concloure en treball.

- Considero que els objectius plantejats s'han assolit, tot i que en l'apartat de l'anàlisi de Virtuoso considero que es podria haver aprofundit molt més. El motiu pel qual això no ha estat possible ha estat que el temps destinat a la recerca d'informació s'ha allargat de manera excessiva durant l'execució de la PAC3.

- No s'ha fet un seguiment exhaustiu de la planificació, tot i que considero que aquesta s'havia fet d'una manera bastant realista. El problema ha estat que el temps destinat a la investigació, a la recerca i a la tria de la informació ha superat el que hi havia destinat en aquesta. Això ha provocat que no s'hagi pogut fer prelliuraments al consultor amb suficient antelació, ja que suposaven disposar dels treballs una setmana abans del termini d'entrega i això no ha estat possible.

- Considero que en aquest projecte han quedat algunes línies de treball pendents sobretot en l'anàlisi del rendiment de Virtuoso. Per exemple, s'hauria pogut incloure un anàlisi del creixement el temps de resposta depenent del volum del conjunt de dades, en conjunts indexats i sense indexar. Tampoc s'ha realitzat un anàlisi del temps de càrrega de dades en relació amb el volum d'aquestes.

7. Glossari

BLOB (Binari Large Object) Conjunt de dades binàries emmagatzemades com una sola entitat en un SGBD

IRI (Internationalized Resource Identifier) Identificador de recursos que permet l'ús de caràcters internacionals (Unicode/ISO10646).

Middleware (Programari intermediari) Programari que permet a una aplicació comunicar-se o interactuar amb d'altres aplicacions.

Read commited És el nivell d'aïllament més baix en base de dades. Permet llegir dades quan potser altra transacció l'està modificant.

SOAP (Simple Object Access Protocol) protocol estàndard perquè dos objectes puguen comunicar-se intercanviant dades XML.

SPARQL Endpoint Protocol conforme a SPARQL que permet als usuaris (normalment automàtics) consultar dades amb aquest llenguatge. Els resultats es proveeixen en un format processable automàticament.

URI (Uniform Resource Identifier) Cadena de caràcters per identificar un recurs. Aquesta identificació permet interactuar amb aquest recurs en una xarxa (per exemple WWW).

W3C (World Wide Web Consortium) Comunitat internacional que desenvolupa estàndards per assegurar el creixement de la web a llarg termini.

WebDAV (Web Distributed Authoring and Versioning) Conjunt d'extensions d'Http per possibilitar l'edició de fitxers d'un lloc web.

8. Bibliografia

1. Antoniou, Grigoris and Van Harmelen, Frank. *A Semantic Web Primer* . London: The MIT Press, 2004.

2. Cacheiro M. L. y Lago B. *La Web Semántica en Educación* [En línia]. (2008). Document disponible a: http://e-spacio.uned.es/fez/eserv.php?pid=bibliuned:425-Mlcacheiro5040&dsID=Documento.pdf

3. Hull, Richard. *Semantic Database Modeling: Survey, Applications, and Research Issues.* Computer Science Department, University of Southern California, Los Angeles. Document disponible a : http://www.cc.gatech.edu/computing/Database/readinggroup/articles/p201-hull.pdf

4. http://www.di.uniovi.es/~labra/cursos/XML/SPARQL.pdf (visitada en data 20/10/12)

5. http://www.unalmed.edu.co/~mstabare/acercamiento_modelos_bases.pdf (visitada en data 10/10/12)

6. http://skos.um.es/TR/rdf-sparql-query/ (visitada en data 20/10/12)

7. http://openaccess.uoc.edu/webapps/o2/bitstream/10609/14980/7/jsansanomTFC0612memoria.pdf (visitada en data 10/10/12)

8. http://seccperu.org/files/bdsemantica.pdf (visitada en data 15/10/12)

9. http://www.lsi.us.es/docs/doctorado/memorias/Torre-Moreno-Pablo-Memoria-Investigacion.pdf (visitada en data 15/10/12)

10. http://www.scribd.com/doc/79420435/BasesDatos (visitada en data 15/10/12)

11. http://es.scribd.com/doc/74901065/Lenguaje-SQL-de-Bases-de-Datos-Objeto-Relacionales-BDOR-y-XML (visitada en data 17/10/12)

12. http://www.w3c.es/Divulgacion/GuiasBreves/WebSemantica (visitada en data 20/10/12)

13. http://blog.evoit.com/2011/08/8-preguntas-sobre-la-web-semantica/ (visitada en data 15/10/12)

14. http://semantizandolaweb.wordpress.com/2011/11/21/el-modelo-abstracto-de-rdf-sentencias/ (visitada en data 10/10/12)

15. http://www.utpl.edu.ec/ecc/wiki/index.php/Aplicaciones_de_Sem%C3%A1ntica_en_Base_de_Datos (visitada en data 10/10/12)

16. http://www.rdfabout.com/comparisons.xpd (visitada en data 20/10/12)

17. http://opendata.euskadi.net/w79-utilizar/es/contenidos/informacion/como_utilizar_datos/es_como_uti/adjuntos/rdf.pdf (visitada en data 08/10/12)

18. http://www.w3.org/TR/owl-guide/ (visitada en data 08/10/12)

19. http://www.w3.org/2007/09/OWL-Overview-es.html (visitada en data 08/10/12)

20. http://www.unalmed.edu.co/~mstabare/acercamiento_modelos_bases.pdf (visitada en data 20/10/12)

21. http://openaccess.uoc.edu/webapps/o2/bitstream/10609/14980/7/jsansanomTFC0612memoria.pdf (pag 37) (visitada en data 20/10/12)

22. http://www.ciw.cl/material/emergentes2006/gutierrez.pdf (visitada en data 12/10/12)

23. http://www.di.uniovi.es/~labra/cursos/ver06/pres/XMLBD.pdf (visitada en data 08/10/12)

24. http://www.w3.org/2001/sw/rdb2rdf/wiki/Mapping_SQL_datatypes_to_XML_Schema_datatypes (visitada en data 20/10/12)

25. http://sinbad.dit.upm.es/docencia/grado/curso1112/2-Slices-BSDT2011-12-Introduccion-a-XML%26SQL.pdf (visitada en data 10/10/12)

26. http://www.w3.org/wiki/LargeTripleStores (visitada en data 22/10/12)

27. https://www.box.com/shared/xqzeenelfq (visitada en data 22/10/12)

28. http://www.scribd.com/doc/79420435/BasesDatos (visitada en data 22/10/12)

29. http://openaccess.uoc.edu/webapps/o2/bitstream/10609/366/1/38369tfc.pdf (visitada en data 22/10/12)

30. http://www.rdfabout.com/intro/?section=9 (visitada en data 20/10/12)

31. http://www.google.es/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&ved=0CCMQFjAA&url=http%3A%2F%2Fserqlsparql.50webs.com%2FSeRQL_SPARQL.doc&ei=KkmSUOqvGPOb1AXaz4HADQ&usg=AFQjCNH2af2RgCdeWGKECrLpLFEjT7NurA (visitada en data 1/11/12)

32. http://en.wikipedia.org/wiki/SPARQL (visitada en data 1/11/12)

33. http://manzanamecanica.org/2012/01/tutorial_creando_aplicaciones_basadas_en_linked_data_parte_13.html (visitada en data 1/11/12)

34. http://www.slideshare.net/ldodds/sparql-query-forms (visitada en data 1/11/12)

35. http://www.desarrolloweb.com/articulos/2263.php (visitada en data 24/11/12)

36. http://mat21.etsii.upm.es/mbs/MechXML/ejemplo_de_fichero_xml_schema.htm (visitada en data 24/11/12)

37. http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VOSUsageWindows (visitada en data 10/11/12)

38. http://wiki.dbpedia.org/Downloads38 (visitada el 1/12/12)

39. http://wiki.dbpedia.org/OnlineAccess#h28-2 (visitada el 1/12/12)

40. http://www.openlinksw.com/dataspace/dav/wiki/Main/VirtBulkRDFLoader (visitada el 01/01/12)

41. http://www.openlinksw.com/dataspace/dav/wiki/Main/VirtBulkRDFLoaderExampleDbpedia (visitada el 1/1/12)

42. http://www.openlinksw.com/dataspace/dav/wiki/Main/VirtRDFPerformanceTuning#Index%20Scheme%20Selection (visitada el 7/12/12)

43. http://rua.ua.es/dspace/bitstream/10045/19111/6/06-XML%20Schema.pdf (visitada el 17/12/12)