Gigantisch semantisch

Marc Lieber
Trivadis AG
Basel CH

Schlüsselworte

Semantic Data Management, Oracle Semantic Web, Ontology, RDF/XML

Einleitung

Haben Sie schon einmal probiert semantisch Daten abzufragen?

Können Sie etwas mit dem Begriff Ontologie anfangen?

Nein, falsch gedacht: Ontologie hat nichts mit Medizin zu tun sondern ist eine Sprache um Wissen zu präsentieren, aber auch um kontextabhängige Abfragen für Endbenutzer zu erleichtern.

Neben einer Einführung in die Begrifflichkeiten semantischer Sprache wird Ihnen konkret anhand eines Projektes beschrieben wie man solche Lösungen implementieren kann, worauf man achten muss und warum semantische Datenmodelle die Zukunft sind.

Als Oracle Entwickler, habe ich viele Jahre lang hauptsächlich mit relationalen Daten-Modellen gearbeitet. XML in der Datenbank ist bereits eine Erweiterung der klassischen Datenmodellierung, aber Oracle Web Semantics ist eine ganz andere, völlig neue Welt. Jedes „Datenmodell“ ist in einer einzelnen Tabelle im DB Schema MDSYS gespeichert, die Daten und die Struktur werden als Triples gemischt abgelegt. Man fragt sich sofort, ob das wirklich performant sein kann, ob es produktiv einsetzbar ist, wie man diese Daten überhaupt abfragen kann?

Unsere Erfahrungen

Hier sind meine ersten Erfahrungen mit Oracle Web Semantics 11gR2:

Ziel unseres Projekts war ein relational strukturiertes Datenmodell in einem Web Semantik Modell zu migrieren. Die Applikation ist ein Domain spezifisches Such-Werkzeug, das zur Darstellung von Wissensmanagement im Pharmaceutical Bereich benutzt wird.

Die Applikation enthält einen Domain- spezifischen Teil, einen Terminologie-Teil (Thesaurus) und Pointers zu URLs . Der Terminologie-Teil beinhaltet Wörter und ihre Relation zu anderen Wörter (preferred/normalized terms, synonyms, broader terms, narrower terms).

Jeder Thesaurus-Verantwortliche in der Applikation (Content Owner) ist verantwortlich für ein oder mehrere spezifische Domains.

Die Arbeitseinheit ist ein Concept. Jedes Concept gehört zu einem ConceptType.
Es gibt zwei Typen von Relationen zwischen Concepts : Relation innerhalb eines ConceptType‘s oder Concept-übergreifende inter-conceptType Beziehungen.

Beispielen:

Der Concept „Oracle Corp.“ gehört zum Concept Type „Company“. Dieser Concept hat unter anderem eine Property “Synonym” („Oracle Corp.“, synonyms „Oracle,Sun“) und dabei muss es einer geben mit property „preferred Term“ („Oracle Corp.“).

Der Concept „DB 11gR2“ gehört zum Concept Type „Product“ und hat eine Property „isLicencedBy“ von Typ inter-conceptType Relation mit dem Concept „Oracle Corp.“.

Der Concept „Homo Sapiens“ gehört zum Concept Type “ Taxonomy“

„Homo Sapiens“ wiederum hat eine Relation von Type „isMemberOf“ mit dem Concept „mammalian“, auch von Type “ Taxonomy“

Der Concept „KLK1“ gehört zum Concept Type „Gene“ und hat eine Relation von Typ „hasOrganism“ mit dem Concept „Homo Sapiens“

Die Pointers (IDs) erlauben hierbei die Cross-Referenzierung von Informationen aus externen wissenschaftlichen Quellen und fremden Applikationen.

Die alte Applikation war in einem klassischen Relationalen Modell gespeichert. Die Daten werden monatlich komplett neu geladen und nachdem sie geprüft wurden per Switch produktiv geschaltet. Die komplexen Abhängigkeiten zwischen Concepts wurden teilweise in SQL durch `CONNECT BY` implementiert und aus Performance-Gründen mit Materialized Views unterstützt. Die neue Version dieser Applikation soll nun komplexere Abfragen unterstützen können und darüber hinaus auch die zunehmenden Anforderungen an Datendurchsatz unterstützen können. Da Flexibilität und Erweiterbarkeit nicht die Stärke einer relationalen DB sind hat sich der Kunde für Oracle Web Semantics entschieden.

Warum?

Auf dem Papier passt die Web Semantics Technologie perfekt zu den Kunden-Bedürfnisse.

Semantische Technologien dienen als Sprache zur Präsentation von Wissen, beschreiben Methoden und enthalten Tools für die Ontologie. Die Abkürzungen RDF, OWL und SKOS stehen für einheitliche, offene Standards für die Beschreibung solche Informationen. Eine grundlegende Anforderung an diese Standards besteht in der Flexibilität und Erweiterbarkeit. Das World Wide Web Consortium (W3C) sorgt dafür, diese Ontologiesprachen-Standards zu beschreiben.

Mit Hilfe von RDF und RDFS werden die Klassen, Subklassen und Properties beschrieben

OWL Ontologien sind Erweiterungen von RDF basierten Ontologien und bringen noch zusätzliche Beschreibungen zu den enthaltenen Klassen und Properties

Die SKOS Techonologie sind durch W3C vordefinierte Regeln und Beschreibungen, die für Terminology und Taxonomy geeignet sind. Nichts desto trotz werden wir es zuerst ohne SKOS versuchen.

Das Semantic Web bietet Methoden zur Schlussfolgerung von „neuen“ Informationen. Diese Methoden extrahieren impliziten Informationen aus den Daten und erlauben uns dann die grösstmögliche Nutzung dieses Wissens aufzubauen.

Oracle bietet Web Semantics in der DB ab Version 10g, doch die größten Fortschritte hat Oracle eindeutig mit der Version 11gR2 gemacht.

Anforderung

- Die alten Funktionalitäten müssen erhalten bleiben:
  - Gute Performanz,
  - Hierarchische Abfragen,
  - Versionierung auf Concept Ebene.
- Jeder Content Owner muss in der Lage sein, die Daten inkrementell zu laden. Da die Einheit ein Concept ist, bedeutet dies, dass die alten Daten für diesen Concept zuerst gelöscht werden müssen. Erst danach werden die neue Daten mit einer neuen Versionsnummer geladen
- Die Daten müssen im RDF/XML Format geliefert werden
- Neue Concept Typen, Properties, Relationen oder Regeln sollen durch die Content Owner jederzeit erstellt werden können, müssen jedoch Laden zuerst von dem Application Aministrator validiert werden.
- Die Produktionsdaten sollen jetzt öfter aktualisiert werden. Eine Loading Exercise beinhaltet Änderungen in mehreren Concept Types und soll die Daten gegen das Daten Modell prüfen können, bevor es mit der Produktion gemerged wird.

In der Migrations-Phase, werden die Daten via PL/SQL aus dem alten Daten-Modell in einer Staging Tabelle geschrieben und als Triples gespeichert (N-Triple Format). Die Struktur dieser Tabelle ist von Oracle vordefiniert und dieselbe Tabelle wird später mit Jena für Bulk-Loads verwendet. Hierfür gibt es wiederum eine Prozedur SEM_API.BULK_LOAD_FROM_STAGING.

Unser erster Versuch erfolgte mit Oracle Workspace und seiner Versionierungs- Funktionalität. Wir wollten die produktive Umgebung so aufbauen, dass jeder Thesaurus-Verantwortliche in einem eigenen Oracle Workspace arbeitet. So kann jeder seine private Version der Semantic-Daten pflegen und gleichzeitig die produktiven Daten anschauen. Dabei bleiben die Daten privat, solange kein Merge mit den produktiven Daten stattgefunden hat. Die produktiven Daten liegen im Workspace „Live“. Der Oracle Workspace bietet das hierfür notwendige Versions-Management.

Leider mussten wir eine andere Lösung finden, einerseits aus Performance-Gründen, aber auch weil es zu viele Restriktionen gab. Eine Limitierung ist zum Beispiel, dass Daten nicht mehr per Bulk-Load importiert werden können.

Aus diesem Grund benutzen wir zwei Semantic Tabellen - eine produktive und eine Arbeitstabelle. Am Ende einer jeden Loading Exercise werden beide Tabellen ausgetauscht.

Probleme und ihre Lösungen

Nun blieben noch ein paar wichtige Probleme zu lösen:

- *Wie kann man die Triples die zu einer alten Concept Version gehören löschen?* Oracle bietet mit SPARQL eine Möglichkeit die Triple IDs zu ermitteln, doch leider ist diese Methode viel zu langsam. Entsprechend haben wir eine viel effizientere Lösung gefunden, die direkt auf die relationale Semantic Tabelle zugreift. Eine der grösste Challenges liegt hierbei

in der Löschung von Triples mit sogenannten Blank Nodes, weil Oracle diese Blank Nodes bei einem Merge umbenennt.

- *Wie kann man ein Rollback zu einer alten Concept Version machen?* Wir haben uns entschieden die Triples in der Staging Tabelle nicht zu löschen. Die Tabelle ist per Version partitioniert. Ein Rollback wird dann mittels PL/SQL gelöst.

- *Wie kann man Versionen vergleichen?* Eine PL/SQL Prozedur generiert das RDF/XML aus der Datenbank zurück. Mit der Oracle Funktion XMLDiff können wir dann die zwei Versionen vergleichen.

- *Wie können wir sicherstellen, dass die neu geladenen Triples valid sind gegenüber das Semantic Modell?* Das ankommende RDF/XML wird zuerst mittels Java Parser gegen ein XML Schema validiert. Nach dem Laden der Staging Tabellen werden nochmals Checks mit PL/SQL gemacht.

Modellierung

Die RDF/OWL Modellierung ist mit Topbraid gemacht. Topbraid bietet die Möglichkeit das OWL Modell direkt in Oracle zu speichern, zu editieren und mit dem Oracle Jena Adapter, die Datenbank gespeicherte Daten mit SPARQL abzufragen. Topbraid ist geignet für die Modellierung und Prototyping, ist jedoch zu ineffizient bei grossen Datenmengen.

Deshalb haben wir uns entschieden den OWL Modell in einem separaten Modell in der DB zu speichern, in einer zweiten Phase aus diesem Modell Triples zu generieren und diese in die erwähnte Staging Tabelle zu speichern.

Inferencing

Nachdem das OWL Modell und die Daten in das Oracle Semantic Datenmodell geladen sind, muss noch ein wichtiger Prozess stattfinden: das „Inferencing“ oder auch „Entailment“ genannt. Das Entailment basiert auf den Regeln, die Sie implementieren wollen. In unserem Fall hat sich der Kunde für OWLPrime entschieden. Das Entailment generiert neue Triples, die physisch im Schema MDSYS gespeichert sind. Diese Vorspeicherung von Inferred Daten führt dazu, dass bei Abfragen kein on-the-fly reasoning gemacht werden muss und ist deshalb ziemlich performant. Ein geläufiges Beispiel für eine Inference : A=B und B=C dann A=C (Inferred)..

Die PL/SQL Prozedur SEM_APIS.CREATE_ENTAILMENT generiert diese zusätzliche Triples für uns.

Die hier im Einsatz befindliche Applikation beinhaltet 52 Million Triples. Die Migration der Daten hierfür dauert 2 Stunden inklusiv Inferencing. Das Inferencing alleine ist nach weniger als eine Stunde beendet. Nach dem Infererencing sind noch 50 Million Triples dazugekommen. Vom Typ Pointer URLs existieren 30 Millionen Einträge und bleiben in einer relationalen Tabelle gespeichert, weil sie kein Inferencing brauchen.

Daten laden

Nach der Migration werden die Daten mit Jena geladen. Damit Sie eine Vorstellung bekommen: ein 2 Gigabyte RDF/XML Dokument braucht ein paar Sekunden für die XML-Schema Validation und ein paar Minuten für den Load in der Staging Tabelle.

Ein grosser Teil der SPARQL Syntax ist in Oracle SEM_MATCH unterstützt. Die Syntax von SEM_MATCH ist nicht sofort einleuchtend. Grundsätzlich verlangt die Implementierung von SPARQL in Oracle schon ein gewisses „fine tuning“, bis das Query optimal läuft. Genau wie bei SQL kann ein ineffizientes Query sehr schnell die Perfomance der Datenbank extrem verschlechtern.

Die gespeicherten URLs werden mit der Funktion SEM_RELATED abgefragt. Die relationale Tabelle braucht noch dazu ein Index von Typ MDSYS.SEM_INDEXTYPE auf der entsprechenden Ontology-Terms Spalte.

Frontend

In der ersten Phase wollten wir das Frontend nicht ändern und uns auf die bestehende Funktionalität beschränken, die die alte Applikation bereits liefern kann.
Die existierende Java-basierte Applikation liest die Daten aus der Oracle DB via SQL Queries auf die vorher bereits erwähnten Materialized Views. Eine wichtige Vorgabe ist, dass der Zugriff immer noch via SQL stattfinden muss. Also wurden die exakt selben Views jetzt mittels SPARQL aus dem Semantic Model gefüllt.

Die Java-basierte Applikation wird dann neu umgeschrieben. Mit Hilfe von Joseki wird es möglich sein für einen Super-User die SPARQL Abfragen selber zu formulieren.

Die Lösung hierfür ist, dass die SEM_MATCH und SEM_RELATED Abfragen teilweise mit Views in SQL übersetzt werden können. Alle anderen speziellen Anforderungen werden mit PL/SQL Funktionen abgefragt

Kernaussage

Unsere ersten Erfahrungen mit Oracle Web Semantics sind sehr positiv. Wir denken, dass wir in der Lage sind die zukünftigen Anforderungen besser unterstützen zu können. Die Performance unserer SQL Abfragen mit SEM_MATCH und SEM_RELATED sind meistens befriedigend (unter einer Sekunde). Jedoch gibt es noch ein großes Potenzial für der Verbesserung der Performance bei den Einstellungen der Datenbank und bei der Nutzung eines dedizierten Linux Server mit hoher Leistungsfähigkeit.

Kontaktadresse:

Marc Lieber

Trivadis AG

Elisabethenanlag 9

CH-4051 Basel

Telefon:	+41 (0) 79-457-97-61
Fax:	+41 (0) 61-279-97-56
E-Mail	marc.lieber@trivadis.com
Internet:	www.trivadis.com