Graph Analytics and RDF with Oracle Database

Technical Feature Overview

April 1, 2020

PURPOSE STATEMENT

This document provides an overview of the Graph features included with Oracle Database. It is intended solely to help you assess the business benefits of using these features and to plan your I.T. projects.

DISCLAIMER

TABLE OF CONTENTS

INTRODUCTION

Above Oracle provides the industry's leading converged, multimodel database management platform. Oracle Database brings advanced features for management and analysis of property graphs and RDF knowledge graph applications to Oracle Database. This paper addresses the graph features of Oracle Database.

Graph analysis is about understanding relationships. As applications and infrastructure evolve, as new technologies and platforms emerge, we find new ways to incorporate and exploit social information into business and analytic workflows. The emergence of cloud services, mobile tracking, real time systems, social media, and Internet of Things create new challenges to manage the volume of data, but more importantly, to discover patterns, connections, and relationships.

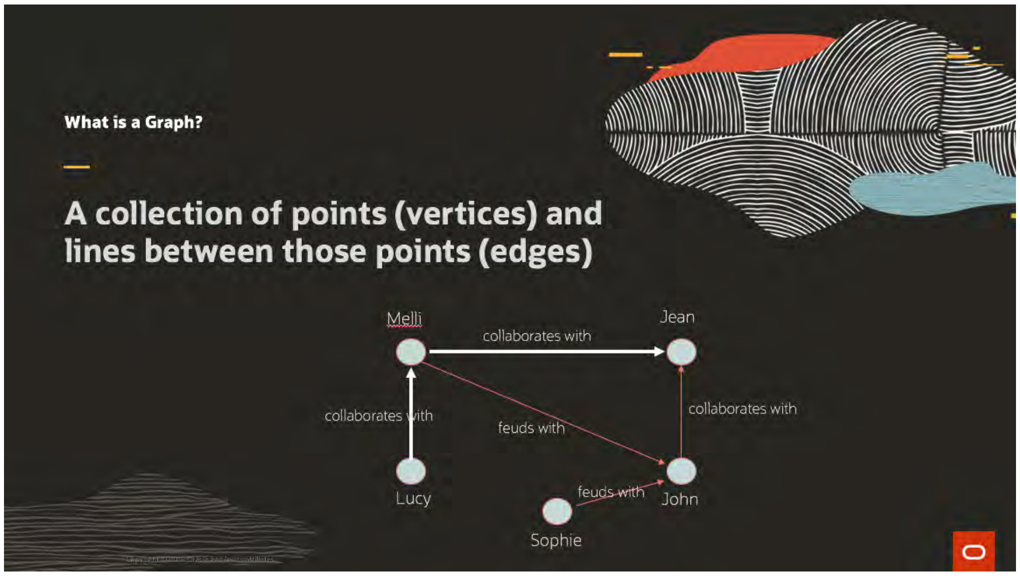
To address these opportunities, Oracle Database includes a property graph feature that provides graph storage, a SQL-like graph query language, and powerful built-in social graph analytics for making recommendations, finding communities and influencers, pattern matching, and identifying fraud and other anomalies. The property graph features and the RDF graph features can be used together; you can invoke graph algorithms and perform property graph queries on an RDF knowledge graph and you can make RDF queries (using the SPARQL language) against a property graph.

While most graph offerings tend to focus on providing either analytics or a graph database, Oracle Database Property Graph is unique in that it has both: a powerful in-memory analyst with over 50 built-in analytics and a scalable graph database.

The RDF graph feature in Oracle Database is a proven, special-purpose graph conforming to World Wide Web Consortium (W3C) standards. It provides parallelized RDF data storage, querying and inferencing that is used in semantic data integration and linked open data applications. The RDF support in Oracle Database is open, secure and scales to over a trillion triples.

PROPERTY GRAPH OVERVIEW

General-purpose property graph support is included in Oracle Database. Much of the Big Data generated these days contains inherent relationships between the collected data entities. These relationships are modeled in Oracle Database as a property graph – a set of connected entities.



What is a Graph?

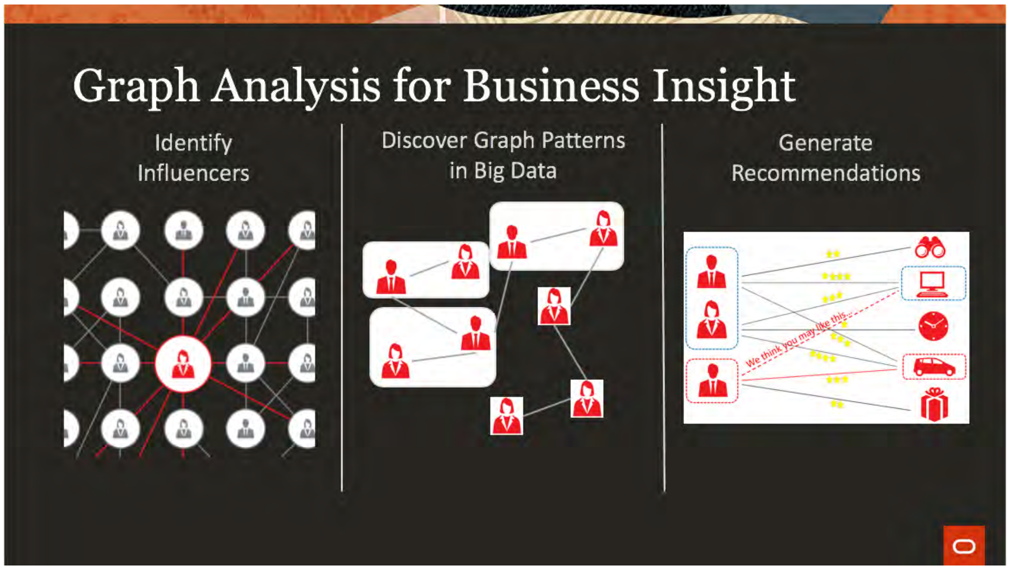
A collection of points (vertices) and lines between those points (edges)

Property graph data model

To accelerate the performance of graph analysis algorithms and graph queries (traversals), the property has an in-memory engine (PGX). To analyze the data in graphs, data scientists and developers use algorithms to detect components and communities, rank the importance of elements in the graph, find patterns, evaluate connections, and explore other relationships. With Oracle Database, in additional to letting you write your own algorithms, you have a wide range of pre-built, high performance algorithms that you can invoke with less than a line of code.

Graph analytics

Graph analytics can be performed using the in-memory analyst (PGX) with over 50 built-in, powerful, parallel, in-memory analytics, including ranking, centrality, recommendation, community detection, and path finding. Several of the most commonly used property graph analytics can also be executed in-database using SQL. SQL-based analysis can be helpful for large graphs, reducing network traffic and obtaining more up-to-date results.



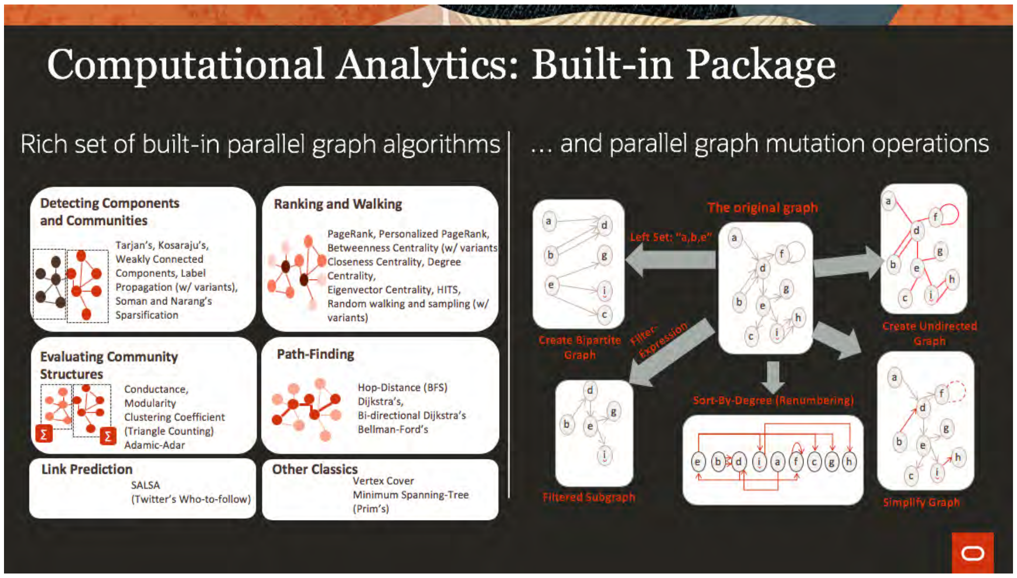
Graph Analysis for Business Insight

Identify Influencers
Discover Graph Patterns in Big Data
Generate Recommendations
We think you may like this

Example property graph use cases

The in-memory analyst (PGX) takes advantage of modern server architecture that parallelizes computation using multiple cores and sizeable memory configurations that enable fast non-sequential data access across a larger portion of a graph read into memory. A parallelized filter query on the database reads a subgraph of interest into memory.

These analytics can either be executed within a Java application or executed in the multi-user, multi-graph in-memory analyst server environment on Oracle WebLogic Server or Apache Tomcat. The output of graph analysis can be another graph, such as a bipartite, filtered, undirected, sorted or simplified edges graph.



Computational Analytics: Built-in Package

Rich set of built-in parallel graph algorithms

Detecting Components and Communities
Tarjan's, Kosaraju's, Weakly Connected Components, Label Propagation (w/ variants), Soman and Narang's Sparsification

Ranking and Walking
PageRank, Personalized PageRank, Betweenness Centrality (w/ variants Closeness Centrality, Degree Centrality, Eigenvector Centrality, HITS, Random walking and sampling (w/ variants)

Evaluating Community Structures
Conductance, Modularity Clustering Coefficient (Triangle Counting) Adamic-Adar

Path-Finding
Hop-Distance (BFS) Dijkstra's, Bi-directional Dijkstra's Bellman-Ford's

Link Prediction
SALSA (Twitter's Who-to-follow)

Other Classics
Vertex Cover Minimum Spanning-Tree (Prim's)

… and parallel graph mutation operations

Categories of analytics and types of output graphs

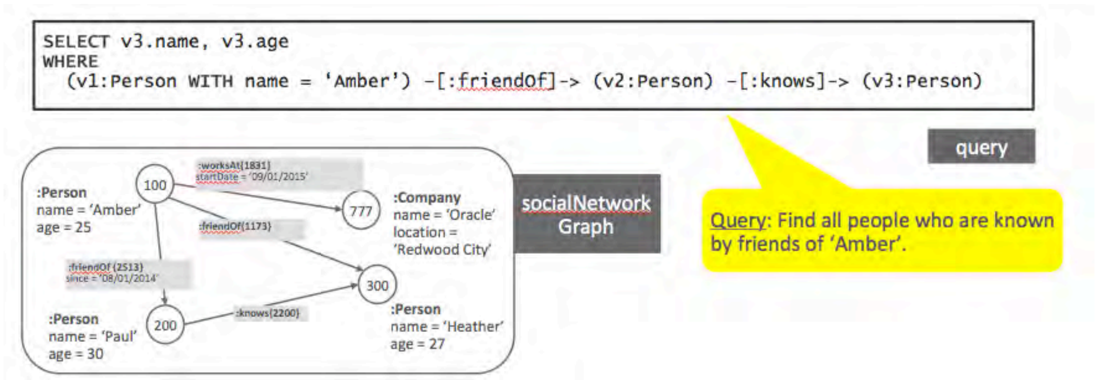
Fast and easy to use querying

Querying using the PGQL query language

A property graph can be queried using PGQL, a declarative SQL-like graph query language. PGQL is designed to be compatible with the proposed SQL/PGQ standards.

A PGQL query describes a graph pattern with vertices, edges, properties, and their relationships. When the query is evaluated against a property graph, the query engine finds all subgraph instances of the graph that match the specified query pattern. Then the query engine returns the selected data entities from each of the matched subgraph instances.

The following example finds all instances of a given pattern or template in the data graph.



```
SELECT v3.name, v3.age
WHERE
  (v1:Person WITH name = 'Amber') -[:friendOf]-> (v2:Person) -[:knows]-> (v3:Person)
```

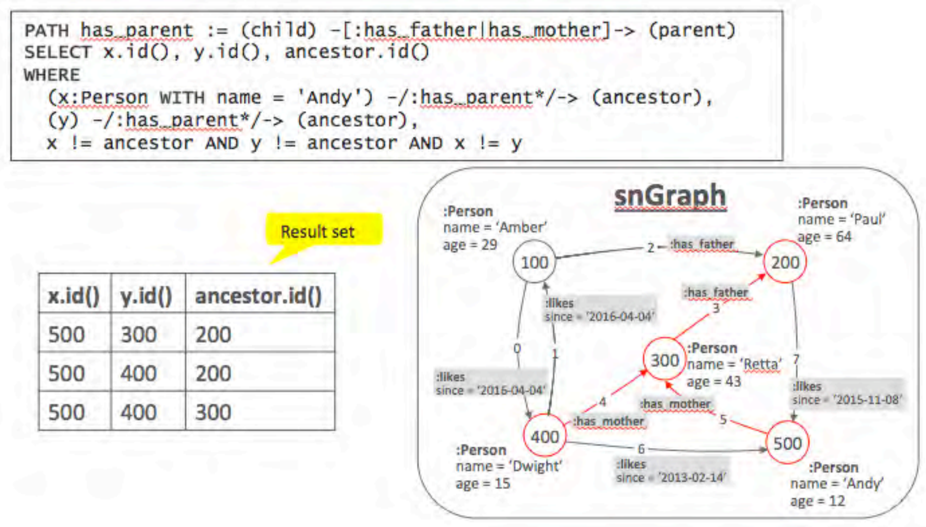
query
Query: Find all people who are known by friends of 'Amber'.
socialNetwork Graph
:Person name = 'Amber' age = 25
100
:worksAt{1831} startDate = '09/01/2015'
777
:Company name = 'Oracle' location = 'Redwood City'
:friendOf{1173}
:friendOf{2513} since = '08/01/2014'
300
:Person name = 'Heather' age = 27
200
:knows{2200}
:Person name = 'Paul' age = 30

PGQL query

PGQL includes support for grouping (GROUP BY), aggregation (e.g. MIN, MAX, AVG), sorting (ORDER BY) and many other familiar SQL constructs.

PGQL also supports regular path queries (recursion) for applications such as reachability analysis. A PGQL regular path query is simpler and more concise than a comparable SQL query would be.

The following example matches a pattern repeatedly by defining a PATH pattern at the top of a query, referring to it in the WHERE clause, and using the star symbol (*) for repeated matching.



```
PATH has_parent := (child) -[:has_father|has_mother]-> (parent)
SELECT x.id(), y.id(), ancestor.id()
WHERE
  (x:Person WITH name = 'Andy') -/:has_parent*/-> (ancestor),
  (y) -/:has_parent*/-> (ancestor),
  x != ancestor AND y != ancestor AND x != y
```

Result set

x.id()	y.id()	ancestor.id()
500	300	200
500	400	200
500	400	300

A regular path query

Property Graphs can be queried programmatically by running PGQL queries from Java applications.

Ease of development and management

Ease of development and management is facilitated through a JShell-based console and a set of Java APIs. The built-in UI shell provides graph database access and in-memory analyst operations. With this command-line shell interface, you can explore the feature's Java APIs to create and drop property graphs, run PGQL queries on the graph, and execute analytics algorithms. You can also add and remove vertices and edges, search for vertices and edges using text search, and perform other manipulations. The JShell allows developers to test Java code snippets more easily without defining objects or compilation.

Notebook interpreters for Apache Zeppelin provide an alternative interface to run the Java APIs, PGQL queries, and analytics algorithms.

Fast search with Oracle Text Indexing

Fast search is enabled through Native Oracle Text indexing. Oracle Text uses standard SQL to index, search, and analyze text values stored in the property columns of the vertices and edges tables. Text queries on Graph are automatically translated into SQL SELECT statements with a "contains" clause.

Oracle Text indexing and search

Automatic text indexing using Oracle Text is supported. Oracle Text uses standard SQL to index, search, and analyze text values stored in the property columns of the vertices and edges tables. Oracle Text indexes all the existing K/V pairs in the property graph.

Fast, parallel bulk loading

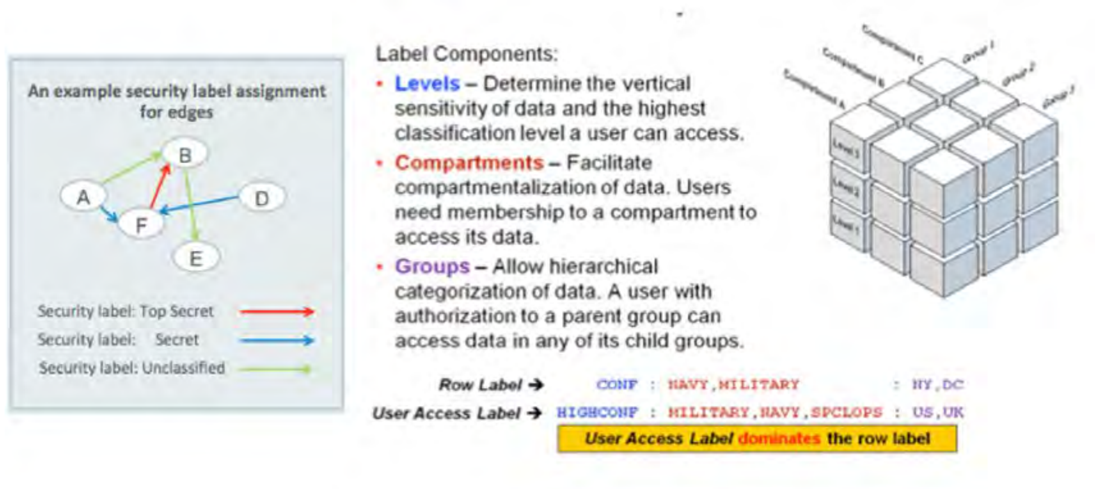
Fast, parallel bulk loading of very large graphs is accomplished with an easy to use, data type-rich Oracle flat file format and Oracle SQL*Loader. The data can be loaded into multiple database partitions. A utility is provided to easily convert Oracle tables and comma separated values (CSV) files into flat file format. The open source graph file formats GraphSON, GraphML and GML are also supported.

Spatial filtering to enhance graph analysis

Spatial filtering in graph queries can enhance graph analysis. A spatial geometry, such as coordinates for an address can be stored as a property and analyzed; for example, a “within distance” query can determine whether to consider the associated entity in further analysis. Support for point, line and polygon geometries and function-based spatial indexing, and access to spatial analytic functions make this a powerful feature.

Multi-level security

Multi-level security can be enforced with graph level access control as well as optional use of the Oracle Label Security option for fine-grained access control to individual vertices and edges



Label Components:

- **Levels** – Determine the vertical sensitivity of data and the highest classification level a user can access.
- **Compartments** – Facilitate compartmentalization of data. Users need membership to a compartment to access its data.
- **Groups** – Allow hierarchical categorization of data. A user with authorization to a parent group can access data in any of its child groups.

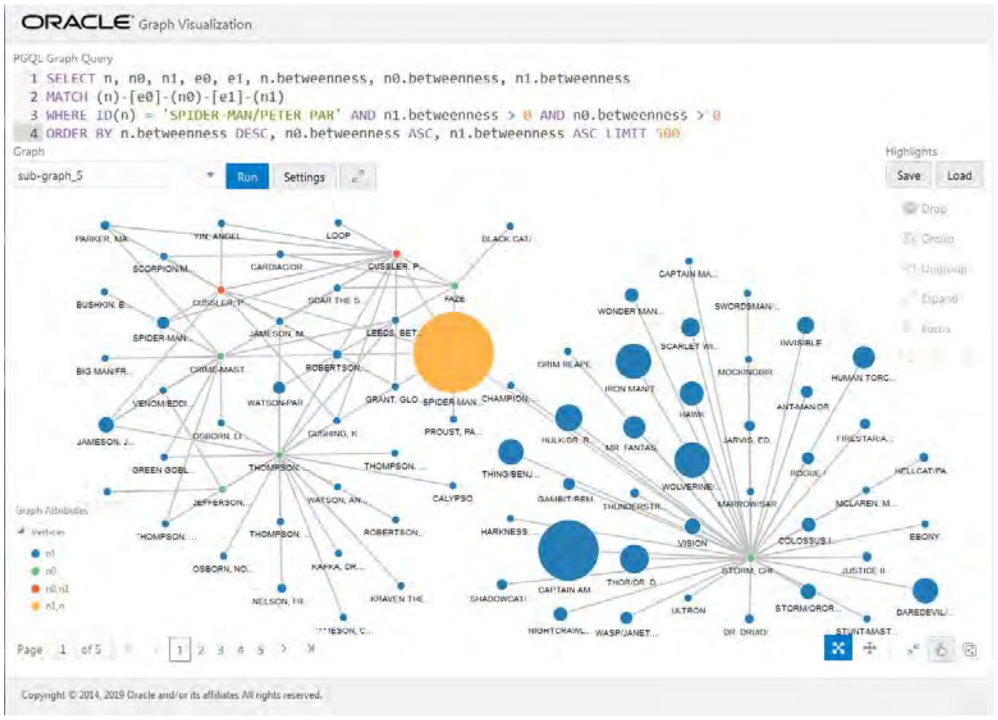
```
        Row Label →     CONF : NAVY,MILITARY          : NY,DC
User Access Label → HIGHCONF : MILITARY,NAVY,SPCLOPS : US,UK
```

User Access Label dominates the row label

Applying Oracle Label Security to graph elements

Graph visualization

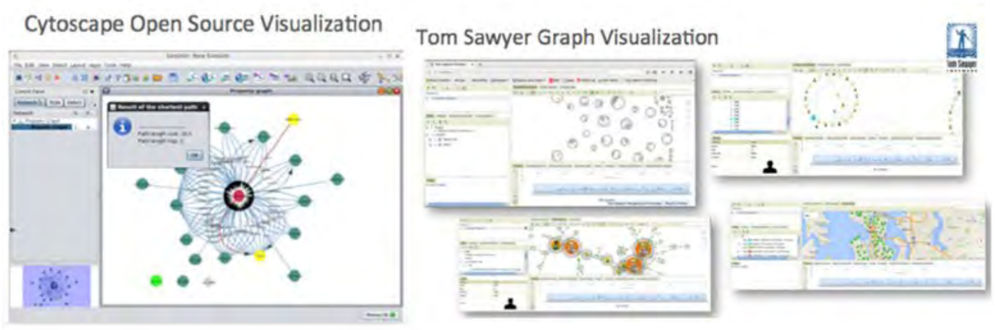
The property graph feature includes a graph visualization tool, where you can enter PGQL queries and visualize the results. A rich set of visualization options enable you to interactively explore the graph, customize layouts, and highlight interesting relationships in your data..



```
SELECT n, n0, n1, e0, e1, n.betweenness, n0.betweenness, n1.betweenness
MATCH (n)-[e0]-(n0)-[e1]-(n1)
WHERE ID(n) = 'SPIDER-MAN/PETER PAR' AND n1.betweenness > 0 AND n0.betweenness > 0
ORDER BY n.betweenness DESC, n0.betweenness ASC, n1.betweenness ASC LIMIT 500
```

Oracle Graph visualization

The property graph feature also supports open source and commercial graph visualization through Cytoscape and Tom Sawyer Perspectives, respectively.


Cytoscape Open Source Visualization
Tom Sawyer Graph Visualization

Graph visualization using Cytoscape and Tom Sawyer Perspectives

RDF GRAPH FEATURES OVERVIEW

The RDF Graph feature is a special purpose graph for linked data and knowledge graph applications conforming to World Wide Web Consortium (W3C) standards. RDF and OWL are standards for representing and defining complex, semantically related data, and SPARQL is a standard protocol and query language designed specifically for linked data. Application developers benefit from the industry's leading open, scalable graph data platform integrated with Oracle Database for scalability, security, performance and high availability.

Storing, Loading, and Data Manipulation

RDF Graph has proven scalability to over one trillion triples (LUBM benchmark). It supports all standard database loading, storing, and data manipulation operations on RDF/OWL models. Each RDF model contains a set of subject – predicate – object – relationship triples organized as a graph of directed, labeled edges. The edge is the link (or relationship) that connects a subject node to an object node and is labeled by a predicate (property). Built-in compression capabilities of Oracle Database provide space-efficient storage for scalable and performant loading, querying, and inferencing.

RDF Graph provides several APIs for data loading and manipulation. A fast bulk loading API is available through both PL/SQL and Java, and SPARQL Update statements can be executed through PL/SQL, Java and HTTP. In addition, plain SQL DML statements can be used to update RDF graphs.

Native Inferencing

Because the graph model for RDF is well-defined and has a formal semantics, it is possible to use logical reasoning to infer implicit or hidden relationships from the existing relationships stated in the data (asserted data). This capability is powerful feature that is unique to RDF graph models. RDF Graph has native, forward-chaining reasoner that supports rules specified using any combination of RDF, RDFS, and OWL 2 RL and EL profiles, as well as user-defined rules for specialized reasoning capabilities. Optimizations include specialized handling of large owl:sameAs sets, incremental inference to update entailments after triple inserts, and parallel inference on multi-core or multi-CPU architectures.

RDF Graph also supports ladder-based inferencing for RDF graphs that use Oracle Label Security so that newly inferred triples are assigned the proper security label.

Querying RDF Graphs in Oracle Database

There are several ways to query RDF graphs in Oracle Database. RDF Graph provides a W3C-standard SPARQL 1.1 endpoint through a tight integration with Apache Jena and Apache Fuseki. Oracle's adapter for Apache Jena also provides a comprehensive Java API for executing SPARQL queries and other operations. RDF graphs can also be queried using SQL; the Oracle SQL SEM_MATCH table function embeds SPARQL graph pattern queries in a SQL query. This is a powerful feature that allows joins of SPARQL queries with other non-RDF data such as relational, JSON, spatial, and XML. In addition, SEM_MATCH allows you to create relational tables and views from the result of SPARQL queries. A virtual model capability provides a view-like feature to combine RDF graphs for querying.

RDF Graph leverages high-performance features of Oracle Database for efficient query execution. SPARQL queries can be executed in parallel with Oracle's parallel SQL engine, and RDF graphs can be loaded into memory with Oracle Database In-Memory. Hardware-based performance features such as Exadata Smart Scan are also fully supported for SPARQL queries.

Viewing Relational Data as an RDF Graph

RDF views can be created on relational tables, views and SQL query results in Oracle Database. These allow you to perform SPARQL queries and other RDF operations on existing relational data. By presenting relation data in RDF triple format, you can connect it with other linked data and RDF graphs to relate and facilitate enterprise data integration. The W3C specifications for automatic mapping (called Direct Mapping), custom mapping (using the W3C R2RML language) are both supported. This virtual RDF data can also be materialized as a natively stored RDF graph, which provides powerful ETL capabilities for generating RDF graphs. Oracle Database features such as Database Links, external tables and Big Data Connectors allow RDF views to be created over other sources.

Support for Spatial Data

RDF Graph supports the Open Geospatial Consortium (OGC) GeoSPARQL standard for representing and querying spatial data in RDF. It leverages the extensive capabilities of Oracle Spatial features for a high-performance GeoSPARQL implementation. GML-based (ogc:gmlLiteral) and WKT-based (ogc:wktLiteral) serializations of spatial geometries are fully supported, and spatial indexes can be created for efficient query execution. RDF Graph understands points, lines, polygons and also more complex geometry collections, and 1000s of different spatial reference systems, including 3-dimensional coordinate systems, can be used.

Full Text Search

RDF Graph has a convenient API for full text indexing. It uses Oracle Text to create a text index on all RDF terms in an RDF graph. This text index can then be used in conjunction with orardf:textContains() and orardf:textScore() SPARQL functions to perform keyword matches over RDF graphs. A simpler orardf:like() SPARQL function is also provided for wild card-based searches over RDF graphs.

Fine-grained Security

Model-level access control is the default for RDF graph data. Triple-level security using the Oracle Label Security is also supported for the most stringent security levels. Sensitivity labels can be defined on individual triples and users to conditionally restrict a user's access to individual triples stored in an RDF model.

Graph Analytics

Oracle Database supports SPARQL 1.1 property path expressions to find graph patterns across any length path. Results from graph queries can be used with Oracle Machine Learning features. In addition, RDF graph data can easily be loaded into Oracle Property Graph analytics (PGX) to run in-memory graph algorithms.

Semantic Indexing for Documents

Semantic indexing for documents provides an index type that lets you search for concepts (such as people, places, organizations, and events) and use ontologies and other RDF features and constructs with unstructured documents, table data and URLs extracted by third party natural language processors and annotators. Oracle RDF Graph creates semantic indexes using RDF names graphs in the RDF database.   Semantically indexed documents can be searched using SEM_CONTAINS operator within a standard SQL query. The search criteria for these documents are expressed using SPARQL query patterns that operate on the information extracted from the documents.

High Availability, Backup and Recovery Features

RDF Graph inherits the built-in high availability, backup and recovery features of Oracle Database. Oracle Real Application Clusters (RAC) can be used for high availability and scalability. Oracle Recovery Manager (RMAN) provides comprehensive backup and recovery functionality. RDF Graph also supports both physical and logical replication for standby databases, and SEM_MATCH supports Flashback Query to execute SPARQL queries against an RDF graph as it existed at a prior time.

ENTERPRISE FEATURES

Oracle Database provides powerful, reliable support for an organization's mission-critical applications. These enterprise features enrich Oracle's Graph capabilities via a flexible Internet deployment architecture, object capabilities, and robust data management utilities that ensure data integrity, data recovery, and data security.

CONCLUSION

Graph analysis is about understanding relationships. Graph analysis can help you understand how banking and financial transactions flow, how people interact, how products and services relate to one another, and what patterns and anomalies may appear in your data. They help to discover fraudulent behavior, business opportunities, and communities of individuals. They can also enable you to link and share related, but disparate data in a semantically consistent way.

With Oracle's graph features, you can use these powerful graph models and analytics as part of a common data platform, integrated with Machine Learning and other analytic and enterprise capabilities of Oracle Database both on premise and in the cloud.

APPENDIX 1: GRAPH FEATURES IN ORACLE DATABASE

Property Graph Features

- Optimized schema for storage of vertices, edges and k/v properties
- PGQL, a SQL-like Graph Query language
- 50+ powerful parallel, in-memory analytics for social network analysis
- Ease of development with Java and JShell UI, Apache Zeppellin Notebook Interpreters, Graph Visualization, SQL queries on graph data
- Analytics execute in Java or in a multi-user, multi-graph in-memory analyst (PGX) server on WebLogic or Tomcat, or in the database using SQL
- Analysis results can flow into a bipartite, filtered, undirected, sorted or simplified edge output graph
- Fast retrieval of vertices and edges using text indexing of properties with Oracle Text
- Property graph analysis of RDF graphs using property graph views
- Parallel bulk load with optimized Oracle flat file format, rich data types and from relational and CSV data
- Open source formats GraphSON, GraphML and GML supported
- Security for graphs and graph elements
- PGQL queries on graph data stored in Oracle Database
- PGQL queries to find in-memory subgraph instances that match a given query pattern
- SQL-based collaborative filtering
- Support for undirected graphs
- An execution and scheduling manager to better control in-memory analyst tasks and resources

RDF Graph Features

- RDF views: present relational data in RDF format for SPARQL queries and connect with other linked data and RDF graphs to relate and facilitate enterprise data integration.
- SPARQL 1.1 in the Database: SPARQL 1.1 Query supported by SEM_MATCH table function and SPARQL 1.1 Update supported by SEM_APIS.UPDATE_MODEL PL/SQL procedure.
- Spatial data: the OGC GeoSPARQL standard is supported for storage and query of Oracle Spatial data in RDF
- Full text indexing: Oracle Text can be used to index and perform keyword-based matching in SPARQL queries.
- Inferencing support: Oracle provides a native, forward-chaining reasoner that supports RDFS, OWL2 RL and EL profiles, and user-defined inferencing.
- Fine-grained security: Oracle Label Security can be used to for triple-level security and ladder-based inferencing ensures inferred triples have the proper security label.

- Graph analytics: SPARQL 1.1 property path expressions find graph patterns across any length path and Property Graph analytics (PGX) can be used to run graph algorithms in memory.
- Graph mining and statistics: Oracle Machine Learning features are supported
- Oracle Database In-Memory: RDF graph data can be loaded into an in-memory columnar representation for fast query execution.
- List-hash composite partitioning: RDF graph data can be partitioned by model and predicate.
- Fast bulk loading of RDF triples and quads: Java and PL/SQL APIs are provided for fast loading of very large RDF graphs.
- Adapter for Apache Jena and Fuseki: an integration with Apache Jena provides a Java API and Fuseki-based REST endpoint for application development.

CONNECT WITH US

Call +1.800.ORACLE1 or visit oracle.com.
Outside North America, find your local office at oracle.com/contact.

blogs.oracle.com   facebook.com/oracle   twitter.com/oracle

Graph Analytics and RDF with Oracle Database
April 2020