ORACLE PROPERTY GRAPHS

Team
Adil Shaik (manager)
Sri Varshini Inakollu
Ruchitha Vishwanath
Purna Sai Kalyan Reddy Bandi
Sahil Varma

Introduction to Property Graphs

- **Data Modeling as a Graph:**
  - Data entities are represented as vertices (nodes) in the graph.
  - Relationships between data entities are represented as edges (connections) in the graph.
  - For example, in a banking context, customer accounts are vertices, and cash transfers between them are edges.

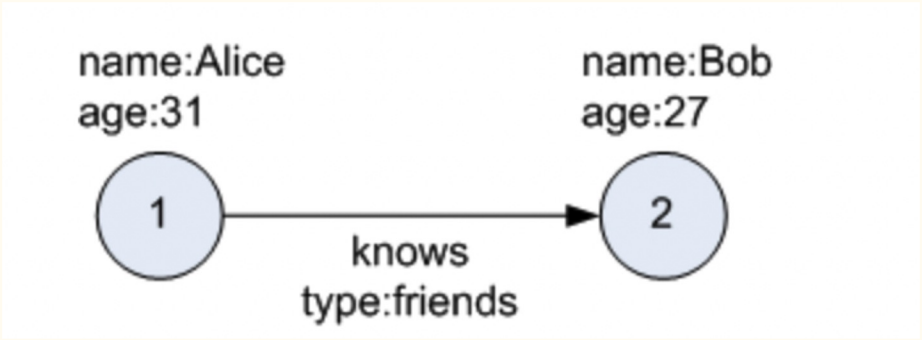
- **Benefits of Viewing Data as a Graph:**
  - Allows for analysis based on the connections and relationships between data entities.
  - Graph analytics algorithms, such as PageRank, can be used to measure the relative importance of data entities based on the relationships (e.g., links between web pages).
  - Graph data modeling can reveal insights such as influencers, communities, and patterns within networks.

What are Property Graphs?

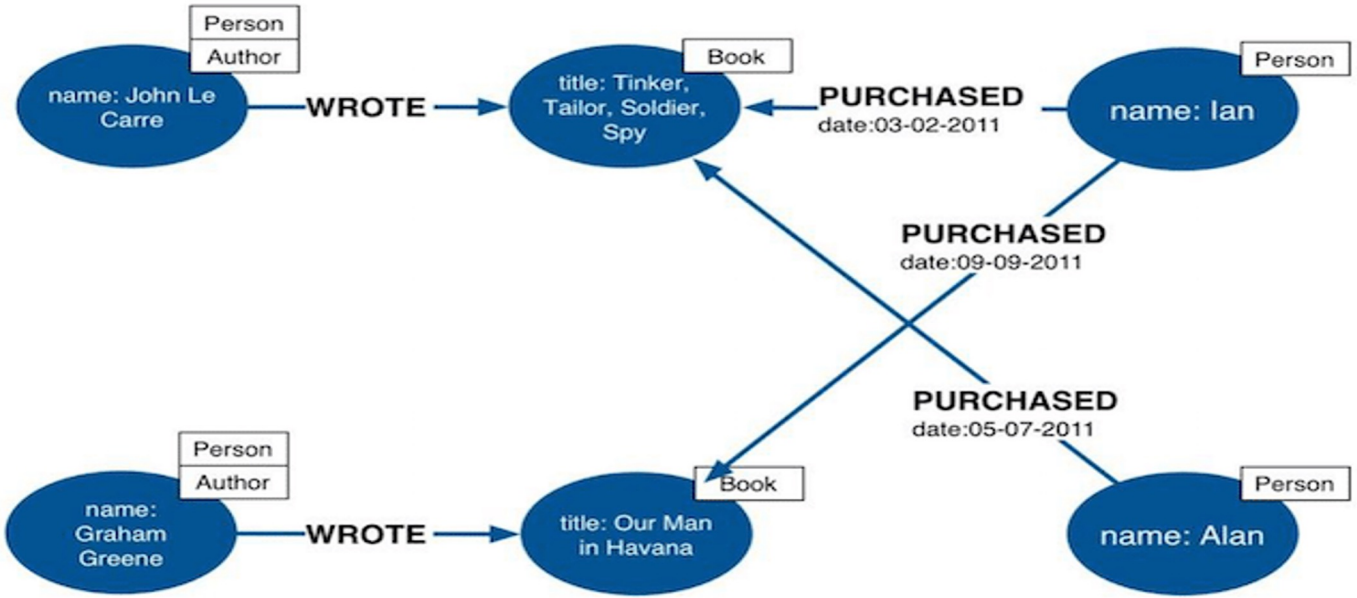
- A property graph consists of a set of objects or **vertices**, and a set of arrows or **edges** connecting the objects.
- **Vertices (Nodes):** These are the data entities, such as people, accounts, or products. Each vertex has a unique identifier and can have multiple properties represented as key-value pairs.
- **Edges (Connections):** These represent the relationships between vertices, such as friendships, transactions, or associations. Each edge connects two vertices, and can also have properties (key-value pairs) describing the nature of the relationship.
- **Modeling Data:** Property graphs provide a flexible and intuitive way to model complex data with interconnected entities and relationships. This allows for advanced analysis of the data based on how entities are linked and interact with each other. For instance, in a social network, property graphs can represent individuals (vertices) and their friendships (edges) along with attributes such as age, location, and interests.

Simple Property Graph Example

- **Vertices:**
  - Two vertices with identifiers 1 and 2.
  - Both vertices have properties:
    - Name
    - Age
- **Edge:**
  - One edge connecting the outgoing vertex 1 to the incoming vertex 2.
  - Edge properties include:
    - Text label: "knows" (describes the relationship).
    - Property type: Identifies the type of relationship between vertices 1 and 2.



Labeled Property Graph Data Model



Property Graph Feature of Oracle Database

The Property Graph feature in Oracle Database allows you to perform advanced graph queries and analytics on your data. It excels at modeling and analyzing relationships within data, making it useful for social networks, cybersecurity, life sciences, and more.

Key Capabilities:

- Scalable graph database
- PGQL and Java graph APIs for developers
- Parallel, in-memory graph server for fast queries and analytics
- Social network analysis functions for ranking, recommendations, and community detection
- Bulk data loading and export
- Graph visualization application
- Jupyter notebook integration

Oracle Property Graphs Applications

- **Graph Operations and Analytics Support:**
  - Includes graph operations, indexing, queries, and search capabilities.
  - Provides in-memory analytics for fast and efficient graph analysis.
- **Graph Data Management:**
  - Manages networks of linked data (vertices, edges, and properties).
  - Models, stores, and analyzes relationships in diverse domains.
- **Graph Analyses Applications:**
  - **Graph Traversal:** Navigate through graph data to find patterns and connections.
  - **Recommendations:** Personalized suggestions based on graph data.
  - **Community Detection:** Identify groups and influencers in networks.
  - **Pattern Matching:** Discover and analyze repeating patterns in graph data.
- **Industries Benefiting from Graphs:**
  - **Telecommunications, Life Sciences, Healthcare:** Analyze relationships and trends in data.
  - **Cybersecurity, Security:** Detect threats and anomalies using graph data.
  - **Media, Publishing:** Understand audience preferences and content engagement.

Overview of Property Graph Architecture

1. Architecture Model for Running Graph Queries in the Database

- **Interaction with Graph Data**:
  i. Use supported client tools to directly interact with graph data stored in relational tables.
  ii. Execute queries within the database environment for optimized performance.

2. Architecture Model for Running Graph Analytics

- **Graph Server Integration**:
  i. Load property graphs into a graph server like PGX for specialized computations.
  ii. Leverage parallel processing and optimized algorithms for complex analytics tasks.

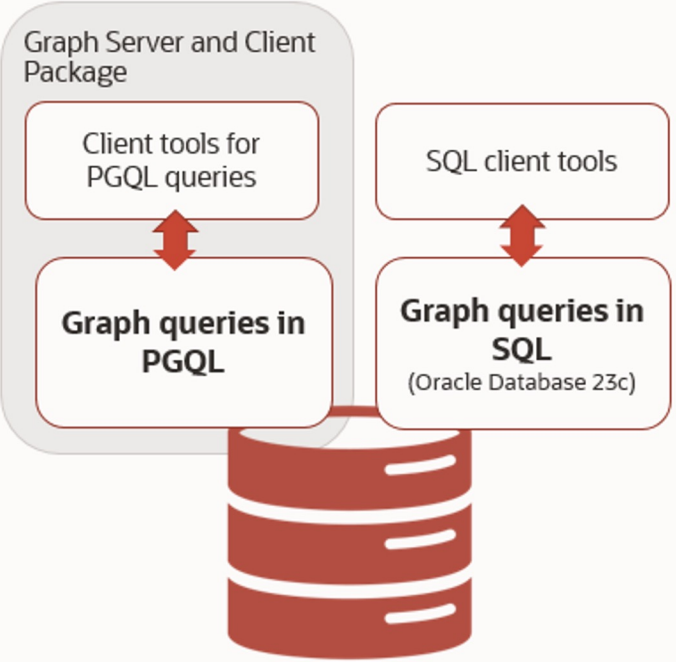
3. Developing Applications Using Graph Server Functionality as a Library

- **Integration of Graph Functions**:
  i. Utilize graph functions provided by a graph server (e.g., PGX) as a library in your applications.
  ii. Seamlessly integrate advanced graph capabilities into your application logic for improved performance and scalability.

1. Architecture Model for Running Graph Queries in the Database

Using any of the supported client tools, you can directly interact with the graph data stored in the relational tables in the database.

This approach runs graph queries, as shown in the following figure.

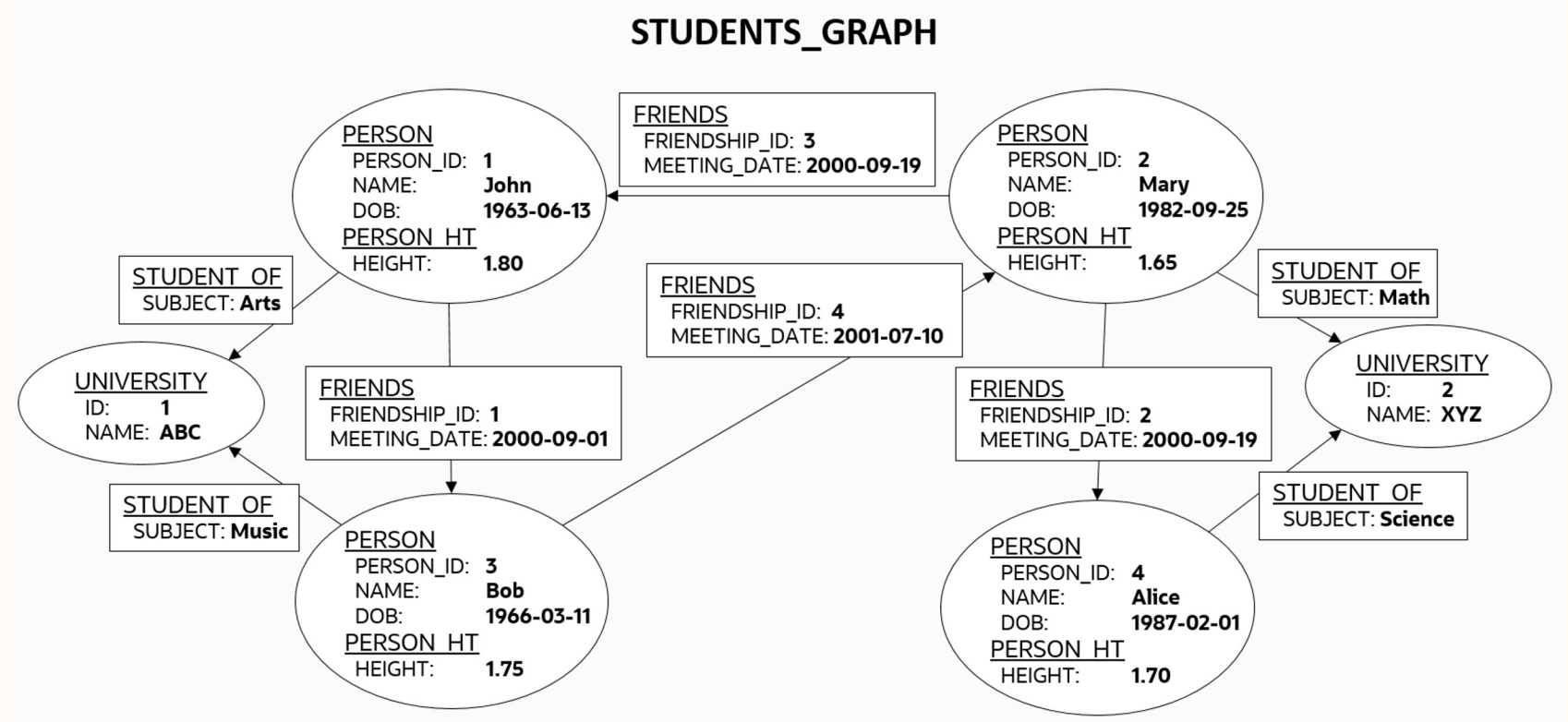


Method 1: SQL Property Graphs

Create a SQL property graph directly over existing database schema objects using **SQL DDL statement.**

- Using the **CREATE PROPERTY GRAPH DDL** statement, you can **create** a property graph object directly in an Oracle Database.
- **Revalidating** a SQL Property Graph using the **ALTER PROPERTY GRAPH COMPILE DDL** statement, you can revalidate an existing property graph object in the database.
- **Dropping** a SQL Property Graph using the **DROP PROPERTY GRAPH DDL** statement, you can remove a property graph object in Oracle Database.
- **JSON Support in SQL Property Graphs**: when creating a SQL property graph, you can define a label property over a JSON data type column using simplified dot notation. You can later access this property inside the SQL graph query.

Method 1 Example: *Creating a SQL Property Graph Using the CREATE PROPERTY GRAPH DDL Statement*



Corresponding SQL property graph DDL statement

```
CREATE PROPERTY GRAPH students_graph
  VERTEX TABLES (
    persons KEY (person_id)
      LABEL person
        PROPERTIES (person_id, name, birthdate AS dob)
      LABEL person_ht
        PROPERTIES (height),
    university KEY (id)
  )
  EDGE TABLES (
    friends
      KEY (friendship_id)
      SOURCE KEY (person_a) REFERENCES persons(person_id)
      DESTINATION KEY (person_b) REFERENCES persons(person_id)
      PROPERTIES (friendship_id, meeting_date),
    student_of
      SOURCE KEY (s_person_id) REFERENCES persons(person_id)
      DESTINATION KEY (s_univ_id) REFERENCES university(id)
      PROPERTIES (subject)
  );
```

Method 2: PGQL Property Graphs

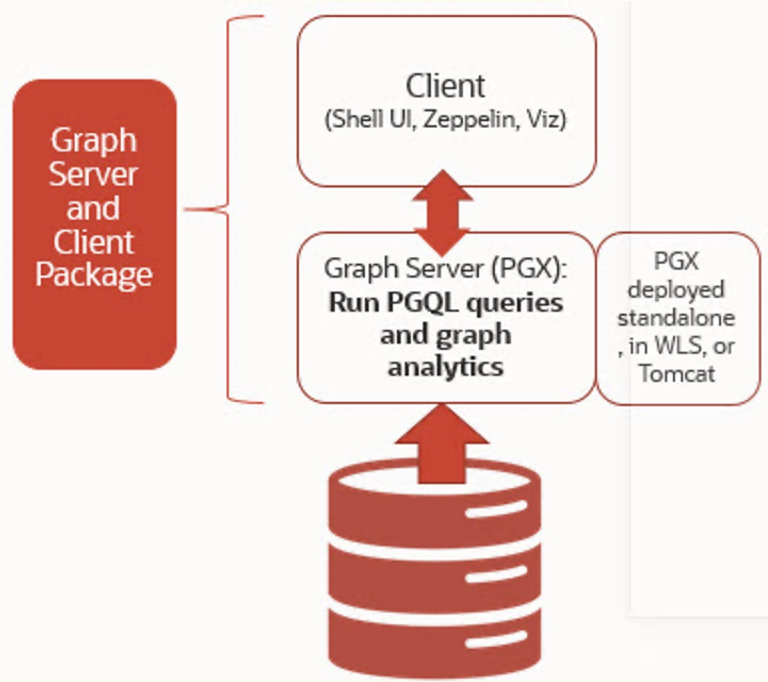
Create a PGQL property graph directly over the graph data in the tables.

The property graph support provides two ways to execute Property Graph Query Language (PGQL) queries through Java APIs:

- Use the **oracle.pgx.api** Java package to query an in-memory snapshot of a graph that has been loaded into the graph server (PGX)
- Use the **oracle.pg.rdbms.pgql** Java package to directly query graph data stored in Oracle Database.

2. Architecture Model for Running Graph Analytics

You can load your property graph into the graph server (PGX) in order to perform specialized graph computations.



Graph Server (PGX):

- PGX functions as a mid-tier server.
- Can run standalone or within containers like Oracle WebLogic Server or Apache Tomcat.

Graph Loading:

- Load property graphs into PGX.
- Create graphs directly from relational tables.
- Load PGQL property graphs or SQL property graphs from the database.

In-Memory Operations:

- Perform graph queries and analytics operations entirely in memory.
- Modify graphs in memory:
- Insert, update, and delete vertices and edges.
- Create new properties for algorithm results.

Non-Persistent Modifications:

- Graph server does not write modifications back to relational tables.
- Ensures data integrity in the underlying database.

Developing Applications using Graph Server as a Library

- **Graph Server as a Library:**
  - The graph server (PGX) can be used as a library in your application to access graph functions directly.
- **Installation Location:**
  - After installing the graph server using RPM, the necessary jar files are located in **/opt/oracle/graph/lib**.
- **Local Server and Client:**
  - In scenarios where the server installation and the client application are on the same machine, development and testing can be streamlined.
- **Interactive Shells:**
  - Development and testing can be done using interactive Java or Python shells in embedded (local) mode.
  - A local PGX instance runs in the same JVM as the client application.
- **Automatic Embedded Mode:**
  - Starting the shell without any parameters launches a local PGX instance in embedded mode, simplifying the development process.

Learn About the Graph Server (PGX)

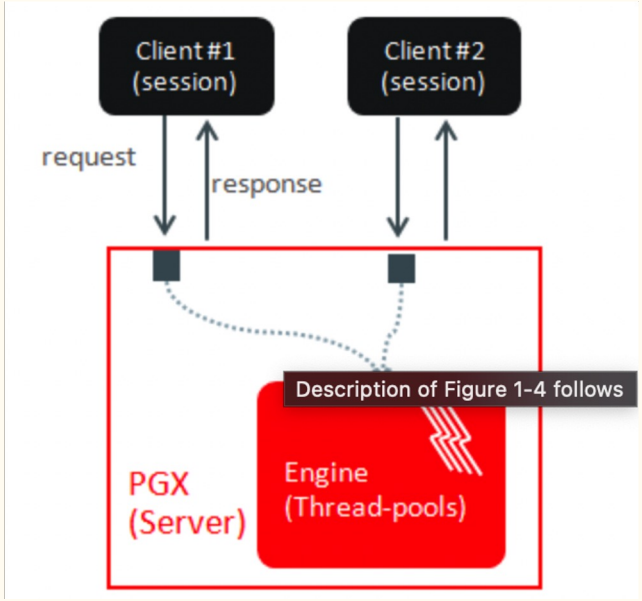
- PGX is a toolkit for graph analysis, supporting both efficient graph algorithms and fast SQL-like graph pattern matching queries.
- Offers over 60 analytic functions across categories such as centrality, component and community, path finding, and community evaluation.
- Functions include Degree Centrality, Eigenvector Centrality, PageRank, Betweenness Centrality, and Closeness Centrality.
- Functions such as Single Source All Destination (Bellman-Ford), Dijkstra's shortest path, and Hop Distance (Breadth-First Search).
- Community detection functions include Strongly Connected Components, Label Propagation, and Twitter's Who-To-Follow.
- Community evaluation functions include Coefficient (Triangle Counting), Conductance, Modularity, and Adamic-Adar counter.

Overview of Graph Server

The graph server can be deployed standalone (it includes an embedded Apache Tomcat instance), or deployed in Oracle WebLogic Server or Apache Tomcat.

- Design of the Graph Server (PGX)
- Usage Modes of the Graph Server (PGX)

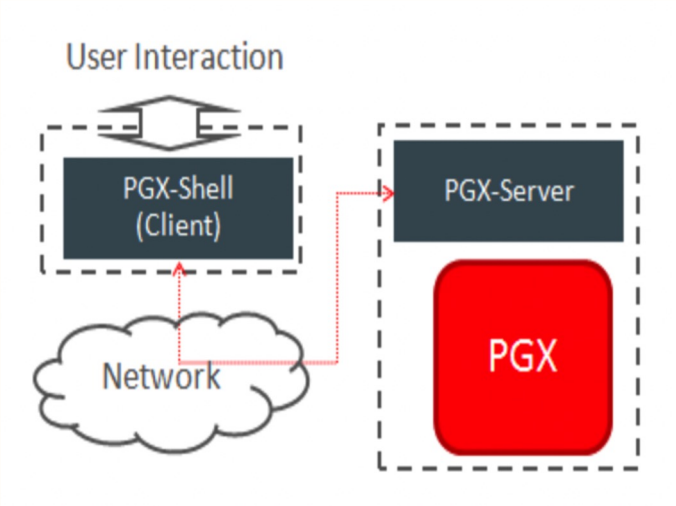
- **Design of the Graph Server(PGX):**
  - Multiple Graph Clients.
  - Asynchronous Client Requests.
  - Internal Engine and Thread Pools.
  - Isolation Between Concurrent Clients



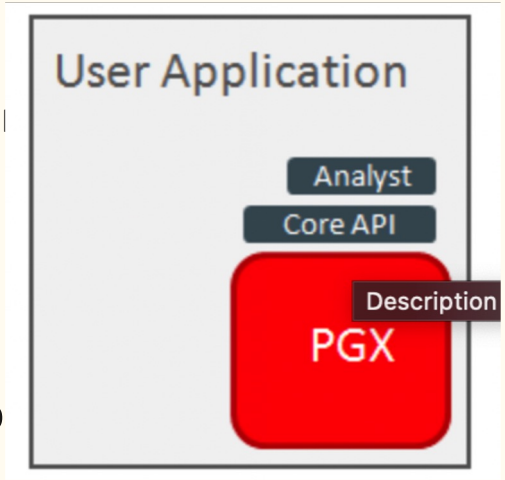
Usage Modes of the Graph Server (PGX)

- Graph server provides two usage modes.
  - **Remote Server Mode**
  - **Using Graph Server(PGX) as a Library**
- **Remote Server Mode:**

  The remote server mode is useful for the following situations where you want to:
  - Perform graph analysis on a large data set with a powerful server-class machine that has many cores and a large memory.
  - The server-class machine is shared by multiple clients



Using Graph Server(PGX) as a Library



You can also include the graph server (PGX) as a normal Java libr

- You can get Direct Access to Graph Functions.
- Local PGX Instance.
- Simplified Development and Testing.

Deploying Graph Server (PGX) as Servlet Web Applicatio

You can deploy the graph server (PGX) as a web application using Apache Tomcat or Oracle WebLogic Server.

- Servlet Web Application Deployment.
- Remote Graph Server Access.
- Scalable and Secure Architecture

Security Best Practices with Graph Data

Sensitive Information:

- Avoid storing unnecessary sensitive information in the graph data.
- Model the graph to include only the relevant subset needed for analysis.
- Ensure vertex and edge identifiers do not reveal sensitive information.
- Deploy the graph product in trusted environments without untrusted client connections.
- Encrypt all communication channels and enable authentication for secure data transfer.

Least Privilege Accounts:

- Use low-privilege, read-only database user accounts for the graph server (PGX) when reading data.
- Utilize separate database user accounts for writing graph data and analyzing it with PGX.
- Implement least privilege principles to minimize access rights and potential security risks.

About Oracle Graph Server and Client Accessibility

Command Line Interface (CLI) Accessibility:

Java and Python command line interfaces support accessibility features such as screen reader compatibility, keyboard navigation, and customizable display settings.

Graph Visualization Application Accessibility:

The Graph Visualization Application, based on Oracle JET, incorporates accessibility features like high contrast modes, keyboard shortcuts, and compatibility with assistive technologies.

About Oracle Graph Server and Client Accessibility

Enabling Accessibility Mode:

Users can enable accessibility mode within Oracle Graph Server and Client interfaces, enhancing navigation, readability, and interaction for users with disabilities.

Training and Support:

Provide access to training materials, guides, and support resources to help users leverage accessibility features effectively.

Standards Compliance:

Mention adherence to accessibility standards such as WCAG (Web Content Accessibility Guidelines) to ensure a consistent and accessible user experience.

Using Oracle Graph with the Autonomous Database

- Oracle Graph with Autonomous Database offers powerful graph analysis capabilities directly within your database environment.
- Key Features:
  - Graph Studio: A fully managed service providing a user-friendly interface for developing graph applications.
  - Autonomous Database Graph Client API: Allows interaction with Graph Studio via client shell CLIs or integration into Java or Python applications.
  - Oracle Graph Server and Client: Offers flexibility to work with any version, compatible with both Serverless and Dedicated deployments.

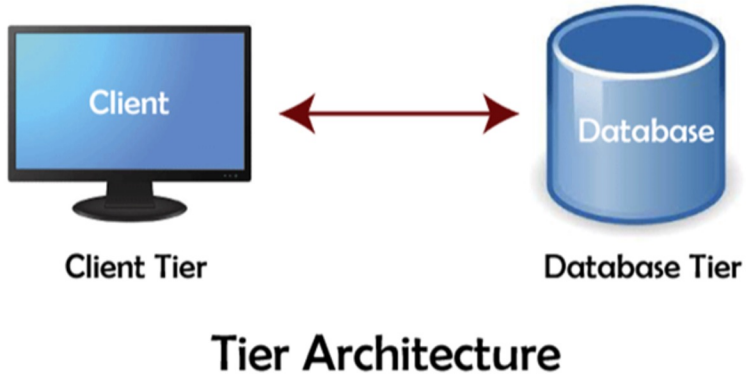
Deployment Options

Two-Tier: Direct connection of client graph application to Autonomous Database.

Three-Tier: Client graph application connects to PGX (Oracle Parallel Graph AnalytiX (PGX) Server) in middle tier, which then connects to the Autonomous Database.

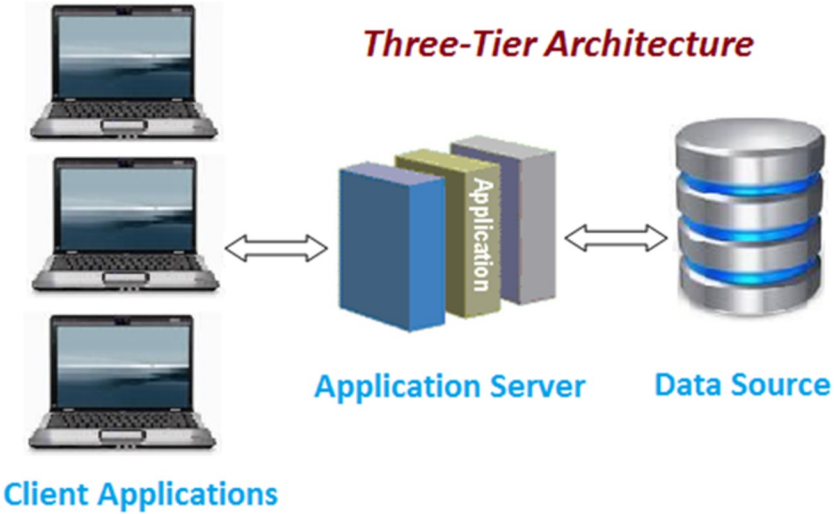
2-Tier Deployment

- Simple architecture with direct communication between the client application and the database.
- Suitable for small to medium-scale applications with low to moderate complexity.
- Limited scalability and flexibility compared to 3-tier architectures.
- May result in performance bottlenecks as the application grows.



3-Tier Deployment

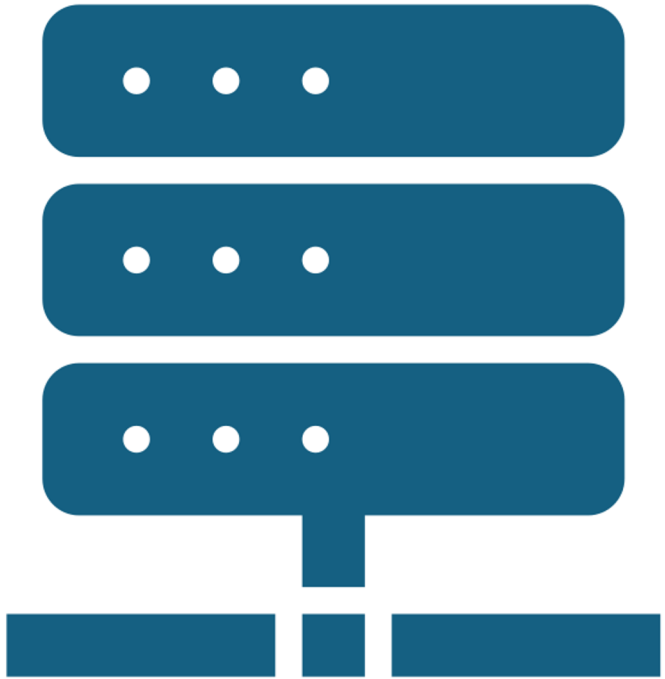
- Adds an additional middle tier between the client and the database, providing a more modular and scalable architecture.
- Enables better separation of concerns, with the middle tier handling business logic, data processing, and communication with the database.
- Offers improved scalability, as additional middle-tier servers can be added to handle increased load.
- Provides better security, as the database is not directly exposed to the client application.



What Tier you should use

The choice between a 2-tier and 3-tier deployment for Oracle Graph Autonomous Database depends on factors such as the complexity of the application, scalability requirements, and security considerations.

While a 2-tier deployment is simpler, a 3-tier deployment offers more flexibility and scalability for larger and more complex applications.



Installation

Before installing the graph server using the RPM file:

- Ensure that you meet the system prerequisites as explained in System Requirements for Installing Oracle Graph Server.
- This link has the detailed steps on how to Install.

Requirement Type	Supported Version
Operating System	• Oracle Linux 7 or 8 x64 • Red Hat Enterprise Linux (RHEL) 7 or 8
JDK version	• Oracle JDK 11 or JDK 17 • OpenJDK JDK 11 or JDK 17 **Note:** Due to a bug in Oracle JDK and OpenJDK, it is recommended to avoid the following JDK versions: • JDK 11.0.9 • JDK 11.0.10 • JDK 11.0.11 • JDK 11.0.12 See this note for more details.