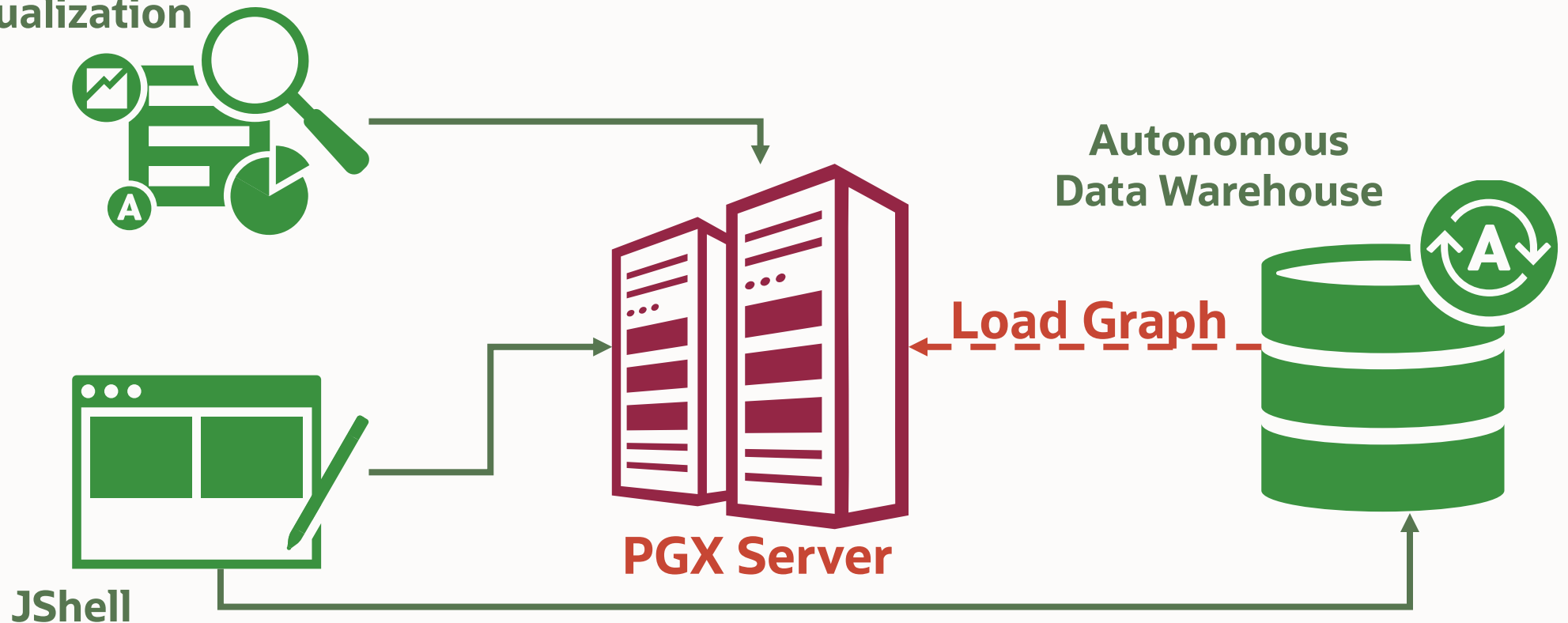




ORACLE

Graph Lab Architecture

Graph Visualization



Autonomous Data Warehouse Provision

Autonomous Data Warehouse Tutorial

Tutorial

- <https://docs.oracle.com/en/cloud/paas/autonomous-data-warehouse-cloud/tutorials.html>

Workshop 1: Autonomous Database Quickstart

- <https://oracle.github.io/learning-library/data-management-library/autonomous-database/shared/adb-quickstart-workshop/freetier/>

Connect Securely Using SQL Developer with a Connection Wallet


- <https://oracle.github.io/learning-library/data-management-library/autonomous-database/shared/adb-advanced-workshop/freetier/?lab=lab-6-using-wallets-for-secure>

Oracle Graph Server Provision









Generating an SSH Key Pair for Oracle Compute Cloud Service Instances

- https://www.oracle.com/webfolder/technetwork/tutorials/obe/cloud/compute-iaas/generating_ssh_key/generate_ssh_key.html

**ORACLE** Cloud


Search for resources, services, and documentation

Japan East (Tokyo) ▾



Service Connector Hub

Data Science >

 Solutions and Platform

Analytics >

Resource Manager >

Email Delivery >

Application Integration >

Monitoring >

Logging >

Developer Services >


Blockchain Platform

Marketplace >

VMware Solution

Applications


Deployed Applications



AUTONOMOUS TRANSACTION PROCESSING

Create an ATP database


3-5 mins



NETWORKING

Set up a network with a wizard


2-3 mins



RESOURCE MANAGER

Create a stack


2-6 mins




All systems operational

View health dashboard

Account Center

 User Management

Add a user to your tenancy

 Billing

Analyze costs

Manage payment method

What's New

Performance Hub ADDM Tab

Oct 20, 2020

Exadata Cloud Service: the flexible X8M shape now available

Oct 16, 2020

Marketplace

All Applications

Deployed Applications

Filters

[clear](#)

TYPE

Any

PUBLISHER

Any

CATEGORY

Any

Oracle Spatial and Graph

All Applications



Oracle Graph Server and Client

Discover new insights into your existing data by leveraging Oracl...

Type: Stack | Price: BYOL



Oracle RDF Graph Server and

Web application for managing and querying RDF data stores

Type: Stack | Price: BYOL



Oracle Graph Server and Client - Image

Discover new insights into your existing data by leveraging Oracl...

Type: Image | Price: BYOL

Marketplace » Oracle Graph Server and Client



Oracle Graph Server and Client

Discover new insights into your existing data by leveraging Oracle's powerful graph technologies

This stack provides a pre-installation of the Oracle Graph Server and Client using Oracle JDK on an Oracle Linux 7 base image.

Categories: Analytics, Packaged Application, Big Data

Type Stack

Version 20.4.0 - default

Compartment MichaelCompartment

hktwiab (root)/TWSELAB01/Michael Compartment

Software Price per OCPU

BYOL

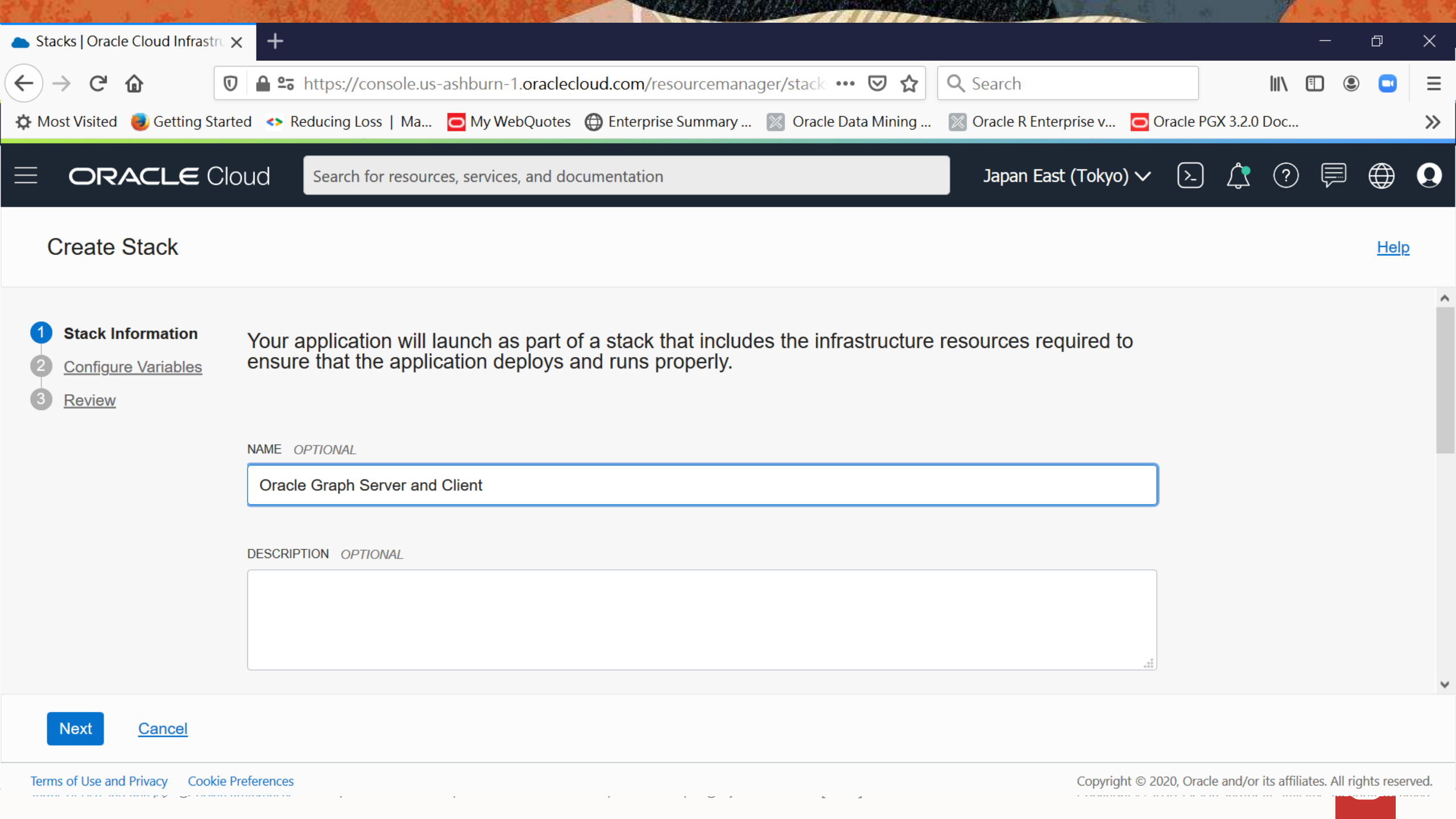
(Bring Your Own License)

There are additional fees for the infrastructure usage. ⓘ

☒ I have reviewed and accept the [Oracle Standard Terms and Restrictions](#).

Launch Stack

- Overview
- Provider
- More Apps
- Usage Instructions



Create Stack

[Help](#)

- 1 **Stack Information**
- 2 [Configure Variables](#)
- 3 [Review](#)

Your application will launch as part of a stack that includes the infrastructure resources required to ensure that the application deploys and runs properly.

NAME OPTIONAL

Oracle Graph Server and Client

DESCRIPTION OPTIONAL

Next [Cancel](#)

Create Stack

[Help](#)

- ✓ [Stack Information](#)
- 2 **Configure Variables**
- 3 [Review](#)

Configure the variables for the infrastructure resources that this stack will create when you run the apply job for this execution plan.

Oracle Graph Server Compute Instance

RESOURCE NAME PREFIX

The name of all created compute and network resources will begin with this prefix

ORACLE GRAPH SERVER COMPARTMENT

The compartment in which to create the Oracle Graph Server compute instance

[Back](#) [Next](#) [Cancel](#)

Create Stack

[Help](#)

- ✓ [Stack Information](#)
- 2 **Configure Variables**
- 3 [Review](#)

MichaelCompartment

The compartment in which to create the Oracle Graph Server compute instance

ORACLE GRAPH SERVER AVAILABILITY DOMAIN

Select an option

The name of the availability domain in which to create the Oracle Graph Server compute instances

ORACLE GRAPH SERVER SHAPE

Select an option

The shape for the Oracle Graph Server compute instance

SSH PUBLIC KEY

Use the corresponding public key to access the Oracle Graph Server compute instance

Stacks | Oracle Cloud Infrastru

+

← → ↺ 🏠

🔒 https://console.us-ashburn-1.oraclecloud.com/resourcemanager/stack ...

🔍 Search

☰ 📄 👤 🗣️

⚙️ Most Visited 🌐 Getting Started 🌐 Reducing Loss | Ma... 📺 My WebQuotes 🌐 Enterprise Summary ... 🗕 Oracle Data Mining ... 🗕 Oracle R Enterprise v... 📺 Oracle PGX 3.2.0 Doc...

☰ ORACLE Cloud

Create Stack

✓ Stack Information

2 **Configure Variables**

3 Review

VM.Standard.B1.2

VM.Standard.B1.4

VM.Standard.B1.8

VM.Standard.E2.1

VM.Standard.E2.2

VM.Standard.E2.4

VM.Standard.E2.8

VM.Standard2.1

VM.Standard2.16

VM.Standard2.2

VM.Standard2.24

VM.Standard2.4

VM.Standard2.8

Select an option

The shape for the Oracle Graph Server compute instance

SSH PUBLIC KEY

Use the corresponding public key to access the Oracle Graph Server compute instance

Back

Next

Cancel

Stacks | Oracle Cloud Infrastru

+

←

→

↶

🏠

🔒🔗https://console.us-ashburn-1.oraclecloud.com/resourcemanager/stack...🔍🌟

🔍 Search

☰📄👤🗣️☰

☰

⚙️ Most Visited🔗 Getting Started🔗 Reducing Loss | Ma...🔴 My WebQuotes🌐 Enterprise Summary ...🗕 Oracle Data Mining ...🗕 Oracle R Enterprise v...🔴 Oracle PGX 3.2.0 Doc...

☰ORACLE Cloud

Search for resources, services, and documentation

Japan East (Tokyo)📄🔔🔍💬🌐👤

Create Stack

Help

✓

Stack Information

2

Configure Variables

3

Review

Select an option

⌵

The shape for the Oracle Graph Server compute instance

SSH PUBLIC KEY

ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAQEAi7WQ8i7ucZdnpFZKqluDVKgEqWa4SDlgwLiwtmifWHV1yM839d/trvM1

Use the corresponding public key to access the Oracle Graph Server compute instance

Instance Network

VIRTUAL CLOUD NETWORK COMPARTMENT

Choose...

⌵

EXISTING VIRTUAL CLOUD NETWORK

Back

Next

Cancel

Terms of Use and Privacy

Cookie Preferences

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Create Stack

[Help](#)

Instance Network

- ✓ [Stack Information](#)
- 2 Configure Variables**
- 3 [Review](#)

VIRTUAL CLOUD NETWORK COMPARTMENT

MichaelCompartment

hktwlab (root)/TWSELAB01/MichaelCompartment

EXISTING VIRTUAL CLOUD NETWORK

michaelvcn

An existing Virtual Cloud Network (VCN) in which to create the compute instance.

SUBNET COMPARTMENT

MichaelCompartment

hktwlab (root)/TWSELAB01/MichaelCompartment

EXISTING SUBNET ⓘ

Back

Next

[Cancel](#)

Create Stack

[Help](#)

- ✓ [Stack Information](#)
- 2 [Configure Variables](#)
- 3 [Review](#)

Graph Server Configuration

JDBC URL FOR AUTHENTICATION

jdbc:oracle:thin:@ords_tp?TNS_ADMIN=/etc/oracle/graph/wallets

JDBC URL to an Oracle Database that will be used for user authentication. URL must be reachable from the compute instance network.

PGQL ENGINE FOR GRAPHVIZ

PGQL-on-PGX

PGQL Engine that will be used for Graph Visualization webapp

jdbc:oracle:thin:@graphdb_high?TNS_ADMIN=/etc/oracle/graph/wallets

[Back](#) [Next](#) [Cancel](#)

[Help](#)

- Verify your configuration variables, and then create your stack. The apply job will automatically run to create resources specified in the configuration. Due to limited space, we show only variables without default values or that you edited.

Oracle Graph Server and Client

Compartment

...pnvbka [Show](#) [Copy](#)

0.12.x

Cancel

Compute

Instances

- Dedicated Virtual Machine Hosts
- Instance Configurations
- Instance Pools
- Cluster Networks
- Autoscaling Configurations
- Custom Images
- Boot Volumes
- Boot Volume Backups
- OS Management

Instances in MichaelCompartment *Compartment*

The [Compute service](#) helps you provision VMs and bare metal instances to meet your compute and application requirements. An [instance](#) is a compute host. Choose between virtual machines (VMs) and bare metal instances. The image that you use to launch an instance determines its operating system and other software.

Create Instance

Name	State	Public IP	Shape	OCPU Count	Memory (GB)	Availability Domain	Fault Domain
pgx	<div></div> Running	158.101.144.184	VM.Standard.E2.8	8	64	AD-1	FD-1
DBSecLab-v3	<div></div> Running	140.238.52.66	VM.Standard.E3.Flex	6	96	AD-1	FD-2

Graph Server Configuration

Oracle Cloud Infrastructure

https://console.us-ashburn-1.oraclecloud.com/db/adb/ocid1.autonomousdatabase.oc1.ap-tokyo-1.abxhiljr6ifb5niyrvqnherg

Search

Most VisitedGetting StartedReducing Loss | Ma...My WebQuotesEnterprise Summary ...Oracle Data Mining ...Oracle R Enterprise v...Oracle PGX 3.2.0 Doc...PChome Cloud1PChome Cloud2Mason978HWTWLabGitHub - APACTestD...

ORACLE Cloud

Search for resources, services, and documentation

Japan East (Tokyo)

Overview » Autonomous Database » Autonomous Database Details

ATP

AVAILABLE

DB Connection

Perform

Autonomous Database

General Inform

Database Name: ords

Workload Type: Transac

Compartment: hktwlab (

OCID: ...ifngxq Show C

Created: Tue, Jul 28, 20

OCPU Count: 1

Storage: 1 TB

License Type: Bring Your Own License (BYOL)

Database Version: 19c

Auto Scaling: Enabled ⓘ

Lifecycle State: Available

Instance Type: Paid

Mode: Read/Write Edit

Database Connection

Help Close

You will need the client credentials and connection information to connect to your database. The client credentials include the wallet, which is required for all types of connections.

Download Client Credentials (Wallet)

To download your client credentials, select the type of wallet, then click **Download Wallet**. You will be asked to create a password for the wallet.

Wallet Type ⓘ

Instance Wallet

Download Wallet

Rotate Wallet

Wallet last rotated: -

Close

Network

Access Type: Allow secure access from everywhere

Access Control List: Disabled Edit

Maintenance ⓘ

Next Maintenance: Sat, Oct 31, 2020, 04:00:00 UTC - 08:00:00 UTC

Data Safe ⓘ

Status: Not Registered Register

Terms of Use and Privacy

Cookie Preferences

Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

Copy ADW wallets

```
$ scp ./Wallet_orcs.zip opc@158.101.144.184:/tmp
$ ssh -i ./oci_key.ppk opc@158.101.144.184
$ sudo su -
# mkdir /etc/oracle/graph/wallets
# cd /etc/oracle/graph
# unzip /tmp/Wallet_graph.zip /etc/oracle/graph/wallets
```

Look at tnsnames.ora under /etc/oracle/graph/wallets

```
# cd /etc/oracle/graph
# vi pgx.conf
```

修改pgx.conf – 權限設定

```
"pgx_role": "GRAPH_DEVELOPER",  
"pgx_permissions": [  
  {  
    "grant": "PGX_SESSION_CREATE"  
  },  
  {  
    "grant": "PGX_SESSION_NEW_GRAPH"  
  },  
  {  
    "grant": "PGX_SESSION_GET_PUBLISHED_GRAPH"  
  },  
  {  
    "grant": "PGX_SESSION_ADD_PUBLISHED_GRAPH"  
  },  
]
```

修改pgx.conf – ADW連線設定

```
...
"pgx_realm": {
  "implementation": "oracle.pg.identity.DatabaseRealm",
  "options": {
    "jdbc_url":
"jdbc:oracle:thin:@graphdb_high?TNS_ADMIN=/etc/oracle/graph/wallets",
    "token_expiration_seconds": 14400,
    "connect_timeout_milliseconds": 10000,
    "max_pool_size": 64,
    "max_num_users": 512
  }
}
...
```

Restart PGX

```
$ sudo systemctl restart pgx
```

Autonomous Data Warehouse

—
User Creation

```
SQL> CREATE USER oraclegraph  
2 IDENTIFIED BY "HKTWLab@orac13.com"  
3 DEFAULT TABLESPACE DATA  
4 QUOTA UNLIMITED ON DATA  
5 TEMPORARY TABLESPACE TEMP;
```

User ORACLEGRAPH created.

```
SQL> GRANT ALTER SESSION,CREATE PROCEDURE,CREATE SESSION,CREATE TABLE, CREATE TYPE to oraclegraph;  
Grant succeeded.
```

```
SQL> CREATE ROLE GRAPH_DEVELOPER;  
Role GRAPH_DEVELOPER created.
```

```
SQL> CREATE ROLE GRAPH_ADMINISTRATOR;  
Role GRAPH_ADMINISTRATOR created.
```

```
SQL> GRANT GRAPH_DEVELOPER to oraclegraph;  
Grant succeeded.
```

```
SQL> GRANT GRAPH_ADMINISTRATOR to oraclegraph;  
Grant succeeded.
```


—
Login SQL Developer

```
CREATE ROLE graph_developer;
```

```
CREATE ROLE graph_administrator;
```

```
create user oraclegraph identified by HKTWLab@orac13.com default tablespace data temporary  
tablespace temp;
```

```
ALTER USER ORACLEGRAPH QUOTA UNLIMITED ON DATA;
```

```
GRANT ALTER SESSION,CREATE PROCEDURE,CREATE SEQUENCE,CREATE SESSION,CREATE TABLE,CREATE  
TRIGGER,CREATE TYPE,CREATE VIEW to ORACLEGRAPH;
```

```
GRANT graph_developer to oraclegraph;
```

```
GRANT select on HR.employees to oraclegraph;
```

```
GRANT select on HR.jobs to oraclegraph;
```

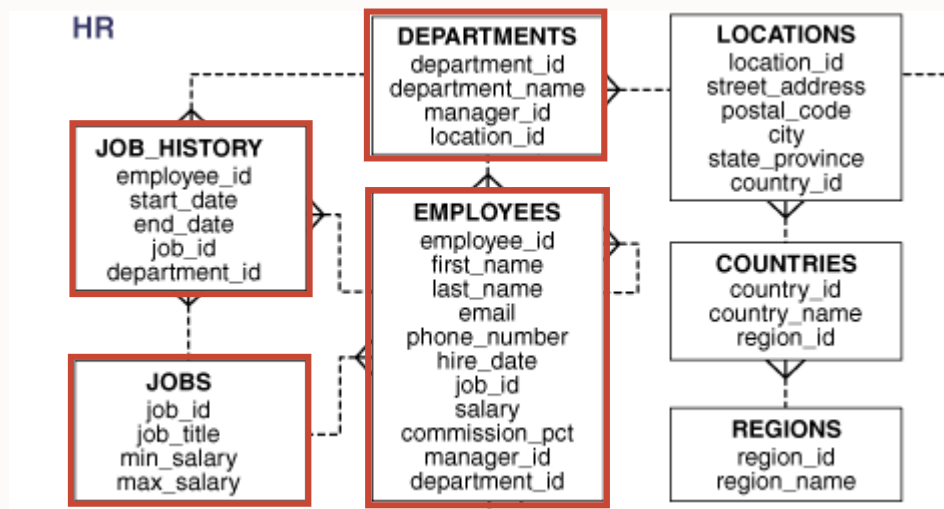
```
GRANT select on HR.departments to oraclegraph;
```

```
GRANT select on HR.job_history to oraclegraph;
```

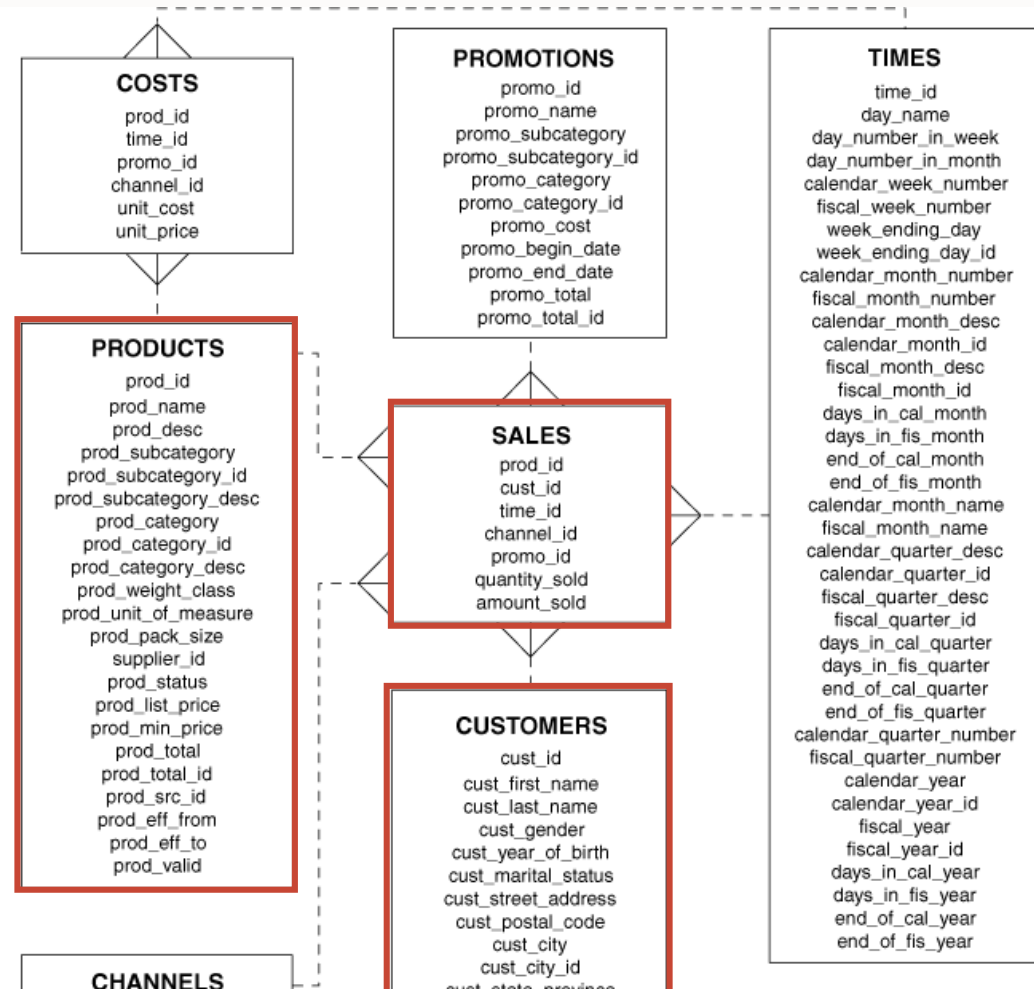
```
GRANT select on HR.countries to oraclegraph;
```

```
GRANT select on HR.regions to oraclegraph;
```

Sample Schema HR, SH



SH



CREATE TABLE SALES

AS

```

select CUST_ID, PROD_ID, SUM(QUANTITY_SOLD) as QUANTITY_SOLD, SUM(AMOUNT_SOLD) AS AMOUNT_SOLD
from sh.sales
group by CUST_ID, PROD_ID;
    
```



Graph Model: HR

Create Graph in PGX from Database Tables

PGQL Create Graph from Database Tables

```
[opc@pgx ~]$ /opt/oracle/graph/bin/opg-jshell --base_url https://localhost:7007 --username oraclegraph
```

```
enter password for user oraclegraph (press Enter for no password):
```

```
For an introduction type: /help intro
```

```
Oracle Graph Server Shell 20.4.0
```

```
Variables instance, session, and analyst ready to use.
```

```
opg-jshell> String statement =
```

```
...> "CREATE PROPERTY GRAPH hr_simplified "+
```

```
...> "  VERTEX TABLES ( "+
```

```
...> "    hr.employees LABEL employee "+
```

```
...> "      PROPERTIES ARE ALL COLUMNS EXCEPT ( job_id, manager_id, department_id ), "+
```

```
...> "    hr.departments LABEL department "+
```

```
...> "      PROPERTIES ( department_id, department_name ) "+
```

```
...> "  ) "+
```

```
...> "  EDGE TABLES ( "+
```

```
...> "    hr.employees AS works_at "+
```

```
...> "      SOURCE KEY ( employee_id ) REFERENCES employees "+
```

```
...> "      DESTINATION departments "+
```

```
...> "      PROPERTIES ( employee_id ) "+
```

```
...> "  );
```

PGQL Query

```
statement ==> "CREATE PROPERTY GRAPH hr_simplified VERTEX TABLES ( hr.employees LABEL employee PROPERTIES ARE ALL
COLUMNS EXCEPT ( job_id, manager_id, department_id ), hr.departments LABEL department PROPERTIES ( department_id,
department_name ) ) EDGE TABLES ( hr.employees AS works_at SOURCE KEY ( employee_id ) REFERENCES employees
DESTINATION departments PROPERTIES ( employee_id ) )"

opg-jshell> session.executePgql(statement);
$6 ==> null

opg-jshell> PgxGraph g = session.getGraph("HR_SIMPLIFIED");
g ==> PgxGraph[name=HR_SIMPLIFIED,N=134,E=106,created=1603952305636]

opg-jshell> String query =
...> "SELECT dep.department_name "+
...> "FROM MATCH (emp:Employee) -[:works_at]-> (dep:Department) "+
...> "WHERE emp.first_name = 'Nandita' AND emp.last_name = 'Sarchand' "+
...> "ORDER BY 1";

query ==> "SELECT dep.department_name FROM MATCH (emp:Employee) -[:works_at]-> (dep:Department) WHERE emp.first_name =
'Nandita' AND emp.last_name = 'Sarchand' ORDER BY 1"
```

PGQL Query

```
opg-jshell> PgqlResultSet resultSet = g.queryPgql(query);
resultSet ==> PgqlResultSetImpl[graph=HR_SIMPLIFIED,numResults=1]
opg-jshell> resultSet.print();
+-----+
| department_name |
+-----+
| Shipping        |
+-----+
$10 ==> PgqlResultSetImpl[graph=HR_SIMPLIFIED,numResults=1]
opg-jshell> String query =
...> "SELECT label(n), COUNT(*) "+
...> "FROM MATCH (n) "+
...> "GROUP BY label(n) "+
...> "ORDER BY COUNT(*) DESC";
query ==> "SELECT label(n), COUNT(*) FROM MATCH (n) GROUP BY label(n) ORDER BY COUNT(*) DESC"
opg-jshell> PgqlResultSet resultSet = g.queryPgql(query);
resultSet ==> PgqlResultSetImpl[graph=HR_SIMPLIFIED,numResults=2]
```


PGQL Query

```
opg-jshell> resultSet.print();
+-----+
| label(n) | COUNT(*) |
+-----+
| EMPLOYEE | 107      |
| DEPARTMENT | 27      |
+-----+
$13 ==> PgqlResultSetImpl[graph=HR_SIMPLIFIED,numResults=2]
opg-jshell> String query =
...> "SELECT label(n) AS srcLbl, label(e) AS edgeLbl, label(m) AS dstLbl, COUNT(*) "+
...> "FROM MATCH (n) -[e]-> (m) "+
...> "GROUP BY srcLbl, edgeLbl, dstLbl "+
...> "ORDER BY COUNT(*) DESC";
query ==> "SELECT label(n) AS srcLbl, label(e) AS edgeLbl, label(m) AS dstLbl, COUNT(*) FROM MATCH (n) -[e]-> (m) GROUP BY
srcLbl, edgeLbl, dstLbl ORDER BY COUNT(*) DESC"
opg-jshell> PgqlResultSet resultSet = g.queryPgql(query);
resultSet ==> PgqlResultSetImpl[graph=HR_SIMPLIFIED,numResults=1]
```

PGQL Query

```
opg-jshell> resultSet.print();

+-----+
| srcLb1 | edgeLb1 | dstLb1      | COUNT(*) |
+-----+
| EMPLOYEE | WORKS_AT | DEPARTMENT | 106      |
+-----+

$16 ==> PgqlResultSetImpl[graph=HR_SIMPLIFIED,numResults=1]

opg-jshell> analyst.pagerank(g)

$17 ==> VertexProperty[name=pagerank,type=double,graph=HR_SIMPLIFIED]

opg-jshell> session.queryPgql("select m.FIRST_NAME, m.LAST_NAME, m.pagerank from HR_SIMPLIFIED match (m:EMPLOYEE) where
m.FIRST_NAME = 'Nandita']").print().close()

+-----+
| m.FIRST_NAME | m.LAST_NAME | m.pagerank      |
+-----+
| Nandita      | Sarchand    | 0.001119402985074627 |
+-----+
```



PGQL Query

```
opg-jshell> session.queryPgql("select m.DEPARTMENT_NAME, m.pagerank from HR_SIMPLIFIED match (m:DEPARTMENT) order by m.pagerank ").print().close();
```

m.DEPARTMENT_NAME	m.pagerank
Government Sales	0.001119402985074627
Retail Sales	0.001119402985074627
Manufacturing	0.001119402985074627
Contracting	0.001119402985074627
Corporate Tax	0.001119402985074627
IT Support	0.001119402985074627
NOC	0.001119402985074627
Control And Credit	0.001119402985074627
Construction	0.001119402985074627
Recruiting	0.001119402985074627
Payroll	0.001119402985074627
Shareholder Services	0.001119402985074627
IT Helpdesk	0.001119402985074627
Benefits	0.001119402985074627
Treasury	0.001119402985074627

PGQL Query

```
opg-jshell> g.publish(VertexProperty.ALL, EdgeProperty.ALL)
```

```
opg-jshell>
```

Graph Visualization – PGQL Graph Query

select e
match ()-[e]-()

Graph HR_SIMPLIFIED

```
SELECT emp, e, dep  
MATCH (emp:EMPLOYEE) -[e:WORKS_AT]->  
(dep:DEPARTMENT)  
WHERE emp.FIRST_NAME = 'Nandita' AND  
emp.LAST_NAME = 'Sarchand'
```

```
SELECT label(n), COUNT(*)  
FROM MATCH (n)  
GROUP BY label(n)  
ORDER BY COUNT(*) DESC
```

```
SELECT label(n) AS srcLbl, label(e) AS edgeLbl,  
label(m) AS dstLbl, COUNT(*)  
FROM MATCH (n) -[e]-> (m)  
GROUP BY srcLbl, edgeLbl, dstLbl  
ORDER BY COUNT(*) DESC
```

```
select m.FIRST_NAME, m.LAST_NAME, m.pagerank  
FROM MATCH (m:EMPLOYEE)  
WHERE m.FIRST_NAME = 'Nandita'
```

```
select m.DEPARTMENT_NAME, m.pagerank  
from HR_SIMPLIFIED  
match (m:DEPARTMENT) order by m.pagerank  
DESC
```

Online Retail Transactions

Graph Analysis for Recommendation

By Ryota Yamanaka

https://github.com/ryotayamanaka/oracle-pg/tree/master/graphs/online_retail



UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

Online Retail Data Set

[Download](#) [Data Folder](#) [Data Set Description](#)

Abstract: This is a transnational data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail.

Data Set Characteristics:	Multivariate, Sequential, Time-Series	Number of Instances:	541909	Area:	Business
Attribute Characteristics:	Integer, Real	Number of Attributes:	8	Date Donated	2015-11-06
Associated Tasks:	Classification, Clustering	Missing Values?	N/A	Number of Web Hits:	520559

Java keytool to store password



Add a password for the 'oraclegraph' connection

```
[opc@pgx ~]$ keytool -importpass -alias oraclegraph -keystore keystore.p12 -storetype pkcs12
```

Enter keystore password:

Re-enter new password:

Enter the password to be stored:

Re-enter password:



Login Jshell with keystore

```
[opc@pgx ~]$ cd /opt/oracle/graph/  
[opc@pgx graph]$ bin/opg-jshell --base_url https://localhost:7007 --username  
oraclegraph --secret_store /home/opc/keystore.p12
```

enter password for user oraclegraph (press Enter for no password):

enter password for keystore /home/opc/keystore.p12:

For an introduction type: /help intro

Oracle Graph Server Shell 20.4.0

Variables instance, session, and analyst ready to use.

opg-jshell>

Graph Configuration config-tables-distinct.json

```
{
  "jdbc_url":"jdbc:oracle:thin:@graph_high?TNS_ADMIN=/e
tc/oracle/graph/wallets",
  "username":"oraclegraph",
  "keystore_alias": "oraclegraph",
  "name":"retail",
  "vertex_providers": [
    {
      "name":"Customer",
      "format":"rdbms",
      "database_table_name":"CUSTOMERS",
      "key_column":"CUSTOMER_ID",
      "key_type":"string",
      "props":[
        {"name":"country", "type":"string"}
      ]
    },
    {
      "name":"Product",
      "format":"rdbms",
      "database_table_name":"PRODUCTS",
      "key_column":"STOCK_CODE",
      "key_type":"string",
      "props":[
        {"name":"description", "type":"string"}
      ]
    }
  ],
}
```

Graph Configuration config-tables-distinct.json

```
"edge_providers": [  
  {  
    "name": "has_purchased",  
    "format": "rdbms",  
    "database_table_name": "PURCHASES_DISTINCT",  
    "key_column": "PURCHASE_ID",  
    "source_column": "CUSTOMER_ID",  
    "destination_column": "STOCK_CODE",  
    "source_vertex_provider": "Customer",  
    "destination_vertex_provider": "Product",  
    "props": [  
      {  
        "name": "purchased_by",  
        "format": "rdbms",  
        "database_table_name": "PURCHASES_DISTINCT",  
        "key_column": "PURCHASE_ID",  
        "source_column": "STOCK_CODE",  
        "destination_column": "CUSTOMER_ID",  
        "source_vertex_provider": "Product",  
        "destination_vertex_provider": "Customer",  
        "props": [  
        ]  
      }  
    ]  
  },  
  {
```

Load Graph with graph configuration

```
opg-jshell> var graph =  
session.readGraphWithProperties("/opt/oracle/graph/data/config-tables-  
distinct.json", "Online Retail")  
graph ==> PgxGraph[name=Online Retail,N=8258,E=532452,created=1604072399247]  
opg-jshell> graph.queryPgql("SELECT n.description MATCH (n:Product) LIMIT  
3").print();  
+-----+  
| n.description |  
+-----+  
| GREEN ROUND COMPACT MIRROR |  
| LA PALMIERA WALL THERMOMETER |  
| SMALL HEART FLOWERS HOOK |  
+-----+  
$2 ==> PgqlResultSetImpl[graph=Online Retail,numResults=3]
```

PGQL Example

```
opg-jsshell> graph.queryPgql(" SELECT ID(c), ID(p), p.description FROM MATCH (c)-[has_purchased]->(p) WHERE ID(c) = 'cust_12353' ").print();
```

+-----+		
ID(c)	ID(p)	description
+-----+		
cust_12353	prod_37449	CERAMIC CAKE STAND + HANGING CAKES
cust_12353	prod_22890	NOVELTY BISCUITS CAKE STAND 3 TIER
cust_12353	prod_37446	MINI CAKE STAND WITH HANGING CAKES
cust_12353	prod_37450	CERAMIC CAKE BOWL + HANGING CAKES
+-----+		

```
$3 ==> PgqlResultSetImpl[graph=Online Retail,numResults=4]
```


Run Personalized PageRank Analysis

```
opg-jshell> var vertex = graph.getVertex("cust_12353");  
vertex ==> PgxVertex[provider=Customer,ID=cust_12353]  
opg-jshell> analyst.personalizedPagerank(graph, vertex);  
$5 ==> VertexProperty[name=pagerank,type=double,graph=Online Retail]  
opg-jshell>
```

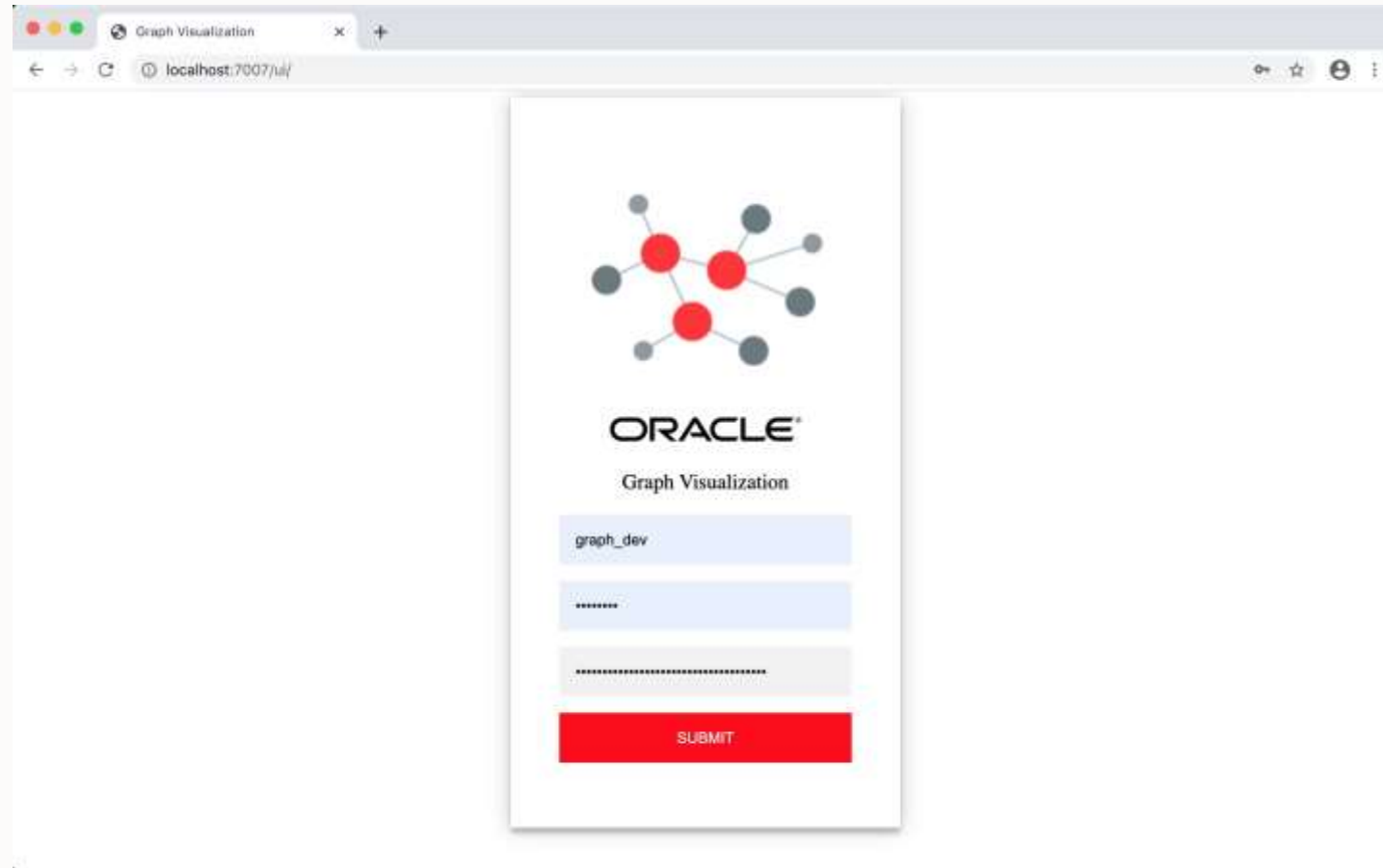
PGQL

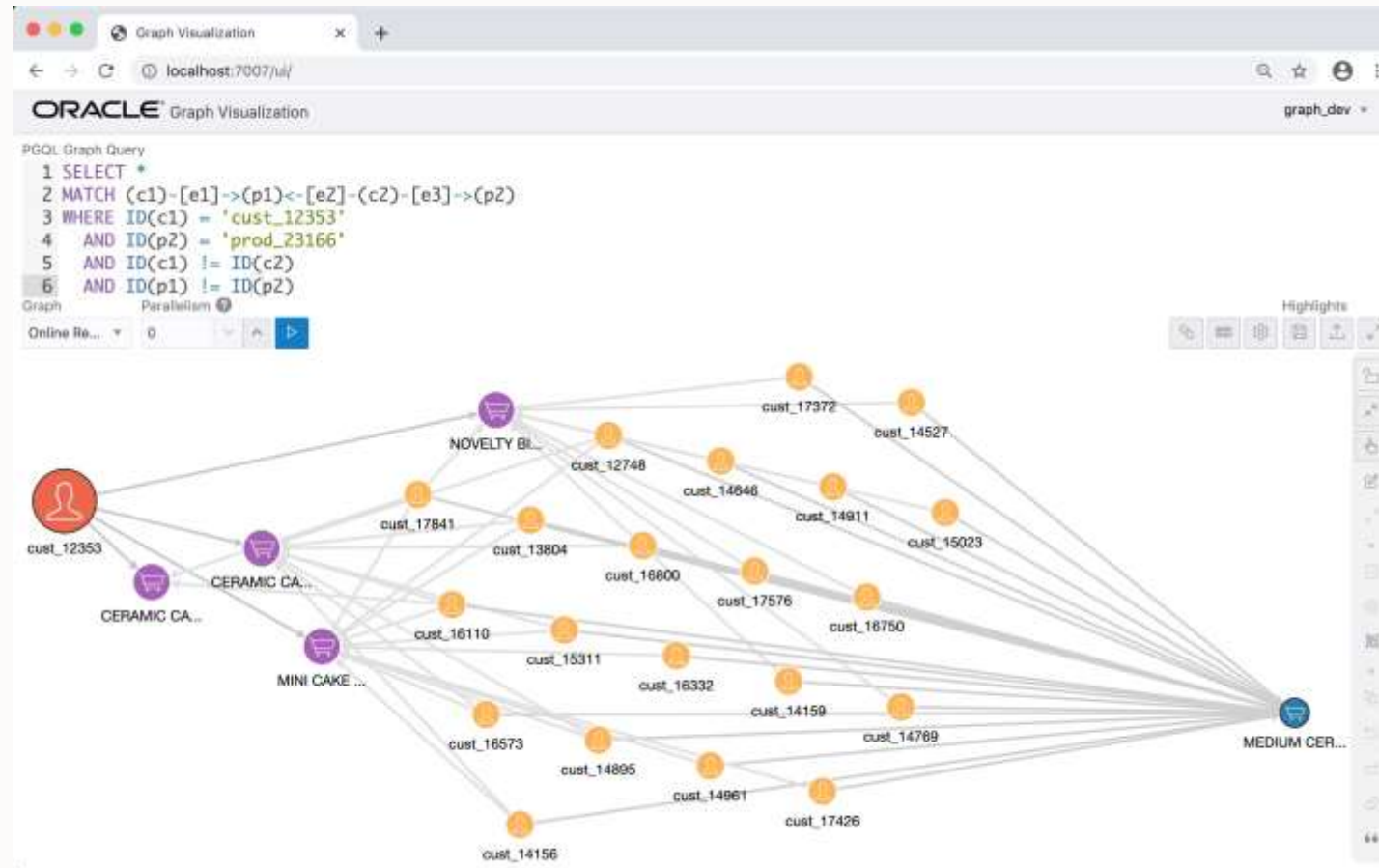
```
opg-jsshell> graph.queryPgql(  
...> "  SELECT ID(p), p.description, p.pagerank " +  
...> "  MATCH (p) " +  
...> "  WHERE LABEL(p) = 'Product' " +  
...> "    AND NOT EXISTS ( " +  
...> "      SELECT * " +  
...> "      MATCH (p)-[:purchased_by]->(a) " +  
...> "      WHERE ID(a) = 'cust_12353' " +  
...> "    ) " +  
...> "  ORDER BY p.pagerank DESC" +  
...> "  LIMIT 10"  
...> ).print();
```



ID(p)	p.description	p.pagerank
prod_22423	REGENCY CAKESTAND 3 TIER	0.0013483656895780121
prod_85123A	WHITE HANGING HEART T-LIGHT HOLDER	0.001300076481737168
prod_21232	STRAWBERRY CERAMIC TRINKET POT	0.001064278703175063
prod_22720	SET OF 3 CAKE TINS PANTRY DESIGN	9.98725982689145E-4
prod_47566	PARTY BUNTING	8.80044605313488E-4
prod_21231	SWEETHEART CERAMIC TRINKET BOX	8.793185974570984E-4
prod_21212	PACK OF 72 RETROSPOT CAKE CASES	7.74948580210001E-4
prod_84991	60 TEATIME FAIRY CAKE CASES	7.561654694509064E-4
prod_85099B	JUMBO BAG RED RETROSPOT	7.25890414385824E-4
prod_84879	ASSORTED COLOUR BIRD ORNAMENT	7.223349157689757E-4



```
opg-jshell> session.getId();  
$7 ==> "7363760b-2398-4a82-9554-e2e9bd5807f9"  
opg-jshell>
```







```
SELECT *  
MATCH (c1)-[e1]->(p1)<-[e2]-(c2)-[e3]->(p2)  
WHERE ID(c1) = 'cust_12353'  
      AND ID(p2) = 'prod_23166'  
      AND ID(c1) != ID(c2)  
      AND ID(p1) != ID(p2)
```

Our mission is to help people
see data in new ways, discover insights,
unlock endless possibilities.





ORACLE