## **Project 2: Phase 2 – Research**

## **NoSQL: Orient DB**

### 1. Introduction

OrientDB is an open source multi-model NoSQL DBMS that supports object oriented concepts, reactivity, documents and native graphs. It is fast and is written in java. We can go through thousands of records very fast. It also has a strong security profiling system, based on user, roles and it also supports various query languages along with SQL. Because of the query type very similar to SQL, OrientDB has a fast-learning curve and is thoroughly enjoyed by people who move from SQL to NoSQL dbases. OrientDB uses several indexing mechanisms based on the B-trees and hashing.

### 2. Motivations

OrientDB can be considered as a graph database, but it can also support documents and key-value pair. Graph databases have risen in popularity due to issue of efficiently exploring relations in the data that is not possible in other types of DBMS. Scalability and speed of data processing is being faced as times are changing.

OrientDB has been getting popular because it handles these issues. Many databases cannot be scaled out, OrientDB scales out very well on multiple machines. Based on the site[2] it can *be up to 302,231,454,903,657 billion (2^78) records for the maximum capacity of 19,807,040,628,566,084 Terabytes of data on a single server or multiple nodes.* OrientdB has also been designed to be extremely fast. It uses best features and concepts from Object databases, graph dbms and NoSQL engines. It manages relation using direct pointers(links) between records instead of using costly operation like JOIN. The size of the records doesn’t matter, the cost of traversing will remain the same.

### 3. Characteristics

OrientDB allows flexibility for complex cases and is designed to manage big-data tasks. It has several benefits and characteristics, some of which are listed below

The multi-master plus sharded architecture of OrientDB means that all servers are masters. This ensures both horizontal and vertical scalability. The program can run on any platform without needing to be installed or configured. It can be used as a drop-in replacement for today's most popular graph databases. Without the sample database, the whole server distribution is roughly 2MB. OrientDB is flexible and reliable enough to be used as an operational database management system. It doesn't require a restart or downtime for maintenance, and it repurposes empty space left over from deleted records while still online.

If you outgrow your dbms, it can be expensive to replace it. OrientDB is the key since it eliminates the need to invest in so many applications to achieve your objectives. OrientDB was built to be a high-performance database. It allows for quick read and write operations. It can handle up to 150,000 records per second in writes. For real-time analytics, the time series module can add 120,000 data per second while keeping the time graph synchronized.

### 4. Data model

OrientDB engine can support graph, document, key/value and object models, because of its real multi-model capabilities, which merge all of the above 4 model features into the core, OrientDB is being chosen by lot of people. Below we will see and simplify the explanations of the different models[4].

4.1. The Document Model

This model's data is kept within documents. A document is a collection of key/value pairs (also known as fields or properties), where the key provides access to the value. Primitive data types, embedded documents, and arrays of other values can all be stored in values. For document grouping, OrientDB employs the notions of "classes" and "clusters" as a type of "collections."

OrientDB's Document model involves the concept of a "LINK" as a document relationship. You may choose whether to embed documents or connect to them directly with OrientDB. OrientDB automatically resolves all links when you fetch a document. This is a significant difference from other Document Databases, such as MongoDB or CouchDB, where the developer must manage the relationships between the documents.

Graphical user interface, application

Description automatically generated

4.2 The Graph Model

Graph model consists of network structure consisting of vertices and edges. OrientDB defines it’s graph model as:

* Vertex - an entity that can be linked with other Vertices and has the following mandatory properties:
  + unique identifier
  + set of incoming Edges
  + set of outgoing Edges
* Edge - an entity that links two Vertices and has the following mandatory properties:
  + unique identifier
  + link to an incoming Vertex (also known as head)
  + link to an outgoing Vertex (also known as tail)
  + label that defines the type of connection/relationship between head and tail vertex

Vertices can also hold custom set of properties and these properties can be defined by users, which can make them similar to documents.

Table

Description automatically generated

4.3 The key/value model

This is easier than the previous two. OrientDB accepts Documents and Graph Elements as values, providing for a more robust model than the traditional Key/Value paradigm. The traditional Key/Value paradigm uses "buckets" to aggregate key/value pairs into distinct containers. The Key/Value Model's most well-known applications are:

* POST the value as payload of the HTTP call -> /<bucket>/<key>
* GET the value as payload from the HTTP call -> /<bucket>/<key>
* DELETE the value by Key, by calling the HTTP call -> /<bucket>/<key>

4.4 The object model

This approach was inherited by Object Oriented programming and provides:

* type inheritance: sub-types expand super-types
* polymorphism: when referring to a base class
* direct binding: from/to Objects used in programming languages.

Graphical user interface, application, table

Description automatically generated

### 5. Brief Manual

5.1. Installation

Here we will see how to download and install orientDB on a windows machine.

Graphical user interface

Description automatically generated1. Download OrientDB from the following website: <https://orientdb.org/download>

On the website you will get multiple options, based on the operating system you have. I will be selecting windows.

After the package is downloaded, we will extract it to a folder. We will also add the path of this folder to environment variables, so we can run server.bat file from any directory/drive.

A picture containing text

Description automatically generated2. Open command prompt and navigate to the location where bin folder is. Type ‘server.bat’ in cmd and click enter, you should something like below.

3. make sure you have java installed; else you would get an error as shown above.

Text

Description automatically generated

After completing the installation, you will be asked to generate a password and also be given a link to the orient db server highlighted above.

4. You will now go to the OrientDB server on your browser.

Graphical user interface, application

Description automatically generated

Once on server, you will see the following screen. A demodb has been created for you to play around with. Your default user for this demodb is ‘root’ and password will be the password you setup in the cmd in the previous step.

Here you have successfully downloaded and installed OrientDB in your windows System.

5.2 Data Definition and Data manipulation

OrientDB is very similar to using SQL when we compare if to Neo4j which is a popular graph dbms.

Graphical user interface, text, application, chat or text message

Description automatically generated

5.2.1. DDL – Data Definition Language.

As the name suggests DDL is used to define data structures. Create, modify and removing database objects are all part of the DDL. Below is the syntax and example of how SQL is used in OrientDB [5]

1. Create

Syntax to create a Vertex:

CREATE VERTEX [<class>] [CLUSTER <cluster>] [SET <field> = <expression>[,]\*]

Example: create a new vertex with a particular cluster

**Orientdb> CREATE VERTEX V1 CLUSTER set1.**

OrientdB provides create operations for class, cluster, edge, user, index etc.

2. Drop

Syntax to drop class:

DROP CLASS <class> [ UNSAFE ]

Unsafe defines if the class that is getting dropped has non-empty edges and vertex classes.

Example: **DROP CLASS Customer**

3. Alter

Alter will update attributes of an existing class.

Syntax: ALTER CLASS <class> <attribute-name> <attribute-value>

Example: define a super-class

**orientdb> ALTER CLASS Employee SUPERCLASS Person**

4. Truncate

Truncate command deletes all the records on the data structure it is being called upon.

Syntax for a cluster:

TRUNCATE CLUSTER <cluster>

Example:

**orientdb> TRUNCATE CLUSTER set1**

5.2.2 Data Manipulation Language

The commands defined in DML are used to carry our operations on already existing data structures and data. Retrieving, inserting, and deleting the data are the most common operations performed on databases.

1. Select

Syntax of the select statement on OrientDB documents:

SELECT [ <Projections> ] [ FROM <Target> [ LET <Assignment>\* ] ]

[ WHERE <Condition>\* ]

[ GROUP BY <Field>\* ]

[ ORDER BY <Fields>\* [ ASC|DESC ] \* ]

[ UNWIND <Field>\* ]

[ SKIP <SkipRecords> ]

[ LIMIT <MaxRecords> ]

[ FETCHPLAN <FetchPlan> ]

[ TIMEOUT <Timeout> [ <STRATEGY> ]

[ LOCK default|record ]

[ PARALLEL ]

[ NOCACHE ]

Target – This can be class, cluster, record id or a set of record ids.

Example: the following query will return all the records from the class person where name starts with Luk.

orientdb> **SELECT FROM Person WHERE name LIKE 'Luk%'**

2. Insert

Insert statement creates a new record in the database. Syntax is :

INSERT INTO [CLASS:]<class>|CLUSTER:<cluster>|INDEX:<index>

[(<field>[,]\*) VALUES (<expression>[,]\*)[,]\*]|

[SET <field> = <expression>|<sub-command>[,]\*]|

[CONTENT {<JSON>}]

[RETURN <expression>]

[FROM <query>]

There are multiple ways you can use insert statement for creating a new record

Example1: using abbreviated syntax

orientdb> **INSERT INTO Profile SET name = 'Jay', surname = 'Miner'**

example2: using JSON as content

orientdb> **INSERT INTO Profile CONTENT {"name": "Jay", "surname": "Miner"}**

3. Update

Syntax:

UPDATE <class>|CLUSTER:<cluster>|<recordID>

[SET|REMOVE <field-name> = <field-value>[,]\*]|[CONTENT|MERGE <JSON>]

[UPSERT]

[RETURN <returning> [<returning-expression>]]

[WHERE <conditions>]

[LOCK default|record]

[LIMIT <max-records>] [TIMEOUT <timeout>]

Example: you can use update to remove a field from all records

orientdb> **UPDATE Profile REMOVE nick**

4. Delete

Syntax:

DELETE FROM <Class>|CLUSTER:<cluster>|INDEX:<index> [LOCK <default|record>] [RETURN <returning>]

[WHERE <Condition>\*] [LIMIT <MaxRecords>] [TIMEOUT <timeout>]

Example:

**DELETE FROM Profile WHERE surname.toLowerCase() = 'unknown'**

### 6. Conclusion

Based on the all the data above, OrientDB is a strong NoSQL dbms combining not only interfaces, but actual function of various models. Since it is open source, it is very likely to be better quality, since the code can be checked, tested and issues can be reported globally. There are many scopes of improvements when compared to its competitors, but we cannot overlook the key features of speed, using links and pointers for data joining, support for function integration which is similar to RDBMS’s stored procedures, OreintDB is a great NoSQL Db to work with.

### 7. References

[1] https://db-engines.com/en/system/Microsoft+SQL+Server%3BOrientDB

[2] https://github.com/orientechnologies/orientdb

[3] https://www.featuredcustomers.com/vendor/orientdb/reviews

[4] https://orientdb.com/docs/3.0.x/datamodeling/Tutorial-Document-and-graph-model.html

[5] https://orientdb.com/docs/3.0.x/sql/SQL-Create-Class.html

[6] https://medium.com/@helloclassmedia/how-and-when-to-use-orientdb-a-study-by-helloclass-b84404e2bc9

### 