About the Tutorial

OrientDB is an Open Source NoSQL Database Management System, which contains the features of traditional DBMS along with the new features of both Document and Graph DBMS. It is written in Java and is amazingly fast. It can store 220,000 records per second on commodity hardware.

In the following chapters of this tutorial, we will look closely at OrientDB, one of the best open-source, multi-model, next generation NoSQL product.

Audience

This tutorial is designed for software professionals who are willing to learn NoSQL Database in simple and easy steps. This tutorial will give a great understanding on OrientDB concepts.

Prerequisites

OrientDB is NoSQL Database technologies which deals with the Documents, Graphs and traditional database components, like Schema and relation. Thus it is better to have knowledge of SQL. Familiarity with NoSQL is an added advantage.

Disclaimer & Copyright

© Copyright 2018 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com.
# Table of Contents

- About the Tutorial ........................................................................................................i
- Audience ......................................................................................................................i
- Prerequisites ...............................................................................................................i
- Disclaimer & Copyright ..............................................................................................i
- Table of Contents ........................................................................................................ii

1. ORIENTDB – OVERVIEW ......................................................................................... 1

2. ORIENTDB – INSTALLATION ..................................................................................... 2
   - Step 1: Download OrientDB Binary Setup File .........................................................2
   - Step 2: Extract and Install OrientDB ........................................................................3
   - Step 3: Configuring OrientDB Server as a Service ..................................................3
   - Step 4: Verifying OrientDB Installation ...................................................................7

3. ORIENTDB – BASIC CONCEPTS .............................................................................. 13
   - Document Model .....................................................................................................13
   - Graph Model ..........................................................................................................13
   - The Key/Value Model .............................................................................................14
   - The Object Model ...................................................................................................15

4. ORIENTDB – DATA TYPES ...................................................................................... 18

5. ORIENTDB – CONSOLE MODES ............................................................................. 21
   - Interactive Mode ...................................................................................................21
   - Batch Mode ............................................................................................................21
   - Enable Echo ..........................................................................................................22

6. ORIENTDB – CREATE DATABASE ............................................................................. 23

7. ORIENTDB – ALTERT DATABASE ............................................................................. 24
27. ORIENTDB – EXPORT RECORD ..............................................................57
28. ORIENTDB – UPDATE RECORD ...........................................................59
29. ORIENTDB – TRUNCATE RECORD .......................................................61
30. ORIENTDB – DELETE RECORD ..........................................................62
31. ORIENTDB – CREATE CLASS ..............................................................64
32. ORIENTDB – ALTER CLASS ...............................................................66
33. ORIENTDB – TRUNCATE CLASS ..........................................................68
34. ORIENTDB – DROP CLASS .................................................................69
35. ORIENTDB – CREATE CLUSTER ..........................................................70
36. ORIENTDB – ALTER CLUSTER ............................................................71
37. ORIENTDB – TRUNCATE CLUSTER ......................................................73
38. ORIENTDB – DROP CLUSTER ..............................................................74
39. ORIENTDB – CREATE PROPERTY .........................................................75
40. ORIENTDB – ALTER PROPERTY ..........................................................76
41. ORIENTDB – DROP PROPERTY ...........................................................78
42. ORIENTDB – CREATE VERTEX ............................................................79
43. ORIENTDB – MOVE VERTEX ...............................................................81
44. ORIENTDB – DELETE VERTEX ............................................................83
45. ORIENTDB – CREATE EDGE ...............................................................84
46. ORIENTDB – UPDATE EDGE ...............................................................85
OrientDB is an Open Source NoSQL Database Management System. **NoSQL Database** provides a mechanism for storing and retrieving NO-relation or NON-relational data that refers to data other than tabular data such as document data or graph data. NoSQL databases are increasingly used in Big Data and real-time web applications. NoSQL systems are also sometimes called "Not Only SQL" to emphasize that they may support SQL-like query languages.

OrientDB also belongs to the NoSQL family. OrientDB is a second generation Distributed Graph Database with the flexibility of Documents in one product with an open source of Apache 2 license. There were several NoSQL databases in the market before OrientDB, one of them being MangoDB.

**MangoDB vs OrientDB**

MangoDB and OrientDB contains many common features but the engines are fundamentally different. MangoDB is pure Document database and OrientDB is a hybrid Document with graph engine.

<table>
<thead>
<tr>
<th>Features</th>
<th>MangoDB</th>
<th>OrientDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationships</td>
<td>Uses the RDBMS JOINS to create relationship between entities. It has high runtime cost and does not scale when database scale increases.</td>
<td>Embeds and connects documents like relational database. It uses direct, super-fast links taken from graph database world.</td>
</tr>
<tr>
<td>Fetch Plan</td>
<td>Costly JOIN operations.</td>
<td>Easily returns complete graph with interconnected documents.</td>
</tr>
<tr>
<td>Transactions</td>
<td>Doesn’t support ACID transactions, but it supports atomic operations.</td>
<td>Supports ACID transactions as well as atomic operations.</td>
</tr>
<tr>
<td>Query language</td>
<td>Has its own language based on JSON.</td>
<td>Query language is built on SQL.</td>
</tr>
<tr>
<td>Indexes</td>
<td>Uses the B-Tree algorithm for all indexes.</td>
<td>Supports three different indexing algorithms so that the user can achieve best performance.</td>
</tr>
<tr>
<td>Storage engine</td>
<td>Uses memory mapping technique.</td>
<td>Uses the storage engine name LOCAL and PLOCAL.</td>
</tr>
</tbody>
</table>

OrientDB is the first Multi-Model open source NoSQL DBMS that brings together the power of graphs and flexibility of documents into a scalable high-performance operational database.
OrientDB installation file is available in two editions:

- **Community Edition:** OrientDB community edition is released by Apache under 0.2 license as an open source.

- **Enterprise Edition:** OrientDB enterprise edition is released as a proprietary software, which is built on community edition. It serves as an extension of the community edition.

This chapter explains the installation procedure of OrientDB community edition because it is open source.

**Prerequisites**

Both community and Enterprise editions can run on any Operating system that implements the Java Virtual Machine (JVM). OrientDB requires Java with 1.7 or later version.

Use the following steps to download and install OrientDB into your system.

**Step 1: Download OrientDB Binary Setup File**

OrientDB comes with built-in setup file to install the database on your system. It provides different pre-compiled binary packages (tarred or zipped packages) for different operating systems. You can download OrientDB files from [Download OrientDB](#) link.

The following screenshot shows the download page of OrientDB. You can download the zipped or tarred file by clicking the suitable operating system icon.
On downloading, you will get the binary package in your Downloads folder.

**Step 2: Extract and Install OrientDB**

Following is the procedure to extract and install OrientDB for different operating systems.

**In Linux**

After download you will get `orientdb-community-2.1.9.tar.gz` file in your Downloads folder. You can use the following command to extract the tarred file.

```
$ tar -zxvf orientdb-community-2.1.9.tar.gz
```

You can use the following command to move all the OrientDB library files from `orientdb-community-2.1.9` to `/opt/orientdb/` directory. Here we are using super user command (sudo) therefore you have to provide super user password to execute the following command.

```
$ sudo mv orientdb-community-2.1.9 /opt/orientdb
```

You can use the following commands to register the `orientdb` command and the Orient server.

```
$ export ORIENTDB_HOME = /opt/orientdb
$ export PATH = $PATH:$ORIENTDB_HOME/bin
```

**In Windows**

- After download you will get `orientdb-community-2.1.9.zip` file in your Downloads folder. Extract the zip file using the zip extractor.

- Move the extracted folder into the C:\ directory.

- Create two environmental variables ORIENTDB_HOME and PATH variables with following given values.

```
ORIENT_HOME = C:\orientdb-community-2.1.9
PATH = C:\orientdb-community-2.1.9\bin
```
Step 3: Configuring OrientDB Server as a Service

By following the above steps you can use the Desktop version of OrientDB. You can start OrientDB database server as a service by using the following steps. The procedure is different, depending on your operating system.

In Linux

OrientDB provides a script file named `orientdb.sh` to run the database as a daemon. You can find it in the bin/directory of your OrientDB installation directory that is `$ORIENTDB_HOME/bin/orientdb.sh`.

Before running the script file, you have to edit `orientdb.sh` file for defining two variables. One is `ORIENTDB_DIR` which defines the path to the installation directory (`/opt/orientdb`) and the second is `ORIENTDB_USER` which defines the username you want run OrientDB for as follows.

```
ORIENTDB_DIR = "/opt/orientdb"
ORIENTDB_USER = "<username you want to run OrientDB>"
```

Use the following command to copy `orientdb.sh` file into `/etc/init.d/` directory for initializing and running the script. Here we are using super user command (sudo) therefore you have to provide super user password to execute the following command.

```
$ sudo cp $ORIENTDB_HOME/bin/orientdb.sh /etc/init.d/orientdb
```

Use the following command to copy the `console.sh` file from OrientDB installation directory that is `$ORIENTDB_HOME/bin` to the system bin directory that is `usr/bin` for accessing the OrientDB’s console.

```
$ sudo cp $ORIENTDB_HOME/bin/console.sh /usr/bin/orientdb
```

Use the following command to start the ORIENTDB database server as service. Here you have to provide the respective user’s password which you mention in the `orientdb.sh` file to start the server.

```
$ service orientdb start
```

Use the following command to know on which PID the OrientDB server daemon is running.

```
$ service orientdb status
```

Use the following command to stop the OrientDB server daemon. Here you have to provide the respective user’s password, which you mention in the `orientdb.sh` file to stop the server.

```
$ service orientdb stop
```
In Windows

OrientDB is a server application therefore it has to perform several tasks before starting shutting down the Java virtual machine process. If you want to shutdown OrientDB server manually then you have to execute `shutdown.bat` file. But the server instances do not stop correctly, when the system shuts down suddenly without executing the above script. The programs which are controlled by the operating system with a set of specified signals are called services in Windows.

We have to use Apache Common Daemon which allow Windows users to wrap Java applications as Windows service. Following is the procedure to download and register Apache common daemon.

- Click the following link for Apache Common Daemons for windows.
- Click on `common-daemon-1.0.15-bin-windows` to download.
- Unzip the `common-daemon-1.0.15-bin-windows` directory. After extracting you will find `prunsrv.exe` and `prunmgr.exe` files inside the directory. In those:
  - `prunsrv.exe` file is a service application for running applications as services.
  - `prunmgr.exe` file is an application used for monitoring and configuring windows services.
- Go to OrientDB installation folder -> create a new directory and name it service.
- Copy the `prunsrv.exe` and `prunmgr.exe` paste it into to the service directory.
- In order to configure OrientDB as Windows service, you have to execute a short script that uses the prusrv.exe as a Windows service.
- Before defining the Windows Services, you have to rename prunsrv and prunmgr according to the name of the service. For e.g. OrientDBGraph and OrientDBGraphw respectively. Here OrientDBGraph is the name of the service.
- Copy the following script into the file named `installService.bat` and place it into `%ORIENTDB_HOME%\service\` directory.

```batch
:: OrientDB Windows Service Installation
@echo off
rem Remove surrounding quotes from the first parameter
set str=%~1
rem Check JVM DLL location parameter
if "%str%" == "" goto missingJVM
set JVM_DLL=%str%
rem Remove surrounding quotes from the second parameter
```
set str=%~2
rem Check OrientDB Home location parameter
if "%str%" == "" goto missingOrientDBHome
set ORIENTDB_HOME=%str%

set CONFIG_FILE=%ORIENTDB_HOME%/config/orientdb-server-config.xml
set LOG_FILE=%ORIENTDB_HOME%/config/orientdb-server-log.properties
set LOG_CONSOLE_LEVEL=info
set LOG_FILE_LEVEL=fine
set WWW_PATH=%ORIENTDB_HOME%/www
set ORIENTDB_ENCODING=UTF8
set ORIENTDB_SETTINGS=-Dprofiler.enabled=true -Dcache.level1.enabled=false -Dcache.level2.strategy=1
set JAVA_OPTS_SCRIPT=-XX:+HeapDumpOnOutOfMemoryError

rem Install service
OrientDBGraphX.X.X.exe //IS --DisplayName=“OrientDB GraphEd X.X.X” ^
--Description="OrientDB Graph Edition, aka GraphEd, contains OrientDB server integrated with the latest release of the TinkerPop Open Source technology stack supporting property graph data model.” ^
--StartClass=com.orientechnologies.orient.server.OServerMain --StopClass=com.orientechnologies.orient.server.OServerShutdownMain ^
--Classpath="%ORIENTDB_HOME%/lib\*" --JvmOptions "-Dfile.Encoding=%ORIENTDB_ENCODING%;-Djava.util.logging.config.file=%LOG_FILE%;-Dorientdb.config.file="%CONFIG_FILE%";-Dorientdb.www.path="%WWW_PATH%";-Dlog.console.level=%LOG_CONSOLE_LEVEL%,-Dlog.file.level=%LOG_FILE_LEVEL%;-Dorientdb.build.number="%BUILD%";-%ORIENTDB_HOME%=%ORIENTDB_HOME%" ^
--StartMode=jvm --StartPath="%ORIENTDB_HOME%/bin" --StopMode=jvm --StopPath="%ORIENTDB_HOME%/bin" --Jvm="%JVM_DLL%" --LogPath="%ORIENTDB_HOME%/log" --Startup=auto

EXIT /B

:missingJVM
echo Insert the JVM DLL location
goto printUsage

:missingOrientDBHome
echo Insert the OrientDB Home
goto printUsage
The script requires two parameters:

- The location of jvm.dll, for e.g. `C:\ProgramFiles\java\jdk1.8.0_66\jre\bin\server\jvm.dll`
- The location of OrientDB installation for e.g. `C:\orientdb-community-2.1.9`

- The service is installed when you execute the OrientDBGraph.exe file (Original prunsrv) and double-click on it.

- Use the following command to install services into Windows.

```
> Cd %ORIENTDB_HOME%\service
> installService.bat "C:\Program Files\Java\jdk1.8.0_66\jre\bin\server\jvm.dll"
C:\orientdb-community-2.1.9
```

Open the Task Manager services, you will find the following screenshot with the registered service name is in it.
Step 4: Verifying OrientDB Installation

This step verifies the OrientDB database server installation using the following steps.

- Run the server.
- Run the console.
- Run the studio.

This is unique according to the operating system.

In Linux

Follow the given procedure for verifying OrientDB installation in Linux.

Running the server: You can use the following command to start the server.

```bash
$ cd $ORIENTDB_HOME/bin
$ ./server.sh
```

Or you can use the following command to start the OrientDB server as UNIX daemon.

```bash
$ service orientdb start
```

If it is installed successfully, you will receive the following output.
OrientDB

GRAPH DATABASE

orientdb.com

2016-01-20 19:17:21:547 INFO OrientDB auto-config DISKCACHE=1,649MB (heap=494MB os=4,192MB disk=199,595MB) [orienttechnologies]

2016-01-20 19:17:21:816 INFO Loading configuration from: /opt/orientdb/config/orientdb-server-config.xml...[OServerConfigurationLoaderXml]

2016-01-20 19:17:22:213 INFO OrientDB Server v2.1.9-SNAPSHOT (build 2.1.x@r; 2016-01-07 10:51:24+0000) is starting up... [OServer]


2016-01-20 19:17:22:361 INFO Port 0.0.0.0:2424 busy, trying the next available... [OServerNetworkListener]

2016-01-20 19:17:22:362 INFO Listening binary connections on 0.0.0.0:2425 (protocol v.32, socket=default) [OServerNetworkListener]

Running the console: You can use the following command to run the OrientDB under console.

$ orientdb
If it is installed successfully, you will receive the following output.

OrientDB console v.2.1.9-SNAPSHOT (build 2.1.x@r; 2016-01-07 10:51:24+0000)
www.orientdb.com
Type 'help' to display all the supported commands.
Installing extensions for GREMLIN language v.2.6.0

orientdb>

Running the Studio: After starting the server you can use the following URL (http://localhost:2480/) on your browser. You will get the following screenshot.

In Windows
Follow the given procedure for verifying OrientDB installation in Windows.

Running the server: You can use the following command to start the server.

> cd %ORIENTDB_HOME%\bin
> .\server.bat

If it is installed successfully, you will receive the following output.
`}$

**Running the console:** You can use the following command to run OrientDB under console.
If it is installed successfully, you will receive the following output.

OrientDB console v.2.1.9-SNAPSHOT (build 2.1.x@r; 2016-01-07 10:51:24+0000)  
www.orientdb.com  
Type 'help' to display all the supported commands.  
Installing extensions for GREMLIN language v.2.6.0

orientdb\>

**Running the Studio:** After starting the server you can use the following URL (http://localhost:2480/) on your browser. You will get the following screenshot.
The main feature of OrientDB is to support multi-model objects, i.e. it supports different models like Document, Graph, Key/Value and Real Object. It contains a separate API to support all these four models.

**Document Model**

The terminology Document model belongs to NoSQL database. It means the data is stored in the Documents and the group of Documents are called as *Collection*. Technically, document means a set of key/value pairs or also referred to as fields or properties.

OrientDB uses the concepts such as classes, clusters, and link for storing, grouping, and analyzing the documents.

The following table illustrates the comparison between relational model, document model, and OrientDB document model:

<table>
<thead>
<tr>
<th>Relational Model</th>
<th>Document Model</th>
<th>OrientDB Document Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Collection</td>
<td>Class or Cluster</td>
</tr>
<tr>
<td>Row</td>
<td>Document</td>
<td>Document</td>
</tr>
<tr>
<td>Column</td>
<td>Key/value pair</td>
<td>Document field</td>
</tr>
<tr>
<td>Relationship</td>
<td>Not available</td>
<td>Link</td>
</tr>
</tbody>
</table>

**Graph Model**

A graph data structure is a data model that can store data in the form of Vertices (Nodes) interconnected by Edges (Arcs). The idea of OrientDB graph database came from property graph. The vertex and edge are the main artifacts of the Graph model. They contain the properties, which can make these appear similar to documents.
The following table shows a comparison between graph model, relational data model, and OrientDB graph model.

<table>
<thead>
<tr>
<th>Relational Model</th>
<th>Graph Model</th>
<th>OrientDB Graph Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Vertex and Edge Class</td>
<td>Class that extends &quot;V&quot; (for Vertex) and &quot;E&quot; (for Edges)</td>
</tr>
<tr>
<td>Row</td>
<td>Vertex</td>
<td>Vertex</td>
</tr>
<tr>
<td>Column</td>
<td>Vertex and Edge property</td>
<td>Vertex and Edge property</td>
</tr>
<tr>
<td>Relationship</td>
<td>Edge</td>
<td>Edge</td>
</tr>
</tbody>
</table>

**The Key/Value Model**

The Key/Value model means that data can be stored in the form of key/value pair where the values can be of simple and complex types. It can support documents and graph elements as values.

The following table illustrates the comparison between relational model, key/value model, and OrientDB key/value model.

<table>
<thead>
<tr>
<th>Relational Model</th>
<th>Key/Value Model</th>
<th>OrientDB Key/Value Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Bucket</td>
<td>Class or Cluster</td>
</tr>
<tr>
<td>Row</td>
<td>Key/Value pair</td>
<td>Document</td>
</tr>
<tr>
<td>Column</td>
<td>Not available</td>
<td>Document field or Vertex/Edge property</td>
</tr>
<tr>
<td>Relationship</td>
<td>Not available</td>
<td>Link</td>
</tr>
</tbody>
</table>
The Object Model

This model has been inherited by Object Oriented programming and supports **Inheritance** between types (sub-types extends the super-types), **Polymorphism** when you refer to a base class and **Direct binding** from/to Objects used in programming languages.

The following table illustrates the comparison between relational model, Object model, and OrientDB Object model.

<table>
<thead>
<tr>
<th>Relational Model</th>
<th>Object Model</th>
<th>OrientDB Object Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Class</td>
<td>Class or Cluster</td>
</tr>
<tr>
<td>Row</td>
<td>Object</td>
<td>Document or Vertex</td>
</tr>
<tr>
<td>Column</td>
<td>Object property</td>
<td>Document field or Vertex/Edge property</td>
</tr>
<tr>
<td>Relationship</td>
<td>Pointer</td>
<td>Link</td>
</tr>
</tbody>
</table>

Before go ahead in detail, it is better to know the basic terminology associated with OrientDB. Following are some of the important terminologies.

**Record**

The smallest unit that you can load from and store in the database. Records can be stored in four types.

- Document
- Record Bytes
- Vertex
- Edge

**Record ID**

When OrientDB generates a record, the database server automatically assigns a unit identifier to the record, called RecordID (RID). The RID looks like #<cluster>:<position>. <cluster> means cluster identification number and the <position> means absolute position of the record in the cluster.

**Documents**

The Document is the most flexible record type available in OrientDB. Documents are softly typed and are defined by schema classes with defined constraint, but you can also insert the document without any schema, i.e. it supports schema-less mode too.
Documents can be easily handled by export and import in JSON format. For example, take a look at the following JSON sample document. It defines the document details.

```
{
   "id" : "1201",
   "name" : "Jay",
   "job" : "Developer",
   "creations" : [
      {
         "name" : "Amiga",
         "company" : "Commodore Inc."
      }, {
         "name" : "Amiga 500",
         "company" : "Commodore Inc."
      }
   ]
}
```

**RecordBytes**

Record Type is the same as BLOB type in RDBMS. OrientDB can load and store document Record type along with binary data.

**Vertex**

OrientDB database is not only a Document database but also a Graph database. The new concepts such as Vertex and Edge are used to store the data in the form of graph. In graph databases, the most basic unit of data is node, which in OrientDB is called a vertex. The Vertex stores information for the database.

**Edge**

There is a separate record type called the Edge that connects one vertex to another. Edges are bidirectional and can only connect two vertices. There are two types of edges in OrientDB, one is regular and another one lightweight.

**Class**

The class is a type of data model and the concept drawn from the Object-oriented programming paradigm. Based on the traditional document database model, data is stored in the form of collection, while in the Relational database model data is stored in tables. OrientDB follows the Document API along with OPPS paradigm. As a concept, the class in OrientDB has the closest relationship with the table in relational databases, but (unlike tables) classes can be schema-less, schema-full or mixed. Classes can inherit from other classes, creating trees of classes. Each class has its own cluster or clusters, (created by default, if none are defined).
Cluster
Cluster is an important concept which is used to store records, documents, or vertices. In simple words, Cluster is a place where a group of records are stored. By default, OrientDB will create one cluster per class. All the records of a class are stored in the same cluster having the same name as the class. You can create up to 32,767(2^15-1) clusters in a database.

The CREATE class is a command used to create a cluster with specific name. Once the cluster is created you can use the cluster to save records by specifying the name during the creation of any data model.

Relationships
OrientDB supports two kinds of relationships: referenced and embedded. Referenced relationships means it stores direct link to the target objects of the relationships. Embedded relationships means it stores the relationship within the record that embeds it. This relationship is stronger than the reference relationship.

Database
The database is an interface to access the real storage. IT understands high-level concepts such as queries, schemas, metadata, indices, and so on. OrientDB also provides multiple database types. For more information on these types, see Database Types.
OrientDB supports several data types natively. Following is the complete table on the same.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1       | Boolean| Handles only the values True or False. **Java types:** java.lang.Boolean  
Min: 0  
Max: 1 |
| 2       | Integer| 32-bit signed integers. **Java types:** java.lang.Interger  
Min: -2,147,483,648  
Max: +2,147,483,647 |
| 3       | Short  | Small 16-bit signed integers. **Java types:** java.lang.short  
Min: -32,768  
Max: 32,767 |
| 4       | Long   | Big 64-bit signed integers. **Java types:** java.lang.Long  
Min: -2^{63}  
Max: +2^{63}-1 |
| 5       | Float  | Decimal numbers. **Java types:** java.lang.Float  
Min: 2^{-149}  
Max: (2-2^{-23})*2^{127} |
| 6       | Double | Decimal numbers with high precision. **Java types:** java.lang.Double.  
Min: 2^{-1074}  
Max: (2-2^{-52})*2^{1023} |
| 7       | Date-time | Any date with the precision up to milliseconds. **Java types:** java.util.Date |
| 8       | String | Any string as alphanumeric sequence of chars. **Java types:** java.lang.String |
| 9       | Binary | Can contain any value as byte array. **Java types:** byte[ ]  
Min: 0  
Max: 2,147,483,647 |
<table>
<thead>
<tr>
<th></th>
<th>Type</th>
<th>Description</th>
<th>Java types</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Embedded</td>
<td>The record is contained inside the owner. The contained record has no RecordId.</td>
<td><strong>Java types:</strong> ORecord</td>
</tr>
</tbody>
</table>
| 10| Embedded list | The records are contained inside the owner. The contained records have no RecordIds and are reachable only by navigating the owner record. | **Java types:** List<objects>  
Min: 0  
Max: 41,000,000 items |
| 11| Embedded set | The records are contained inside the owner. The contained records have no RecordId and are reachable only by navigating the owner record. | **Java types:** set<objects>  
Min: 0  
Max: 41,000,000 items |
| 12| Embedded map | The records are contained inside the owner as values of the entries, while the keys can only be strings. The contained records have no RecordId and are reachable only by navigating the owner Record. | **Java types:** Map<String, ORecord>  
Min: 0  
Max: 41,000,000 items |
| 13| Link | Link to another Record. It's a common one-to-one relationship | **Java Types:** ORID, <? extends ORecord>  
Min: 1  
Max: 32767:2^63-1 |
| 14| Link list | Links to other Records. It's a common one-to-many relationship where only the RecordIds are stored. | **Java types:** List<? Extends ORecord>  
Min: 0  
Max: 41,000,000 items |
| 15| Link set | Links to other records. It's a common one-to-many relationship. | **Java types:** Set<? extends ORecord>  
Min: 0  
Max: 41,000,000 items |
| 16| Link map | Links to other records as value of the entries, while keys can only be strings. It's a common one-to-many relationship. Only the RecordIds are stored. | **Java types:** Map<String, ? extends Record>  
Min: 0  
Max: 41,000,000 items |
<table>
<thead>
<tr>
<th>No.</th>
<th>Type</th>
<th>Description</th>
<th>Java Types</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Byte</td>
<td>Single byte. Useful to store small 8-bit signed integers.</td>
<td>java.lang.Byte</td>
<td>-128</td>
<td>+127</td>
</tr>
<tr>
<td>18</td>
<td>Transient</td>
<td>Any value not stored on database.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Date</td>
<td>Any date as year, month and day.</td>
<td>java.util.Date</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Custom</td>
<td>Used to store a custom type providing the Marshall and Unmarshall methods.</td>
<td>OSerializableStream</td>
<td>0</td>
<td>x</td>
</tr>
<tr>
<td>21</td>
<td>Decimal</td>
<td>Decimal numbers without rounding.</td>
<td>java.math.BigDecimal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>LinkBag</td>
<td>List of RecordIds as specific RidBag.</td>
<td>ORidBag</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Any</td>
<td>Not determinate type, used to specify collections of mixed type, and null.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In the following chapters, how to use these data types in OrientDB is discussed.
The OrientDB Console is a Java Application made to work against OrientDB databases and Server instances. There are several console modes that OrientDB supports.

### Interactive Mode

This is the default mode. Just launch the console by executing the following script `bin/console.sh` (or `bin/console.bat` in MS Windows systems). Make sure to have execution permission on it.

```
orientdb>
```

Once done, the console is ready to accept commands.

### Batch Mode

To execute commands in batch mode run the following `bin/console.sh` (or `bin/console.bat` in MS Windows systems) script passing all the commands separated with semicolon `;`.

```
orientdb> console.bat "connect remote:localhost/demo;select * from profile"
```

Or call the console script passing the name of the file in text format containing the list of commands to execute. Commands must be separated with semicolon `;`.

### Example

Command.txt contains the list of commands which you want to execute through OrientDB console. The following command accepts the batch of commands from the command.txt file.

```
orientdb> console.bat commands.txt
```

In batch mode, you can ignore errors to let the script continue the execution by setting the "ignoreErrors" variable to true.

```
orientdb> set ignoreErrors true
```
Enable Echo

When you run console commands in pipeline, you will need to display them. Enable "echo" of commands by setting it as property at the beginning. Following is the syntax to enable echo property in OrientDB console.

orientdb> set echo true
End of ebook preview

If you liked what you saw...

Buy it from our store @ https://store.tutorialspoint.com