Big Data Hadoop and Spark Developer

simplilearn

Apache Hive

Learning Objectives

By the end of this lesson, you will be able to:

- Summarize Hive and its architecture
- Create and maintain tables using Hue Web UI and Beeline
- Study the various file formats supported by Hive
- Create tables and run queries using HiveQL DDL

DATA AND
ARTIFICIAL INTELLIGENCE

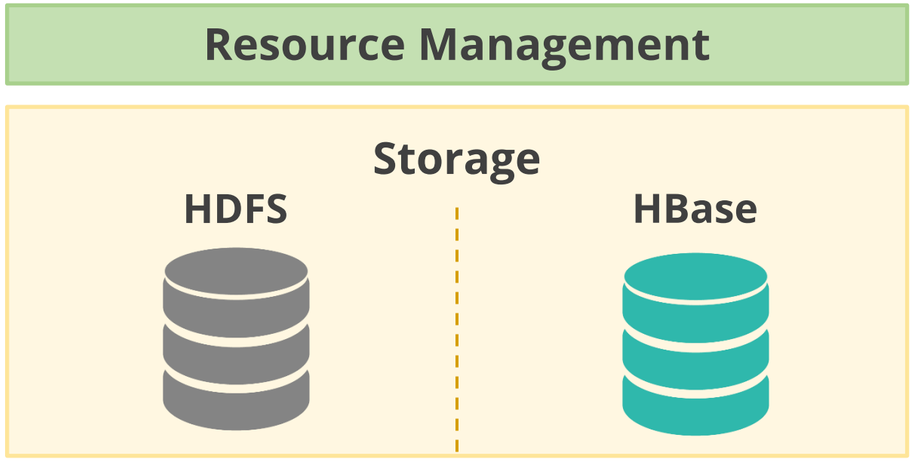
Hive: SQL over Hadoop MapReduce

Apache Hive

Apache Hive is a SQL-like open-source data warehousing application that extracts data from Hadoop and related systems.

Batch Processing


```
SELECT t1.a1 as c1, t2.b1 as c2
FROM t1 JOIN t2 ON (t1.a2=t2.b2);
```



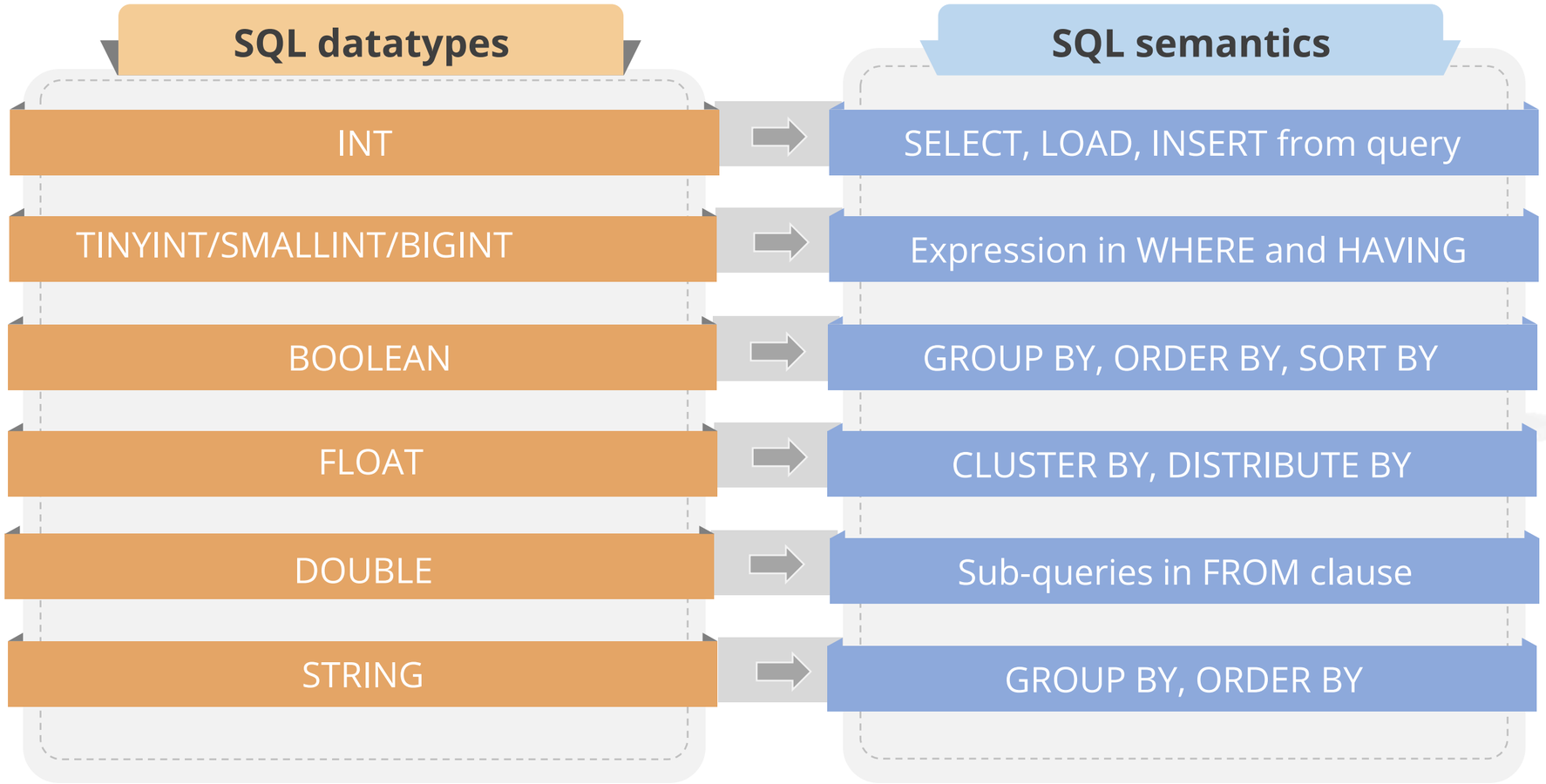
Features of Hive

Facebook originally developed Hives around 2007. It has the following features:

- Is originally developed by Facebook around 2007
- Is an open-source Apache project
- Provides high-level abstraction layer on top of MapReduce and Apache Spark
- Uses HiveQL
- Is suitable for structured data

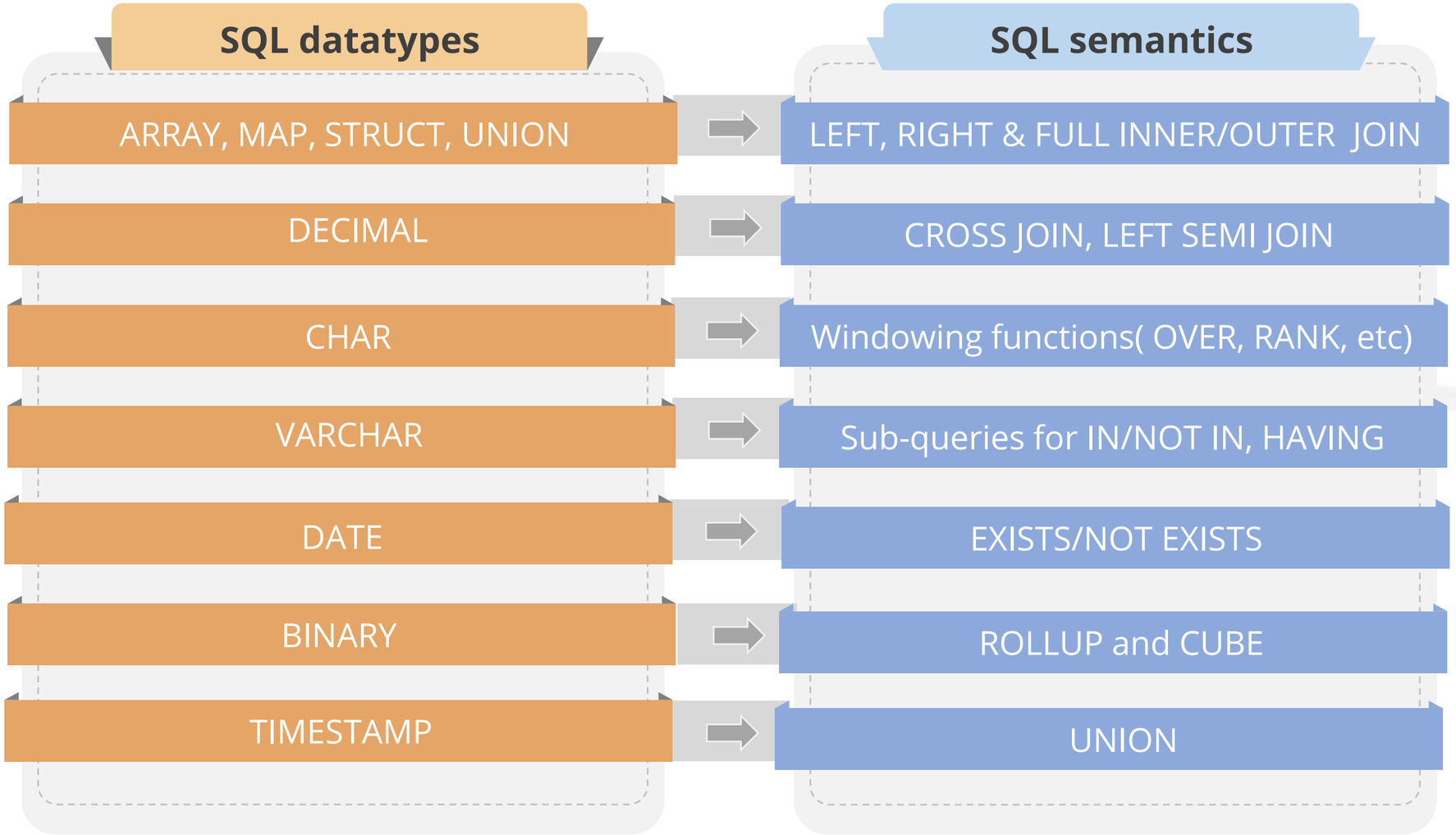
Hive's Alignment with SQL

The following figure describes Hive's alignment with SQL:



Hive's Alignment with SQL

The following figure describes Hive's alignment with SQL:



DATA AND
ARTIFICIAL INTELLIGENCE

Hive: Case Study

Case Study: FedEx

Problem Statement:

Amazon, eBay, and Grab were committed to making timely deliveries throughout the COVID period. Their focus was on courier services. FedEx-like organizations served as the backbone, ensuring that all national or international deliveries were made on time or with a tiny delay.

FedEx had a record-breaking quarter. The new CEO appreciates the drivers' efforts and considers sharing some of the profits with them. He requested a list of the top ten drivers based on the previous year's timesheet from the data engineering team.

Case Study: FedEx

Current Constraint:

The bulk of the data engineering team is conversant with SQL data processing and has limited Java knowledge. So, what should they do if they can't utilize MapReduce?

Solution:

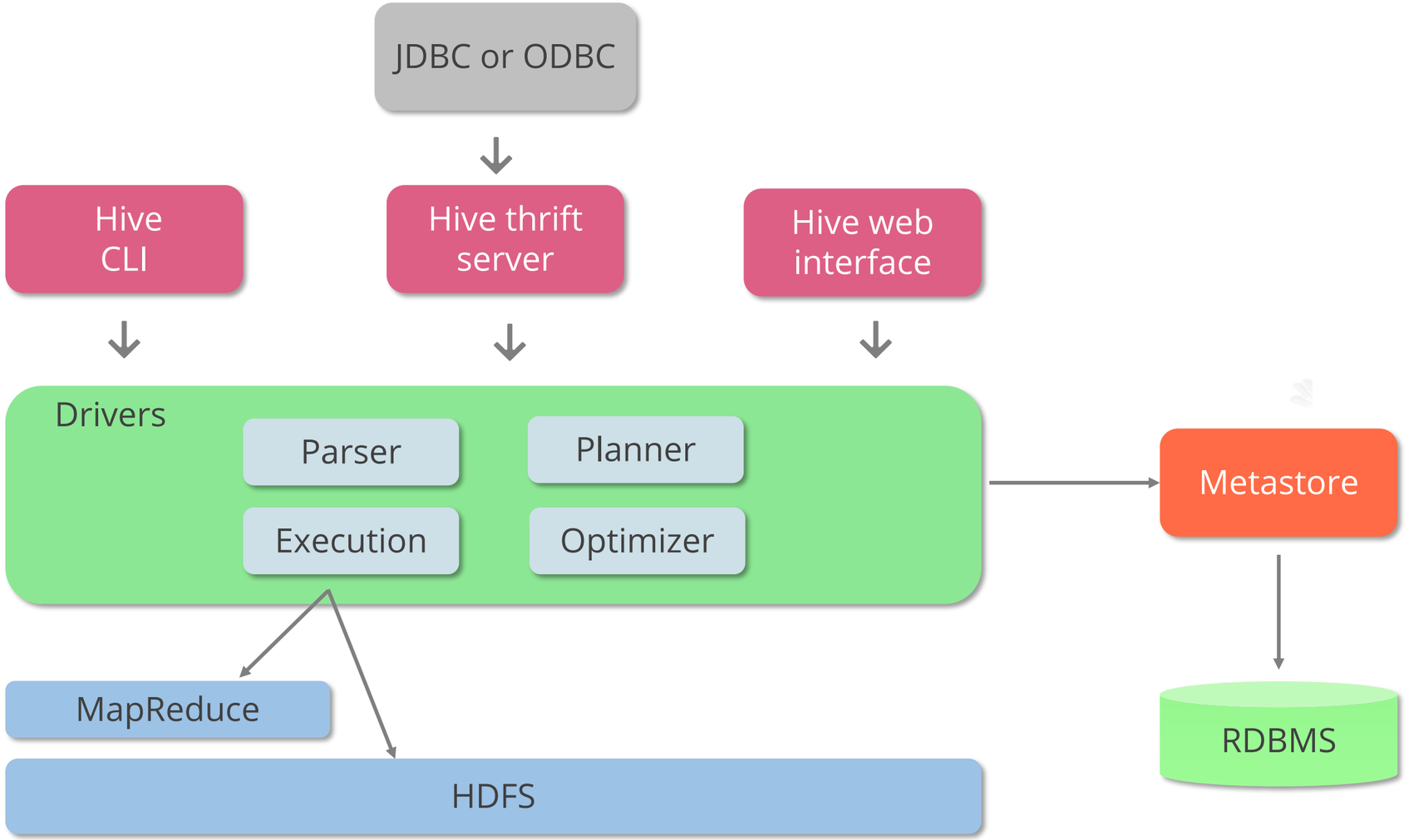
Apache Hive is a warehousing engine that processes data by writing SQL syntax.

DATA AND
ARTIFICIAL INTELLIGENCE

Hive Architecture

Hive Architecture

The main components of the hive architecture include Hadoop core components, metastore, driver, and hive clients.



Job Execution Flow in Hive

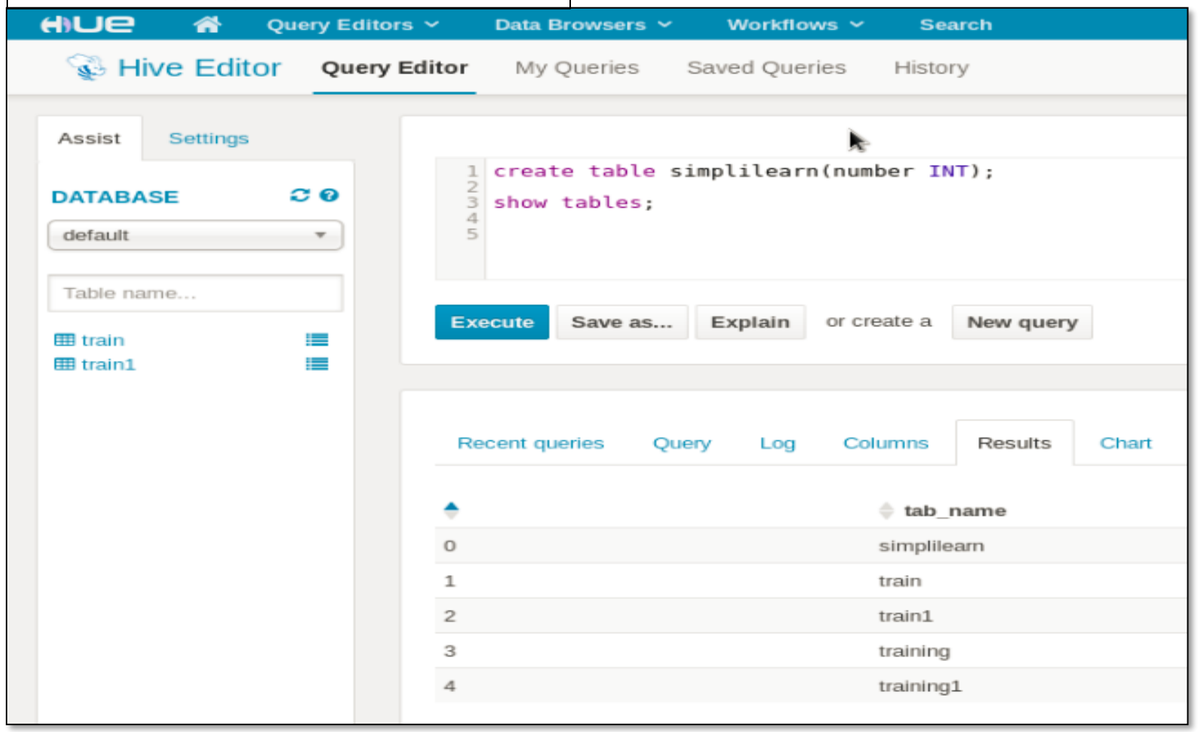
The job execution in Hive is completed in the following steps:

1. Parse HiveQL
2. Make optimizations
3. Plan execution
4. Submit job(s) to cluster
5. Monitor progress
6. Process data in MapReduce or Apache Spark
7. Store the data in HDFS

Interfaces to Run Hive Queries

Hive provides a variety of interfaces for performing queries. These are:

Hive Query Editor



- **Command-line shell**
  Hive: Beeline
- **Hue Web UI**
  Hive Query Editor
- **Metastore Manager**
  ODBC / JDBC

Connecting with Hive

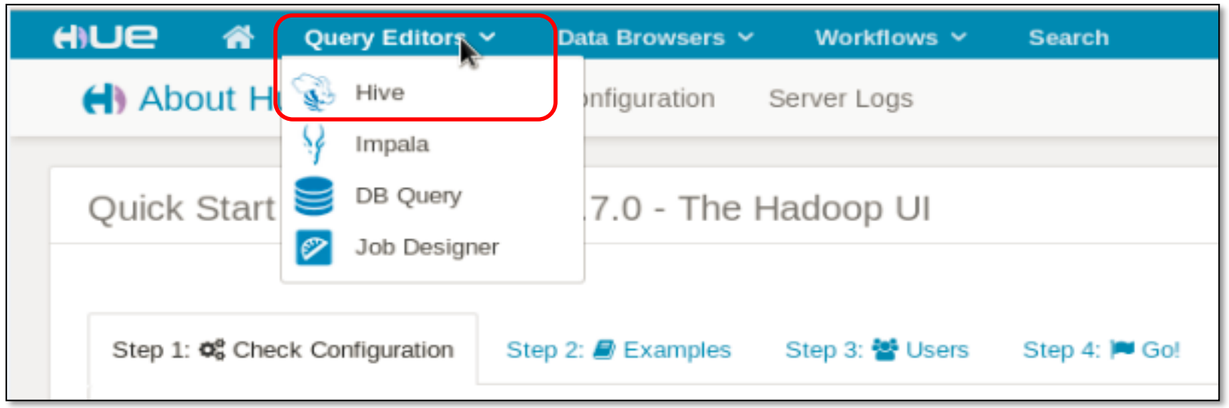
Beeline runs Hive.

training@localhost:~

```
[training@localhost ~]$ beeline
Beeline version 3.1.3000.7.2.12.4-1 by Apache Hive
0: jdbc:hive2://bdh-cluster2-master10.bdh-env>
```

Connecting with Hive

Hue writes a Hive query from a UI.



Running Hive Queries Using Beeline

The following are some Beeline commands:

- !exit – to quit the shell
- !help – to display a list of all available commands
- !verbose – to display additional query details

Note

'!' is used to execute Beeline commands.

Running Hive Queries Using Beeline

Use the following command to run Hive queries using beeline:

training@localhost:~

```
[training@localhost ~]$ beeline -u jdbc:hive2://localhost:1000
2016-07-28 21:36:22,075 WARN [main] mapreduce.TableMapReduceUtil: The hbase-pre
fix-tree module jar containing PrefixTreeCodec is not present. Conitnuing
without it.
Scan complete in 14ms
Connecting to jdbc:hive2://localhost:10000
Connected to: Apache Hive (version 3.1.3000.7.2.12.4-1)
Driver: Hive JDBC (version 3.1.3000.7.2.12.4-1)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 3.1.3000.7.2.12.4-1 by Apache Hive
0: jdbc:hive2://bdh-cluster2-master10.bdh-env>
```

Running Beeline from Command Line

The following statements and syntax run Beeline:

To execute a file, use the –u option.	beeline –u … -f simplilearn.hql
To use HiveQL from the command line, use the -e option.	beeline –u … -e 'SELECT * FROM users‘
To execute the script again if an error occurs	beeline –u … -force=TRUE

Running Hive Query

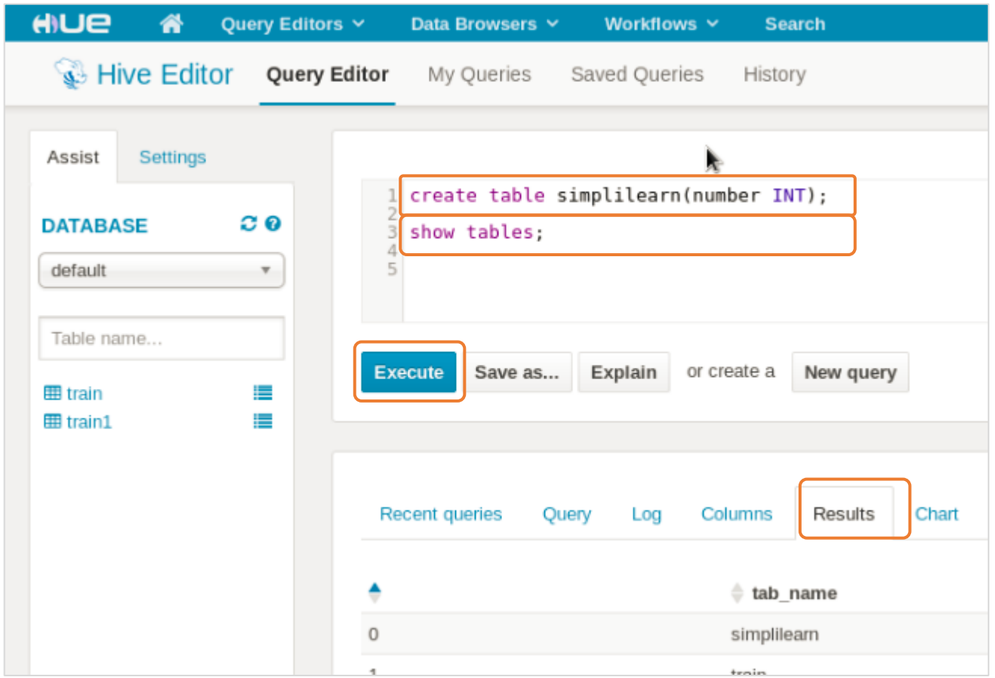
A semicolon (;) terminates the SQL commands.

training@localhost:~

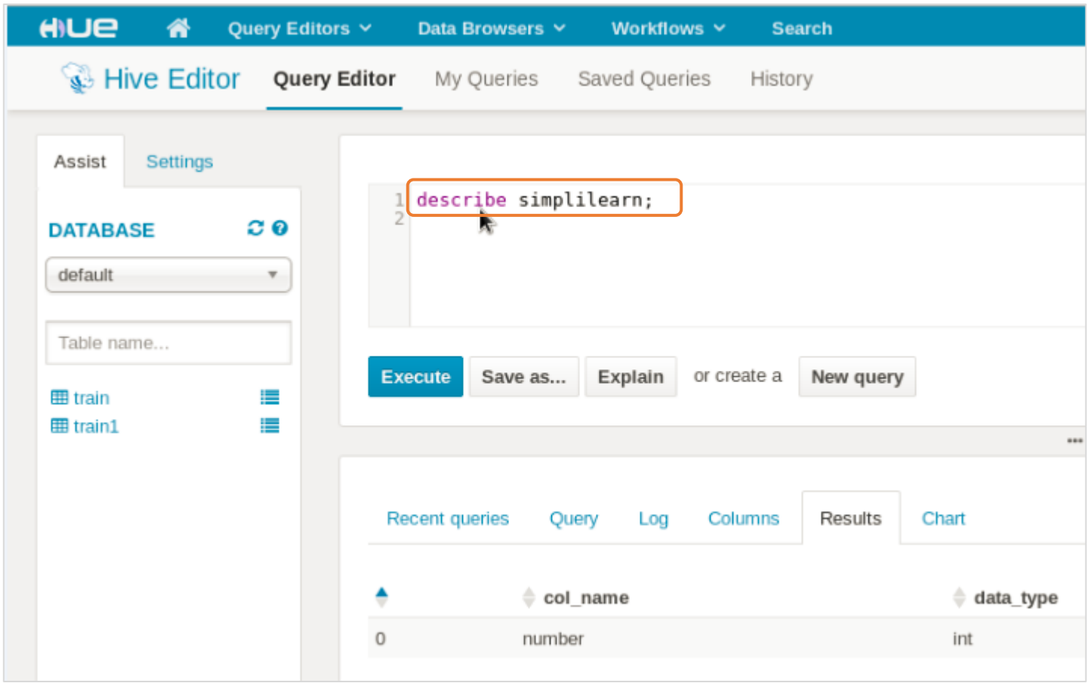
```
Hive> select * from device
    > LIMIT 5;
OK
1        2008-10-21 00:00:00       Sorrento F00L     phone
2        2010-04-19 00:00:00       Titanic 2100      phone
3        2011-02-18 00:00:00       MeeToo 3.0        phone
4        2011-09-21 00:00:00       MeeToo 3.1        phone
5        2008-10-21 00:00:00       iFruit 1 phone
Time taken: 0.296 seconds, Fetched: 5 row(s)
```

Hive Editors in Hue

Hue is an open-source Hadoop web user interface.



The creation of a table Simplilearn



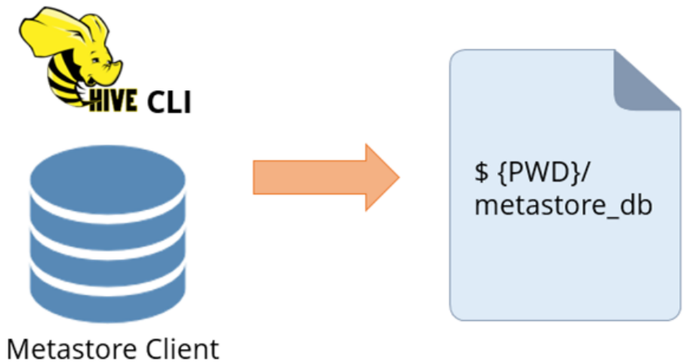
The description Simplilearn command

DATA AND
ARTIFICIAL INTELLIGENCE

Hive Metastore

What Is Hive Metastore?

The metastore is the component that stores the system catalog, which contains metadata about tables, columns, and partitions.



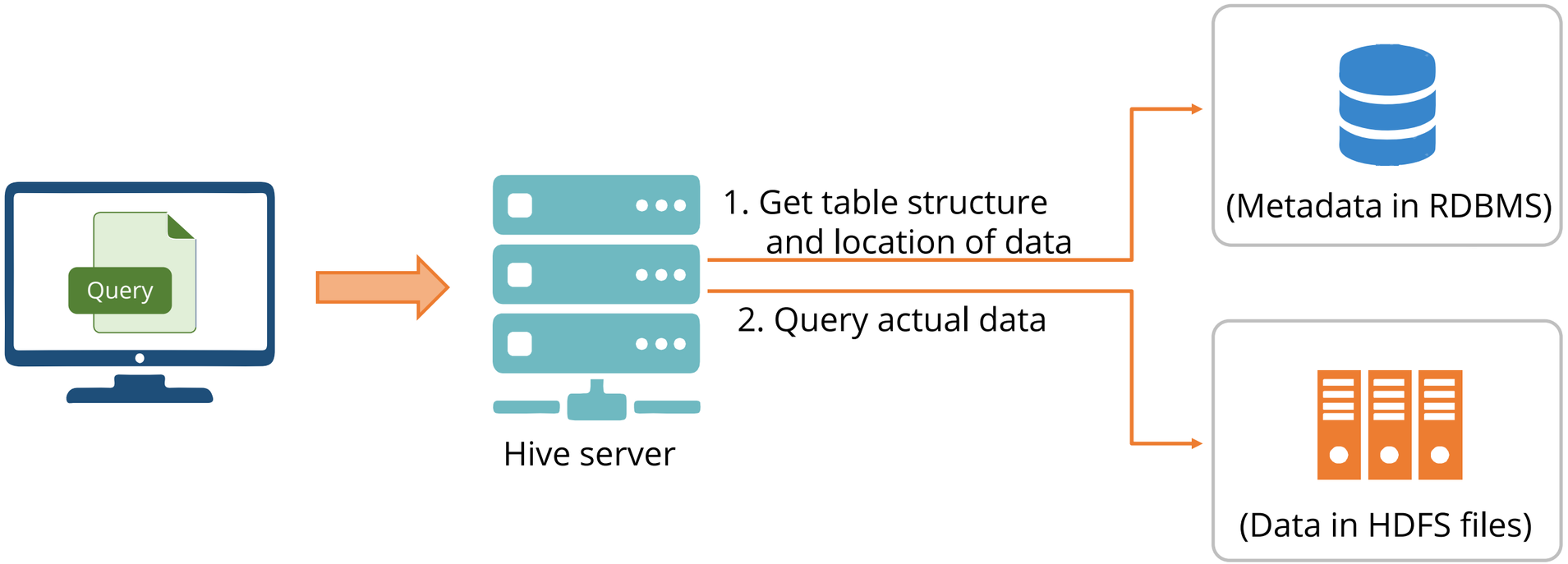
What Is Hive Metastore?

The metastore is the component that stores the system catalog, which contains metadata about tables, columns, and partitions.

Parameter	Description	Example
Javax.jdo.option.ConnectionURL	JDBC connection URL along with database name containing metadata	jdbc:derby;;databaseName=metastore_db;create=true
Javax.jdo.option.ConnectionDriverName	JDBC driver name, embedded Derby for single-user mode	Org.apache.derby.jdbc.EmbeddedDriver
Javax.jdo.option.ConnectionUserName	Username for Derby database	APP
Javax.jdo.option.ConnectionPassword	Password	mine

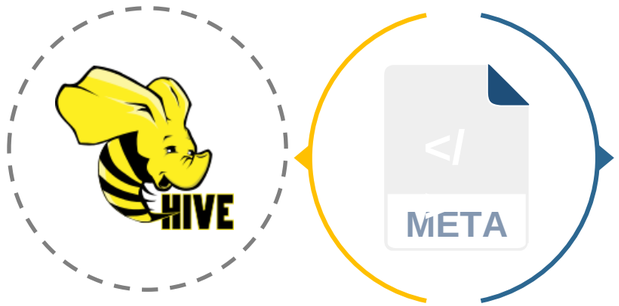
Use of Metastore in Hive

Hive uses metastore to get table structure and location of data. The server queries actual data which is stored in HDFS.



Managing Data with Hive

Hive uses the metastore service to store metadata for Hive tables.



- A table is an HDFS directory containing zero or more files.
- Table supports many formats for data storage and retrieval.
- The metadata produced in an RDBMS, such as MySQL, is stored in the metastore.
- Hive tables are stored in HDFS, and the relevant metadata is stored in the metastore

Default path: **/user/hive/warehouse/<*table_name*>**

Data Warehouse Directory Structure

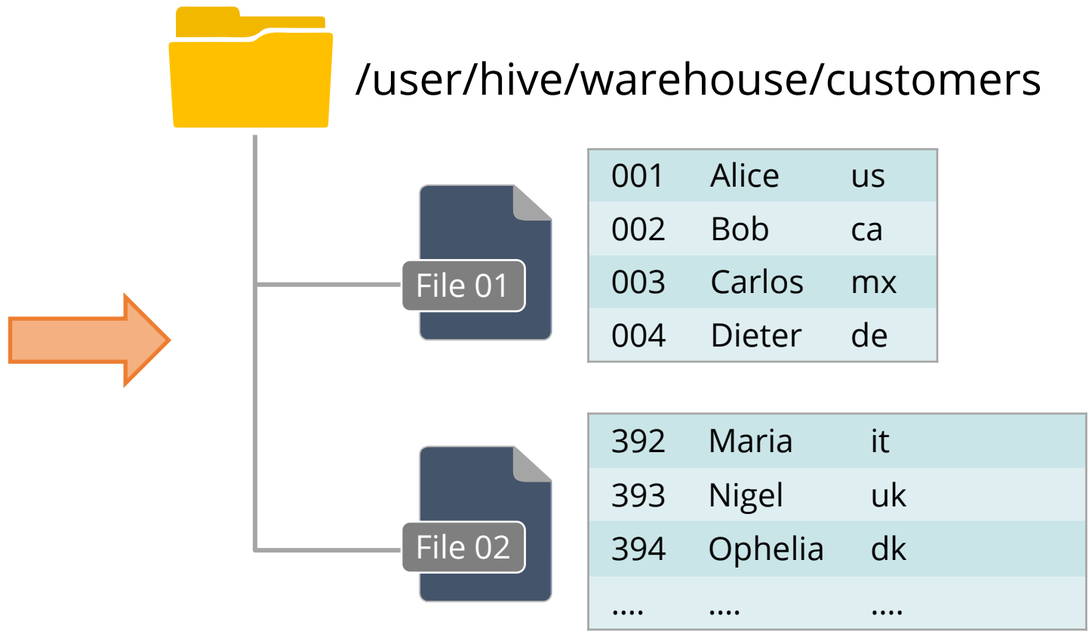
Each table is a directory within the default location having one or more files

By default, all data gets stored in:

/user/hive/warehouse

Customers table

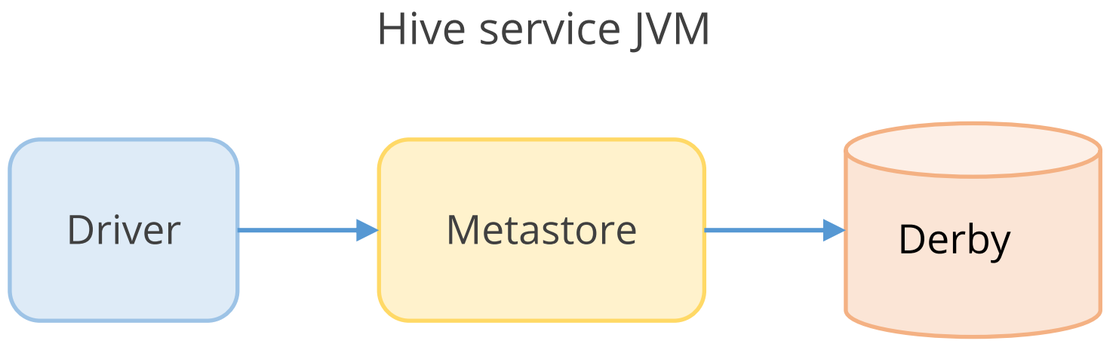
customer_id	name	country
001	Alice	us
002	Bob	ca
003	Carlos	mx
....	....	....
392	Maria	it
393	Nigel	uk
394	Ophelia	dk
....	....	....



Hive Metastore Modes

Embedded

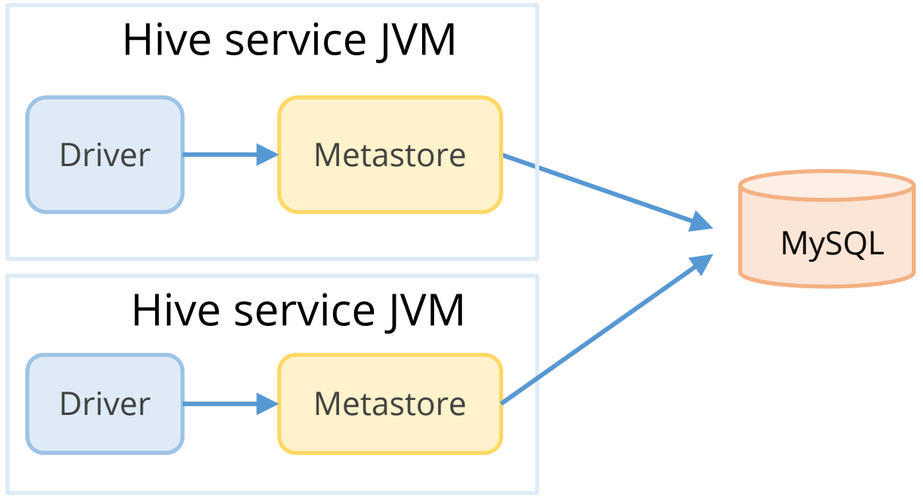
- The metastore and hive services share the same JVM and use the default embedded Derby DB.
- Only one session at a time is allowed.



Hive Metastore Modes

Local

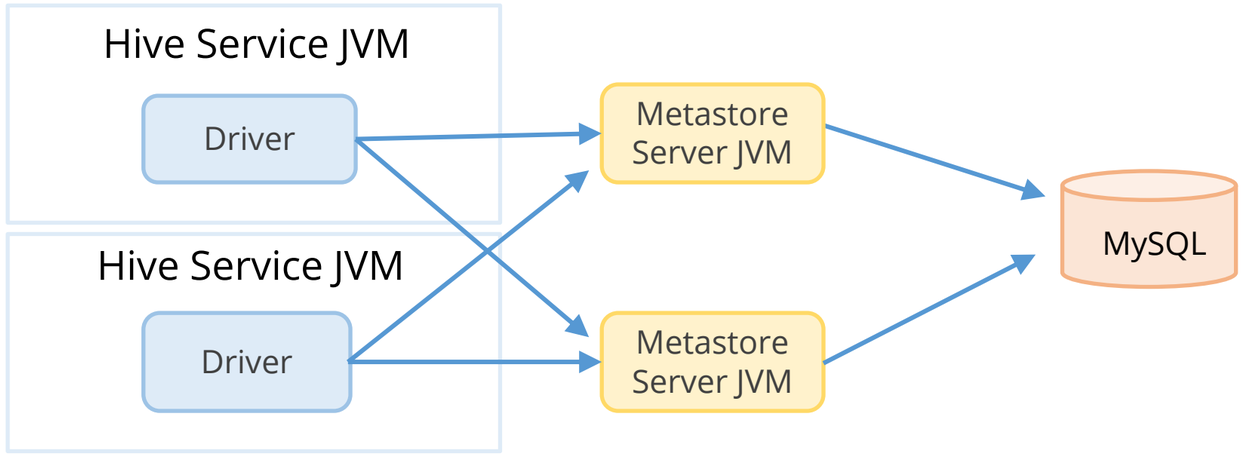
- Many users can have many sessions active at the same time, and instead of Derby, any JDBC compatible DB, such as MySQL or PostgreSQL, can be utilized.
- Metastore operates in the same JVM as Hive.



Hive Metastore Modes

Remote

- The metastore runs in a separate JVM than the Hive service.
- It is usually accessible through the hive.metastore.uris.

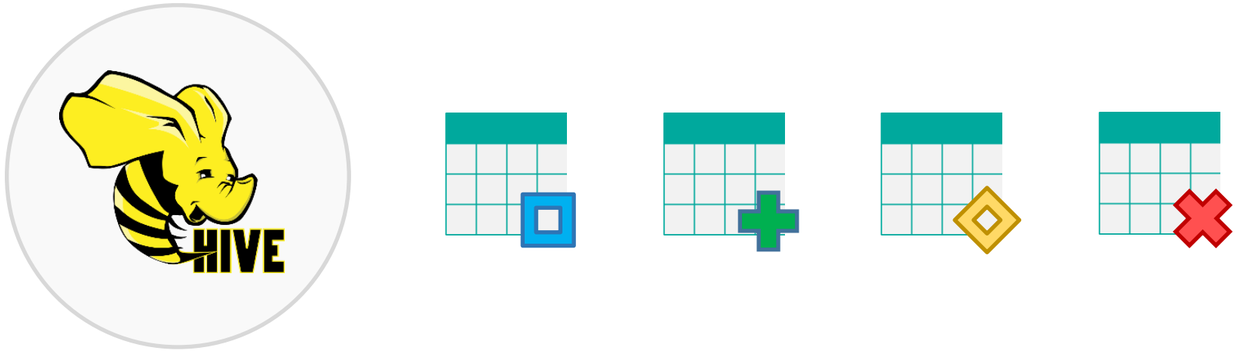


DATA AND
ARTIFICIAL INTELLIGENCE

Hive DDL and DML

Defining Database

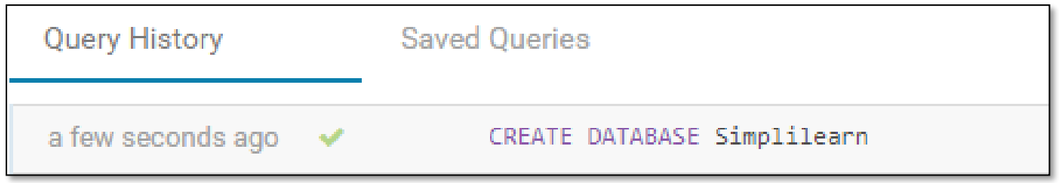
HiveQL's DDL (Data Definition Language) constructs and maintains databases and tables. They are similar to traditional SQL DDL.



Creating a Database

```
training@localhost:~
CREATE DATABASE Simplilearn;

CREATE DATABASE IF NOT EXISTS Simplilearn;
```

Output:



- To create a new database:

  CREATE DATABASE <dbname>;

- To avoid errors if the database Simplilearn already exists:

  CREATE DATABASE IF NOT EXISTS <dbname>;

Deleting a Database

- The process of removing a database is similar to the process of creating one.
  - Instead of CREATE, use DROP

    DROP DATABASE <dbname>;
  - In case the database already exists, use

    DROP DATABASE IF EXISTS <dbname>;

This might remove data in HDFS.

Deleting a Database

The DROP DATABASE command is used to delete an existing database.

training@localhost:~

```
DROP DATABASE Simplilearn;
```

Output:

Creating New Table

training@localhost:~

```
Syntax:

CREATE TABLE tablename( colname DATATYPE, ….)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY char

STOREDAS {TEXTFILE|SEQUENCEFILE|…}
```

Syntax to create a new table

- Syntax creates a subdirectory in HDFS's warehouse directory for the database.
  - Default database

    /user/hive/warehouse/tablename
  - Named database

    /user/hive/warehouse/dbname.db/tablename

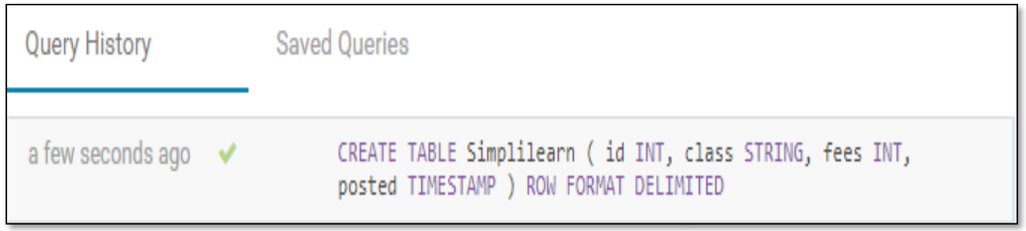
Table Creation: Example

The following example shows how to create a new table named *Simplilearn.*

training@localhost:~

```
CREATE TABLE Simplilearn (
id INT,
class STRING,
fees INT,
posted TIMESTAMP
)
ROW FORMAT DELIMITED;
```

Output:



DATA AND
ARTIFICIAL INTELLIGENCE

Hive: Data Types

Data Types in Hive

Hive has the following three data types:

Data Types

Primitive types

- Integers: TINYINT, SMALLINT, INT, and BIGINT
- Boolean: BOOLEAN
- Floating point numbers: FLOAT and DOUBLE
- String: STRING

Complex types

- Structs: {a INT; b INT}
- Maps: M['group']
- Arrays: ['a', 'b', 'c'], A[1] returns 'b‘

User-defined types

- Structures with attributes

Changing Table Data Location

training@localhost:~

```
CREATE EXTERNAL TABLE Simplilearn (
id INT,
class STRING,
fees INT,
posted TIMESTAMP
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY “,”
LOCATION ‘/user/username/Simplilearn';
```

- Table data is saved in the default warehouse location by default:

  user/hive/warehouse

- Specify the directory in HDFS where you want your data to be stored using LOCATION

Output:

2 minutes ago ✔

```
CREATE EXTERNAL TABLE SimplilearnTest1 ( id INT, class STRING, fees INT,
posted TIMESTAMP ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ","
LOCATION '/user/testdemomay1301mailinator/Simplilearn'
```

External Managed Table

training@localhost:~

```
CREATE EXTERNAL TABLE Simplilearn (
id INT,
class STRING,
fees INT,
posted TIMESTAMP
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY “,”
LOCATION '/bda/Simplilearn';
```

- Tables are "managed" or "internal“ by default.
- One can use EXTERNAL to create an external managed table.
- Dropping an external table removes only its metadata.

Output:

Validation of Data

- Hive does not verify data when it is being inserted.
- In Hive, files are simply placed into position, which speeds up the loading of data into tables.
- When queries are performed, file format errors are identified.

NULL signifes missing data.

Loading of Data

training@localhost:~

```
LOAD DATA INPATH '/simplilearn/data'

OVERWRITE INTO TABLE simplilearn;
```

- Data can be moved from the HDFS file directly to the Hive table by using the following path:

  hdfs dfs -mv /simplilearn/data /user/hive/warehouse/simplilearn/

- Data can be loaded by using the syntax:

LOAD DATA INPATH '/simplilearn/data'

OVERWRITE INTO TABLE simplilearn;

Output:

```
a few seconds ago    LOAD DATA INPATH '/user/testdemomay1301mailinator/hivedemo1'
                     OVERWRITE INTO TABLE SimplilearnTest1
```

Loading Data from RDBMS

training@localhost:~

```
sqoop import \
        -connect jdbc:mysql://localhost/simplilearn \
        -username training \
        -password training \
        -fields-terminated-by '\t' \
        -table employees \
        -hive-import
```

- Sqoop provides support for importing data into Hive
- Using the hive-import option in Sqoop, users can:
  - create a table in Hive metastore
  - import data from the RDBMS to the table's directory in HDFS

hive-import creates a table accessible in Hive.

What Is HCatalog?

HCatalog is a Hive sub-project that provides access to the metastore.

- It allows defining tables using HiveQL DDL syntax
- It is accessible through the command line and REST API.
- It accesses tables created through HCatalog from Hive, MapReduce, Pig, and other tools.

What Is HCatalog?

The following syntax can create the table:

training@localhost:~

```
CREATE EXTERNAL TABLE simplilearnTest2 (
year INT,
month INT,
day INT,
carrier STRING,
origin STRING,
dest STRING,
depdelay INT,
arrdelay INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED by '9'
STORED AS TEXTFILE
LOCATION
'/user/testdemomay1301mailinator/Simplilearn';
```

Output:

a few seconds ago ✔

```
CREATE EXTERNAL TABLE simplilearnTest2 ( year INT, month INT, day INT,
carrier STRING, origin STRING, dest STRING, depdelay INT, arrdelay INT )
ROW FORMAT DELIMITED  FIELDS TERMINATED by '9' STORED AS TEXTFILE
LOCATION '/user/testdemomay1301mailinator/Simplilearn'
```

HCatalog Command

Use the following command to initialize the HCatalog command line:

training@localhost:~

```
cd $HCAT_HOME/bin ./hcat
```

Output:

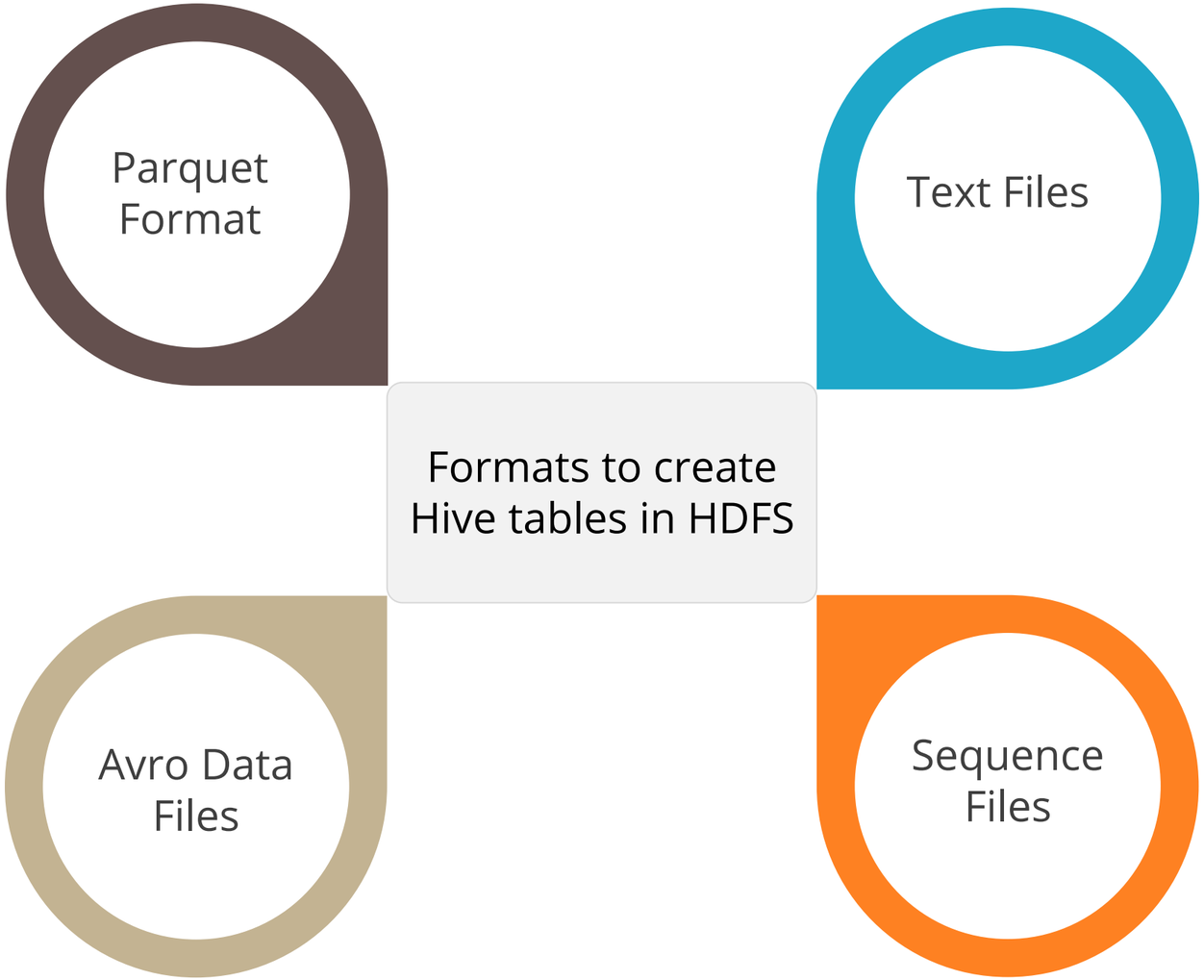
```
[testdemomay1301mailinator@bdh-cluster2-edgenode10 ~]$ cd $HCAT_HOME/bin ./hcat
[testdemomay1301mailinator@bdh-cluster2-edgenode10 bin]$
```

DATA AND
ARTIFICIAL INTELLIGENCE

File Format Types

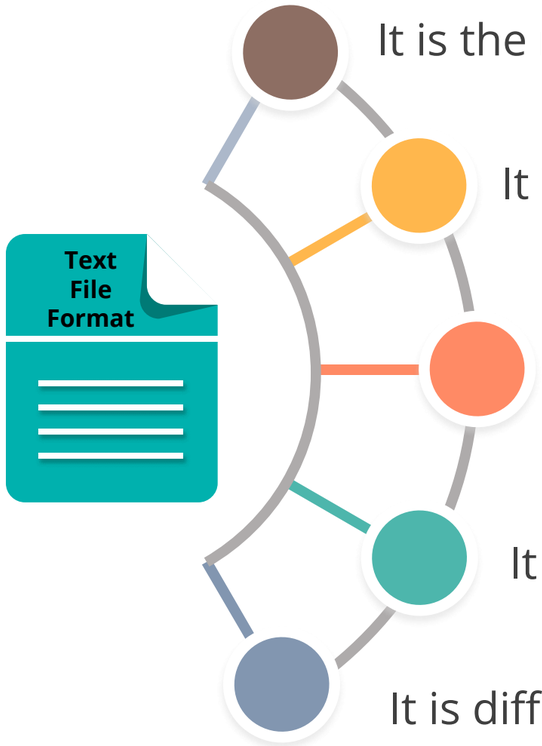
File Format Types

There are four types of file formats:



Text File Format

The basics of the text file format are listed below:



- It is the most basic and human-readable file.
- It is readable and writable in any programming language.
- A comma or a tab delimits the list.
- It consumes more space to store numeric values as string.
- It is difficult to represent binary data.

Sequence File Format

The specifications of the sequence file format are:

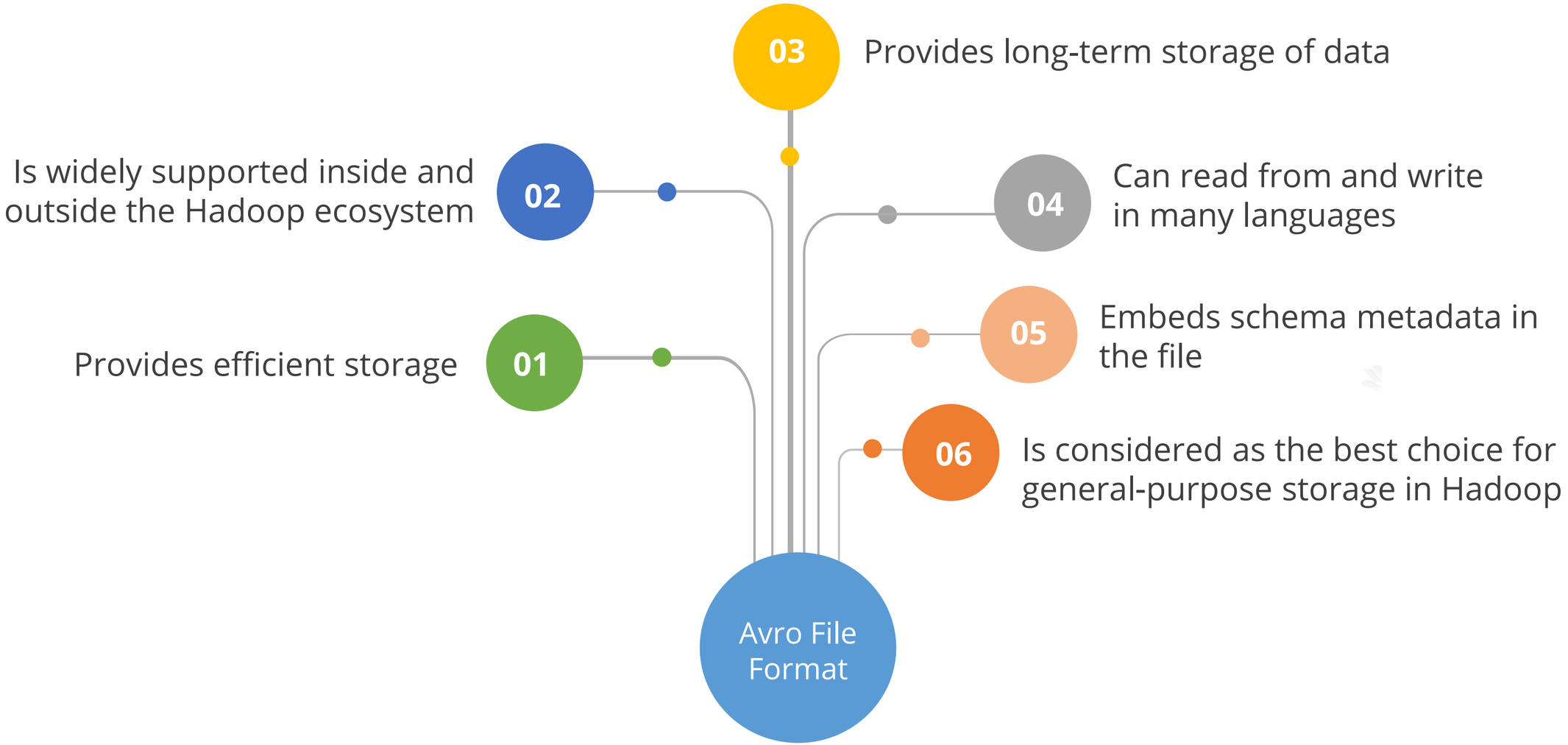
Stores key-value pairs in a binary container format

Is more efficient than a text file

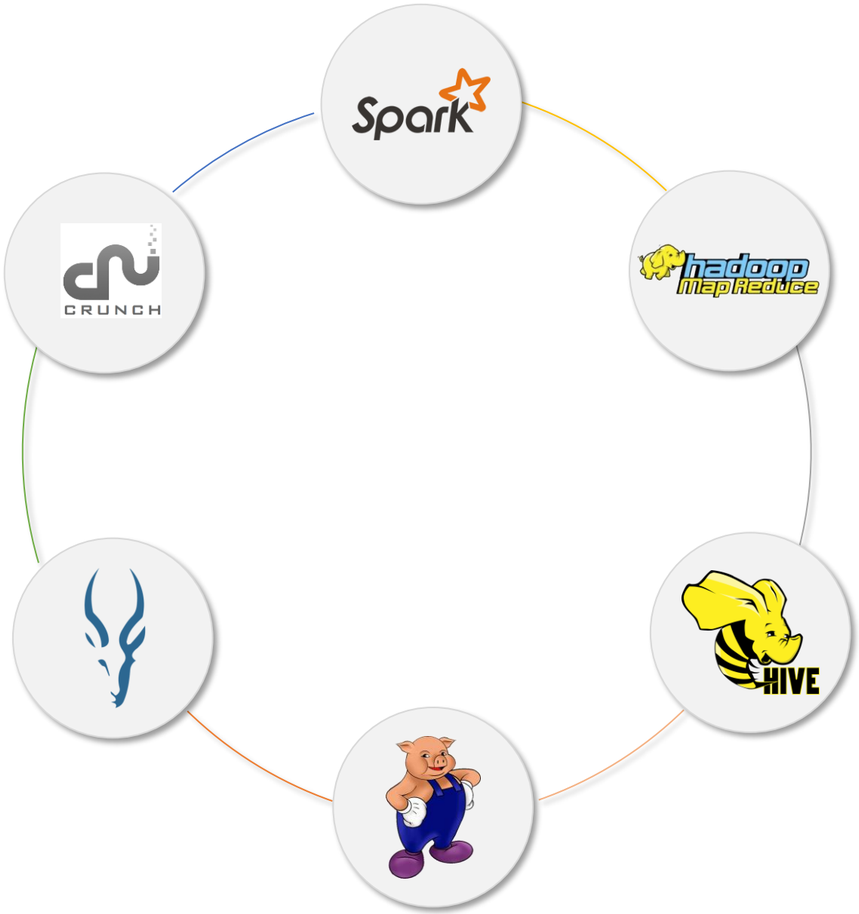
Is not human-readable

Avro File Format

The Avro file format has the following features:



Parquet File Format



- It is a columnar format developed by Cloudera and Twitter.
- It uses advanced optimizations described in Google's Dremel paper.
- It is considered the most efficient for adding multiple records at a time.

DATA AND
ARTIFICIAL INTELLIGENCE

Hive: Data Serialization

What Is Data Serialization?

Data serialization is a way to represent data in the storage memory as a series of bytes.

How do you serialize the number 123456789?

It can be serialized as 4 bytes when stored as a Java int and 9 bytes when stored as a Java String.

Data Serialization Framework

- It is efficient data serialization framework
- It is widely supported throughout Hadoop and its ecosystem
- It supports Remote Procedure Calls (RPC)
- It offers compatibility

Data Types Supported in Avro

The data types supported by Avro are shown in the table below:

Name	Description
null	An absence of a value
boolean	A binary
int	32-bit signed integer
long	64-bit signed integer
float	Single-precision floating point value
double	Double-precision floating point value
bytes	Sequence of 8-bit unsigned bytes
string	A sequence of Unicode characters

Complex Data Types Supported in Avro Schemas

The complex data types supported by Avro schemas are shown in the table below:

Name	Description
record	A user-defined type composed of one or more named fields
enum	A specified set of values
array	Zero or more values of the same type
map	Set of key-value pairs; key is string while value is of specified type
union	Exactly one value matching a specified set of types
fixed	A fixed number of 8-bit unsigned bytes

Hive Table, Avro Schema, and Avro Operations

The following code runs Hive Table, Avro Schema, and Avro Operations:

training@localhost:~

```
(id INT, name STRING, title STRING)

{"namespace":"com.simplilearn",
"type":"record",
"name":"orders",
"fields":[
{"name":"id", "type":"int"},
{"name":"name", "type":"string"},
{"name":"title", "type":"string"}]
}

{"namespace":"com.simplilearn",
"type":"record",
"name":"orders",
"fields":[
{"name":"id", "type":"int"},
{"name":"name", "type":"string", "default":"simplilearn"},
{"name":"title", "type":"string","default":"bigdata"}]
}
```

Hive Table - CREATE TABLE orders

Avro Schema

Avro Operations

Create New Table with Parquet

Use the following command to initialize the HCatalog command line:

training@localhost:~

```
[training@localhost simplilearn]$ hive

Logging initialized using configuration in jar:file:/opt/cloudera/parcels/CDH-6.3.2-
1.cdh6.3.2.p0.1605554/jars/hive-common-2.1.1-cdh6.3.2.jar!/hive-log4j2.properties Async: false
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> CREATE TABLE order_parquet (
      id INT,
      number INT)
      STORED AS PARQUET;
OK
INFO  : Completed executing command(queryId=hive_20220613105145_03155e79-5ae5-4a74-9f52-
a38d659596df); Time taken: 0.279 seconds
INFO  : OK
```

Reading Parquet Files Using Tools

Use the following command to initialize the HCatalog command line:

training@localhost:~

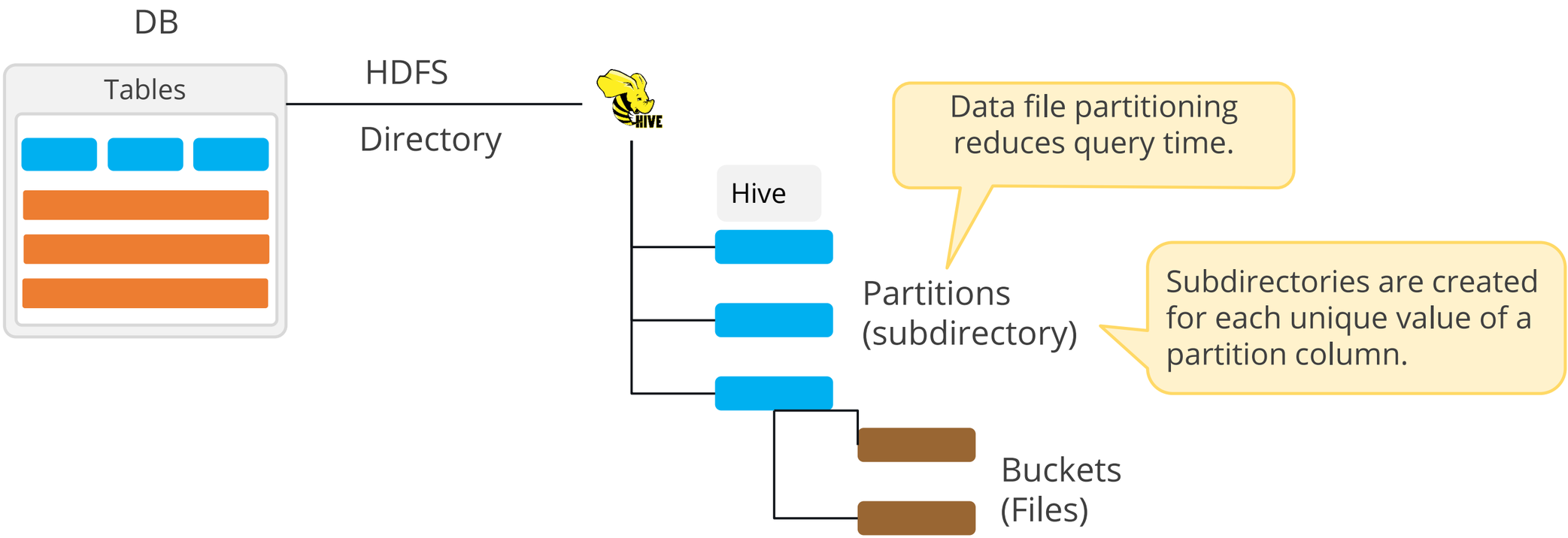
```
[training@localhost simplilearn]S parquet-tools head orders.parquet
acct_num = 97338
acct_create_dt = 1380082589000
first_name = Joshua
last_name = Pierce
address = 2289 Emeral Dreams Drive
city = Phoenix
state = AZ
zipcode = 85304
phone_number = 9283693115
created = 1395174771000
modified = 1395174771080

acct_num = 97331
acct_create_dt = 1362988704080
first_name = Susan
last_name = Newton
address = 2356 Nicholas Street
city = Ely
state = NV
zipcode = 89398
phone_number = 7755369146
created = 1395174771888
modified = 1395174771080
```

DATA AND
ARTIFICIAL INTELLIGENCE

Hive Optimization: Partitioning, Bucketing, and Skewing

Data Storage

All files in a data set are stored in a single Hadoop Distributed File System, or HDFS directory.



Example: Non-Partitioned Table

training@localhost:~

```
[training@localhost - ]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> CREATE EXTERNAL TABLE accounts
> cust_id INT,
> fname STRING,
> lname STRING,
> address STRING,
> city STRING,
> state STRING,
> zipcode STRING)
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY ‘,’
> LOCATION '/simplilearn/accounts_by_state’;
```

Example: Non-Partitioned Table

The customer details are required to be partitioned by the state for fast retrieval of subset data pertaining to the customer category.

- Hive will need to read all the files in a table's data directory.
- It can be a very slow and expensive process, especially when the tables are large.

Example of a Partitioned Table

training@localhost:~

```
[training@localhost - ]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> CREATE EXTERNAL TABLE accounts
> cust_id INT,
> fname STRING,
> lname STRING,
> address STRING,
> city STRING,
> state STRING,
> zipcode STRING)
> PARTITIONED BY (state STRING)
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY ‘,’
> LOCATION '/simplilearn/accounts_by_state’;
```

A partition column is a “virtual column” where data is not actually stored in the file.

Example of a Partitioned Table

Partitions are horizontal slices of data that allow larger sets of data to be separated into more manageable chunks.

Partitions are horizontal data slices that allow enormous collections of data which may utilize hive partitioning to store data in various files according to state.

Data Insertion

Data insertion into partitioned tables can be done in two ways or modes:

Static partitioning

Dynamic partitioning

Static Partitioning

training@localhost:~

```
[training@localhost -]$ hive
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> ALTER TABLE accounts
    > ADD PARTITION (accounts_date='2016-30-02');
```

Add a partition for each new day of account data

- Input data files individually into a partition table
- Create new partitions; define them using the ADD PARTITION clause
- While loading data, specify the partition to store data in; specify partition column value
- Add a partition in the table; move the file into the partition of the table

Dynamic Partitioning

training@localhost:~

```
[training@localhost -]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> INSERT OVERWRITE TABLE accounts_by_state
    > PARTITION(state)
    > SELECT custid, fname, inane, address,
    > city, zipcode, state FROM accounts;
```

With a large amount of data stored in a table, a dynamic partition is suitable

- Partitions get created automatically at load times
- New partitions can be created dynamically from existing data
- Partitions are automatically created based on the value of the last column
- Existing partitions will be overwritten by the OVERWRITE keyword

Dynamic Partitioning in Hive

By default, dynamic partitioning is disabled in Hive to prevent accidental partition creation.

training@localhost:~

```
[training@localhost -]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> INSERT OVERWRITE TABLE accounts_by_state
    > PARTITION(state)
    > SELECT custid, fname, inane, address,
    > city, zipcode, state FROM accounts;
```

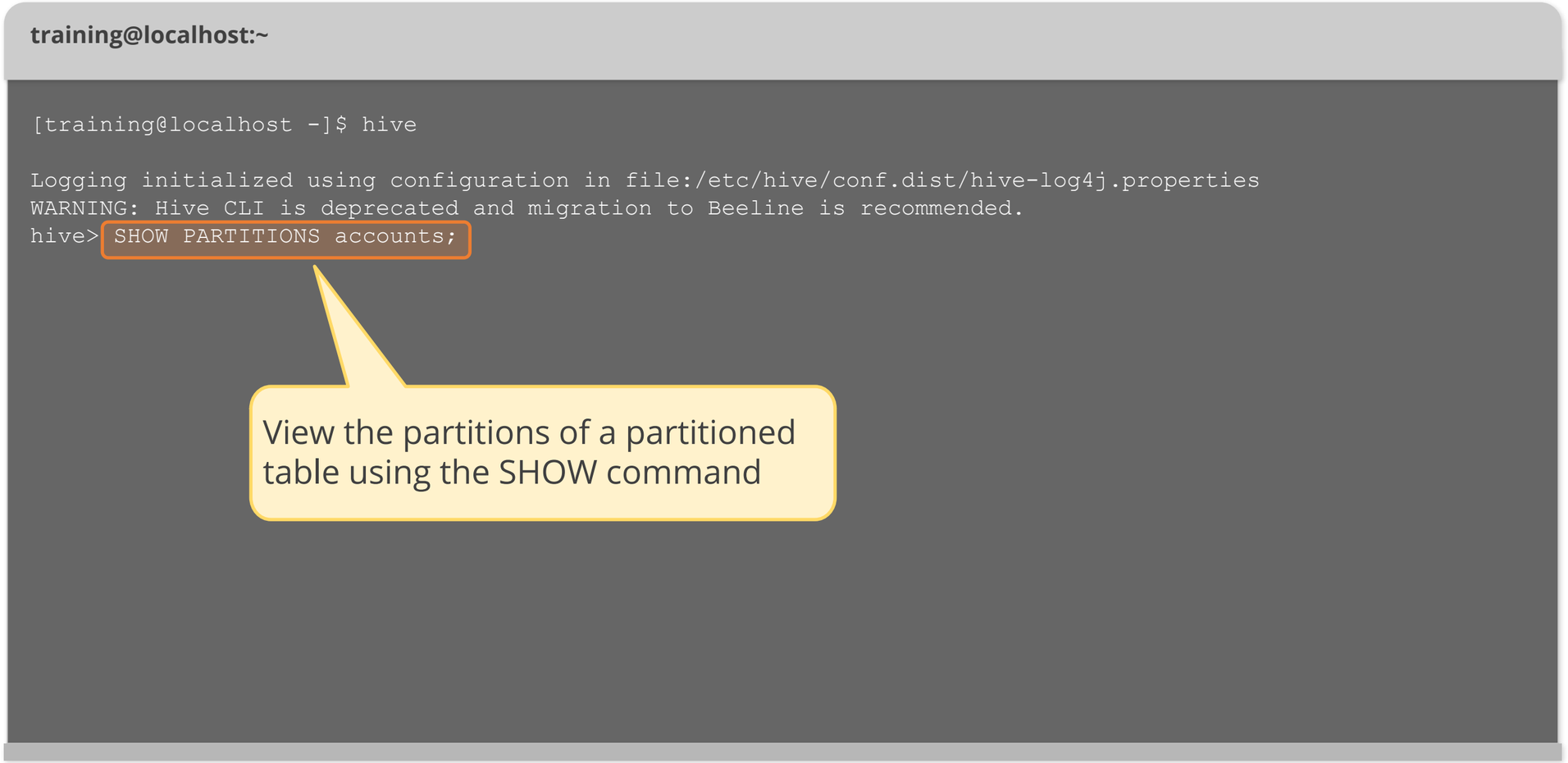
Enable the following settings to use dynamic partitioning:

SET hive.exec.dynamic.partition=true;

SET hive.exec.dynamic.partition.mode=nonstrict;

Viewing Partitions

Hive partitioned tables support the following commands to view partitions:



Deleting Partitions

Hive partitioned tables support the following commands to delete partitions:

training@localhost:~

```
[training@localhost -]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
hive> ALTER TABLE accounts
    > DROP PARTITION (accounts_date='2016-30-02');
```

Use ALTER command:

- To delete partitions
- To add or change partitions

When to Use Partitioning

Following are the instances when users need to use partitioning for tables:

When reading the entire data set takes too long

When queries almost always filter on the partition columns

When there are a reasonable number of different values for partition columns

When Not to Use Partitioning

Users should avoid using partitioning in the following instances:

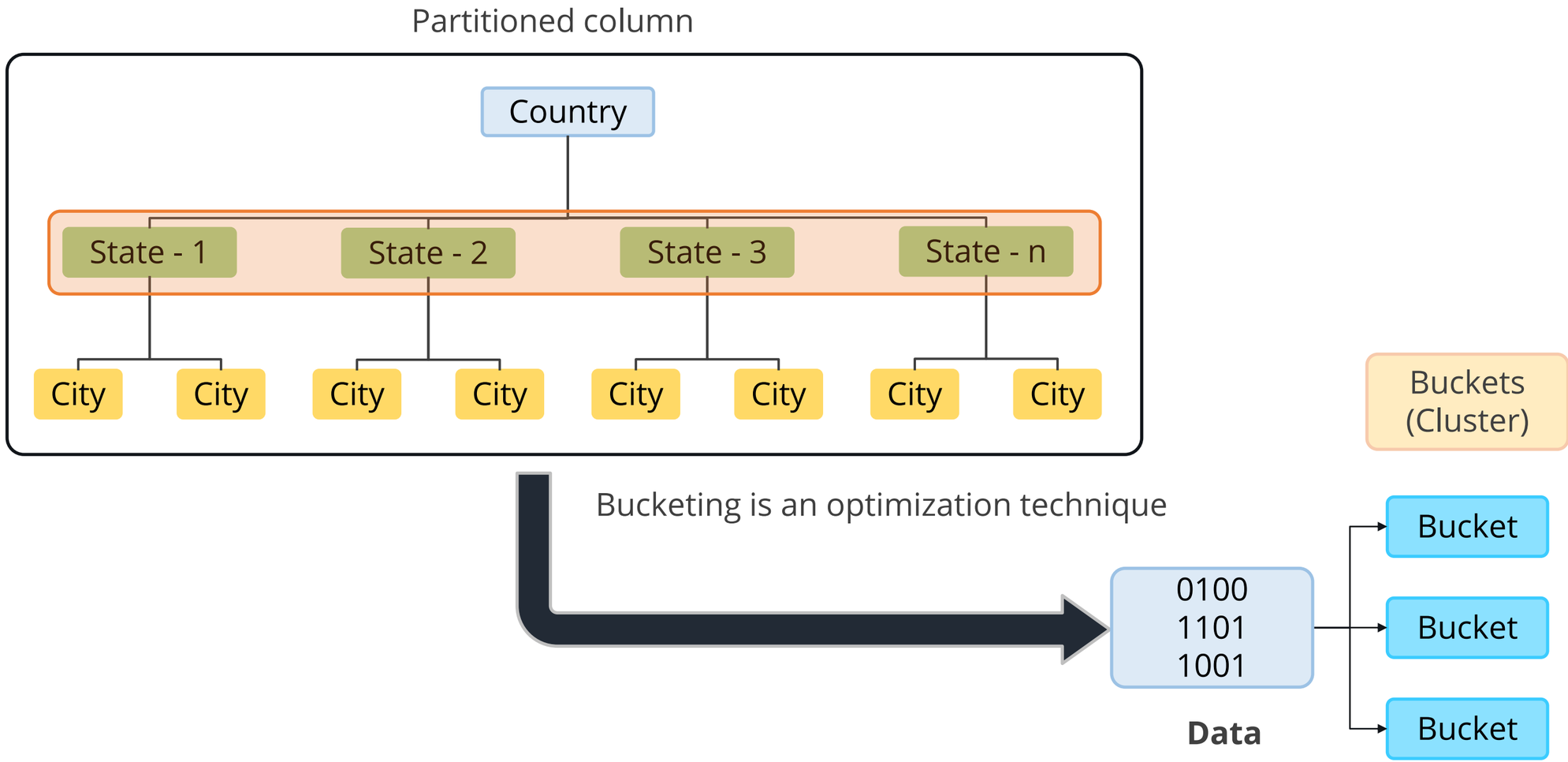
When columns have many unique rows

When creating a dynamic partition, as can lead to a high number of partitions

When the partition is less than 20k

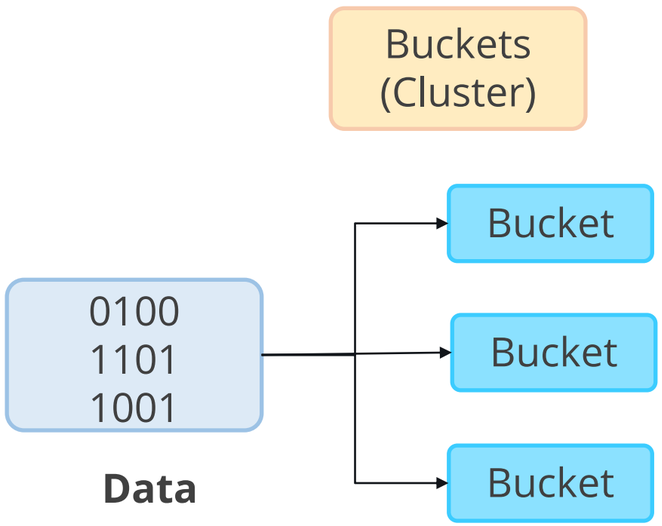
Bucketing in Hive

Bucketing separates data on a field with high cardinality (the number of potential values a field might have.)



What Are Buckets Used For?

Buckets distribute the data load into a user-defined set of clusters by calculating the hash code of the key mentioned in the query.



The syntax for creating a bucketed table is:

```
CREATE TABLE page_views( user_id INT, session_id BIGINT, url STRING)
PARTITIONED BY (day INT)
CLUSTERED BY (user_id) INTO 100;
```

The processor will first calculate the hash number of the user_id in the query and will look for only that bucket.

Skewed Tables

Data skew generally refers to a non-uniform distribution of a data set. A non-uniform distribution may influence the system if the appropriate execution plan is not chosen based on the data values.

training@localhost:~

```
CREATE TABLE Customers (
    id int,
    username string,
    zip int
)
SKEWED BY (zip) ON (57701, 57702)
STORED as DIRECTORIES;
```

Output:

Query History | Saved Queries

2 minutes ago ✔

```
CREATE TABLE Customersz ( id int, username string, zip int )
SKEWED BY (zip) ON (57701, 57702) STORED as DIRECTORIES
```

Hive Cost-Based Optimization (CBO)

- The ***Cost-Based Optimization (CBO)*** engine uses statistics within Hive tables to produce optimal query plans.
- Two types of stats are used for optimization:
  - Table stats
  - Column stats
- It uses an open-source framework called ***Calcite.***
- To use CBO, one needs to:
  - Analyze the table and relevant columns
  - Set the appropriate properties

Hive Optimization Tips

01 Divide data into files that can be trimmed using partitions, buckets, and skews

02 Make use of the ORC (Optimized Row Columnar) file format

03 Perform sorting and bucketing based on common join keys

Hive Optimization Tips

04 Make use of map (broadcast) joins whenever feasible

05 Increase the replication factor for hot data (which reduces latency)

06 Use Tez as it is advantageous

Hive Query Tunings

In addition, set the important join and bucket properties to **true** in **hive-site.xml** or by using the **set** command.

- mapreduce.input.fileinputformat.split.maxsize
- mapreduce.input.fileinputformat.split.minsize
- mapreduce.tasks.io.sort.mb

Sorting Data

Hive has two following two sorting clauses:

Order by

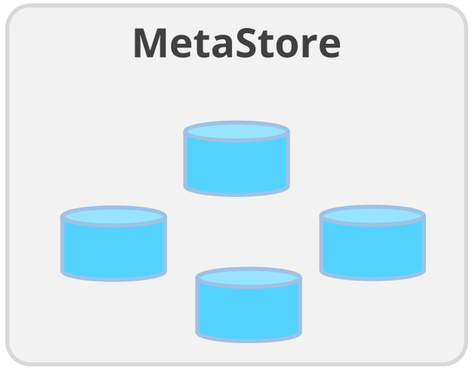
- It ensures global ordering but achieves it by passing all data via a single reducer.
- As a result, order by gets one sorted file as an output.

Sort by

- It orders data to each of the N reducers, but each reducer may receive data from many sources.
- As a result, users sort by having N or more sorted files with overlapping ranges.

Hive Query Language: Introduction

HiveQL is a SQL-like query language for Hive to process and analyze structured data in a metastore.

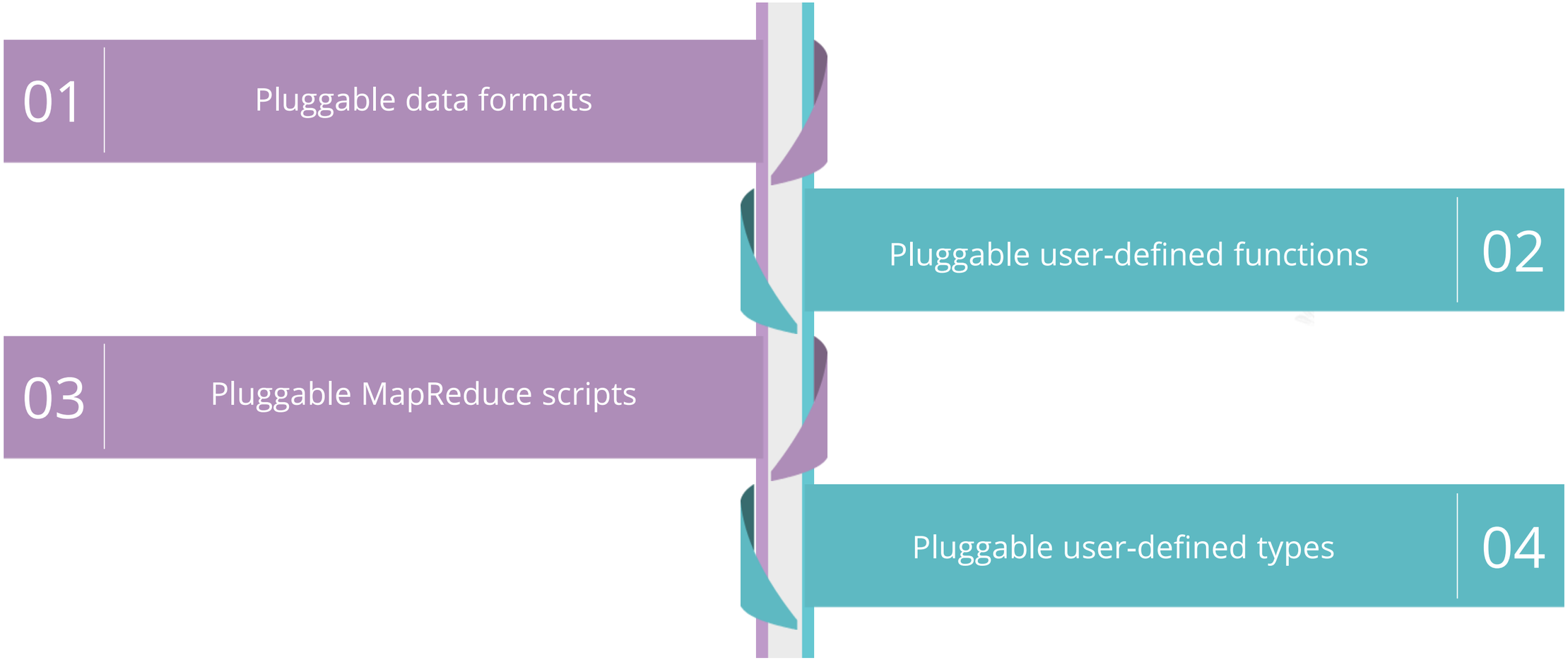


training@localhost:~

```
SELECT
dt,
COUNT (DISTINCT (user_id))
FROM events
GROUP BY dt;
```

HiveQL: Extensibility

An important principle of HiveQL is its extensibility. HiveQL can be extended in multiple ways:



DATA AND
ARTIFICIAL INTELLIGENCE

Hive Analytics: UDF and UDAF

User-Defined Function

UDFs extend the functionality of Hive, with a function written in Java, that can be evaluated in HiveQL statements.

- All UDFs extend the Hive UDF class. After that, a UDF subclass implements one or more methods named ‘evaluate.’
- Evaluation should never be a void method. It can return a null value if required.

Code for Extending UDF

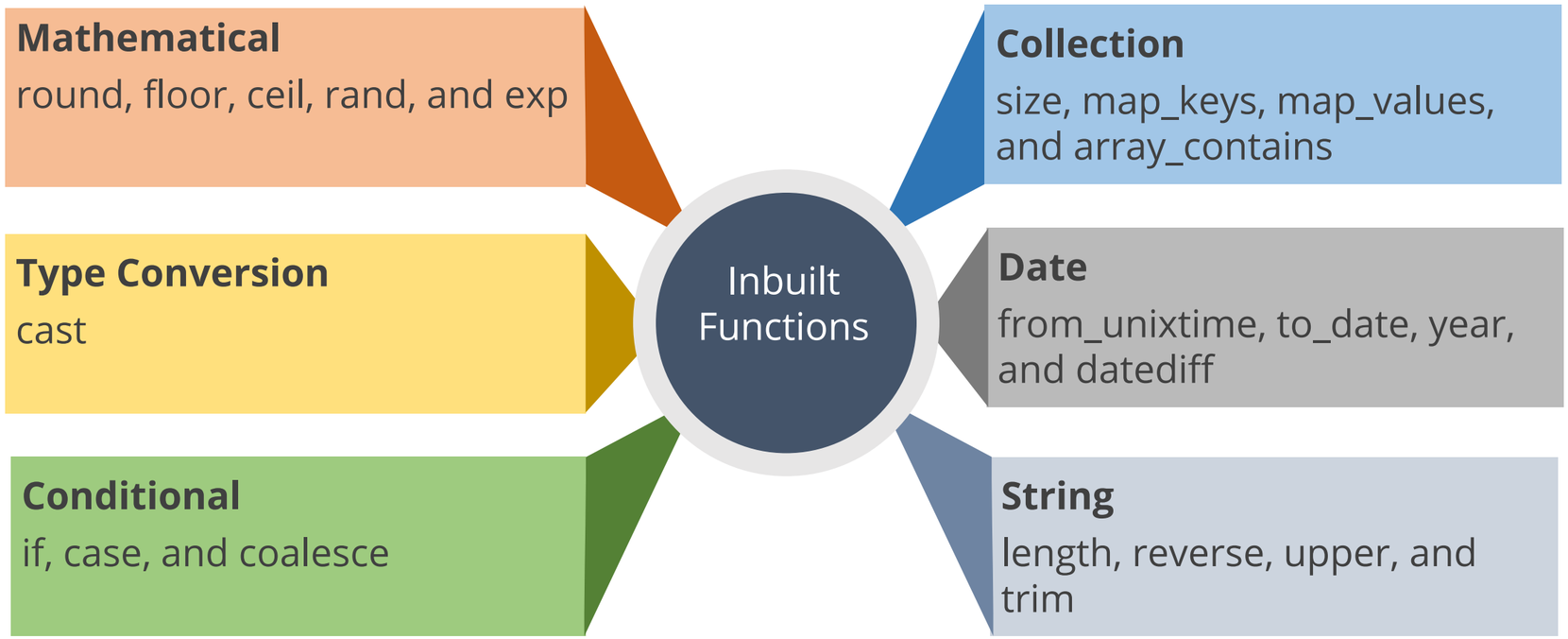
Here is a code that one can use to extend the UDF:

training@localhost:~

```
package com.example.hive.udf;

import org.apache.hadoop.hive.ql.exec.UDF;
import org.apache.hadoop.io.Text;

public final class Lower extends UDF {
  public Text evaluate(final Text s) {
    if (s == null) { return null; }
    return new Text(s.toString().toLowerCase());
  }
}
```

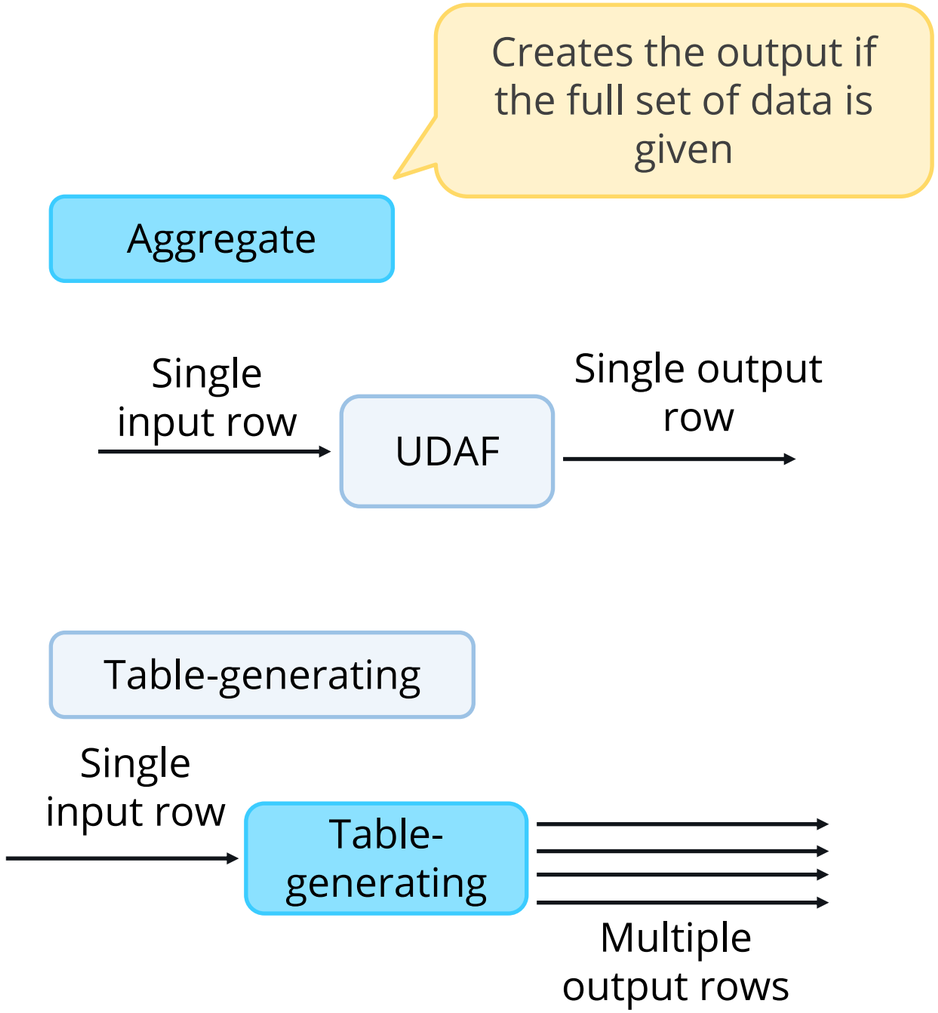
Built-in Functions of Hive

The UDF is created when the functions are written in Java scripts. Hive also provides some inbuilt functions that can be used to avoid having to create their own UDFs.



Other Functions of Hive

Aggregate and table generating functions of Hive function are:



Other Functions of Hive

The syntax and lateral view of Hive functions are:

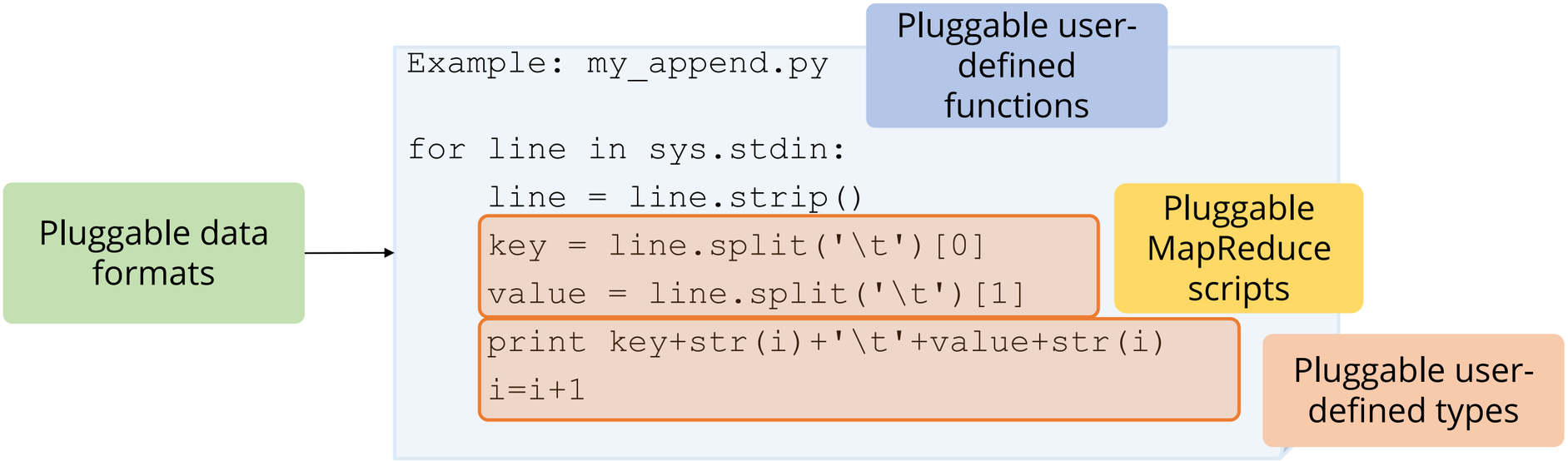
String pageid	Array<int> adid_list
"front_page"	[1,2,3]
"contact_page"	[3, 4, 5]

```
SELECT pageid, adid FROM pageAds
LATERAL VIEW explode(adid_list) adTable
AS adid;
```

String pageid	intadid
"front_page"	1
"front_page"	2
.....	.....

MapReduce Scripts

MapReduce scripts are written in scripting languages such as Python.



Using the function:

```
SELECT TRANSFORM (foo, bar) USING 'python ./my_append.py' FROM sample;
```

UDF/UDAF vs. MapReduce Scripts

The table below compares UDF/UDAF scripts with MapReduce scripts:

Attribute	UDF/UDAF	MapReduce scripts
Language	Java	Any language
1/1 input/output	Supported via UDF	Supported
n/1 input/output	Supported via UDAF	Supported
1/n input/output	Supported via UDTF	Supported
Speed	Faster (in same process)	Slower (spawns new process)

Industry Wide Use Cases

Some of the industry-wide use cases are as follows:

1. **Vodafone** installed around 600 Hadoop servers and is storing about 12 PB of data in Hive on top of Apache TEZ to speed digital transformation for its corporate clients.

2. **Wayfair** used Apache Hadoop to analyze data from their operational SQL, which saves data from apps in Hive. They were able to enhance inventory and demand forecasts to enable our suppliers to make better decisions and create more income faster.

Industry Wide Use Cases

Some of the industry-wide use cases are as follows:

Paytm, a global leader in digital payments with 17 million merchants and millions of transactions per day, used hive to provide more product branding recommendations.

3

4

Redfin is a full-service real estate agency with a tiny crew that receives millions of requests for rental inquiries per hour. They utilize AWS S3 for storage and EMR (Hive) to store real-time data in it, which is then processed with SparkSQL.

Industry Wide Use Cases

Some of the industry-wide use cases are as follows:

Nielsen, a global, independent measurement and analytics business, began storing 30 petabytes of data on AWS for ad-hoc analysis to determine what trends customers are seeing in advertisements.

Assisted Practice: Working with Hive Query Editor

Duration: 15 mins

Problem Scenario: In this demonstration, you will work with the Hive Query editor.

Objective: This assisted practice will help you will use the Hive Query editor to create a table in a directory.

Dataset Name: "drivers.csv“

Tasks to Perform:

Step 1: Upload the **“drivers.csv”** file into HDFS by logging in hue

Step 2: Open the Hive editor and create a database and a table in it

Step 3: Load data into Hive table using the LOAD command

Step 4: Verify data by executing the select command

Note: The solution to this assisted practice is provided under the course resources section.

Assisted Practice: Working with Hive Query Editor Using Metadata

Duration: 15 mins

Problem Scenario: In this demonstration, you will work with the Hive Query editor and external table.

Objective: This assisted practice will help you use Hive Query editor to create an EXTERNAL table where after dropping tables, data will remain intact, and metadata will be cleaned.

Dataset Name: "drivers.csv“

Tasks to Perform:

Step 1: Upload the **“drivers.csv”** file into HDFS by logging in hue

Step 2: Open the Hive editor and create a database and a table in it

Step 3: Load data into Hive table using the LOAD command

Step 4: Verify data by executing the select command

Step 5: Drop the table and verify that the the data is still present there

Note: The solution to this assisted practice is provided under the course resources section.

Key Takeaways

- Apache Hive is an open-source data warehousing application with a SQL-like interface that allows users to extract data from Hadoop and related databases.
- HiveQL's DDL (Data Definition Language) constructs and maintains databases and tables.
- Data serialization is a way to represent data in the storage memory as a series of bytes.
- UDFs extend the functionality of Hive with a function written in Java that can be evaluated in HiveQL statements.

DATA AND
ARTIFICIAL INTELLIGENCE

Knowledge Check

Knowledge Check

1

Deleting an individual record is possible in_____ .

a. Hive

b. RDBMS

c. Both A and B

d. None of the above

Knowledge Check

1

Deleting an individual record is possible in_____ .

a. Hive

b. RDBMS

c. Both A and B

d. None of the above

The correct answer is **B**

Hive cannot delete individual records, but an RDBMS can.

Knowledge Check

2

In which HDFS directory is the Hive table created by default?

a. /hive

b. /user/hive/

c. /user/hive/warehouse

d. All of the above

Knowledge Check

2

In which HDFS directory is the Hive table created by default?

a. /hive

b. /user/hive/

c. /user/hive/warehouse

d. All of the above

The correct answer is **C**

Hive table gets created by default in /user/hive/warehouse in HDFS directory.

Knowledge Check

3

Which of the following statements is true for sequential file format?

a. More efficient than a text file

b. Store key-value pairs in a binary container format

c. Not human-readable

d. All of the above

Knowledge Check

3

Which of the following statements is true for sequential file format?

a. More efficient than a text file

b. Store key-value pairs in a binary container format

c. Not human-readable

d. All of the above

The correct answer is **D**

Sequential File format stores key-value pairs in a binary container format, is more efficient than a text file, and it is not human–readable.

Knowledge Check

4

Which of the following statements is true about when not to use partitions?

a. Try to limit partition to less than 20k

b. Avoid partition on columns having many unique rows

c. Be cautious while creating dynamic partition as it can lead to high number of partition

d. All of the above

Knowledge Check

4

Which of the following statements is true about when not to use partitions?

a. Try to limit partition to less than 20k

b. Avoid partition on columns having many unique rows

c. Be cautious while creating dynamic partition as it can lead to high number of partitions

d. All of the above

The correct answer is **D**

Users should avoid using partitions when columns have many unique rows while creating a dynamic partition as it can lead to a high number of partitions and limit partitions to less than 20k.

Knowledge Check

5

Which of the following is a way of representing data in memory as a series of bytes?

a. File formatting

b. Data serialization

c. Both A and B

d. None of the above

Knowledge Check

5

Which of the following is a way of representing data in memory as a series of bytes?

a. File formatting

b. Data serialization

c. Both A and B

d. None of the above

The correct answer is **B**

Data serialization is a way of representing data in memory as a series of bytes.

Lesson-End Project

Post Office Data Analysis using Hive

Problem Scenario: Philips works as a data analyst for PwC and has been delegated to another country since the government is looking to make some significant economic changes, which could mean a lot of work for PwC. The Philips team was recently invited to the PM's office and informed that the PM wants to completely revamp the post offices of the country he has been delegated to because they have the greatest reach to every corner of the country. The revamping will begin in any state, and if the response is positive, it will be spread across the country. The PM's vision is to give every post office the power of banking, allowing people to open accounts, deposit money, and take advantage of schemes. As the leader, Philips chooses a state as the PoC state. You must assist Philips in completing the analysis.

Objective: The objective is to create an optimized solution using hive to read the data from a table.

LESSON-END PROJECT

Lesson-End Project

Tasks to Perform:

1. Upload sample dataset to HDFS
2. Create a sample table on Hive with relevant details
3. Run the query selecting one of the states of your choice from the dataset and observe the time
4. Create an optimized solution with the help of dynamic partitioning
5. Create a new table with the selected state as a partition
6. Upload data from the sample table created earlier
7. Retrieve the data from the portioned table and compare it with the earlier time

LESSON-END PROJECT

DATA AND
ARTIFICIAL INTELLIGENCE

Thank You