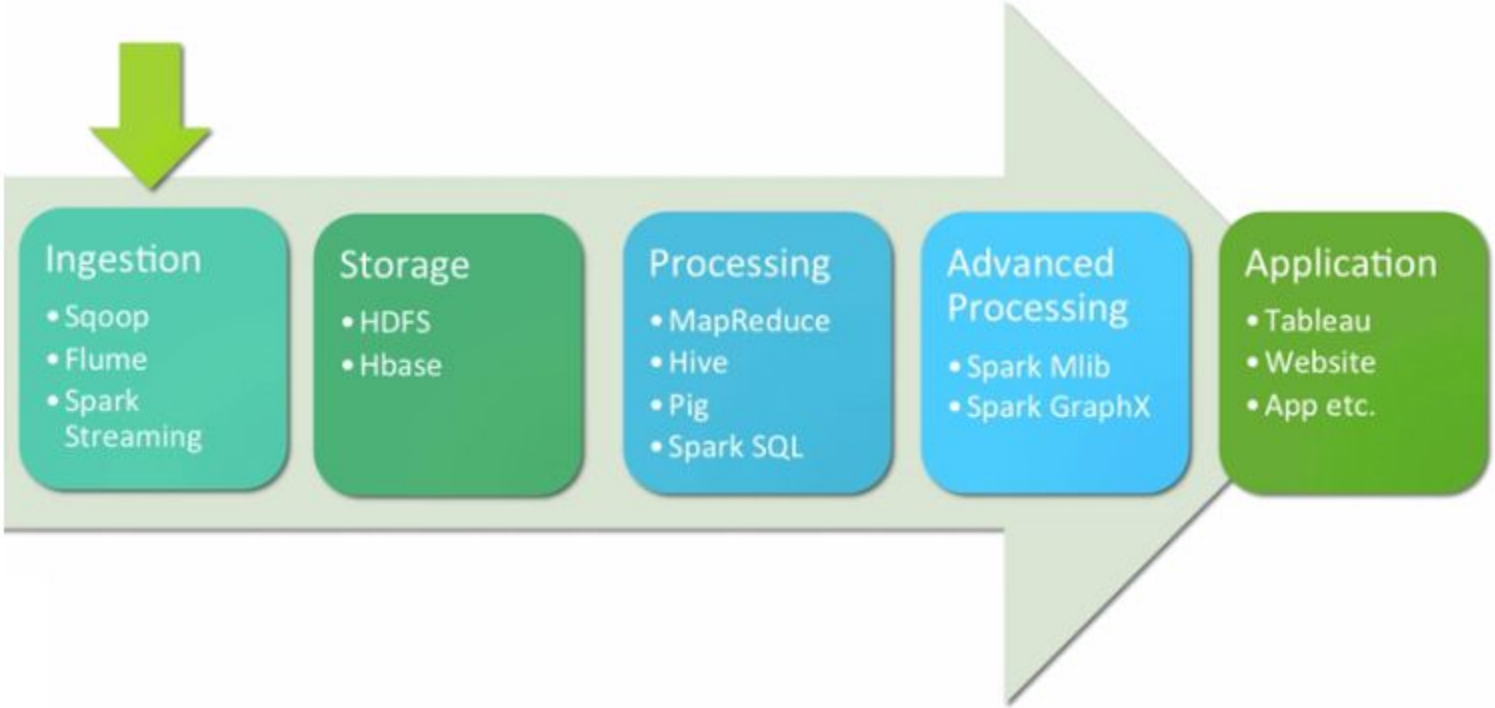
https://sqoop.apache.org/

Master IDDLO

By: Abderrahmane Ezzahout

Big Data Project Stage



Ingestion
- Sqoop
- Flume
- Spark Streaming

Storage
- HDFS
- Hbase

Processing
- MapReduce
- Hive
- Pig
- Spark SQL

Advanced Processing
- Spark Mlib
- Spark GraphX

Application
- Tableau
- Website
- App etc.

Apache Sqoop :

- Apache Sqoop is a tool designed for efficient bulk data transfer between Hadoop and structured datastores such as relational databases or NoSQLdatabases.

- Using Sqoop , you can import data from external systems on to HDFS and populate tables in Hive and HBase.

- Sqoop uses a connector-based architecture, which supports plug-ins.It is extendable to new types of external sources.

- Sqoop comes with connectors for common database systems such as MySQL, PostgreSQL, Oracle, SQLServer, and DB2.

- Sqoop slices up every dataset that needs to be transferred into partitions, and a map-only job is launched for each such partition to handle transferring this data to its destination.

Why Apache Sqoop :

- *One of the very commonly used tools for data transfer for Apache Hadoop.*
- In the data acquisition layer, we have chosen Apache Sqoop as the main technology.

- There are multiple options that can be used in this layer. Also, in place of one technology, there are other options that can be swapped.

- Apache Sqoop is one of the main technologies being used to transfer data to and from structured data stores such as RDBMS, traditional data warehouses, and NoSQL data stores to Hadoop.

- Apache Hadoop finds it very hard to talk to these traditional stores and Sqoop helps to do that integration very easily.

- Sqoop helps in the bulk transfer of data from these stores in a very good manner and, because of this reason, ***Sqoop was chosen as a technology in this layer.***

When to use Sqoop :

Apache Sqoop could be employed for many of the data transfer requirements in a **data lake**, which has HDFS as the main data storage for incoming data from various systems.

Cases where Apache Sqoop makes more sense:

- For regular batch and micro-batch to transfer data to and from RDBMS to Hadoop (HDFS/Hive/HBase), use Apache Sqoop. Apache Sqoop is one of the main and widely used technologies in the data acquisition layer.
- For transferring data from NoSQL data stores like MongoDB and Cassandra into the Hadoop filesystem.
- Enterprises having good amounts of applications whose stores are based on RDBMS, Sqoop is the best option to transfer data into a Data Lake.
- Hadoop is a de-facto standard for storing massive data. Sqoop allows you to transfer data easily into HDFS from a traditional database with ease.
- Use Sqoop when performance is required, as it is able to split and parallelize data transfer.
- Sqoop has a concept of connectors and, if your enterprise has diverse business applications with different data stores, Sqoop is an ideal choice.

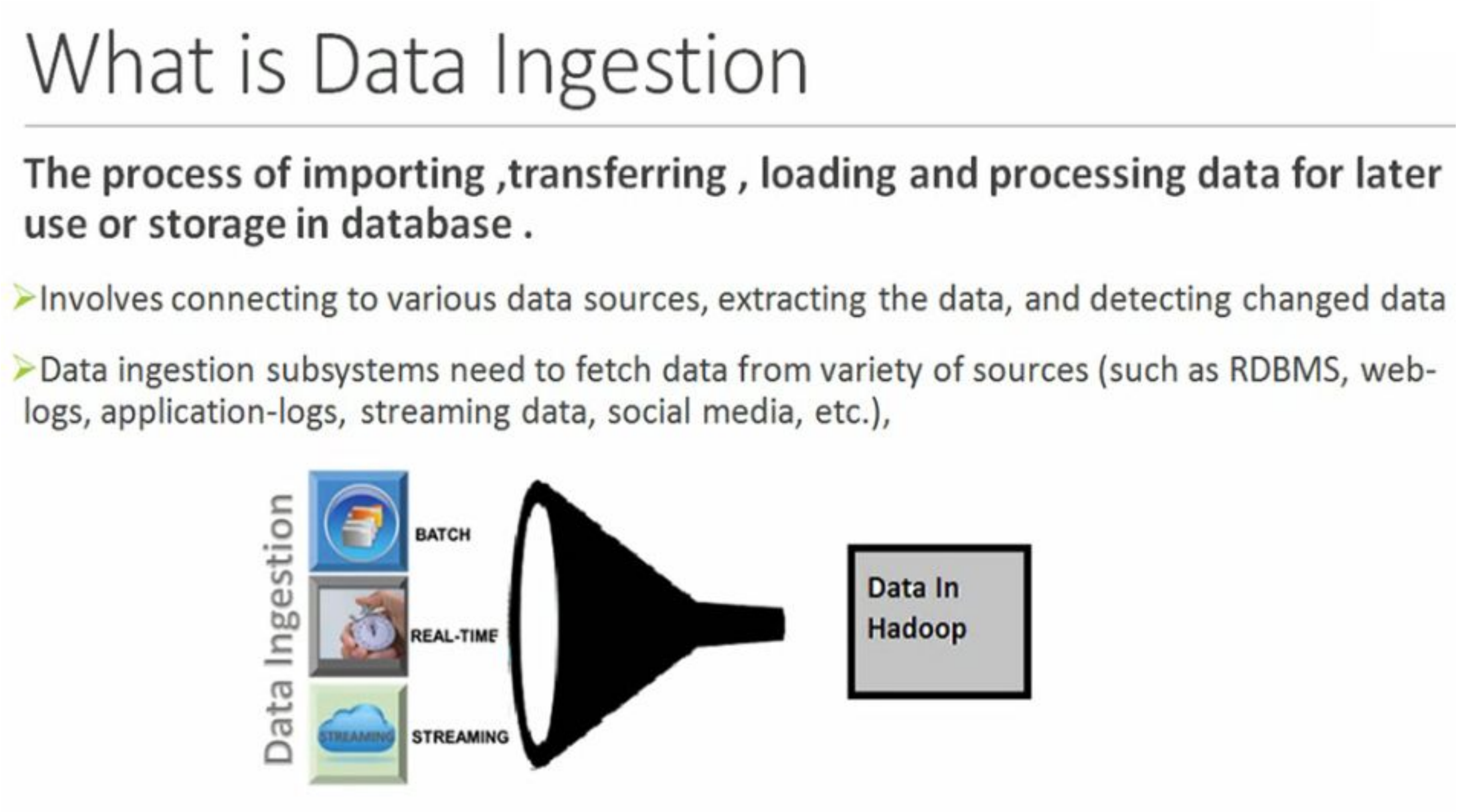
When not to use Sqoop :

Sqoop is the best suited tool when your data lives in database systems such as Oracle, MySQL, PostgreSQL, and Teradata.

Sqoop is not a best fit for event driven data handling.
For event driven data, it's apt to go for Apache Flume as against Sqoop.

To summarize, below are the points when Sqoop should not be used:

- For event driven data.
- For handling and transferring data which are streamed from various business applications. For example data streamed using JMS from a source system.
- For handling real-time data as opposed to regular bulk/batch data and micro-batch.
- Handling data which is in the form of log files generated in different web servers where the business application is hosted.
- If the source data store should not be put under pressure when a Sqoop job is being executed, it's better to avoid Sqoop.
- Also, if the bulk/batch have high volumes of data, the pressure that it would put on the source data store would be even greater, which is usually not desirable.

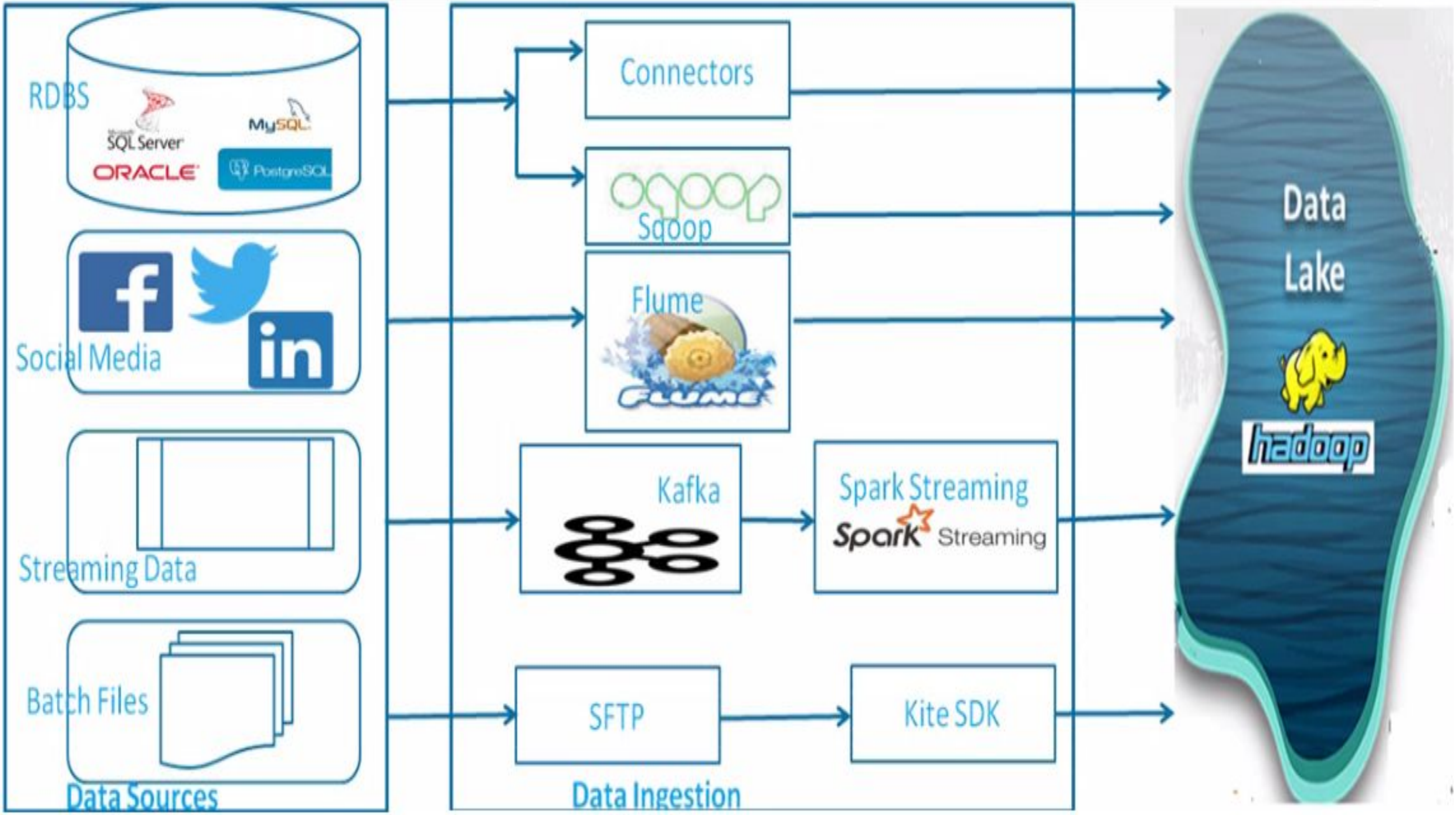


What is Data Ingestion

The process of importing ,transferring , loading and processing data for later use or storage in database .

- Involves connecting to various data sources, extracting the data, and detecting changed data
- Data ingestion subsystems need to fetch data from variety of sources (such as RDBMS, web-logs, application-logs, streaming data, social media, etc.),

Data Ingestion



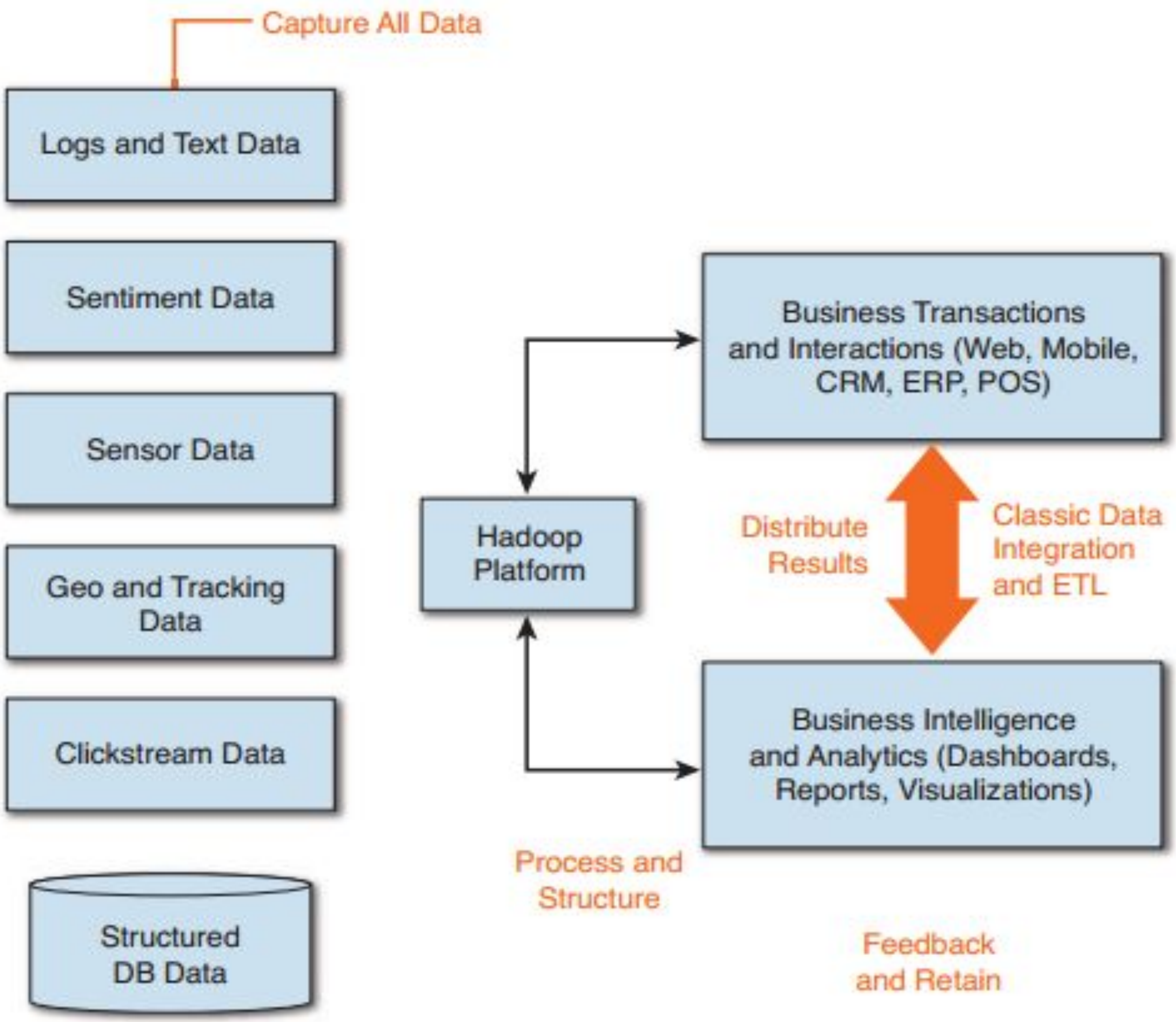
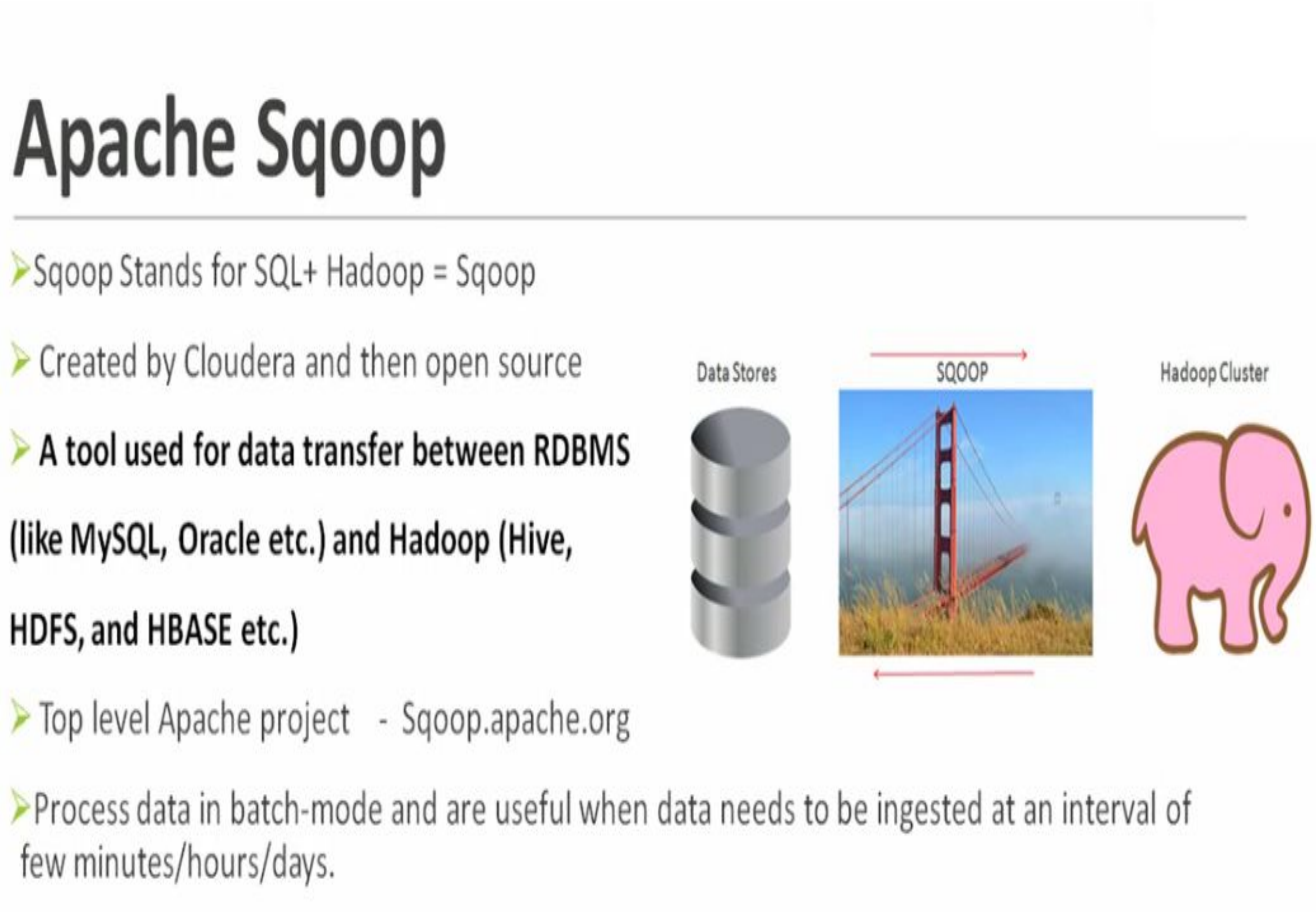
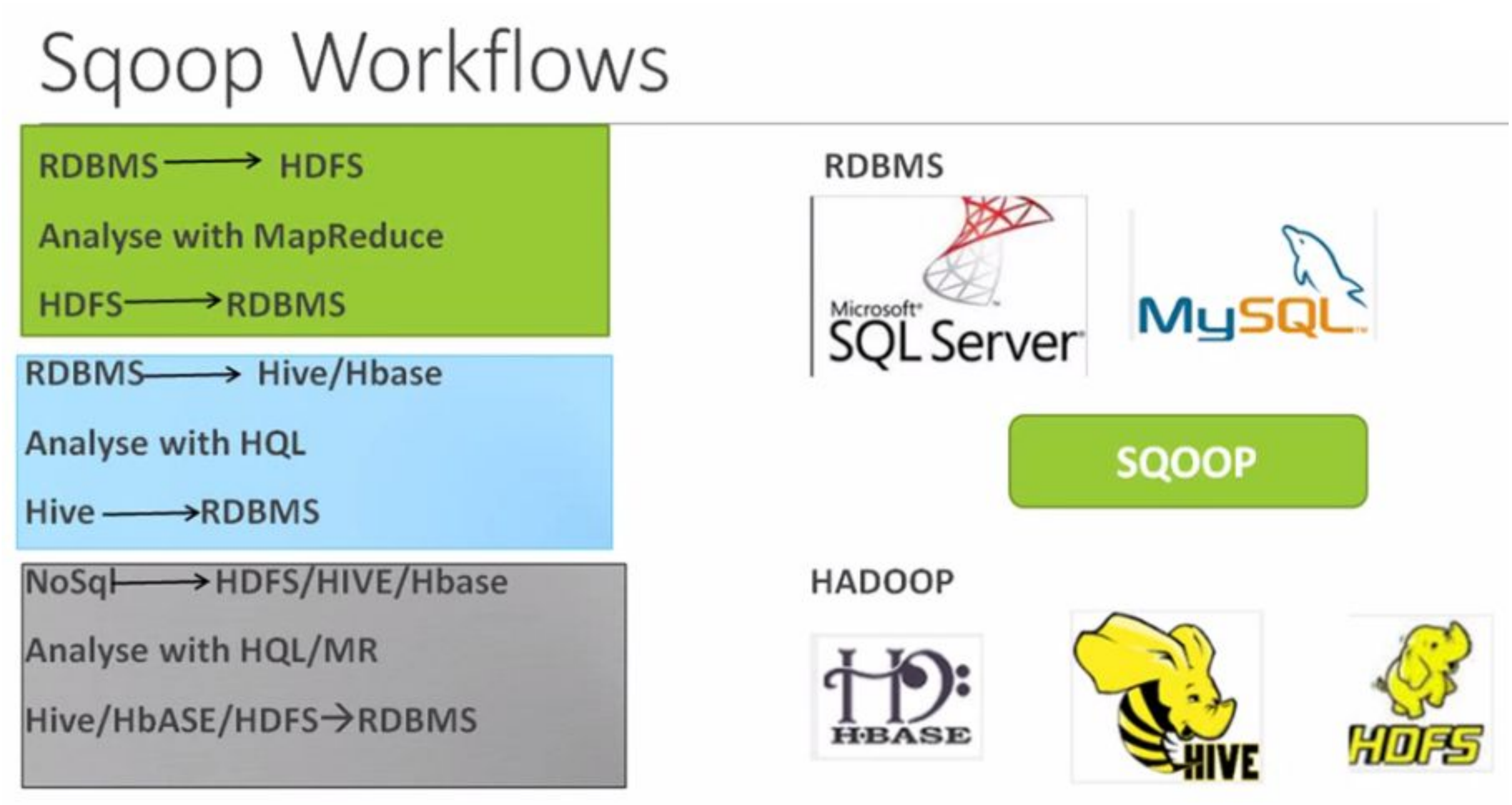


Figure 1.1 A Hadoop-based data lake architecture, with data from a variety of sources flowing into Hadoop, which processes the data and sends it to ETL and BI components



Apache Sqoop

- Sqoop Stands for SQL+ Hadoop = Sqoop
- Created by Cloudera and then open source
- **A tool used for data transfer between RDBMS (like MySQL, Oracle etc.) and Hadoop (Hive, HDFS, and HBASE etc.)**
- Top level Apache project - Sqoop.apache.org
- Process data in batch-mode and are useful when data needs to be ingested at an interval of few minutes/hours/days.



Sqoop Workflows

RDBMS ⟶ HDFS

Analyse with MapReduce

HDFS ⟶ RDBMS

RDBMS ⟶ Hive/Hbase

Analyse with HQL

Hive ⟶ RDBMS

NoSql ⟶ HDFS/HIVE/Hbase

Analyse with HQL/MR

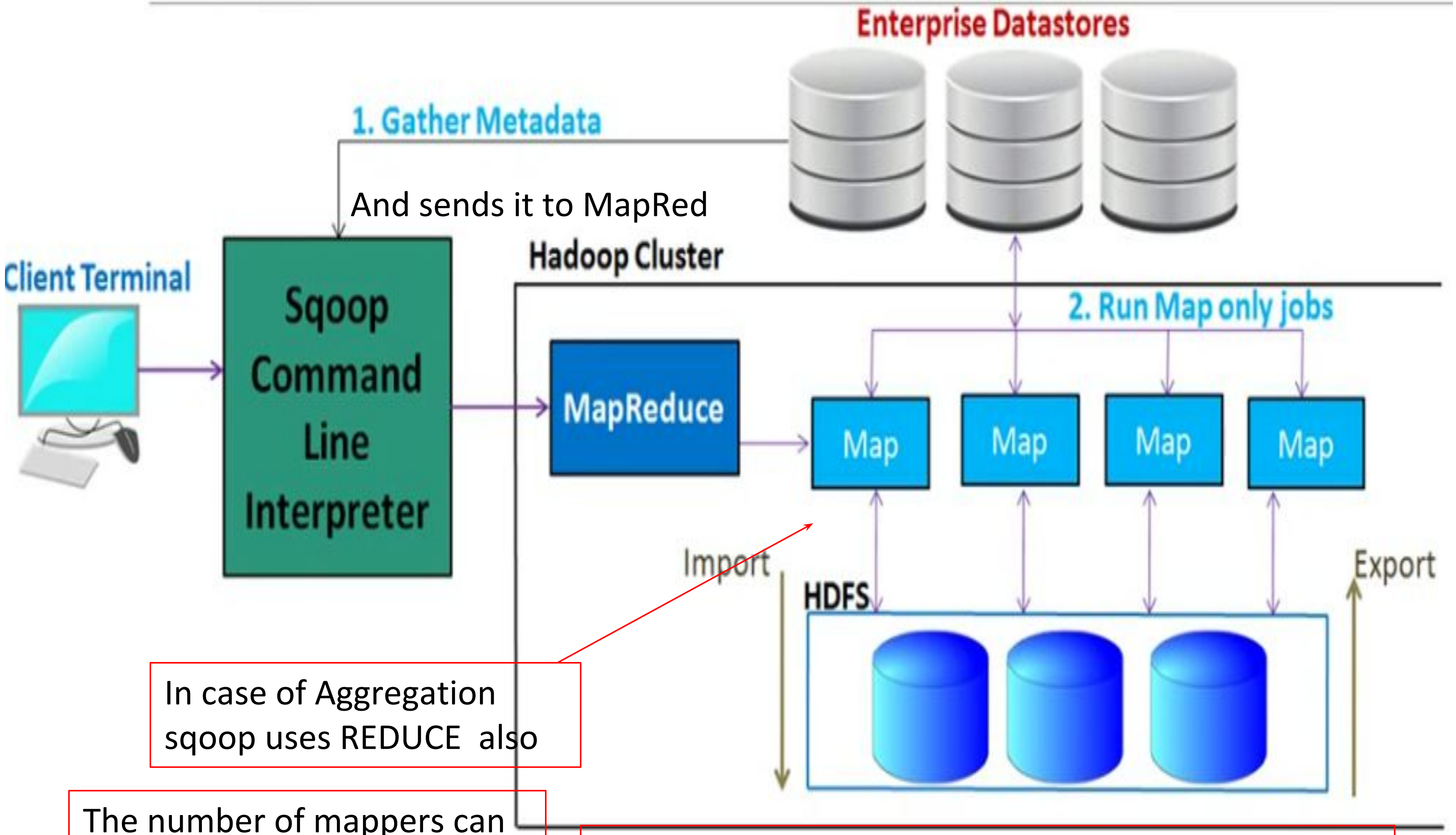
Hive/HbASE/HDFS→RDBMS

RDBMS

SQOOP

HADOOP

Sqoop Architecture

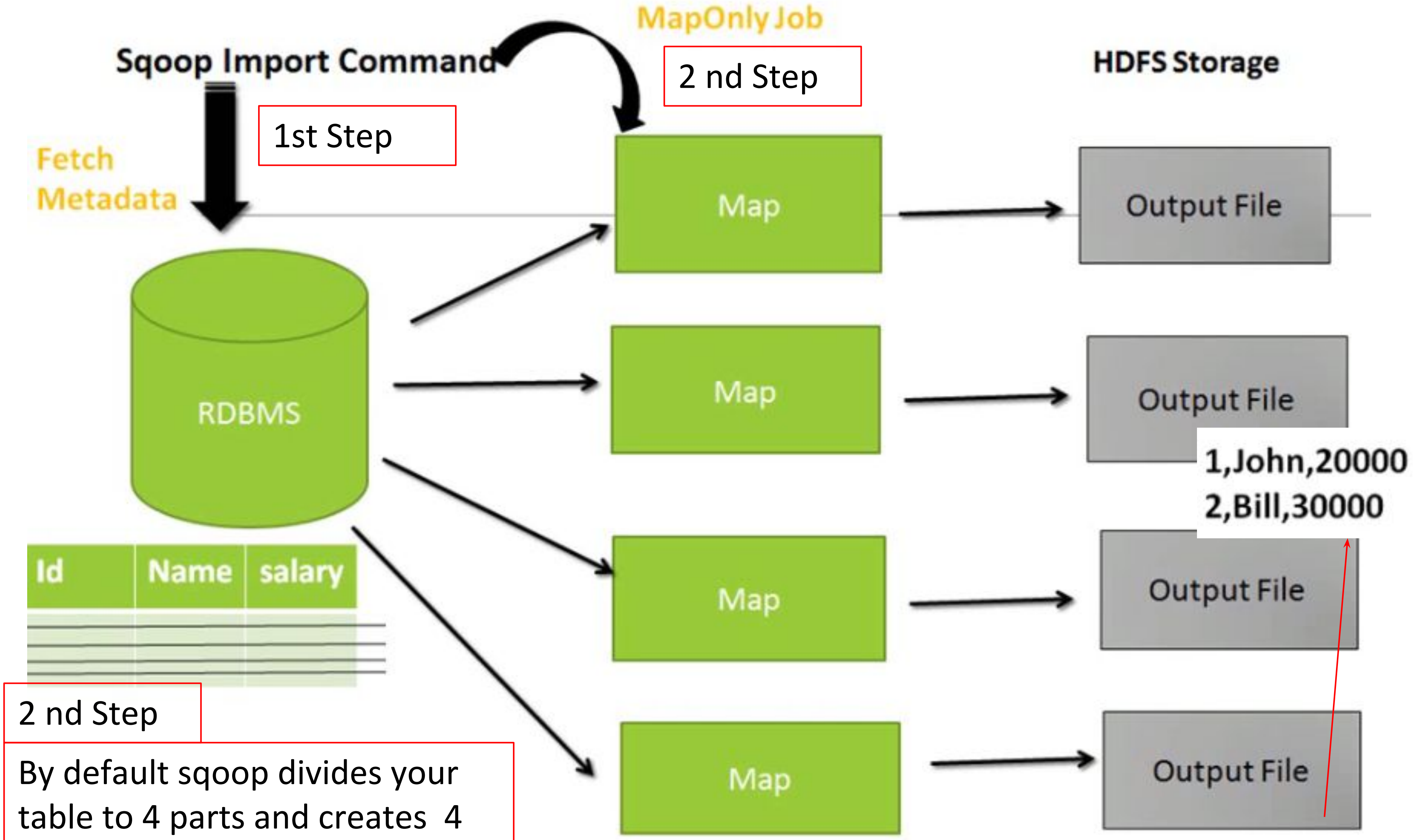


And sends it to MapRed

In case of Aggregation sqoop uses REDUCE also

The number of mappers can Be defined by the user.

Sqoop assigns Data to each MAPPER. Then the mappers connect to Datastores using JDBC connectors to write those Data to HDFS.



1st Step

2 nd Step

2 nd Step

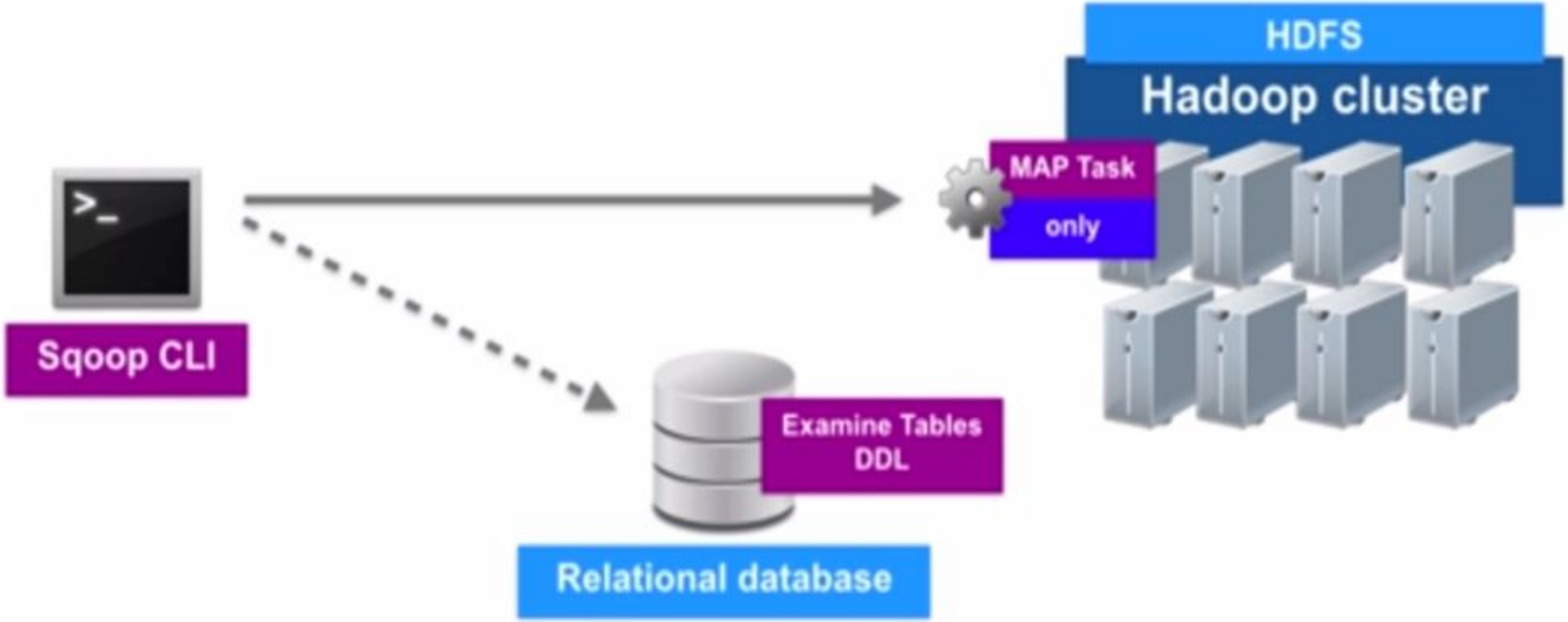
By default sqoop divides your table to 4 parts and creates 4 Maps

By default the output format is comma separated field and lines are recorded separated by New lines

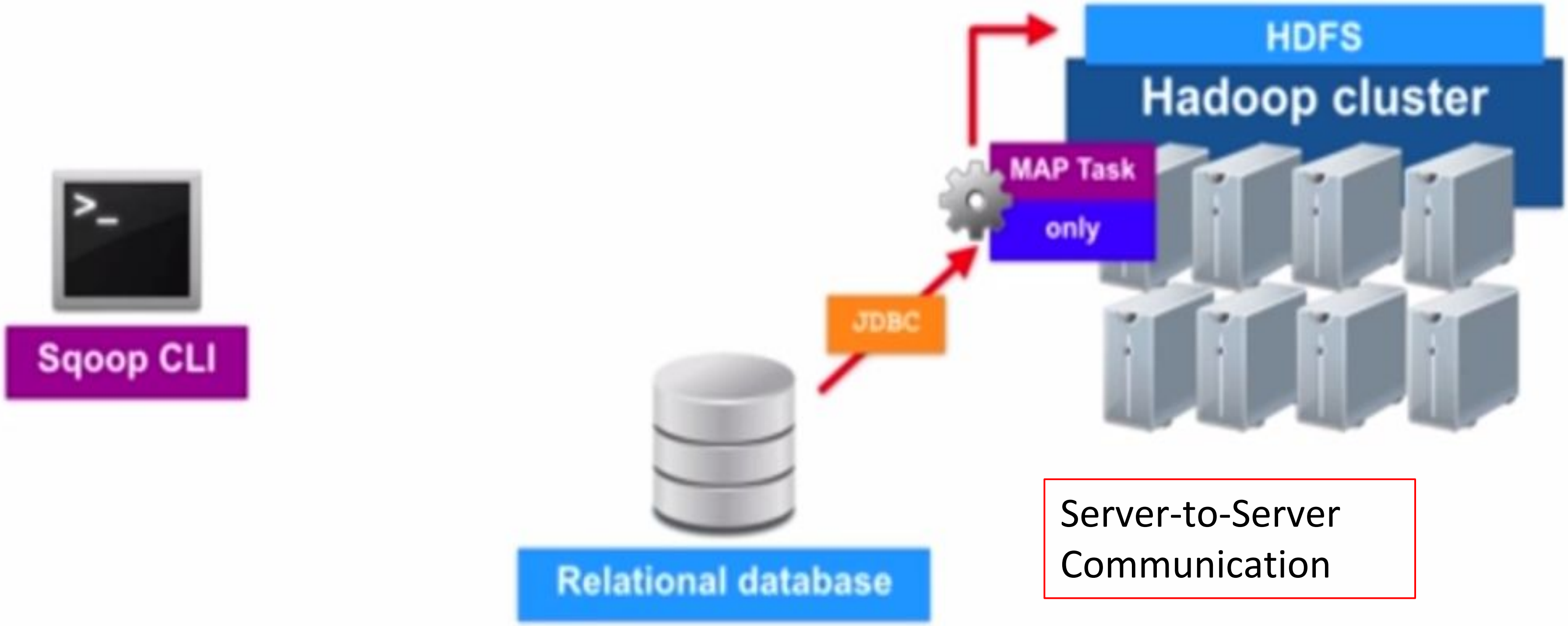
Job command :

- A job is stored as a metadata entity within Sqoop metastore.
- The primary objective of creating a job is to define a one-time configuration between source and target systems and reuse it multiple times to perform export or import.
- Like any other entity, even job supports lifecycle methods such as create, get, update, and delete with a few other functions.

SQOOP CLI BASICS



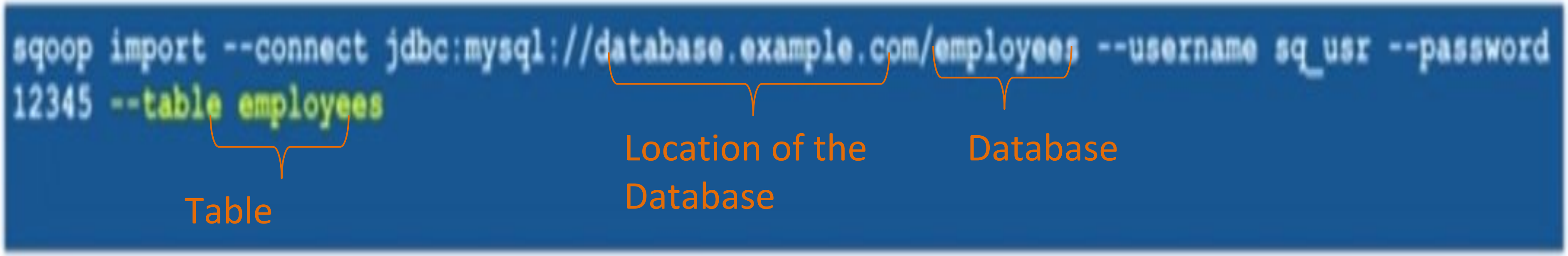
SQOOP CLI BASICS



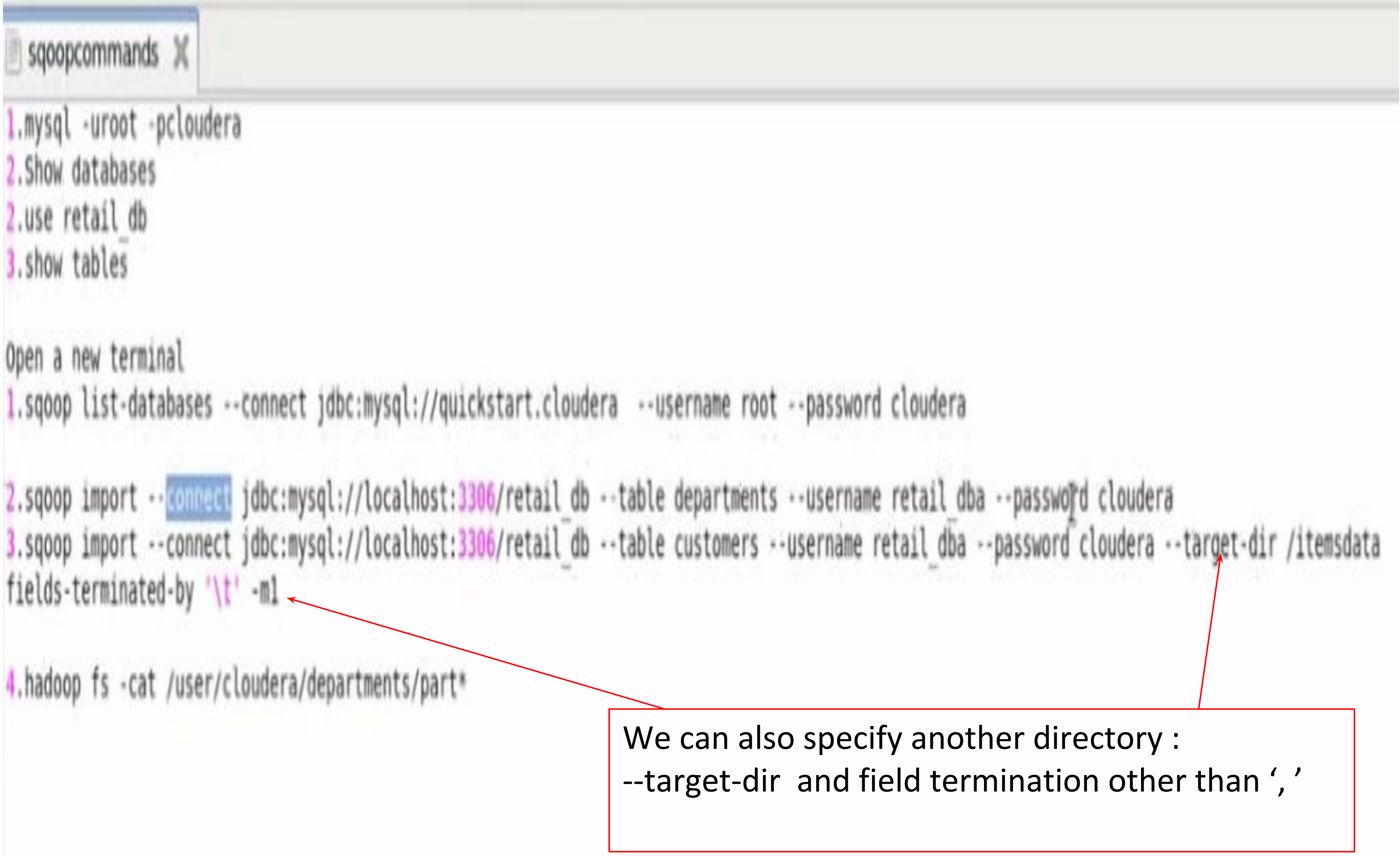
Server-to-Server Communication

SQOOP CLI BASICS

- Basic Sqoop import CLI example
- Use “-P” instead of “--password" to read password from console
- Import the entire “employees” table

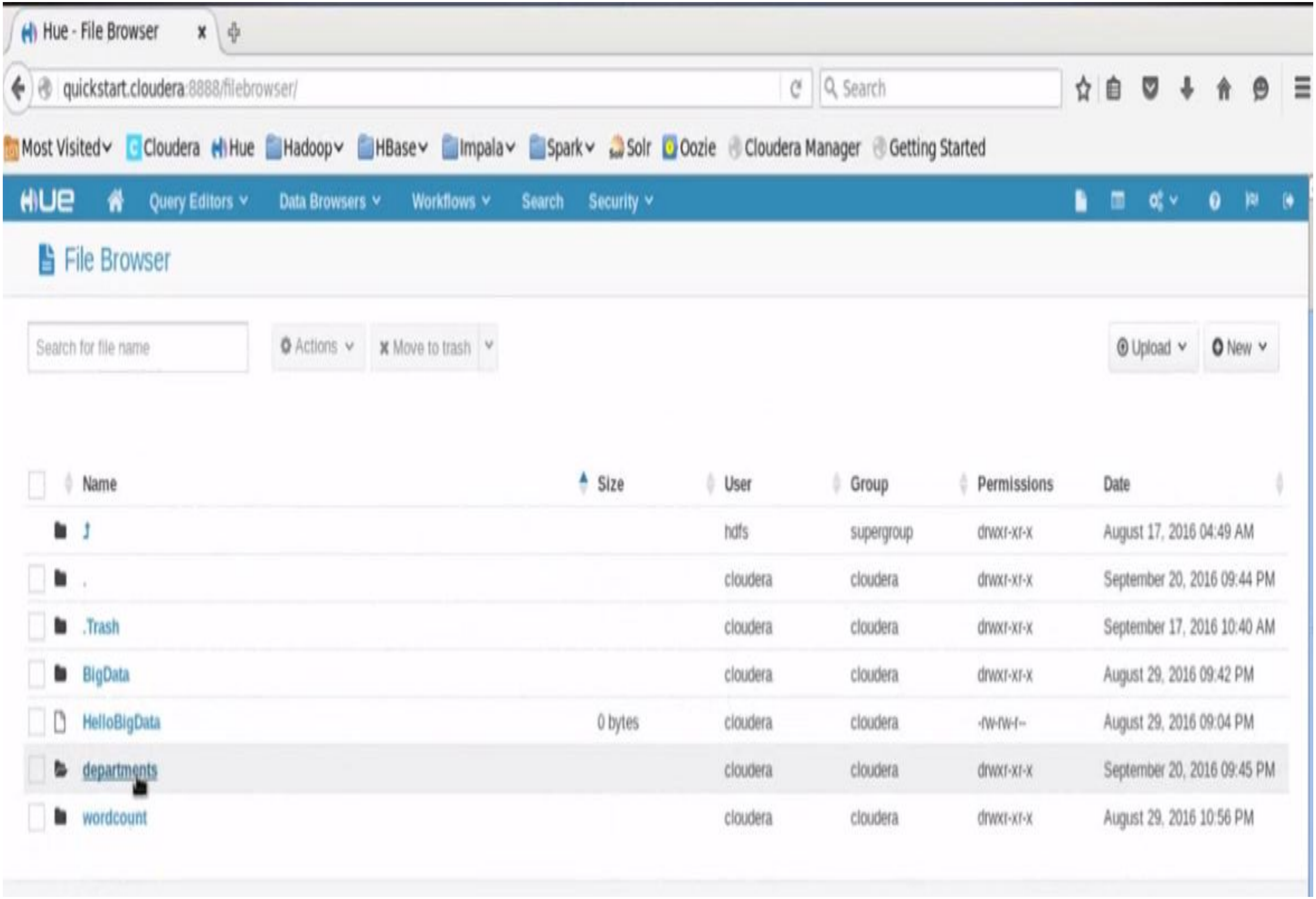


```
sqoop import --connect jdbc:mysql://database.example.com/employees --username sq_usr --password
12345 --table employees
```

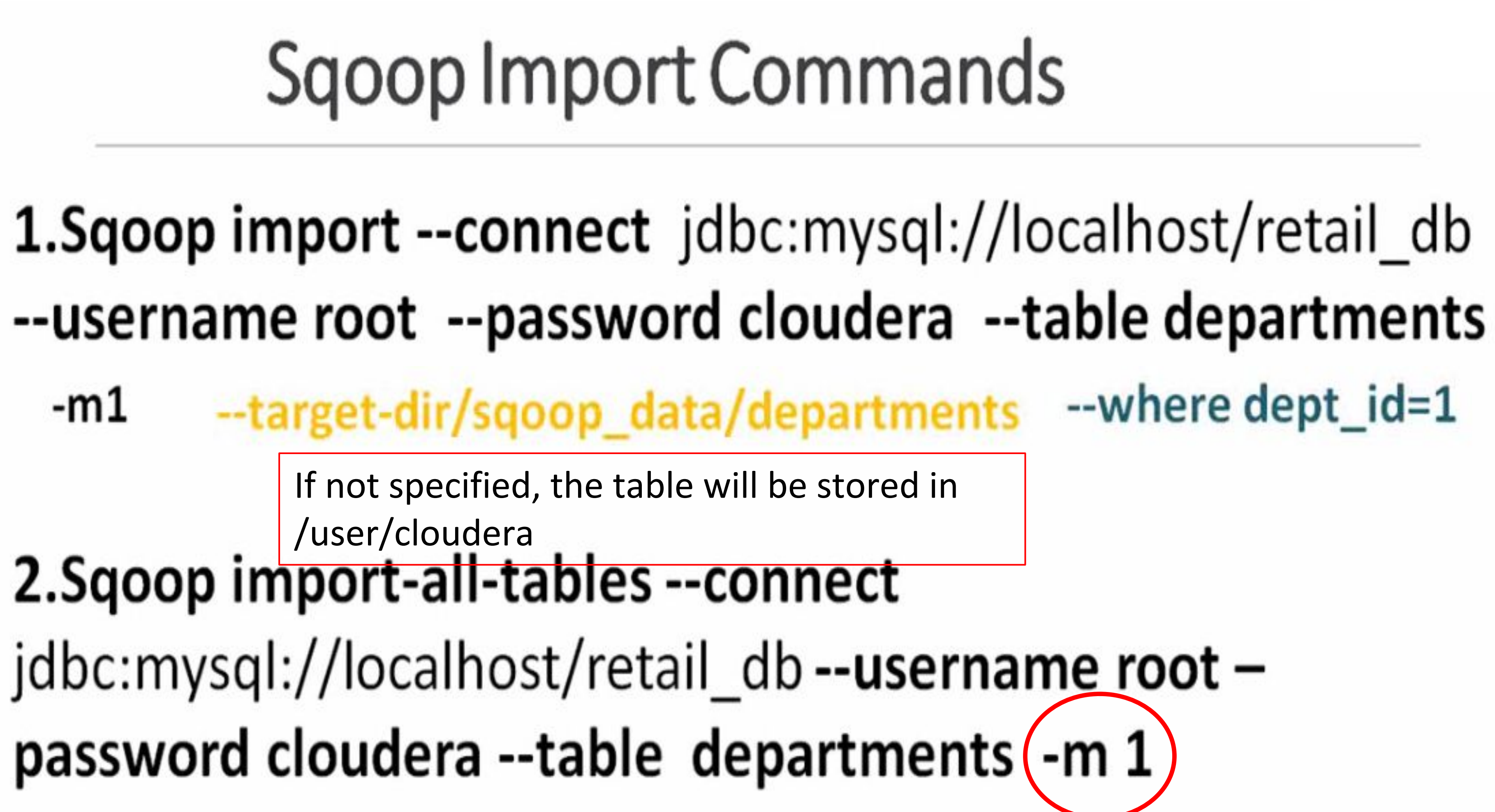


```
1.mysql -uroot -pcloudera
2.Show databases
2.use retail_db
3.show tables

Open a new terminal
1.sqoop list-databases --connect jdbc:mysql://quickstart.cloudera  --username root --password cloudera

2.sqoop import --connect jdbc:mysql://localhost:3306/retail_db --table departments --username retail_dba --password cloudera
3.sqoop import --connect jdbc:mysql://localhost:3306/retail_db --table customers --username retail_dba --password cloudera --target-dir /itemsdata
fields-terminated-by '\t' -m1

4.hadoop fs -cat /user/cloudera/departments/part*
```

We can also specify another directory :
--target-dir and field termination other than ‘, ’



□	Name	Size	User	Group	Permissions	Date
	⤴		hdfs	supergroup	drwxr-xr-x	August 17, 2016 04:49 AM
□	.		cloudera	cloudera	drwxr-xr-x	September 20, 2016 09:44 PM
□	.Trash		cloudera	cloudera	drwxr-xr-x	September 17, 2016 10:40 AM
□	BigData		cloudera	cloudera	drwxr-xr-x	August 29, 2016 09:42 PM
□	HelloBigData	0 bytes	cloudera	cloudera	-rw-rw-r--	August 29, 2016 09:04 PM
□	departments		cloudera	cloudera	drwxr-xr-x	September 20, 2016 09:45 PM
□	wordcount		cloudera	cloudera	drwxr-xr-x	August 29, 2016 10:56 PM



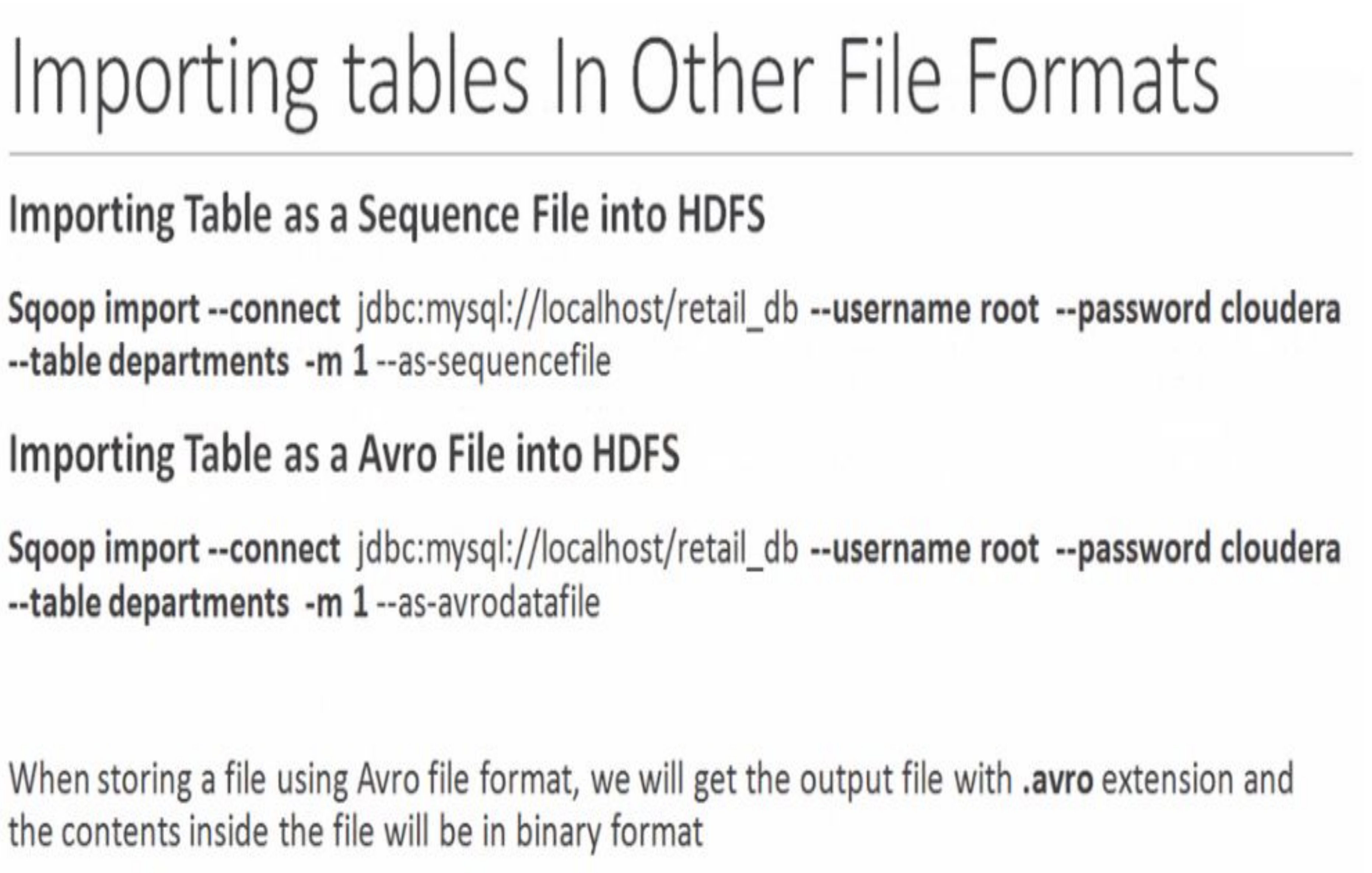
Sqoop Import Commands

```
1.Sqoop import --connect  jdbc:mysql://localhost/retail_db
--username root  --password cloudera  --table departments
  -m1     --target-dir/sqoop_data/departments   --where dept_id=1
```

If not specified, the table will be stored in /user/cloudera

```
2.Sqoop import-all-tables --connect
jdbc:mysql://localhost/retail_db --username root –
password cloudera --table  departments -m 1
```

This parameter allows you to control parallelism :
- m 1 means we are using 1 Mapper.
NB : By default sqoop uses 4 Mappers



Importing tables In Other File Formats

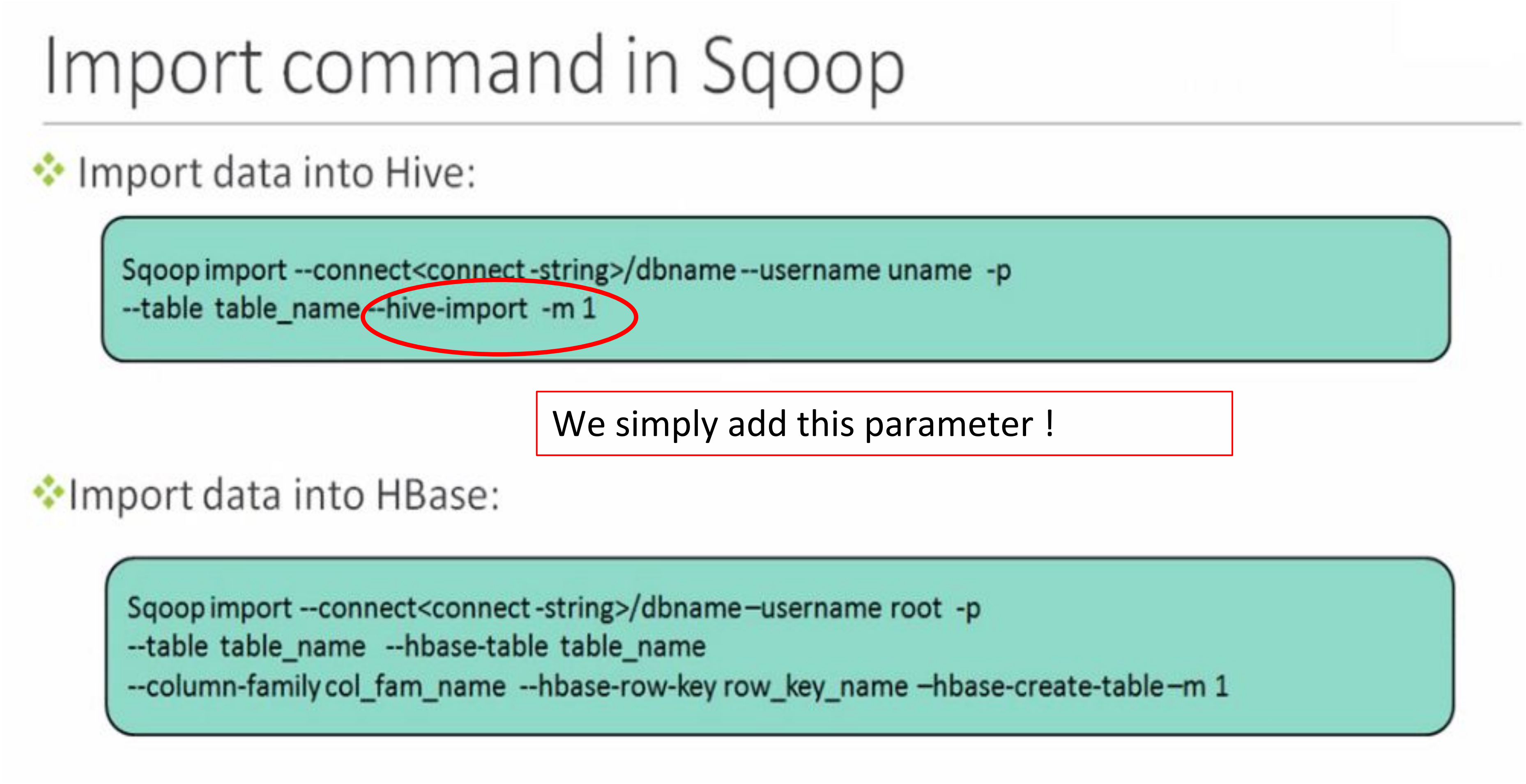
Importing Table as a Sequence File into HDFS

```
Sqoop import --connect  jdbc:mysql://localhost/retail_db --username root  --password cloudera
--table departments  -m 1 --as-sequencefile
```

Importing Table as a Avro File into HDFS

```
Sqoop import --connect  jdbc:mysql://localhost/retail_db --username root  --password cloudera
--table departments  -m 1 --as-avrodatafile
```

When storing a file using Avro file format, we will get the output file with **.avro** extension and the contents inside the file will be in binary format



Import command in Sqoop

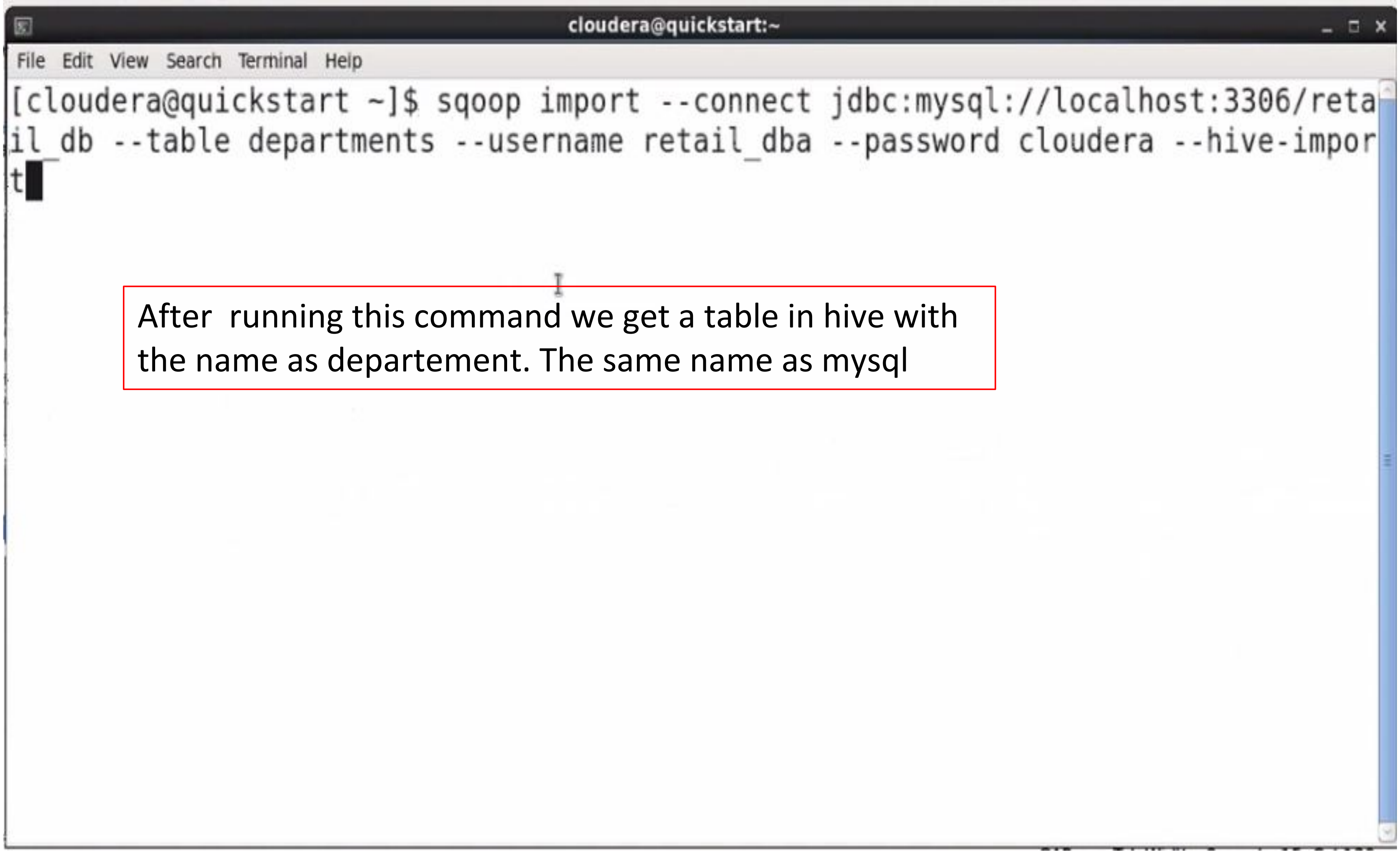
- Import data into Hive:

```
Sqoop import --connect<connect-string>/dbname--username uname  -p
--table  table_name --hive-import  -m 1
```

We simply add this parameter !

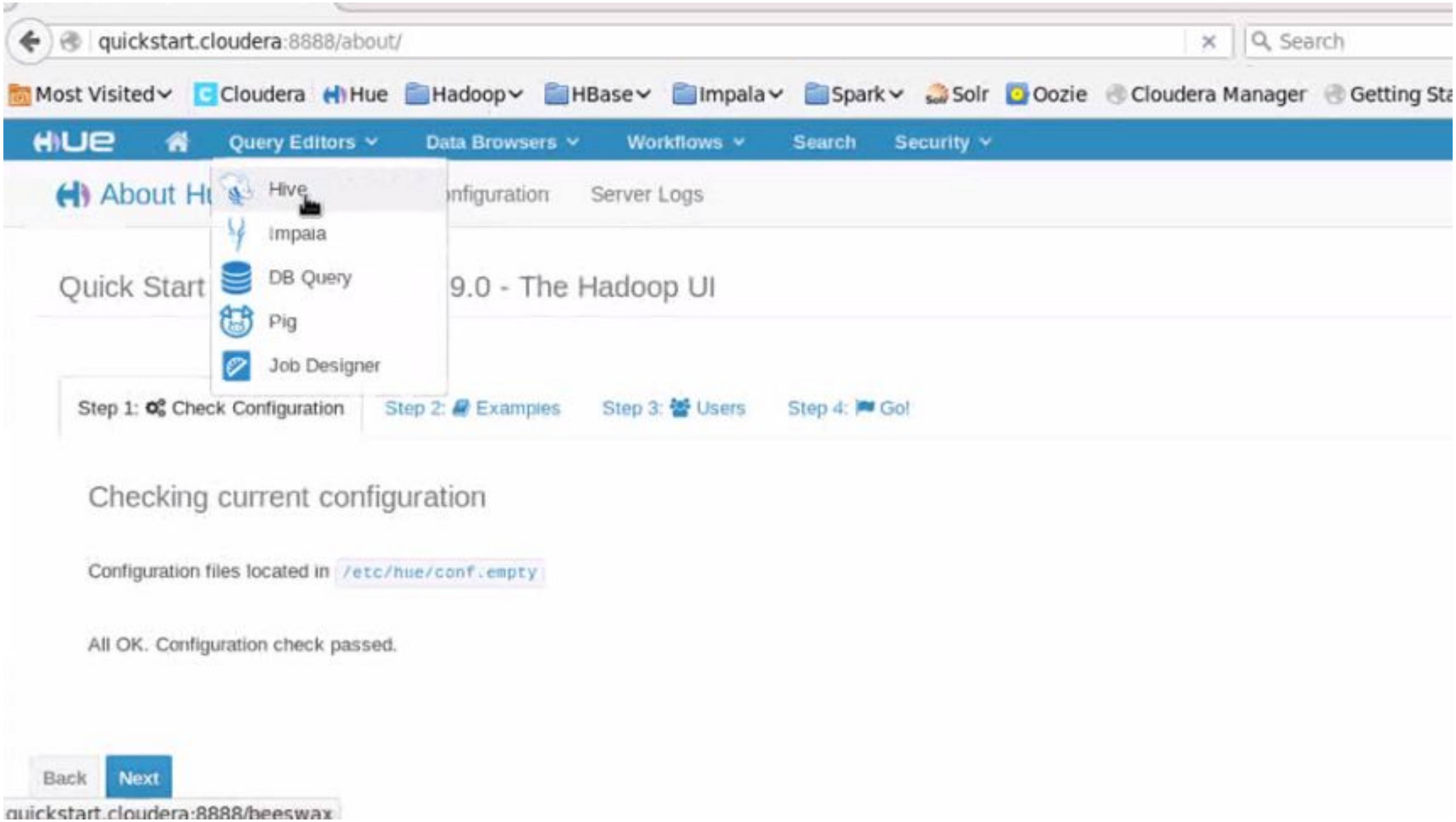
- Import data into HBase:

```
Sqoop import --connect<connect-string>/dbname–username root  -p
--table  table_name   --hbase-table  table_name
--column-family col_fam_name  --hbase-row-key row_key_name –hbase-create-table–m 1
```



```
[cloudera@quickstart ~]$ sqoop import --connect jdbc:mysql://localhost:3306/reta
il_db --table departments --username retail_dba --password cloudera --hive-impor
t
```

After running this command we get a table in hive with the name as departement. The same name as mysql

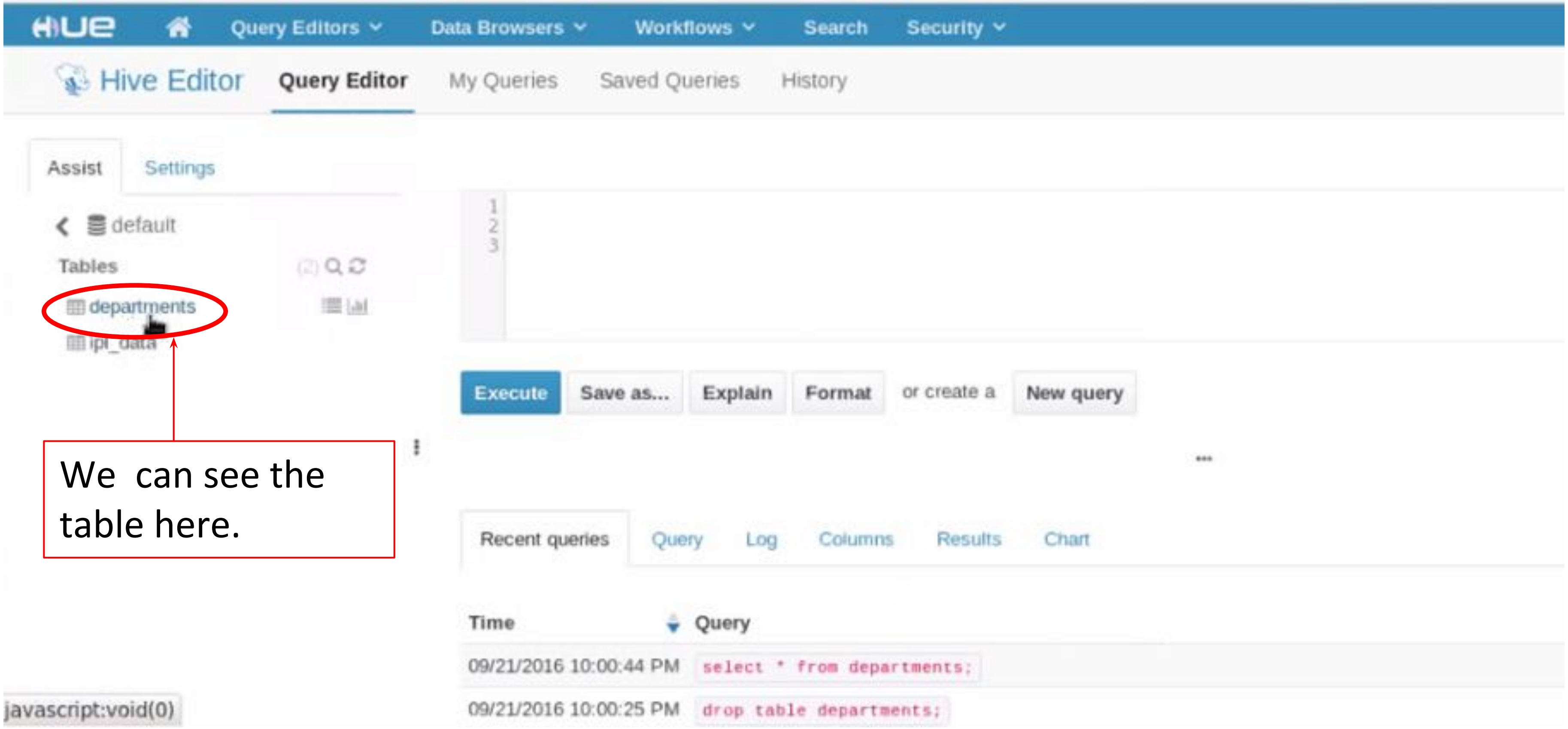


Quick Start 9.0 - The Hadoop UI

Checking current configuration

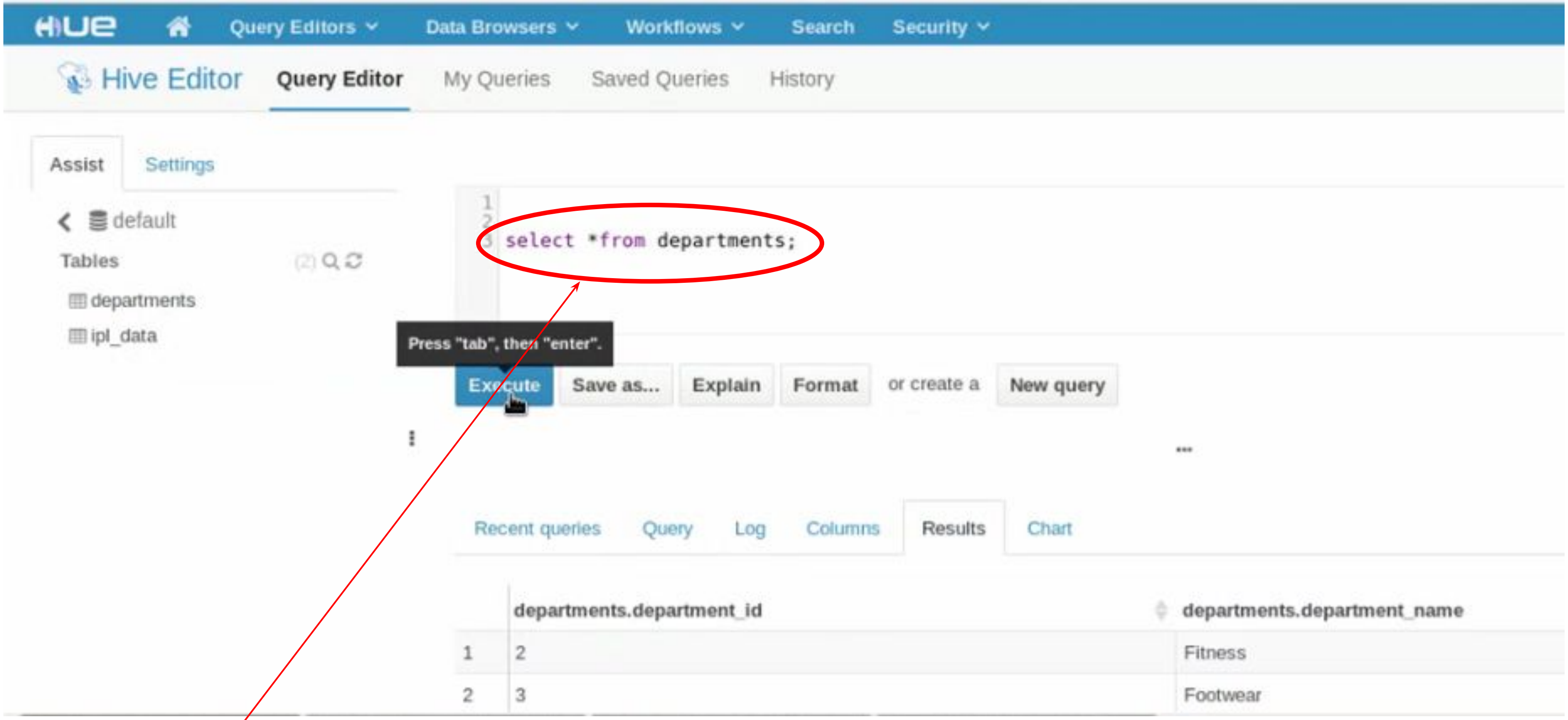
Configuration files located in `/etc/hue/conf.empty`

All OK. Configuration check passed.



Time	Query
09/21/2016 10:00:44 PM	`select * from departments;`
09/21/2016 10:00:25 PM	`drop table departments;`

We can see the table here.



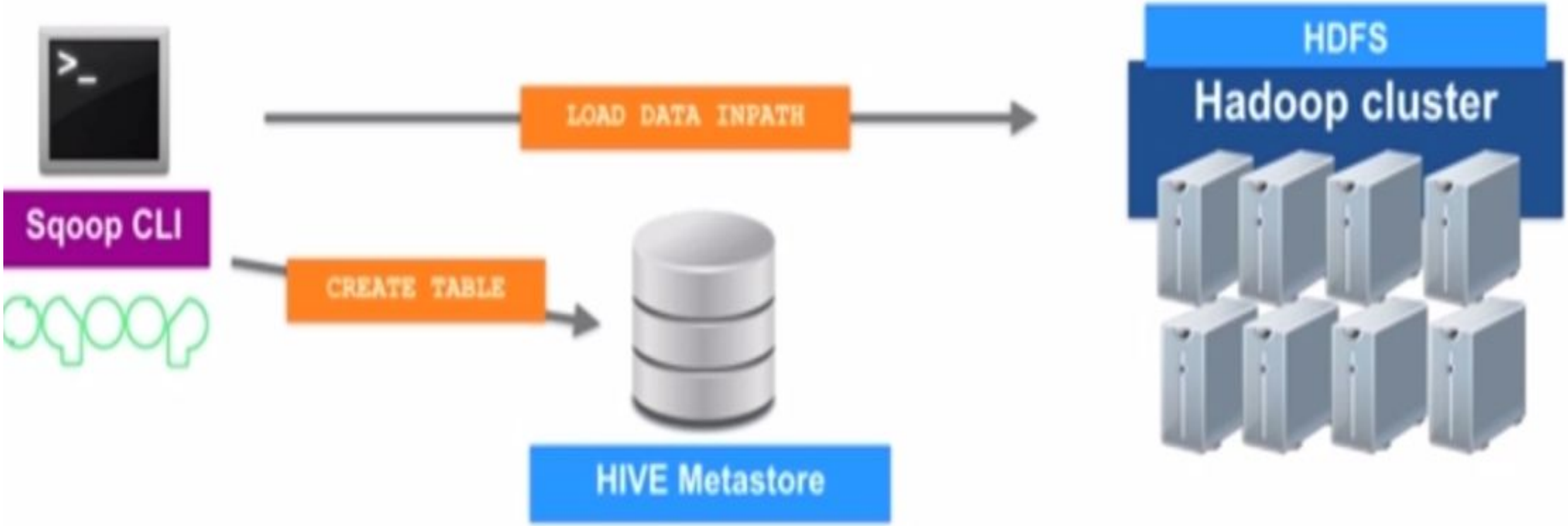
```
select *from departments;
```

Press "tab", then "enter".

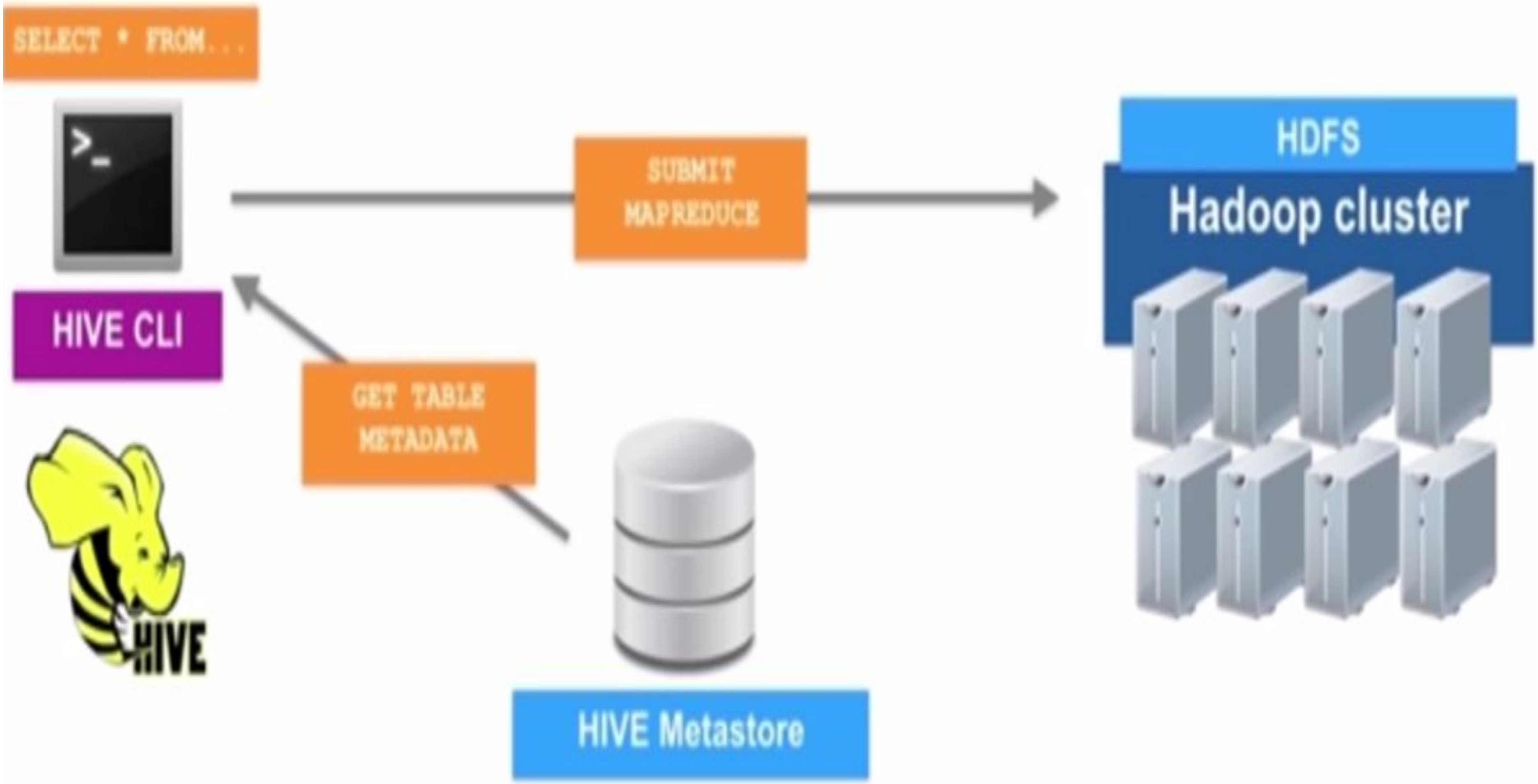
	departments.department_id	departments.department_name
1	2	Fitness
2	3	Footwear

We can then execute this query

SQOOP HIVE IMPORT



SQOOP HIVE IMPORT



SELECT * FROM...

SUBMIT MAPREDUCE

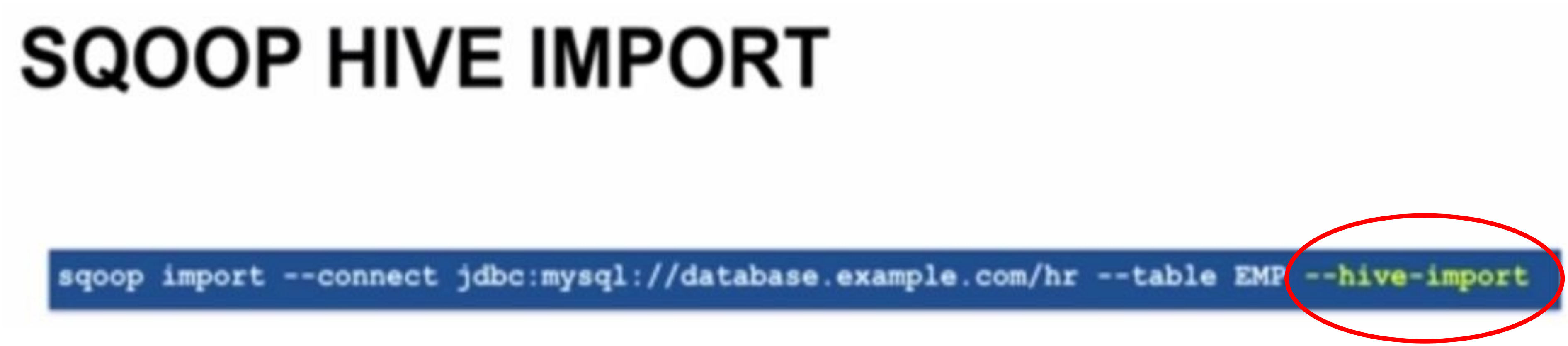
HDFS

Hadoop cluster

HIVE CLI

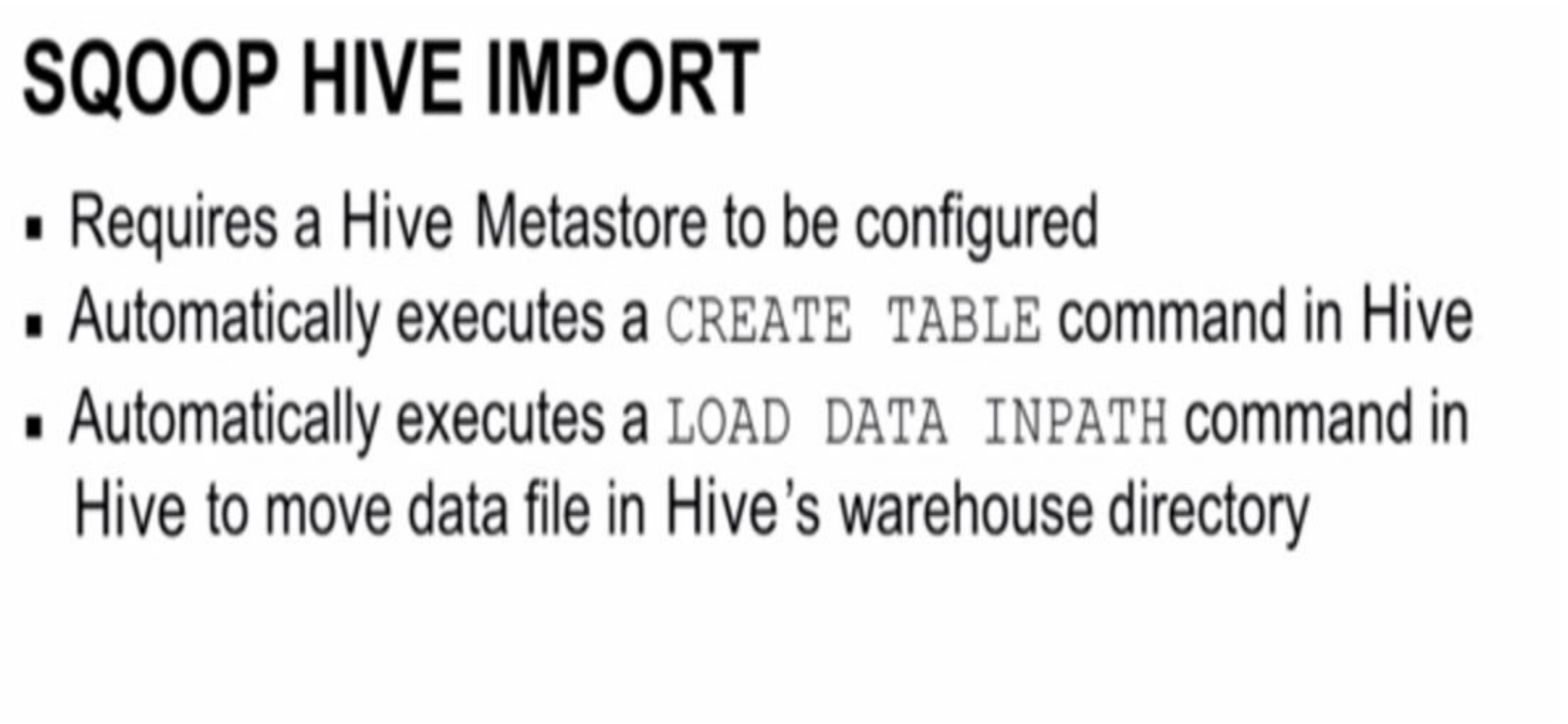
GET TABLE METADATA

HIVE Metastore



SQOOP HIVE IMPORT

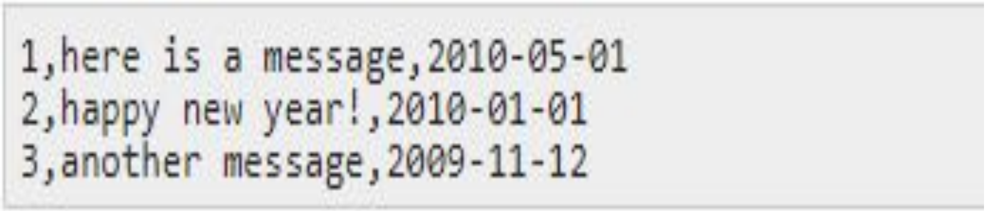
```
sqoop import --connect jdbc:mysql://database.example.com/hr --table EMP --hive-import
```



SQOOP HIVE IMPORT

- Requires a Hive Metastore to be configured
- Automatically executes a `CREATE TABLE` command in Hive
- Automatically executes a `LOAD DATA INPATH` command in Hive to move data file in Hive's warehouse directory

- Sqoop import command allows you to change the default setting by explicitly specifying the field separator and the line delimiter also.
- You can import data in one of two file formats: delimited text or SequenceFiles.
- Delimited text is the default import format. You can also specify it explicitly by using the --as-textfile argument. This argument will write string-based representations of each record to the output files, with delimiter characters between individual columns and rows. These delimiters may be commas, tabs, or other characters. (The delimiters can be selected; see "Output line formatting arguments.") The following is the results of an example text-based import:



```
1,here is a message,2010-05-01
2,happy new year!,2010-01-01
3,another message,2009-11-12
```

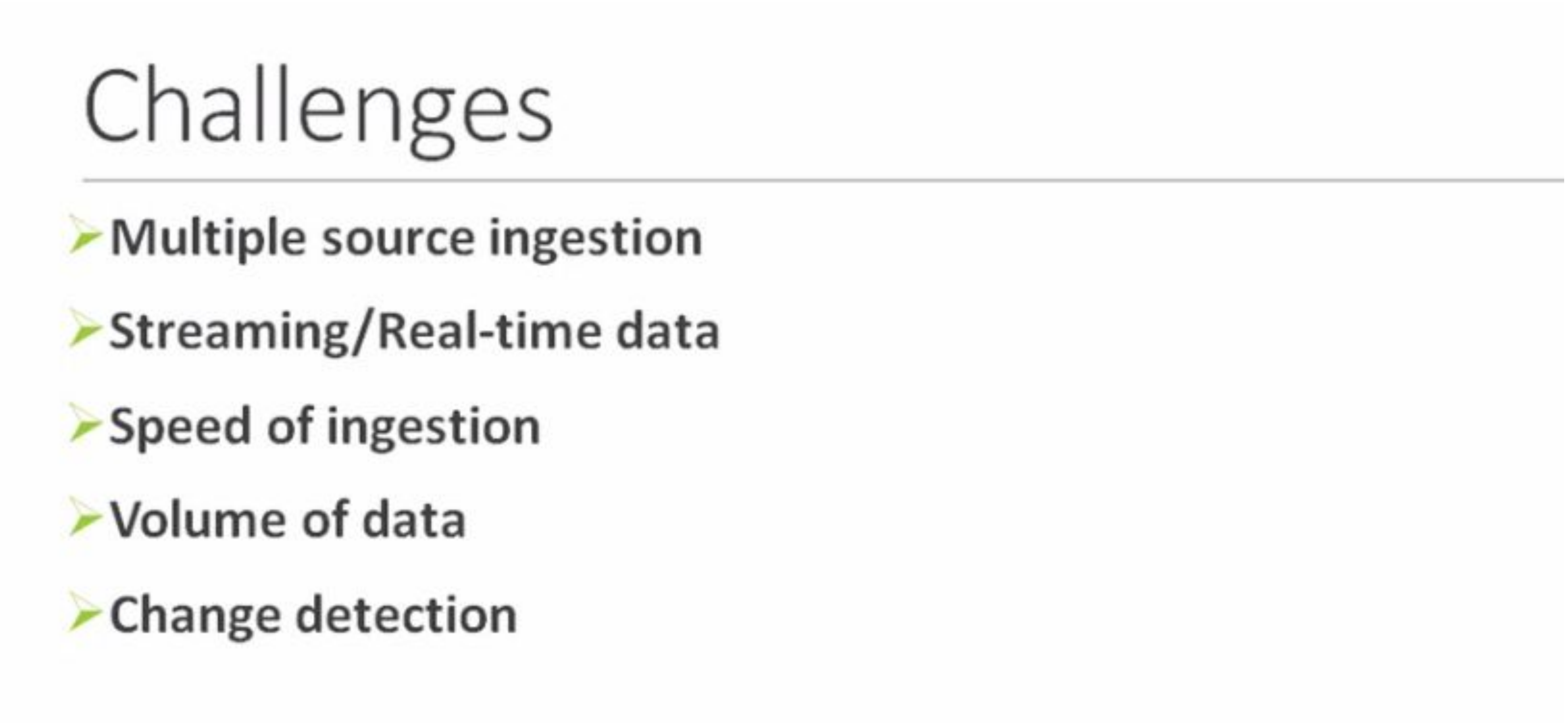
NOTE : Reading from Sequence file better than
Delimited text in terms of performance.

- Delimited text is appropriate for most non-binary data types. It also readily supports further manipulation by other tools, such as Hive.

- SequenceFiles are a binary format that store individual records in custom record-specific data types. These data types are manifested as Java classes.

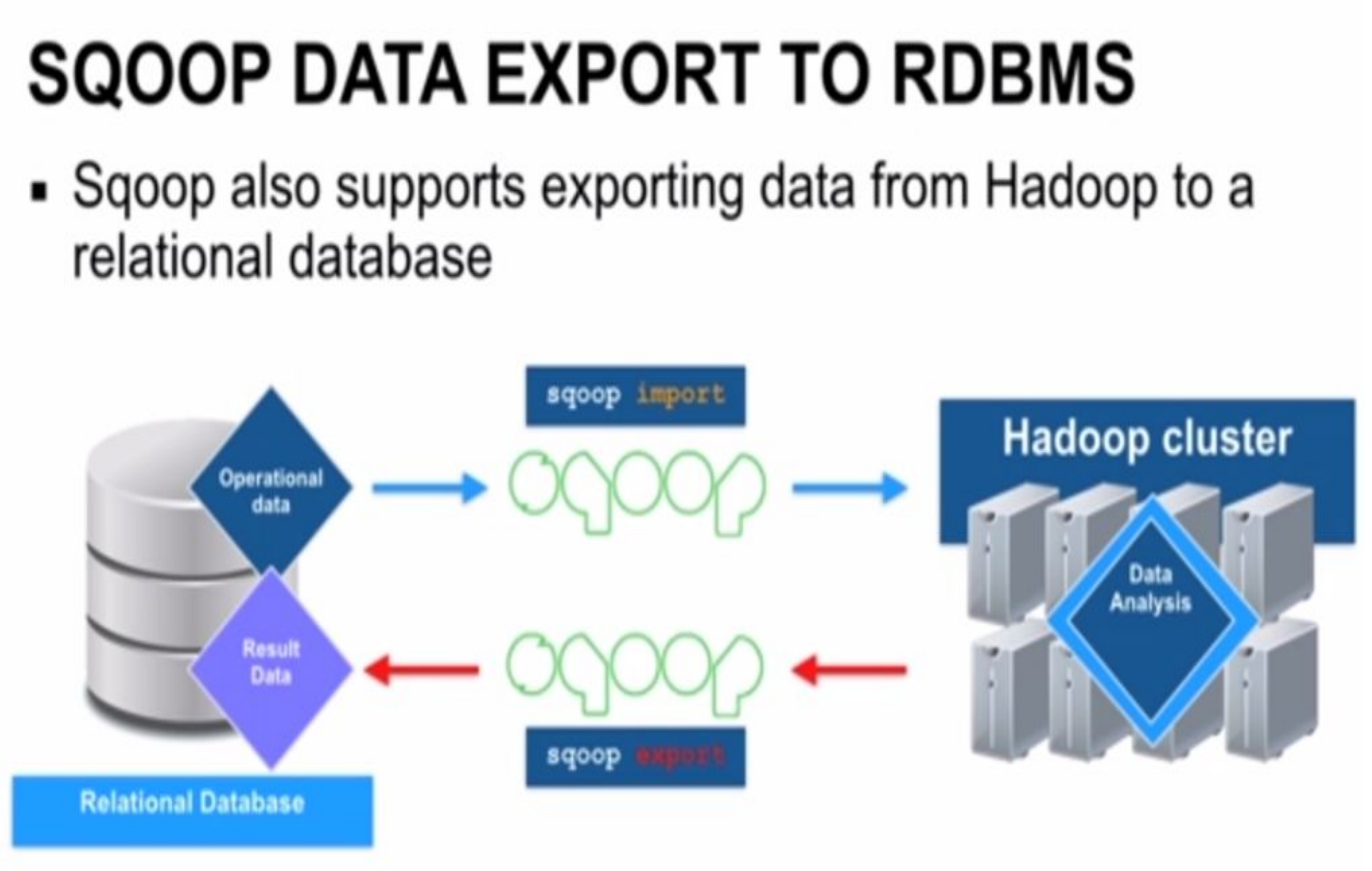
- Sqoop will automatically generate these data types for you. This format supports exact storage of all data in binary representations, and is appropriate for storing binary data (for example, VARBINARY columns), or data that will be principly manipulated by custom MapReduce programs (reading from SequenceFiles is higher-performance than reading from text files, as records do not need to be parsed).

- Avro data files are a compact, efficient binary format that provides interoperability with applications written in other programming languages. Avro also supports versioning, so that when, e.g., columns are added or removed from a table, previously imported data files can be processed along with new ones.
- By default, data is not compressed. You can compress your data by using the deflate (gzip) algorithm with the -z or --compress argument, or specify any Hadoop compression codec using the --compression-codec argument. This applies to SequenceFile, text, and Avro files.



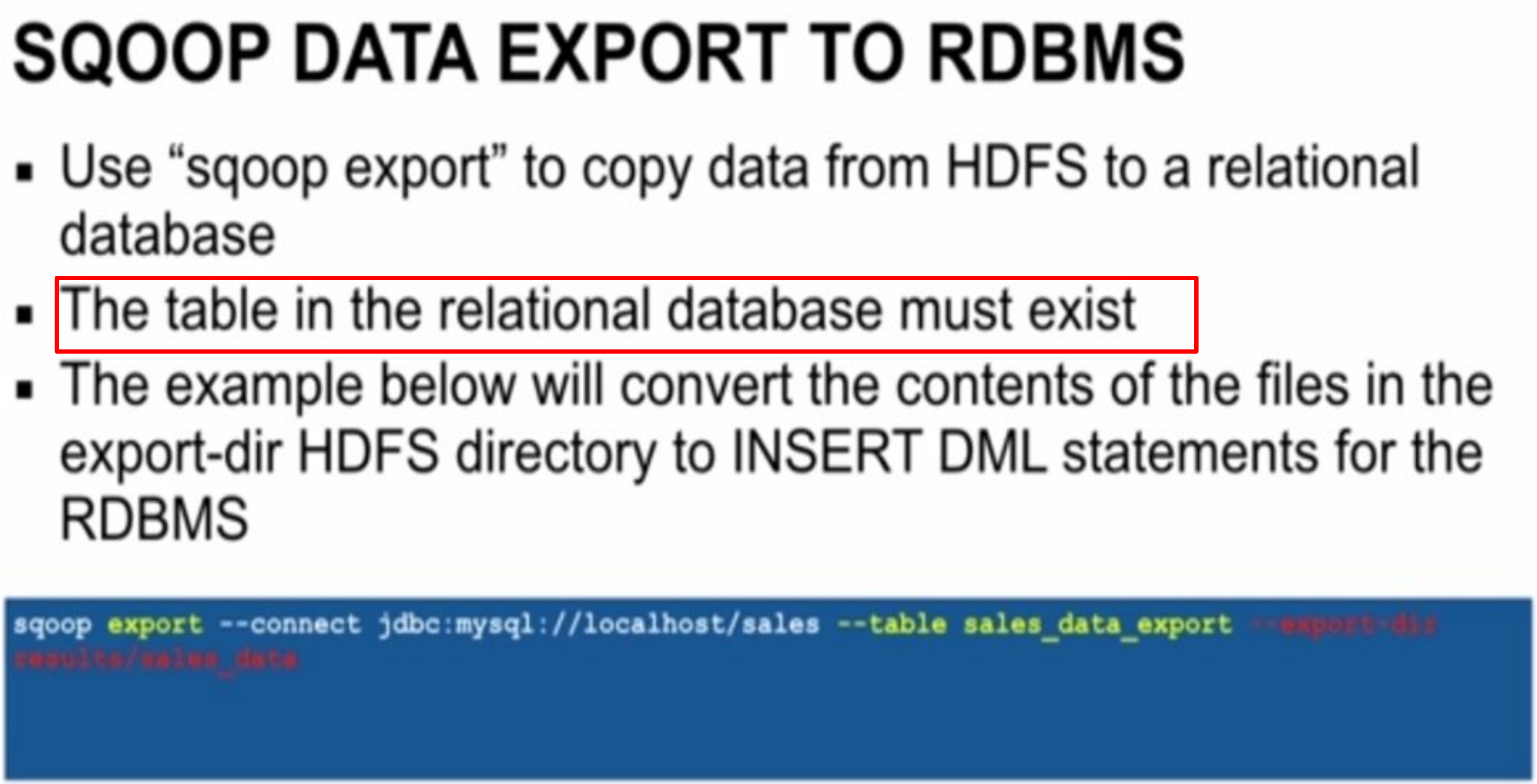
Challenges

- Multiple source ingestion
- Streaming/Real-time data
- Speed of ingestion
- Volume of data
- Change detection



SQOOP DATA EXPORT TO RDBMS

- Sqoop also supports exporting data from Hadoop to a relational database



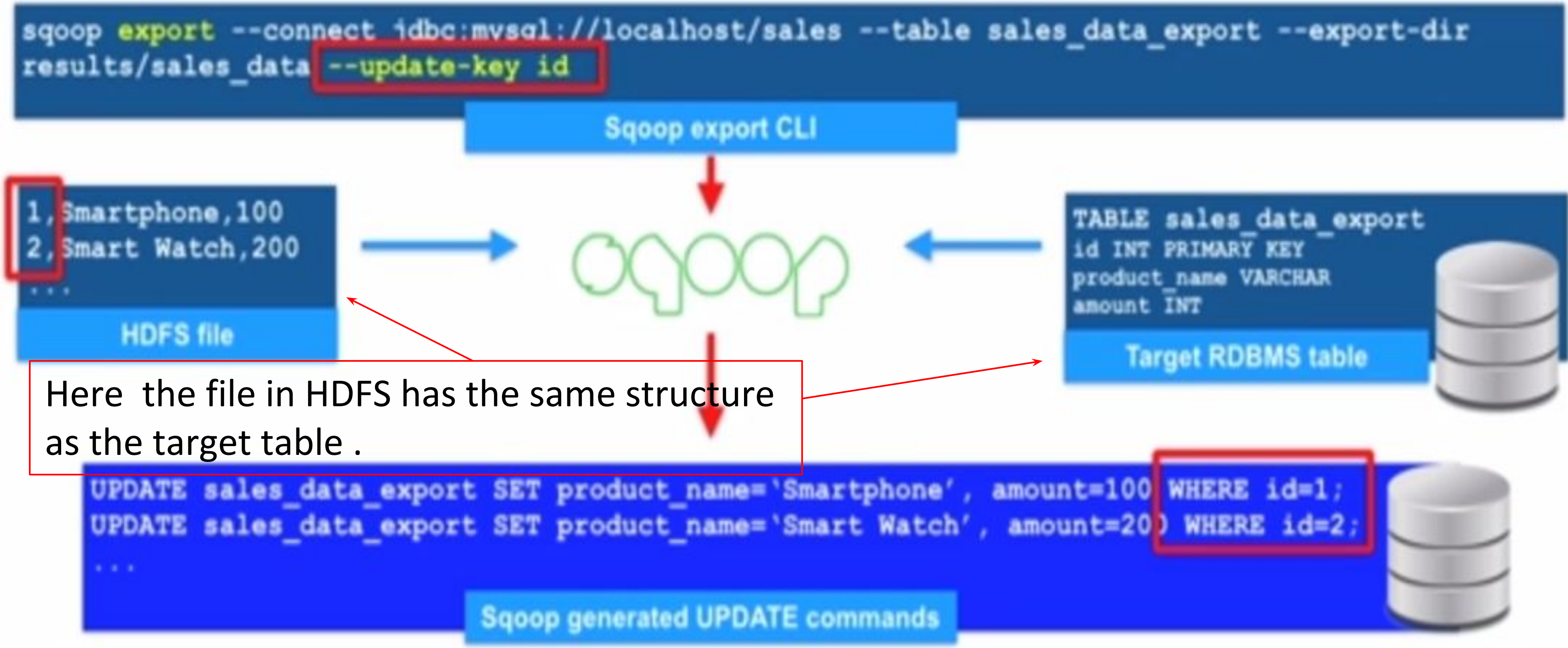
SQOOP DATA EXPORT TO RDBMS

- Use “sqoop export” to copy data from HDFS to a relational database
- The table in the relational database must exist
- The example below will convert the contents of the files in the export-dir HDFS directory to INSERT DML statements for the RDBMS

```
sqoop export --connect jdbc:mysql://localhost/sales --table sales_data_export --export-dir
results/sales_data
```

SQOOP DATA EXPORT TO RDBMS

- Sqoop export also supports updating existing rows in an RDBMS



```
sqoop export --connect jdbc:mysql://localhost/sales --table sales_data_export --export-dir
results/sales_data --update-key id
```

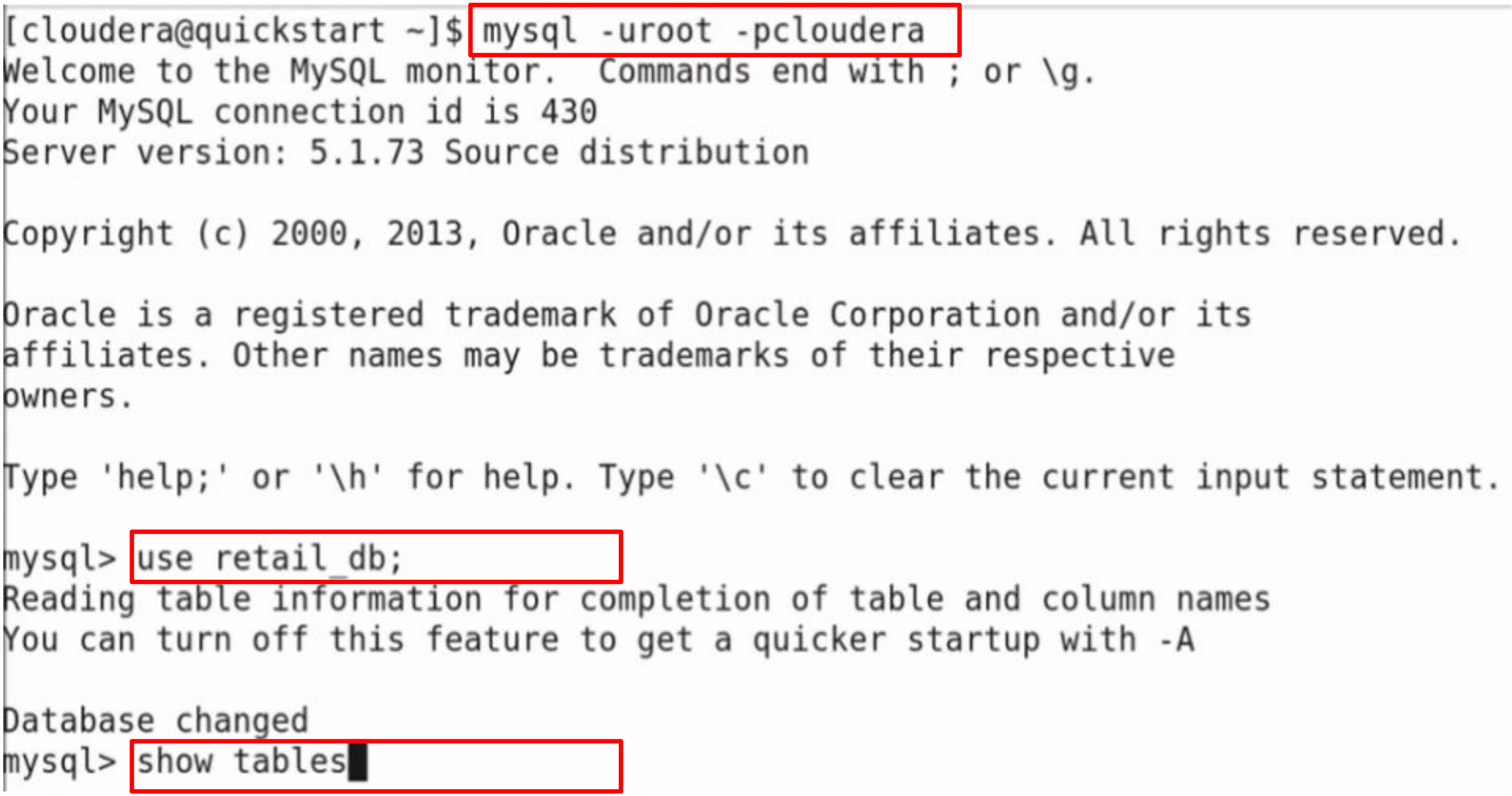
```
1,Smartphone,100
2,Smart Watch,200
...
```

```
TABLE sales_data_export
id INT PRIMARY KEY
product_name VARCHAR
amount INT
```

```
UPDATE sales_data_export SET product_name='Smartphone', amount=100 WHERE id=1;
UPDATE sales_data_export SET product_name='Smart Watch', amount=200 WHERE id=2;
...
```

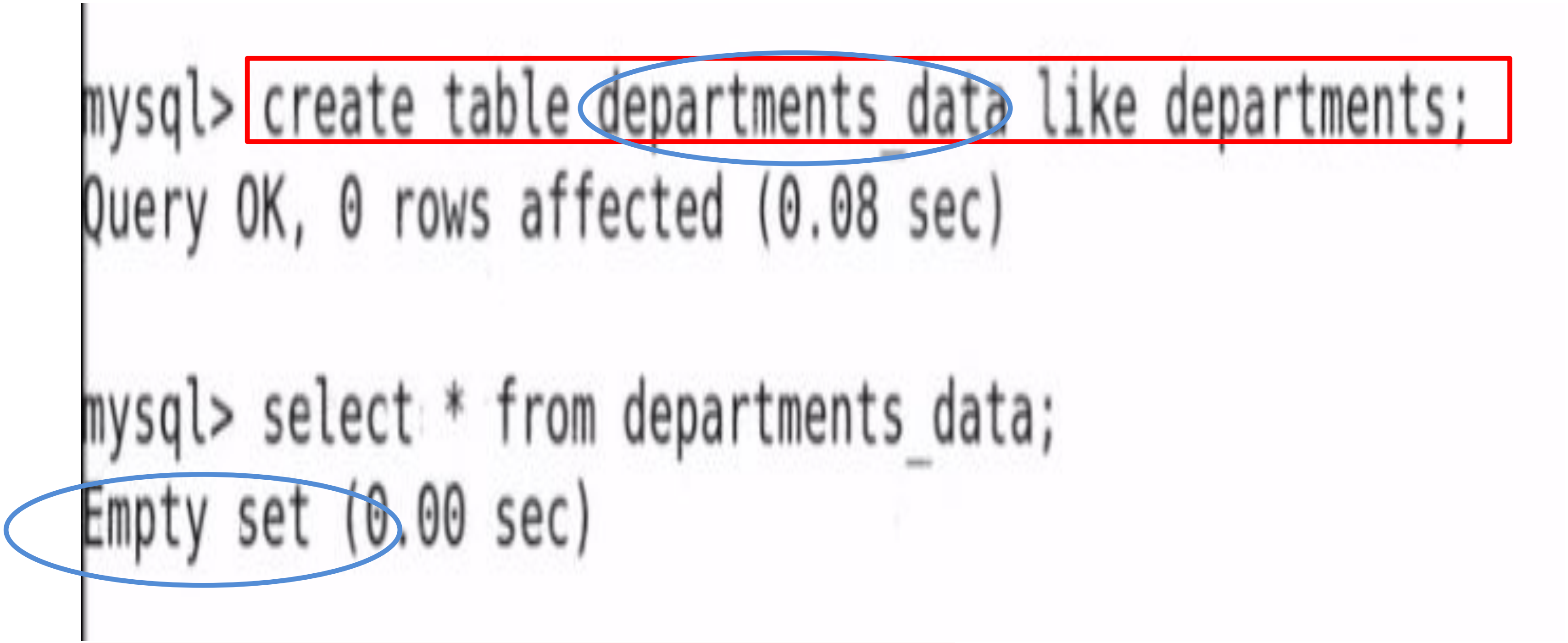
Here the file in HDFS has the same structure as the target table .

Sqoop will generate this code :



```
[cloudera@quickstart ~]$ mysql -uroot -pcloudera
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 430
Server version: 5.1.73 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use retail_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables
```

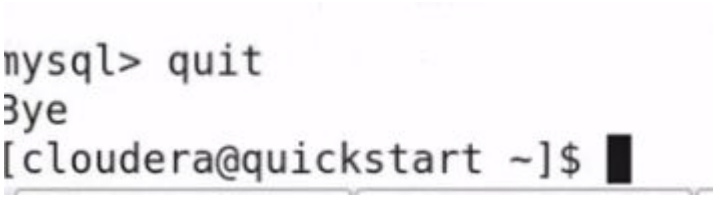
Befor using the export command , First we create a table in mysql where we will put the exported data



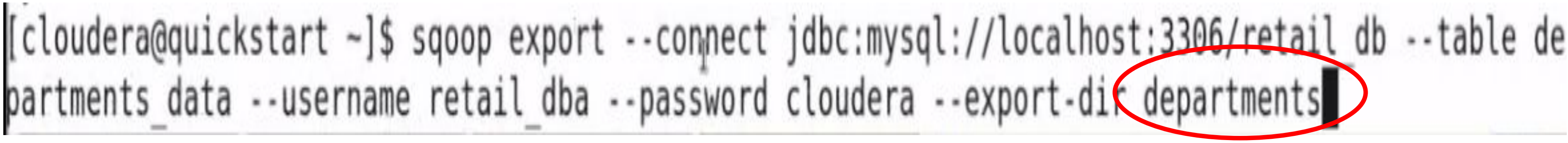
```
mysql> create table departments_data like departments;
Query OK, 0 rows affected (0.08 sec)

mysql> select * from departments_data;
Empty set (0.00 sec)
```

The table is empty

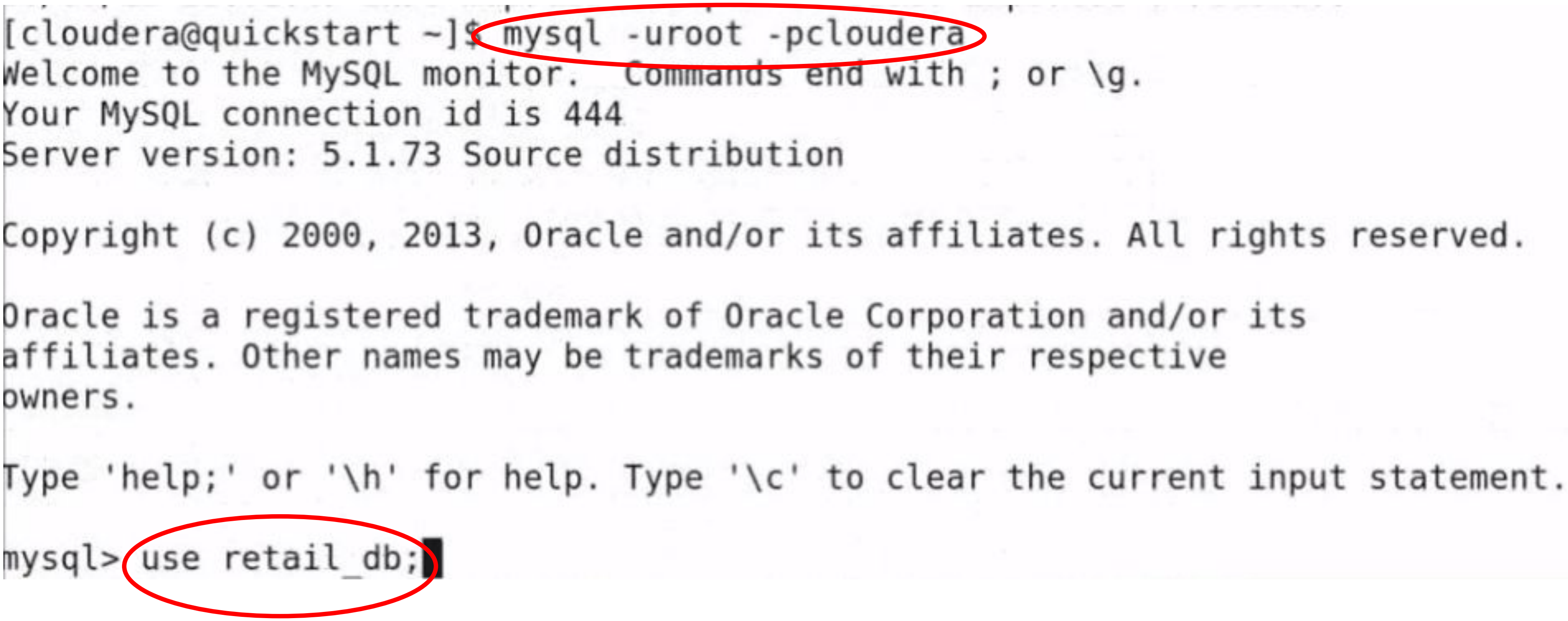
Now we create a table which is the same as the departement (same structure)



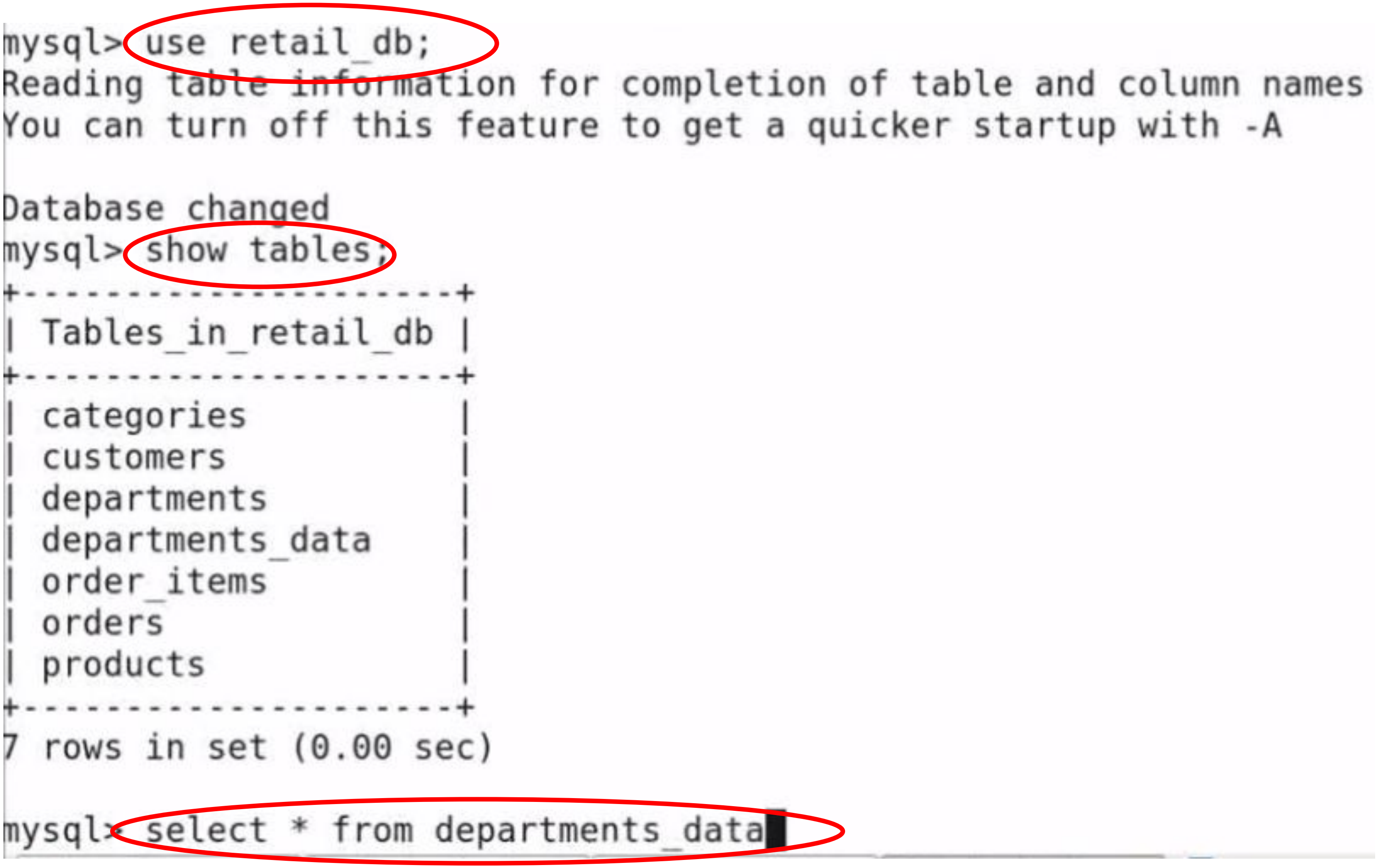
```
mysql> quit
Bye
[cloudera@quickstart ~]$ 
```



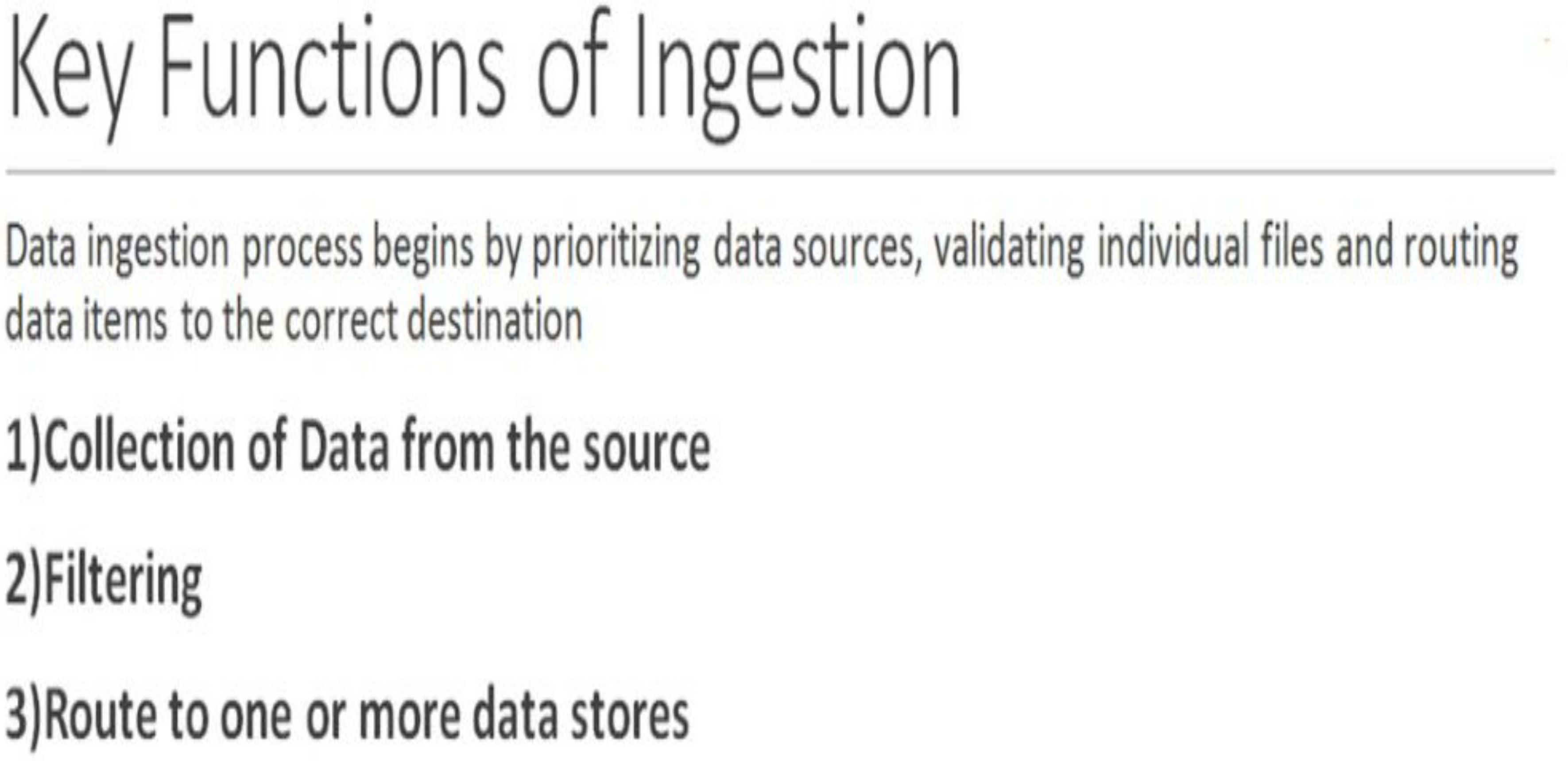
```
[cloudera@quickstart ~]$ sqoop export --connect jdbc:mysql://localhost:3306/retail_db --table de
partments_data --username retail_dba --password cloudera --export-dir departments
```



```
[cloudera@quickstart ~]$ mysql -uroot -pcloudera
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 444
Server version: 5.1.73 Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use retail_db;
```



```
mysql> use retail_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------------+
| Tables_in_retail_db |
+----------------------+
| categories           |
| customers            |
| departments          |
| departments_data     |
| order_items          |
| orders               |
| products             |
+----------------------+
7 rows in set (0.00 sec)

mysql> select * from departments_data
```



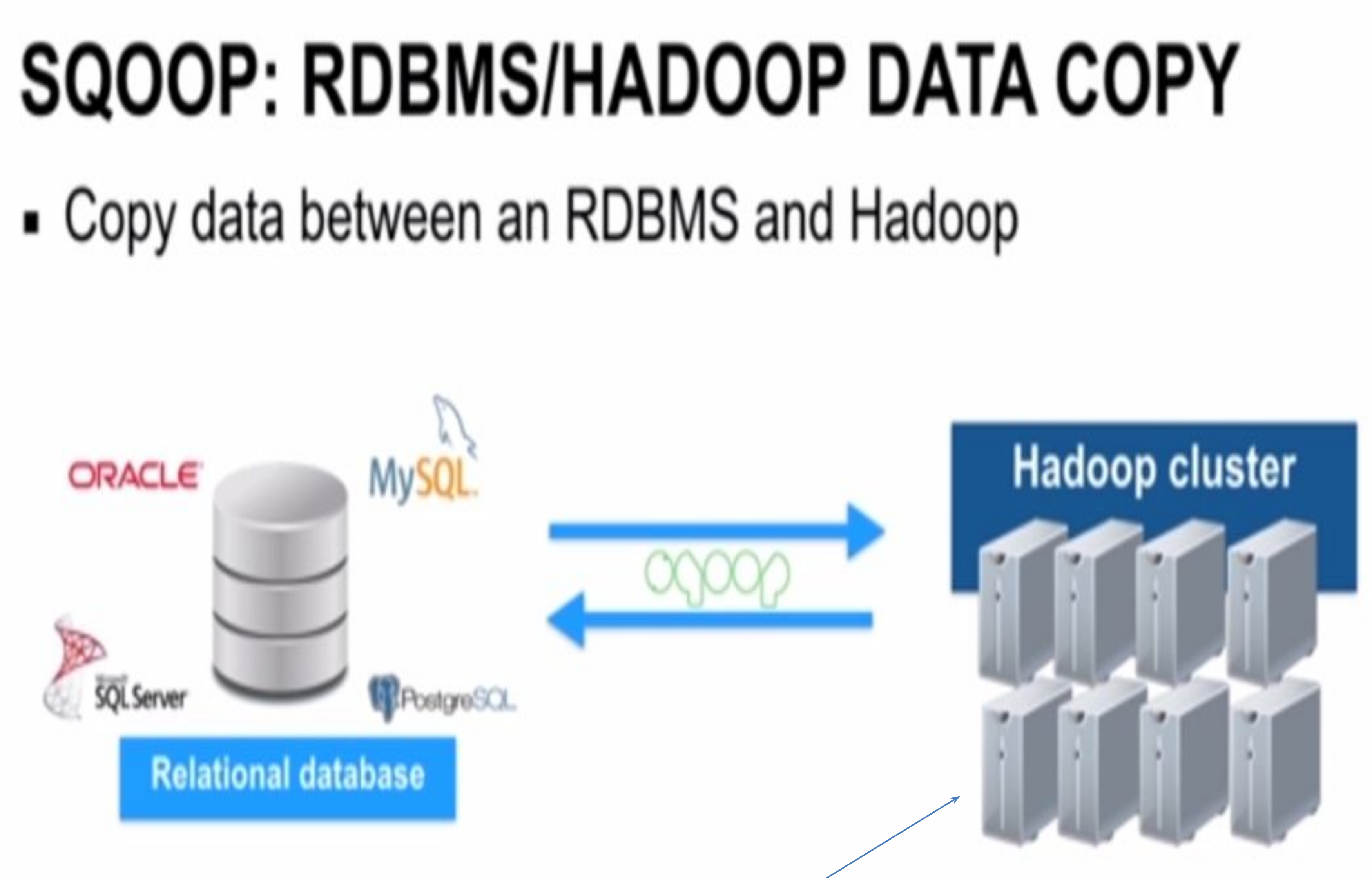
Key Functions of Ingestion

Data ingestion process begins by prioritizing data sources, validating individual files and routing data items to the correct destination

1)Collection of Data from the source

2)Filtering

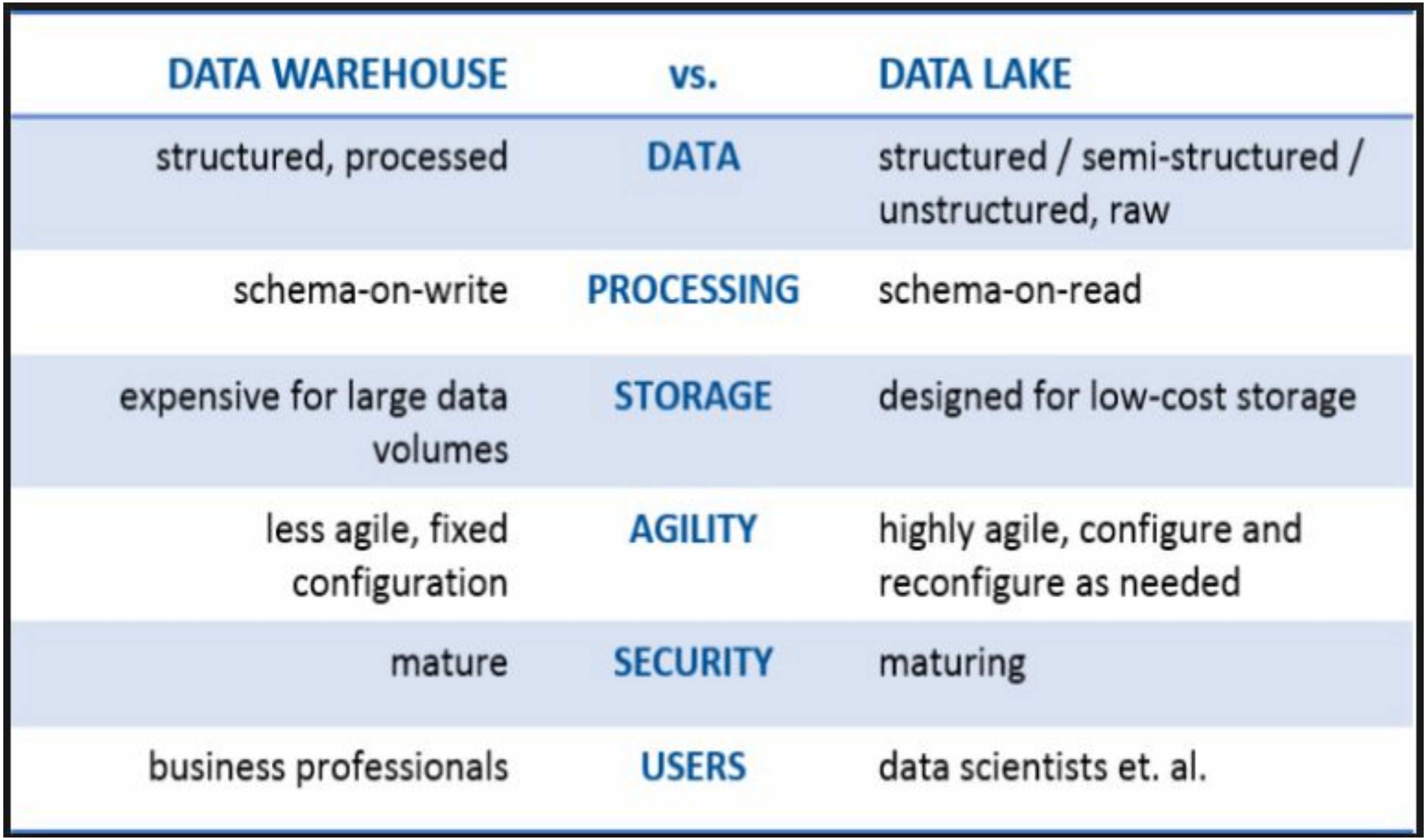
3)Route to one or more data stores



SQOOP: RDBMS/HADOOP DATA COPY

- Copy data between an RDBMS and Hadoop

Hadoop becomes a DATA LAKE :
A Single place to store ALL your
DATA .

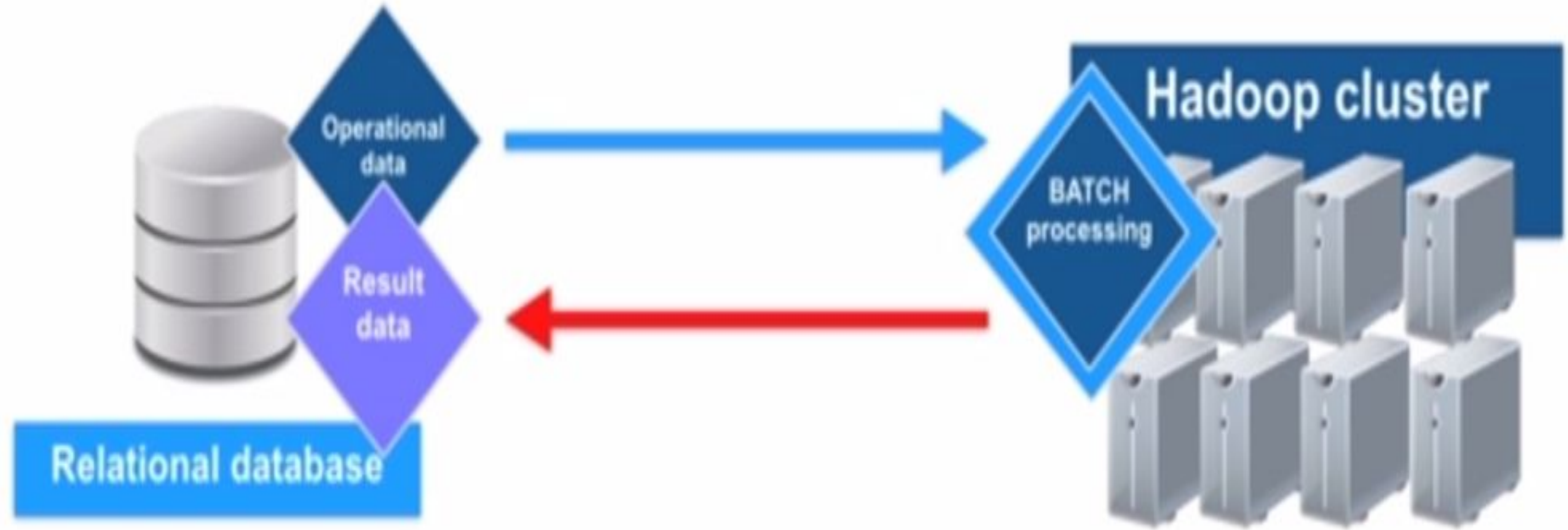


DATA WAREHOUSE	vs.	DATA LAKE
structured, processed	DATA	structured / semi-structured / unstructured, raw
schema-on-write	PROCESSING	schema-on-read
expensive for large data volumes	STORAGE	designed for low-cost storage
less agile, fixed configuration	AGILITY	highly agile, configure and reconfigure as needed
mature	SECURITY	maturing
business professionals	USERS	data scientists et. al.

USE CASE #1: ELT

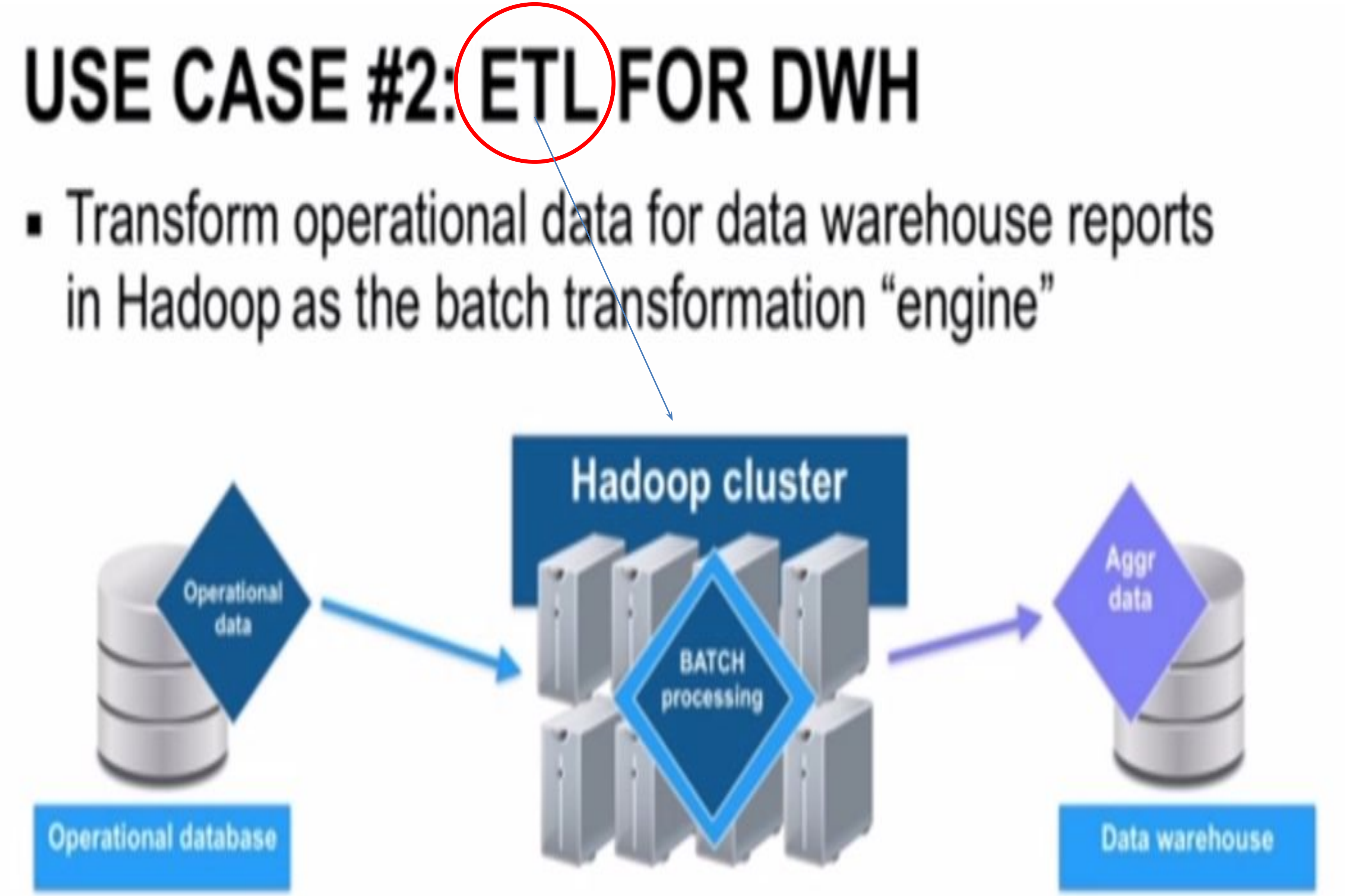
: EXTRACT and LOAD Data from a Data Base and then TRANSFORM and PROCESS it in HADOOP .

- Extract operational data from RDBMS, process in Hadoop, return ***result*** to RDBMS



FRONT END : Handles interactive transactional Workloads better when compared with Hadoop.

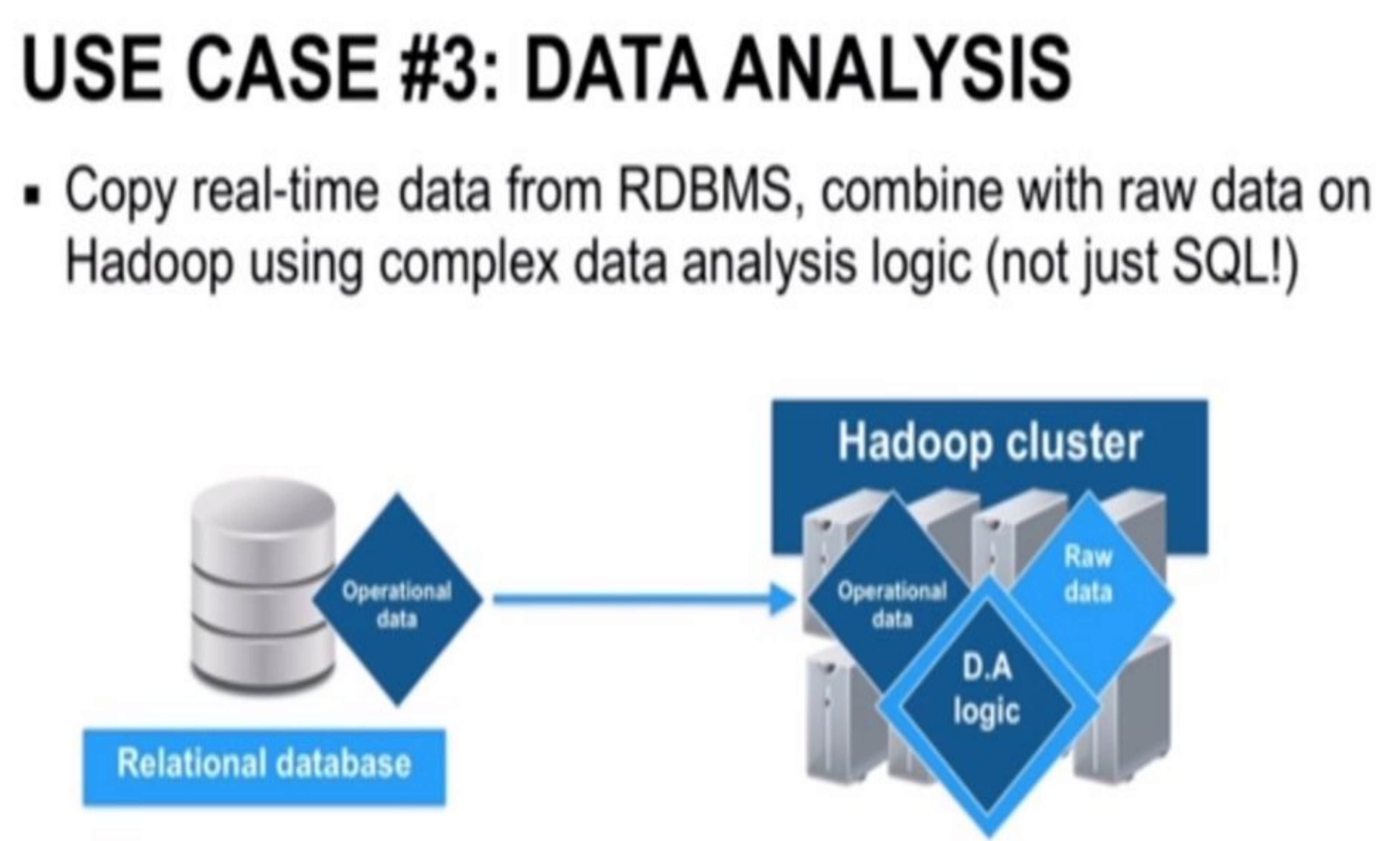
BACK END : BATCH Processing .



USE CASE #2: ETL FOR DWH

- Transform operational data for data warehouse reports in Hadoop as the batch transformation "engine"

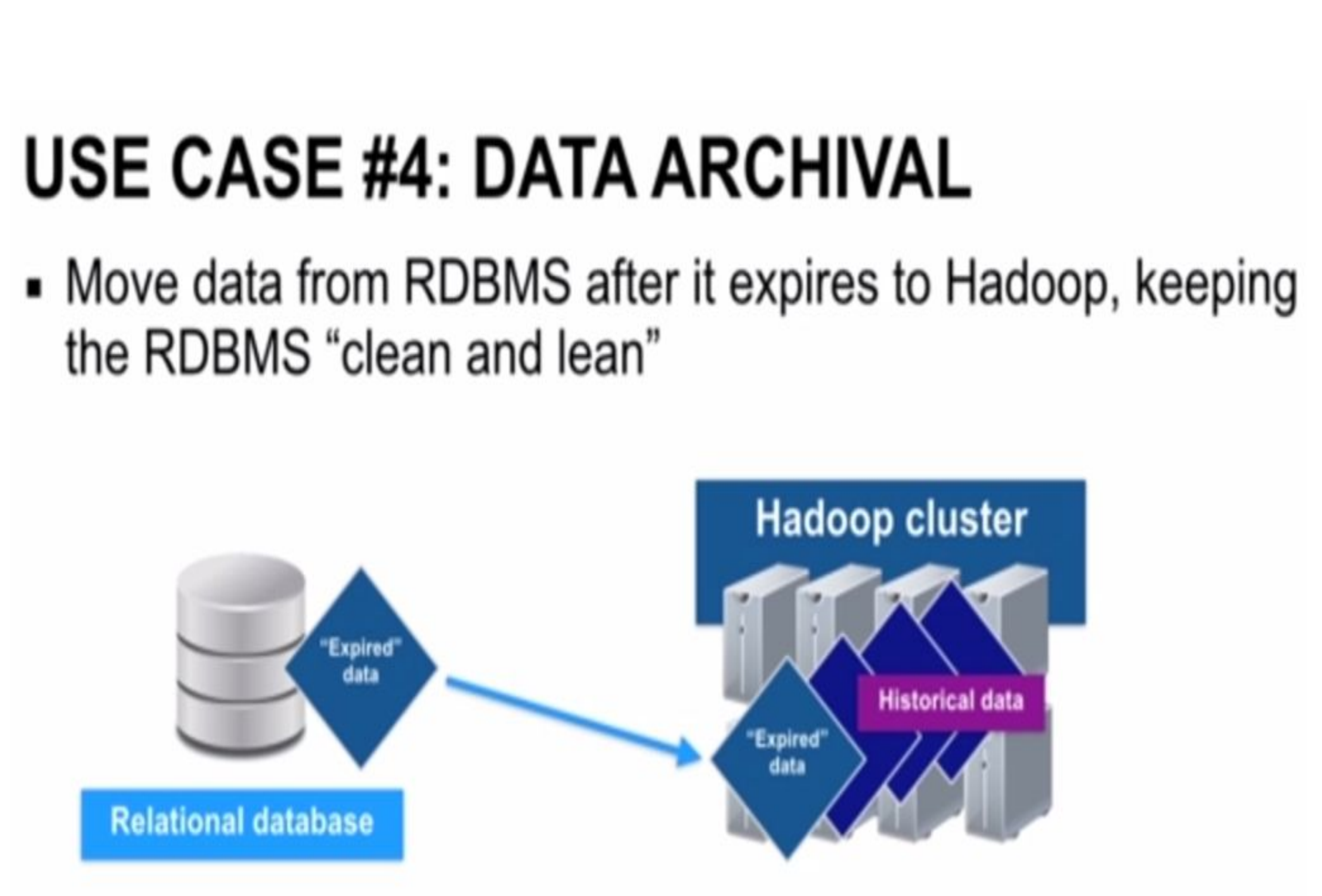
Here Hadoop is used as a Data warehouse TRANSFORMATION Engine : sqoop is used to copy the RESULT to the Data warehouse (instead of bring it back to the original Database).



USE CASE #3: DATA ANALYSIS

- Copy real-time data from RDBMS, combine with raw data on Hadoop using complex data analysis logic (not just SQL!)

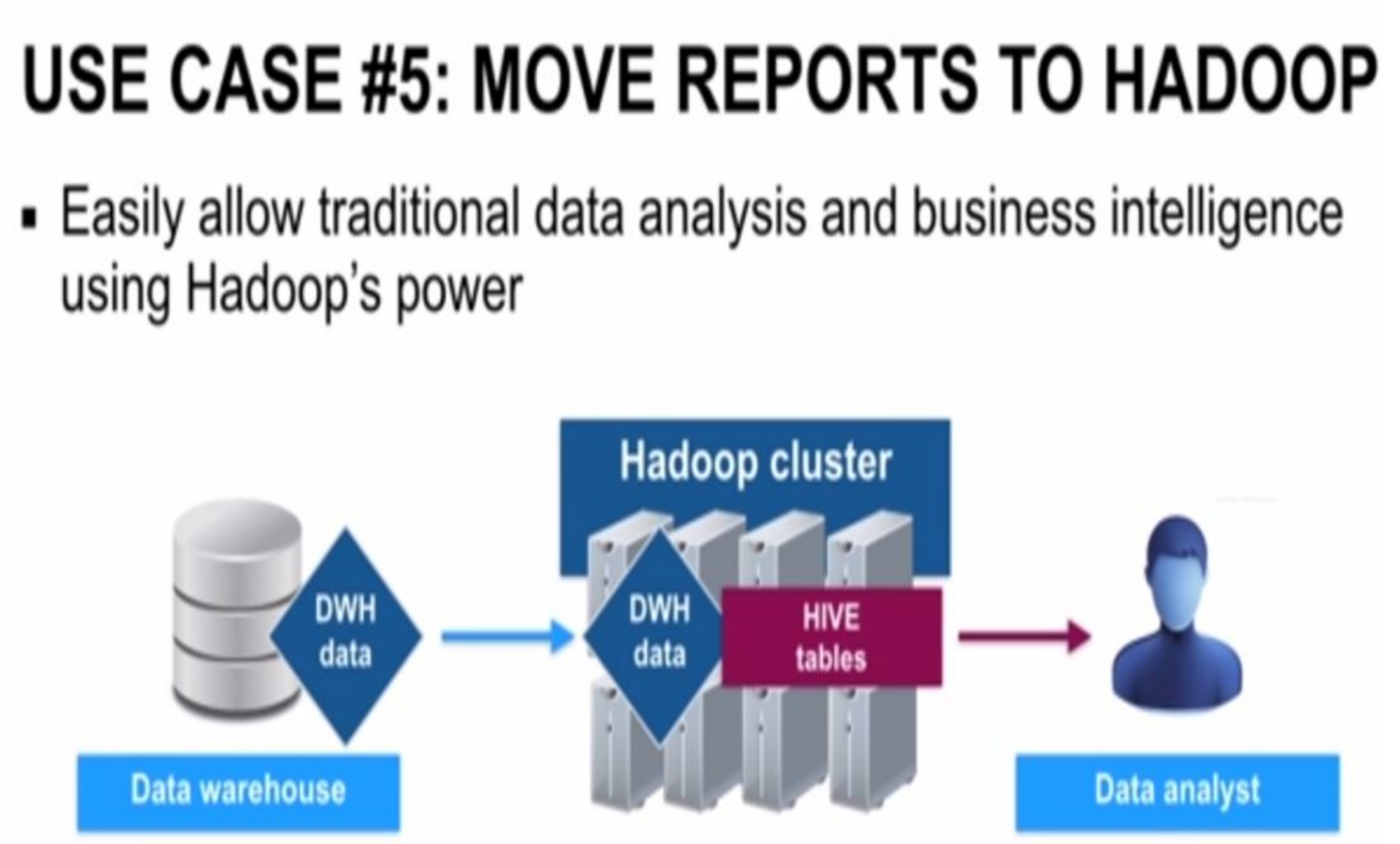
Data copied on regular intervals to Combine New Data with Archived Data in Hadoop for Data Analytics : Trends, prediction…



USE CASE #4: DATA ARCHIVAL

- Move data from RDBMS after it expires to Hadoop, keeping the RDBMS “clean and lean”

Data style accessible for processing Compared when saved in Tapes (stored some where).



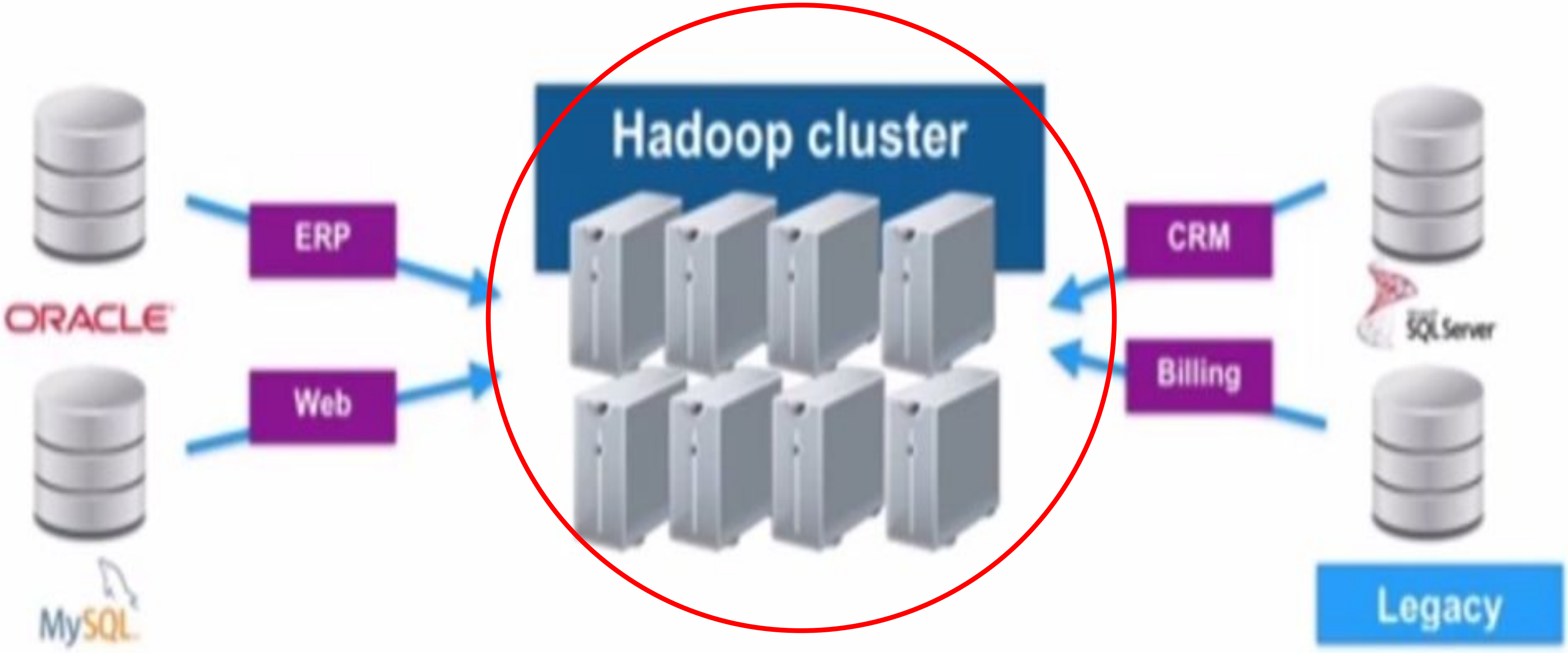
USE CASE #5: MOVE REPORTS TO HADOOP

- Easily allow traditional data analysis and business intelligence using Hadoop's power

Here, we use sqoop to copy data from Relational Database to Hadoop -> Create Hive tables on top of this data = Easy Migration of the existing data warehouse to Hadoop.

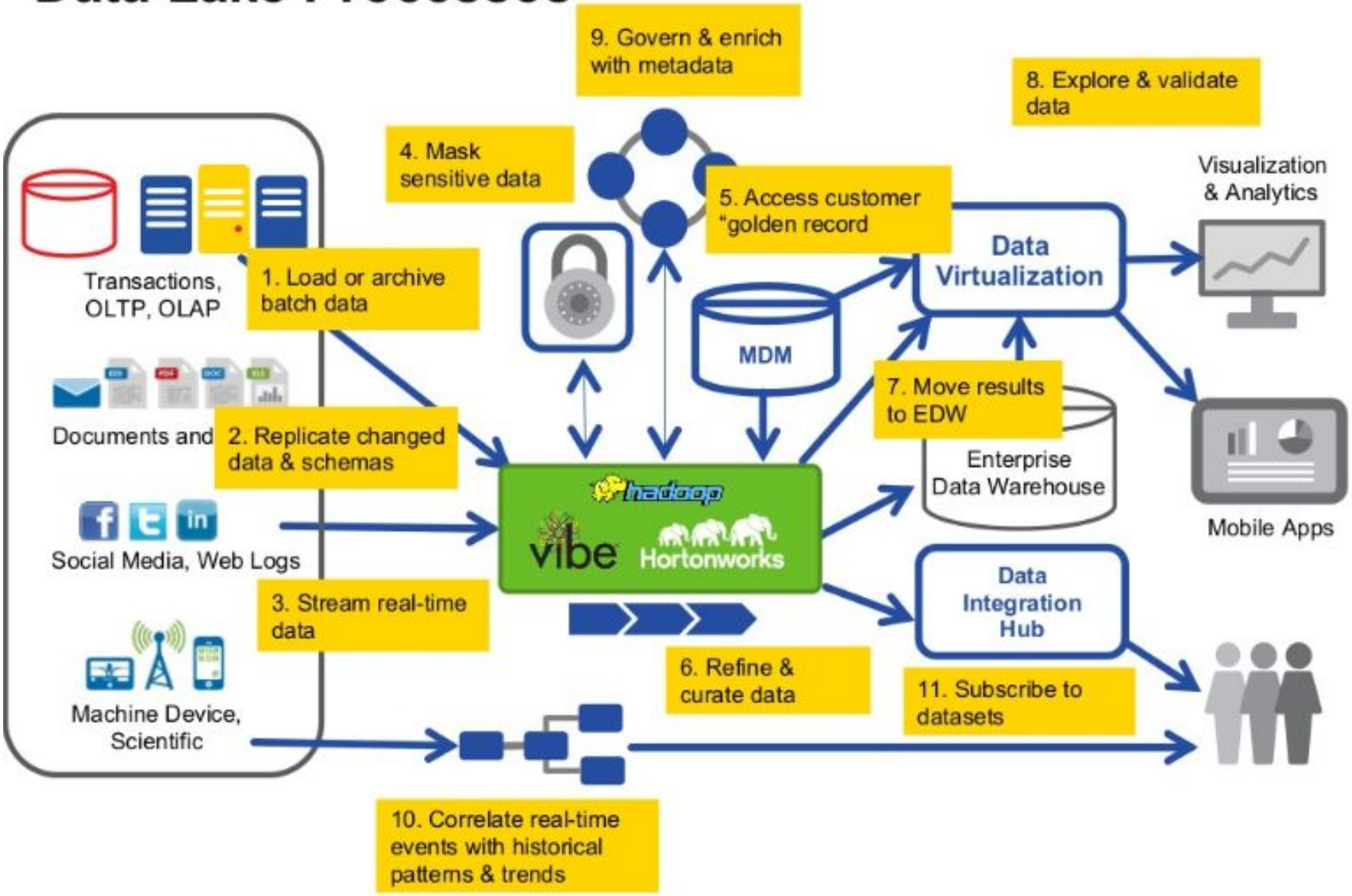
USE CASE #6: DATA CONSOLIDATION

- Integrate data from various organizational “data stores” to Hadoop for various data processing requirements



DATA LAKE

Data Lake Processes



1. Load or archive batch data

2. Replicate changed data & schemas

3. Stream real-time data

4. Mask sensitive data

5. Access customer "golden record

6. Refine & curate data

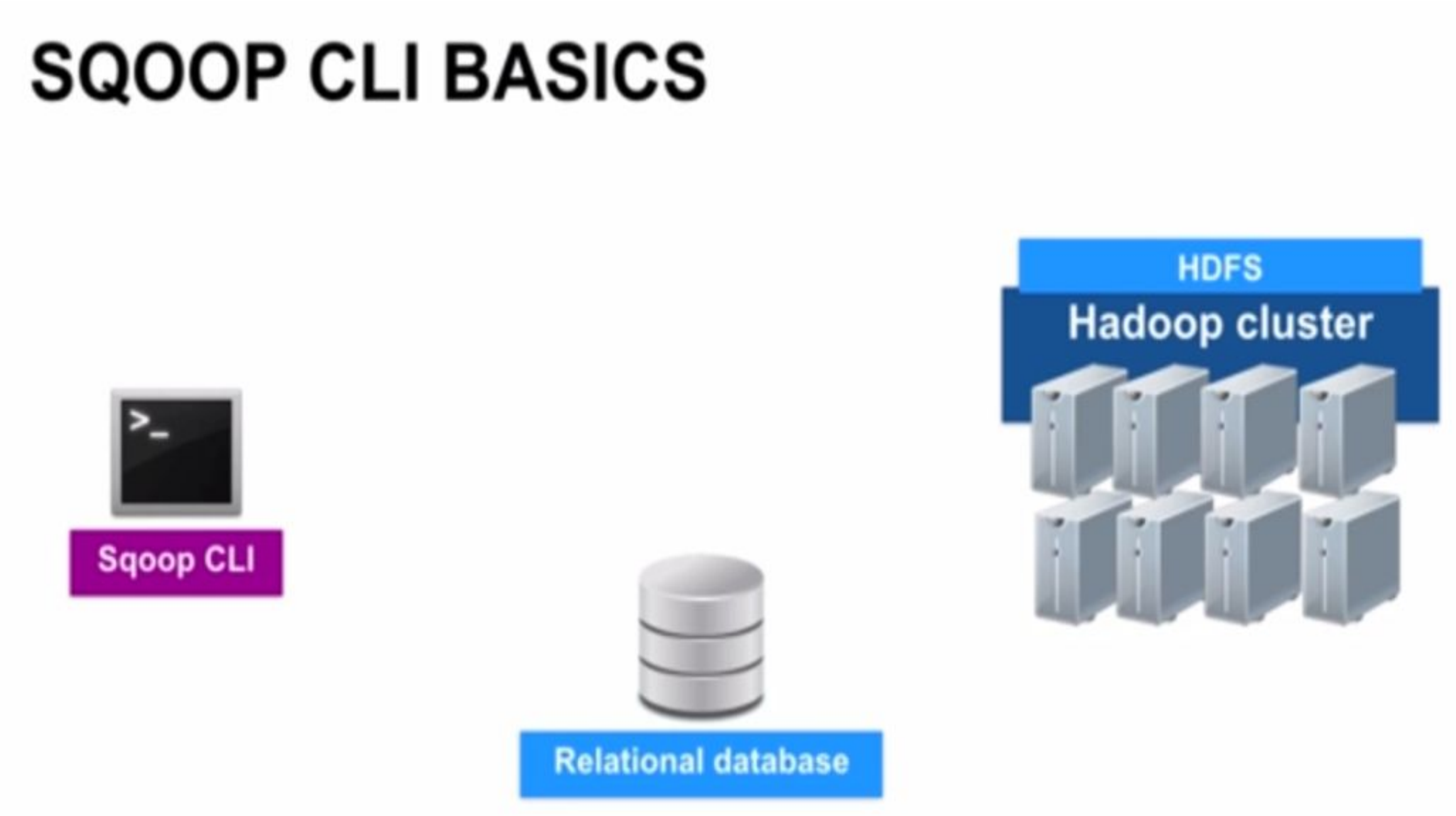
7. Move results to EDW

8. Explore & validate data

9. Govern & enrich with metadata

10. Correlate real-time events with historical patterns & trends

11. Subscribe to datasets



SQOOP CLI BASICS

Data acquisition layer

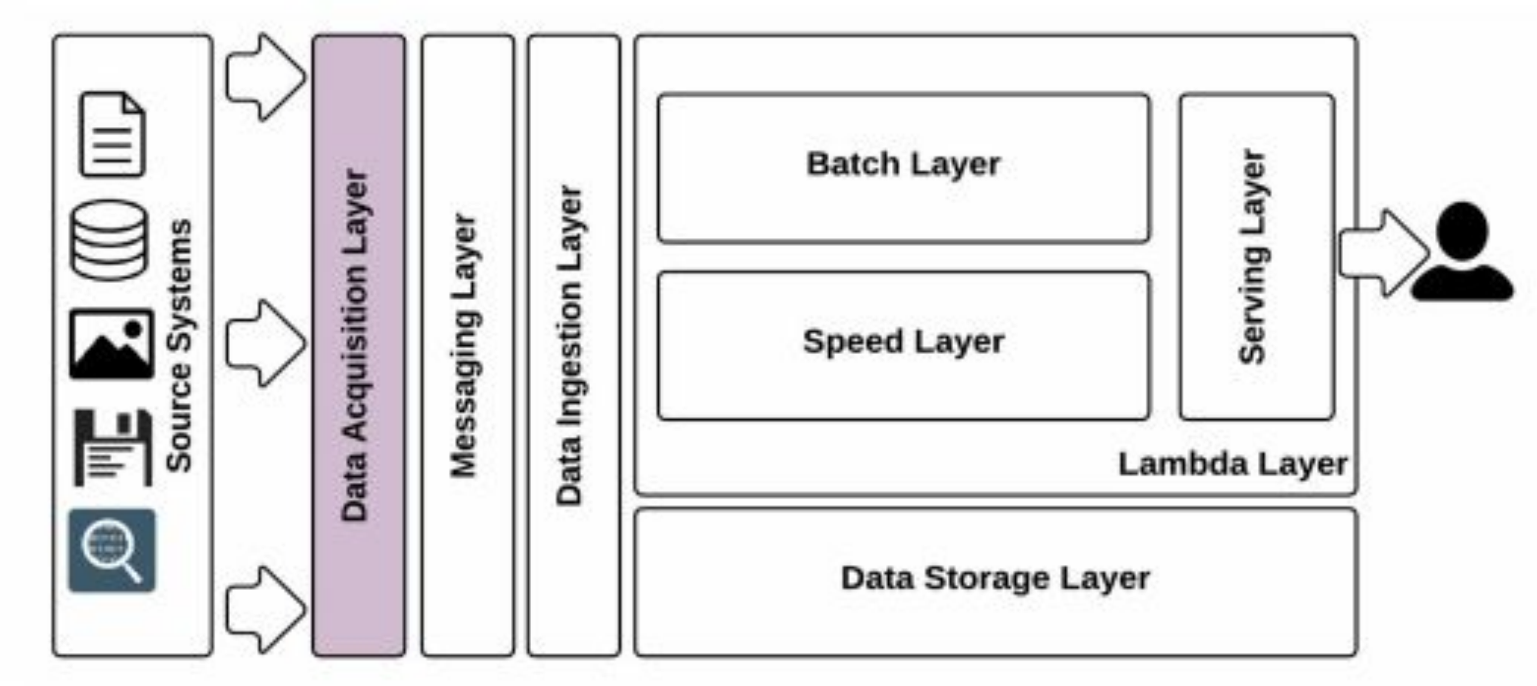


Figure 01: Data lake - data acquisition layer

Data acquisition of batch data - technology mapping

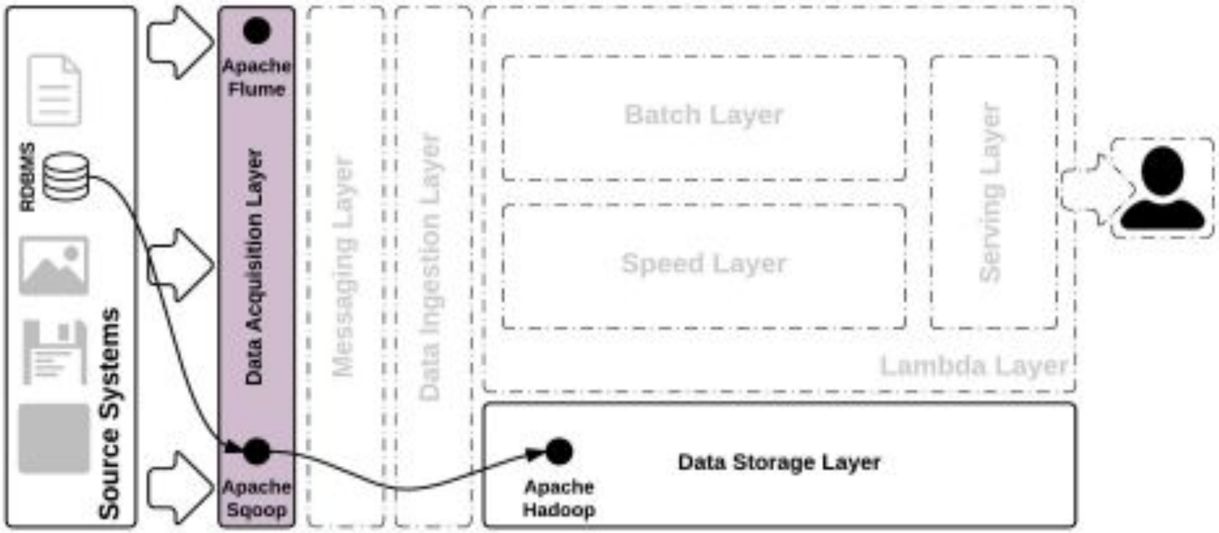


Figure 02: Technology mapping for acquisition layer

- While analyzing data, data analysts often have to gather data from different sources such as external relational databases and bring it into HDFS for processing. Also, after processing data in Hadoop, analysts may also send the data from HDFS back to some external relational data stores. Apache Sqoop is just the tool for such requirements.

- Sqoop is used to transfer data between HDFS and relational database systems such as MySQL and Oracle.

From cloudera Admin

- Sqoop expects the external database to define the schema for the imports to HDFS. Here, the schema refers to metadata or the structure of the data. The importation and exportation of data in Sqoop is done using MapReduce, thereby leveraging the robust features of MapReduce to perform its operations.

- When importing data from an external relational database, Sqoop takes the table as an input, reads the table row by row, and generates output files that are placed in HDFS. The Sqoop import runs in a parallel model (MapReduce), generating several output files for a single input table.
-

  The following diagram shows the two-way flow of data from RDBMS to HDFS and vice versa:

- Once the data is in HDFS, analysts process this data, which generates subsequent output files. These results, if required, can be exported to an external relational database system using Sqoop.
- Sqoop reads delimited files from HDFS, constructs database records, and inserts them into the external table.
- Sqoop is a highly configurable tool where you can define the columns that need to be imported/exported to and from HDFS. All operations in Sqoop are done using the command-line interface.
- Sqoop 2, a newer version of Sqoop, now provides an additional web user interface to perform the importations and exportations.
- Sqoop is a client-side application whereas the new Sqoop 2 is a server-side (Sqoop server) application.
- The Sqoop 2 server also provides a REST API for other applications to easily talk to Sqoop 2.

Problem :

You have a table in a relational database (e.g., MySQL) and you need to transfer the table's contents into Hadoop's Distributed File System (HDFS).

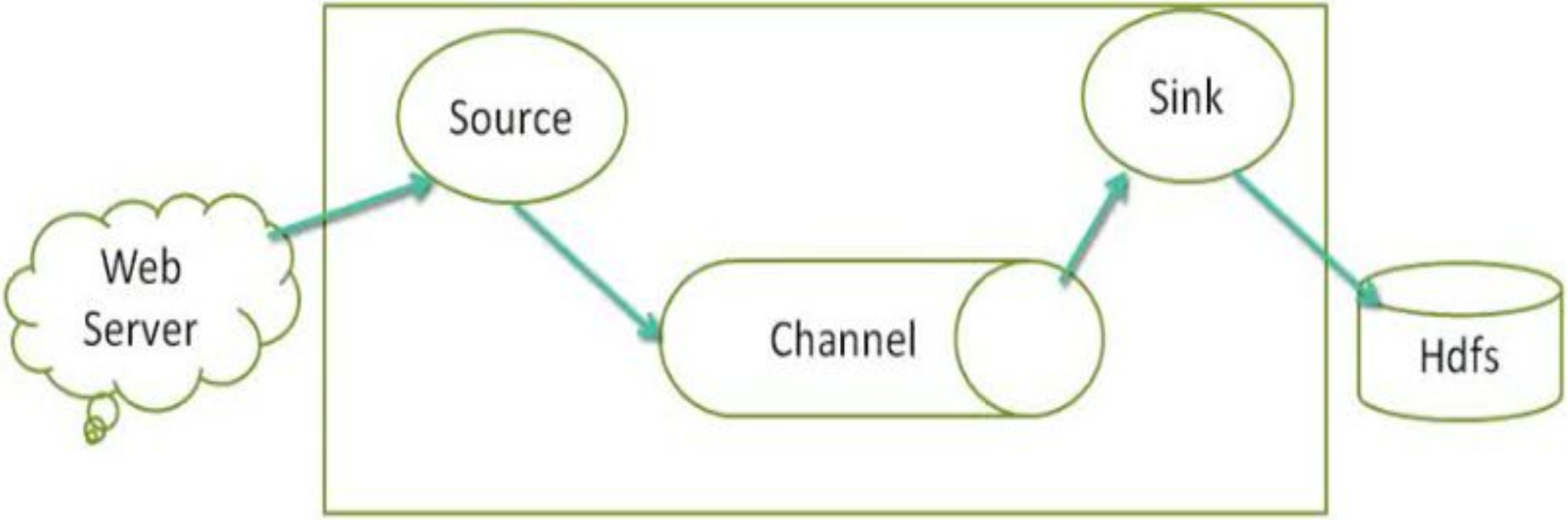
Solution:

Importing one table with Sqoop is very simple: you issue the Sqoop import command and specify the database credentials and the name of the table to transfer :

```
sqoop import \
--connect jdbc:mysql://mysql.example.com/sqoop \
--username sqoop \
--password sqoop \
--table cities
```

Apache Flume

- Distributed Data collection service
- Getting **Streaming event data** from different sources
- Apache Flume is useful in processing log-data.
- Moving large amounts of **log** data from many different sources to a centralized data store.

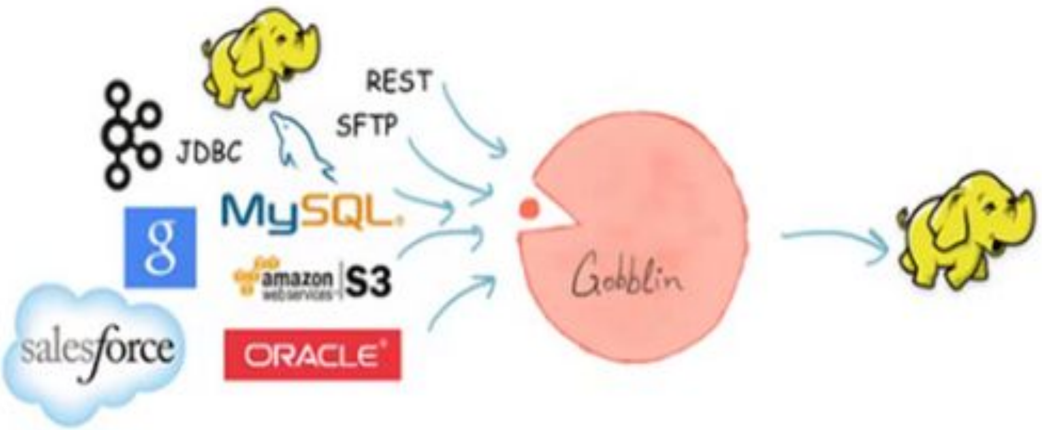


Gobblin

- Gobblin is Universal data ingestion framework for extracting, transforming, and loading large volume of data from variety of data source.

- Gobblin handles the common routine task required for all data ingestion ETLs Including job, task scheduling, task partitioning, error handling, state management, data quality check, data publishing etc.

- It's a LinkedIn 's unified Data Ingestion Platform and an open source.


REST
SFTP
JDBC
MySQL
amazon web services S3
salesforce
ORACLE
Gobblin