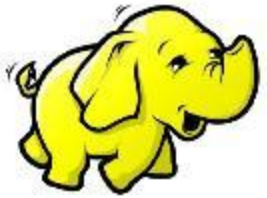
edureka!

Big Data & Hadoop



# Course Topics

edureka!

- ✓ Module 1
  - ✓ Understanding Big Data
  - ✓ Hadoop Architecture
- ✓ Module 2
  - ✓ Hadoop Cluster Configuration
  - ✓ Data loading Techniques
  - ✓ Hadoop Project Environment
- ✓ Module 3
  - ✓ Hadoop MapReduce framework
  - ✓ Programming in Map Reduce
- ✓ Module 4
  - ✓ Advance MapReduce
  - ✓ MRUnit testing framework
- ✓ Module 5
  - ✓ Analytics using Pig
  - ✓ Understanding Pig Latin
- ✓ Module 6
  - ✓ **Analytics using Hive**
  - ✓ **Understanding HIVE QL**
- ✓ Module 7
  - ✓ Advance Hive
  - ✓ NoSQL Databases and HBASE
- ✓ Module 8
  - ✓ Advance HBASE
  - ✓ Zookeeper Service
- ✓ Module 9
  - ✓ Hadoop 2.0 – New Features
  - ✓ Programming in MRv2
- ✓ Module 10
  - ✓ Apache Oozie
  - ✓ Real world Datasets and Analysis
  - ✓ Project Discussion

# Topics for Today

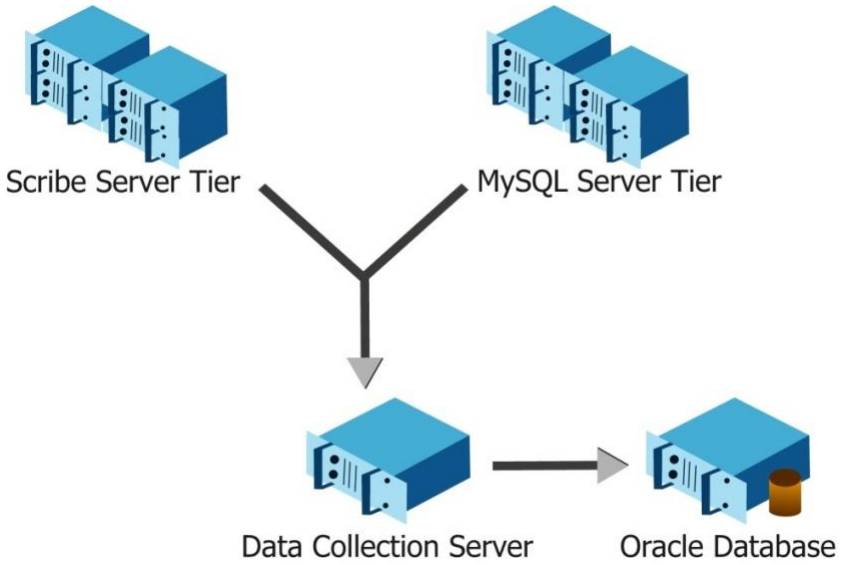
edureka!

- ✓ What is Hive?
- ✓ Where to use Hive
- ✓ Why go for Hive when PIG is there?
- ✓ Let's start - Hive Architecture
- ✓ Hive Components
- ✓ Hive Background
- ✓ How Facebook uses Hive
- ✓ Limitation of Hive
- ✓ Abilities of Hive Query Language
- ✓ Differences with Traditional RDBMS
- ✓ Hive Types
- ✓ Examples

# Hive Background

edureka!

- ✓ Started at **Facebook**.
- ✓ Data was collected by nightly cron jobs into **Oracle DB**.
- ✓ "**ETL**" via hand-coded python.
- ✓ Grew from **10s of GBs** (2006) to **1 TB/day** new data (2007), now 10x that.

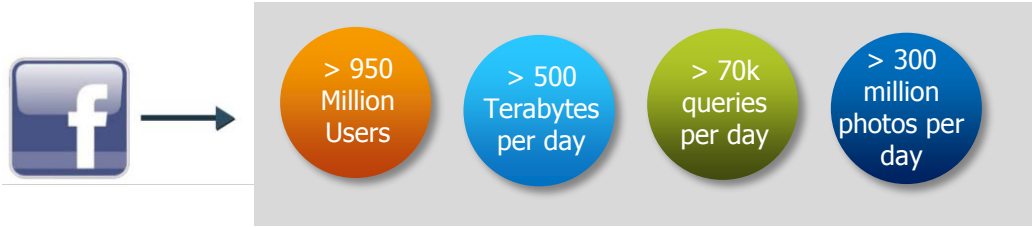


# Hive Use Case @ Facebook

# Hive Use Case @ Facebook

edureka!

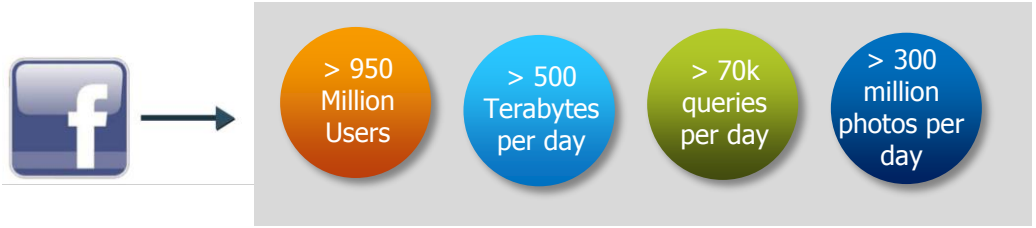
## Challenge...



# Hive Use Case @ Facebook

edureka!

## Challenge...

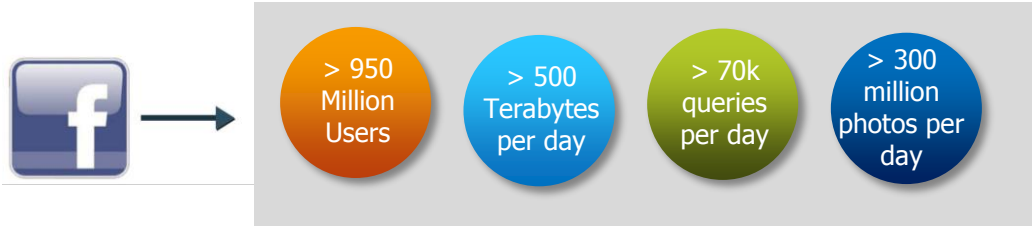



**Traditional RDBMS...X**

# Hive Use Case @ Facebook

edureka!

**Challenge...**




**Traditional RDBMS...** X

**Solution...**

# Hive Use Case @ Facebook

edureka!

## Challenge...

- > 950 Million Users
- > 500 Terabytes per day
- > 70k queries per day
- > 300 million photos per day

**Traditional RDBMS...** X

## Solution...


- Hard to program
- Users know SQL well

# Hive Use Case @ Facebook

edureka!

## Challenge...

- > 950 Million Users
- > 500 Terabytes per day
- > 70k queries per day
- > 300 million photos per day

**Traditional RDBMS...** X

## Solution...

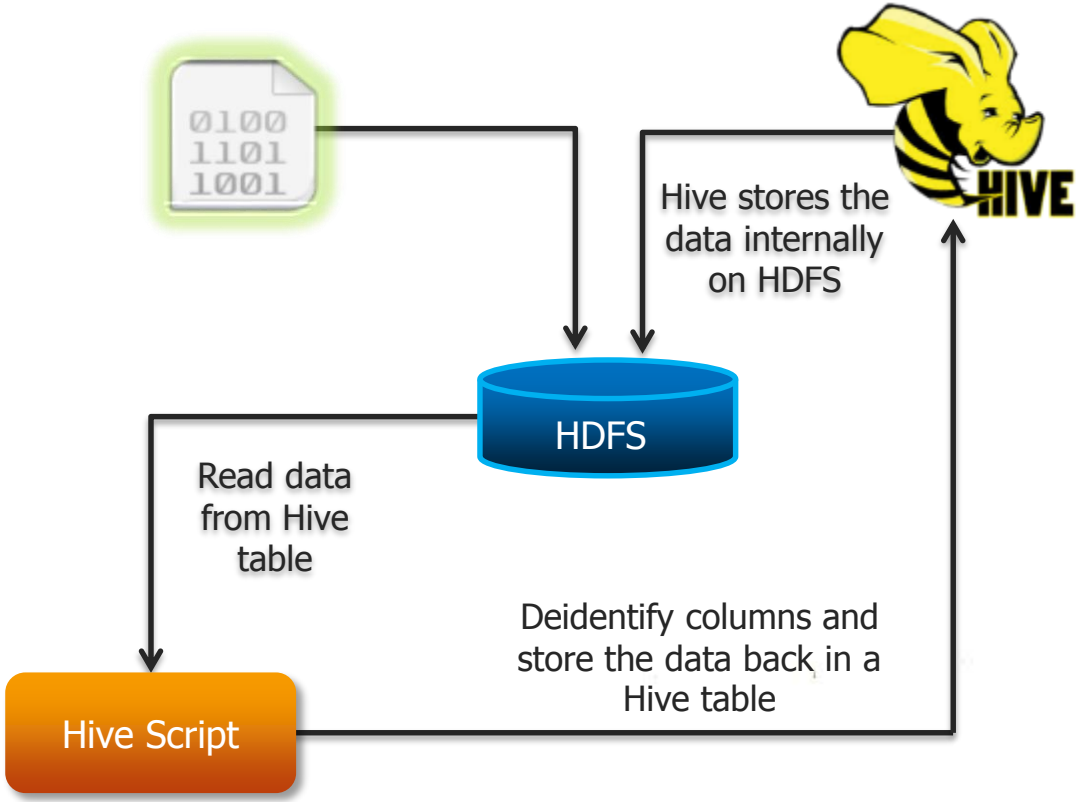
- Hard to program
- Users know SQL well

- Tables can be partitioned and bucketed
- Schema flexibility and evolution
- Easy to plug-in custom mapper/ reducer code
- JDBC/ODBC drivers are available
- HIVE tables can be defined directly on HDFS
- Extensible: Types, Formats, Functions, Scripts

# Use Case In Healthcare

edureka!

## Load CSV file into Hive



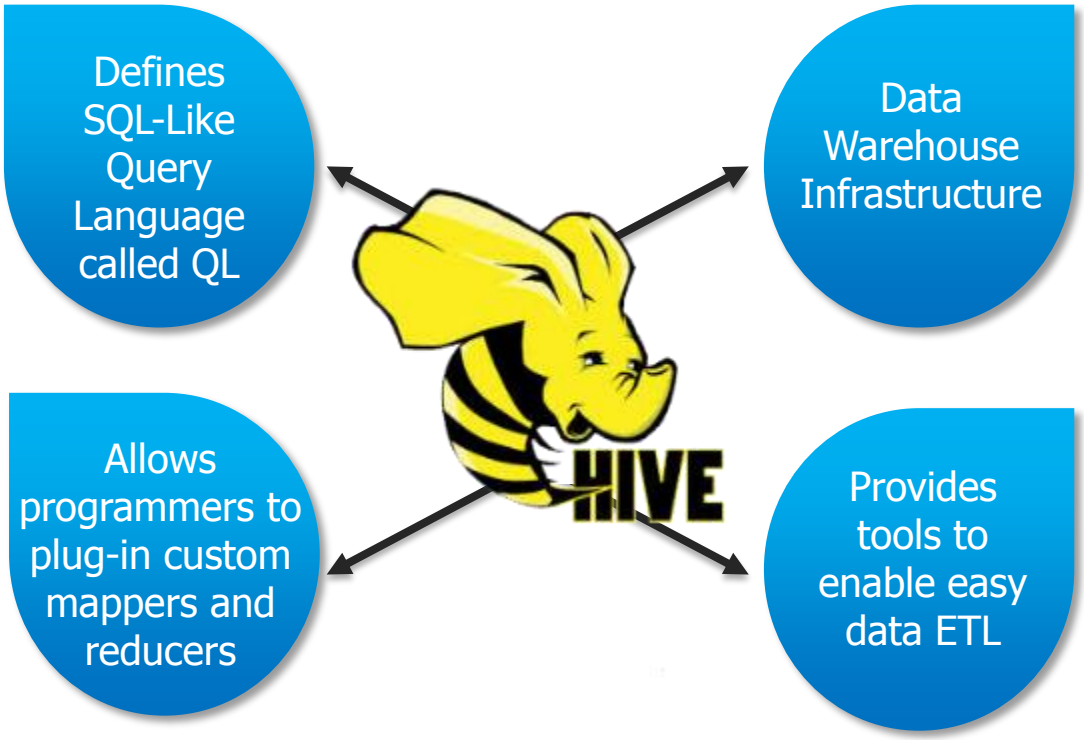
# What Is Hive?

edureka!

- ✓ Data Warehousing package built on top of Hadoop
- ✓ Used for data analysis
- ✓ Targeted towards users comfortable with SQL
- ✓ It is similar to SQL and called HiveQL
- ✓ For managing and querying structured data
- ✓ Abstracts complexity of Hadoop
- ✓ No need learn java and Hadoop APIs
- ✓ Developed by Facebook and contributed to community
- ✓ Facebook analyzed several Terabytes of data everyday using Hive

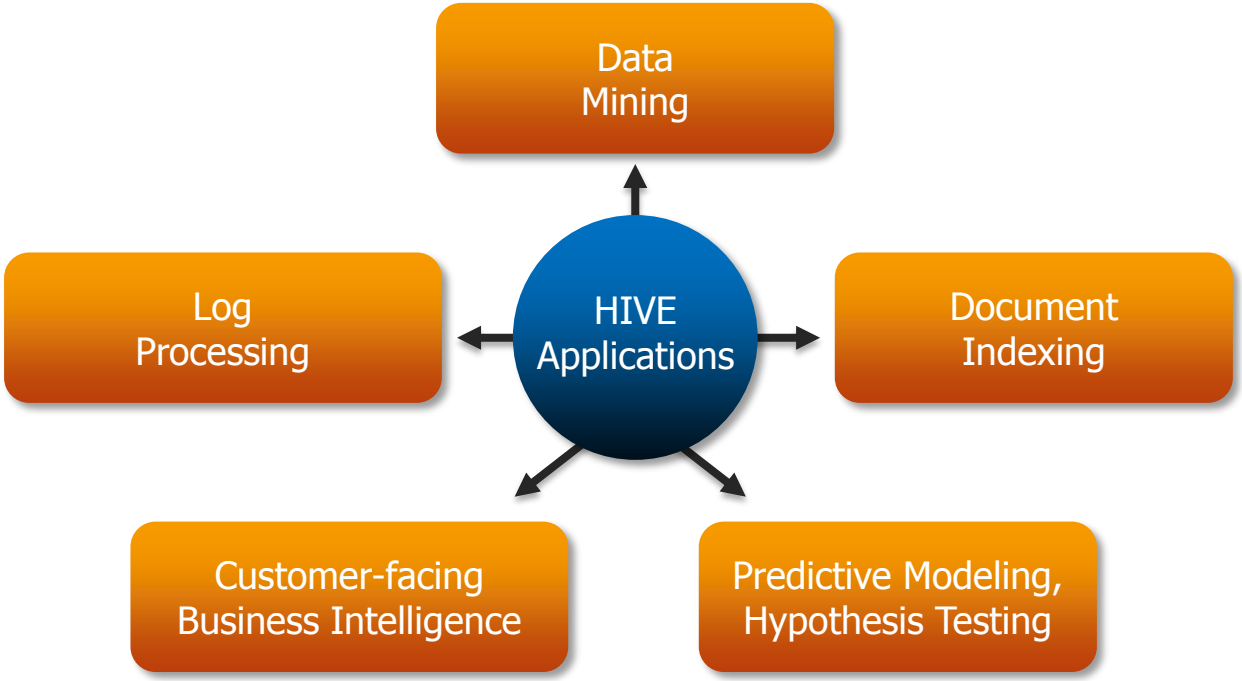
# What Is Hive?

edureka!



# Where To Use Hive?

edureka!



# Why Go For Hive When Pig Is There?

edureka!


**PigLatin:**

- ✓ Procedural data-flow language
  A = load 'mydata';
  Dump A;
- ✓ Pig is used by Programmers and Researchers

**HiveQL:**

- ✓ Declarative SQLish language
  Select * from 'mytable';
- ✓ Hive is used by Analysts generating daily reports

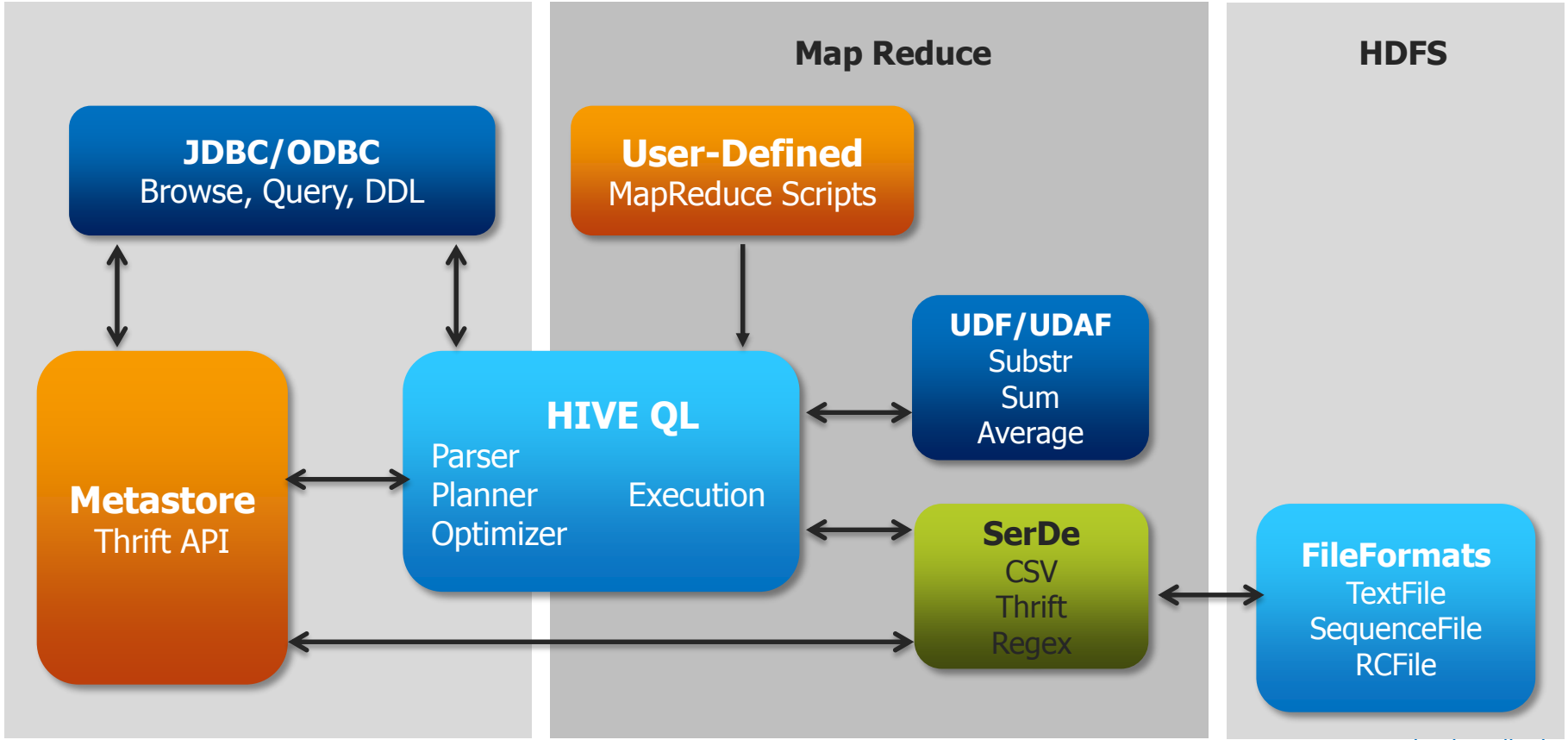
# Why Go For Hive When Pig Is There?

edureka!

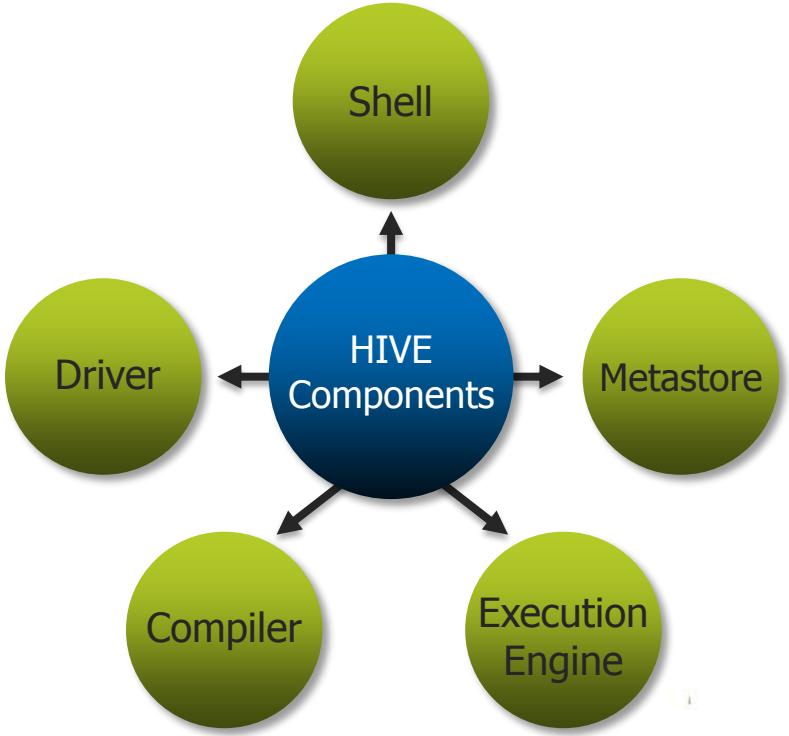
| Features | Hive | Pig |
|---|---|---|
| Language | SQL-like | PigLatin |
| Schemas/Types | Yes (explicit) | Yes (implicit) |
| **Partitions** | **Yes** | **No** |
| Server | Optional (Thrift) | No |
| User Defined Functions (UDF) | Yes (Java) | Yes (Java) |
| Custom Serializer/Deserializer | Yes | Yes |
| DFS Direct Access | Yes (implicit) | Yes (explicit) |
| Join/Order/Sort | Yes | Yes |
| Shell | Yes | Yes |
| Streaming | Yes | Yes |
| Web Interface | Yes | No |
| JDBC/ODBC | Yes (limited) | No |

# Hive Architecture

edureka!

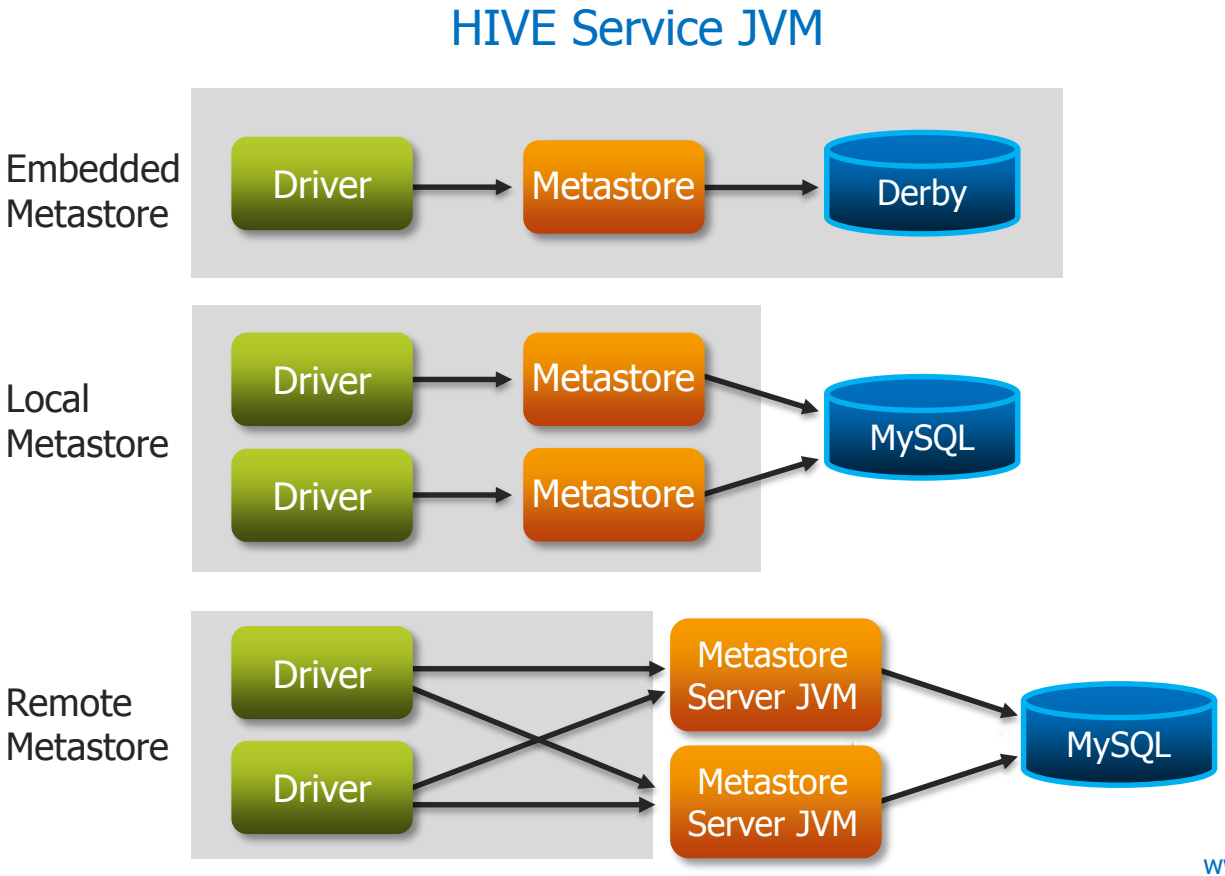


# Hive Components



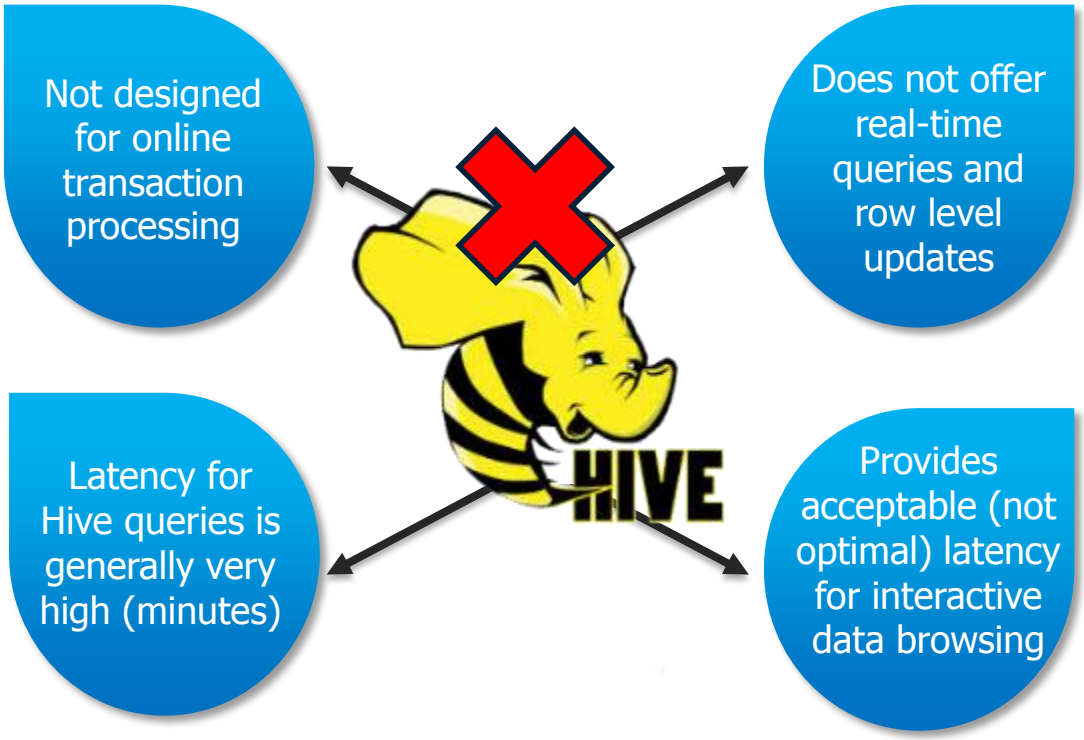
# Masterstore

edureka!



# Limitations of HIVE?

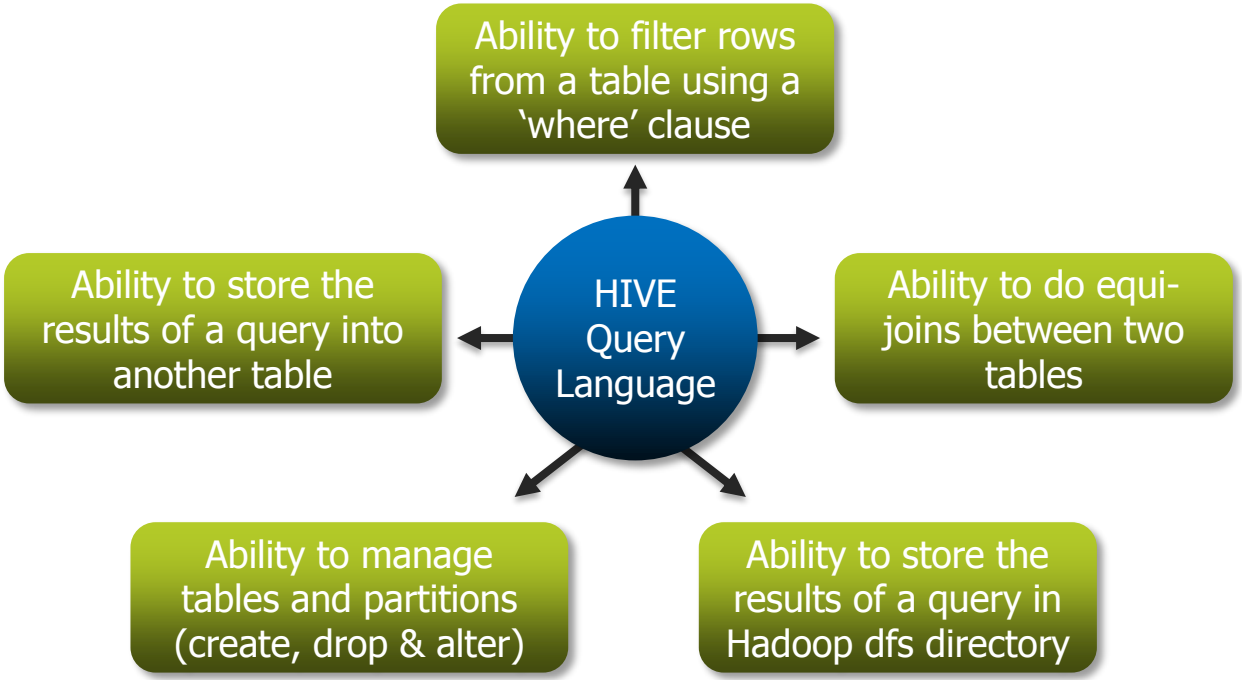
edureka!



# Abilities of HIVE Query Language

edureka!

Hive Query Language provides the basic SQL-like operations



# Differences With Traditional RDBMS

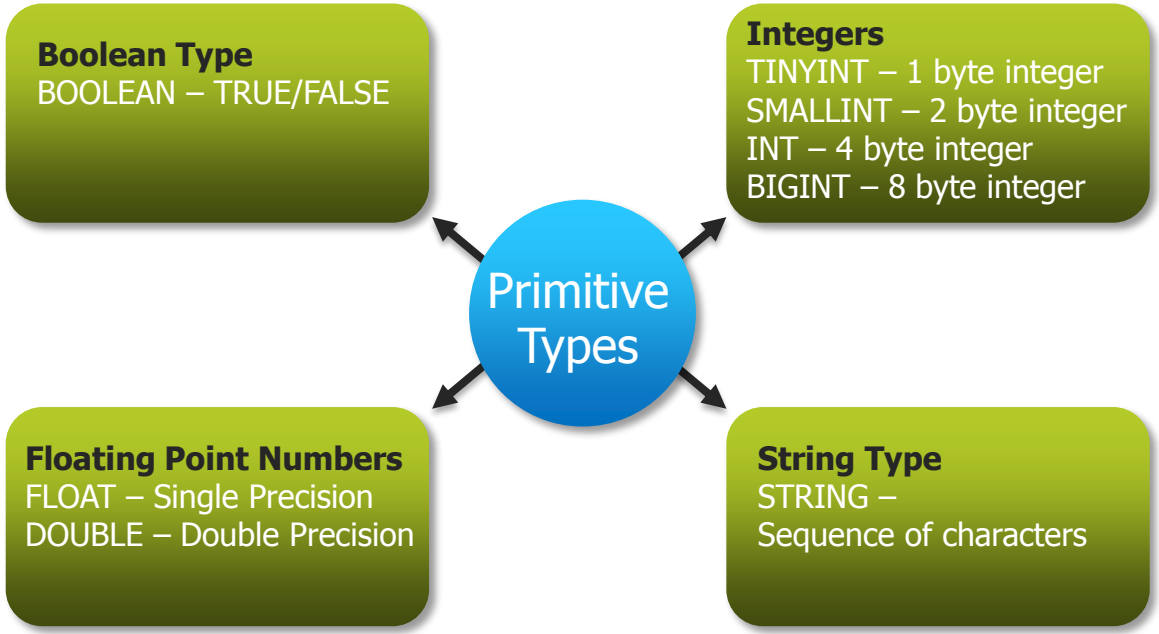
edureka!

- **Schema on Read vs Schema on Write**
  - **Hive does not verifies the data when it is loaded**, but rather when a query is issued.
  - Schema on read makes for a **very fast initial load**, since the data does not have to be read, parsed and serialized to disk in the database's internal format. The load operation is just a file copy or move.

- **No Updates, Transactions and Indexes.**

# Type System

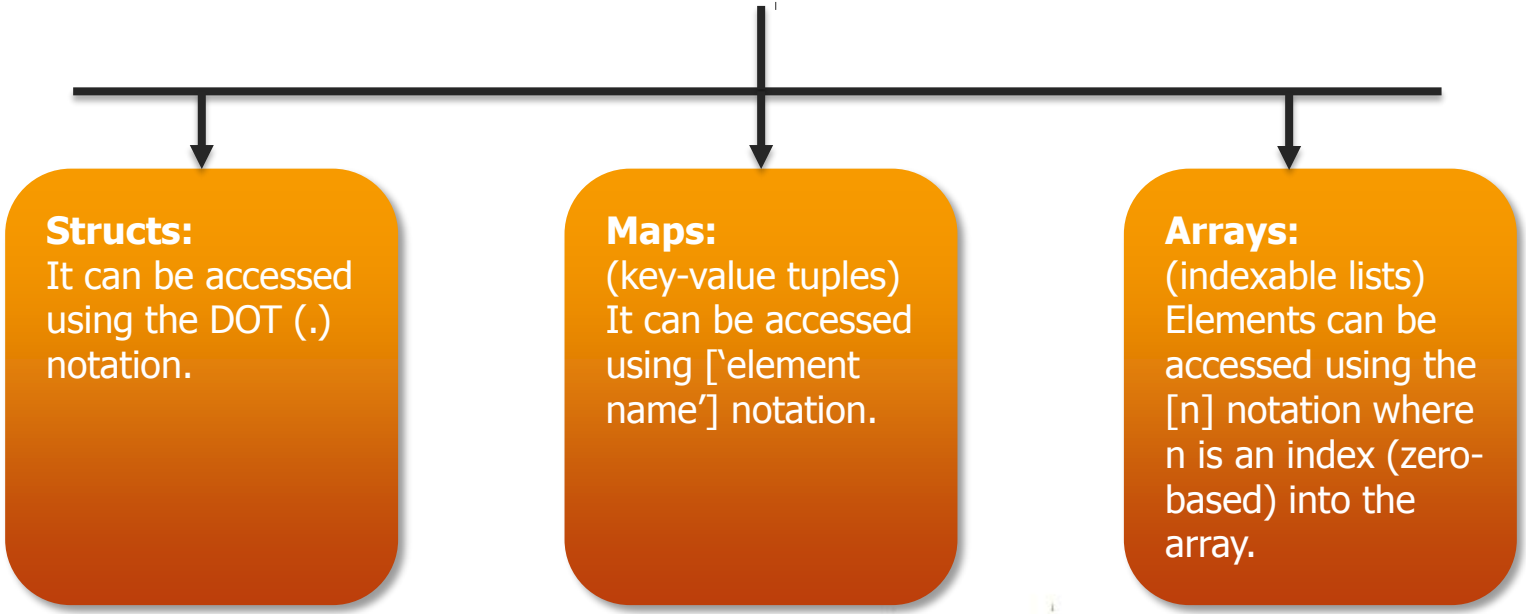



Boolean Type
BOOLEAN – TRUE/FALSE
Integers
TINYINT – 1 byte integer
SMALLINT – 2 byte integer
INT – 4 byte integer
BIGINT – 8 byte integer
Primitive Types
Floating Point Numbers
FLOAT – Single Precision
DOUBLE – Double Precision
String Type
STRING –
Sequence of characters

# Complex Types

edureka!

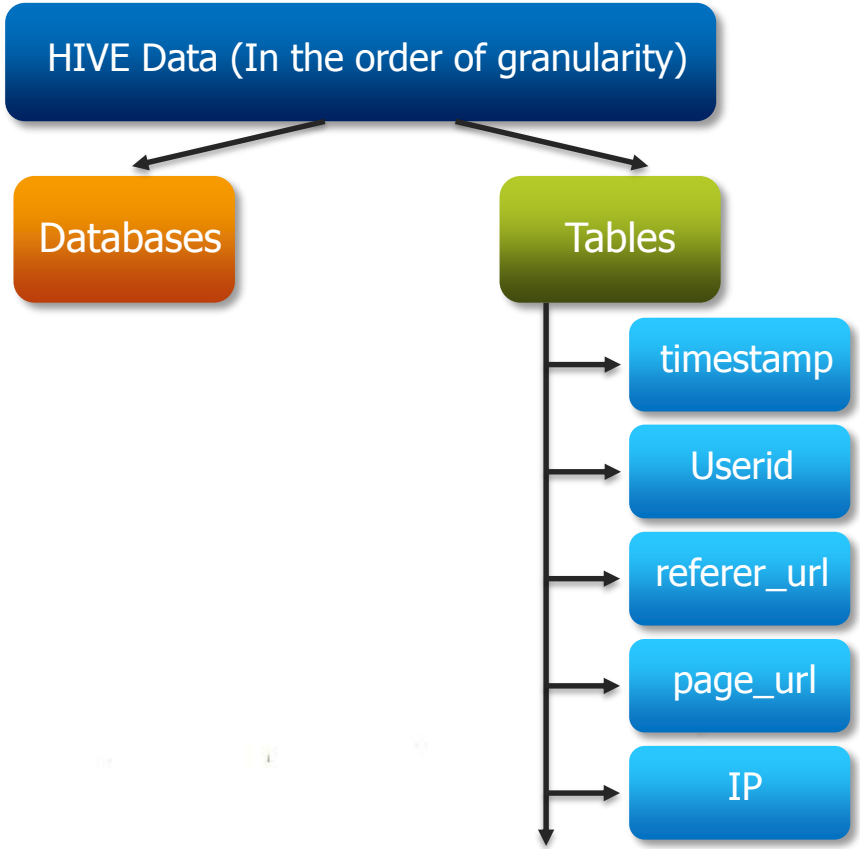
Complex Types can be built up from primitive types and other composite types using the following three operators:



# Hive Data Models

edureka!

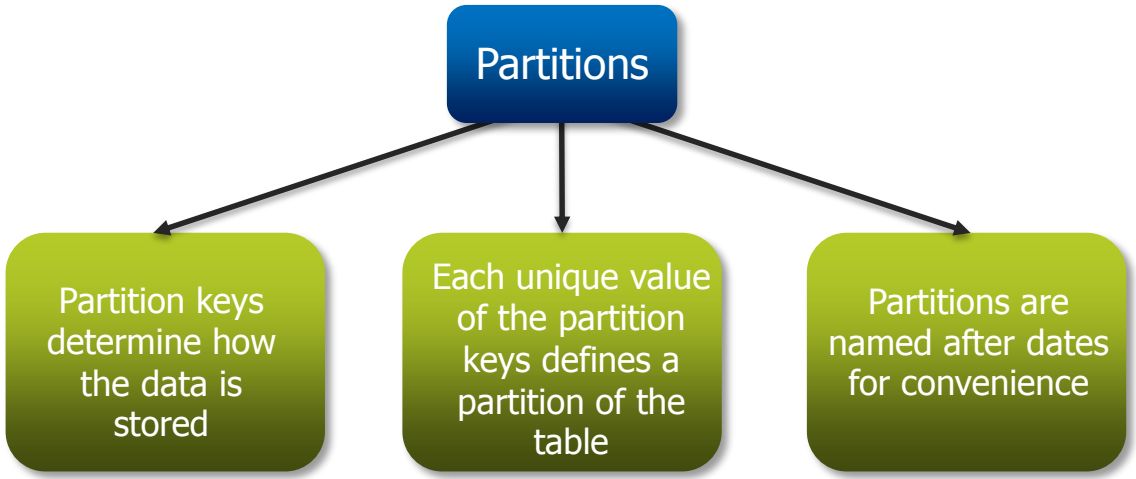
- **Databases**
  - Namespaces
- **Tables**
  - Schemas in namespaces
- **Partitions**
  - How data is stored in HDFS
  - Grouping data bases on some column
  - Can have one or more columns
- **Buckets or Clusters**
  - Partitions divided further into buckets bases on some other column
  - Use for data sampling



# Partitions

edureka!

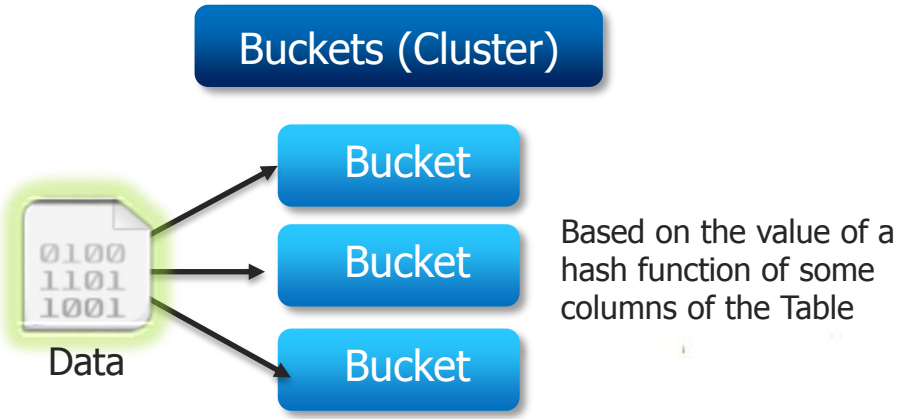
**Partition** means dividing a table into a coarse grained parts based on the value of a partition column such as a date. This make it faster to do queries on slices of the data.



# Buckets

edureka!

- ✓ Buckets give extra structure to the data that may be used for more efficient queries.
  - ✓ A join of two tables that are bucketed on the same columns – including the join column can be implemented as a Map Side Join.
  - ✓ Bucketing by user ID means we can quickly evaluate a user based query by running it on a randomized sample of the total set of users.



# Create Database And Table

edureka!

- **Create Database**
  - Create database retail;
- **Use Database**
  - Use retail;
- **Create table for storing transactional records**
  - Create table txnrecords(txnno INT, txndate STRING, custno INT, amount DOUBLE, category STRING, product STRING, city STRING, state String, Spendby String )
  - Row format delimited
  - Fields terminated by ',' stored as textfile

# External Tables

edureka!

- ✓ **Create the table in another hdfs location** and not in warehouse directory
- ✓ **Not managed by hive**
  - ✓ CREATE EXTERNAL TABLE external_Table (dummy STRING)
  - ✓ LOCATION '/user/notroot/external_table';


- ✓ **Hive does not delete the table (or hdfs files) even when the tables are dropped.**
  It leaves the table untouched and only metadata about the tables are deleted.

# Load Data

edureka!

- ✓ **Load the data into the table**
  - ✓ LOAD DATA LOCAL INPATH '/home/ubuntu/notroot/data/txn.csv'
  - ✓ OVERWRITE INTO TABLE txnrecords;

- ✓ **Describing metadata or schema of the table**
  - ✓ Describe txnrecords;

# Queries

edureka!

- **Select**
  - Select count(*) from txnrecords;
- **Aggregation**
  - Select count (DISTINCT category) from txnrecords;
- **Grouping**
  - Select category, sum( amount ) from txnrecords group by category

# Managing Outputs

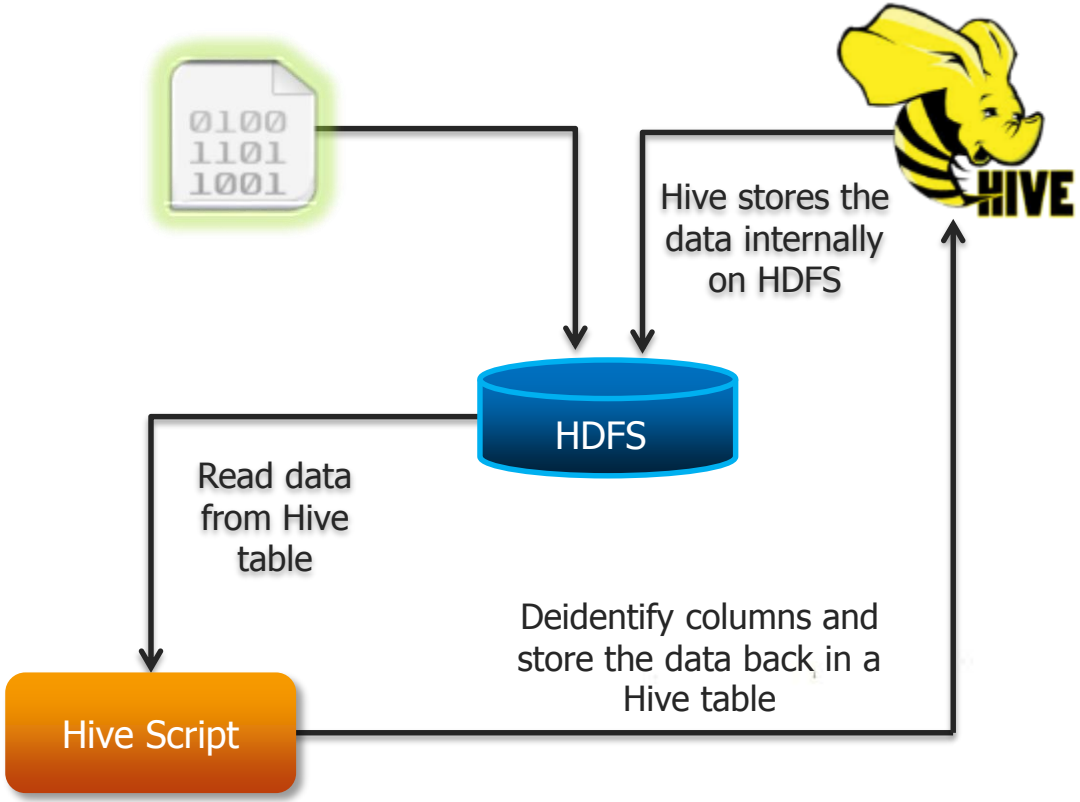
edureka!

- ✓ **Inserting Output into another table**
  - ✓ INSERT OVERWRITE TABLE results ( SELECT * from txnrecords)
- ✓ **Inserting into local file**
  - ✓ INSERT OVERWRITE LOCAL DIRECTORY'tmp/results' (SELECT * from txnrecords)

# Revisiting Use Case in Healthcare

edureka!

## Load CSV file into Hive



# Joining Two tables

edureka!

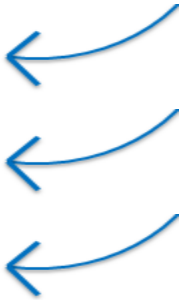
| User Table | | | |
|---|---|---|---|
| **Id** | **Email** | **Language** | **Location** |
| **1** | edureka@1.com | EN | US |
| **2** | edureka@2.com | EN | GB |
| **3** | edureka@3.com | FR | FR |

| Transaction Table | | | | |
|---|---|---|---|---|
| **Id** | **Product Id** | **UserId** | **Purchase Amount** | **Item Description** |
| **1** | Prod-1 | **1** | 300 | A jumper |
| **2** | Prod-1 | **2** | 300 | A jumper |
| **3** | Prod-1 | **2** | 300 | A jumper |
| **4** | Prod-2 | **3** | 100 | A rubber chicken |
| **5** | Prod-1 | **3** | 300 | A jumper |

# Joining Two tables

edureka!

| User Table | | | |
|---|---|---|---|
| **Id** | **Email** | **Language** | **Location** |
| **1** | edureka@1.com | EN | US |
| **2** | edureka@2.com | EN | GB |
| **3** | edureka@3.com | FR | FR |



Prod 1

| Transaction Table | | | | |
|---|---|---|---|---|
| **Id** | **Product Id** | **UserId** | **Purchase Amount** | **Item Description** |
| **1** | Prod-1 | **1** | 300 | A jumper |
| **2** | Prod-1 | **2** | 300 | A jumper |
| **3** | Prod-1 | **2** | 300 | A jumper |
| **4** | Prod-2 | **3** | 100 | A rubber chicken |
| **5** | Prod-1 | **3** | 300 | A jumper |

# Joining Two tables

edureka!

| User Table | | | |
|---|---|---|---|
| **Id** | **Email** | **Language** | **Location** |
| **1** | edureka@1.com | EN | US |
| **2** | edureka@2.com | EN | GB |
| **3** | edureka@3.com | FR | FR |

Prod 2

| Transaction Table | | | | |
|---|---|---|---|---|
| **Id** | **Product Id** | **UserId** | **Purchase Amount** | **Item Description** |
| **1** | Prod-1 | **1** | 300 | A jumper |
| **2** | Prod-1 | **2** | 300 | A jumper |
| **3** | Prod-1 | **2** | 300 | A jumper |
| **4** | Prod-2 | **3** | 100 | A rubber chicken |
| **5** | Prod-1 | **3** | 300 | A jumper |

# Joining Two tables

edureka!

**User Table**

| Id | Email | Language | Location |
|---|---|---|---|
| 1 | edureka@1.com | EN | US |
| 2 | edureka@2.com | EN | GB |
| 3 | edureka@3.com | FR | FR |

| Product | Location |
|---|---|
| Prod-1 | 3 |
| Prod-2 | 1 |

**Transaction Table**

| Id | Product Id | UserId | Purchase Amount | Item Description |
|---|---|---|---|---|
| 1 | Prod-1 | 1 | 300 | A jumper |
| 2 | Prod-1 | 2 | 300 | A jumper |
| 3 | Prod-1 | 2 | 300 | A jumper |
| 4 | Prod-2 | 3 | 100 | A rubber chicken |
| 5 | Prod-1 | 3 | 300 | A jumper |

# Assignment For Hive

edureka!

Attempt the following assignment using the document present in the LMS under the tab Week 5:

- ✓ Running Hive on Cloudera
- ✓ Execute Hive Queries on Txns Dataset
- ✓ Execute Health Care Use-Case
- ✓ Attempt Assignment Week 5

edureka!

# Thank You

See You in Class Next Week