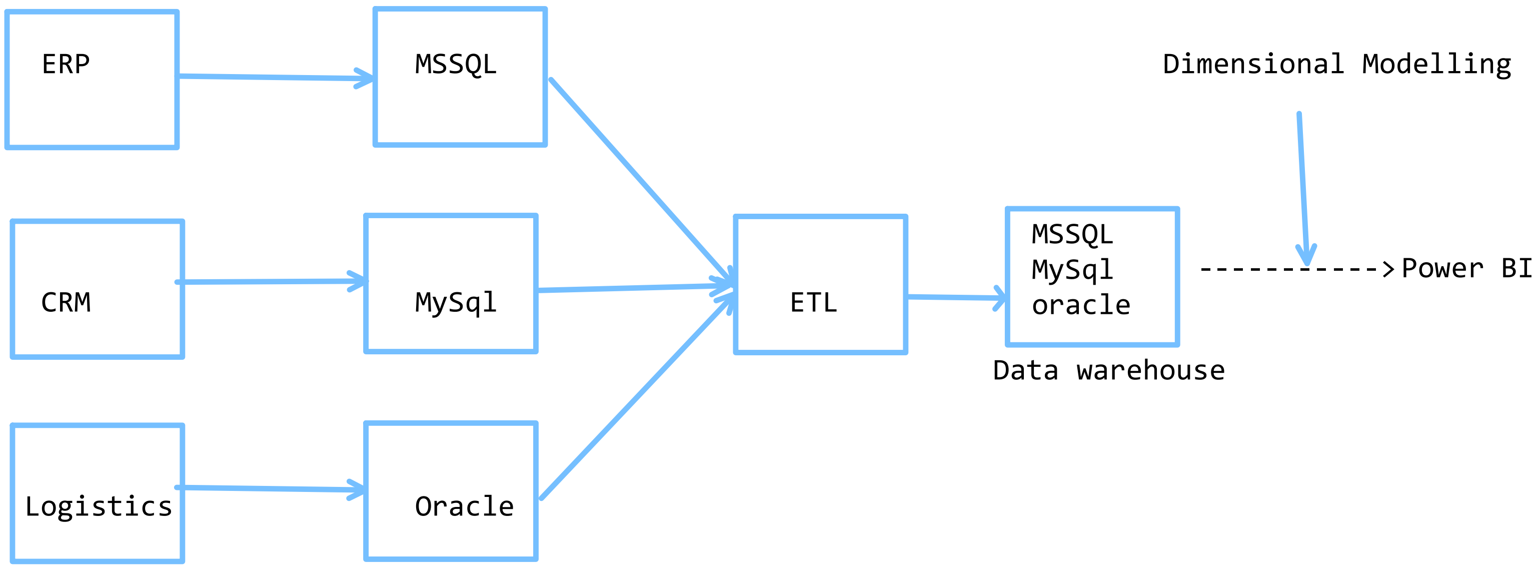
BI and Data Warehousing

19 November 2025 09:13



1. BI

Transforming raw data into useful information for business analysis.

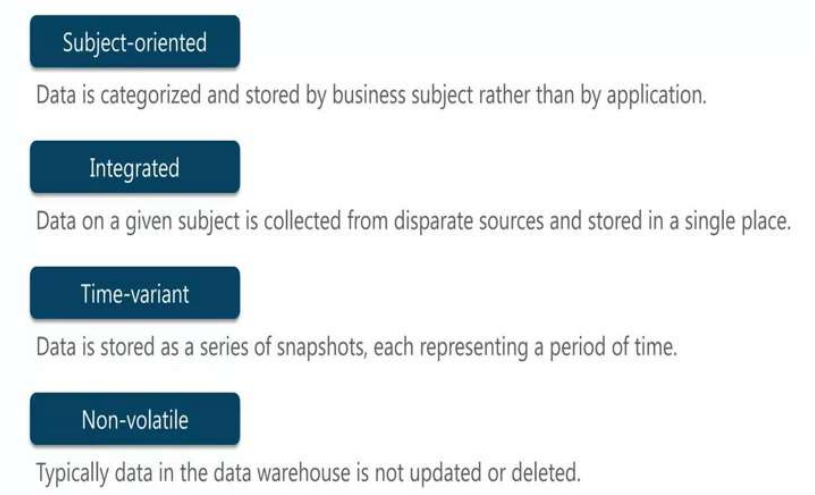
Data from company's databases is transformed and put into data warehouse.
Then analyzed using BI tools.

2. Data Warehouse

Data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data.

Data Warehouse Founder : Bill Inmon.

- Can be used for decision making.
- A central location where data from multiple sources is stored.
- End users access it as per their needs.
- Faster and accurate.
- Only once in a day/week ETL process happens on databases and then it is stored in data warehouse.
- Data in data warehouse in never deleted.
- We need time to create data warehouse.



Subject-oriented

Data is categorized and stored by business subject rather than by application.

Integrated

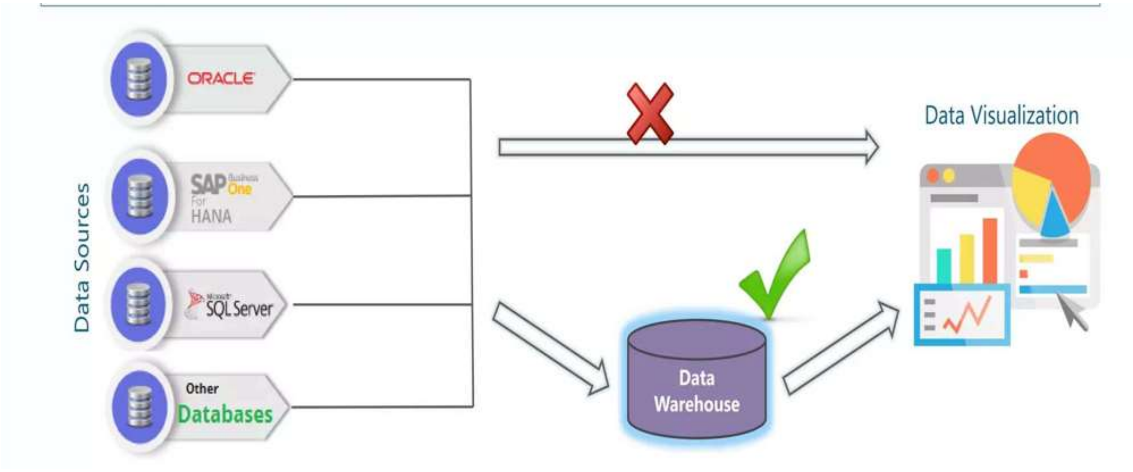
Data on a given subject is collected from disparate sources and stored in a single place.

Time-variant

Data is stored as a series of snapshots, each representing a period of time.

Non-volatile

Typically data in the data warehouse is not updated or deleted.



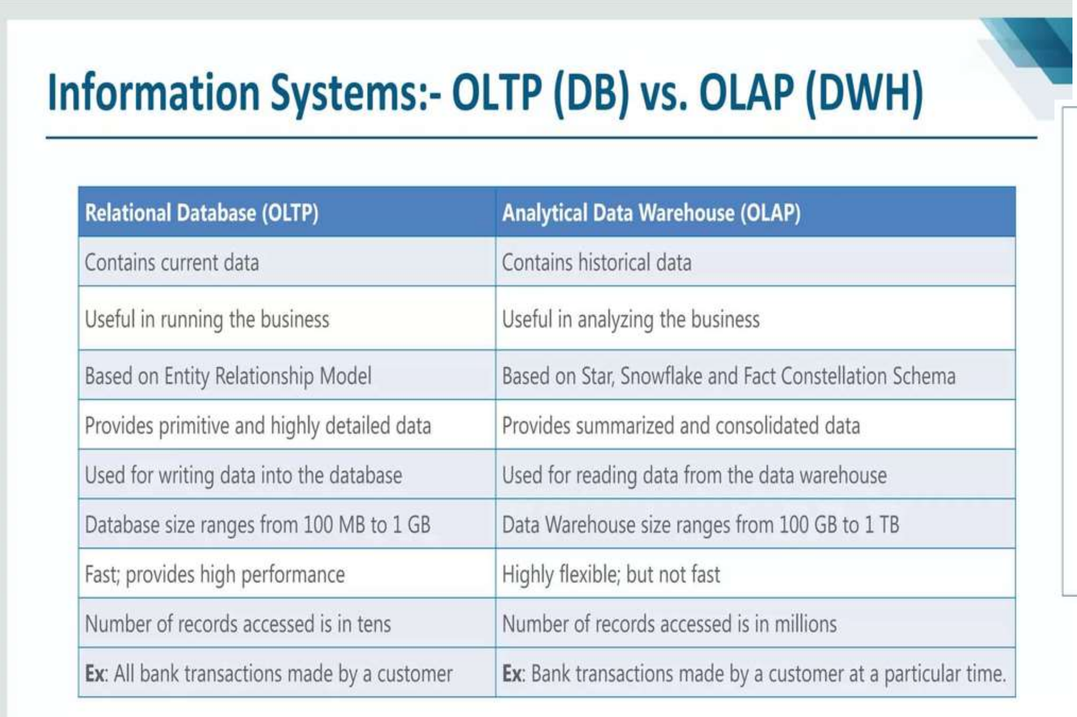
- Data from databases cannot be analyzed directly properly.

- So we have to transform it correctly and store into data warehouse and then analyze it.

Reason :

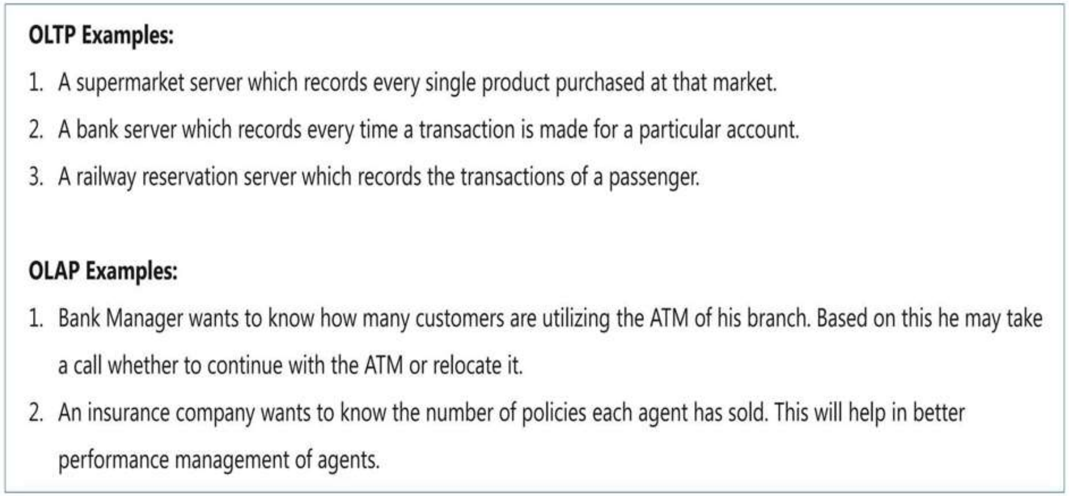
- Databases are already busy writing the data. Why increasing load on them.
- Data in databases is not clean/transformed

clean/transformed.



Information Systems:- OLTP (DB) vs. OLAP (DWH)

Relational Database (OLTP)	Analytical Data Warehouse (OLAP)
Contains current data	Contains historical data
Useful in running the business	Useful in analyzing the business
Based on Entity Relationship Model	Based on Star, Snowflake and Fact Constellation Schema
Provides primitive and highly detailed data	Provides summarized and consolidated data
Used for writing data into the database	Used for reading data from the data warehouse
Database size ranges from 100 MB to 1 GB	Data Warehouse size ranges from 100 GB to 1 TB
Fast; provides high performance	Highly flexible; but not fast
Number of records accessed is in tens	Number of records accessed is in millions
Ex: All bank transactions made by a customer	**Ex**: Bank transactions made by a customer at a particular time.



OLTP Examples:

1. A supermarket server which records every single product purchased at that market.
2. A bank server which records every time a transaction is made for a particular account.
3. A railway reservation server which records the transactions of a passenger.

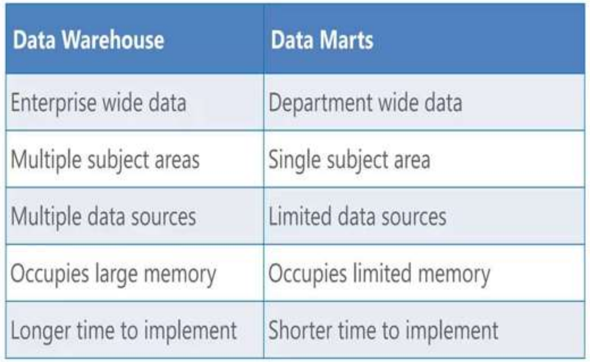
OLAP Examples:

1. Bank Manager wants to know how many customers are utilizing the ATM of his branch. Based on this he may take a call whether to continue with the ATM or relocate it.
2. An insurance company wants to know the number of policies each agent has sold. This will help in better performance management of agents.

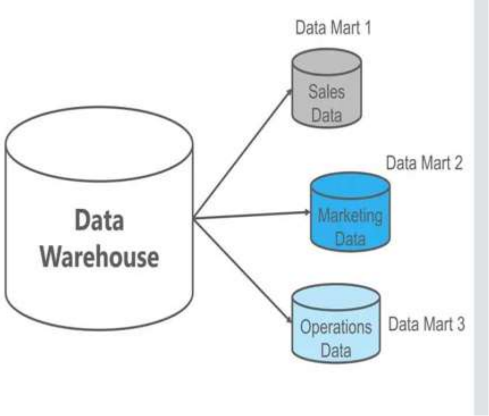
Data Mart

Data mart is smaller version of data warehouse which deals with only 1 subject.

- Subset of data warehouse.
- Can be created quickly.
- Focuses on one area so sources are also less.



Data Warehouse	Data Marts
Enterprise wide data	Department wide data
Multiple subject areas	Single subject area
Multiple data sources	Limited data sources
Occupies large memory	Occupies limited memory
Longer time to implement	Shorter time to implement



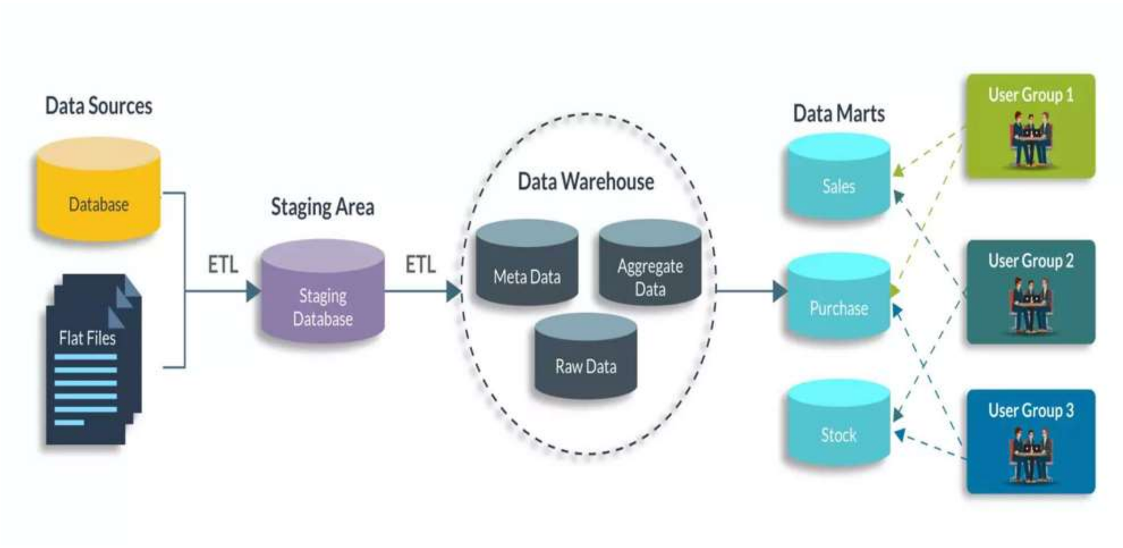
Numerical Data :

- Data in the form of numbers, but we must be able to perform meaningful operations on it.
- Operations like mean, median, mode etc.
- Ex. Sales, age, price, profit, loss.

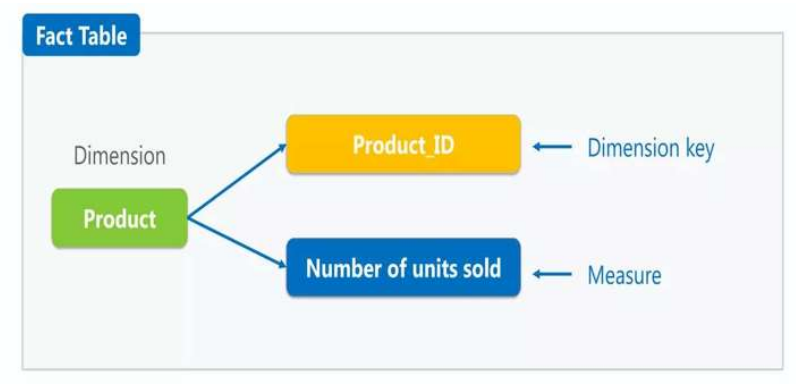
Categorical Data :

Data on which we can not perform operations.
Ex. Id, mobile no., name, address..

Data Warehouse Architecture



```
Dimensions :  (categorical columns)

  - Tables that describe dimensions are called dimension tables.
  - End user queries on these tables to get the data.



Facts / Measures  :  (Numerical columns)

  - Fact is a measure that can be summed, averaged, manipulated.
  - Tables that contain facts are called fact tables.
```

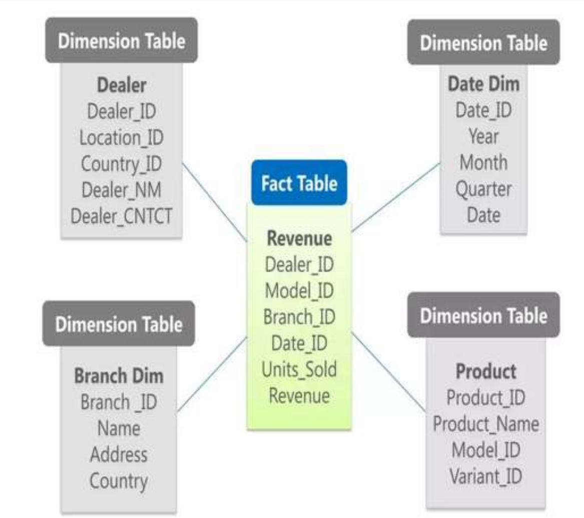


Schema : Logical structure of a dataset.

Types :

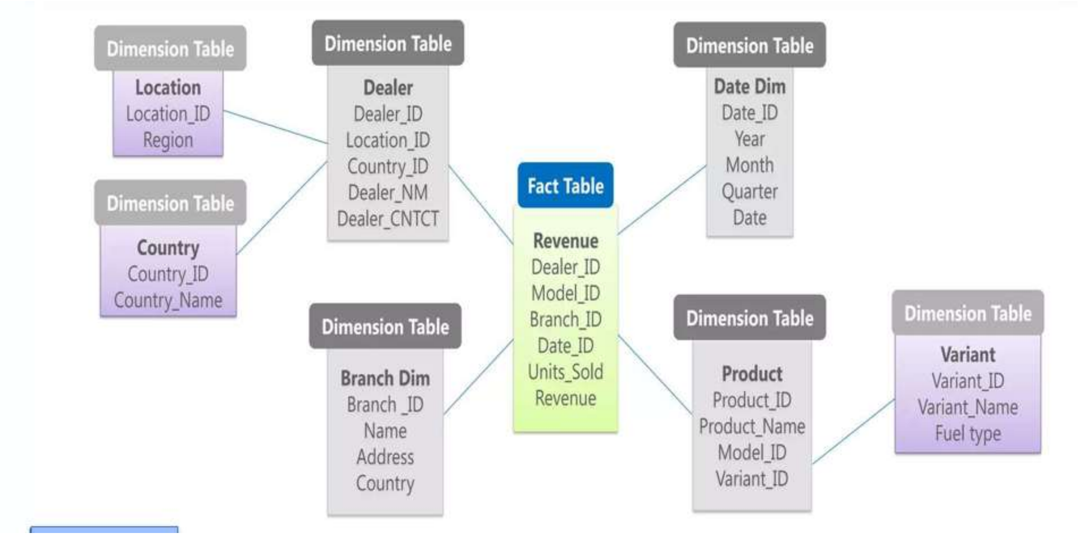
1. Star Schema

   - Each dimension in star schema is represented with one dimensional table which contains set of attributes.
   - Fact table is at the center, which contains keys to every dimension table and attributes like : units sold and revenue.



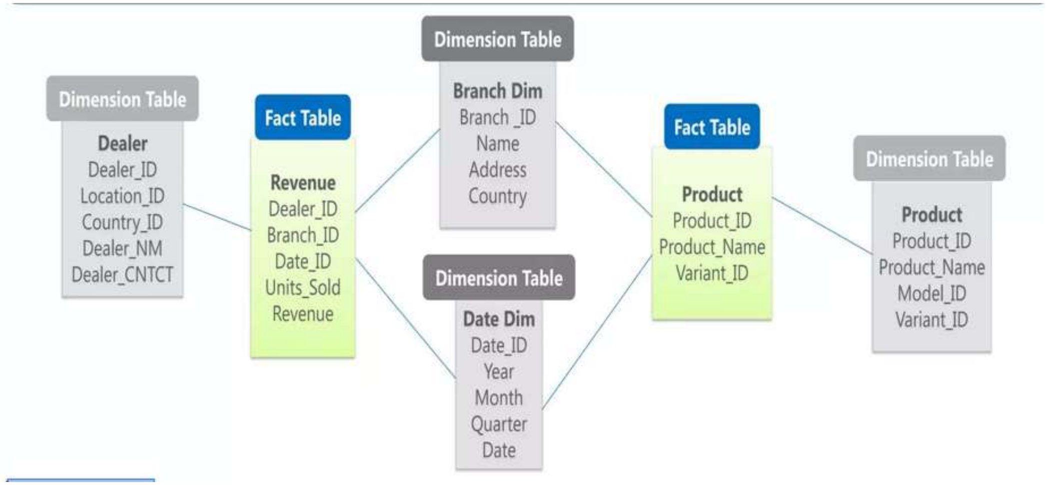
2. Snowflake Schema

- Dimensions in snowflake schema are normalized.
- Dimensions are split into further more tables.
- Extended version of snowflake schema.



3. Galaxy schema / Fact Constellation

- Contains more than 1 fact tables.
- Dimensions which are shared are called Conformed Dimensions.



Hive

- Hive is data warehouse in hadoop.
- It uses HiveQL (Hive query language). 99% similar to SQL.
- Hive queries are converted into MapReduce/Tez jobs.
- All the data in hive tables resides on HDFS typically.

One liner : Hive is a data warehousing tool built on top of Hadoop.

Tool - software.

Hive is like Athena in AWS.

There are 2 types of tables in Hive :

1. Managed Table / Internal Table (default)
2. External Table

Exercise with managed table:

1. Create an EMR cluster.
2. Connect to hadoop on MobaXterm.

3. Upload yellow.csv to data folder in Home directory.
   - Using hue gui
   - OR
   - Using mobaxterm :
   - Upload yellow.csv to master node locally.
   - hdfs dfs -mkdir /user/hadoop/data/
   - hdfs dfs -put yellow.csv /user/hadoop/data/

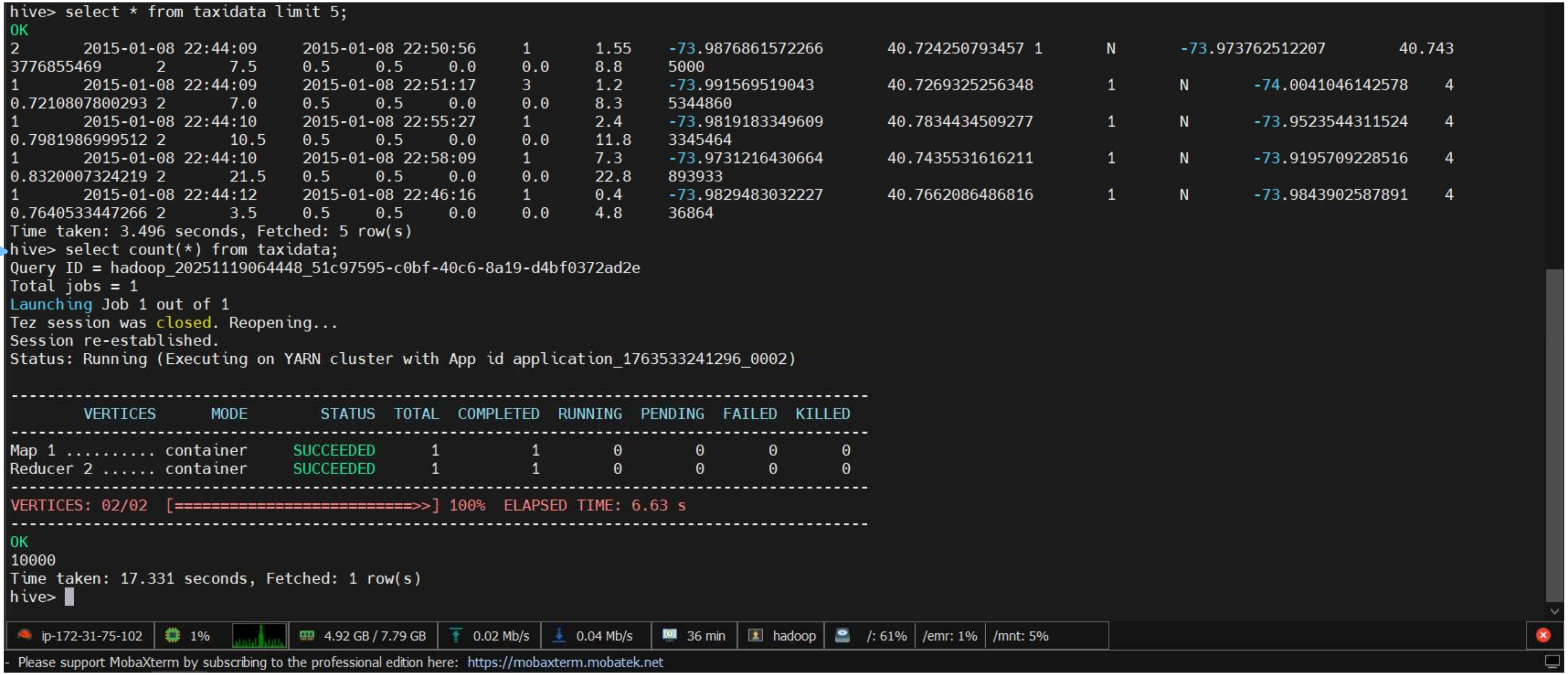
4. Type hive in mobaxterm(it loads hive interactive shell)

5. Run commands
   - !clear; --> to clear the console
   - show databases;
   - create database demo;
   - use demo;

6. Run commands from commands.txt, solutions_Taxi trip analysis.pdf

7. Original hive CLI does not display column names.

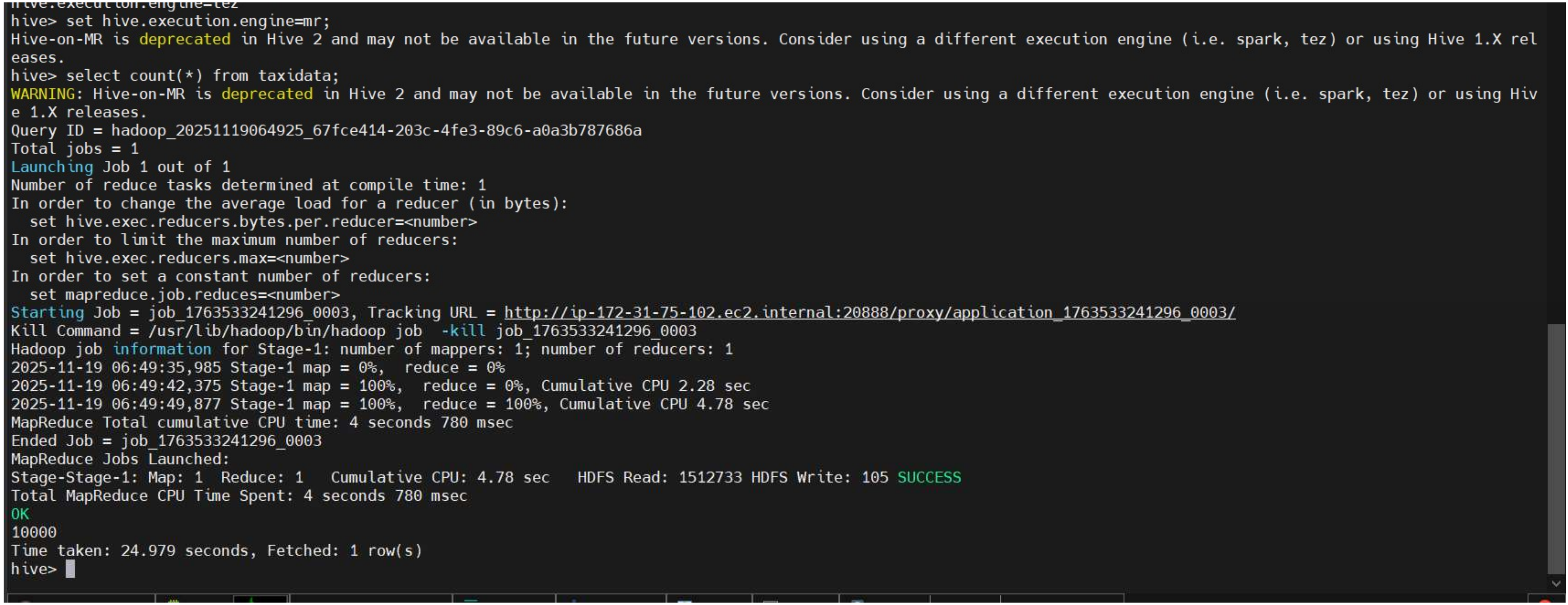
8. All hive queries are converted into Tez jobs.
   - But select * from taxidata limit 5;
   - Is not converted into Tez job,
   - But
   - Select count(*) from taxidata;
   - Will be converted into Tez job.



```
hive> select * from taxidata limit 5;
OK
2       2015-01-08 22:44:09     2015-01-08 22:50:56     1       1.55    -73.9876861572266       40.724250793457 1       N       -73.973762512207        40.743
3776855469      2       7.5     0.5     0.5     0.0     0.0     8.8     5000
1       2015-01-08 22:44:09     2015-01-08 22:51:17     3       1.2     -73.991569519043        40.7269325256348        1       N       -74.0041046142578       4
0.7210807800293 2       7.0     0.5     0.5     0.0     0.0     8.3     5344860
1       2015-01-08 22:44:10     2015-01-08 22:55:27     1       2.4     -73.9819183349609       40.7834434509277        1       N       -73.9523544311524       4
0.7981986999512 2       10.5    0.5     0.5     0.0     0.0     11.8    3345464
1       2015-01-08 22:44:10     2015-01-08 22:58:09     1       7.3     -73.9731216430664       40.7435531616211        1       N       -73.9195709228516       4
0.8320007324219 2       21.5    0.5     0.5     0.0     0.0     22.8    893933
1       2015-01-08 22:44:12     2015-01-08 22:46:16     1       0.4     -73.9829483032227       40.7662086486816        1       N       -73.9843902587891       4
0.7640533447266 2       3.5     0.5     0.5     0.0     0.0     4.8     36864
Time taken: 3.496 seconds, Fetched: 5 row(s)
hive> select count(*) from taxidata;
Query ID = hadoop_20251119064448_51c97595-c0bf-40c6-8a19-d4bf0372ad2e
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1763533241296_0002)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 6.63 s
----------------------------------------------------------------------------------------------
OK
10000
Time taken: 17.331 seconds, Fetched: 1 row(s)
hive> 
```

Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net

```
 9. To see execution engine use command :
        set hive.execution.engine;

10. Set execution engine to MR
        set hive.execution.engine=mr;

11. Query : select count(*) from taxidata;
  - This will take more time, cause MR us slow than Tez.
```



```
hive.execution.engine=tez
hive> set hive.execution.engine=mr;
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X rel
eases.
hive> select count(*) from taxidata;
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hiv
e 1.X releases.
Query ID = hadoop_20251119064925_67fce414-203c-4fe3-89c6-a0a3b787686a
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1763533241296_0003, Tracking URL = http://ip-172-31-75-102.ec2.internal:20888/proxy/application_1763533241296_0003/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1763533241296_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2025-11-19 06:49:35,985 Stage-1 map = 0%,  reduce = 0%
2025-11-19 06:49:42,375 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.28 sec
2025-11-19 06:49:49,877 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.78 sec
MapReduce Total cumulative CPU time: 4 seconds 780 msec
Ended Job = job_1763533241296_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.78 sec   HDFS Read: 1512733 HDFS Write: 105 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 780 msec
OK
10000
Time taken: 24.979 seconds, Fetched: 1 row(s)
hive>
```

Tt

12. Keep tez engine
    - set hive.execution.engine=tez;

13. Run queries from commands.txt
14. Open Resource Manager Application from AWS EMR page. See details of queries run.
15. If we look up to the yellow.csv that we uploaded to hadoop at /user/hadoop/data/yellow.csv

- It'll be moved to warehouse location whenever we load the data using "load" command.
- /user/hadoop/warehouse/
- In our case :
- /user/hive/warehouse/demo.db/taxidata

- This is done for easier access.

16. This is not beneficial because it may be needed to other apps in our pipeline.

17. Now we have to reupload this file to that location.

18. But next time if we don't want to do this way, we have to create the table in following way :

```
CREATE TABLE IF NOT EXISTS taxidatamylocation
(vendor_id string, pickup_datetime string,
dropoff_datetime string, passenger_count int, trip_distance
DOUBLE,
pickup_longitude DOUBLE, pickup_latitude DOUBLE, rate_code int,
store_and_fwd_flag string, dropoff_longitude DOUBLE,
dropoff_latitude
DOUBLE,
payment_type string, fare_amount DOUBLE, extra DOUBLE,
mta_tax DOUBLE,tip_amount DOUBLE,tolls_amount DOUBLE,
total_amount DOUBLE,trip_time_in_secs int)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED as TEXTFILE
LOCATION '/user/hadoop/data/'
TBLPROPERTIES ("skip.header.line.count"="1");
```

19. Here we are telling hive, create a table, my data is here, don't move my file into warehouse location.

20. In this way, we don't need to load the data into table. Because we're telling hive the location of data.

21. If there are more than 1 file, all will be accessed (files must be of same format).

22. When we do:

```
   describe formatted taxidata;
```

23. It'll display the details of table.

```
hive> describe formatted taxidata;
OK
# col_name              data_type
comment

vendor_id               string
pickup_datetime         string
dropoff_datetime        string
passenger_count         int
trip_distance           double
pickup_longitude        double
pickup_latitude         double
rate_code               int
store_and_fwd_flag      string
dropoff_longitude       double
dropoff_latitude        double
payment_type            string
fare_amount             double
extra                   double
mta_tax                 double
tip_amount              double
tolls_amount            double
total_amount            double
trip_time_in_secs       int

# Detailed Table Information
Database:               demo
Owner:                  hadoop
CreateTime:             Wed Nov 19 06:34:14 UTC 2025
LastAccessTime:         UNKNOWN
Retention:              0
Location:
hdfs://ip-172-31-75-102.ec2.internal:8020/user/hive/w
arehouse/demo.db/taxidata
Table Type:             MANAGED_TABLE
Table Parameters:
        numFiles                1
        numRows                 0
        rawDataSize             0
        skip.header.line.count  1
        totalSize               1512490
        transient_lastDdlTime   1763534330

# Storage Information
SerDe Library:
org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:
org.apache.hadoop.mapred.TextInputFormat
OutputFormat:
org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputF
ormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        serialization.format    ,
Time taken: 0.092 seconds, Fetched: 49 row(s)
hive>
```

Exercise with external table :

1. Create table

```
CREATE EXTERNAL TABLE IF NOT EXISTS taxidata_exc
(vendor_id string, pickup_datetime string,
dropoff_datetime string, passenger_count int, trip_distance
DOUBLE,
pickup_longitude DOUBLE, pickup_latitude DOUBLE, rate_code
int,
store_and_fwd_flag string, dropoff_longitude DOUBLE,
dropoff_latitude
DOUBLE,
payment_type string, fare_amount DOUBLE, extra DOUBLE,
mta_tax DOUBLE,tip_amount DOUBLE,tolls_amount DOUBLE,
total_amount DOUBLE,trip_time_in_secs int)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED as
TEXTFILE
LOCATION '/user/hadoop/external';
```

- Even if our /external/ is not present, Hive will create it.

2. Upload the data (yellow.csv) to Hue manually.

Finally :

1. drop table taxidata;

Yellow.csv will be gone from
/user/hive/warehouse/demo.db

2. drop table taxidata_exc

Yellow.csv will still be on the location
/user/hadoop/external/yellow.csv

Interview Question :

1. When to ues managed and when to use external table

Ans :

a. When I'm the only one using the data, I'll use managed table.
   - Or the data is inside the hadoop cluster.

b. When others are also working with me on that data, I'll use external table.
   - Or the data is outside hadoop.
   - And when we want the data to not be deleted

when we drop the table.

Here, we've used location for external location inside hadoop cluster, but we can mention another database also, like aws s3, cassandra, mongodb, Mysql, etc .

We're connecting to hive using original hive client.

Hive things:

1. Hive is Not a database.

2. Hive queries are slow because they are converted into Tex/MR.

3. Hive does not provide real time queries as well as row level updates. It is not suitable for Online Transaction Processing (OLTP) systems.

File Formats :

CSV --> structured, slow to read
JSON -> semi-structured, lightweight, machine readable, stores data in object
XML --> semi-structured

Big data file formats :

1. Parquet --> Most common, most popular, compression, created by cloudera.

2. ORC -- > compression, open source Apache

Parquet and orc store data in columnar format, means each column in stored in separate file, which makes operations on that column faster.

Before ORC we had only RC
RC - means Row Columnar format
ORC - means Optimized Row Columnar format

Parquet vs ORC --> ORC is better

3. Avro ----> Serialization file format.
   - Serialization is effective when we're sending data over network.
   - it converts our data in a format that is easy to transfer over network

- SerDe is a library inside Hive.

  SerDe --> Serializer Deserializer

  - They help Hive to read and understand different file formats.

  - Hive has Regex SerDe, it has a regex that we can use to apply regex on input data to convert it into structured format.

  - It has JSON SerDe, XML SerDe, CSV SerDe etc

  - All of them convert unstructured data into structured format (tablular)

HTTP Response codes :

a. 1xx - informational response
b. 2xx - success
c. 3xx - redirection
d. 4xx - client error
e. 5xx - server error

Exercise Nasa logs casestudy:

1. Create EMR
2. Open Hue application
3. Create a folder data
4. Upload access file inside that.
5. Connect hadoop to Mobaxterm
6. Type hive in MobaXterm
7. Create database demo;
8. Use database demo;
9. Run Command from edited nasa web log case study.pdf

```
CREATE TABLE IF NOT EXISTS nasa_log (host String, identity String,
userIdentity String, time String, request String, status String,
size String) ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.RegexSerDe' WITH SERDEPROPERTIES
("input.regex" = "([^ ]*) ([^ ]*) ([^ ]*) (- |\\[[^\\]]*\\]) ([^
\"]*|\"[^\"]*\") (-|[0-9]*) (-|[0-9]*)" , "output.format.string" =
"%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s") STORED AS TEXTFILE;
```

10. Load data into table

```
    LOAD DATA INPATH '/user/hadoop/logs/access' INTO TABLE nasa_log;
```

11. Run -> set hive.execution.engine=mr;

12. Run next commands from that pdf.

13. Find the top endpoints that received server side error

```
 SELECT status, count(request) FROM nasa_log GROUP BY status HAVING
status == regexp_extract(status, '^50.', 0) ORDER BY status DESC
LIMIT 5;
```

- In this query, map reduce makes 2 jobs instead of 1,
- because map reduce creates 1 job for 1 aggregation,
- here we're doing 2 aggregations, group by and order by

14. Open Name Node application from AWS EMR page

15. From horizontal menu, click on utilities -> browse file system

16. From Name column, click on user -> hive -> warehouse -> demo.db -> nasa_log -> access

17. Click on access, beside block information, click on Block 0, we can see a drop down Block 0 and Block 1, because the block size 128MB restriction.

18. Run 4th query from that pdf.

   - In both the queries' output we can see no. of mappers and reducers etc details.

Exit the hive using Ctrl + C   or   type exit/quit

Beeline client connector :

- Command to connect -> beeline -u "jdbc:hive2://localhost:10000/default"

- It's a jdbc connection, hive2 means hive server 2, default is a database.

- We can run same queries on this, the output comes structured. Just like SQL

- To exit, type !q   or Ctrl + C