Unit – V - HIVE

Data Warehouse – Hadoop - Data Query And Analysis

Hive

- Hive is a **Data Warehouse** built on top of Apache Hadoop for providing **Data Query And Analysis**.
- Hive gives an SQL-like interface to query data stored in various databases and file systems that integrate with Hadoop.
- Hive using SQL, and it provides **HiveQL** (Hive Query Language).
- Hive is widely used in various industries, including
    1. finance
    2. healthcare
    3. Retail
    4. telecommunications,
- It is a popular choice for data warehousing because it is
    1. scalable,
    2. fault-tolerant,
    3. easy to use

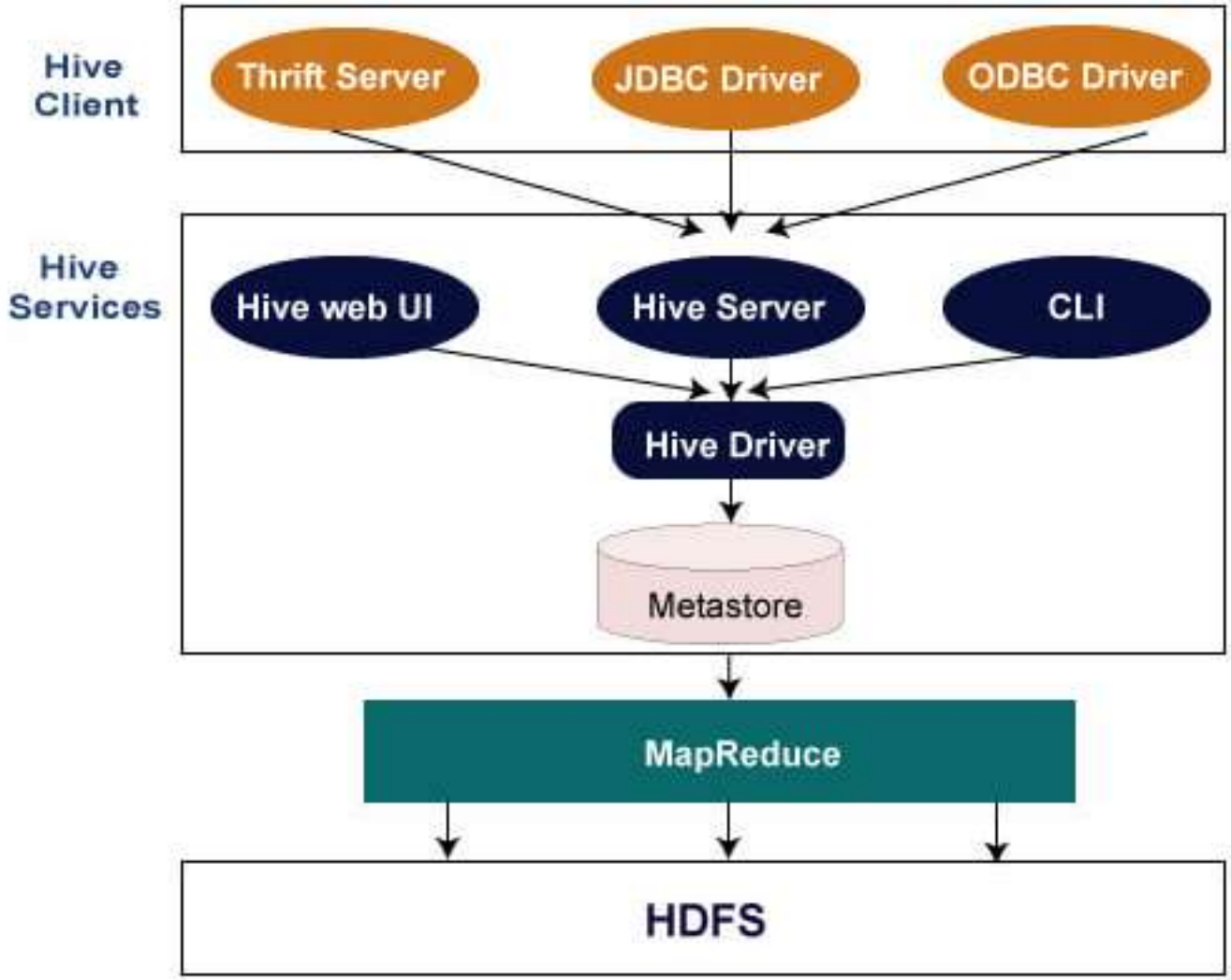
Here is a comparison of Hive and traditional databases:

Feature	Hive	Traditional databases
Schema	Schema-on-read	Schema-on-write
Data types	Supports a variety of data types, including structured, semi-structured, and unstructured data	Supports structured data only
Scalability	Highly scalable to handle large datasets	Scalability can be limited for large datasets
Performance	Optimized for batch processing	Optimized for online transaction processing (OLTP)
Query language	HiveQL, which is similar to SQL	SQL
Use cases	Ideal for large-scale data warehousing and analytics	Ideal for OLTP and other applications that require real-time data access

✓OLTP stands for Online Transaction Processing.

✓It's a software program or operating system that supports transaction-oriented applications. .

Hive Architecture



How The Hive Architecture Works:-

- The **Hive Client** submits a **HiveQL** (Hive Query Language) query to the **Hive Server**.

- **Hive Server** - **Translates** the HiveQL query into a MapReduce job.
- **Hive Server** - **Submits** the MapReduce job to the Hadoop cluster.
- **Hadoop Cluster** - **Executes** the MapReduce job.
- **Hive Server** - **Collects** the results of the MapReduce job and returns them to the Hive client.
- **Hive Client** - **Displays** the results of the query to the user.

Advantages

- **Use Cases**: Hive is ideal for data warehousing and analytics, where large datasets are analyzed to extract insights and patterns.
- **Schema Flexibility**: Hive provides **Schema-on-read**, allowing users to define the structure of data during query execution.
- **Cost-Effectiveness**: Hive, built on top of the open-source Hadoop ecosystem, offers a cost-effective solution for large-scale data analysis.
- **Scalability**: Hive is designed for massive datasets, leveraging the distributed processing power of Hadoop to handle Petabytes of data efficiently.
- **Data Structure**: Hive is optimized for handling structured and semi-structured data.
- **Query Language**: HiveQL, Hive's query language, is similar to SQL but tailored for handling large datasets in a distributed environment.
- **Processing Speed**: Hive excels in batch processing, efficiently handling large-scale data analysis tasks.

Dis-Advantages

- **Real-time Processing**: Hive is not ideal for real-time processing due to its batch-oriented nature.
- Traditional databases are better suited for applications requiring immediate response to queries.

- Overall, Hive is a powerful tool for data warehousing and analytics.
- It is well-suited for organizations that need to process and analyze large datasets.
- However, it is important to note that Hive is not a replacement for traditional databases for all workloads.
- The choice between Hive and traditional databases depends on the specific data requirements and processing needs of the application.

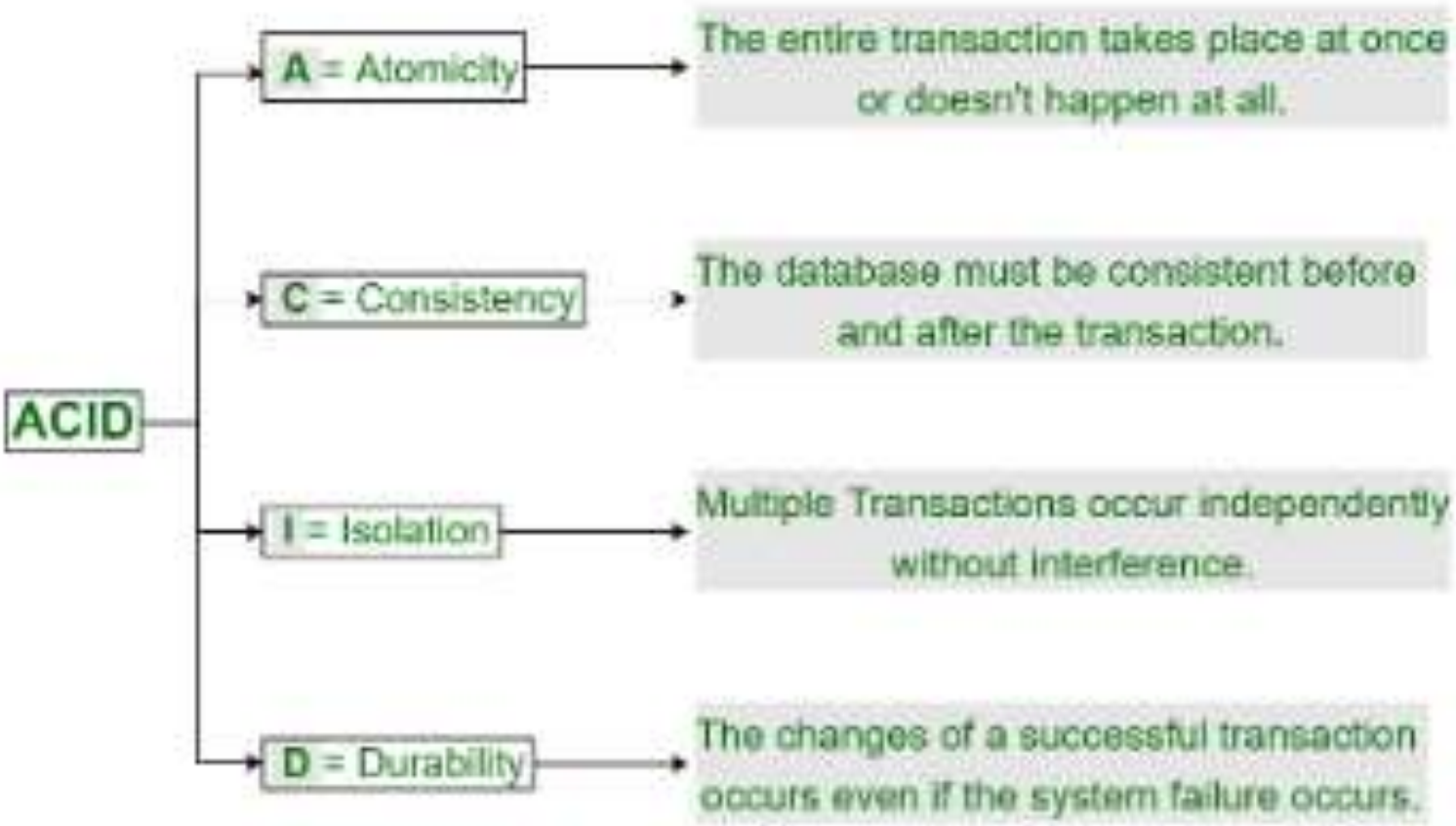
When to use Hive:

- ✓ When you need to process and analyze large datasets.
- ✓ When you need a schema-flexible data warehouse.
- ✓ When you need a cost-effective big data analytics solution.

When to use a traditional database:

- ✓ When you need real-time data access.
- ✓ When you need ACID transactions (Atomicity, Consistency, Isolation, Durability.)
- ✓ When you need a database with a well-defined schema.

ACID Properties in DBMS



Hive QL – Hive Query Language

- HiveQL (Hive Query Language) is a query language similar to SQL that is used to query data stored in Hive.
- It is a **Declarative Language**, meaning that the user tells Hive what they want, but not how to do it.
- HiveQL queries are executed using **MapReduce jobs**.
- Hive **Translates** HiveQL queries into MapReduce jobs, which are then **Executed** on the Hadoop cluster.
- This makes HiveQL a very powerful tool for querying large datasets.

HiveQL can be used to perform a wide variety of tasks, such as:

- Querying data from various databases and file systems.
- Performing data analysis and aggregation.
- Creating and managing tables and databases.
- Loading and exporting data.

HiveQL & SQL

HiveQL supports a variety of SQL statements, including:

- SELECT: Retrieves data from a table.
- INSERT: Inserts data into a table.
- UPDATE: Updates data in a table.
- DELETE: Deletes data from a table.
- CREATE TABLE: Creates a new table.
- DROP TABLE: Drops a table.
- ALTER TABLE: Modifies the structure of a table.

HiveQL also supports a variety of built-in functions, such as mathematical functions, string functions, and date functions

Querying Data

Here are some tips for querying data in Hive:

SHOW DATABASES And **SHOW TABLES** - To view the list of databases and tables in Hive.

DESCRIBE TABLE - To view the schema of a table.

WHERE clause - filter the results of a query.

ORDER BY clause - To sort the results of a query.

LIMIT clause - To limit the number of rows returned by a query.

- ✓ You can also use HiveQL to create and manage views.
- ✓ Views are virtual tables that are based on other tables.
- ✓ Views can be used to simplify complex queries or to provide different views of the same data.

SELECT * FROM table_name;

- This query will select all of the rows from the table table_name.

SELECT id, name, age

FROM table_name

WHERE age > 18;

- This query will select all of the rows from the table table_name where the age column is greater than 18.

You can also save HiveQL queries to files and then execute them using the `hive` command. `query.hql` :

Code snippet

```
SELECT * FROM table_name;
```

Use code with caution. Learn more

You would use the following command:

```
echo "SELECT * FROM table_name;" > query.hql
```

Then, to execute the query, you would use the following command:

```
hive -f query.hql
```

This will execute the query and print the results to the console.

Tables

- Hive tables are used to store and organize data in Hive.
- They are similar to tables in traditional relational databases, but they can also **Store Semi-structured And Unstructured Data**.

Types of Hive tables:

Hive tables can be classified into two types:

Managed Tables: Managed tables are **Owned By Hive**.

- Hive is responsible for creating and deleting the underlying data files.

External Tables: External tables are **Not Owned By Hive**.

- Hive simply provides a way to query the data files that are already present in the Hadoop Distributed File System (HDFS).

Benefits of using Hive tables

Scalability: Hive tables can be scaled to handle large datasets.

Flexibility: Hive tables can store a variety of data types, including structured, semi-structured, and unstructured data.

Performance: Hive tables are optimized for batch processing, which is ideal for many data warehousing and analytics workloads.

Cost-effectiveness: Hive is open source and free to use.

Use cases for Hive tables

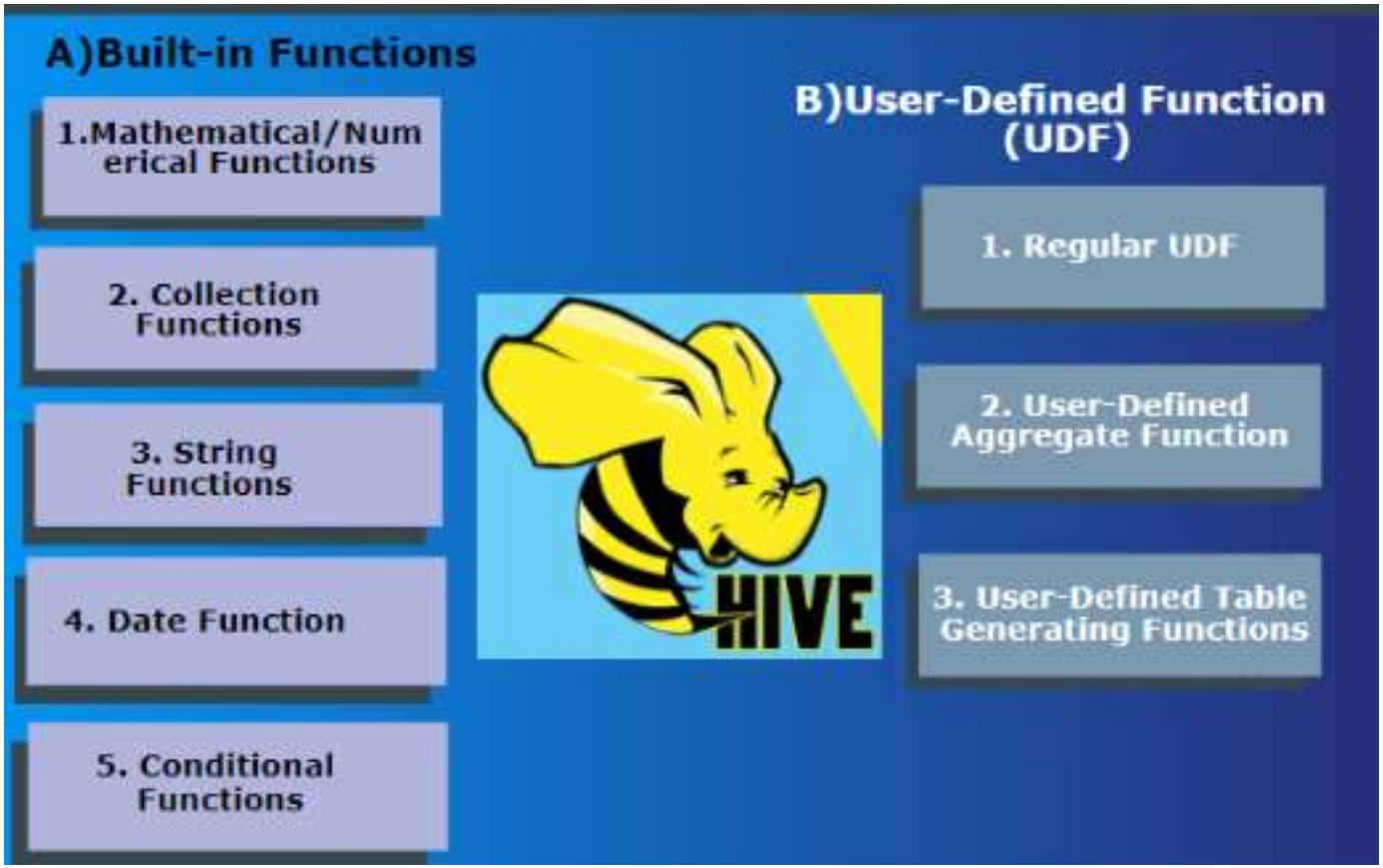
Hive tables are commonly used for a variety of data warehousing and analytics workloads, such as:

- **Data Warehousing**: Hive tables can be used to **Store And Analyze Large Datasets** to gain insights into business trends and patterns.
- **Data Mining**: Hive tables can be used to mine large datasets **For Hidden Patterns And Correlations**.
- **Machine Learning**: Hive tables can be used to **Train And Deploy Machine Learning Models**.

Conclusion:

- Hive tables are a powerful and versatile tool for storing and analyzing data. They are well-suited for organizations that need to process and analyze large datasets.

Type Of Hive Function



User Defined Functions

- Hive user-defined functions (UDFs) are custom functions that can be written in **Java and integrated with Hive**.
- UDFs can be used to extend the functionality of Hive and to perform **Complex Computations** that would not be possible with built-in Hive functions.
- To **Create A Hive UDF**, you need to write a **Java class** that implements the org.apache.hadoop.hive.ql.exec.UDF interface.
- The UDF class must also define a method that performs the desired computation.
- Once you have written the UDF class, you need to **Compile It Into A JAR File**.
- You can then **Register** the JAR file with Hive using the **CREATE FUNCTION Statement**.

Hive UDFs Can Be Used To Perform A Variety Of Tasks, Such As:-

Data Transformation: UDFs can be used to transform data from one format to another. For example, a UDF could be used to convert a date string from one format to another, or to convert a string to a number.

Data Aggregation: UDFs can be used to perform aggregate operations on data. For example, a UDF could be used to calculate the average value of a column, or to count the number of rows in a table.

Data Filtering: UDFs can be used to filter data. For example, a UDF could be used to filter out rows where a column value is equal to a certain value, or to filter out rows where a column value is missing.

Hive UDFs can be a powerful tool for extending the functionality of Hive and for performing complex computations on large datasets.

Here are some additional examples of Hive UDFs:

- A UDF that **Calculates The Distance Between Two Points** on a map.
- A UDF that **Extracts The Sentiment Of A Piece Of Text**.
- A UDF that **Recommends Products To Customers Based On Their Past Purchase History.**

Hive UDFs can be written to perform a wide variety of tasks, and they can be a valuable tool for data scientists and analysts who need to work with large datasets.