UNIT V – FRAMEWORKS

QUESTION BANK

PART A (2 Marks each)

1. What is Pig? **Imp**
2. What are the advantages of Pig over MapReduce?
3. What are the disadvantages of Pig over MapReduce?
4. List out the applications on Big data using Pig.
5. What is Hive?
6. How querying data in Hive?

PART B (5 Marks each)

7. Briefly explain data processing operators in Pig.
8. Explain Hive services in detail. **Imp**
9. Explain Hive QL. **Imp**
10. Explain HBase. Write about its characteristics.
11. Explain Zookeeper. Write about its characteristics.

PART C (15 Marks each)

12. Explain Different data processing operators in Pig. **Imp**
13. Explain the fundamentals of **Imp**
    a) HBase
    b) Zookeeper

NOTES

WHAT IS PIG?

- It is a high-level platform or tool which is used to process large datasets.
- It provides a high level of abstraction for processing over MapReduce.
  - MapReduce working out how to fit data processing into a pattern, which often requires multiple MapReduce stages, can be a challenge.
- The data structures are much richer in Pig.
- Typically, being multivalued and nested.
- Set of transformations can apply to the data are much more powerful—they include joins etc.
- Pig is made up of two pieces:
  • It provides a high-level scripting language, known as ***Pig Latin*** which is used to develop the data analysis codes.
  • The execution environment to run Pig Latin programs.
- There are currently two environments:
  - Local execution in a single JVM
  - Distributed execution on a Hadoop cluster.

ADVANTAGES OF PIG OVER MAPREDUCE

- The development cycle of MapReduce is very long.
- Writing the mappers and reducers, compiling, and packaging the code, submitting the job(s), and retrieving the results is time-consuming in MapReduce.
- Pig can process terabytes of data simply by issuing a half-dozen lines of Pig Latin.
- Pig was created at Yahoo! to make it easier for researchers and engineers to mine the huge datasets.
- Pig is very supportive for a programmer writing a query, since it provides several commands for introspecting (ആത്മപരിശോധന) the data structures in the program.
- Pig was designed to be extensible.
- Virtually all parts of the processing path are customizable: loading, storing, filtering, grouping, and joining can all be altered by **user-defined functions (UDFs**).
- These functions operate on Pig's nested data model, so they can integrate very deeply with Pig's operators.
- As another benefit, UDFs tend to be more reusable than the libraries developed for writing MapReduce programs.
- Writing queries in Pig Latin will save time.

DISADVANTAGES OF PIG OVER MAPREDUCE

- Pig is not suitable for all data processing tasks.
- Like MapReduce, it is designed for batch processing of data.
- If you want to perform a query that touches only a small amount of data in a large dataset, then Pig will not perform well, since it is set up to scan the whole dataset, or at least large portions of it.
- In some cases, Pig does not perform as well as programs written in MapReduce.
- However, the Pig team implements sophisticated algorithms for implementing Pig's relational operators.

APPLICATIONS ON BIG DATA USING PIG

- For exploring large datasets Pig Scripting is used.
- Provides supports across large data sets for Ad-hoc queries.
- In the prototyping of large data-sets processing algorithms.
- Required to process the time-sensitive data loads.
- For collecting large amounts of datasets in form of search logs and web crawls.
- Used where the analytical insights are needed using the sampling.

DATA PROCESSING OPERATORS IN PIG

1) LOADING & STORING DATA

- Pig Storage to store tuples as plain-text values separated by a colon character:
- EX:

  Joe:cherry:2
  Ali:apple:3
  Joe:banana:2
  Eve:apple:7

2) FILTERING DATA (FILTER OPERATOR)

- Once some data loaded into a relation, the next step is to filter it to remove the data that not interested in.
- By filtering early, minimize the amount of data flowing through the system, which can improve efficiency.

3) STREAM

- The **STREAM** operator allows you to transform data in a relation using an external program or script.
- It is named by analogy with Hadoop Streaming, which provides a similar capability for MapReduce.
- **STREAM** can use built-in commands with arguments.
- The **STREAM** operator uses PigStorage to serialize and deserialize relations to and from the program's standard input and output streams.

4) JOIN

- Pig has very good built-in support for join operations, making it much more approachable.
- Pig supports inner joins & outer joins.

5) COGROUP

- **JOIN** always gives a flat structure: a set of tuples.
- The COGROUP statement is similar to JOIN, but creates a nested set of output tuples.

6) GROUP

- Although **COGROUP** groups the data in two or more relations, the **GROUP** statement groups the data in a single relation.
- GROUP supports grouping by more than equality of keys.

7) CROSS

- Pig Latin includes the cross-product operator (also known as the cartesian product), which joins every tuple in a relation with every tuple in a second relation.
- The size of the output is the product of the size of the inputs, potentially making the output very large.

8) SORTING DATA

- Relations are unordered in Pig.

- We can ascend and descend data.
- The **LIMIT** statement is useful for limiting the number of results.
- Using LIMIT can improve the performance of a query.

9) COMBINING AND SPLITTING DATA

- We have several relations to combine into one. For this, the **UNION** statement is used.
- The **SPLIT** operator is the opposite of UNION; it partitions a relation into two or more relations.

HIVE

- One of the biggest ingredients in the Information Platform built by Jeff's team at Facebook was Hive, a framework for data warehousing on top of Hadoop.
- Hive grew from a need to manage and learn from the huge volumes of data.
- After trying a few different systems, the team chose Hadoop for storage and processing, since it was cost-effective and met their scalability needs.
- Hive was created to make it possible for analysts with strong SQL skills (but meager Java programming skills) to run queries on the huge volumes of data that Facebook stored in HDFS.
- Today, Hive is a successful Apache project used by many organizations as a general-purpose, scalable data processing platform.

HIVE SERVICES

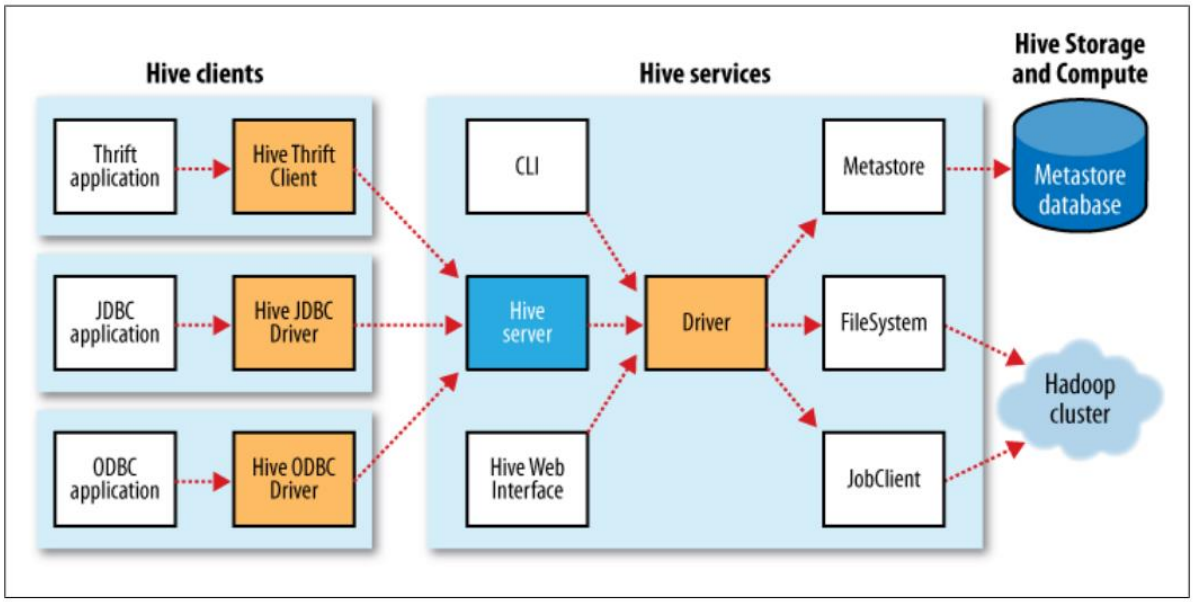
- You can specify the service to run using the **--service** option.
- Type **hive --service help** to get a list of available service names; the most useful are described below.

1) cli
   - The command line interface to Hive. This is the default service.
2) hiveserver
   - Runs Hive as a server, enabling access from a range of clients written in different languages. Applications using the JDBC, and ODBC connectors need to run a Hive server to communicate with Hive.
3) hwi
   - The Hive Web Interface.
4) jar
   - The Hive equivalent to **hadoop jar**, a convenient way to run Java applications that includes both Hadoop and Hive classes.
5) metastore
   - By default, the metastore is run in the same process as the Hive service.

- Using this service, it is possible to run the metastore as a standalone (remote) process.

HIVE ARCHITECTURE

- If you run Hive as a server (**hive --service hiveserver**), then there are a number of different mechanisms for connecting to it from applications.
- The relationship between Hive clients and Hive services is illustrated in the following figure.



HIVE QL

- Hive's SQL dialect, called HiveQL.
- The below table provides a high-level comparison of SQL and HiveQL.

Feature	SQL	HiveQL
Updates	UPDATE, INSERT, DELETE	INSERT OVERWRITE TABLE (populates whole table or partition)
Transactions	Supported	Not supported
Indexes	Supported	Not supported
Latency	Sub-second	Minutes
Data types	Integral, floating point, fixed point, text and binary strings, temporal	Integral, floating point, boolean, string, array, map, struct
Functions	Hundreds of built-in functions	Dozens of built-in functions
Multitable inserts	Not supported	Supported
Create table as select	Not valid SQL-92, but found in some databases	Supported
Select	SQL-92	Single table or view in the FROM clause. SORT BY for partial ordering. LIMIT to limit number of rows returned.

Feature	SQL	HiveQL
Joins	SQL-92 or variants (join tables in the FROM clause, join condition in the WHERE clause)	Inner joins, outer joins, semi joins, map joins. SQL-92 syntax, with hinting.
Subqueries	In any clause. Correlated or noncorrelated.	Only in the FROM clause. Correlated subqueries not supported
Views	Updatable. Materialized or nonmaterialized.	Read-only. Materialized views not supported
Extension points	User-defined functions. Stored procedures.	User-defined functions. MapReduce scripts.

QUERYING DATA IN HIVE

- The use various forms of the SELECT statement to retrieve data from Hive.

SORTING & AGGREGATING

- Sorting data in Hive can be achieved by use of a standard **ORDER BY** clause.
- **ORDER BY** produces a result that is totally sorted.
- **SORT BY** produces a sorted file per reducer.

MapReduce Script

- Using an approach like Hadoop Streaming, the **TRANSFORM, MAP,** and **REDUCE** clauses make it possible to invoke an external script or program from Hive.

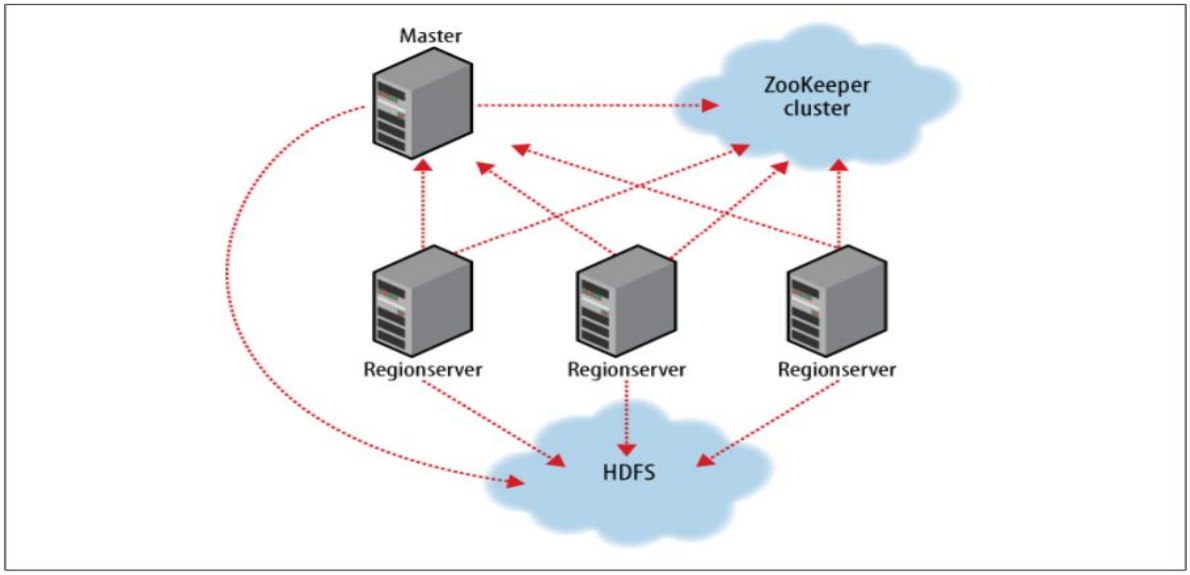
JOINS

- The simplest kind of join is the **inner join**, where each match in the input tables results in a row in the output.
- **Outer joins** allow you to find nonmatches in the tables being joined.

FUNDAMENTALS OF HBASE

- HBase is a distributed column-oriented database built on top of HDFS.
- HBase is the Hadoop application to use when you require real-time read/write random-access to very large datasets.
- HBase comes at the scaling problem from the opposite direction. It is built from the ground-up to scale linearly just by adding nodes.

- HBase is not relational and does not support SQL, but given the proper problem space, it is able to do what an RDBMS cannot: host very large, sparsely populated tables on clusters made from commodity hardware.
- Production users of HBase include Adobe, StumbleUpon, Twitter, and groups at Yahoo!.
- The below figure shows HBase cluster members.



CHARACTERISTICS OF HBASE

1) **No real indexes**: Rows are stored sequentially, as are the columns within each row. Therefore, no issues with index, and insert performance is independent of table size.
2) **Automatic partitioning**: As your tables grow, they will automatically be split into regions and distributed across all available nodes.
3) **Scale linearly and automatically with new nodes**: Add a node, point it to the existing cluster, and run the region server. Regions will automatically rebalance and load will spread evenly.
4) **Commodity hardware**: Clusters are built on less cost. RDBMSs are I/O hungry, requiring more costly hardware.
5) **Fault tolerance**: No need to worry about individual node downtime.
6) **Batch processing**: MapReduce integration allows fully parallel, distributed jobs against your data with locality awareness.

FUNDAMENTALS OF ZOOKEEPER

- For building general distributed applications using Hadoop's distributed coordination service, called **ZooKeeper**.
- ZooKeeper can't make partial failures go away.
- ZooKeeper give you a set of tools to build distributed applications that can safely handle partial failures.

- ZooKeeper also has the following **characteristics**:
  - **It is simple**
  - **It is expressive**: The ZooKeeper primitives are a rich set of building blocks that can be used to build a large class of coordination data structures and protocols.
  - **It is highly available**
  - **It facilitates loosely coupled interactions**: ZooKeeper interactions support participants that do not need to know about one another.
  - **ZooKeeper is a library**: ZooKeeper provides an open source, shared repository of implementations and recipes of common coordination patterns. Over time, the community can add to and improve the libraries, which is to everyone's benefit.
  - **ZooKeeper is highly performant**