Module 4: Hive and Pig

Hive

1. What's Hive?

- It's data warehousing architecture.
- Uses MapReduce & HDFS.
- Provides HQL.

2. Hive Functionality

- Querying/ analyzing.
- Manage unstructured data as structured.
- Leverage SQL skills.

3. Hadoop with Hive VS. RDBMS

- **Notes:**
  I. Built for different purposes and have their own pros and cons.
  II. Hive is not an alternative for RDBMS.
  III. Can co-exist in one system.
- **Comparison:**

	Hadoop with Hive	RDBMS
Supported data types / size	Petabytes of unstructured, semi-structured, and structured data	Terabytes of data and only structured data
Application latency	Supports high latency queries	Supports both high and low latency queries
Software type	Open source, flexible, fast and still evolving	Most are proprietary and defined constraints
Supported architecture	Distributed	Client server
Hardware requirements	Can run on commodity hardware	High-end server required for data intensive applications
Cost	Cost efficient	High cost to scale
Data handling features	Some traditional data handling features are not available in Hive. ACID principles are not available	Provides traditional features such as transaction management and ACID principles for data reliability
Schema policy	Schema on read policy	Schema on write policy

4. Hive Components

- Interfaces to Hadoop Framework: **Web UI, CLI, JDBC, ODBC**.
- **Driver** maintains a session handle and session statistics for query processing.
  I. Compiler
  - Parses the Hive query.
  - Converts queries into a MapReduce task.
  - Generates an execution plan.

II. **Optimizer**
   - Handles optimization tasks:
     - ✓ Column Pruning.
     - ✓ Partition Pruning.
     - ✓ Repartitioning of Data.

III. **Executor**
   - Executes the tasks.
   - Interacts with the underlying Hadoop instance.

- **Metastore**
  - Stores the system catalog, containing metadata about: Tables, Columns, Partitions…etc.
  - Stored in an RDBMS.
- **Thrift Server**
  - It's an optional server.
  - Exposes a client API to execute HQL statements.
  - Provides cross-language services.

5. Hive Interaction via CLI

- Most common way to interact with Hive.
- Provides the ability to issue DDL and metadata exploration commands.
- CLI is used to communicate with the Hadoop framework.

6. Hive Architecture: data Organization

Object	Description	Benefits
	• Catalog of namespaces that separate tables and other data units to avoid name conflicts `Hive> CREATE DATABASE Employee`	• Organizes production tables into logical groups • Load database into HDFS • Schema can evolve over time
Tables ID | Date 2 | 03/05/14	• Logical concept consisting of files in HDFS `Hive> CREATE TABLE sample(id int, name string);`	
Partitions 	• A directory	• Easier to query portions of the data • Reduces data read and filtered in map stages • Reduces mappers, I/O operations, and time
Buckets 	• A file in a table directory • Separates table data into more manageable parts	• Avoids having to create thousands of tiny partitions • Provides for more efficient types of queries

7. Data Organization: Two Table Types

- Internal/ Managed & External

	Internal / Managed	External
Storage	❑ HDFS	❑ Stored outside of Hive
Control	❑ Hive controls life cycle. ❑ Associated data is deleted with table	❑ Data does not get deleted when a table is deleted

8. Data Organization: View

- Allow queries to be saved and treated like a table.
- Reduce query complexity.

9. Data Organization: Indexes

- Indexes act as a reference to the records in a table.
- two types of indexes:
    1. Compaction
    2. Bitmap

10. Data Organization: Hive Metastore

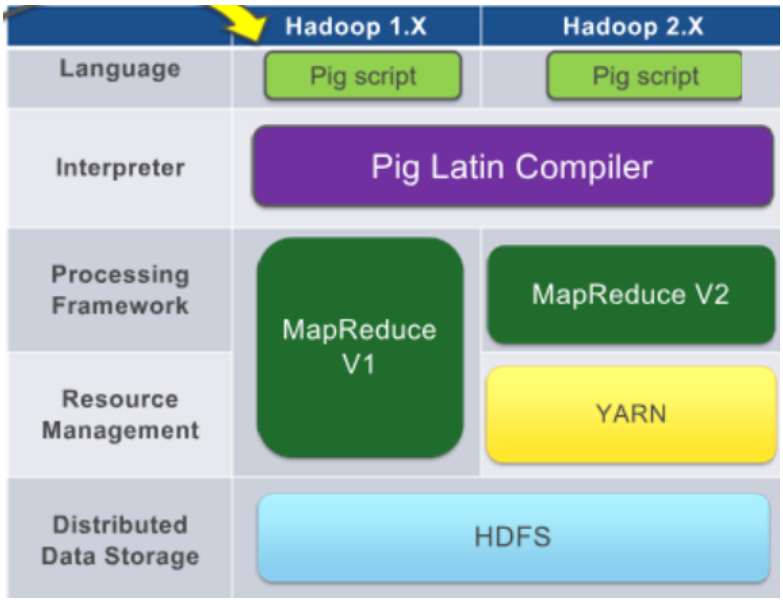
==========

Pig

11. Pig Overview

- Scripting language for analyzing large datasets.
- Appeals those familiar with scripting languages and SQL.

12. Pig Architecture: Overview



13. Pig Latin: Features

- Multi-query approach.
- Operators: join, sort, filer, etc.
- Nested data types: tuple, bags, and maps.
- Automatic Optimization.
- User Defined Functions (UDF).
- Structured and unstructured data.

14. Pig Latin Data Types

- **Data Atom**
  - **Description**
    - ✓ Stores a simple atomic data value.
    - ✓ Values are stored as a strings but can be used as either strings or numbers.
  - **Ex:** AWS
- **Tuple**
  - **Description**
    - ✓ A data record consisting of a sequence of “fields”.
    - ✓ Each filed is a piece of data of any type such as an atom, tuple or data bag.
    - ✓ In Pig, tuple in a bag can be compared to the rows in a table in a relational database.
    - ✓ A Tuple can also contain an ordered set of values.
  - **Ex:** (1, 2, 3)
- **Data Bag**
  - **Description**
    - ✓ A set of tuples.
    - ✓ Duplicate tuples are allowed.
    - ✓ Think of a data bag as a “table”, except that Pig does not require that the tuple field types match, or even that the tuples have the same number of fields.
    - ✓ Bags are an unordered collection of tuples.
  - **Ex:** {(1, 2),(3,4)}
- **Data Map**
  - **Description**
    - ✓ A set of key/ value pairs. Accessing a map with a specify key will return the value associated with that key.
  - **Ex:** [frog#kermit]

15. Pig Relations vs RDBMS Relations

- **Tuples**

Pig Relation
Big of tuples. It may have dedicated tuples.
RDBMS Relation
Set of tuples where every tuple is unique.

- **Columns**

Pig Relation
May have different number of columns.
RDBMS Relation
Has a fixed number of columns.

- **Column data types**

Pig Relation
Columns in the same position may have different data types.
RDBMS Relation
Columns in the same position have the same data type.

- **Procedural vs Declarative**

Pig Relation
Pig Latin is procedural.
RDBMS Relation
SQL is declarative.

- **Ability to add code**

Pig Relation
Pig Latin allows developers to insert their own code almost anywhere in the data pipeline.
RDBMS Relation
With traditional RDBMS systems, additional ETL tools are currently used to do customization of data.

- **Split support**

Pig Relation
Supports splits in the pipeline and data can be stores at any point in the pipeline.
RDBMS Relation
Splits are not supported and intermediate storage is not available.

- **Extract, Transform, Load**

Pig Relation
Pig uses ETL natively.
RDBMS Relation
Separate ETL tools are needed.

- **Evaluation**

Pig Relation
Pig use lazy evaluation.
RDBMS Relation
Instant invocation of commands happens in RDBMS.

- **Control statements**

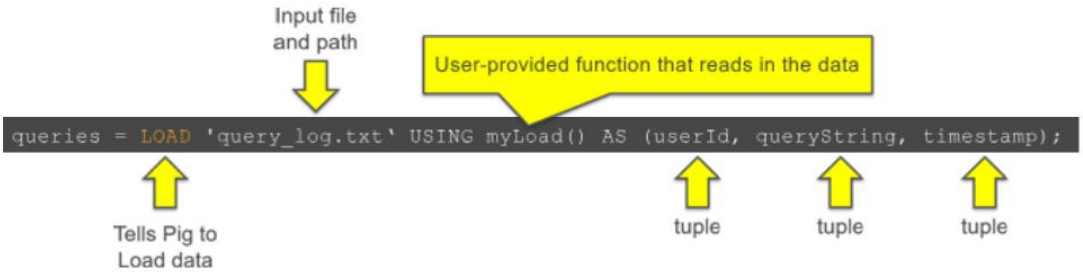
Pig Relation
There are no control statements such as if and else.
RDBMS Relation
Control statements are available.

16. Pig Latins: Schemas

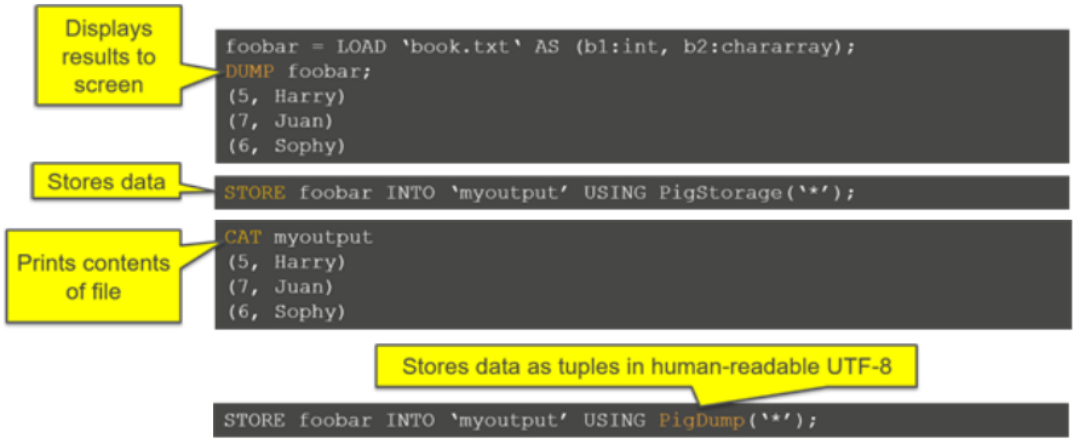
- Schemas assigns names and types to fields.
- Types provide better parse-time error checking.

Includes Field Name	Includes Data Type	Example
☑	☑	`foobar = LOAD 'book.txt' AS (b1:int, b2:chararray);`
☑	☒	`foobar = LOAD 'book.txt' AS (b1, b2);`
No Schema / Schema Unknown		`foobar = LOAD 'book.txt'`

17. Pig Latin: Input Data Flow



18. Pig Latin Architecture: Output Data Flow



19. Pig Latin Architecture: Running Pig Programs

Types of Execution

	Local Mode	MapReduce Mode
Description	• Pig only runs on one machine. • Files are run on localhost and file system.	• Pig translates queries into MapReduce jobs and runs them on a Hadoop cluster.
Example	`pig -x local`	`pig -x mapreduce`

Types of Invocation

	Interactive Mode / Grunt Shell	Batch Mode / Script Shell
Description	• Manual commands using Grunt • Useful for troubleshooting	• Group of Pig Latin statements in a Pig script to be run • Used in production environments

20. Pig Latin: User Defined Functions (UDF)

- Pig functions defined by user.
- Allow users to create custom processing.

21. Pig Join

- **Types of Joins**
  1. Self join.
  2. Inner join.
  3. Outer join: left/ right/ full.

22. Special Joins: Fragment Replicate Joins

- Improves performance.
- Requires one or more relations fit in memory.
- Large relation followed by one or more small relations.

Reference:

https://ithelp.ithome.com.tw/articles/10190597
https://zh.wikipedia.org/wiki/Apache_Hadoop
http://pcse.pw/9AR8F
https://pse.is/AQWTD