(5) Give a list of Builtin operators in PIG

Relational operators:

↳ allows you to transform data by sorting, grouping, joining, projecting and filtering

(i) LOAD

↳ load data from (LFS, HDFS) into pig relation

Ex: load1 = load '/—' using PigStorage(';') as (——)

(ii) FOREACH

↳ generates data transformations based on columns of data

It is used to add or remove fields from a relation

Ex: for_each = foreach load1 generate url, id;

(iii) FILTER

↳ selects tuples based on a condition

Ex: filter_group = filter load1 by id > 8;

(iv) JOIN

↳ perform a inner join of two or more relations based on common field values

Ex: join1 = join loading1 by url, loading1 by url;

(v) ORDERBY

order used to sort a relation based on one or more fields

Ex: loady = ORDER load1 by Mobj AVG;

(v) DISTINCT:

↳ removes duplicate tuples in a relation

Ex: load3 = distinct load1;

(vi) STORE:

↳ used to save results to the file system

Ex: store load3 into '/strg';

(vii) GROUP:

↳ groups tuples together based on same key values

↳ Key field will be a tuple if the group by has more than one field

Ex: ge = group load1 by url;

(viii) COGROUP:

↳ same as Group, used when multiple relations are involved

Ex: cg = cogroup loadstg1 by url, loadstg2 by url;

(x) CROSS:

↳ used to compute cross product of two or more relations

Ex: ce = cross load1, load2

(xi) LIMIT:

↳ limits the number of output tuples

Ex: le = limit load1 3;

(xii) SPLIT: partition a relation into two or more based on some expression

Ex: Split load1 into x if sal1>x, y if sal1<=8;

1) What are UDFs in Pig

Pig was designed to be extensible & customizable: all parts of Pig: loading, storing, filtering, grouping, joining can be altered by user defined functions. So If a function is not available you can write your own user defined function. UDFs can be written in Python, Java, Java Script, Ruby etc

- FilterUDFs are subclasses of EvalFunc which has a abstract method exec() where we implement our code. To use the new function we first compile it and package it in a JAR file. - then we tell pig using REGISTER operator the we can invoke the function

- An EvalUDF is a small step up from writing a filter function. To write a general eval udf you need to consider what the output's schema looks like. for scalars pig takes care but for complex types such as bags, tuples / maps. Pig needs more help you should implement outputSchema()

- A LoadUDF

1) ⑦ Compare Apache Pig, Map reduce & My SQL

Pig

- Pig Latin is a procedural language, high level language
- platform for analyzing large data sets
- Easy and convenient to program uses less lines of code
- High level of Abstraction
- development effort is less Code efficiency is relatively less

Hadoop Mapreduce

- Compiled language
- Lower level of Abstraction
- More lines of code
- More development effort is needed
- Code efficiency is high when compared to Pig

SQL

- declarative language
- Used for transactional & analytical queries
- Schema is mandatory
- Relatively less lines of code

3) Diagnostic Operators

(i) DUMP:

↳ runs pig comands & displays the results on screen

Ex: dump load1

(ii) DESCRIBE:

↳ to review the schema of a particular relation

Ex: describe load1

(iii) ILLUSTRATE:

↳ to review how data is transformed through a sequence of Pig Latin statement

↳ but used to debug

Ex: illustrate load1;

(iv) EXPLAIN:

↳ prints the logical & physical plans.

Ex: explain load1;

(2) Describe CAP THEOREM

↳ States that it is impossible for a distributed data store to simultaneously provide more than two out of the following

1) CONSISTENCY : Every read receives the most recent write or an error

2) AVAILABILITY : Every request a non-error response without a guarantee that it contains the most recent write

3) PARTITION TOLERANCE: The system continues to operate despite an arbitrary number of messages being dropped (delayed) by the n/w between nodes

(ie) In the presence of a n/w partition, one has to choose b/w consistency & availability

(4) What is the difference between No SQL and SQL

SQL databases are primarily categorised as Relational databases they have fixed (or) static or) predefined schema. SQL databases display data in form of tables and are vertically Scalable. SQL databases are a powerful Structured Query language to define, manipulate data. Best suited for complex queries and not suitable for hierarchial data storage. Ex: MySQL, Oracle, MS-SQL etc

NoSQL

- categorized as Non-relational or distributed database systems.
- have dynamic schema
- Display data as a collection of key-pair values, documents, graph databases
- horizontally scalable
- collection of documents are used to query the data. It is also called unstructured query language
- not so good for complex queries but best suited for hierarchical data storage

Ex: MongoDB, BigTable, Redis, RavenDB, Cassandra etc.