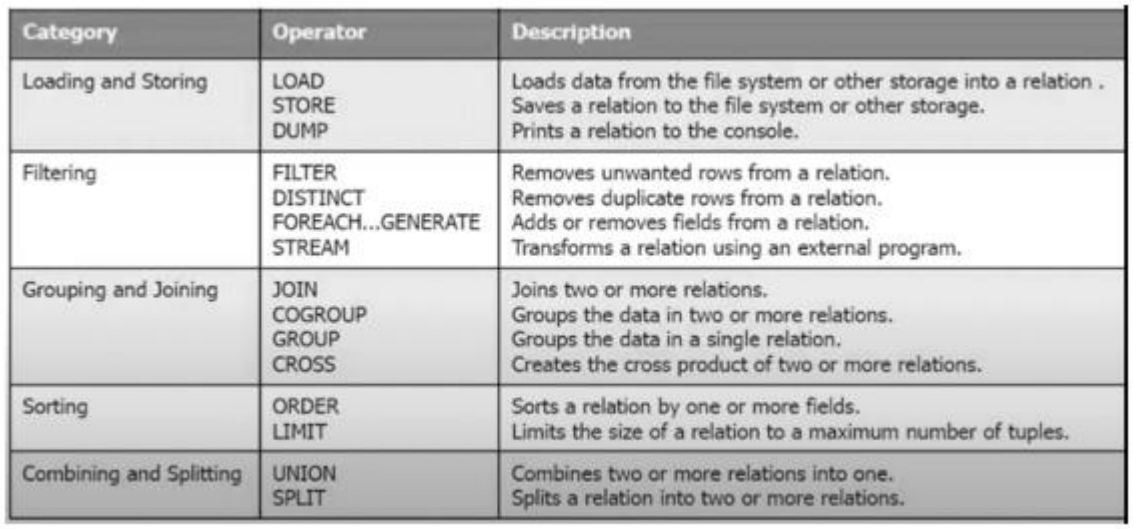
Pig Relational Operators:



Category	Operator	Description
Loading and Storing	LOAD STORE DUMP	Loads data from the file system or other storage into a relation . Saves a relation to the file system or other storage. Prints a relation to the console.
Filtering	FILTER DISTINCT FOREACH...GENERATE STREAM	Removes unwanted rows from a relation. Removes duplicate rows from a relation. Adds or removes fields from a relation. Transforms a relation using an external program.
Grouping and Joining	JOIN COGROUP GROUP CROSS	Joins two or more relations. Groups the data in two or more relations. Groups the data in a single relation. Creates the cross product of two or more relations.
Sorting	ORDER LIMIT	Sorts a relation by one or more fields. Limits the size of a relation to a maximum number of tuples.
Combining and Splitting	UNION SPLIT	Combines two or more relations into one. Splits a relation into two or more relations.

Loading and Storing

LOAD

$LOAD 'info' [USING FUNCTION] [AS SCHEMA];

- o LOAD is a relational operator.
- o 'info' is a file that is required to load. It contains any type of data.
- o USING is a keyword.
- o FUNCTION is a load function.
- o AS is a keyword.
- o SCHEMA is a schema of passing file, enclosed in parentheses.

Example:

• File in local file system

```
$cat data.txt
1010,10,3
2020,20,4
3030,30,5
4040,40,2
```

• Loading data file into HDFS file system

```
$ hdfs dfs -put data.txt /pigtest
```

• Starting pig grunt shell

```
$pig -x mapreduce or $pig
```

• Loading data into pig by defining schema and fields are separated with comma.

```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',') AS (d1:int,d2:int,d3:int) ;
```

• Printing loaded data on console

```
grunt> DUMP A;
(1010,10,3)
(2020,20,4)
(3030,30,5)
(4040,40,2)
```

STORE

• Stores or saves results to the file system.

```
grunt>STORE A INTO 'myoutput' USING PigStorage ('*');
```

<u>Filtering :</u>

FILTER

• File in local file system

```
$cat data.txt
1,10,3
2,20,4
3,10,3
4,20,4
```

• Loading data file into HDFS file system

```
$ hdfs dfs -put data.txt /pigtest
```

• Starting pig grunt shell

```
$pig -x mapreduce or $pig
```

• Loading data into pig by defining schema and fields are separated with comma.

```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',') AS (d1:int,d2:int,d3:int) ;
```

• To remove duplicate data

```
grunt>B = FILTER A BY d2 == 10;
```

• Printing loaded data on console

```
grunt> DUMP B;
(1,10,3)
(3, 10,3)
```

FOREACH

• File in local file system

```
$cat data.txt
1,2,3,4
5,6,7,8
8,7,6,5
4,3,2,1
```

• Loading data file into HDFS file system

```
$ hdfs dfs -put data.txt /pigtest
```

• Starting pig grunt shell

```
$pig -x mapreduce or $pig
```

• Loading data into pig by defining schema and fields are separated with comma.

```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',') AS (d1:int,d2:int,d3:int, d4:int) ;
```

• To fetch second and fourth columns

```
grunt>B = FOREACH A GENERATE d2,d4;
```

• Printing loaded data on console

```
grunt> DUMP B;
(2,4)
(6,8)
(7,5)
(3,1)
```

DISTINCT

• File in local file system

```
$cat data.txt
1,10,3
2,20,4
1,10,3
2,20,4
```

• Loading data file into HDFS file system

```
$ hdfs dfs -put data.txt /pigtest
```

• Starting pig grunt shell

```
$pig -x mapreduce or $pig
```

• Loading data into pig by defining schema and fields are separated with comma.

```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',') AS (d1:int,d2:int,d3:int) ;
```

• To remove duplicate data

```
grunt>B = DISTINCT A
```

• Printing loaded data on console

```
grunt> DUMP B;
(1, 10, 3)
(2, 20, 4)
```

Grouping and Joining

CROSS

• File in local file system

```
$cat data1.txt
1,2
2,3
$cat data2.txt
3,4,5
4,5,6
```

• Loading data file into HDFS file system

```
$ hdfs dfs -put data1.txt /pigtest
$ hdfs dfs -put data2.txt /pigtest
```

- Starting pig grunt shell

```
$pig
```

- Loading data into pig by defining schema and fields are separated with comma.

```
grunt> A = LOAD '/pigtest/data1.txt' USING PigStorage(',') AS (d1:int,d2:int) ;
grunt> B = LOAD '/pigtest/data2.txt' USING PigStorage(',') AS (d1:int,d2:int,d3:int) ;
```

- Cross product of data1.txt and data2.txt

```
grunt> C=CROSS A,B;
```

- Printing final output

```
grunt> DUMP C;
(1,2,3,4,5)
(1,2,4,5,6)
(2,3,3,4,5)
(2,3,4,5,6)
```

GROUP BY

- File in local file system

```
$cat data.txt
John,Ram,3
Clark,John,2
Nike,Ram,5
Imran,John,6
```

- Loading data file into HDFS file system

```
$ hdfs dfs -put data.txt /pigtest
```

- Starting pig grunt shell

```
$pig -x mapreduce or $pig
```

- Loading data into pig by defining schema and fields are separated with comma.

```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',')
AS (d1:chararray,d2:chararray,d3:int) ;
```

- To group the data based on d2 column data

```
grunt>B = GROUP A BY d2;
```

- Printing loaded data on console

```
grunt> DUMP B;
(Ram, {(John,Ram,3), (Nike,Ram,5)})
(John, {(Clark,John,2), (Imran,John,6)}
```

JOIN

- File in local file system

```
$cat student.txt
1,Ram,9.8
2,John,7.8
3,Ram,6.7
4,John,6.6
```

```
$cat department.txt
1,101,IT
2,101,IT
3,101,IT
4,101,IT
```

- Loading data file into HDFS file system
```
$ hdfs dfs -put student.txt /pigtest
$ hdfs dfs -put department.txt /pigtest
```
- Starting pig grunt shell
```
$pig -x mapreduce or $pig
```
- Loading data into pig by defining schema and fields are separated with comma.
```
grunt> A = LOAD '/pigtest/student.txt' USING PigStorage(',')
AS (rollno:int, name:chararray ,gpa:float ) ;
grunt> B = LOAD '/pigtest/department.txt' USING PigStorage(',')
AS (rollno:int, deptno:int ,deptname:chararray ) ;
```
- To join the data based on rollno
```
grunt>C = JOIN A BY rollno, B BY rollno;
```
- Printing loaded data on console
```
grunt> DUMP C;
1,Ram,9.8,1,101,IT
2,John,7.8,1,101,IT
3,Ram,6.7,1,101,IT
4,John,6.6,1,101,IT
```

Sorting:

ORDER BY

- File in local file system
```
$cat data.txt
John,Ram,3
Clark,John,2
Nike,Ram,5
Imran,John,6
```
- Loading data file into HDFS file system
```
$ hdfs dfs -put data.txt /pigtest
```
- Starting pig grunt shell
```
$pig -x mapreduce or $pig
```
- Loading data into pig by defining schema and fields are separated with comma.
```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',')
        AS (d1:chararray,d2:chararray,d3:int) ;
```
- To sort tuples in an Order
```
grunt>B = ORDER A BY d3 DESC;
```

• Printing loaded data on console
```
grunt> DUMP B;
          Imran,John,6
          Nike,Ram,5
          John,Ram,3
          Clark,John,2
```

LIMIT

• File in local file system
```
$cat data.txt
John,Ram,3
Clark,John,2
Nike,Ram,5
Imran,John,6
```
• Loading data file into HDFS file system
```
$ hdfs dfs -put data.txt /pigtest
```
• Starting pig grunt shell
```
$pig -x mapreduce or $pig
```
• Loading data into pig by defining schema and fields are separated with comma.
```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',')
        AS (d1:chararray,d2:chararray,d3:int) ;
```
• To print only first two tuples
```
grunt>B = LIMIT A 2;
```
• Printing loaded data on console
```
grunt> DUMP B;
         John,Ram,3
         Clark,John,2
```

<u>Combining and Splitting:</u>

UNION

• File in local file system
```
$cat data1.txt
John,Ram,3
Clark,John,2
$cat data2.txt
Nike,Ram,5
Imran,John,6
```
• Loading data file into HDFS file system
```
$ hdfs dfs -put data1.txt /pigtest
$ hdfs dfs -put data2.txt /pigtest
```
• Starting pig grunt shell
```
$pig -x mapreduce or $pig
```
• Loading data into pig by defining schema and fields are separated with comma.
```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',')
          AS (d1:chararray,d2:chararray,d3:int) ;
```

```
grunt> B = LOAD '/pigtest/data2.txt' USING PigStorage(',')
        AS (d1:chararray,d2:chararray,d3:int) ;
```
• To combine two bags as one bag

```
grunt>C = UNION A,B;
```
• Printing loaded data on console
```
grunt> DUMP C;
        John,Ram,3
        Clark,John,2
        Nike,Ram,5
        Imran,John,6
```

SPLIT

• File in local file system
```
$cat data.txt
1,2
2,4
3,6
4,8
5,7
6,5
7,3
8,1
```
• Loading data file into HDFS file system
```
$ hdfs dfs -put data.txt /pigtest
```
• Starting pig grunt shell
```
$pig -x mapreduce or $pig
```
• Loading data into pig by defining schema and fields are separated with comma.
```
grunt> A = LOAD '/pigtest/data.txt' USING PigStorage(',') AS (d1:int,d2:int) ;
```
• To Split the tuples based on field values
```
grunt> SPLIT A INTO X IF d1<=5, Y IF d1>=6;
```
• Printing loaded data on console
```
grunt> DUMP X;
(1,2)
(2,4)
(3,6)
(4,8)
(5,7)
grunt> DUMP Y;
(6,5)
(7,3)
(8,1)
```

WEEK-10

PIG PROGRAMS

OBJECTIVE:

1. Run the Pig Latin Scripts to find Word Count.
2. Run the Pig Latin Scripts to find a max temp for each and every year.

PROGRAM LOGIC:

Run the Pig Latin Scripts to find Word Count.

```
lines = LOAD '/user/hadoop/HDFS_File.txt' AS (line:chararray);
words = FOREACH lines GENERATE FLATTEN(TOKENIZE(line)) as word;
grouped = GROUP words BY word;
wordcount = FOREACH grouped GENERATE group, COUNT(words);
DUMP wordcount;
```

Run the Pig Latin Scripts to find a max temp for each and every year

```
records = LOAD 'input/ncdc/micro-tab/sample.txt' AS (year:chararray, temperature:int, quality:int);
filtered_records = FILTER records BY temperature != 9999
AND
(quality == 0 OR quality == 1 OR quality == 4 OR quality == 5 OR quality == 9);
grouped_records = GROUP filtered_records BY year;
max_temp = FOREACH grouped_records GENERATE group, MAX(filtered_records.temperature);
DUMP max_temp;
```

(Execute above two programs and write the output) on the left page with input file data and the output