COSC 416
NoSQL Databases

Apache Pig

Dr. Ramon Lawrence
University of British Columbia Okanagan
ramon.lawrence@ubc.ca

Apache Pig

Apache Pig is a high-level language for expressing Map-Reduce programs.

Pig defines a language called *Pig Latin* that is translated into a sequence of Map-Reduce programs.

This speeds up the time to write Map-Reduce data analysis programs and improves performance rather than users writing code themselves.

Pig Latin

Pig Latin is very similar to relational algebra.

Each operator takes a relation as input and produces a relation as output. Each statement may use expressions and a schema.

Basic program structure:

- LOAD – one or more files from HDFS
- Perform transformation statements
- DUMP (to write output to screen) or STORE to save results to a file. Note that a Map-Reduce program is only generated and run when a DUMP or STORE is encountered.

Pig Latin Basic Rules

1) Names (aliases) of relations and fields are case-sensitive.

2) Function names (e.g. COUNT) are case-sensitive.

3) Operator keywords (e.g. LOAD, store, GROUP) are not case-sensitive.

4) Identifiers must start with a letter and may have digits and underscore.

5) Can reference fields by name or by position. First field is referenced by $0.

6) A relation is a bag. A bag is a collection of tuples. A tuple is an ordered set of fields. A field is data. Each tuple does not have to have the same fields.

Pig Latin Operators LOAD

LOAD reads a file from HDFS and references it with a variable.

You may specify the loader class and provide a schema to describe the records (both optional).

Syntax:

```
LOAD 'data' [USING function] [AS schema];
```

Example:

```
R = LOAD 'myfile.txt' USING PigStorage()
             AS (id:int, name:chararray);
```

- Loads text file using default loader and applies given schema.
- File is now referenced with identifier **R**.

Pig Latin Operators FOREACH (Projection/Iteration)

FOREACH performs column transformations of data such as projections and expression generation.

- Loops through input records one at a time and produces relation of output records.

Syntax:

```
alias  = FOREACH { block | nested_block };
```

Example:

```
X = FOREACH R GENERATE A1, A2;
Y = FOREACH R GENERATE A1, SUM(A2), A3+A4;
```

- Expressions and functions are allowed.
- Can nest FOREACH to two levels.
- FLATTEN operator handles nested tuples.

Pig Latin Operators FILTER (Selection)

FILTER performs selection (filters) on input.

Syntax:

```
alias  = FILTER alias BY expression;
```

Example:

```
X = FILTER R BY A1 == 3;
Y = FILTER R BY A1 > A2;
```

Pig Latin Operators JOIN

JOIN performs relational inner and outer joins.

Syntax:

```
alias  = JOIN alias BY expression, alias BY expression, …
```

Example:

```
X = JOIN R BY R1, S BY S1;
```

- Special settings to handle skew and to select merge joins.
- May also specify LEFT/RIGHT/FULL OUTER joins.

Pig Latin Operators ORDER BY

ORDER BY performs sorting.

Syntax:

```
alias  = ORDER alias BY field [ASC | DESC]
```

Example:

```
 B = ORDER A BY F1;
```

- Sorting is not stable (may change between runs).
- Cannot order by fields with complex types or expressions.
- Can specify * to order by entire tuple.

Pig Latin Operators GROUP

GROUP performs relational grouping.

Syntax:

```
alias   = GROUP alias BY expression, alias BY expression, …
```

Example:

```
B = GROUP A BY F1;
C = FOREACH B GENERATE group, COUNT(A);
C = FOREACH B GENERATE $0, $1
```

- May use expressions for grouping.
- Can group on multiple relations at the same time.
- If do not specify a relation, can refer to grouping expression using group or positional notation.

Pig Latin Operators DUMP/STORE

DUMP writes an output relation to standard output.

STORE writes an output relation to a HDFS file.

Syntax:

```
DUMP alias;
STORE alias INTO 'file' [USING function];
```

Example:

```
DUMP R;
STORE R INTO 'myoutput.txt';
```

Pig Latin Operators
Other Useful Operators

DISTINCT removes duplicate tuples in a relation.

SAMPLE partitions a relation into two or more relations. Takes a random sample from the input relation.

SPLIT partitions a relation into two or more relations using an expression.

STREAM sends data to an external script or program.

UNION computes the union of two or more relations.

REGISTER registers a JAR that contains UDFs.

Pig Latin Operators DESCRIBE and EXPLAIN

DESCRIBE shows the relation for the alias.

EXPLAIN shows the execution plan.

ILLUSTRATE provides an example execution.

Example:

```
R = LOAD 'myfile.txt' AS (id:int, name:chararray);
A = FILTER R BY id > 5;
DESCRIBE A;
Output:
A: {id: int, name: chararray}

EXPLAIN A;
Output: Shows an execution plan in Map Reduce.

ILLUSTRATE A;
Output: Shows an example output on each stage of the plan.
```

Conclusion

Apache Pig simplifies building Map-Reduce program by+ constructing scripts of relational operators.

These operators, like relational algebra, provide abstraction from the computation and data. They are easier to write and maintain than Map-Reduce programs themselves.

Objectives

Understand the basic operators in Pig Latin.

Be able to write queries in Pig to answer English questions.