**Different complex data types in pig**

**Tuple**

A record that is formed by an ordered set of fields is known as a tuple, the fields can be of any type. A tuple is similar to a row in a table of RDBMS.

**Example** − (Raja, 30)

**Bag**

A bag is an unordered set of tuples. In other words, a collection of tuples (non-unique) is known as a bag. Each tuple can have any number of fields (flexible schema). A bag is represented by ‘{}’. It is similar to a table in RDBMS, but unlike a table in RDBMS, it is not necessary that every tuple contain the same number of fields or that the fields in the same position (column) have the same type.

**Example** − {(Raja, 30), (Mohammad, 45)}

A bag can be a field in a relation; in that context, it is known as **inner bag**.

**Example** − {Raja, 30, **{9848022338, raja@gmail.com,}**}

**Map**

A map (or data map) is a set of key-value pairs. The **key** needs to be of type chararray and should be unique. The **value** might be of any type. It is represented by ‘[]’

**Example** − [name#Raja, age#30]

**Relation**

A relation is a bag of tuples. The relations in Pig Latin are unordered (there is no guarantee that tuples are processed in any particular order).

**Interact with the shell in Apache pig**

## **Invoking the Grunt Shell**

You can invoke the Grunt shell in a desired mode (local/MapReduce) using the **−x** option as shown below.

**Command −**

$ ./pig –x local

$ ./pig -x mapreduce

After invoking the Grunt shell, you can run your Pig scripts in the shell. In addition to that, there are certain useful shell and utility commands provided by the Grunt shell. This chapter explains the shell and utility commands provided by the Grunt shell.

**Explain how pig differs from Map reduce**

**PIG** is a data flow language, the key focus of Pig is manage the flow of data from input source to output store. As part of managing this data flow it moves data feeding it to p1, taking output and feeding it to p2. The core features are preventing execution of subsequent stages if previous stage fails, manages temporary storage of data and most importantly compresses and rearranges processing steps for faster processing. While this can be done for any kind of processing tasks Pig is written specifically for managing data flow of Map reduce type of jobs. Most if not all jobs in a Pig are map reduce jobs or data movement jobs. Pig allows for custom functions to be added which can be used for processing in Pig, some default ones are like ordering, grouping, distinct, count etc.

**Map reduce** on the other hand is a data processing paradigm, it is a framework for application developers to write code in so that its easily scaled to PB of tasks, this creates a separation between the developer that writes the application vs the developer that scales the application. Not all applications can be migrated to Map reduce but good few can be including complex ones like k-means to simple ones like counting uniques in a dataset.

**Explain how pig differs from sql**

Usability is one difference between Pig and Sql.

Lets consider the scenario of ad-hoc queries. Data analyst/scientist want to analyze a particular trend on a particular day. He will also need to analyze a diffrent trend another day as well. If he is going to use hive for this he will suffer for sure. He will be writing map-reduce jobs everytime and modifying his queries. This is where Pig comes to rescue. He just needs to learn Pig Latin then deal directly with unstructured/semi-structured data.

**Explain the scalar data types in pig**

**Scalar Data Types:**

1. int: Signed 32-bit integer. Example: 10
2. long: Signed 64-bit integer. Example: 10l
3. float: 32-bit floating point. Example: 10.5F
4. double: 64-bit floating point. Example: 10.5e2
5. Arrays:
6. chararray: Character array (string) in Unicode UTF-8 format. Example: hello world.
7. bytearray: Byte array (blob)