1. **Explain about the different complex data types in pig**

There are 4 complex DataTypes.They are

1.Atom

2.Tuple

3.Bag

4.Map

5.Relation

**Atom**

1. Any single value in Pig Latin, irrespective of their data or type is known as an Atom.
2. It is stored as bytearray by default and can be used as string or number like int, long, float, double, chararray, and bytearray are the atomic values of Pig.
3. A piece of data or a simple atomic value is known as a field. Example − ‘raja’ or ‘30’

**Tuple**

1. A record that is formed by an ordered set of fields is known as a tuple, the fields can be of any type.
2. A tuple is similar to a row in a table of RDBMS.
3. A tuple is represented by ‘()’.

Example − (Raja, 30)

**Bag**

1. A bag is an unordered set of tuples.
2. In other words, a collection of tuples (non-unique) is known as a bag.
3. Each tuple can have any number of fields (flexible schema).
4. A bag is represented by ‘{}’.
5. 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)}

1. 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,}](mailto:raja@gmail.com,%7d)}

**Map**

1. A map (or data map) is a set of key-value pairs.
2. The key needs to be of type chararray and should be unique.
3. The value might be of any type. It is represented by ‘[]’ Example: [name#Raja, age#30]

**Relation**

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

**2.How can you interact with the shell in Apache pig**

We can interact Pig shell by

**1.** With Grunt Shell

**2.** Using Script Files

**With Grunt Shell**

• Interactive Shell for executing Pig Commands

• Used when script file is not provided

• Can execute scripts from Grunt via run or exec commands

**Using Script Files**

• Executes Commands in a file • pig ScriptFile.pig

• Pig commands are executed using script files as batch Jobs

1. **Explain how pig differs from Map reduce**

|  |  |
| --- | --- |
| **PIG** | **MAP REDUCE** |
| It can be used for Structured and semi structured data | It can be used for Structured ,semi structured and un structured data |
| It can be useful for effectively performing joins,querying etc | It can be used for only map reduce |
| Development time is less (code development) | Development time is high (code development) |
| Processing Time is high since it runs on top of map reduce | Processing Time is comparatively less |
| Useful for performing anlysis |  |

**4.Explain how pig differs from sql**

|  |  |
| --- | --- |
| **Pig** | **SQL** |
| Pig Latin is a **procedural** language. | SQL is a **declarative** language. |
| In Apache Pig, **schema** is optional. We can store data without designing a schema (values are stored as $01, $02 etc.) | Schema is mandatory in SQL. |
| The data model in Apache Pig is **nested relational**. | The data model used in SQL **is flat relational**. |
| Apache Pig provides limited opportunity for **Query optimization**. | There is more opportunity for query optimization in SQL. |

**5.Explain the scalar data types in pig**

|  |  |  |
| --- | --- | --- |
| **S.N.** | **Data Type** | **Description & Example** |
| 1 | int | Represents a signed 32-bit integer.  **Example** : 8 |
| 2 | long | Represents a signed 64-bit integer.  **Example** : 5L |
| 3 | float | Represents a signed 32-bit floating point.  **Example** : 5.5F |
| 4 | double | Represents a 64-bit floating point.  **Example** : 10.5 |
| 5 | chararray | Represents a character array (string) in Unicode UTF-8 format.  **Example** : ‘tutorials point’ |
| 6 | Bytearray | Represents a Byte array (blob). |
| 7 | Boolean | Represents a Boolean value.  **Example** : true/ false. |
| 8 | Datetime | Represents a date-time.  **Example** : 1970-01-01T00:00:00.000+00:00 |
| 9 | Biginteger | Represents a Java BigInteger.  **Example** : 60708090709 |
| 10 | Bigdecimal | Represents a Java BigDecimal  **Example** : 185.98376256272893883 |