Problem Statement:

Explain File Formats in Hive in Brief with an example.

**File Format:**  
A **file format** is a way in which information is stored or encoded in a computer file. In Hive it refers to how records are stored inside the file. As we are dealing with structured data, each record has to be its own structure. How records are encoded in a file defines a file format.  
These file formats mainly vary between data encoding, compression rate, usage of space and disk I/O.  
Hive does not verify whether the data that you are loading matches the schema for the table or not. However, it verifies if the file format matches the table definition or not.

**1. TEXTFILE:**

TEXTFILE format is a famous input/output format used in Hadoop. In Hive if we define a table as TEXTFILE it can load data of from CSV (Comma Separated Values), delimited by Tabs, Spaces, and JSON data. This means fields in each record should be separated by comma or space or tab or it may be JSON (JavaScript Object Notation) data.  
By default, if we use TEXTFILE format then each line is considered as a record.

**We can create a TEXTFILE format in Hive as follows:**

|  |  |
| --- | --- |
| 1 | create table table\_name (schema of the table) row format delimited fields terminated by ',' | stored as TEXTFILE. |

At the end, we need to specify the type of **file format**. If we do not specify anything it will consider the file format as TEXTFILE format.  
The TEXTFILE input and TEXTFILE output format are present in the Hadoop package as shown below:

|  |  |
| --- | --- |
| 1  2 | org.apache.hadoop.mapred.TextInputFormat  org.apache.hadoop.mapred.TextOutputFormat |

Let us see one example in Hive about how to create TEXTFILE table format, how to load data into TEXTFILE format and perform one basic select operation in Hive.

**Creating TEXTFILE**

|  |  |
| --- | --- |
| 1 | create table olympic(athelete STRING,age INT,country STRING,year STRING,closing STRING,sport STRING,gold INT,silver INT,bronze INT,total INT) row format delimited fields terminated by '\t' stored as textfile; |

Here we are creating a table with name “olympic” and the schema of the table is as specified above. The data inside the above input file is delimited by tab space. As explained earlier the file format is specified as TEXTFILE at the end. The schema of the table created above can be checked using describe olympic;

**2. SEQUENCEFILE:**

We know that Hadoop’s performance is drawn out when we work with a small number of files with big size rather than a large number of files with small size. If the size of a file is smaller than the typical block size in Hadoop, we consider it as a small file. Due to this, a number of metadata increases which will become an overhead to the NameNode. To solve this problem sequence files are introduced in Hadoop. Sequence files act as a container to store the small files.

Sequence files are flat files consisting of binary key-value pairs. When Hive converts queries to MapReduce jobs, it decides on the appropriate key-value pairs to be used for a given record. Sequence files are in the binary format which are able to split and the main use of these files is to club two or more smaller files and make them as a one sequence file.

In Hive we can create a sequence file by specifying STORED AS SEQUENCEFILE in the end of a CREATE TABLE statement.  
There are three types of sequence files:  
• Uncompressed key/value records.  
• Record compressed key/value records – only ‘values’ are compressed here  
• Block compressed key/value records – both keys and values are collected in ‘blocks’ separately and compressed. The size of the ‘block’ is configurable.

Hive has its own SEQUENCEFILE reader and SEQUENCEFILE writer for reading and writing through sequence files.

**In Hive we can create a sequence file format as follows:**

|  |  |
| --- | --- |
| 1 | create table table\_name (schema of the table) row format delimited fields terminated by ',' | stored as SEQUENCEFILE |

Hive uses the SEQUENCEFILE input and output formats from the following packages:

|  |  |
| --- | --- |
| 1  2 | org.apache.hadoop.mapred.SequenceFileInputFormat  org.apache.hadoop.hive.ql.io.HiveSequenceFileOutputFormat |

**Creating SEQUENCEFILE**

|  |  |
| --- | --- |
| 1 | create table olympic\_sequencefile(athelete STRING,age INT,country STRING,year STRING,closing STRING,sport STRING,gold INT,silver INT,bronze INT,total INT) row format delimited fields terminated by '\t' stored as sequencefile |

**3. RCFILE:**

RCFILE stands of Record Columnar File which is another type of binary file format which offers high compression rate on the top of the rows.  
RCFILE is used when we want to perform operations on multiple rows at a time.  
RCFILEs are flat files consisting of binary key/value pairs, which shares much similarity with SEQUENCEFILE. RCFILE stores columns of a table in form of record in a columnar manner. It first partitions rows horizontally into row splits and then it vertically partitions each row split in a columnar way. RCFILE first stores the metadata of a row split, as the key part of a record, and all the data of a row split as the value part. This means that RCFILE encourages column oriented storage rather than row oriented storage.  
This column oriented storage is very useful while performing analytics. It is easy to perform analytics when we “hive’ a column oriented storage type.  
Facebook uses RCFILE as its default file format for storing of data in their data warehouse as they perform different types of analytics using Hive.

**In Hive we can create a RCFILE format as follows:**

|  |  |
| --- | --- |
| 1 | create table table\_name (schema of the table) row format delimited fields terminated by ',' | stored as RCFILE |

Hive has its own RCFILE Input format and RCFILE output format in its default package:

|  |  |
| --- | --- |
| 1  2 | org.apache.hadoop.hive.ql.io.RCFileInputFormat  org.apache.hadoop.hive.ql.io.RCFileOutputFormat |

**Creating RCFILE**

|  |  |
| --- | --- |
| 1 | create table olympic\_rcfile(athelete STRING,age INT,country STRING,year STRING,closing STRING,sport STRING,gold INT,silver INT,bronze INT,total INT) row format delimited fields terminated by '\t' stored as rcfile |

**4. ORCFILE:**

ORC stands for Optimized Row Columnar which means it can store data in an optimized way than the other file formats. ORC reduces the size of the original data up to 75%. As a result the speed of data processing also increases. ORC shows better performance than Text, Sequence and RC file formats.  
An ORC file contains rows data in groups called as Stripes along with a file footer. ORC format improves the performance when Hive is processing the data.

**In Hive we can create a RCFILE format as follows:**

|  |  |
| --- | --- |
| 1 | create table table\_name (schema of the table) row format delimited fields terminated by ',' | stored as ORC |

Hive has its own ORCFILE Input format and ORCFILE output format in its default package:

|  |  |
| --- | --- |
| 1 | org.apache.hadoop.hive.ql.io.orc |

**Creating ORCFILE**

|  |  |
| --- | --- |
| 1 | create table olympic\_orcfile(athelete STRING,age INT,country STRING,year STRING,closing STRING,sport STRING,gold INT,silver INT,bronze INT,total INT) row format delimited fields terminated by '\t' stored as orcfile; |