***Transformation Check Test Framework***

This framework is basically compares the data frames from two different data source and checks for the schema comparison, Row comparison, Data Comparison. A data ingestion framework which will help to carry out rigorous testing in less time and less Efforts. This framework has two main components:-

1. configuration File
2. code file.

The framework code is written in the simplest form but it’s completely robust.

**Configuration File:-**

Configuration file is basically a **JSON** file with 4 parts. Each part is a key which a Dictionary value.

Let’s go through each of the part one by one:-

1. **Source:**

The Source is explaining us about the source of the data. Currently we are supporting three types of data source and they are: Hive, RDBMS and Flat files. Whatever be the data source we need to enable that key with true value.

1. "isDirectHive": true // This key tells the data source is Hive if True  
   "driverName":"", // This key takes the driver name if any for connection  
   "databaseUrl":"", // We have to specify database url in case the data

source is database  
"databaseUserName":"", // Username for database  
"databasePassword":"", // Password to connect to the database  
"query":"", //Query we need to test  
"isFile":"", // This key tells the data source is Flat File if True   
"fileName":"", // Specify the filename and path here  
"fileWithSchema":"" // Specify the schema and delimiter of the file here.

1. **Target:**

The Target is explaining us about the target of the data. Currently we are supporting three types of data target and they are: Hive, RDBMS and Flat files. Whatever be the data source we need to enable that key with true value.

1. "isDirectHive": true // This key tells the data target is Hive if True  
   "driverName":"", // This key takes the driver name if any for connection  
   "databaseUrl":"", // We have to specify database url in case the data

target is database  
"databaseUserName":"", // Username for database  
"databasePassword":"", // Password to connect to the database  
"query":"", //Query we need to test for target table  
"isFile":"", // This key tells the data target is Flat File if True   
"fileName":"", // Specify the filename and path here  
"fileWithSchema":"" // Specify the schema and delimiter of the file here.

1. **Test:**

This component explains us what all test we want to perform. We have the option to run all three test together or in any sort of combination. Whatever key will have a true value that particular test will be executed. It will also write the mismatched data to a file.

1. "schemaCompare":true, // This key enables schema Compare test if True  
   "rowCountCompare":true, // This key enables row count Compare test if

True  
"dataCompare": true, // This key enables data Compare test if True  
"dataCompareFileName":"path" // File name and path to write mismatch data

1. **Log:**

This component tells us where the log files will be kept. We are maintain a log File which gives us a lot of info about the execution of our code.

"fileName":"path" // This takes the file name and file path where we want to store the logs of execution.