Analyse Employee Dataset using Hadoop :

1. Using Hadoop command move all those employees data into HDFS directory "/user/your_user_name/employees_data" directory ?

Creating Directory in the HDFS :

```
hdfs dfs -mkdir /user/satish.patil678_gmail/employees_data
```

Coping data from local linux system to newly created HDFS Directory :

```
$ hdfs dfs -put Consultentdata_-943_Records.txt
/user/satish.patil678_gmail/employees_data
```

```
$ hdfs dfs -ls /user/satish.patil678_gmail/employees_data
```

Found 1 items

-rw-r--r-- 3 satish.patil678_gmail hadoop 23571 2018-06-17 09:19
/user/satish.patil678_gmail/employees_data/Consultentdata_-943_Records.txt

2. Create an external Hive table "employees_Table" representing this "employees_data". This table will have 5 fields id,age,gender,role and salary. ?

Creating Database and using :

```
create database miniproject_Bigdata_Hadoop;
```

```
use miniproject_Bigdata_Hadoop;
```

Creating external table representing employees data :

```
hive> create external table if not exists employees_Table (id int,age int,gender CHAR(10),role
string,salary int)comment 'Data about employees' row format delimited fields terminated by '|' stored
as textfile location '/user/satish.patil678_gmail/employees_data';
```

```
hive> select * from employees_Table;
```

Time taken: 0.263 seconds, Fetched: 943 row(s)

4. create a new bucketed table "Consultant_Table_Bucket" having 4 buckets on the field salary. This table should store the data into columnar format ORC ?

   Creating ORC file table bucketed on salary column :

```
hive> create table Consultant_Table_Bucket (id int,age int,gender
char(10),role string,salary int) clustered by (salary) into 4
buckets stored as orcfile;
```

5. Insert all those employees whose salary is greater than 5000 into bucketed table "Consultant_Table_Bucket" from "employees_Table" table. While inserting into "Consultant_Table_Bucket" table you need to convert "consultant" role into "BigData Consultant" role.?

   Adding data into bucketed table from earlier created external hive table and applying condition on salary column and making some data transformation on Role field and 4 output files are created based on hashing on salary fields.

```
hive> insert into consultant_Table_Bucket select
id,age,gender,if(role='consultant','BigData Consultant',role)as
role,salary from employees_table where salary>5000;
```

```
hive> select * from Consultant_Table_Bucket;
```

Time taken: 0.102 seconds, Fetched: 871 row(s)

6. Write a Hive query to find out Max, min salary of "BigData Consultant" from the "Consultant_Table_Bucket" table ?

```
hive> select max(salary),min(salary) from consultant_Table_Bucket
where role='BigData Consultant';
```

95403 8052

OR

using group by, If we do partition on Role filed query optimization will be good.

```
hive> select role,max(salary) as Highest,min(salary) as Lowest from
Consultant_Table_Bucket group by role having role='BigData
Consultant';
```

BigData Consultant 95403 8052

Case : Replacing Non-Integer(Null) Salary values with average salary of that role. And making use of partition on role for performance of a query:

4) Creating Partition table on role and bucketing on salary as orcfileformat:

```
create table consultant_rolePartation_SalaryBucket(id int,age
int,gender char(10),salary float)partitioned by(role
string)clustered by (salary) into 4 buckets stored as orcfile;
```

5) Inserting values from external hive table :employees_Table into the newly created partitioned bucketed table. While Inserting converting null values on salary field to average salary of that particular role. And converting the role consultant to BigData Consultant.

```
insert into consultant_rolePartation_SalaryBucket partition(role)
select id,age,gender,salary,role from (select
id,age,gender,if(e.role='consultant','BigData Consultant',e.role) as
role,nvl(e.salary,r) as salary from employees_Table as e , (select
role,avg(salary) as r from employees_Table group by role) as t where
t.role=e.role) as outer where outer.salary>5000;
```

```
hive> select * from consultant_rolePartation_SalaryBucket;
```

Time taken: 0.982 seconds, Fetched: 889 row(s)

6) The below query is efficient now due to partition on role column,

```
select max(salary) as Max_Salary,min(salary) as Min_Salary from
consultant_rolePartation_SalaryBucket where role='BigData
Consultant';
```

95403.0 8052.0

Time taken: 8.934 seconds, Fetched: 1 row(s)

Or

```
select role,max(salary) as Max_Salary,min(salary) as Min_Salary
from consultant_rolePartation_SalaryBucket group by role having
role='BigData Consultant';
```

BigData Consultant 95403.0 8052.0