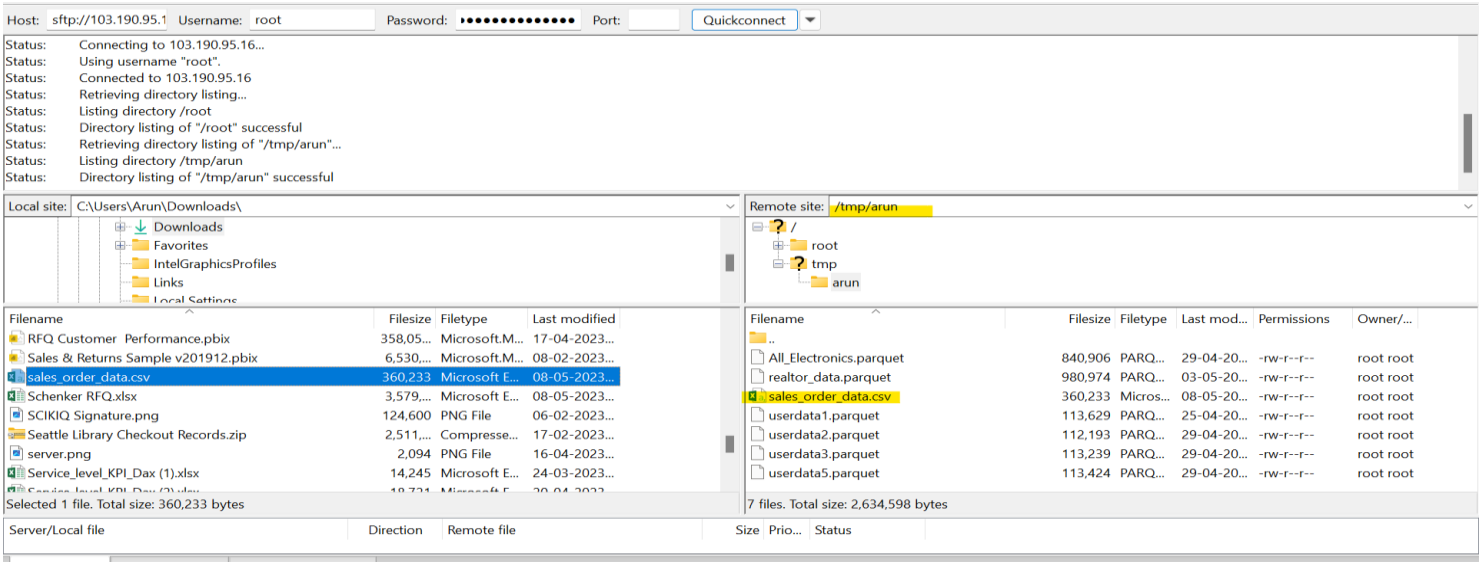
Q1. Download vehicle sales data -> https://github.com/shashank-mishra219/Hive-Class/blob/main/sales_order_data.csv

Q2. Store raw data into hdfs location

For storing the raw file from local to HDFS location we can use **FileZilla** as shown below



Q3. Create a internal hive table "sales_order_csv" which will store csv data sales_order_csv .. make sure to skip header row while creating table

first I have create database assignment

```
0: jdbc:hive2://103.190.95.12:10000> create database assignment1;
INFO  : Compiling command(queryId=hive_20230509231404_22b82e9d-a26a-4482-ad74-84218b24010b): create database assignment1
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hive_20230509231404_22b82e9d-a26a-4482-ad74-84218b24010b); Time taken: 0.005 secon
ds
INFO  : Executing command(queryId=hive_20230509231404_22b82e9d-a26a-4482-ad74-84218b24010b): create database assignment1
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20230509231404_22b82e9d-a26a-4482-ad74-84218b24010b); Time taken: 0.042 secon
ds
INFO  : OK
```

2nd I have create table using below command.

```
0: jdbc:hive2://103.190.95.12:10000> create table sales_order_csv
. . . . . .> ( ORDERNUMBER int, QUANTITYORDERED int, PRICEEACH float, ORDERLINENUMBER int, SALES float, STATUS string, QTR_ID int,
. . . . . .> MONTH_ID int, YEAR_ID int, PRODUCTLINE string, MSRP int, PRODUCTCODE string, PHONE string, CITY string, STATE string,
. . . . . .> POSTALCODE string, COUNTRY string, TERRITORY string, CONTACTLASTNAME string, CONTACTFIRSTNAME string, DEALSIZE string )
. . . . . .> row format delimited
. . . . . .> fields terminated by ','
. . . . . .> tblproperties("skip.header.line.count"="1");
```

Q4. Load data from hdfs path into "sales_order_csv"

load data inpath **'/tmp/arun/sales_order_data.csv'** into table **sales_order_csv**;

Q5. Create an internal hive table which will store data in ORC format "sales_order_orc"

Below are the queries to create and load the data into "sales_order_orc" table

```
No rows affected (0.348 seconds)
0: jdbc:hive2://103.190.95.12:10000> create table sales_order_orc
. . . . . .> ( ORDERNUMBER int, QUANTITYORDERED int, PRICEEACH float, ORDERLINENUMBER int, SALES float, STATUS string, QTR_ID int,
. . . . . .> MONTH_ID int, YEAR_ID int, PRODUCTLINE string, MSRP int, PRODUCTCODE string, PHONE string, CITY string, STATE string,
. . . . . .> POSTALCODE string, COUNTRY string, TERRITORY string, CONTACTLASTNAME string, CONTACTFIRSTNAME string,DEALSIZE string )
. . . . . .> stored as orc;
INFO  : Compiling command(queryId=hive_20230509232817_6d8cad7b-f3e9-4c51-8417-f6751339b8eb): create table sales_order_orc
```

```
0: jdbc:hive2://103.190.95.20:10000> from sales_order_csv insert overwrite table sales_order_orc select *;
INFO  : Compiling command(queryId=hive_20230510001652_5cc4e153-e03a-4152-bf9c-7b6d62d158fa): from sales_order_csv insert overwrite table s
ales_order_orc select *
```

Q a. Calculatye total sales per year

```
select YEAR_ID, sum(SALES) as Total_sales from sales_order_orc
group by YEAR_ID
order by YEAR_ID;
```

```
+----------+---------------------+
| year_id  |     total_sales     |
+----------+---------------------+
| 2003     | 3516979.547241211   |
| 2004     | 4724162.593383789   |
| 2005     | 1791486.7086791992  |
+----------+---------------------+
3 rows selected (5.959 seconds)
0: jdbc:hive2://103.190.95.20:10000>
```

Q b. Find a product for which maximum orders were placed

```
select PRODUCTLINE, sum(QUANTITYORDERED) as Max_order from sales_order_orc
group by PRODUCTLINE
order by sum(QUANTITYORDERED) desc;
```

```
+-------------------+------------+
|    productline    | max_order  |
+-------------------+------------+
| Classic Cars      | 33992      |
| Vintage Cars      | 21069      |
| Motorcycles       | 11663      |
| Trucks and Buses  | 10777      |
| Planes            | 10727      |
| Ships             | 8127       |
| Trains            | 2712       |
+-------------------+------------+
7 rows selected (10.308 seconds)
0: jdbc:hive2://103.190.95.20:10000>
```

Classic Cars the maximum numbers of order

Q c. Calculate the total sales for each quarter

```
select QTR_ID, sum(SALES) as QTR_Sales from sales_order_orc
group by QTR_ID
order by QTR_ID;
```

```
+---------+---------------------+
| qtr_id  |      qtr_sales      |
+---------+---------------------+
| 1       | 2350817.726501465   |
| 2       | 2048120.3029174805  |
| 3       | 1758910.808959961   |
| 4       | 3874780.010925293   |
+---------+---------------------+
4 rows selected (10.348 seconds)
0: jdbc:hive2://103.190.95.20:10000>
```

Q d. In which quarter sales was minimum

```
select QTR_ID, sum(SALES) as QTR_Sales from sales_order_orc
group by QTR_ID
order by sum(SALES) asc;
```

```
+---------+---------------------+
| qtr_id  |      qtr_sales      |
+---------+---------------------+
| 3       | 1758910.808959961   |
| 2       | 2048120.3029174805  |
| 1       | 2350817.726501465   |
| 4       | 3874780.010925293   |
+---------+---------------------+
4 rows selected (10.406 seconds)
```

QTR 3 has minimum sales

Q e. In which country sales was maximum and in which country sales was minimum

```
select COUNTRY, sum(SALES) as Sales from sales_order_orc
group by COUNTRY
order by sum(SALES) desc;
```

country	sales
USA	3627982.825744629
Spain	1215686.9223632812
France	1110916.5217895508
Australia	630623.0987548828
UK	478880.45892333984
Italy	374674.3109741211
Finland	329581.91033935547
Norway	307463.69970703125
Singapore	288488.4102783203
Denmark	245637.15063476562
Canada	224078.55993652344
Germany	220472.0897216797
Sweden	210014.21020507812
Austria	202062.53033447266
Japan	188167.81060791016
Switzerland	117713.55859375
Belgium	108412.61962890625
Philippines	94015.73046875
Ireland	57756.43029785156

19 rows selected (10.491 seconds)
0: jdbc:hive2://103.190.95.20:10000>

U.S.A has maximum sales

Ireland has minimum sales

Q f. Calculate quartelry sales for each city

```
select CITY,QTR_ID, sum(SALES) as Sales from sales_order_orc
group by CITY, QTR_ID
order by CITY,QTR_ID;
```

city	qtr_id	sales
Aaarhus	4	100595.5498046875
Allentown	2	6166.7998046875
Allentown	3	71930.61041259766
Allentown	4	44040.729736328125
Barcelona	2	4219.2001953125
Barcelona	4	74192.66003417969
Bergamo	1	56181.320068359375
Bergamo	4	81774.40008544922
Bergen	3	16363.099975585938
Bergen	4	95277.17993164062
Boras	1	31606.72021484375
Boras	3	53941.68981933594
Boras	4	48710.92053222656
Boston	2	74994.240234375
Boston	3	15344.640014648438
Boston	4	63730.7802734375
Brickhaven	1	31474.7802734375
Brickhaven	2	7277.35009765625
Brickhaven	3	114974.53967285156
Brickhaven	4	11528.52978515625

Q h. Find a month for each year in which maximum number of quantities were sold

```
select YEAR_ID as year,QTR_ID, sum(QUANTITYORDERED) as Sales from sales_order_orc
group by YEAR_ID,QTR_ID
order by YEAR_ID,sum(QUANTITYORDERED) desc ;
```

year	qtr_id	quantityordered
2003	4	18183
2003	3	6209
2003	2	5659
2003	1	4561
2004	4	19965
2004	3	10909
2004	1	8284
2004	2	7666
2005	1	10640
2005	2	6991

If we sort on the bases of Year the first-row item of each year has the highest quantity ordered w.r.t QTR