MySQL Code

1. Create table with column name

```
Create table realestate.property_tbl
(
SlNo      int,
Property_Name            varchar(255),
Property_Type            varchar(255),
Address           varchar(255),
Landmarks         varchar(255),
City      varchar(255),
State_code        varchar(255),
Country           varchar(255),
Carpet_area       int,
Builtup_Area      int,
Floor     int,
Beds      int,
Bathroom          int,
Balcony           int,
BHK_Type          int,
Status    varchar(255),
Furnished_status          varchar(255),
Price_per_square_feet     int,
Year_Built        int,
Transaction_type          varchar(255),
Facing    varchar(255),
Car_Parking       varchar(255),
Type_of_ownership         varchar(255),
Booking_amount            int,
Buy_total_price           int
)
```

2. Load data from text file

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL 8.0/Uploads/magicbricks.txt'
INTO TABLE realestate.property_tbl
FIELDS TERMINATED BY '\t'
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;
```

3. Create table1 table

```
Create table Table1 as
SELECT SLNo, BHK_Type, Property_Name, Property_Type, Beds, Bathroom, Balcony,
Address, City, State_code, Country, Landmarks
 FROM realestate.property_tbl;
```

4. Create table2 table

```
Create table Table2 as
SELECT SLNo, Carpet_area, Status, Floor, Transaction_type, Year_Built,
Price_per_square_feet
 FROM realestate.property_tbl;
```

5. Create table3 table

```
Create table Table3 as
SELECT SLNo, Furnished_status, Facing, Car_Parking, Type_of_ownership,
Booking_amount, Buy_total_price
 FROM realestate.property_tbl;
```

6. SQL Code for Table1
    a. Retrieve properties with balconies, sorted by the number of bedrooms in descending order.
    SELECT * FROM realestate.property_tbl WHERE Balcony > 0 ORDER BY Beds DESC;
    b. Find the top 5 cities with the highest average number of bedrooms per property.
    SELECT City, AVG(Beds) AS Avg_Beds FROM realestate.property_tbl GROUP BY City ORDER BY Avg_Beds DESC LIMIT 5;
    c. Count the number of properties in each city.
    SELECT City, COUNT(*) AS Property_Count FROM realestate.property_tbl GROUP BY City;
    d. Retrieve all properties with at least 3 bedrooms and 2 bathrooms.
    SELECT * FROM realestate.property_tbl WHERE Beds >= 3 AND Bathroom >= 2;
    e. Find properties in a specific state with a certain landmark. (take state and landmark on your own).
    SELECT * FROM realestate.property_tbl WHERE Landmarks is Not Null;
7. SQL Code for Table2
    a. Calculate the average price per square foot for properties built before 2010.
    SELECT AVG(Price_per_square_feet) AS Avg_Price_Per_SqFt FROM table2 WHERE Year_Built < 2010;
    b. Find the total number of properties on each floor.
    SELECT Floor, COUNT(*) AS Total_Properties FROM table2 GROUP BY Floor;
    c. Retrieve properties with a carpet area greater than 1000 square feet and a status of 'Under Construction'.
    SELECT * FROM table2 WHERE Carpet_area > 1000 AND Status = 'Under Construction';
    d. Calculate the average price per square foot for each transaction type.
    SELECT Transaction_type, AVG(Price_per_square_feet) AS Avg_Price_Per_SqFt FROM table2 GROUP BY Transaction_type;
    e. Find the properties with the highest price per square foot, sorted in descending order.
    SELECT * FROM table2 ORDER BY Price_per_square_feet DESC;

8. SQL Code for Table3
    a. Retrieve all properties with a furnished status of 'Fully Furnished' and a facing direction of 'East'.
    SELECT * FROM table3 WHERE Furnished_status = 'Furnished' AND Facing = 'East';
    b. Calculate the average booking amount for properties with and without car parking:
    SELECT Car_Parking, AVG(Booking_amount) AS Avg_Booking_Amount FROM table3 GROUP BY Car_Parking ORDER BY Car_Parking;
    c. Find the total price of properties with different types of ownership.
    SELECT Type_of_ownership, SUM(Buy_total_price) AS Total_Price FROM table3 GROUP BY Type_of_ownership;
    d. Retrieve properties with a booking amount greater than 50000 and a furnished status of 'Semi Furnished'.

SELECT * FROM table3 WHERE Booking_amount > 50000 AND Furnished_status = 'Semi-Furnished';

e. Find the property with the highest booking amount.
SELECT * FROM table3 ORDER BY Booking_amount DESC LIMIT 1;

9. Join SQL Queries using all 3 tables
    a. Retrieve properties from table1 that have a higher price per square foot than the average price per square foot in table2.
    SELECT t1.*
    FROM table1 t1
    JOIN table2 t2 ON t1.SLNo = t2.SLNo
    WHERE t2.Price_per_square_feet > (SELECT AVG(Price_per_square_feet) FROM table2);

    b. Find the properties in table1 that are located in cities where the average price per square foot in table2 is higher than the overall average price per square foot.
    SELECT t1.City
    FROM table1 t1
    JOIN table2 t2 ON t1.SLNo = t2.SLNo
    GROUP BY t1.City
    HAVING AVG(t2.Price_per_square_feet) > (SELECT AVG(Price_per_square_feet) FROM table2);

    c. Retrieve properties from table1 with a certain landmark that have a lower price per square foot than the average price per square foot for properties with the same landmark in table2. (Choose landmark on our own)
    SELECT t1.*
    FROM table1 t1
    JOIN table2 t2 ON t1.SLNo = t2.SLNo
    WHERE t1.Landmarks is not null AND t2.Price_per_square_feet <
    (SELECT AVG(Price_per_square_feet) FROM table2 WHERE Landmarks is not null);

    d. Retrieve properties from table2 with a price per square foot higher than the average booking amount in table3
    SELECT t2.*
    FROM table2 t2
    WHERE t2.Price_per_square_feet > (SELECT AVG(Booking_amount) FROM table3);

    e. Count the number of properties in table2 with more bedrooms than the maximum number of bedrooms in table3.
    SELECT COUNT(*)
    FROM table2 t2
    WHERE t2.SLNo IN
    (SELECT t1.SLNo FROM table1 t1 WHERE t1.Beds >
    (SELECT MAX(Beds) FROM table3));

    f. Find the cities where the average booking amount in table3 is higher than the overall average booking amount, and retrieve properties from table1 located in those cities.
    SELECT t1.City
    FROM table1 t1
    JOIN table3 t3 ON t1.SLNo = t3.SLNo

```
GROUP BY t1.City
HAVING AVG(t3.Booking_amount) >
(SELECT AVG(Booking_amount) FROM table3);
```

g. Retrieve properties from table1 with a furnished status of 'Unfurnished' and a facing direction that does not exist in table3.

```
SELECT t1.*
FROM table1 t1
JOIN table3 t3 ON t1.SLNo = t3.SLNo
WHERE t3.Furnished_status = 'Unfurnished'
AND t3.Facing = '';
```