Objective and Analysis

Objective:

The objective of this project is to conduct an in-depth analysis of the car dataset to understand various aspects of the car market. The analysis includes exploring data through basic SQL queries, addressing real-life questions relevant to the dataset, and performing correlation analysis to understand the relationships between different numerical factors such as selling price, mileage, engine capacity, and year.

Analysis Approach:

1. **Basic Data Exploration:**
   - Retrieved initial rows from the dataset to understand its structure.
   - Created indexes on `selling_price`, `year`, and `fuel` to optimize query performance.
2. **Descriptive Analysis:**
   - Calculated average selling prices for different car names.
   - Identified cars sold in specific years.
   - Filtered cars based on mileage and selling price ranges.
   - Sorted cars by selling price and mileage to determine trends.
3. **Aggregation and Grouping:**
   - Counted the number of cars sold by each seller type.
   - Computed average selling prices grouped by fuel type.
   - Used the `HAVING` clause to find fuel types with high average selling prices.
4. **Subqueries and Window Functions:**
   - Used subqueries to compare individual selling prices with the average selling price.
   - Applied `LEAD` and `LAG` functions to analyze price trends.
   - Ranked cars based on selling price and mileage.
5. **Year-over-Year Analysis:**
   - Compared the average selling price and total sales for each year to identify trends and patterns over time.
6. **Correlation Analysis:**
   - Computed the Pearson correlation coefficients between numerical factors (`selling_price`, `mileage`, `engine`, and `year`) to understand their relationships

Queries :

```sql
USE cars;

-- Display first 2 rows from car_data table
SELECT * FROM car_data
LIMIT 2;
```

Name	year	selling_price	km_driven	fuel	seller_type	transmission	owner	mileage	engine	max_power	torque	seats
Maruti Alto 800 LXI Opt	2023	410000	10000	Petrol	Individual	Manual	First Owner	19.03	999	71.01	96Nm	5
Skoda Slavia 1.0 TSI Ambition	2023	1350000	10000	Petrol	Individual	Manual	First Owner	14.08	1956	167.67	350nm	5

```sql
-- Indexing:
```

```sql
-- Create an index on the selling_price column to speed up queries
involving price ranges.
CREATE INDEX I_selling_price ON car_data(selling_price);

-- Create a composite index on year and fuel to optimize queries filtering
by these columns.
CREATE INDEX IC_year_fuel ON car_data(year, fuel);

-- Basic Queries:
-- 1. Select Clause:
-- Retrieve the names and average selling prices of all cars
SELECT Name, FLOOR(AVG(selling_price)) AS avg_selling_price
FROM car_data
GROUP BY Name;
```

Name	avg_selling_price
Maruti Alto 800 LXI Opt	384000
Skoda Slavia 1.0 TSI Ambition	1350000
BMW 3 Series Gran Limousine 320Ld Luxury Line	5800000
MG ZS EV Exclusive	2650000
Tata Punch Adventure	715000
Maruti S-Presso VXi Plus	450000

```sql
-- List all cars that were sold in the year 2023.
SELECT Name
FROM car_data
WHERE year = 2023;
```

Name
BMW 3 Series Gran Limousine 320Ld Luxury Line
MG ZS EV Exclusive
Maruti Alto 800 LXI Opt
Skoda Slavia 1.0 TSI Ambition
Tata Punch Adventure
Maruti S-Presso VXi Plus

```sql
-- 2. Where Clause:
-- Find all cars with a mileage greater than 15 kmpl.
SELECT Name, mileage
FROM car_data
WHERE mileage > 15;
```

Name	mileage
Maruti Alto 800 LXI Opt	19.03
BMW 3 Series Gran Limousine 320Ld Luxury Line	18.15
MG ZS EV Exclusive	32.52
Maruti S-Presso VXi Plus	19.03
Maruti S-Presso LXi	19.47
Renault KWID CLIMBER	18.15

```sql
-- Retrieve cars where the selling price is between 400,000 and 1,500,000
SELECT Name
FROM car_data
WHERE selling_price BETWEEN 400000 AND 1500000;
```

	Name
▶	Maruti Alto 800 LXI Opt
	Skoda Slavia 1.0 TSI Ambition
	Tata Punch Adventure
	Maruti S-Presso VXi Plus
	Maruti S-Presso LXi
	Renault Kiger RXT AMT Opt DT

```sql
-- 3. Order By Clause:
-- List all cars sorted by their selling price in descending order.
SELECT Name, selling_price
FROM car_data
ORDER BY selling_price DESC;
```

	Name	selling_price
▶	Volvo XC90 T8 Excellence BSIV	10000000
	BMW X7 xDrive 30d DPE	7200000
	Audi A6 35 TFSI Matrix	6523000
	Audi A6 35 TFSI Matrix	6223000
	BMW 6 Series GT 630d Luxury Line	6000000

```sql
-- Sort cars by their mileage in ascending order.
SELECT Name, mileage
FROM car_data
ORDER BY mileage ASC;
```

	Name	mileage
▶	Hyundai Santro Xing (Non-AC)	0
	Hyundai Santro Xing GL	0
	Land Rover Freelander 2 TD4 HSE	0
	Mahindra Bolero Pik-Up CBC 1.7T	0
	Mahindra Bolero Pik-Up FB 1.7T	0
	Mercedes-Benz GLC 220d 4MATIC	0

```sql
-- Aggregation and Grouping:
-- 4. Group By Clause:
-- Count the number of cars sold by each seller type.
SELECT seller_type, COUNT(*) AS number_of_cars
FROM car_data
GROUP BY seller_type;
```

	seller_type	number_of_cars
▶	Individual	6577
	Dealer	1113
	Trustmark Dealer	236

```sql
-- Find the average selling price of cars grouped by fuel type
SELECT fuel, AVG(selling_price) AS avg_selling_price
FROM car_data
GROUP BY fuel;
```

fuel	avg_selling_price
Petrol	474468.1706
Diesel	804352.4082
Electric	2650000.0000
CNG	317903.7885
LPG	210885.7143

```
-- 5. Having Clause:
-- Find fuel types with an average selling price greater than 1,000,000
SELECT fuel, AVG(selling_price) AS avg_selling_price
FROM car_data
GROUP BY fuel
HAVING AVG(selling_price) > 1000000;
```

fuel	avg_selling_price
Electric	2650000.0000

```
-- Subqueries:
-- 10. Subquery in Select Clause:
-- Retrieve the selling price of each car along with the average selling
price of all cars.
SELECT selling_price,
       (SELECT AVG(selling_price) FROM car_data) AS avg_price
FROM car_data;
```

selling_price	avg_price
29999	651686.2575
30000	651686.2575
31000	651686.2575
31504	651686.2575
33351	651686.2575
35000	651686.2575

```
-- 11. Subquery in Where Clause:
-- Find all cars that have a selling price greater than the average selling
price.
SELECT Name, selling_price
FROM car_data
WHERE selling_price > (SELECT AVG(selling_price) FROM car_data);
```

Name	selling_price
Maruti Swift AMT VXI	654000
Maruti Swift Dzire VXI 1.2	655000
Hyundai i20 1.2 Spotz	655000
Maruti Swift AMT ZXI	655000
Mahindra TUV 300 T8	655000
Maruti Swift Dzire ZDI Plus	655000

```
-- Window Functions:
-- 12. Lead and Lag:
-- Use LEAD to show the selling price of the next more expensive car.
SELECT Name, selling_price,
       LEAD(selling_price) OVER(ORDER BY selling_price ASC) AS
next_expensive_car
```

```
FROM car_data
ORDER BY selling_price;
```

Name	selling_price	next_expensive_car
Maruti 800 AC	29999	30000
Maruti Zen LXI	30000	31000
Maruti 800 Std	31000	31504
Maruti 800 Std	31504	33351
Maruti Wagon R VXI	33351	35000
Maruti 800 AC	35000	35000

```
-- Use LAG to display the selling price of the previous less expensive car.
SELECT Name, selling_price,
       LAG(selling_price) OVER(ORDER BY selling_price ASC) AS
less_expensive_car
FROM car_data
ORDER BY selling_price;
```

Name	selling_price	less_expensive_car
Maruti 800 AC	29999	NULL
Maruti Zen LXI	30000	29999
Maruti 800 Std	31000	30000
Maruti 800 Std	31504	31000
Maruti Wagon R VXI	33351	31504
Maruti 800 AC	35000	33351

```
-- 13. Rank and Dense Rank:
-- Assign ranks to cars based on their selling price
SELECT Name, selling_price,
       RANK() OVER(ORDER BY selling_price ASC) AS rank_price,
       DENSE_RANK() OVER(ORDER BY selling_price ASC) AS dense_rank_price
FROM car_data
ORDER BY selling_price;
```

Name	selling_price	rank_price	dense_rank_price
Maruti 800 AC	29999	1	1
Maruti Zen LXI	30000	2	2
Maruti 800 Std	31000	3	3
Maruti 800 Std	31504	4	4
Maruti Wagon R VXI	33351	5	5
Maruti 800 AC	35000	6	6
Maruti 800 Std BSII	35000	6	6
Maruti 800 AC	35000	6	6
Maruti 800 Std	39000	9	7
Maruti 800 AC	40000	10	8
Maruti 800 Std	40000	10	8
Maruti 800 Std	40000	10	8
Maruti 800 Std	40000	10	8
Maruti 800 Std BSIII	40000	10	8
Maruti 800 AC	40000	10	8

```
-- Assign dense ranks to cars based on their mileage.
SELECT Name, mileage,
       DENSE_RANK() OVER(ORDER BY mileage ASC) AS dense_rank_mileage
```

```
FROM car_data
ORDER BY mileage ASC;
```

Name	mileage	dense_rank_mileage
Hyundai Santro Xing GL	0	1
Hyundai Santro Xing GL	0	1
Mahindra Bolero Pik-Up FB 1.7T	0	1
Mahindra Bolero Pik-Up CBC 1.7T	0	1
Mercedes-Benz GLC 220d 4MATIC	0	1
Maruti Omni LPG CARGO BSIII W IMM...	5.3737	2
Maruti Omni LPG CARGO BSIII W IMM...	5.3737	2
Chevrolet Spark 1.0 LT LPG	6.5076	3
Hyundai Accent Executive LPG	6.5076	3
Hyundai Santro Xing GLS LPG	6.63085	4
Hyundai Santro Xing GL Plus LPG	6.63085	4
Hyundai Santro Xing GLS LPG	6.63085	4

```
-- Real-Life Scenario:
-- Suppose you are analyzing sales data to understand trends in the car
market. Write a query to find the top 3 most sold car
-- models for each year. Use window functions to rank the models within
each year and display their names, selling prices, and total sales.
WITH Ranked_cars AS (
    SELECT Name, selling_price, fuel, year, COUNT(*) AS total,
           ROW_NUMBER() OVER (PARTITION BY year ORDER BY COUNT(*) DESC) AS
Rankings
    FROM car_data
    GROUP BY Name, selling_price, year, fuel
)
SELECT Name, selling_price, fuel, year, total, Rankings
FROM Ranked_cars
WHERE Rankings <= 3;
```

Name	selling_price	fuel	year	total	Rankings
Maruti Baleno Alpha 1.3	740000	Diesel	2018	29	1
Maruti Swift AMT ZXI	600000	Petrol	2018	29	2
Volvo V40 D3 R-Design	2475000	Diesel	2018	29	3
Lexus ES 300h	5150000	Petrol	2019	34	1
Honda Amaze V CVT Petrol BSIV	779000	Petrol	2019	31	2
BMW X4 M Sport X xDrive20d	5400000	Diesel	2019	30	3
Tata Zest Revotron 1.2 XT	500000	Petrol	2020	4	1
Honda Civic ZX Diesel BSIV	2125000	Diesel	2020	2	2
Hyundai Xcent 1.2 VTVT SX	524000	Petrol	2020	2	3
Hyundai Creta SX Diesel AT	1950000	Diesel	2021	1	1
Renault KWID CLIMBER	567000	Petrol	2021	1	2
Nissan Magnite XV Premium	850000	Petrol	2021	1	3
Maruti S-Presso LXi	425000	Petrol	2022	1	1
Hyundai Creta SX Turbo	1895000	Petrol	2022	1	2
Renault Kiger RXT AMT Opt DT	842000	Petrol	2022	1	3
Maruti Alto 800 LXI Opt	410000	Petrol	2023	1	1
Skoda Slavia 1.0 TSI Ambition	1350000	Petrol	2023	1	2
BMW 3 Series Gran Limousine 320L...	5800000	Diesel	2023	1	3

```
-- Query to identify the type of fuel (Petrol, Diesel, Electric) with the
highest average selling price.
SELECT fuel, AVG(selling_price) AS avg_selling_price
```

```sql
FROM car_data
GROUP BY fuel
ORDER BY avg_selling_price DESC
LIMIT 1;
```

fuel	avg_selling_price
Electric	2650000.0000

```sql
-- Query to count the total number of first-time owners who have sold their cars.
SELECT COUNT(owner) AS total_first_owners
FROM car_data
WHERE owner = 'First Owner';
```

total_first_owners
5232

```sql
-- Query to calculate the average mileage of all cars with a transmission type of "Manual".
SELECT ROUND(AVG(mileage), 2) AS avg_mileage
FROM car_data
WHERE transmission = 'Manual';
```

avg_mileage
19.6

```sql
-- Query to identify the car with the highest maximum power.
SELECT Name, MAX(max_power) AS max_power
FROM car_data
GROUP BY Name
ORDER BY max_power DESC
LIMIT 1;
```

Name	max_power
Volvo XC90 T8 Excellence BSIV	400

```sql
-- Query to find the minimum and maximum selling price of all cars in the dataset.
SELECT Name,
       ROUND(MIN(selling_price), 2) AS minimum_selling_price,
       ROUND(MAX(selling_price), 2) AS maximum_selling_price
FROM car_data
GROUP BY Name;
```

Name	minimum_selling_price	maximum_selling_price
Maruti Alto 800 LXI Opt	358000	410000
Skoda Slavia 1.0 TSI Ambition	1350000	1350000
BMW 3 Series Gran Limousine 320Ld Luxury Line	5800000	5800000
MG ZS EV Exclusive	2650000	2650000
Tata Punch Adventure	715000	715000
Maruti S-Presso VXi Plus	450000	450000
Maruti S-Presso LXi	425000	425000
Hyundai Creta SX Turbo	1895000	1895000
Renault Kiger RXT AMT Opt DT	842000	842000
Renault KWID CLIMBER	567000	567000
Mahindra XUV300 W8 Diesel Sunroof	1197000	1197000
Mahindra XUV700 AX5 Diesel AT	2275000	2275000
Renault Triber RXT	800000	800000
Hyundai Creta SX Diesel AT	1950000	1950000
Nissan Magnite XV Premium	850000	850000

```sql
-- Query to find the number of seats in the car with the highest selling price.
SELECT Name, seats
FROM car_data
WHERE selling_price = (SELECT MAX(selling_price) FROM car_data);
```

Name	seats
Volvo XC90 T8 Excellence BSIV	4

```sql
-- Query to identify all electric cars in the dataset and their respective selling prices.
SELECT Name, selling_price
FROM car_data
WHERE fuel = 'Electric';
```

Name	selling_price
MG ZS EV Exclusive	2650000

```sql
-- Query to calculate the average engine capacity of all cars sold by dealers.
SELECT AVG(engine) AS avg_engine_capacity
FROM car_data
WHERE seller_type = 'Dealer';
```

avg_engine_capacity
1610.5400

```sql
-- What were the most common transmission types in the dataset?
SELECT transmission, COUNT(*) AS count
FROM car_data
GROUP BY transmission
ORDER BY count DESC
LIMIT 1;
```

transmission	count
Manual	6865

```sql
-- How did the average selling price vary across different fuel types?
SELECT fuel, AVG(selling_price) AS avg_selling_price
FROM car_data
GROUP BY fuel;
```

fuel	avg_selling_price
Petrol	474468.1706
Diesel	804352.4082
Electric	2650000.0000
CNG	317903.7885
LPG	210885.7143

```sql
-- How can you use a window function to find the running total of the selling prices for all cars, grouped by their fuel type?
```

```sql
SELECT fuel, selling_price,
       SUM(selling_price) OVER(PARTITION BY fuel ORDER BY selling_price) AS
running_total
FROM car_data;
```

fuel	selling_price	running_total
CNG	120000	120000
CNG	150000	270000
CNG	155000	580000
CNG	155000	580000
CNG	160000	740000
CNG	165000	905000
CNG	178000	1083000
CNG	185000	1268000
CNG	200000	1668000
CNG	200000	1668000
CNG	209000	1877000
CNG	211000	2088000

```sql
-- Cumulative average of selling prices by transmission type
SELECT transmission, selling_price,
       AVG(selling_price) OVER(PARTITION BY transmission ORDER BY
selling_price) AS cumulative_avg
FROM car_data
ORDER BY transmission;
```

transmission	selling_price	cumulative_avg
Automatic	842000	842000.0000
Automatic	1895000	1368500.0000
Automatic	1950000	1562333.3333
Automatic	2275000	1740500.0000
Automatic	2650000	1922400.0000
Automatic	2975000	2097833.3333
Automatic	5800000	2626714.2857
Manual	358000	358000.0000
Manual	410000	384000.0000
Manual	425000	397666.6667
Manual	450000	410750.0000
Manual	567000	462833.3333
Manual	567000	462833.3333
Manual	715000	498857.1429
Manual	785000	534625.0000

```sql
-- Car with the highest selling price in each category
WITH row_rank AS (
    SELECT *, ROW_NUMBER() OVER (PARTITION BY fuel ORDER BY selling_price
DESC) AS row_num
    FROM car_data
)
SELECT *
FROM row_rank
WHERE row_num = 1;
```

Name	year	selling_price	km_driven	fuel	seller_type	transmission	owner	mileage	engine	max_power	torque	seats
Maruti Ertiga VXI CNG Limited Edition	2014	545000	35000	CNG	Individual	Manual	First Owner	16.2792	1373	80.9	110Nm@ 4000rpm	7
BMW X7 xDrive 30d DPE	2020	7200000	5000	Diesel	Individual	Automatic	First Owner	13.38	2993	265	620Nm@ 1500-2500rpm	7
MG ZS EV Exclusive	2023	2650000	10000	Electric	Dealer	Automatic	First Owner	32.52	998	58.33	78Nm	5
Hyundai i10 Sportz 1.1L LPG	2015	375000	60000	LPG	Individual	Manual	First Owner	9.4656	1086	68.05	99.04Nm@ 4500rpm	5
Volvo XC90 T8 Excellence BSIV	2017	10000000	30000	Petrol	Individual	Automatic	First Owner	42	1969	400	640Nm@ 1740rpm	4

```sql
-- Calculate the Pearson correlation coefficient between all pairs of
numerical columns
WITH stats AS (
    SELECT
        AVG(selling_price) AS avg_price,
        AVG(mileage) AS avg_mileage,
        AVG(engine) AS avg_engine,
        AVG(year) AS avg_year,
        SUM((selling_price - (SELECT AVG(selling_price) FROM car_data)) *
            (mileage - (SELECT AVG(mileage) FROM car_data))) AS
cov_price_mileage,
        SUM((selling_price - (SELECT AVG(selling_price) FROM car_data)) *
            (engine - (SELECT AVG(engine) FROM car_data))) AS
cov_price_engine,
        SUM((selling_price - (SELECT AVG(selling_price) FROM car_data)) *
            (year - (SELECT AVG(year) FROM car_data))) AS cov_price_year,
        SUM((mileage - (SELECT AVG(mileage) FROM car_data)) *
            (engine - (SELECT AVG(engine) FROM car_data))) AS
cov_mileage_engine,
        SUM((mileage - (SELECT AVG(mileage) FROM car_data)) *
            (year - (SELECT AVG(year) FROM car_data))) AS cov_mileage_year,
        SUM((engine - (SELECT AVG(engine) FROM car_data)) *
            (year - (SELECT AVG(year) FROM car_data))) AS cov_engine_year,

        SUM((selling_price - (SELECT AVG(selling_price) FROM car_data)) *
            (selling_price - (SELECT AVG(selling_price) FROM car_data))) AS
var_price,
        SUM((mileage - (SELECT AVG(mileage) FROM car_data)) *
            (mileage - (SELECT AVG(mileage) FROM car_data))) AS
var_mileage,
        SUM((engine - (SELECT AVG(engine) FROM car_data)) *
            (engine - (SELECT AVG(engine) FROM car_data))) AS var_engine,
        SUM((year - (SELECT AVG(year) FROM car_data)) *
            (year - (SELECT AVG(year) FROM car_data))) AS var_year
    FROM car_data
)
SELECT
    cov_price_mileage / SQRT(var_price * var_mileage) AS
correlation_price_mileage,
    cov_price_engine / SQRT(var_price * var_engine) AS
correlation_price_engine,
    cov_price_year / SQRT(var_price * var_year) AS correlation_price_year,
    cov_mileage_engine / SQRT(var_mileage * var_engine) AS
correlation_mileage_engine,
    cov_mileage_year / SQRT(var_mileage * var_year) AS
correlation_mileage_year,
    cov_engine_year / SQRT(var_engine * var_year) AS
correlation_engine_year
FROM stats;
```

correlation_price_mileage	correlation_price_engine	correlation_price_year	correlation_mileage_engine	correlation_mileage_year	correlation_engine_year
-0.11484196234207068	0.4530162037995352	0.413281422452504	-0.5555763582883966	0.3279728614399174	0.017317633749637143

Conclusion:

The analysis provides valuable insights into the car market, highlighting the premium positioning of electric cars, consumer preferences for traditional fuel types, and significant market trends over the years. The correlation analysis further elucidates the relationships between key numerical factors, aiding in a deeper understanding of the dynamics affecting car prices and sales.

- **Selling Price vs. Mileage:** There is a weak negative correlation, suggesting that cars with higher mileage tend to have lower selling prices.
- **Selling Price vs. Engine Capacity:** A moderate positive correlation indicates that cars with larger engines tend to have higher selling prices.
- **Selling Price vs. Year:** The positive correlation suggests that newer cars tend to have higher selling prices.
- **Mileage vs. Engine Capacity:** A moderate negative correlation indicates that cars with larger engines tend to have lower mileage.
- **Mileage vs. Year:** A weak positive correlation suggests that newer cars tend to have higher mileage.
- **Engine Capacity vs. Year:** The very weak positive correlation suggests a negligible relationship between engine capacity and the year of the car.