

# Data Analytics SQL Mini-Project

Insight no. 1 -

- Get the average price at which cars are sales.
- Print the name or models of the cars.
- Sort the result in ascending order.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1 --#1 Get the average price at which cars are sales. Print the name or models of the cars. Sort the result in ascending order.
2 select name, avg(selling_price) as avg
3 from cars24
4 group by 1
5 order by 1;
```

The Results window displays the following data:

| name                                 | avg          |
|--------------------------------------|--------------|
| Ambassador CLASSIC 1500 DSL AC       | 75000.0000   |
| Ambassador Classic 2000 DSZ AC PS    | 99000.0000   |
| Ambassador Grand 1500 DSZ BSHI       | 120000.0000  |
| Ambassador Grand 2000 DSZ PW CL      | 200000.0000  |
| Ashok Leyland Stile LE               | 300000.0000  |
| Audi A3 35 TDI Premium Plus          | 2800000.0000 |
| Audi A3 40 TFSI Premium              | 1689999.0000 |
| Audi A4 1.8 TFSI                     | 750000.0000  |
| Audi A4 2.0 TDI                      | 1550000.0000 |
| Audi A4 2.0 TDI 177 Bhp Premium Plus | 1200000.0000 |
| Audi A4 35 TDI Premium Plus          | 2249749.7500 |
| Audi A6 2.0 TDI                      | 2200000.0000 |
| Audi A6 2.0 TDI Design Edition       | 1689999.0000 |
| Audi A6 2.0 TDI Premium Plus         | 2000000.0000 |
| Audi A6 2.0 TDI Technology           | 1750000.0000 |

The Output window shows the execution progress:

| # | Time     | Action  | Message              | Duration / Fetch      |
|---|----------|---|----------------------|-----------------------|
| 1 | 21:19:47 | use masai   | 0 row(s) affected    | 0.000 sec             |
| 2 | 21:20:08 | select * from cars24 LIMIT 0, 50000   | 8128 row(s) returned | 0.000 sec / 0.015 sec |
| 3 | 21:25:12 | select name, avg(selling_price) as avg from cars24 group by 1 order by 1 LIMIT 0, 50000 | 2038 row(s) returned | 0.032 sec / 0.000 sec |

## Insight no. 2 -

--Print the car model, manufactured year and previous owners.

Sort the output in ascending order.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1  ##2. Print the car model, manufactured year and previous owners. Sort the output in ascending order.
2  *
3  select name, year, owner
4  from cars24
5  group by 1,2,3
6  order by 1;
```

The Results grid displays the following data:

| name                                 | year | owner        |
|--------------------------------------|------|--------------|
| Ambassador CLASSIC 1500 DSL AC       | 2000 | Second Owner |
| Ambassador Classic 2000 DSL AC PS    | 1994 | Second Owner |
| Ambassador Grand 1500 DSL BSHH       | 2008 | Second Owner |
| Ambassador Grand 2000 DSL PW CL      | 2008 | Third Owner  |
| Ashok Leyland Stile LE               | 2013 | Second Owner |
| Audi A3 35 TDI Premium Plus          | 2017 | First Owner  |
| Audi A3 35 TDI Premium Plus          | 2018 | First Owner  |
| Audi A3 40 TFSI Premium              | 2017 | First Owner  |
| Audi A4 1.8 TFSI                     | 2010 | Second Owner |
| Audi A4 2.0 TDI                      | 2014 | First Owner  |
| Audi A4 2.0 TDI                      | 2014 | Second Owner |
| Audi A4 2.0 TDI 177 Bhp Premium Plus | 2013 | First Owner  |
| Audi A4 35 TDI Premium Plus          | 2015 | First Owner  |
| Audi A4 35 TDI Premium Plus          | 2016 | First Owner  |
| Audi A4 35 TDI Premium Plus          | 2018 | First Owner  |

The Output tab shows the execution log:

| # | Time     | Action  | Message              | Duration / Fetch      |
|---|----------|---|----------------------|-----------------------|
| 1 | 21:19:47 | use masal   | 0 row(s) affected    | 0.000 sec             |
| 2 | 21:20:08 | select * from cars24 LIMIT 0, 50000   | 8128 row(s) returned | 0.000 sec / 0.015 sec |
| 3 | 21:25:12 | select name, avg(selling_price) as avg from cars24 group by 1 order by 1 LIMIT 0, 50000 | 2038 row(s) returned | 0.032 sec / 0.000 sec |
| 4 | 21:46:48 | select name, year, owner from cars24 group by 1,2,3 order by 1 LIMIT 0, 50000           | 4714 row(s) returned | 0.031 sec / 0.016 sec |

### Insight no. 3 -

--Identify top 10 car models in the database, which had manufactured in Year '2015'. Sort the output in ascending order.

The screenshot displays the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a tree view of the 'masai' database, including tables like 'cars24', 'category', 'comments', 'customers', 'marks', 'orderdetails', 'orders', 'payments', 'persons', 'products', 'shippers', 'student\_marks', 'supplies', 'users', 'Views', 'Stored Procedures', and 'Functions'. The 'cars24' table is selected. The main editor shows a SQL query in 'SQL File 5\*':

```
1 --#3. Identify top 10 car models in the database, which had manufactured in year '2015'. Sort the output in ascending order.
2 select name
3 from cars24
4 where year = '2015'
5 group by 1
6 order by 1
7 limit 10;
```

Below the query editor, the 'Result Grid' shows the output of the query. The results are sorted by 'name' in ascending order:

| name                              |
|-----------------------------------|
| Audi A4 35 TDI Premium Plus       |
| Audi Q5 45 TDI quattro Technology |
| Audi Q7 35 TDI Quattro Premium    |
| BMW 3 Series 320d Prestige        |
| Chevrolet Beat Diesel LT          |
| Chevrolet Beat LT                 |
| Chevrolet Cruze LTZ AT            |
| Chevrolet Enjoy TCD LS 8 Seater   |
| Chevrolet Sail 1.2 LS             |
| Chevrolet Sail 1.2 LS ABS         |

At the bottom, the 'Output' pane shows the execution log with the following entries:

| # | Time     | Action  | Message              | Duration / Fetch      |
|---|----------|---|----------------------|-----------------------|
| 1 | 21:19:47 | use masai   | 0 row(s) affected    | 0.000 sec             |
| 2 | 21:20:08 | select * from cars24 LIMIT 0, 50000   | 8128 row(s) returned | 0.000 sec / 0.015 sec |
| 3 | 21:25:12 | select name, avg(selling_price) as avg from cars24 group by 1 order by 1 LIMIT 0, 50000 | 2038 row(s) returned | 0.032 sec / 0.000 sec |
| 4 | 21:46:48 | select name, year, owner from cars24 group by 1,2,3 order by 1 LIMIT 0, 50000           | 4714 row(s) returned | 0.031 sec / 0.016 sec |
| 5 | 21:55:17 | select name from cars24 where year = '2015' group by 1 order by 1 limit 10              | 10 row(s) returned   | 0.015 sec / 0.000 sec |

## Insight no. 4 -

--Write a query in latest car models in the database. Which model manufactured in '2017'. Sort the output in ascending order.

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
1 --#4. Write a query in latest car models in the database, which models manufactured in '2017'. Sort the output in ascending order'
2 select name, year
3 from cars24
4 where year = '2017'
5 group by 1,2
6 order by 1 asc;
```

The Results Grid displays the following data:

| name                                 | year |
|--------------------------------------|------|
| Audi A3 35 TDI Premium Plus          | 2017 |
| Audi A3 40 TFSI Premium              | 2017 |
| Audi Q3 2.0 TDI Quattro Premium Plus | 2017 |
| BMW 3 Series 320d                    | 2017 |
| BMW 3 Series 320d GT Luxury Line     | 2017 |
| BMW 3 Series 320d Luxury Line Plus   | 2017 |
| BMW X1 sDrive 20d xLine              | 2017 |
| Chevrolet Beat LT                    | 2017 |
| Chevrolet Enjoy 1.3 TCdL LTZ 7       | 2017 |
| Datsun GO Anniversary Edition        | 2017 |
| Datsun GO Plus Anniversary Edition   | 2017 |
| Datsun GO Plus T                     | 2017 |
| Datsun GO T BSV                      | 2017 |
| Datsun RediGO 1.0 S                  | 2017 |

The Output tab shows the execution details of the query:

| # | Time     | Action   | Message              | Duration / Fetch      |
|---|----------|--|----------------------|-----------------------|
| 2 | 21:20:08 | select * from cars24 LIMIT 0, 50000  | 8128 row(s) returned | 0.000 sec / 0.015 sec |
| 3 | 21:25:12 | select name, avg(selling_price) as avg from cars24 group by 1 order by 1 LIMIT 0, 50000      | 2038 row(s) returned | 0.032 sec / 0.000 sec |
| 4 | 21:46:48 | select name, year, owner from cars24 group by 1,2,3 order by 1 LIMIT 0, 50000                | 4714 row(s) returned | 0.031 sec / 0.016 sec |
| 5 | 21:55:17 | select name from cars24 where year = '2015' group by 1 order by 1 limit 10                   | 10 row(s) returned   | 0.015 sec / 0.000 sec |
| 6 | 22:05:25 | select name, year from cars24 where year = '2017' group by 1,2 order by 1 asc LIMIT 0, 50000 | 420 row(s) returned  | 0.016 sec / 0.000 sec |

## Insight no. 5 –

--Identify 20 car models minimum price whose launch in year '2016'. Sort the output in descending order.

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following query:

```
1 --5. Identify 20 models minimum price whose launch in year '2016'. Sort the output in descending order.
2 select name, min(selling_price) as min_price
3 from cars24
4 where year = '2016'
5 group by 1
6 order by 1
7 limit 20;
```

The Results Grid displays the following data:

| name                                 | year |
|--------------------------------------|------|
| Audi A3 35 TDI Premium Plus          | 2017 |
| Audi A3 40 TFSI Premium              | 2017 |
| Audi Q3 2.0 TDI Quattro Premium Plus | 2017 |
| BMW 3 Series 320d                    | 2017 |
| BMW 3 Series 320d GT Luxury Line     | 2017 |
| BMW 3 Series 320d Luxury Line Plus   | 2017 |
| BMW X1 sDrive 200 xLine              | 2017 |
| Chevrolet Beat LT                    | 2017 |
| Chevrolet Enjoy 1.3 TCD LTZ 7        | 2017 |
| Datsun GO Anniversary Edition        | 2017 |
| Datsun GO Plus Anniversary Edition   | 2017 |
| Datsun GO Plus T                     | 2017 |
| Datsun GO T BSV                      | 2017 |

The Output tab shows the execution log:

| # | Time     | Action   | Message              | Duration / Fetch      |
|---|----------|--|----------------------|-----------------------|
| 2 | 21:20:08 | select * from cars24 LIMIT 0, 50000  | 8128 row(s) returned | 0.000 sec / 0.015 sec |
| 3 | 21:25:12 | select name, avg(selling_price) as avg from cars24 group by 1 order by 1 LIMIT 0, 50000      | 2038 row(s) returned | 0.032 sec / 0.000 sec |
| 4 | 21:46:48 | select name, year, owner from cars24 group by 1,2,3 order by 1 LIMIT 0, 50000                | 4714 row(s) returned | 0.031 sec / 0.016 sec |
| 5 | 21:55:17 | select name from cars24 where year = '2015' group by 1 order by 1 limit 10                   | 10 row(s) returned   | 0.015 sec / 0.000 sec |
| 6 | 22:05:25 | select name, year from cars24 where year = '2017' group by 1,2 order by 1 asc LIMIT 0, 50000 | 420 row(s) returned  | 0.016 sec / 0.000 sec |

## Insight no. 6 –

--Print the name, mileage in the car models and the mileage which have less than '10.1'. Sort the output in ascending order.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
2 select name, mileage
3 from cars24
4 group by 1,2
5 having mileage < '10.2'
6 order by 1;
```

The Results tab displays the output of the query in a table with two columns: name and mileage.

| name  | mileage |
|---|---------|
| Honda Accord V6 AT                                | 9       |
| Honda CR-V 2.4 4WD AT                             | 10.1    |
| Hyundai Santro Xing (Non-AC)                      | 0       |
| Hyundai Santro Xing GL                            | 0       |
| Jeep Wrangler 2016-2019 3.6 4x4                   | 9.5     |
| Land Rover Freelander 2 TD4 HSE                   | 0       |
| Mahindra Bolero Pick-Up C3C 1.7T                  | 0       |
| Mahindra Bolero Pick-Up FB 1.7T                   | 0       |
| Mercedes-Benz E-Class E 220 CDI Avantgarde        | 10      |
| Mercedes-Benz GLC 220d 4MATIC                     | 0       |
| Mercedes-Benz M-Class ML 350 4Matic               | 0       |
| Tata Indica Vista Aura Selfie Anniversary Edit... | 0       |
| Volkswagen Polo GT TSI BSV                        | 0       |

The Output tab shows the execution log with the following entries:

| # | Time     | Action  | Message              | Duration / Fetch      |
|---|----------|---|----------------------|-----------------------|
| 3 | 21:25:12 | select name, avg(selling_price) as avg from cars24 group by 1 order by 1 LIMIT 0, 50000         | 2038 row(s) returned | 0.032 sec / 0.000 sec |
| 4 | 21:46:48 | select name, year, owner from cars24 group by 1,2,3 order by 1 LIMIT 0, 50000                   | 4714 row(s) returned | 0.031 sec / 0.016 sec |
| 5 | 21:55:17 | select name from cars24 where year = '2015' group by 1 order by 1 limit 10                      | 10 row(s) returned   | 0.015 sec / 0.000 sec |
| 6 | 22:05:25 | select name, year from cars24 where year = '2017' group by 1,2 order by 1 asc LIMIT 0, 50000    | 420 row(s) returned  | 0.016 sec / 0.000 sec |
| 7 | 22:16:16 | select name, mileage from cars24 group by 1,2 having mileage < '10.2' order by 1 LIMIT 0, 50000 | 14 row(s) returned   | 0.032 sec / 0.000 sec |

## Insight no. 7 –

--Print all the car models in the database, whose mileage is greater than '20.01' and `engine[cc]` is greater than '1000'. Sort the output in ascending order.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1 --6. Print all the car models in the database whose mileage is greater than '20.01' and engine[cc] is greater than '1000'. Sort the result set in ascending order.
2 select name, mileage, engine [cc]
3 from cars24
4 having engine [cc] > '1000' and mileage > '20'
5 order by 1;
```

The Result Grid displays the following data:

| name                               | mileage | engine [cc] |
|------------------------------------|---------|-------------|
| Ashok Leyland Stile LE             | 20.07   | 1461        |
| Audi A3 35 TDI Premium Plus        | 20.38   | 1968        |
| Audi A3 35 TDI Premium Plus        | 20.38   | 1968        |
| BMW 3 Series 320d                  | 22.69   | 1995        |
| BMW 3 Series 320d Luxury Line      | 22.69   | 1995        |
| BMW 3 Series 320d Luxury Line      | 22.69   | 1995        |
| BMW 3 Series 320d Luxury Line Plus | 22.69   | 1995        |
| BMW 3 Series 320d Luxury Line Plus | 22.69   | 1995        |
| BMW 3 Series GT Luxury Line        | 21.76   | 1995        |
| BMW 5 Series 520d Sport Line       | 22.48   | 1995        |
| BMW 5 Series 520d Sport Line       | 22.48   | 1995        |
| BMW 5 Series 520d Sport Line       | 22.48   | 1995        |
| BMW 5 Series 520d Sport Line       | 22.48   | 1995        |
| BMW 5 Series 520d Sport Line       | 22.48   | 1995        |
| BMW 5 Series 520d Sport Line       | 22.48   | 1995        |

The Output pane shows the execution log:

| # | Time     | Action   | Message              | Duration / Fetch      |
|---|----------|--|----------------------|-----------------------|
| 4 | 21:46:48 | select name, year, owner from cars24 group by 1,2,3 order by 1 LIMIT 0, 50000                          | 4714 row(s) returned | 0.031 sec / 0.016 sec |
| 5 | 21:55:17 | select name from cars24 where year = '2015' group by 1 order by 1 limit 10                             | 10 row(s) returned   | 0.015 sec / 0.000 sec |
| 6 | 22:05:25 | select name, year from cars24 where year = '2017' group by 1,2 order by 1 asc LIMIT 0, 50000           | 420 row(s) returned  | 0.016 sec / 0.000 sec |
| 7 | 22:16:16 | select name, mileage from cars24 group by 1,2 having mileage < '10.2' order by 1 LIMIT 0, 50000        | 14 row(s) returned   | 0.032 sec / 0.000 sec |
| 8 | 22:23:31 | Select name, mileage, engine [cc] from cars24 having engine [cc] > '1000' and mileage > '20' order ... | 2623 row(s) returned | 0.016 sec / 0.015 sec |

## Insight no. 8 –

--Write a query with 10 car models in the database. Print name, year, price .  
Whose manufactured in '2018' and price in which have '560000'. Sort the  
output in ascending order.

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1  ##8. write a query with 10 car models in the database. print name, year, price. whose manufactured in '2018' and price in which have '560000'. Sort the result set :
2  *
3  select name, year, selling_price
4  from cars24
5  where year = '2018'
6  having selling_price = '560000'
7  order by 1
8  limit 10;
```

The Results Grid shows the following data:

| name  | year | selling_price |
|---|------|---------------|
| Honda Amaze S Option i-DETC                 | 2018 | 560000        |
| Maruti Baleno Delta 1.2                     | 2018 | 560000        |
| Maruti Swift VXi                            | 2018 | 560000        |
| Tata Tiago 1.2 Revokron VZ WO Alloy         | 2018 | 560000        |
| Volkswagen Ameo 1.5 TDI Comfortline         | 2018 | 560000        |
| Volkswagen Polo 2015-2019 1.0 MPI Trendline | 2018 | 560000        |
| Volkswagen Polo 2015-2019 1.0 MPI Trendline | 2018 | 560000        |

The Output tab shows the execution log:

| # | Time     | Action   | Message              | Duration / Fetch      |
|---|----------|--|----------------------|-----------------------|
| 5 | 21:55:17 | select name from cars24 where year = '2015' group by 1 order by 1 limit 10                                 | 10 row(s) returned   | 0.015 sec / 0.000 sec |
| 6 | 22:05:25 | select name, year from cars24 where year = '2017' group by 1,2 order by 1 asc LIMIT 0, 50000               | 420 row(s) returned  | 0.016 sec / 0.000 sec |
| 7 | 22:16:16 | select name, mileage from cars24 group by 1,2 having mileage < '10.2' order by 1 LIMIT 0, 50000            | 14 row(s) returned   | 0.032 sec / 0.000 sec |
| 8 | 22:23:31 | Select name, mileage, engine [cc] from cars24 having engine [cc] > '1000' and mileage > '20' order ...     | 2623 row(s) returned | 0.016 sec / 0.015 sec |
| 9 | 22:29:10 | select name, year, selling_price from cars24 where year = '2018' having selling_price = '560000' order ... | 7 row(s) returned    | 0.015 sec / 0.000 sec |



## Insight no. 9 –

--Identify top 25 car models which have maximum power is greater than '100'. Print name, number of kilometer and maximum power. Sort the result set in ascending order.

The screenshot displays the MySQL Workbench interface. The SQL editor contains the following query:

```
1  --9. Identify top 25 car models which have maximum power is greater than '100'. Print name, number of kilometer and maximum power. sort the result in ascending order
2  select name, km_driven, max_power
3  from cars24
4  having max_power > '100'
5  order by 1
6  limit 25 ;
7
```

The 'Result Grid' shows the following data:

| name                                 | km_driven | max_power |
|--------------------------------------|-----------|-----------|
| Audi A3 35 TDI Premium Plus          | 20000     | 143       |
| Audi A2 35 TDI Premium Plus          | 70000     | 143       |
| Audi A3 40 TFSI Premium              | 8000      | 177.5     |
| Audi A4 1.8 TFSI                     | 55000     | 163.2     |
| Audi A4 2.0 TDI                      | 60000     | 147.51    |
| Audi A4 2.0 TDI                      | 49000     | 147.51    |
| Audi A4 2.0 TDI 177 Bhp Premium Plus | 110000    | 174.33    |
| Audi A4 35 TDI Premium Plus          | 37000     | 187.74    |
| Audi A4 35 TDI Premium Plus          | 15000     | 187.74    |
| Audi A4 35 TDI Premium Plus          | 46000     | 187.74    |
| Audi A4 35 TDI Premium Plus          | 30000     | 187.74    |
| Audi A6 2.0 TDI                      | 60000     | 174.33    |
| Audi A6 2.0 TDI Design Edition       | 50000     | 187.74    |

The 'Output' tab shows the execution log:

| #  | Time     | Action  | Message              | Duration / Fetch      |
|----|----------|---|----------------------|-----------------------|
| 6  | 22:05:25 | select name, year from cars24 where year = 2017 group by 1,2 order by 1 asc LIMIT 0, 50000              | 420 row(s) returned  | 0.016 sec / 0.000 sec |
| 7  | 22:16:16 | select name, mileage from cars24 group by 1,2 having mileage < 10.2 order by 1 LIMIT 0, 50000           | 14 row(s) returned   | 0.032 sec / 0.000 sec |
| 8  | 22:23:31 | select name, mileage, 'engine [cc]' from cars24 having 'engine [cc]' > 1000 and mileage > 20' order ... | 2523 row(s) returned | 0.016 sec / 0.015 sec |
| 9  | 22:29:18 | select name, year, selling_price from cars24 where year = 2018 having selling_price = 560000' order ... | 7 row(s) returned    | 0.015 sec / 0.000 sec |
| 10 | 22:32:54 | select name, km_driven, max_power from cars24 having max_power > 100' order by 1 limit 25               | 25 row(s) returned   | 0.016 sec / 0.000 sec |

## Insight no. 10 –

--Print all the car database and one car model which is 'Maruti Alto Xs' in the database. Sort the result in ascending order.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'masai' selected. The main editor shows a SQL query in 'SQL File 5\*':

```
1 --10. Print all the car database and one car model which is 'Maruti Alto Xs' in the database. Sort the output in ascending order.
2 select *
3 from cars24
4 where name like 'Maruti Alto Xs'
5 order by 1;
```

The 'Result Grid' displays the query results in a table with the following columns: name, year, selling\_price, km\_driven, fuel, seller\_type, transmission, owner, mileage, engine [cc], max\_power, and seats. The results are sorted by the 'name' column in ascending order.

| name           | year | selling_price | km_driven | fuel   | seller_type | transmission | owner                | mileage | engine [cc] | max_power | seats |
|----------------|------|---------------|-----------|--------|-------------|--------------|----------------------|---------|-------------|-----------|-------|
| Maruti Alto LX | 2000 | 108000        | 206000    | Petrol | Individual  | Manual       | Fourth & Above Owner | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2011 | 175000        | 175000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2003 | 100000        | 164000    | Petrol | Individual  | Manual       | Third Owner          | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2005 | 85000         | 150000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2010 | 123000        | 120000    | Petrol | Individual  | Manual       | Third Owner          | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2007 | 110000        | 120000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2004 | 60000         | 113000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2012 | 160000        | 110000    | Petrol | Individual  | Manual       | First Owner          | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2009 | 115999        | 110000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2010 | 170000        | 100000    | Petrol | Individual  | Manual       | First Owner          | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2006 | 150000        | 100000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2006 | 120000        | 100000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2004 | 100000        | 100000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2005 | 90000         | 100000    | Petrol | Individual  | Manual       | Fourth & Above Owner | 19.7    | 796         | 46.3      | 5     |
| Maruti Alto LX | 2006 | 65000         | 100000    | Petrol | Individual  | Manual       | Second Owner         | 19.7    | 796         | 46.3      | 5     |

The 'Output' pane at the bottom shows the execution log with the following entries:

| #  | Time     | Action  | Message              | Duration / Fetch      |
|----|----------|---|----------------------|-----------------------|
| 7  | 22:16:16 | select name, mileage from cars24 group by 1,2 having mileage < '10.2' order by 1 LIMIT 0, 50000             | 14 row(s) returned   | 0.032 sec / 0.000 sec |
| 8  | 22:23:31 | Select name, mileage, 'engine [cc]' from cars24 having 'engine [cc]' > '1000' and mileage > '20' order by 1 | 2623 row(s) returned | 0.016 sec / 0.015 sec |
| 9  | 22:29:18 | select name, year, selling_price from cars24 where year = '2018' having selling_price = '560000' order by 1 | 7 row(s) returned    | 0.015 sec / 0.000 sec |
| 10 | 22:32:54 | select name, km_driven, max_power from cars24 having max_power > '100' order by 1 limit 25                  | 25 row(s) returned   | 0.016 sec / 0.000 sec |
| 11 | 22:36:27 | select * from cars24 where name like 'Maruti Alto Xs' order by 1 LIMIT 0, 50000                             | 44 row(s) returned   | 0.031 sec / 0.000 sec |