

# GROUP K

Date: MAY 28, 2021

# *Our Team*



*Sukhman*



*Bhumanyu Singh*



*Varun Nanda*



*Himanshu Dhahana*



*Muhammad Arsalan Alam*

# Reports Highlights



- Database Design Details
- ER Diagrams
- SQL Commands
- Evidence And Question
- SQL QUERIES
- SCREENSHOTS

# **TYPES OF DATABASE**

There are various types of data

1. Structural Database
2. Free-Form Database
3. Operational Database
4. Analytical Database
5. Rational Database

For this project, we have used  
Relational Database.

1. **DEFINITION:** A database structured to recognize relations between stored items of information.

# Types of relations in a Relational Database

There are three types of relation as follows

## 1. One-to-One

A row in a table can have only one matching row in the second table, and vice versa.

## 2. One-to-Many (or Many-to-One)

A row in one table can have many matching rows in other, but a row in second table can have only one matching row in the first one.

## 3. Many-to-Many

In a many-to-many relationship, a row in first table can have many matching rows in second table, and vice versa.

# *Details of Database Design*

We have created  
three tables

Each Table has  
attributes

Relationships  
between tables

Most used  
commands

Table 1 Cars

Sales\_id ,name,year,km  
\_driven,selling\_price,fuel  
,mileage,engine,max\_power,  
torque and seats

Table 2 Seller

Seller\_id,Seller\_type  
,and owner

Table 3 Dealers

Seller\_id and  
dealer

Seller table has many  
to many relation with  
cars table

Cars table has many to  
one relation with  
dealers table

CREATE TABLE ,SELECT, SELECT,UPDATE,DELETE,INSERT INTO,CREATE DATABASE,ALTER  
DATABASE,DROP TABLE.

# OUR FEATURES

## FEATURE 3

We have generated total\_revenue using SUM command

## FEATURE 1

We have created a primary key - Sales\_id in the table Cars and this act as foreign key in rest of the two tables

## FEATURE 2

The database has been cleaned before working and contains no null values

## FEATURE 4

We have calculated minimum selling price using SELECT MIN() command

## FEATURE 5

We have calculated minimum mileage using SELECT MIN() command



# SQL Commands

SELECT- extracts data from a database

UPDATE- updates data in a database

DELETE- deletes data from a database

INSERT INTO- inserts new data into a database

CREATE DATABASE- creates a new database

CREATE TABLE- creates a new table

ALTER TABLE- modifies a table

DROP TABLE- deletes a table

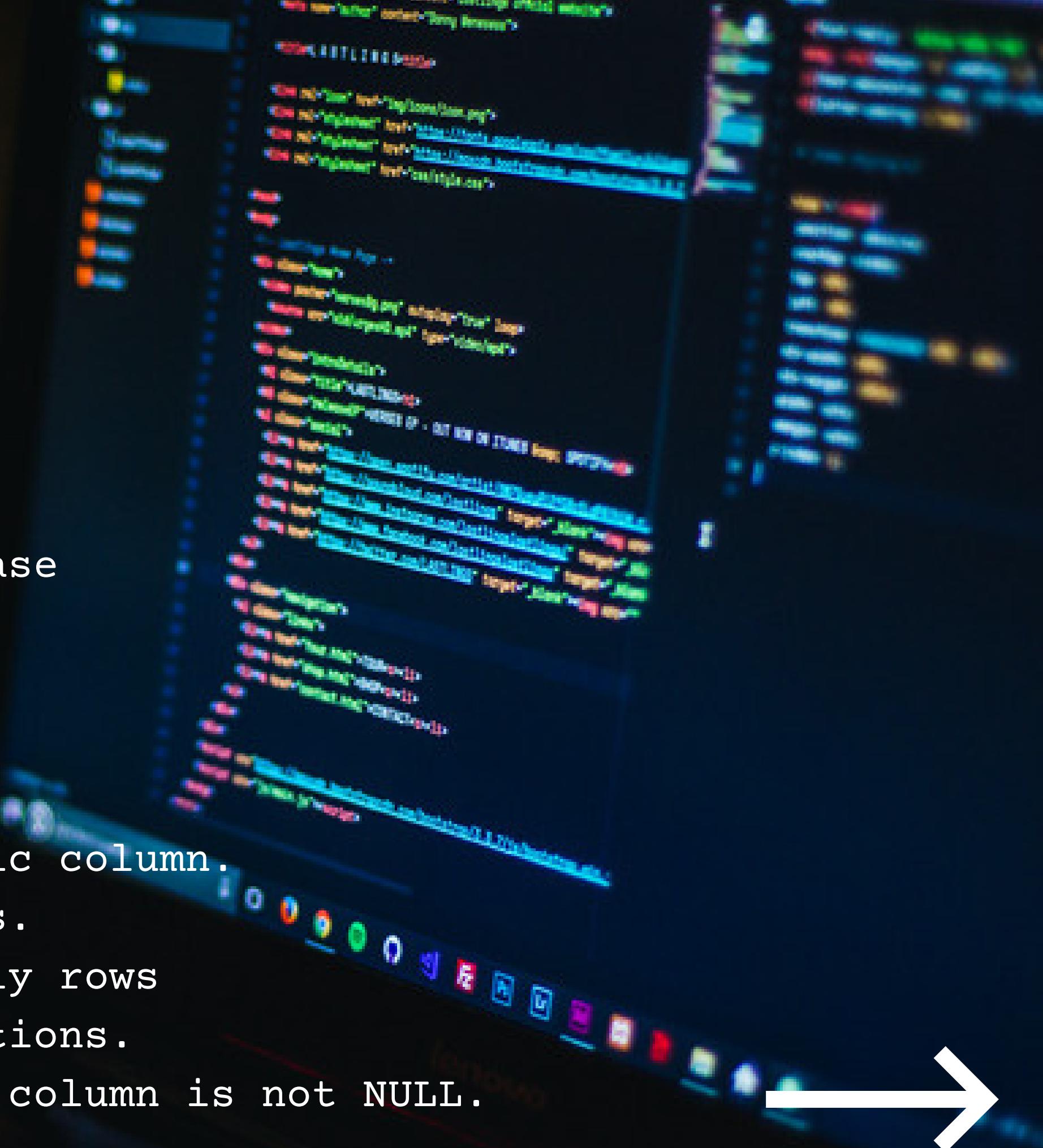
AVG- returns the average value for a numeric column.

As- rename a column or table using an alias.

WHERE- filter the result set to include only rows

AND is an operator that combines two conditions.

COUNT- counts the number of rows where the column is not NULL.



**GROUP BY-** used with aggregate functions.

**SUM-** is a function that takes the name of a column as an argument and returns the sum of all the values in that column.

**Inner JOIN-** will combine rows from different tables if the join condition is true.

**ORDER BY-** sort the result set by a particular column

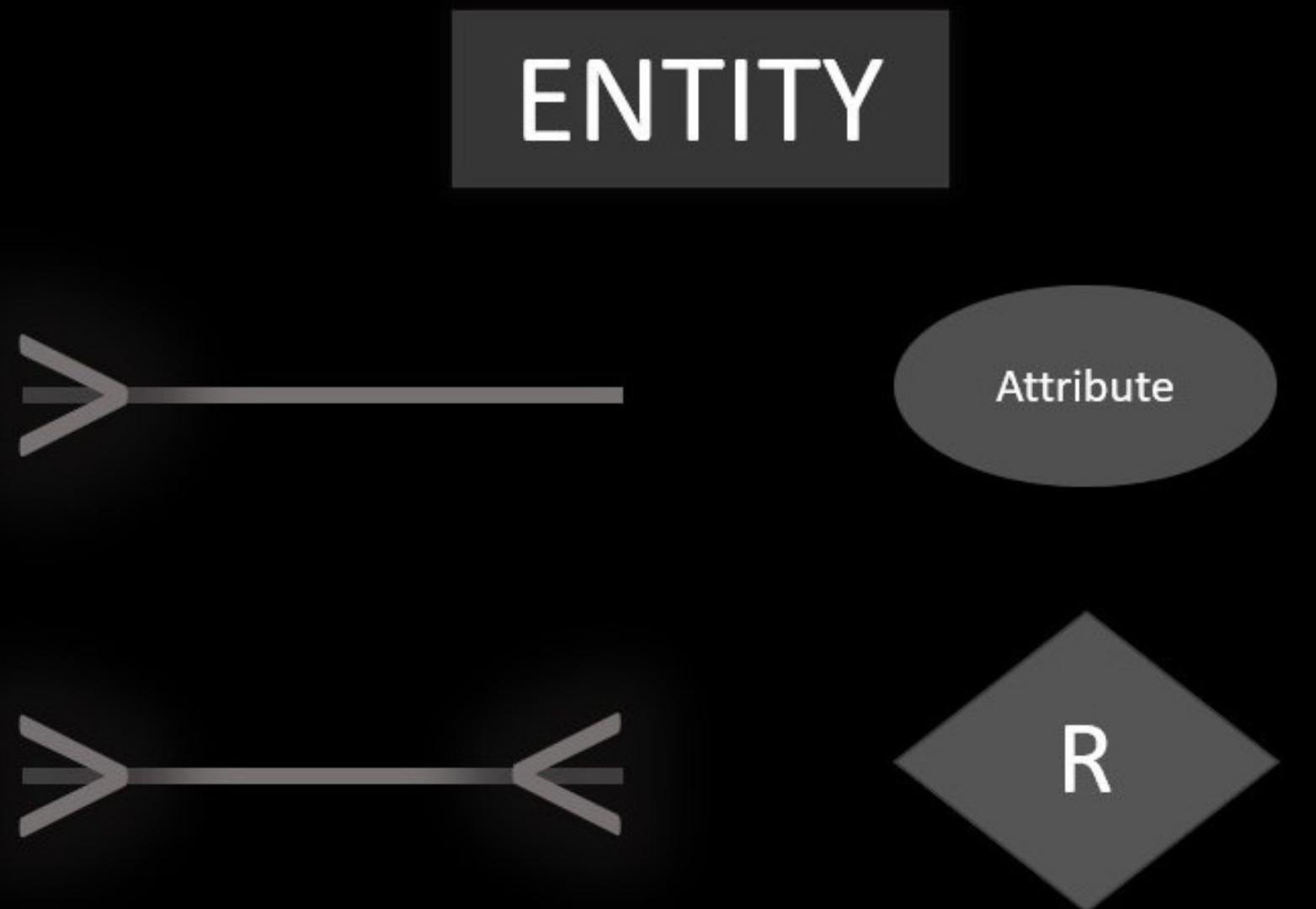
**LIMIT-** specify the maximum number of rows

**MIN-** returns the smallest value in that column.

**MAX-** returns the largest value in that column.

**OR-** filters the result set

# ER DIAGRAM explanation



Entities are shown with rectangles and attributes with ovals shape .



The relational line joining cars table to dealers table has three joining lines at cars table and one at dealers table, this shows the MANY to ONE relation .

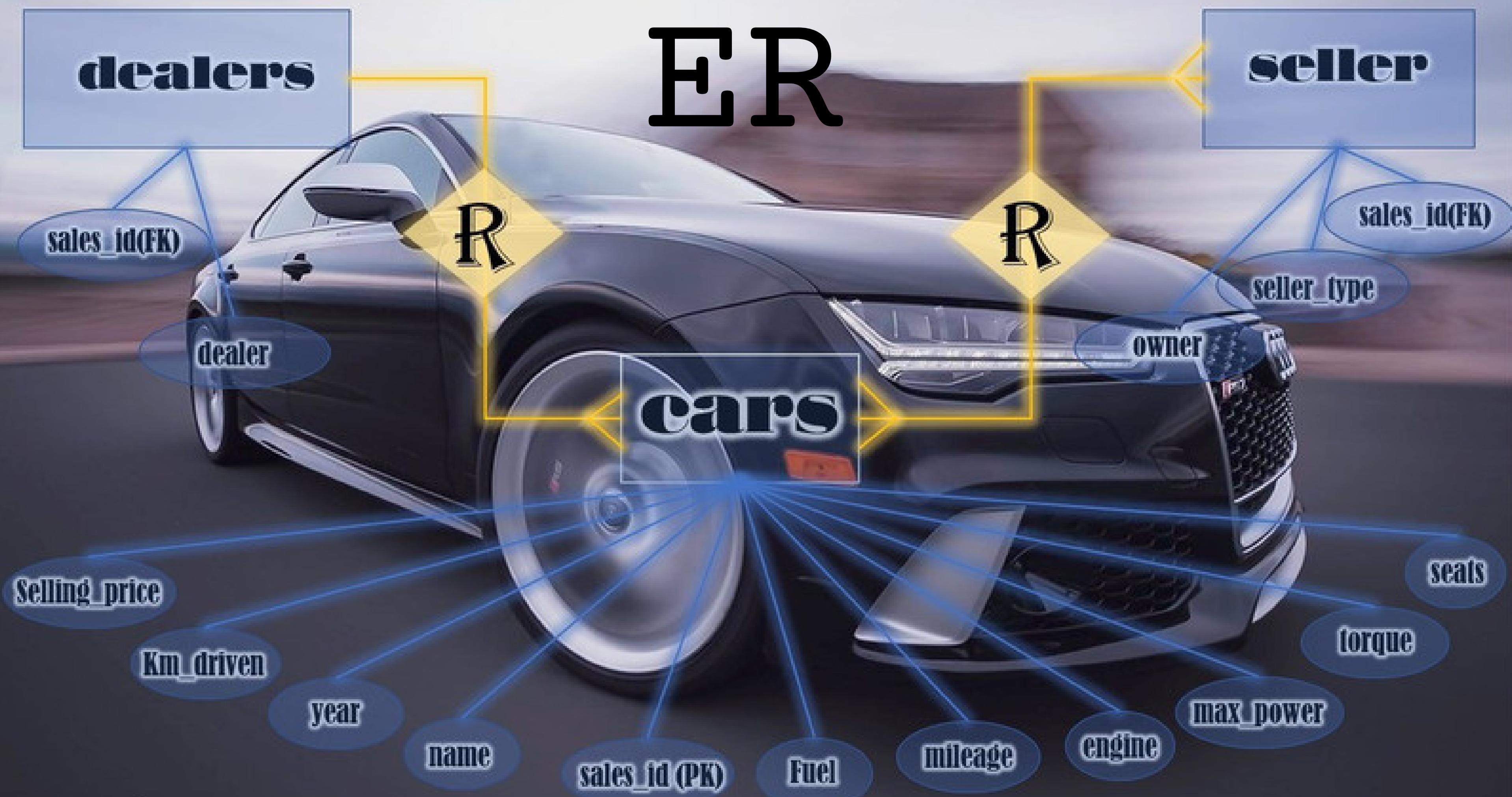


The relational line joining cars table to seller table has three joining lines at both tables, this shows the MANY to MANY relation .



The R in diamond shape shows that there is a relation between tables.

# ER





# Evidence And QUESTION



NEXT →

# Find out which car model sold the most ?

11/17

1. We have given the command `SELECT car_mod,COUNT(*)` this command will count total cars of each type from cars table,
2. Then we have given the command `ORDER BY COUNT(*) DESC` which will arrange it in descending order.
3. We have added `LIMIT 1` which will give us the value on the top

**CONCLUSION - MARUTI SUZUKI SWIFT DESIRE VDI WAS SOLD THE MOST (162).**

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

```
1 * SELECT car_mod, COUNT(*)
2   FROM cars
3   GROUP BY car_mod
4   ORDER BY COUNT(*) DESC
5   LIMIT 1;
```

The results grid shows one row of data:

car_mod	COUNT(*)
Maruti Swift Desire VDI	162

A watermark or logo for "HAWKDB" is visible on the right side of the screen.

# Which dealers' sold more cars, and what is his total sales?

1. We have used the command `SELECT Dealer , COUNT(dealers.sales_id) as count,SUM(selling_price) as total_sales` this command will count the number of dealers ,will sum up the selling price from dealers table .
2. LEFT JOIN will join the tables cars and dealers .
3. ORDER BY total\_sales will tell us the total amount of sales by each dealer

**CONCLUSION - The MOST SALES ARE DONE BY ANNY. SHE DID 828 DEALS FOR 1203367970.00**

The screenshot shows the MySQL Workbench interface with a query editor window. The query is:`1 • SELECT Dealer , COUNT(dealers.sales_id) as count, SUM(selling_price) as total_sales  
2 FROM dealers LEFT JOIN cars ON dealers.sales_id = cars.sales_id  
3 GROUP BY Dealer  
4 ORDER BY total_sales desc;`The results grid displays the following data:

	Dealer	count	total_sales
▶	Anny	828	1203367970
▶	David	175	372871991
▶	Henry	304	138879996

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

Schemas

car\_sales

Tables

car dataset-mind\_new

cars

dealers

seller

Views

Stored Procedures

Functions

hawkdb

sys

Administration Schemas

No object selected

Query 1

1. SELECT car\_mod, AVG(selling\_price) as average\_price  
FROM cars  
GROUP BY car\_mod  
ORDER BY average\_price DESC;

Result Grid

car_mod	average_price
Volvo XC90 T8 Excellence BSV	53000000.0000
BMW X7 xDrive 30d DPT	72000000.0000
Audi A4 35 TFSI Matrix	67000000.0000
Mercedes-Benz S-Class S 350 CDI	59625000.0000
BMW 6 Series GT 630d Luxury Line	58600000.0000
Volvo XC60 Inscription D5 BSV	55000000.0000
Volvo S90 D4 Inscription BSV	55000000.0000
BMW X4 M Sport X xDrive20d	54612900.3226
Mercedes-Benz E-Class E 200 BSV	52000000.0000
Lexus ES 300h	51500000.0000
Mercedes-Benz GL-Class 220d 4MATIC Sport	46000000.0000
Land Rover Discovery Sport TD4 SE	45000000.0000

Result 4

Action Output

#	Time	Action
1	15:42:20	SELECT seller_type, SUM(selling_price) as total_revenue FROM seller L
2	15:45:03	SELECT car_mod, COUNT(*) FROM cars GROUP BY car_mod ORDER
3	15:45:23	SELECT Dealer , COUNT(dealers.seller_id) as count, SUM(selling_price)
4	15:48:00	SELECT car_mod, AVG(selling_price) as average_price FROM cars GR

# What is the average price for each car model?

1. We have given the command `SELECT car_mod, AVG(selling_price) as average_price from cars table`, this will calculate the average selling price .

2. We have used the command `ORDER BY average_price DESC`; this will arrange the average selling price of each mode in descending order.

**CONCLUSION - CAN BE CALCULATED AS SHOWN IN THE SCREENSHOT .**

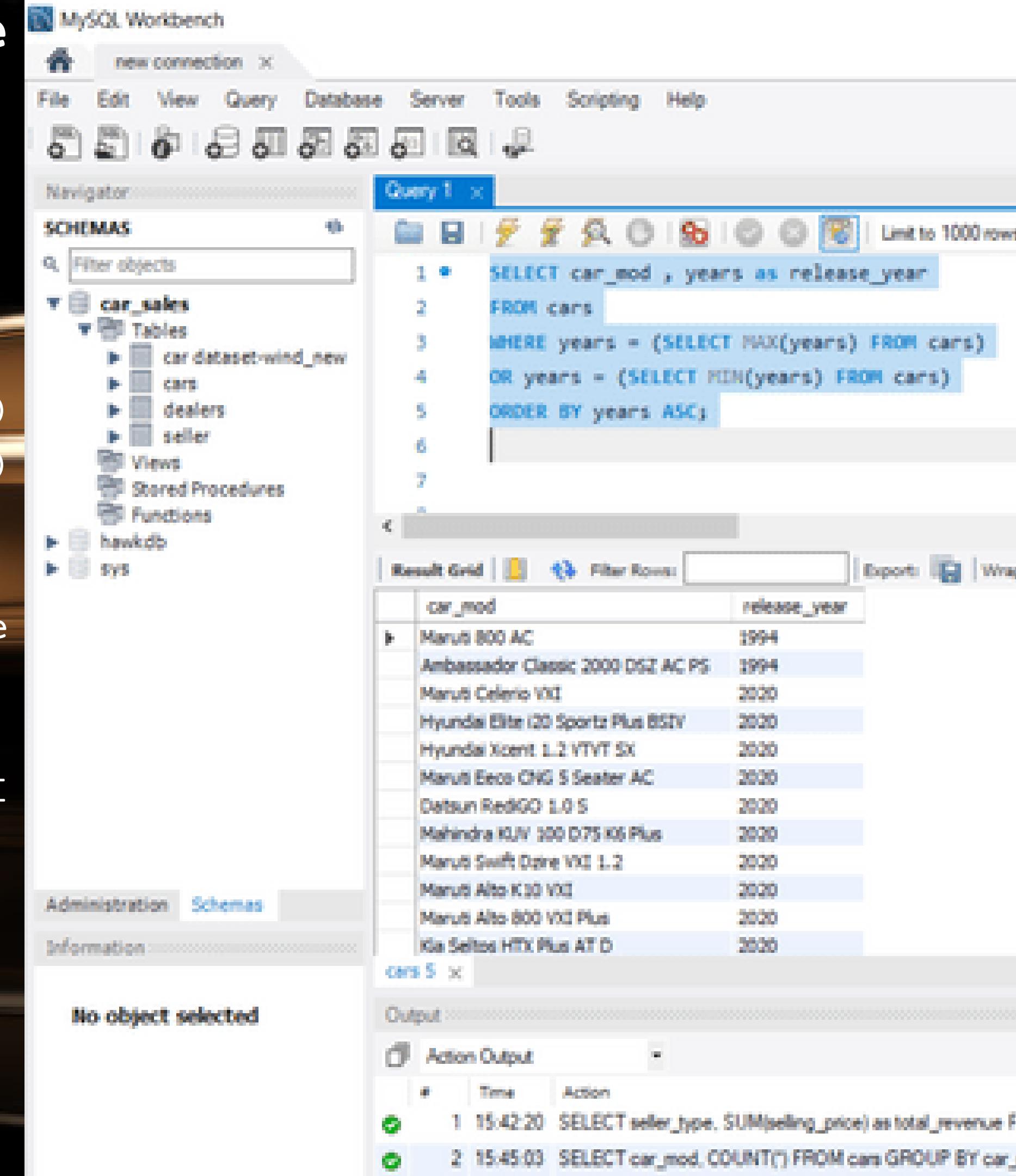
# What is the newest car and the oldest car?

We have used the command `SELECT car_mod , years as release_year from table cars ,` this will show the release year of cars .

We have given the command `(SELECT MAX(years) FROM cars), and (SELECT MIN(years) FROM cars)` , this will give the minimum and maximum years attributes .  
`ORDER BY years ASC;` will arrange the outcome data in ascending order .

CONCLUSION - NEWEST CAR - MARUTI CELERIO VXI , HYUNDAI ELITE I20 SPORTS PLUS BS IV(2020) (+38 more cars in same year, due to no data on exact release date) ETC .

OLDEST CAR IS MARUTI 800 AC (1994) AND AMBASSADOR CLASSIC 2000 DSZ AC PS.



The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The left sidebar has sections for Navigator, Schemas, and a Query editor tab labeled 'Query 1'. The 'Schemas' section shows a tree view of database objects under the 'car\_sales' schema, including Tables (car\_datsetwind\_new, cars, dealers, seller), Views, Stored Procedures, Functions, and two temporary tables (#tmp\_1, #tmp\_2). The 'Query 1' tab contains the following SQL code:

```
1 • SELECT car_mod , years as release_year  
2 FROM cars  
3 WHERE years = (SELECT MAX(years) FROM cars)  
4 OR years = (SELECT MIN(years) FROM cars)  
5 ORDER BY years ASC;
```

The results grid below displays the data:

car_mod	release_year
Maruti 800 AC	1994
Ambassador Classic 2000 DSZ AC PS	1994
Maruti Celerio VXI	2020
Hyundai Elite i20 Sports Plus BSIV	2020
Hyundai Xcent 1.2 VTVT SX	2020
Maruti Eeco CNG 5 Seater AC	2020
Datsun RedGO 1.0 S	2020
Mahindra KUV 100 D75 AC Plus	2020
Maruti Swift Dzire VXI L3	2020
Maruti Alto K10 VXI	2020
Maruti Alto 800 VXI Plus	2020
Kia Seltos HTX Plus AT D	2020

The bottom status bar shows 'No object selected' and the output pane indicates two actions were run at 15:42:20 and 15:45:03.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

**Navigator**

**SCHEMAS**

Filter objects

**car\_sales**

- Tables
  - car datasetwind\_new
  - cars
  - dealers
  - seller
- Views
- Stored Procedures
- Functions

hawkdb

sys

Administration Schemas

Information

No object selected

Output

Action Output

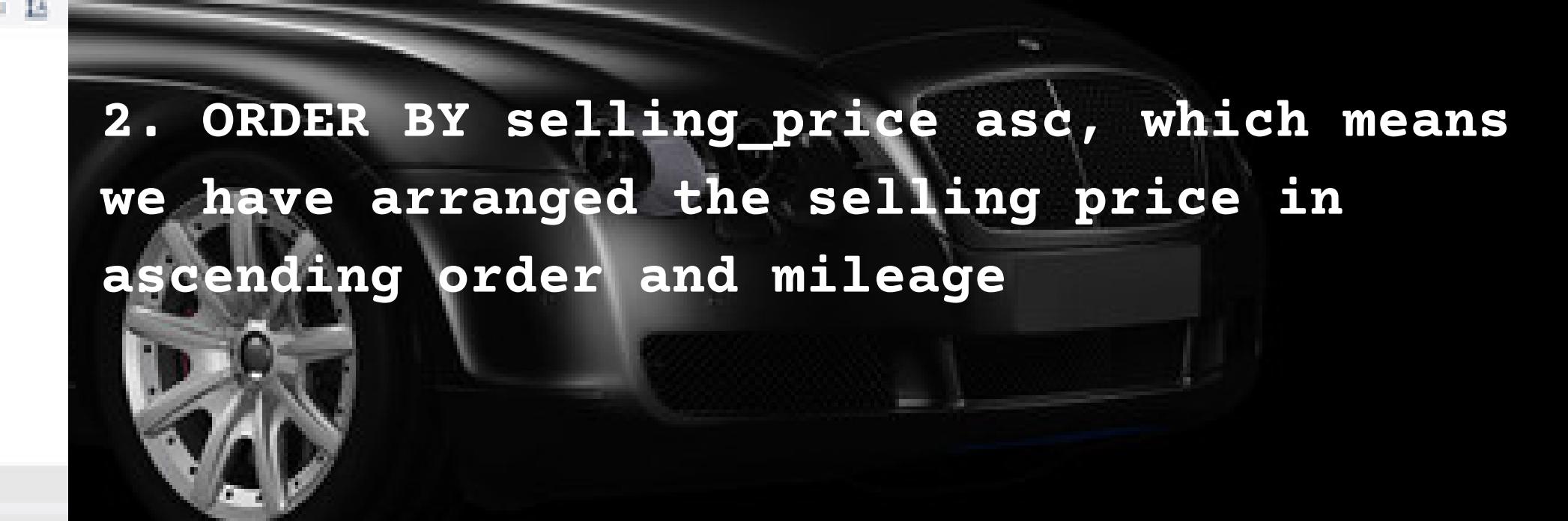
#	Time	Action
1	15:42:20	SELECT seller_type, SUM(selling_price) as total_revenue FROM seller LEFT JOIN car ON seller.seller_id = car.seller_id GROUP BY seller_type ORDER BY total_revenue DESC;
2	15:45:03	SELECT car_mod, COUNT(*) FROM cars GROUP BY car_mod ORDER BY COUNT(*) DESC;
3	15:45:23	SELECT Dealer.COUNT(dealer_sales_id) as count, SUM(selling_price) as total_revenue FROM dealer LEFT JOIN car ON dealer.dealer_id = car.dealer_id GROUP BY Dealer.COUNT(dealer_sales_id) ORDER BY total_revenue DESC;
4	15:48:00	SELECT car_mod, AVG(selling_price) as average_price FROM cars GROUP BY car_mod;
5	15:48:33	SELECT car_mod, year as release_year FROM cars WHERE year = (SELECT MIN(year) FROM cars);
6	15:49:16	SELECT car_mod, selling_price, mileage FROM cars WHERE selling_price = (SELECT MIN(selling_price) FROM cars) OR mileage = (SELECT MIN(mileage) FROM cars) ORDER BY selling_price asc, mileage asc;

Query Completed

# Which is the best car related to price, and low mileage?

1. We have used the command `SELECT car_mod, selling_price, mileage` this will select the given attributes from cars table , then we will find the minimum selling price and mileage.

2. `ORDER BY selling_price asc`, which means we have arranged the selling price in ascending order and mileage



**CONCLUSION - MARUTI 800 AC HAS A SELLING OF 29999.00 and mileage 16.1 kmpl as shown in the screenshot. THE OTHER CARS HAS 0.0KMPL MILEAGE WHICH IS UNFEASIBLE .**

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

car\_sales

Tables

- car\_datasetwind\_new
- cars
- dealers
- seller

Views

Stored Procedures

Functions

hawkdbs

sys

Query 1

```
1 * | SELECT seller_type, SUM(selling_price) as total_revenue
2   FROM seller LEFT JOIN cars ON seller.sales_id = cars.sales_id
3   WHERE seller_type = 'Individual';
```

Result Grid | Filter Rows | Export | Wrap Cell Contents

seller_type	total_revenue
Individual	3332153320

# What is the total revenue from Individual sale?

1. We have given the command  
SELECT seller\_type,  
SUM(selling\_price) as total\_revenue  
,this will calculate the sum of selling  
price .
2. With the command LEFT JOIN will  
join the table seller and cars on Sales\_id  
the WHERE command will give the  
results for individual seller.



# SQL QUERY USED

01  

```
ALTER TABLE cars
ADD PRIMARY KEY (sales_id)
```

02  

```
ALTER TABLE dealers
ADD FOREIGN KEY (sales_id)
REFERENCES cars(sales_id);
```

03  

```
ALTER TABLE seller
ADD FOREIGN KEY (sales_id)
REFERENCES cars(sales_id);
```

04  

```
SELECT car_mod, COUNT(*)
FROM cars
GROUP BY car_mod
ORDER BY COUNT(*) DESC
```

05  

```
SELECT Dealer , COUNT(dealers.sales_id) as count,
SUM(selling_price) as total_sales
FROM dealers LEFT JOIN cars on dealers.sales_id =
cars.sales_id
GROUP BY Dealer
ORDER BY total_sales desc;
```

06  

```
SELECT car_mod, AVG(selling_price) as average_price
FROM cars
GROUP BY car_mod
ORDER BY average_price DESC;
```

07  

```
SELECT car_mod , years as release_year
FROM cars
WHERE years = (SELECT MAX(years) FROM cars)
OR years = (SELECT MIN(years) FROM cars)
ORDER BY years ASC;
```

08  

```
SELECT car_mod, selling_price, mileage
FROM cars WHERE selling_price = (SELECT MIN(selling_price)
FROM cars) OR mileage = (SELECT MIN(mileage) FROM cars)
ORDER BY selling_price asc, mileage asc;
```

09  

```
SELECT seller_type, SUM(selling_price) as total_revenue
FROM seller LEFT JOIN cars ON
seller.sales_id = cars.sales_id
WHERE seller_type = 'Individual';
```

**HERE  
ARE  
SOME  
SCREENSHOTS**





MySQL Shell 8.0.23

Copyright (c) 2016, 2021, Oracle and/or its affiliates.  
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.  
Other names may be trademarks of their respective owners.

Type 'help' or '?' for help; 'quit' to exit.

MySQL [5] > \c root@localhost:3306

Creating a session to 'root@localhost:3306'

Fetching schema names for autocomplete... Press ^C to stop.

Your MySQL connection id is 40485

Server version: 8.0.23 MySQL Community Server - GPL

No default schema selected; type \use <schema> to set one.

MySQL localhost:3306 ssl [5] > \sql

Switching to SQL mode... Commands end with ;

MySQL localhost:3306 ssl [SQL] > CREATE DATABASE Car\_Sales;

Query OK, 1 row affected (1.0695 sec)

MySQL localhost:3306 ssl [SQL] >



MySQL Workbench

Local instance MySQL80 ×

File Edit View Query Database Server Tools Scripting Help

# TABLE COMMANDS SCREENSHOT

Navigator

SCHEMAS

Filter objects

- projectdb
- Tables
  - cars
  - dealers
  - main
  - seller
- Views
- Stored Procedures
- Functions
- sys

Administration Schemas

Information

Table: dealers

Columns:

- dealer char(30)
- sales\_id int

Query 1 SQL File 3\* SQL File 4\* main cars seller dealers

```
18 * ① create table Cars(
19     sales_id int,
20     name varchar(100),
21     year int(10),
22     km_driven int(10),
23     selling_price char(10),
24     fuel varchar(20),
25     transmission char(30),
26     mileage char(20),
27     engine char(30),
28     max_power char(30),
29     torque char(100),
30     seats int (3)
31 );
32
33 * ② create table Seller(
34     owner varchar(30),
35     seller_type varchar(30),
36     sales_id int(3)
37 );
38
39 * ③ create table dealers(
40     dealer char(30),
41 );
```

SQLAdditions

< > | Jump to

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

TO BE CONTINUE →

Action Output

#	Time	Action	Message	Duration / Fetch
34	22:58:22	PREPARE stmt FROM INSERT INTO 'projectdb'.'dealers' ('dealer', 'sales_id') VALUES (?,?)	OK	0.000 sec
35	22:58:28	DEALLOCATE PREPARE stmt;	OK	0.000 sec
36	22:58:36	SELECT * FROM projectdb.dealers LIMIT 0, 10000	1107 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

Search

23:58 22-05-2021

MySQL Workbench

Local instance MySQL80 ×

File Edit View Query Database Server Tools Scripting Help

Navigator

Query 1 SQL File 3\* SQL File 4\* main cars seller dealers

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

22      km\_driven int(10),  
23      selling\_price char(10),  
24      fuel varchar(20),  
25      transmission char(20),  
26      mileage char(20),  
27      engine char(20),  
28      max\_power char(20),  
29      torque char(100),  
30      seats int (3)  
31     );  
32  
33 \* ① create table Seller(  
34      owner varchar(20),  
35      seller\_type varchar(20),  
36      sales\_id int(5)  
37     );  
38  
39 \* ② create table dealers(  
40      dealer char(30),  
41      sales\_id int  
42     );  
43  
44

Output

Action Output

#	Time	Action	Message	Duration / Fetch
34	22:58:22	PREPARE stmt FROM INSERT INTO 'projectdb'.'dealers' ('dealer','sales_id') VALUES(?,?)	OK	0.000 sec
35	22:58:28	DEALLOCATE PREPARE stmt;	OK	0.000 sec
36	22:58:36	SELECT * FROM projectdb.dealers LIMIT 0, 10000	1107 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

Search

23:58 22-05-2021

## CARS TABLE Screenshot

Navigator

**SCHEMAS**

- Filter objects
- projectdb
- projectdb**
  - Tables
    - cars**
    - dealers
    - main
    - seller
  - Views
  - Stored Procedures
  - Functions
- sys

Administration Schemas

Information

Table: **dealers**

Columns:

dealer	char(30)
sales_id	int

CARS TABLE SCREENTSHOT

Query 1 SQL File 3\* SQL File 4\* main cars seller dealers

1 \* `SELECT * FROM projectdb.cars;`

Result Grid | Filter Rows | Export | Wrap Cell Contents

	sales_id	name	year	km_driven	selling_price	fuel	transmission	mileage	engine	max_power	torque	seats
1	1	Maruti Swift Dzire VDI	2014	145500	450000	Diesel	Manual	23.4 kmpl	1248 CC	74 bhp	190Nm@ 2000rpm	5
2	2	Skoda Rapid 1.5 TDI Ambition	2014	120000	370000	Diesel	Manual	21.14 kmpl	1498 CC	103.52 bhp	250Nm@ 1500-2500rpm	5
3	3	Honda City 2017-2020 EXi	2006	140000	158000	Petrol	Manual	17.71 kmpl	1497 CC	78 bhp	12.7@ 2,700(kgm@ rpm)	5
4	4	Hyundai i20 Sportz Diesel	2019	127000	225000	Diesel	Manual	23.0 kmpl	1396 CC	90 bhp	22.4 kgm at 1700-2700rpm	5
5	5	Maruti Swift VXi BSIII	2007	120000	130000	Petrol	Manual	16.1 kmpl	1298 CC	88.2 bhp	11.9@ 4,500(kgm@ rpm)	5
6	6	Hyundai Xcent 1.2 VTVT E Plus	2017	45000	440000	Petrol	Manual	20.14 kmpl	1397 CC	81.86 bhp	113.75Nm@ 4000rpm	5
7	7	Maruti Wagon R LXI DUO BSIII	2007	175000	96000	LPG	Manual	17.3 km/kg	1061 CC	57.5 bhp	7.8@ 4,500(kgm@ rpm)	5
8	8	Maruti 800 DX BSII	2001	5000	45000	Petrol	Manual	16.1 kmpl	796 CC	37 bhp	59Nm@ 2500rpm	4
9	9	Toyota Etios VXD	2011	90000	350000	Diesel	Manual	23.59 kmpl	1364 CC	67.1 bhp	170Nm@ 1800-2400rpm	5
10	10	Ford Figo Diesel Celebration Edition	2013	169999	299999	Diesel	Manual	20.0 kmpl	1399 CC	68.1 bhp	160Nm@ 2000rpm	5
11	11	Renault Duster 110PS Diesel RxL	2014	68000	500000	Diesel	Manual	19.01 kmpl	1461 CC	108.45 bhp	240Nm@ 2250rpm	5
12	12	Maruti Zen LX	2005	300000	92000	Petrol	Manual	17.3 kmpl	993 CC	60 bhp	78Nm@ 4500rpm	5
13	13	Maruti Swift Dzire VDI	2009	140000	280000	Diesel	Manual	19.3 kmpl	1248 CC	73.9 bhp	190Nm@ 2000rpm	5
14	14	Maruti Wagon R LXI Minor	2009	90000	180000	Petrol	Manual	18.9 kmpl	1061 CC	67 bhp	84Nm@ 3500rpm	5
15	15	Mahindra KUV 100 mFALCON G80 ...	2016	40000	400000	Petrol	Manual	18.15 kmpl	1198 CC	82 bhp	115Nm@ 3500-3600rpm	5
16	16	Maruti Ertiga SHVS VDI	2016	70000	778000	Diesel	Manual	24.52 kmpl	1248 CC	88.5 bhp	200Nm@ 1750rpm	7

cars 1 \* Read Only Context Help Snippets

Action Output

#	Time	Action	Message	Duration / Fetch
34	22:58:22	PREPARE stmt FROM INSERT INTO 'projectdb'.'dealers' ('dealer', 'sales_id') VALUES (?,?)	OK	0.000 sec
35	22:58:28	DEALLOCATE PREPARE stmt	OK	0.000 sec
36	22:58:36	SELECT * FROM projectdb.dealers LIMIT 0, 10000	1107 row(s) returned	0.015 sec / 0.000 sec

# SELLER TABLE Screenshot

The screenshot shows the MySQL Workbench interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 *   SELECT * FROM projectdb.seller;
```

The results grid displays the following data:

	owner	seller_type	sales_id
1	First Owner	Individual	1
2	Second Owner	Individual	2
3	Third Owner	Individual	3
4	First Owner	Individual	4
5	First Owner	Individual	5
6	First Owner	Individual	6
7	First Owner	Individual	7
8	Second Owner	Individual	8
9	First Owner	Individual	9
10	First Owner	Individual	10
11	Second Owner	Individual	11
12	Second Owner	Individual	12
13	Second Owner	Individual	13
14	Second Owner	Individual	14
15	First Owner	Individual	15
16	Second Owner	Individual	16

# DEALER TABLE SCREENSHOT

This screenshot shows the MySQL Workbench interface with a focus on the 'dealers' table.

**Schemas:**

- projectdb (selected)
- Tables:
  - cars
  - dealers
  - main
  - seller
- Views
- Stored Procedures
- Functions

**Table: dealers**

**Columns:**

dealer	sales_id
Henry	41
Henry	42
Henry	43
Henry	44
Henry	45
Henry	46
Henry	47
David	48
David	49
Anny	50
David	51
David	52
David	53
Anny	54
Anny	55
David	56

**Output:**

#	Time	Action	Message	Duration / Fetch
34	22:58:22	PREPARE stmt FROM INSERT INTO 'projectdb'.'dealers' ('dealer','sales_id') VALUES(?,?)	OK	0.000 sec
35	22:58:28	DEALLOCATE PREPARE stmt	OK	0.000 sec
36	22:58:36	SELECT * FROM projectdb.dealers LIMIT 0, 10000	1107 row(s) returned	0.015 sec / 0.000 sec