

- **SQL Scripts to Create tables and output screens-**

First, I show a screenshot of the MySQL Workbench environment with the script (Figures 8 and 14). Then, I present the script for each table followed by a screenshot of the table created (Figures 9, 10, 11, 12, 13, 15, 16, 17 and 18).

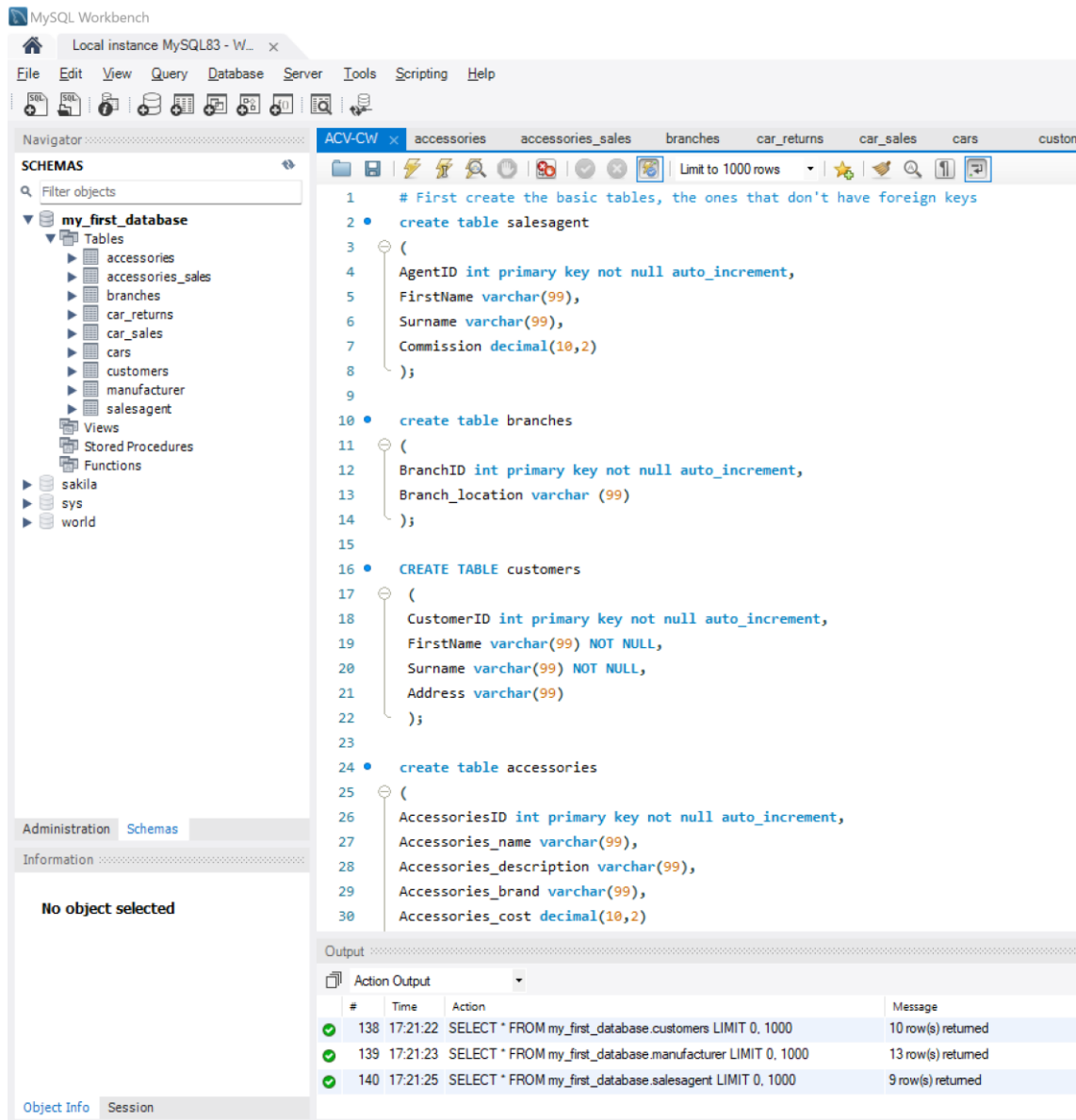


Figure 1. Screenshot of the script after running in MySQL Workbench

### # Script to create table salesagent:

```
create table salesagent
(  
  AgentID int primary key not null auto_increment,  
  FirstName varchar(99),  
  Surname varchar(99),  
  Commission decimal(10,2)  
);
```

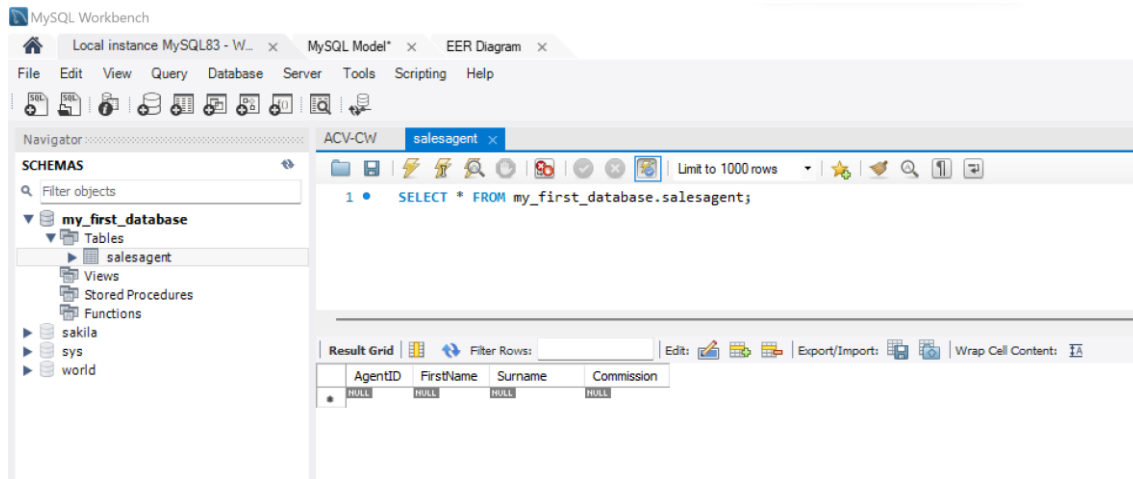


Figure 2. Screenshot of table Sales Agent in MySQL Workbench

### # Script to create table branches:

```
create table branches
(  
  BranchID int primary key not null auto_increment,  
  Branch_location varchar (99)  
);
```

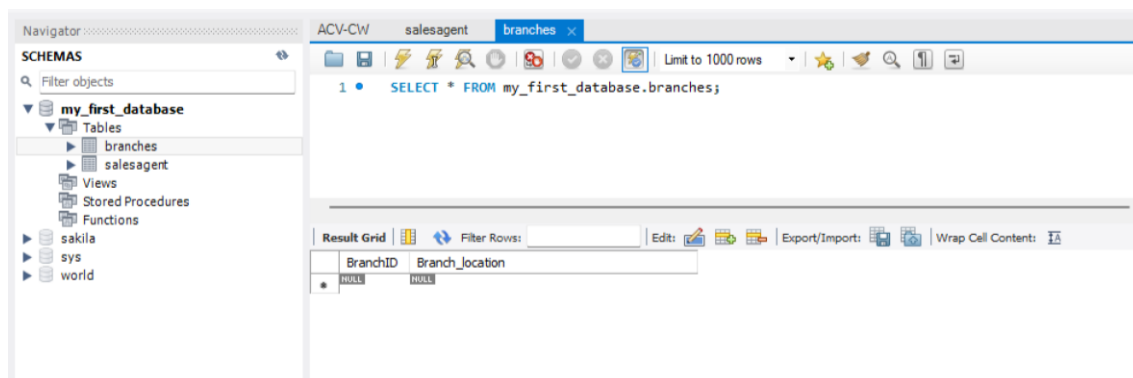


Figure 3. Screenshot of table Branches in MySQL Workbench

### # Script to create table customers:

```
CREATE TABLE customers
(
  CustomerID int primary key not null auto_increment,
  FirstName varchar(99) NOT NULL,
  Surname varchar(99) NOT NULL,
  Address varchar(99)
);
```

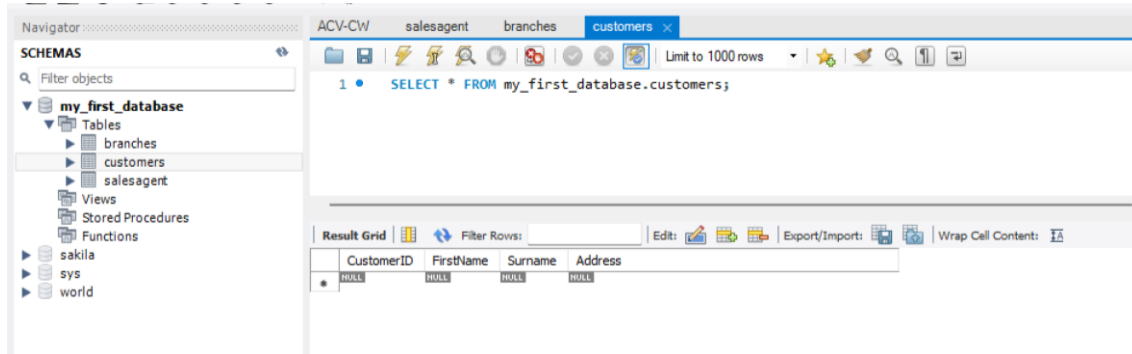


Figure 4. Screenshot of table Customers in MySQL Workbench

### # Script to create table accessories:

```
create table accessories
(
  AccessoriesID int primary key not null auto_increment,
  Accessories_name varchar(99),
  Accessories_description varchar(99),
  Accessories_brand varchar(99),
  Accessories_cost decimal(10,2)
);
```

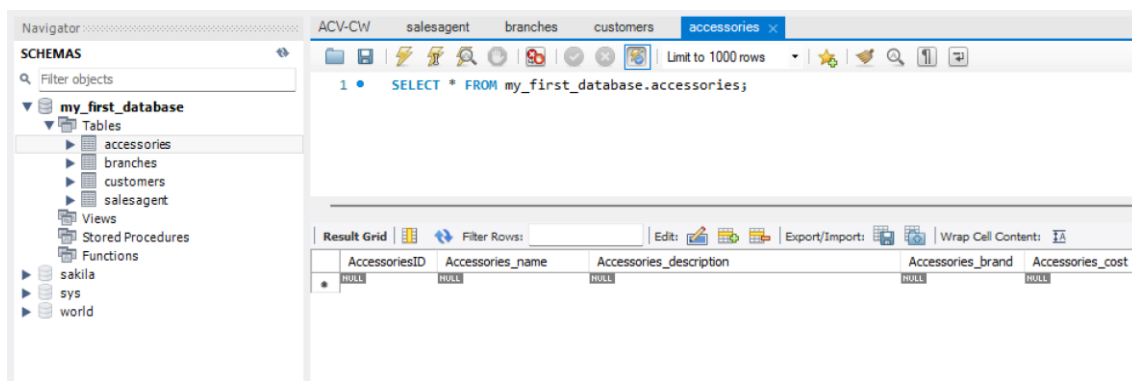


Figure 5. Screenshot of table Accessories in MySQL Workbench

## # Script to create table manufacturer:

```
CREATE TABLE manufacturer
(
  ManufacturerID int primary key not null auto_increment,
  Manufacturer varchar(99),
  Manufacturing_date date
);
```

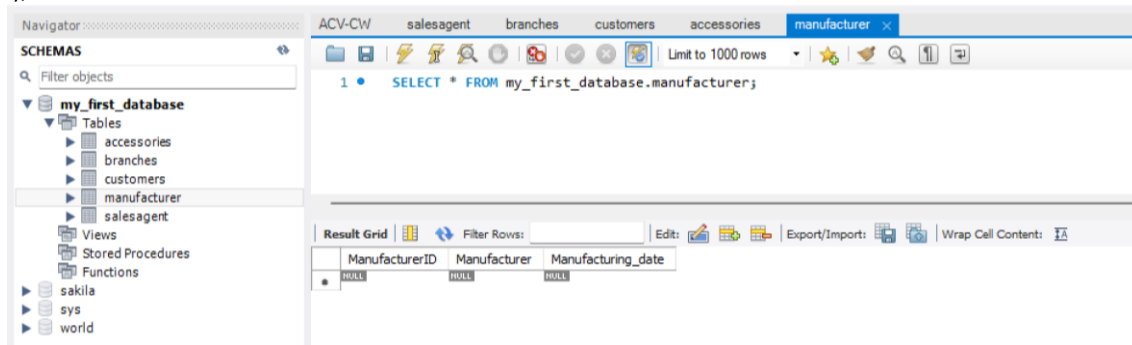


Figure 6. Screenshot of table Manufacturer in MySQL Workbench

Now the tables with foreign keys

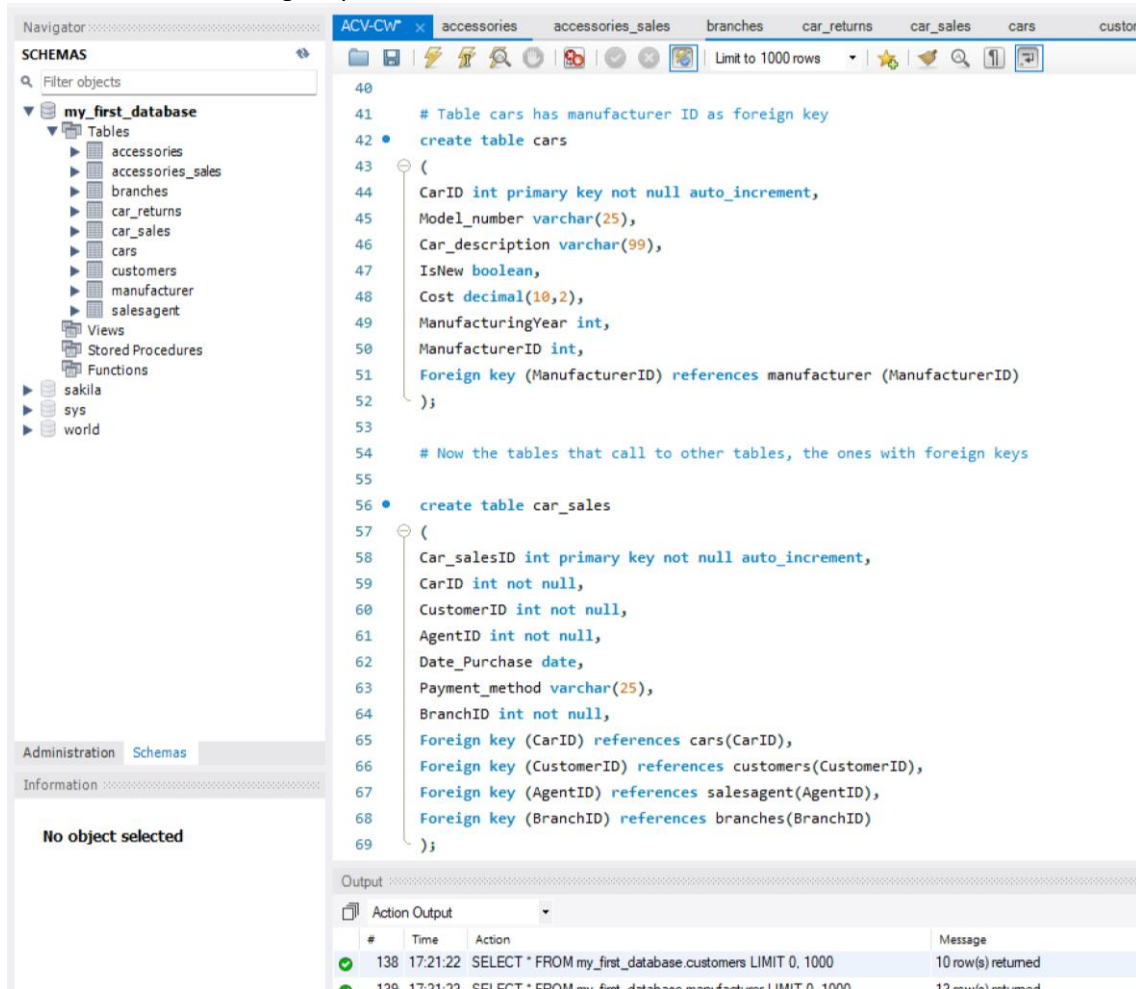


Figure 7. Screenshot of the script to create the tables after running in MySQL Workbench

### # Script to create table cars:

```
create table cars
(
  CarID int primary key not null auto_increment,
  Model_number varchar(25),
  Car_description varchar(99),
  IsNew boolean,
  Cost decimal(10,2),
  ManufacturingYear int,
  ManufacturerID int,
  Foreign key (ManufacturerID) references manufacturer (ManufacturerID)
);
```

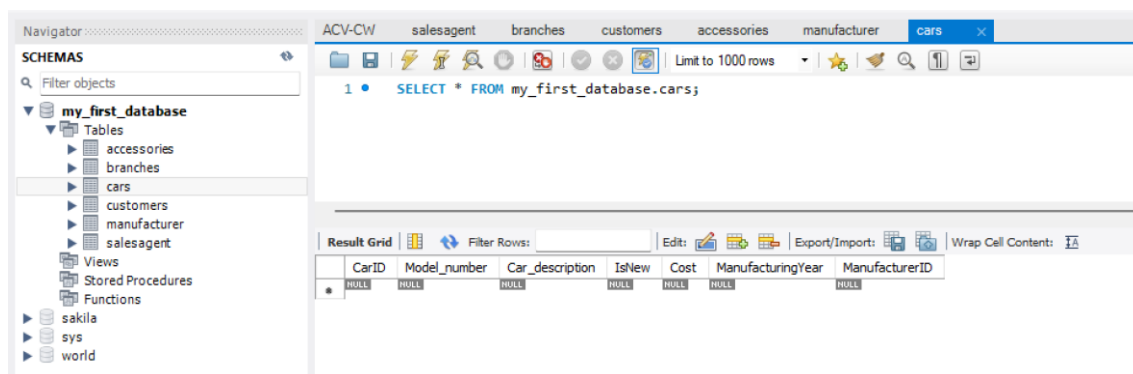


Figure 8. Screenshot of table Cars in MySQL Workbench

### # Script to create table car\_sales:

```
create table car_sales
(
  Car_salesID int primary key not null auto_increment,
  CarID int not null,
  CustomerID int not null,
  AgentID int not null,
  Date_Purchase date,
  Payment_method varchar(25),
  BranchID int not null,
  Foreign key (CarID) references cars(CarID),
  Foreign key (CustomerID) references customers(CustomerID),
  Foreign key (AgentID) references salesagent(AgentID),
  Foreign key (BranchID) references branches(BranchID)
);
```

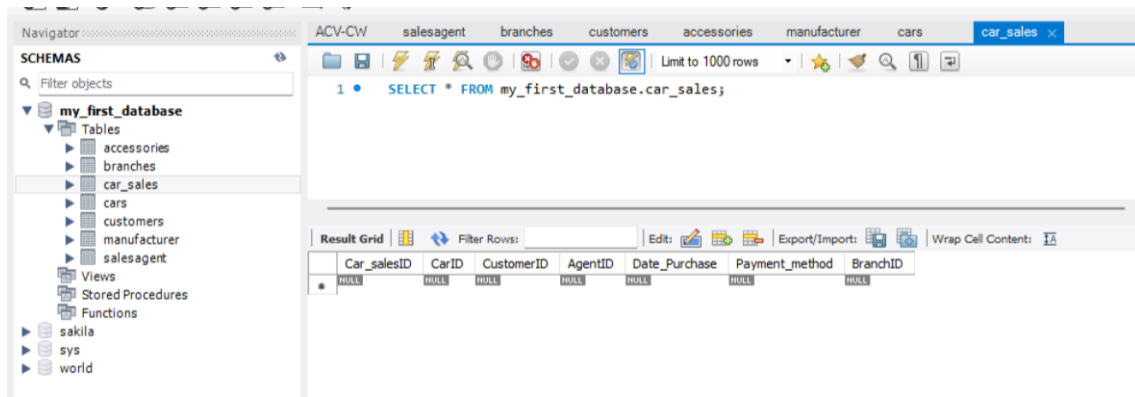


Figure 9. Screenshot of table Car Sales in MySQL Workbench

### # Script to create table accessories\_sales:

```
create table accessories_sales
(
Accessories_salesID int primary key not null auto_increment,
AccessoriesID int not null,
CustomerID int not null,
AgentID int not null,
Date_Purchase date,
Payment_method varchar(25),
BranchID int not null,
Foreign key (AccessoriesID) references accessories(AccessoriesID),
Foreign key (CustomerID) references customers(CustomerID),
Foreign key (AgentID) references salesagent(AgentID),
Foreign key (BranchID) references branches(BranchID)
);
```

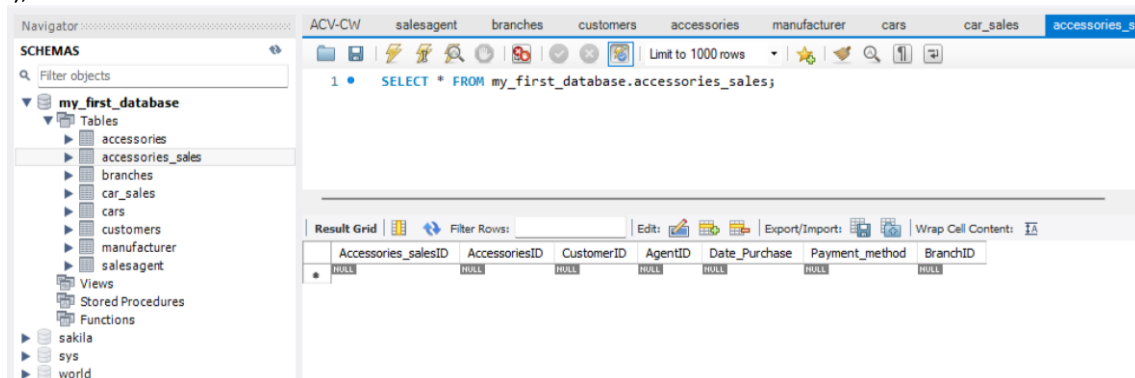


Figure 10. Screenshot of table Accessories Sales in MySQL Workbench

### # Script to create table car\_returns:

```
create table car_returns
(
  car_returnID int primary key not null auto_increment,
  CarID int not null,
  CustomerID int not null,
  AgentID int not null,
  Date_return date,
  Payment_method varchar(25),
  BranchID int not null,
  Foreign key (CarID) references cars(CarID),
  Foreign key (CustomerID) references customers(CustomerID),
  Foreign key (AgentID) references salesagent(AgentID),
  Foreign key (BranchID) references branches(BranchID)
);
```

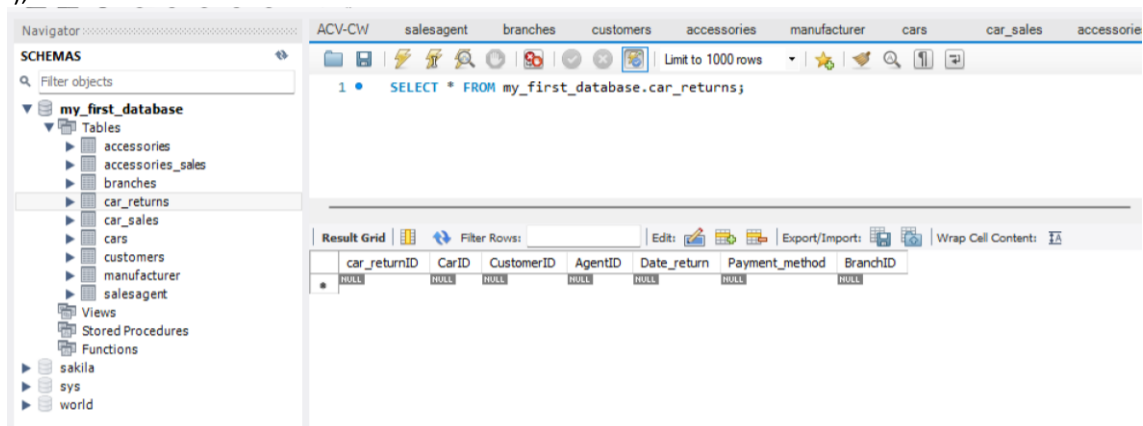


Figure 11. Screenshot of table Car Returns in MySQL Workbench

- **SQL Scripts to Populate the database-**

Figure 19 shows a screenshot of the MySQL Workbench environment with the script of this section. Then, I copied the segment of the script to populate each table, accompanied by the screenshot of the corresponding table.

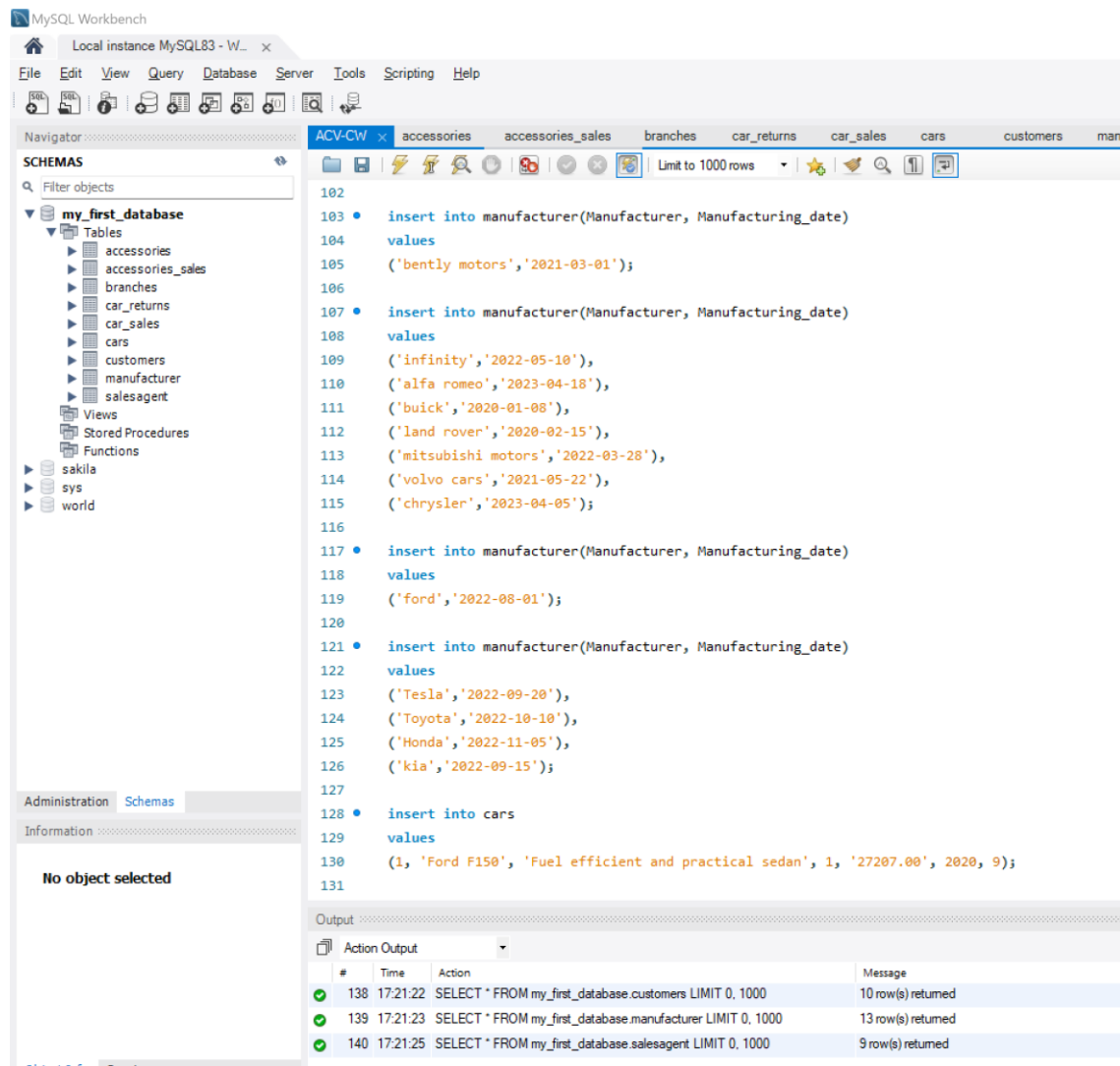


Figure 12. Screenshot of the script in MySQL workbench, populating the tables

## # script to populate table manufacturer

```

insert into manufacturer(Manufacturer, Manufacturing_date)
values
('bently motors','2021-03-01');

```

```

insert into manufacturer(Manufacturer, Manufacturing_date)
values
('infinity','2022-05-10'),
('alfa romeo','2023-04-18'),
('buick','2020-01-08'),
('land rover','2020-02-15'),
('mitsubishi motors','2022-03-28'),

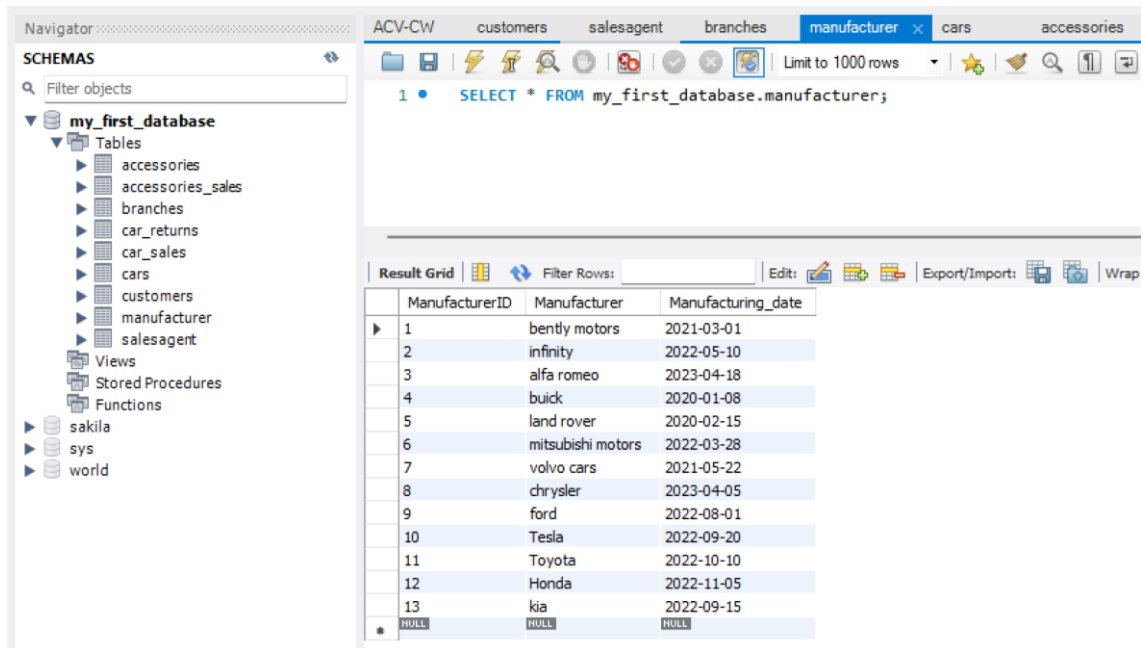
```



```
('volvo cars','2021-05-22'),
('chrysler','2023-04-05');
```

```
insert into manufacturer(Manufacturer, Manufacturing_date)
values
('ford','2022-08-01');
```

```
insert into manufacturer(Manufacturer, Manufacturing_date)
values
('Tesla','2022-09-20'),
('Toyota','2022-10-10'),
('Honda','2022-11-05'),
('kia','2022-09-15');
```



The screenshot shows a database interface with a 'Schemas' pane on the left and a 'Result Grid' on the right. The 'Result Grid' displays the data from the 'manufacturer' table. The data is as follows:

ManufacturerID	Manufacturer	Manufacturing_date
1	bently motors	2021-03-01
2	infinity	2022-05-10
3	alfa romeo	2023-04-18
4	buick	2020-01-08
5	land rover	2020-02-15
6	mitsubishi motors	2022-03-28
7	volvo cars	2021-05-22
8	chrysler	2023-04-05
9	ford	2022-08-01
10	Tesla	2022-09-20
11	Toyota	2022-10-10
12	Honda	2022-11-05
13	kia	2022-09-15
NULL	NULL	NULL

Figure 13. Screenshot of Manufacturers table with data

## # script to populate table cars

```
insert into cars
values
(1, 'Ford F150', 'Fuel efficient and practical sedan', 1, '27207.00', 2020, 9);
```

```
insert into cars(Model_number, Car_description, IsNew, Cost, ManufacturingYear, ManufacturerID)
values
('Tesla Model 3', 'Sleek and reliable sedan', 1, '22090.00', 2021, 10),
('Toyota Camry', 'Luxurious and high-tech electric car', 1, '33090.00', 2019, 11),
('Honda Accord', 'Safe and reliable SUV', 0, '20500.00', 2021, 12),
('Honda Accord', 'Fuel-efficient and practical sedan', 0, '22490.00', 2022, 12),
('Honda Accord', 'Spacious and comfortable family car', 0, '20200.00', 2020, 12),
('Toyota Camry', 'Fuel-efficient and practical sedan', 1, '42090.00', 2020, 11),
('Kia Telluride', 'Spacious and comfortable family car', 0, '22090.00', 2019, 13);
```

ACV-CW customers salesagent branches manufacturer accessories cars accessories\_sales

Limit to 1000 rows

1 • SELECT \* FROM my\_first\_database.cars;

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: I

CarID	Model_number	Car_description	IsNew	Cost	ManufacturingYear	ManufacturerID
1	Ford F150	Fuel efficient and practical sedan	1	27207.00	2020	9
2	Tesla Model 3	Sleek and reliable sedan	1	22090.00	2021	10
3	Toyota Camry	Luxurious and high-tech electric car	1	33090.00	2019	11
4	Honda Accord	Safe and reliable SUV	0	20500.00	2021	12
5	Honda Accord	Fuel-efficient and practical sedan	0	22490.00	2022	12
6	Honda Accord	Spacious and comfortable family car	0	20200.00	2020	12
7	Toyota Camry	Fuel-efficient and practical sedan	1	42090.00	2020	11
8	Kia Telluride	Spacious and comfortable family car	0	22090.00	2019	13
...	...	...	...	...	...	...

Figure 14. Screenshot of Cars table with data

## # script to populate table customers

insert into customers

values

(10001, 'Eric', 'Carl', '33a, Peckham Road, London, SE5 8BA');

insert into customers(FirstName, Surname, Address)

values

('Roald', 'Dahl', '147, The Quays, London, SE1 2LZ'),

('Quentin', 'Blake', '242, Deptford High Street, London, SE8 5DH'),

('Tim', 'Minchin', '173, Lewisham High Street, London, SE13 6JN'),

('Kate', 'Pankhurst', '27, The Cut, London, SE1 8LF'),

('Michael', 'Rosen', '12, Creek Road, Greenwich, London, SE8 3RJ'),

('Julia', 'Donaldson', '107, Queen Elizabeth Street, London, SE1 9NE'),

('Lynley', 'Dodd', '27, Royal Hill, Greenwich, London, SE10 8RF'),

('Oliver', 'Jeffers', '152, Stansted Road, London, SE23 1EW'),

('Jill', 'Murphy', '41, Blackheath Hill, London, SE10 8DJ');

ACV-CW customers salesagent manufacturer cars accessories accessories\_sales

Limit to 1000 rows

1 • SELECT \* FROM my\_first\_database.customers;

CustomerID	FirstName	Surname	Address
10001	Eric	Carl	33a, Peckham Road, London, SE5 8BA
10002	Roald	Dahl	147, The Quays, London, SE1 2LZ
10003	Quentin	Blake	242, Deptford High Street, London, SE8 5DH
10004	Tim	Minchin	173, Lewisham High Street, London, SE13 6JN
10005	Kate	Pankhurst	27, The Cut, London, SE1 8LF
10006	Michael	Rosen	12, Creek Road, Greenwich, London, SE8 3RJ
10007	Julia	Donaldson	107, Queen Elizabeth Street, London, SE1 9NE
10008	Lynley	Dodd	27, Royal Hill, Greenwich, London, SE10 8RF
10009	Oliver	Jeffers	152, Stansted Road, London, SE23 1EW
10010	Jill	Murphy	41, Blackheath Hill, London, SE10 8DJ
NULL	NULL	NULL	NULL

Figure 15. Screenshot of Customers table with data

## # script to populate table salesagent

insert into salesagent

values

(001, 'Harry', 'Potter', 0.03);

insert into salesagent(FirstName, Surname, Commission)

values

('Mildred', 'Hubble', 0.04),

('Matilda', 'Wormwood', 0.05),

('Danny', 'Williams', 0.04),

('Willy', 'Wonka', 0.03),

('Sophie', 'Giant', 0.04),

('Maisie', 'Mouse', 0.03),

('Harry', 'Caterpillar', 0.03),

('Elmer', 'Elephant', 0.04);

ACV-CW customers salesagent manufacturer cars accessories accessories\_sales

1 • `SELECT * FROM my_first_database.salesagent;`

AgentID	FirstName	Surname	Commission
1	Harry	Potter	0.03
2	Mildred	Hubble	0.04
3	Matilda	Wormwood	0.05
4	Danny	Williams	0.04
5	Willy	Wonka	0.03
6	Sophie	Giant	0.04
7	Maisie	Mouse	0.03
8	Harry	Caterpillar	0.03
9	Elmer	Elephant	0.04
*	NULL	NULL	NULL

Figure 16. Screenshot of sales agents table with data

## # script to populate table branches

insert into branches

values

(10, '21 Rolt Street, Deptford, London SE8 4NF');

insert into branches(Branch\_location)

values

('N Woolwich Rd, London, E16 2HP'),

('1 Midnight Ave, London, SE5 0SE'),

('173, Lewisham High Street, London, SE13 6JN'),

('27, The Cut, London, SE1 8LF'),

('31 Elmira Rd, London, "SE13 7DW');

ACV-CW customers salesagent branches manufacturer cars accessories

1 • `SELECT * FROM my_first_database.branches;`

BranchID	Branch_location
10	21 Rolt Street, Deptford, London SE8 4NF
11	N Woolwich Rd, London, E16 2HP
12	1 Midnight Ave, London, SE5 0SE
13	173, Lewisham High Street, London, SE13 6JN
14	27, The Cut, London, SE1 8LF
15	31 Elmira Rd, London, "SE13 7DW
*	NULL

Figure 17. Screenshot of branches table with data

## # script to populate table accessories

insert into accessories

values

(1000, 'Phone mount', 'It attaches to your car's air vent or dashboard and is adjustable to fit most phones',  
'Anker', 12.99);

insert into accessories(Accessories\_name, Accessories\_description, Accessories\_brand, Accessories\_cost)

values

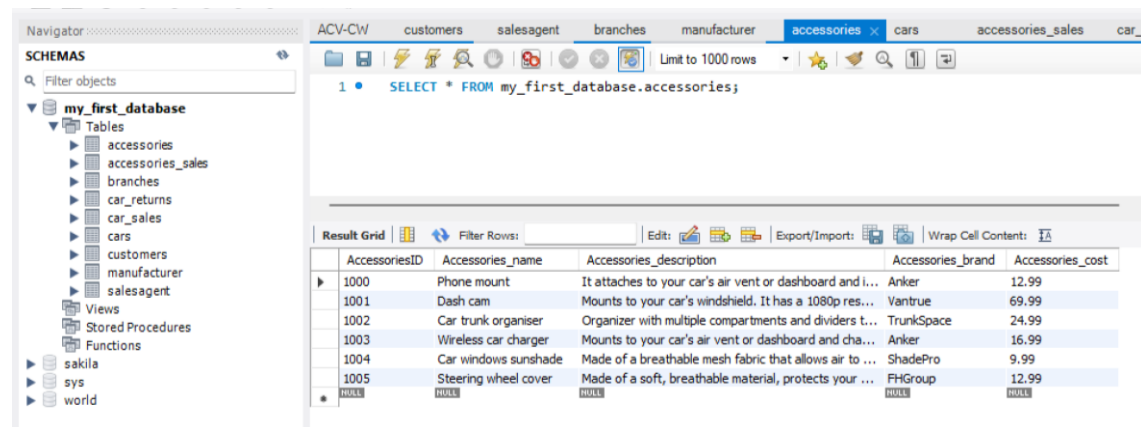
('Dash cam', 'Mounts to your car's windshield. It has a 1080p resolution and a wide field of view', 'Vantrue',  
69.99),

('Car trunk organiser', 'Organizer with multiple compartments and dividers to keep everything neat and tidy',  
'TrunkSpace', 24.99),

('Wireless car charger', 'Mounts to your car's air vent or dashboard and charges your phone through its  
case.', 'Anker', 16.99),

('Car windows sunshade', 'Made of a breathable mesh fabric that allows air to circulate while still blocking the  
sun.', 'ShadePro', 9.99),

('Steering wheel cover', 'Made of a soft, breathable material, protects your steering wheel from wear and  
tear', 'FHGroup', 12.99 );



The screenshot shows a database management interface with a 'Schemas' pane on the left and a 'Result Grid' on the right. The 'Result Grid' displays the contents of the 'accessories' table. The table has five columns: AccessoriesID, Accessories\_name, Accessories\_description, Accessories\_brand, and Accessories\_cost. The data is as follows:

AccessoriesID	Accessories_name	Accessories_description	Accessories_brand	Accessories_cost
1000	Phone mount	It attaches to your car's air vent or dashboard and is adjustable to fit most phones	Anker	12.99
1001	Dash cam	Mounts to your car's windshield. It has a 1080p resolution and a wide field of view	Vantrue	69.99
1002	Car trunk organiser	Organizer with multiple compartments and dividers to keep everything neat and tidy	TrunkSpace	24.99
1003	Wireless car charger	Mounts to your car's air vent or dashboard and charges your phone through its case.	Anker	16.99
1004	Car windows sunshade	Made of a breathable mesh fabric that allows air to circulate while still blocking the sun.	ShadePro	9.99
1005	Steering wheel cover	Made of a soft, breathable material, protects your steering wheel from wear and tear	FHGroup	12.99

Figure 18. Screenshot of accessories table with data

## # script to populate table car\_sales

insert into car\_sales

values

(240000, 2, 10003, 004, '2024-01-22', 'Credit', 11);

insert into car\_sales(CarID, CustomerID, AgentID, Date\_Purchase, Payment\_method, BranchID)

values

(3, 10004, 005, '2024-01-23', 'Bank Transfer', 12),

(4, 10005, 003, '2024-02-24', 'Lease', 13),

(5, 10002, 002, '2024-03-25', 'Trade-in', 14);

ACV-CW customers salesagent branches manufacturer accessories **car\_sales** cars

1 • `SELECT * FROM my_first_database.car_sales;`

Limit to 1000 rows

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

Car_salesID	CarID	CustomerID	AgentID	Date_Purchase	Payment_method	BranchID
240000	2	10003	4	2024-01-22	Credit	11
240001	3	10004	5	2024-01-23	Bank Transfer	12
240002	4	10005	3	2024-02-24	Lease	13
240003	5	10002	2	2024-03-25	Trade-in	14
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 19. Screenshot of Car sales table with data

### # script to populate table accessories\_sales

insert into accessories\_sales

values

(2400000, 1000, 10001, 001, '2024-02-22', 'Bank Transfer', 10);

insert into accessories\_sales(AccessoriesID, CustomerID, AgentID, Date\_Purchase, Payment\_method, BranchID)

values

(1001, 10010, 009, '2024-02-23', 'Bank Transfer', 11),

(1002, 10009, 008, '2024-03-24', 'Cash', 12),

(1003, 10008, 007, '2024-03-25', 'Credit', 13);

ACV-CW customers salesagent branches manufacturer accessories **accessories\_sales** car\_sa

1 • `SELECT * FROM my_first_database.accessories_sales;`

Limit to 1000 rows

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

Accessories_salesID	AccessoriesID	CustomerID	AgentID	Date_Purchase	Payment_method	BranchID
2400000	1000	10001	1	2024-02-22	Bank Transfer	10
2400001	1001	10010	9	2024-02-23	Bank Transfer	11
2400002	1002	10009	8	2024-03-24	Cash	12
2400003	1003	10008	7	2024-03-25	Credit	13
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 20. Screenshot of Accessories sales table with data

### # script to populate table car\_returns

insert into car\_returns

values

(24000, 6, 10008, 007, '2022-11-15', 'Trade-in', 10);

insert into car\_returns(CarID, CustomerID, AgentID, Date\_return, Payment\_method, BranchID)

values

(5, 10007, 006, '2022-11-16', 'Trade-in', 11),

(4, 10006, 005, '2022-11-17', 'Trade-in', 12);

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'SCHEMAS' pane displays a tree view of the 'my\_first\_database' schema, including tables like 'accessories', 'accessories\_sales', 'branches', 'car\_returns', 'car\_sales', 'cars', 'customers', 'manufacturer', 'salesagent', 'Views', 'Stored Procedures', and 'Functions'. The 'car\_returns' table is selected. The main pane shows the SQL query: `SELECT * FROM my_first_database.car_returns;` and the 'Result Grid' displaying the data.

car_returnID	CarID	CustomerID	AgentID	Date_return	Payment_method	BranchID
24000	6	10008	7	2022-11-15	Trade-in	10
24001	5	10007	6	2022-11-16	Trade-in	11
24002	4	10006	5	2022-11-17	Trade-in	12
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 21. Screenshot of Car returns table with data

## 2. Demonstration of functionality.

- SQL Scripts to manipulate the database

# To see all the registers in the car\_sales table:

select \* from car\_sales;

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'SCHEMAS' pane displays a tree view of the 'my\_first\_database' schema, including tables like 'accessories', 'accessories\_sales', 'branches', 'car\_returns', 'car\_sales', 'cars', 'customers', 'manufacturer', 'salesagent', 'Views', 'Stored Procedures', and 'Functions'. The 'car\_sales' table is selected. The main pane shows the SQL query: `select * from car_sales;` and the 'Result Grid' displaying the data.

Car_salesID	CarID	CustomerID	AgentID	Date_Purchase	Payment_method	BranchID
240000	2	10003	4	2024-01-22	Credit	11
240001	3	10004	5	2024-01-23	Bank Transfer	12
240002	4	10005	3	2024-02-24	Lease	13
240003	5	10002	2	2024-03-25	Trade-in	14
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 22. Screenshot of query select and output.

# To show in the cars table all the cars manufactured in the year 2020

select \* from cars where ManufacturingYear=2020;

The screenshot shows a database query tool interface. On the left, a 'SCHEMAS' pane lists a database named 'my\_first\_database' with various tables including 'cars'. The main query editor displays the following SQL query:

```
230  
231  
232 # To show in the cars table all the cars manufactured in the year 2020  
233  
234 • select * from cars where ManufacturingYear=2020;  
235
```

Below the query editor, the 'Result Grid' shows the output of the query. The grid has columns: CarID, Model\_number, Car\_description, IsNew, Cost, ManufacturingYear, and ManufacturerID. The results are as follows:

CarID	Model_number	Car_description	IsNew	Cost	ManufacturingYear	ManufacturerID
1	Ford F150	Fuel efficient and practical sedan	1	27207.00	2020	9
6	Honda Accord	Spacious and comfortable family car	0	20200.00	2020	12
7	Toyota Camry	Fuel-efficient and practical sedan	1	42090.00	2020	11
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 23. Screenshot of query select... where and output.

# To show all the sales agents sorted by surname in ascending order:

select \* from salesagent  
order by Surname asc;

The screenshot shows the same database query tool interface. The 'SCHEMAS' pane is visible on the left. The main query editor displays the following SQL query:

```
236  
237 # To show all the sales agents sorted by surname in ascending order:  
238  
239 • select * from salesagent  
240 order by Surname asc;  
241
```

Below the query editor, the 'Result Grid' shows the output of the query. The grid has columns: AgentID, FirstName, Surname, and Commission. The results are sorted by Surname in ascending order:

AgentID	FirstName	Surname	Commission
8	Harry	Caterpillar	0.03
9	Elmer	Elephant	0.04
6	Sophie	Giant	0.04
2	Mildred	Hubble	0.04
7	Maisie	Mouse	0.03
1	Harry	Potter	0.03
4	Danny	Williams	0.04
5	Willy	Wonka	0.03
3	Matilda	Wormwood	0.05
*	NULL	NULL	NULL

Figure 24. Screenshot of query select... order by and output.



- **SQL Codes to join two or more table and output screens**

# To show the names of the customers from the accessories\_sales table, and the accessories they bought

```
select Accessories_name, FirstName, Surname
from customers inner join accessories_sales
on customers.CustomerID = accessories_sales.CustomerID
inner join accessories
on accessories_sales.AccessoriesID = accessories.AccessoriesID;
```

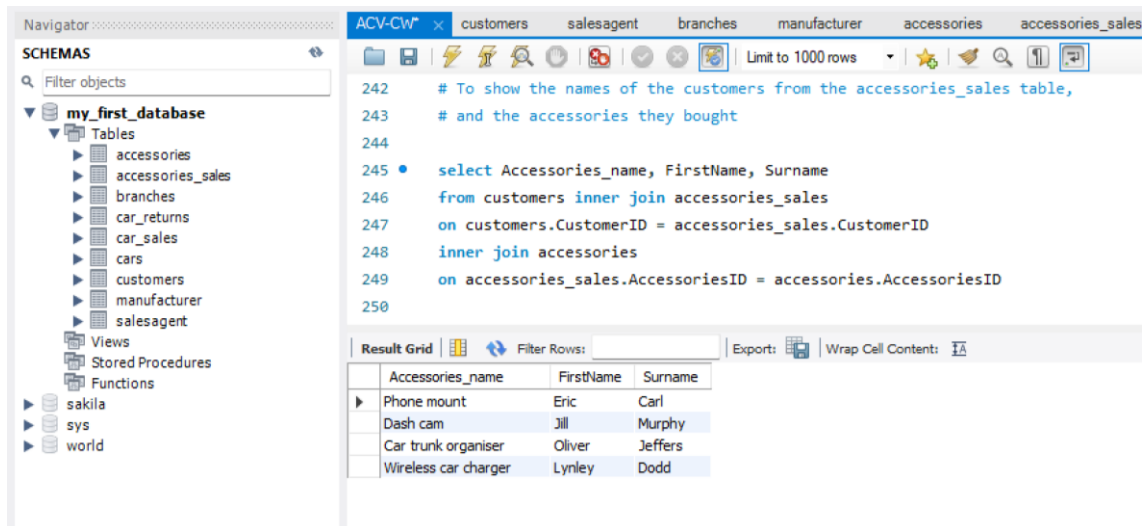


Figure 25. Screenshot of query with inner join and output.

# To show the information of the cars that appear in both the car\_returns table and the car\_sales table

```
select * from car_returns inner join car_sales
on car_returns.CarID = car_sales.CarID;
```

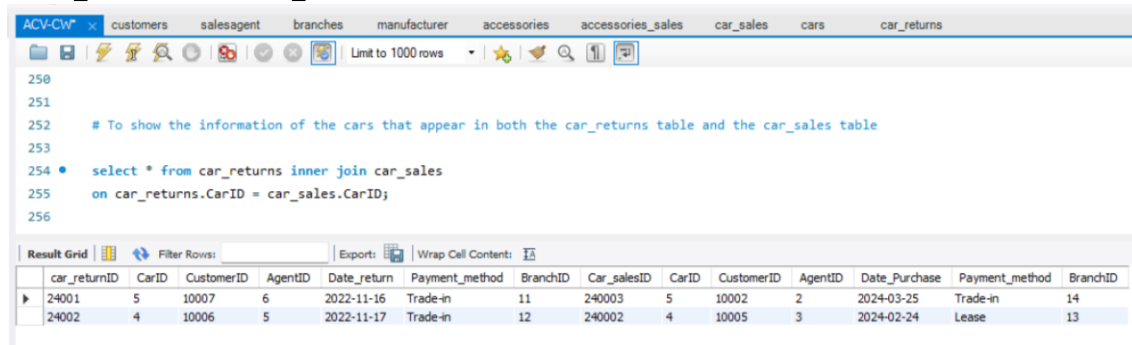


Figure 26. Screenshot of query with inner join and output.