DBMS – **Order rerouting system**

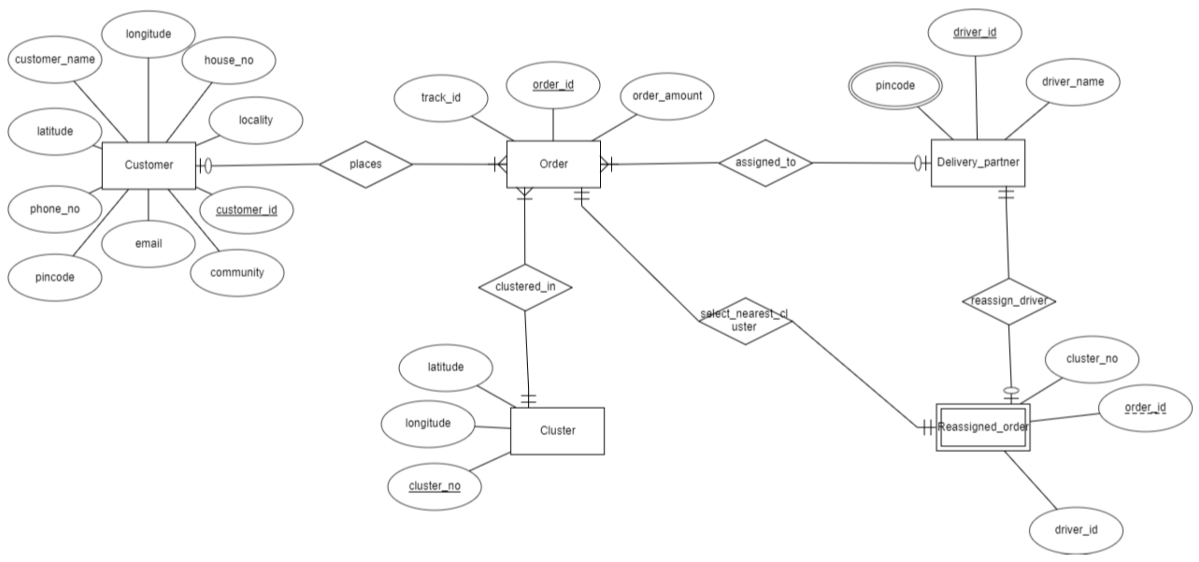
Submitted By:
Name : Siddharth P
SRN : PES1UG20CS427
V Semester Section G

Short Description and Scope of the Project

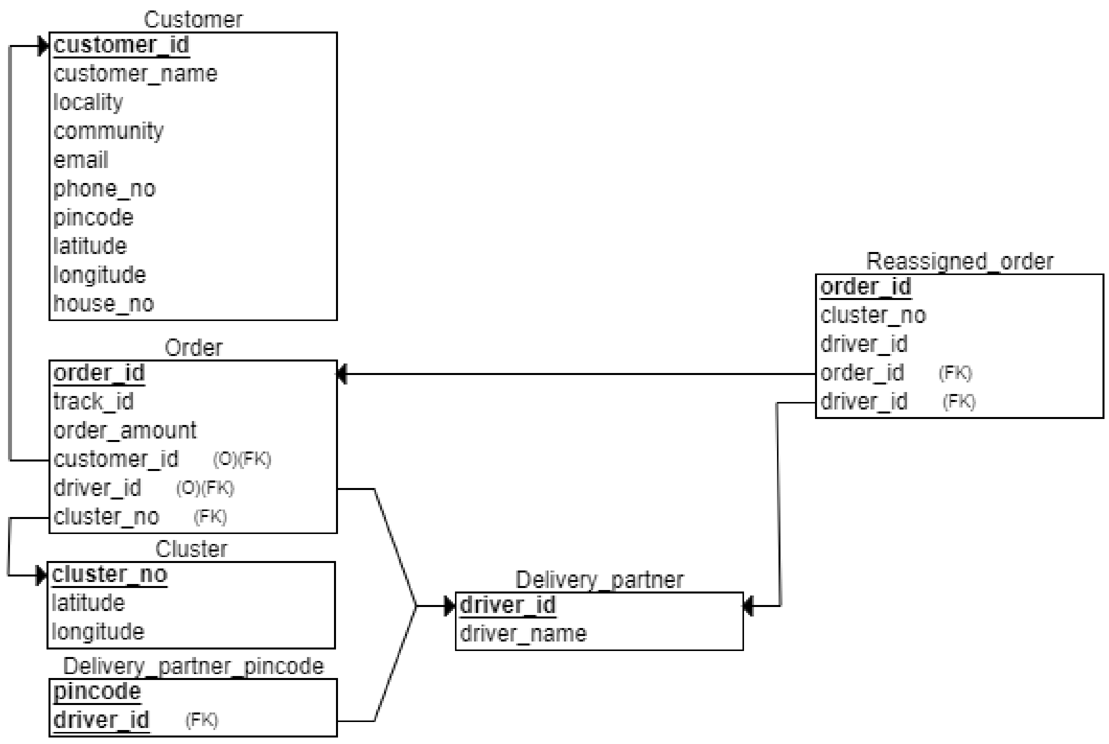
Order rerouting system for a day's worth of orders for an E-selling company. The orders are uploaded to the server along with the customer details. There is pre-existing information given on which locations a driver is mapped to. This is given in terms of pin code to driver mapping. Thus, each order is mapped to a specific driver on insertion.

The project, on given this data, makes use of machine learning algorithms and distance-based clustering algorithms in order to upload cluster information, and details of which driver each order can be reassigned to. The admin then has an option to reassign the order based on a given information. Therefore, if there is any issue with a specific driver, or if he is carrying too many orders, specific orders can then be reassigned to the most convenient driver

ER Diagram

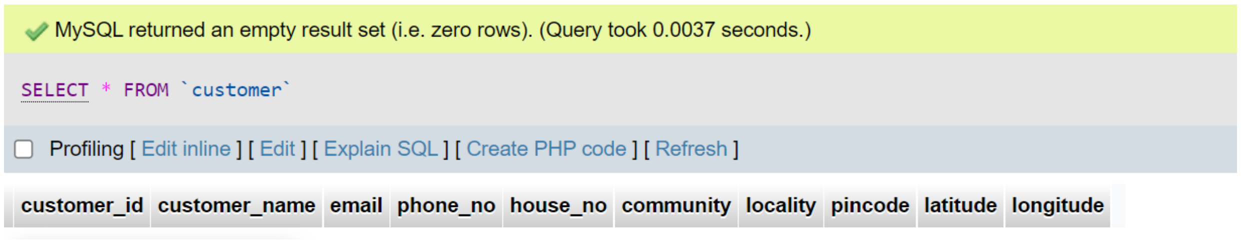

longitude
customer_name
house_no
latitude
locality
Customer
phone_no
customer_id
email
pincode
community
places
track_id
order_id
order_amount
Order
clustered_in
latitude
longitude
Cluster
cluster_no
select_nearest_cl
uster
assigned_to
driver_id
pincode
driver_name
Delivery_partner
reassign_driver
cluster_no
order_id
Reassigned_order
driver_id

Relational Schema



DDL statements - Building the database

```
CREATE TABLE Customer
(
  customer_id INT NOT NULL,
  customer_name varchar(30),
  email varchar(50),
  phone_no varchar(20),
  house_no varchar(50),
  community varchar(50),
  locality varchar(50),
  pincode INT ,
  latitude DECIMAL(8,5),
  longitude DECIMAL(8,5),
  PRIMARY KEY (customer_id)
);
```

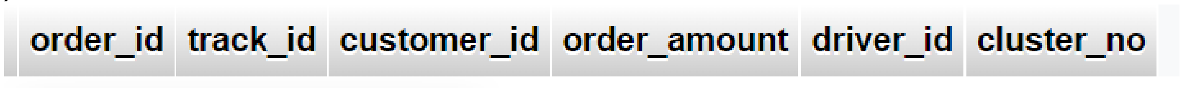


```
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0037 seconds.)

SELECT * FROM `customer`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

customer_id	customer_name	email	phone_no	house_no	community	locality	pincode	latitude	longitude

```
CREATE TABLE orders
(
  order_id INT NOT NULL,
  track_id DECIMAL(5,2),
  customer_id INT,
  order_amount DECIMAL(7,2),
  cluster_no INT,
  driver_id INT,
  PRIMARY KEY (order_id),
  FOREIGN KEY (customer_id) REFERENCES Customer(customer_id),
  FOREIGN KEY (driver_id) REFERENCES Delivery_partner(driver_id),
  FOREIGN KEY (cluster_no) REFERENCES Cluster(cluster_no)
);
```



order_id	track_id	customer_id	order_amount	driver_id	cluster_no

```
CREATE TABLE Cluster
(
  cluster_no INT NOT NULL,
  cluster_latitude DECIMAL(5,2),
  cluster_longitude DECIMAL(5,2),
  cluster_community VARCHAR(50),
  PRIMARY KEY(cluster_no)
);
CREATE TABLE Delivery_partner_pincode
(
  pincode INT NOT NULL,
  driver_id INT NOT NULL,
  PRIMARY KEY (pincode, driver_id),
  FOREIGN KEY (driver_id) REFERENCES Delivery_partner(driver_id)
);
CREATE TABLE Reassigned_order
(
  order_id INT NOT NULL,
  cluster_no INT,
  driver_id INT,
  PRIMARY KEY (order_id),
  FOREIGN KEY (order_id) REFERENCES Orders(order_id),
  FOREIGN KEY (driver_id) REFERENCES Delivery_partner(driver_id)
);
```

Populating the Database

Individually

insert into customer values(1,'RIJAS E','rijas.ebrahim@gmail.com','9845219894','Flat 111','Parijatha Apartments','Hongasandra',560068,12.89073,77.62694);

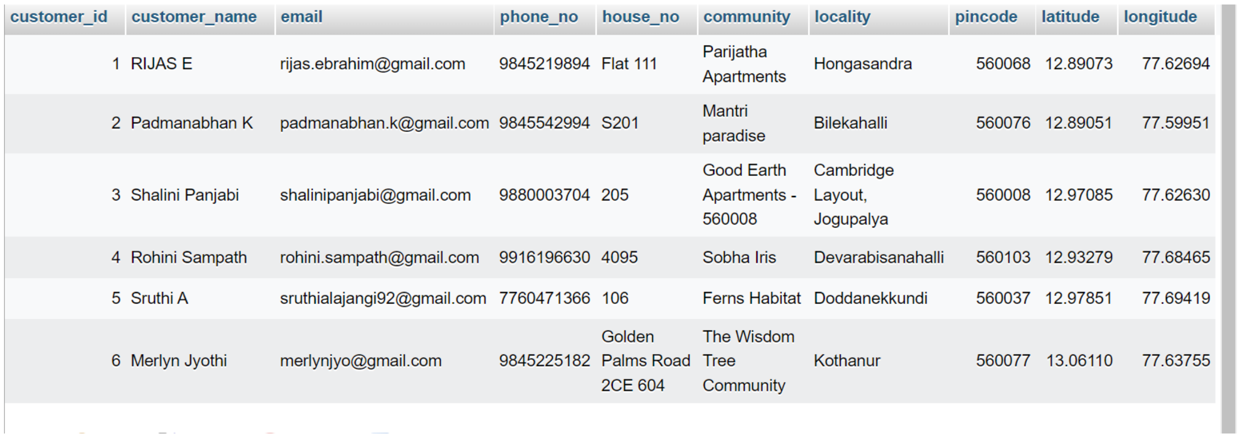
insert into customer values(2,'Padmanabhan K','padmanabhan.k@gmail.com','9845542994','S201','Mantri paradise','Bilekahalli',560076,12.89051,77.59951);

insert into customer values(3,'Shalini Panjabi','shalinipanjabi@gmail.com','9880003704','205','Good Earth Apartments - 560008','Cambridge Layout, Jogupalya',560008,12.970847,77.626299);

insert into customer values(4,'Rohini Sampath','rohini.sampath@gmail.com','9916196630','4095','Sobha Iris','Devarabisanahalli',560103,12.932789,77.684653);

insert into customer values(5,'Sruthi A','sruthialajangi92@gmail.com','7760471366','106','Ferns Habitat','Doddanekkundi',560037,12.978514,77.694189);

insert into customer values(6,'Merlyn Jyothi','merlynjyo@gmail.com','9845225182','Golden Palms Road 2CE 604','The Wisdom Tree Community','Kothanur',560077,13.0610954,77.637551417);



customer_id	customer_name	email	phone_no	house_no	community	locality	pincode	latitude	longitude
1	RIJAS E	rijas.ebrahim@gmail.com	9845219894	Flat 111	Parijatha Apartments	Hongasandra	560068	12.89073	77.62694
2	Padmanabhan K	padmanabhan.k@gmail.com	9845542994	S201	Mantri paradise	Bilekahalli	560076	12.89051	77.59951
3	Shalini Panjabi	shalinipanjabi@gmail.com	9880003704	205	Good Earth Apartments - 560008	Cambridge Layout, Jogupalya	560008	12.97085	77.62630
4	Rohini Sampath	rohini.sampath@gmail.com	9916196630	4095	Sobha Iris	Devarabisanahalli	560103	12.93279	77.68465
5	Sruthi A	sruthialajangi92@gmail.com	7760471366	106	Ferns Habitat	Doddanekkundi	560037	12.97851	77.69419
6	Merlyn Jyothi	merlynjyo@gmail.com	9845225182	Golden Palms Road 2CE 604	The Wisdom Tree Community	Kothanur	560077	13.06110	77.63755

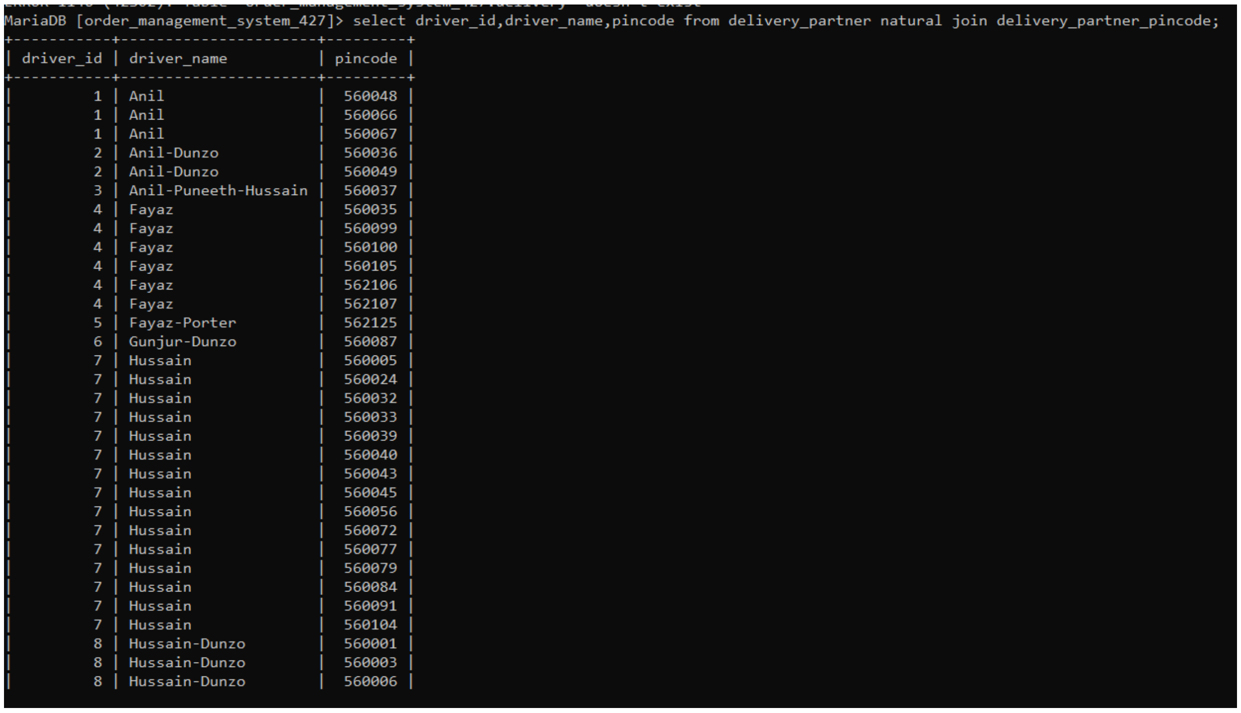
BY LOADING A FILE BY COMMAND

```
LOAD DATA INFILE "Customers.csv" into TABLE customer
COLUMNS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
ESCAPED BY '"'
LINES TERMINATED BY '\n';
```

Rest of data loaded by import csv option on phpMyAdmin server.

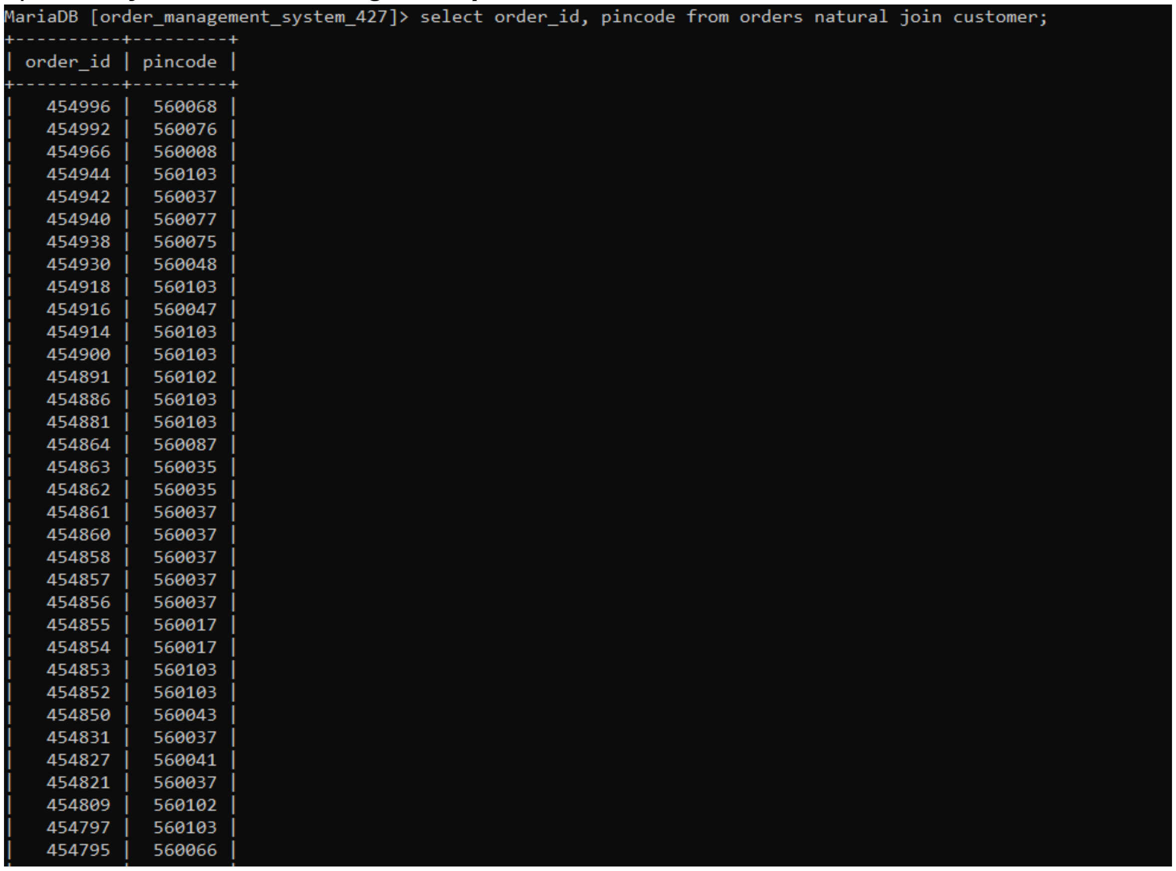
Join Queries

Showcase at least 4 join queries
Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results
1)**Delivery_partner join delivary_partner_pincode to map pincode to driver_id**



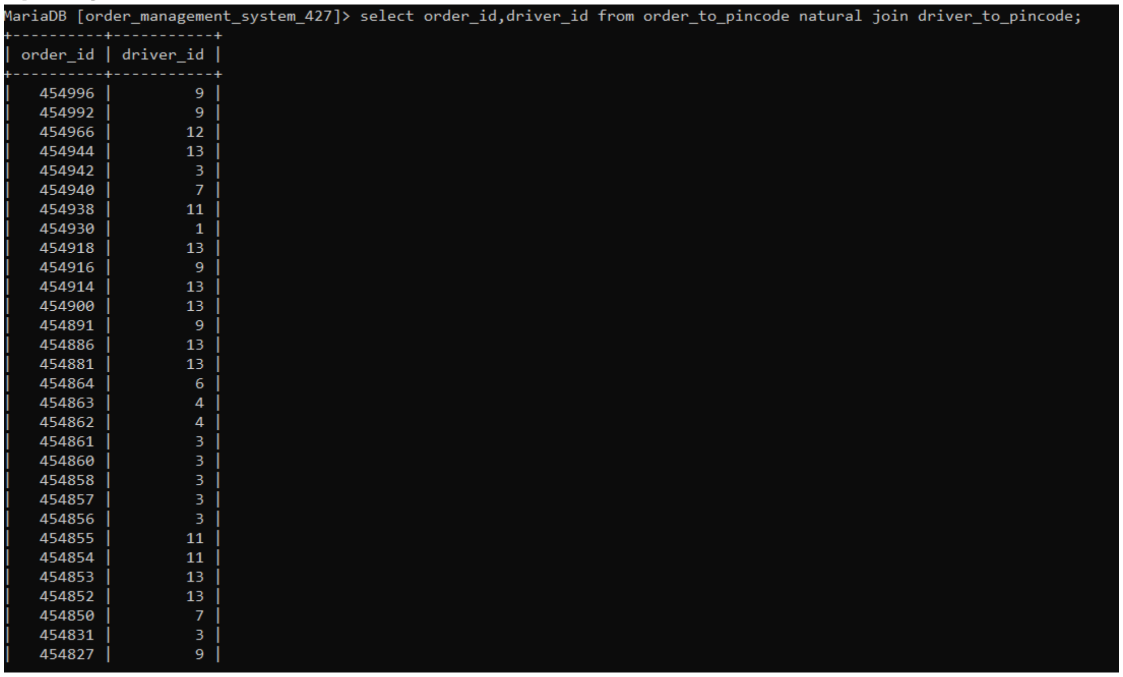
```
MariaDB [order_management_system_427]> select driver_id,driver_name,pincode from delivery_partner natural join delivery_partner_pincode;
+-----------+---------------------+---------+
| driver_id | driver_name         | pincode |
+-----------+---------------------+---------+
|         1 | Anil                |  560048 |
|         1 | Anil                |  560066 |
|         1 | Anil                |  560067 |
|         2 | Anil-Dunzo          |  560036 |
|         2 | Anil-Dunzo          |  560049 |
|         3 | Anil-Puneeth-Hussain |  560037 |
|         4 | Fayaz               |  560035 |
|         4 | Fayaz               |  560099 |
|         4 | Fayaz               |  560100 |
|         4 | Fayaz               |  560105 |
|         4 | Fayaz               |  562106 |
|         4 | Fayaz               |  562107 |
|         5 | Fayaz-Porter        |  562125 |
|         6 | Gunjur-Dunzo        |  560087 |
|         7 | Hussain             |  560005 |
|         7 | Hussain             |  560024 |
|         7 | Hussain             |  560032 |
|         7 | Hussain             |  560033 |
|         7 | Hussain             |  560039 |
|         7 | Hussain             |  560040 |
|         7 | Hussain             |  560043 |
|         7 | Hussain             |  560045 |
|         7 | Hussain             |  560056 |
|         7 | Hussain             |  560072 |
|         7 | Hussain             |  560077 |
|         7 | Hussain             |  560079 |
|         7 | Hussain             |  560084 |
|         7 | Hussain             |  560091 |
|         7 | Hussain             |  560104 |
|         8 | Hussain-Dunzo       |  560001 |
|         8 | Hussain-Dunzo       |  560003 |
|         8 | Hussain-Dunzo       |  560006 |
```

2)**orders join customer to get the pincodes of each order**



```
MariaDB [order_management_system_427]> select order_id, pincode from orders natural join customer;
+----------+---------+
| order_id | pincode |
+----------+---------+
|   454996 |  560068 |
|   454992 |  560076 |
|   454966 |  560008 |
|   454944 |  560103 |
|   454942 |  560037 |
|   454940 |  560077 |
|   454938 |  560075 |
|   454930 |  560048 |
|   454918 |  560103 |
|   454916 |  560047 |
|   454914 |  560103 |
|   454900 |  560103 |
|   454891 |  560102 |
|   454886 |  560103 |
|   454881 |  560103 |
|   454864 |  560087 |
|   454863 |  560035 |
|   454862 |  560035 |
|   454861 |  560037 |
|   454860 |  560037 |
|   454858 |  560037 |
|   454857 |  560037 |
|   454856 |  560037 |
|   454855 |  560017 |
|   454854 |  560017 |
|   454853 |  560103 |
|   454852 |  560103 |
|   454850 |  560043 |
|   454831 |  560037 |
|   454827 |  560041 |
|   454821 |  560037 |
|   454809 |  560102 |
|   454797 |  560103 |
|   454795 |  560066 |
```

3) To get the driver id of each order



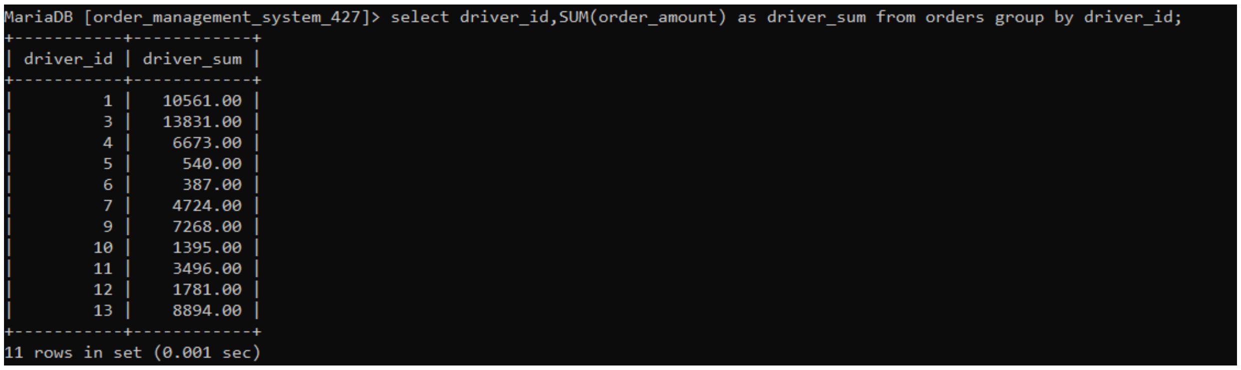
```
MariaDB [order_management_system_427]> select order_id,driver_id from order_to_pincode natural join driver_to_pincode;
+----------+-----------+
| order_id | driver_id |
+----------+-----------+
|   454996 |         9 |
|   454992 |         9 |
|   454966 |        12 |
|   454944 |        13 |
|   454942 |         3 |
|   454940 |         7 |
|   454938 |        11 |
|   454930 |         1 |
|   454918 |        13 |
|   454916 |         9 |
|   454914 |        13 |
|   454900 |        13 |
|   454891 |         9 |
|   454886 |        13 |
|   454881 |        13 |
|   454864 |         6 |
|   454863 |         4 |
|   454862 |         4 |
|   454861 |         3 |
|   454860 |         3 |
|   454858 |         3 |
|   454857 |         3 |
|   454856 |         3 |
|   454855 |        11 |
|   454854 |        11 |
|   454853 |        13 |
|   454852 |        13 |
|   454850 |         7 |
|   454831 |         3 |
|   454827 |         9 |
```

These joins are performed in order to perform driver to order mapping to assign a driver to an order based on the location of each order and which location a driver is assigned.

Aggregate Functions

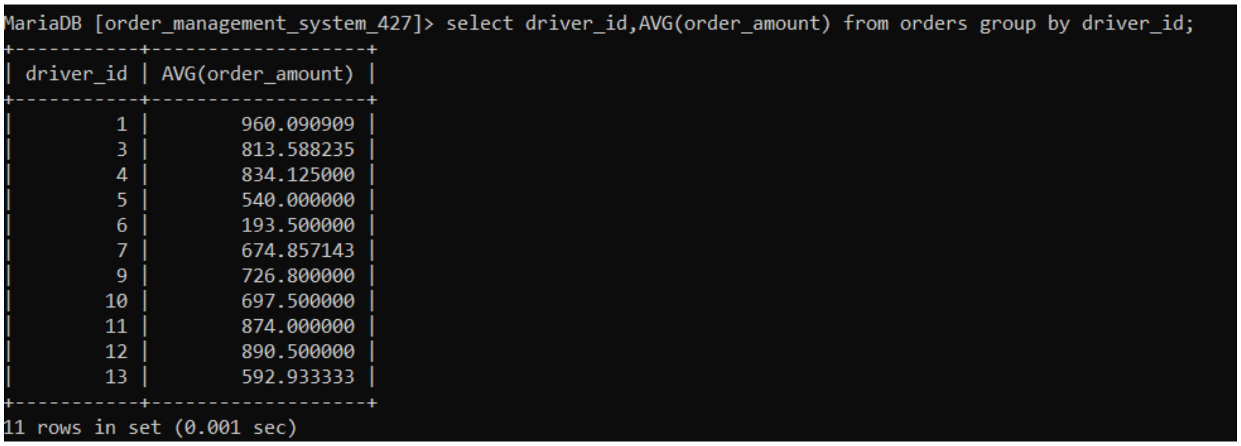
Showcase at least 4 Aggregate function queries
Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

1) **To get the sum of order amounts which each driver is carrying**



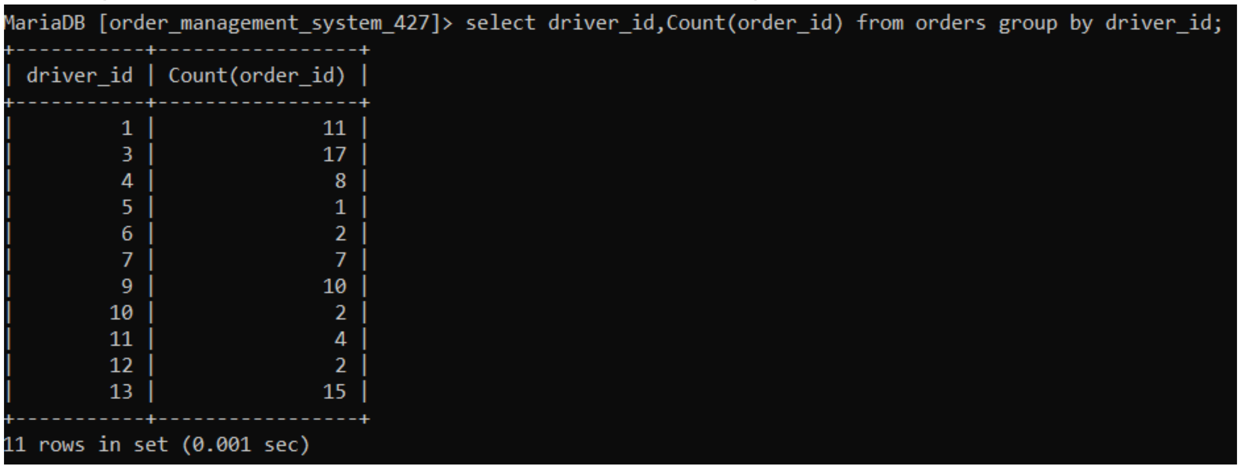
```
MariaDB [order_management_system_427]> select driver_id,SUM(order_amount) as driver_sum from orders group by driver_id;
+-----------+------------+
| driver_id | driver_sum |
+-----------+------------+
|         1 |   10561.00 |
|         3 |   13831.00 |
|         4 |    6673.00 |
|         5 |     540.00 |
|         6 |     387.00 |
|         7 |    4724.00 |
|         9 |    7268.00 |
|        10 |    1395.00 |
|        11 |    3496.00 |
|        12 |    1781.00 |
|        13 |    8894.00 |
+-----------+------------+
11 rows in set (0.001 sec)
```

2) **To get the average of orders which each driver is carrying**



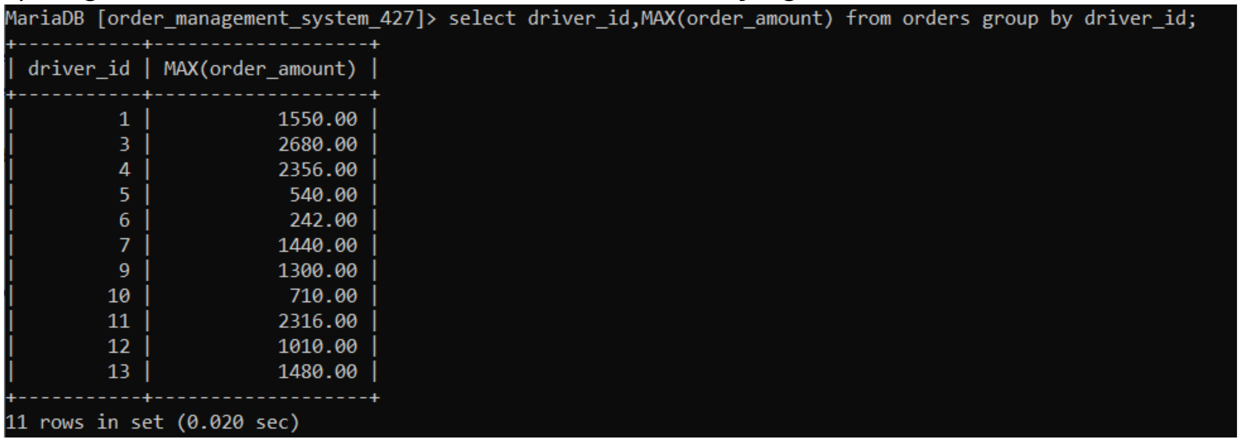
```
MariaDB [order_management_system_427]> select driver_id,AVG(order_amount) from orders group by driver_id;
+-----------+-------------------+
| driver_id | AVG(order_amount) |
+-----------+-------------------+
|         1 |        960.090909 |
|         3 |        813.588235 |
|         4 |        834.125000 |
|         5 |        540.000000 |
|         6 |        193.500000 |
|         7 |        674.857143 |
|         9 |        726.800000 |
|        10 |        697.500000 |
|        11 |        874.000000 |
|        12 |        890.500000 |
|        13 |        592.933333 |
+-----------+-------------------+
11 rows in set (0.001 sec)
```

3) **To get the number of orders each driver is carrying**



```
MariaDB [order_management_system_427]> select driver_id,Count(order_id) from orders group by driver_id;
+-----------+-----------------+
| driver_id | Count(order_id) |
+-----------+-----------------+
|         1 |              11 |
|         3 |              17 |
|         4 |               8 |
|         5 |               1 |
|         6 |               2 |
|         7 |               7 |
|         9 |              10 |
|        10 |               2 |
|        11 |               4 |
|        12 |               2 |
|        13 |              15 |
+-----------+-----------------+
11 rows in set (0.001 sec)
```

4) **To get the maximum amount which a driver is carrying**

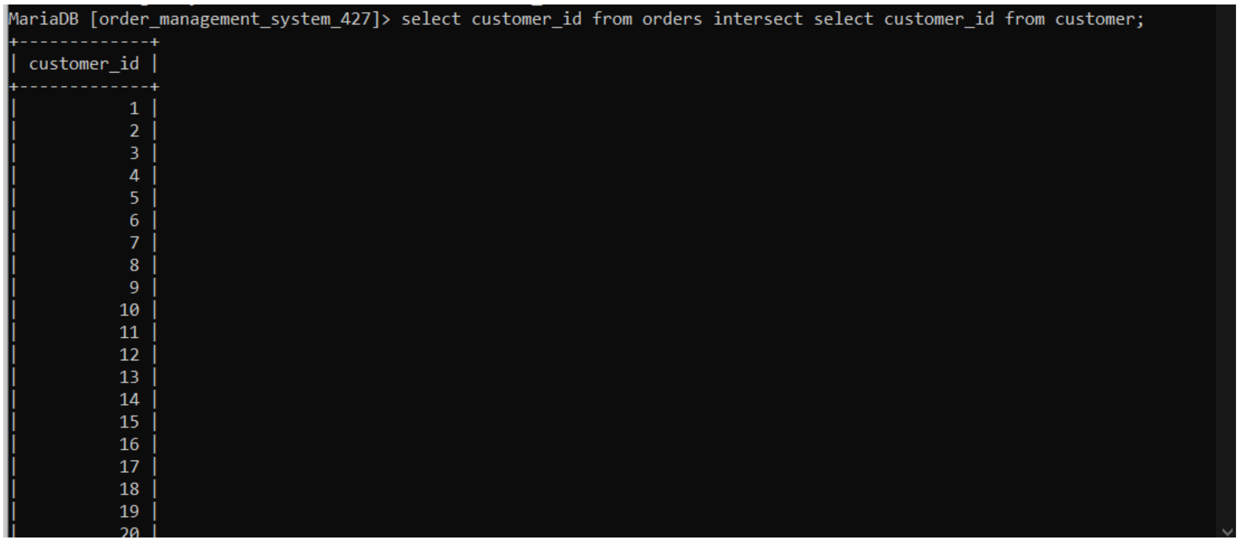


```
MariaDB [order_management_system_427]> select driver_id,MAX(order_amount) from orders group by driver_id;
+-----------+-------------------+
| driver_id | MAX(order_amount) |
+-----------+-------------------+
|         1 |           1550.00 |
|         3 |           2680.00 |
|         4 |           2356.00 |
|         5 |            540.00 |
|         6 |            242.00 |
|         7 |           1440.00 |
|         9 |           1300.00 |
|        10 |            710.00 |
|        11 |           2316.00 |
|        12 |           1010.00 |
|        13 |           1480.00 |
+-----------+-------------------+
11 rows in set (0.020 sec)
```

Set Operations

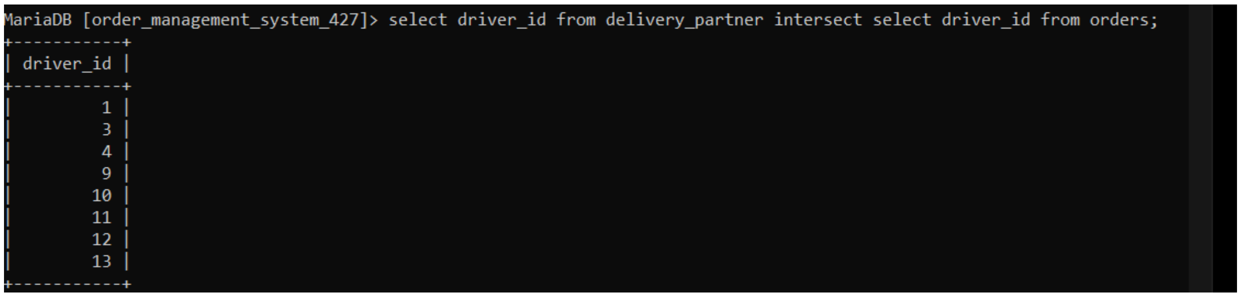
Showcase at least 4 Set Operations queries
Write the query in English Language, Show the equivalent SQL statement and also a screenshot of the query and the results

1)**Select customers who placed an order**



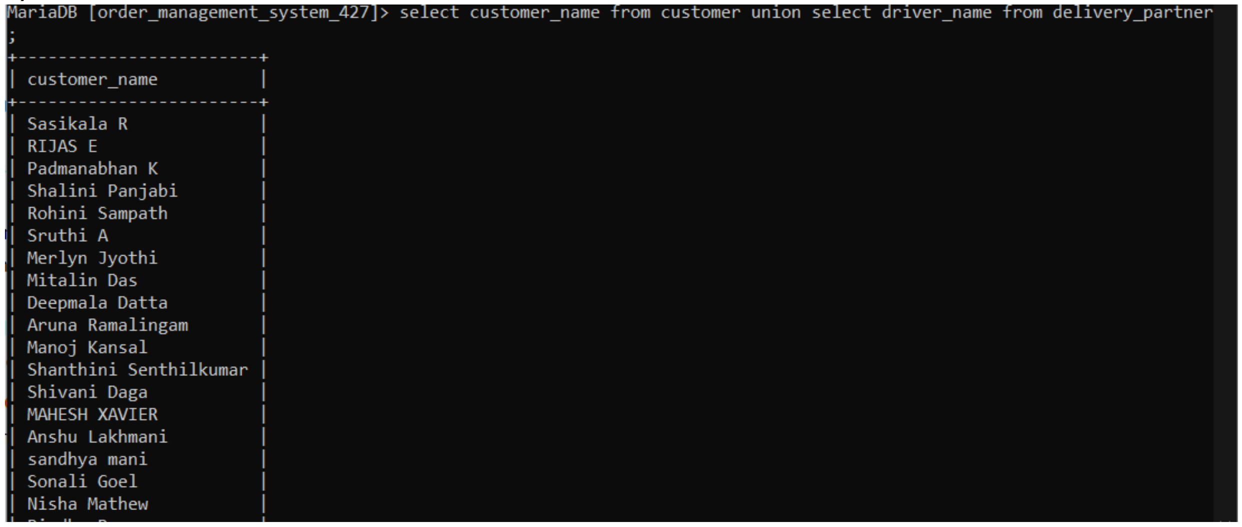
```
MariaDB [order_management_system_427]> select customer_id from orders intersect select customer_id from customer;
+-------------+
| customer_id |
+-------------+
|           1 |
|           2 |
|           3 |
|           4 |
|           5 |
|           6 |
|           7 |
|           8 |
|           9 |
|          10 |
|          11 |
|          12 |
|          13 |
|          14 |
|          15 |
|          16 |
|          17 |
|          18 |
|          19 |
|          20 |
```

2)**Select drivers who have been assigned an order**



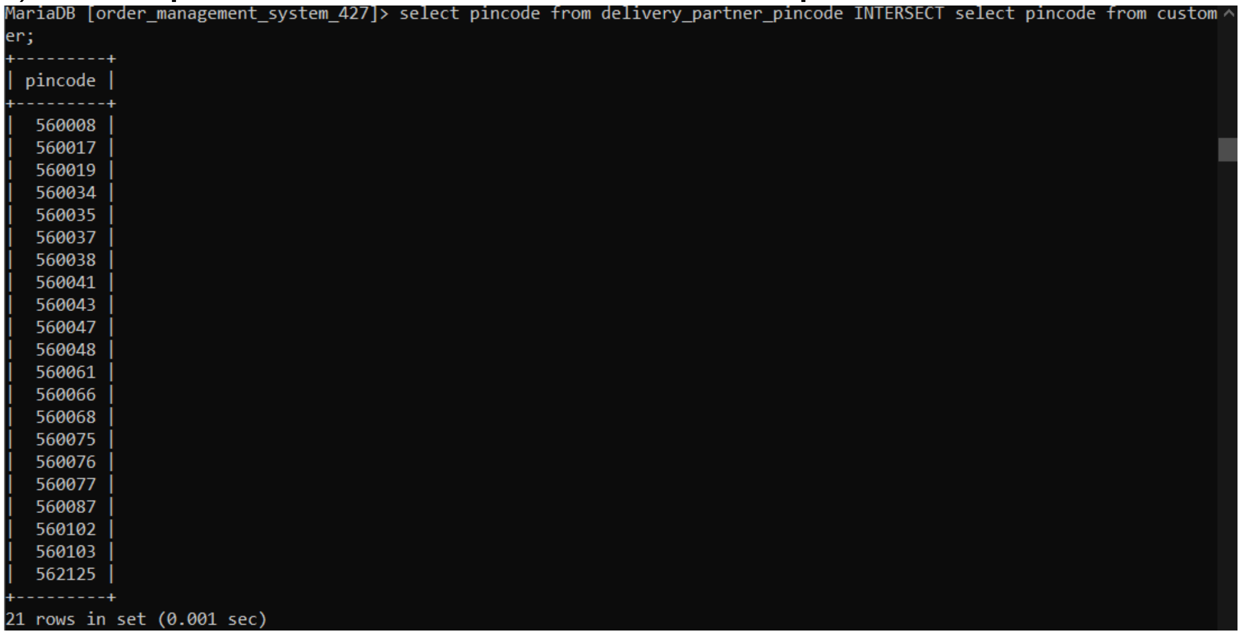
```
MariaDB [order_management_system_427]> select driver_id from delivery_partner intersect select driver_id from orders;
+-----------+
| driver_id |
+-----------+
|         1 |
|         3 |
|         4 |
|         9 |
|        10 |
|        11 |
|        12 |
|        13 |
+-----------+
```

3) **Select all names within drivers and customer**



```
MariaDB [order_management_system_427]> select customer_name from customer union select driver_name from delivery_partner
;
+------------------------+
| customer_name          |
+------------------------+
| Sasikala R             |
| RIJAS E                |
| Padmanabhan K          |
| Shalini Panjabi        |
| Rohini Sampath         |
| Sruthi A               |
| Merlyn Jyothi          |
| Mitalin Das            |
| Deepmala Datta         |
| Aruna Ramalingam       |
| Manoj Kansal           |
| Shanthini Senthilkumar |
| Shivani Daga           |
| MAHESH XAVIER          |
| Anshu Lakhmani         |
| sandhya mani           |
| Sonali Goel            |
| Nisha Mathew           |
```

4) **select all pincodes common to customer and driver to pincode table**



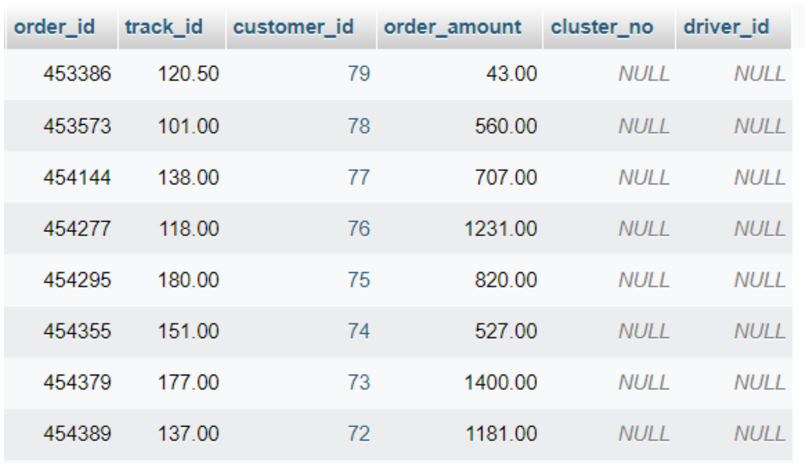
```
MariaDB [order_management_system_427]> select pincode from delivery_partner_pincode INTERSECT select pincode from custom
er;
+---------+
| pincode |
+---------+
|  560008 |
|  560017 |
|  560019 |
|  560034 |
|  560035 |
|  560037 |
|  560038 |
|  560041 |
|  560043 |
|  560047 |
|  560048 |
|  560061 |
|  560066 |
|  560068 |
|  560075 |
|  560076 |
|  560077 |
|  560087 |
|  560102 |
|  560103 |
|  562125 |
+---------+
21 rows in set (0.001 sec)
```

Functions and Procedures

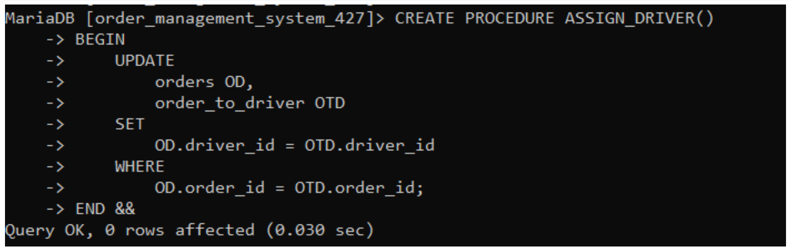
Create a Function and Procedure. State the objective of the function / Procedure. Run and display the results.

1) **STORED PROCEDURE TO ASSIGN A DRIVER**

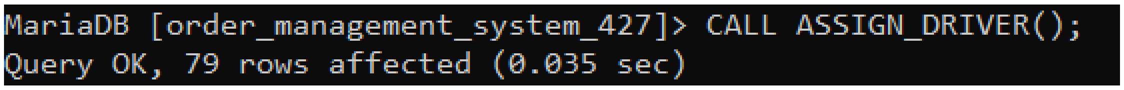
Before assigned



order_id	track_id	customer_id	order_amount	cluster_no	driver_id
453386	120.50	79	43.00	NULL	NULL
453573	101.00	78	560.00	NULL	NULL
454144	138.00	77	707.00	NULL	NULL
454277	118.00	76	1231.00	NULL	NULL
454295	180.00	75	820.00	NULL	NULL
454355	151.00	74	527.00	NULL	NULL
454379	177.00	73	1400.00	NULL	NULL
454389	137.00	72	1181.00	NULL	NULL

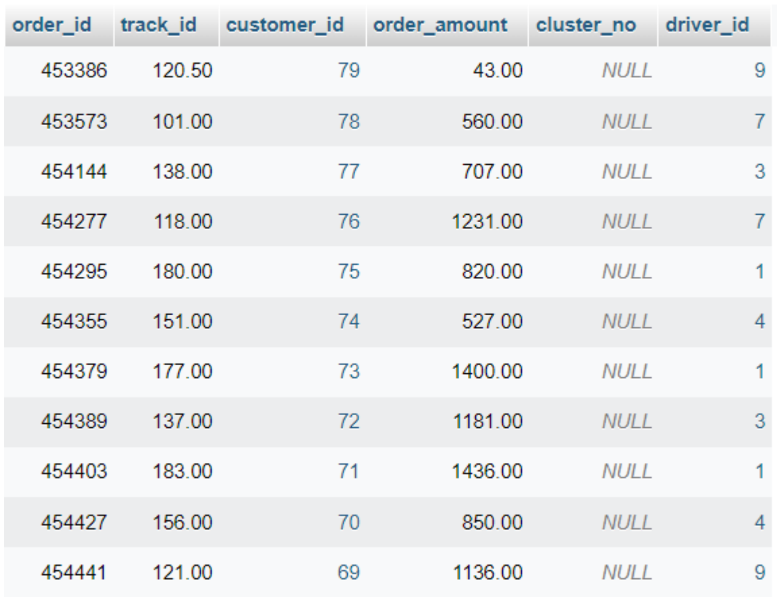


```
MariaDB [order_management_system_427]> CREATE PROCEDURE ASSIGN_DRIVER()
    -> BEGIN
    ->     UPDATE
    ->         orders OD,
    ->         order_to_driver OTD
    ->     SET
    ->         OD.driver_id = OTD.driver_id
    ->     WHERE
    ->         OD.order_id = OTD.order_id;
    -> END &&
Query OK, 0 rows affected (0.030 sec)
```



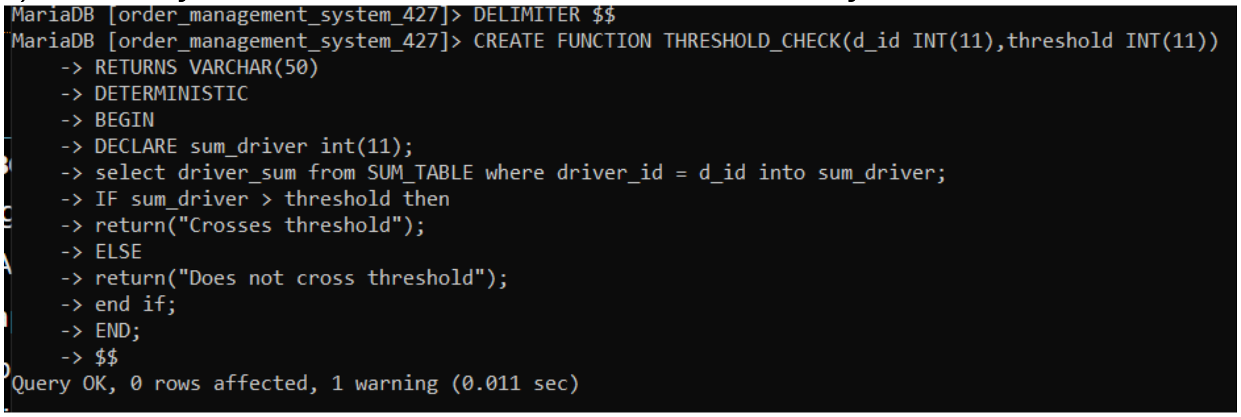
```
MariaDB [order_management_system_427]> CALL ASSIGN_DRIVER();
Query OK, 79 rows affected (0.035 sec)
```

After assigned

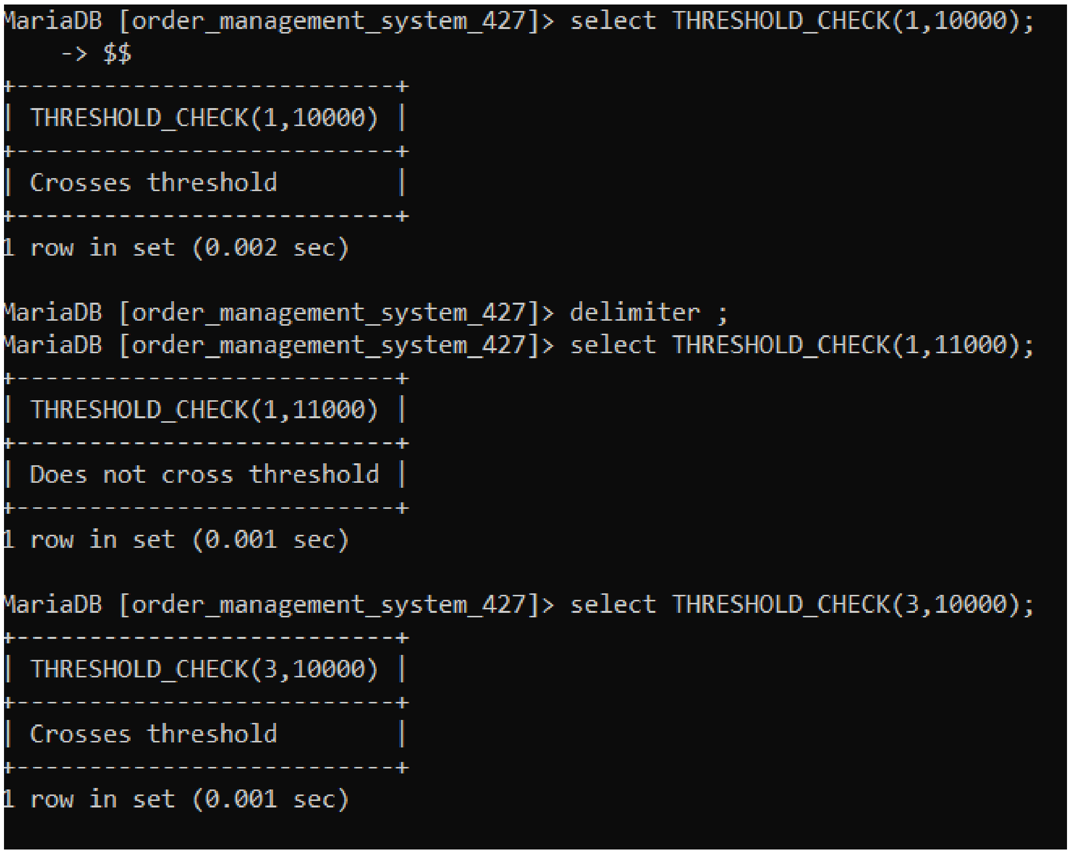


order_id	track_id	customer_id	order_amount	cluster_no	driver_id
453386	120.50	79	43.00	NULL	9
453573	101.00	78	560.00	NULL	7
454144	138.00	77	707.00	NULL	3
454277	118.00	76	1231.00	NULL	7
454295	180.00	75	820.00	NULL	1
454355	151.00	74	527.00	NULL	4
454379	177.00	73	1400.00	NULL	1
454389	137.00	72	1181.00	NULL	3
454403	183.00	71	1436.00	NULL	1
454427	156.00	70	850.00	NULL	4
454441	121.00	69	1136.00	NULL	9

2) **Find if any driver crosses a threshold amount he can carry**

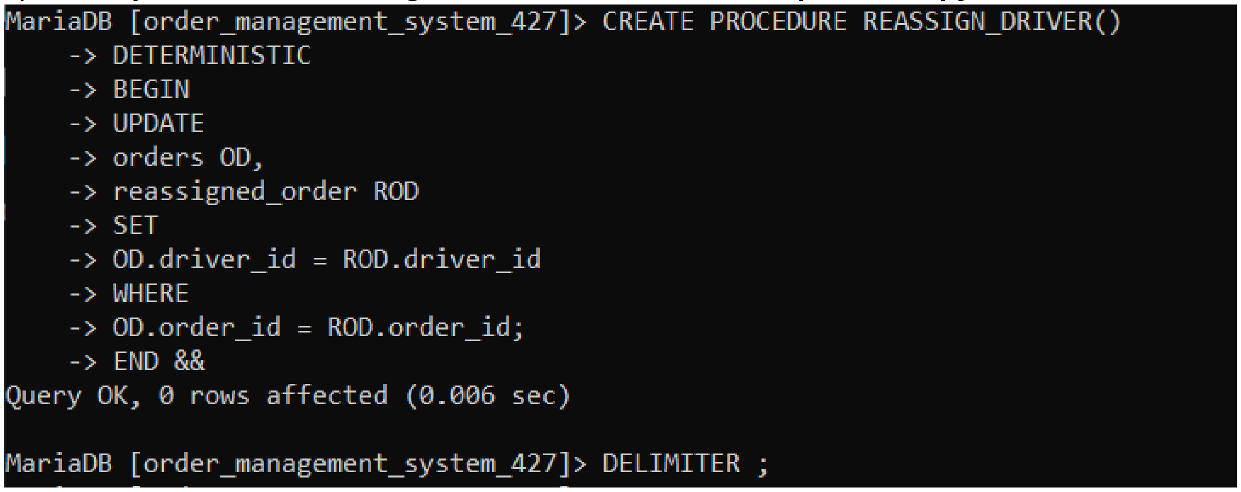


```
MariaDB [order_management_system_427]> DELIMITER $$
MariaDB [order_management_system_427]> CREATE FUNCTION THRESHOLD_CHECK(d_id INT(11),threshold INT(11))
    -> RETURNS VARCHAR(50)
    -> DETERMINISTIC
    -> BEGIN
    -> DECLARE sum_driver int(11);
    -> select driver_sum from SUM_TABLE where driver_id = d_id into sum_driver;
    -> IF sum_driver > threshold then
    -> return("Crosses threshold");
    -> ELSE
    -> return("Does not cross threshold");
    -> end if;
    -> END;
    -> $$
Query OK, 0 rows affected, 1 warning (0.011 sec)
```

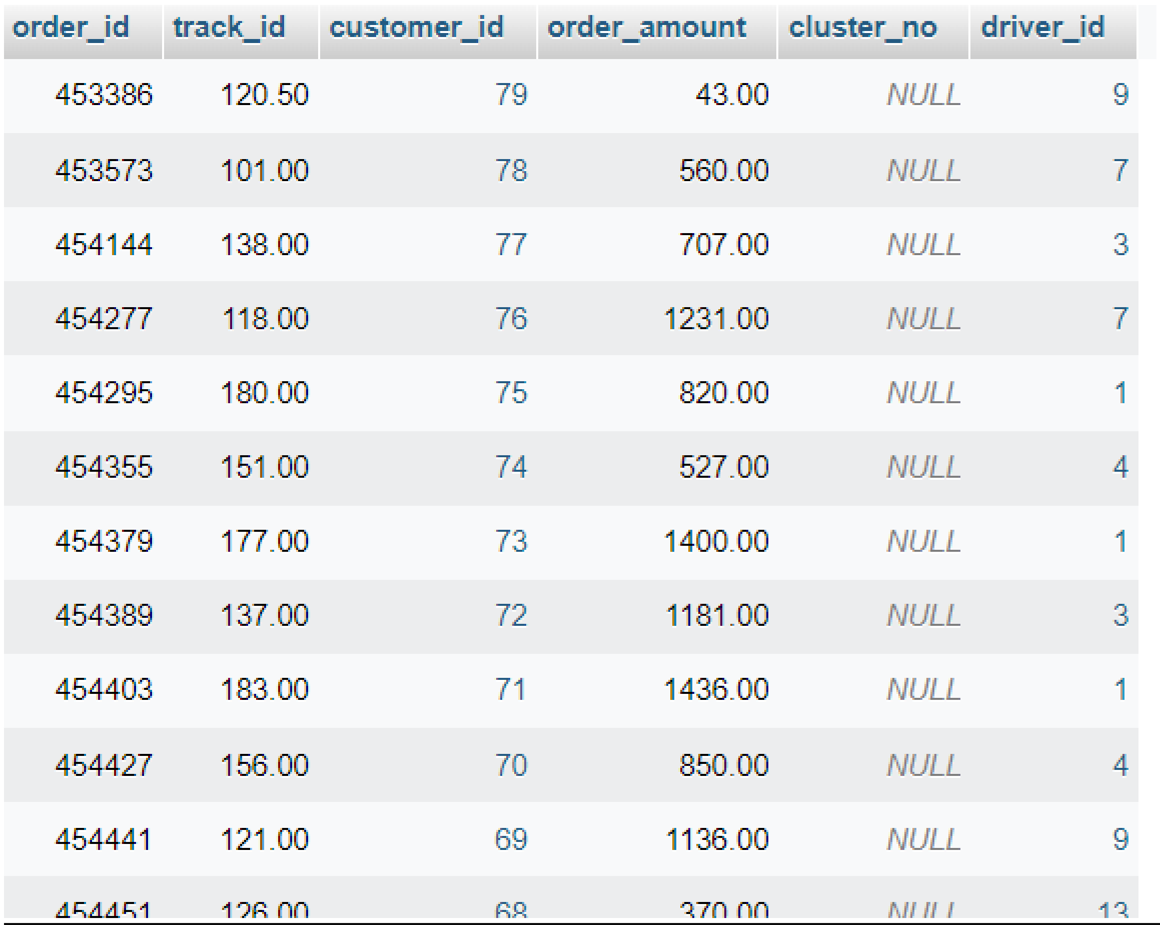


```
MariaDB [order_management_system_427]> select THRESHOLD_CHECK(1,10000);
    -> $$
+--------------------------+
| THRESHOLD_CHECK(1,10000) |
+--------------------------+
| Crosses threshold        |
+--------------------------+
1 row in set (0.002 sec)

MariaDB [order_management_system_427]> delimiter ;
MariaDB [order_management_system_427]> select THRESHOLD_CHECK(1,11000);
+--------------------------+
| THRESHOLD_CHECK(1,11000) |
+--------------------------+
| Does not cross threshold |
+--------------------------+
1 row in set (0.001 sec)

MariaDB [order_management_system_427]> select THRESHOLD_CHECK(3,10000);
+--------------------------+
| THRESHOLD_CHECK(3,10000) |
+--------------------------+
| Crosses threshold        |
+--------------------------+
1 row in set (0.001 sec)
```

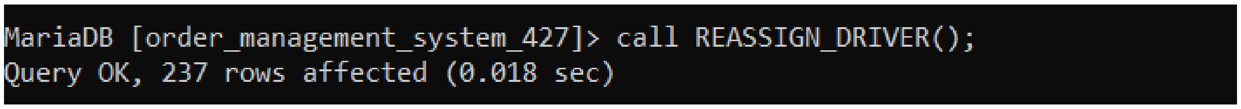
3)**Stored procedure to reassign a driver based on the output of the python code**



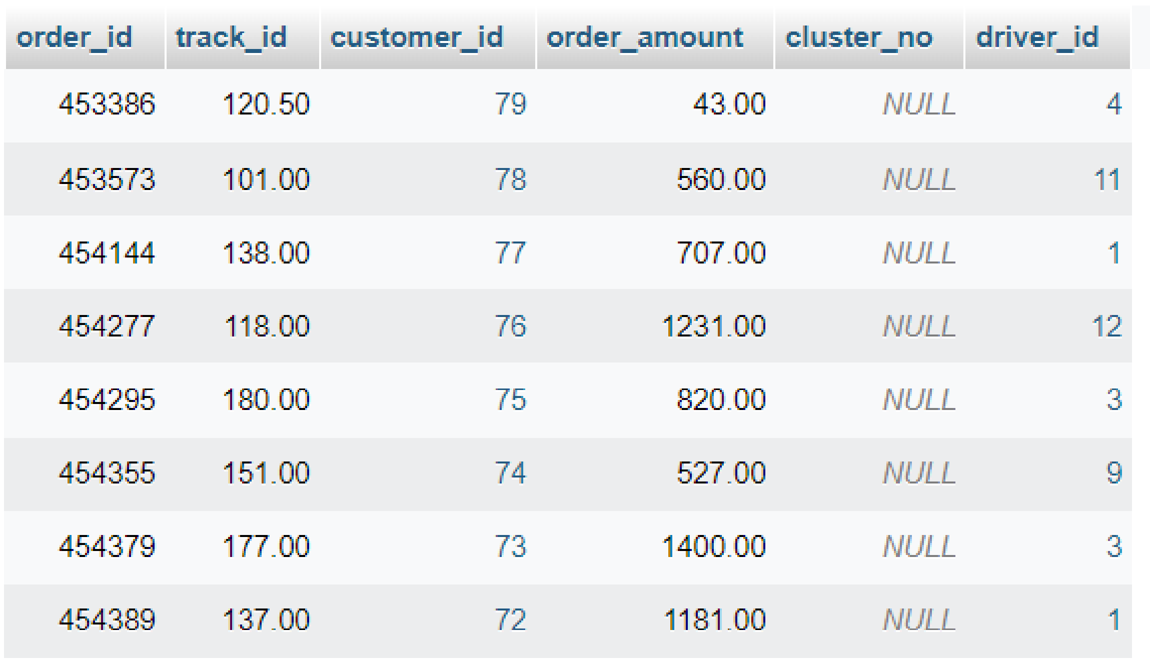
```
MariaDB [order_management_system_427]> CREATE PROCEDURE REASSIGN_DRIVER()
    -> DETERMINISTIC
    -> BEGIN
    -> UPDATE
    -> orders OD,
    -> reassigned_order ROD
    -> SET
    -> OD.driver_id = ROD.driver_id
    -> WHERE
    -> OD.order_id = ROD.order_id;
    -> END &&
Query OK, 0 rows affected (0.006 sec)

MariaDB [order_management_system_427]> DELIMITER ;
```



order_id	track_id	customer_id	order_amount	cluster_no	driver_id
453386	120.50	79	43.00	NULL	9
453573	101.00	78	560.00	NULL	7
454144	138.00	77	707.00	NULL	3
454277	118.00	76	1231.00	NULL	7
454295	180.00	75	820.00	NULL	1
454355	151.00	74	527.00	NULL	4
454379	177.00	73	1400.00	NULL	1
454389	137.00	72	1181.00	NULL	3
454403	183.00	71	1436.00	NULL	1
454427	156.00	70	850.00	NULL	4
454441	121.00	69	1136.00	NULL	9
454451	126.00	68	370.00	NULL	13



```
MariaDB [order_management_system_427]> call REASSIGN_DRIVER();
Query OK, 237 rows affected (0.018 sec)
```

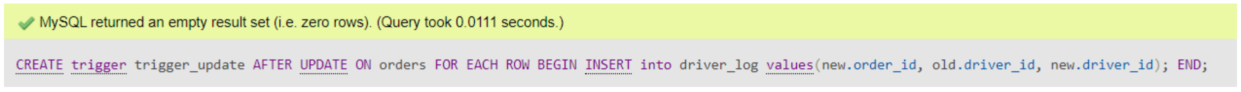


order_id	track_id	customer_id	order_amount	cluster_no	driver_id
453386	120.50	79	43.00	NULL	4
453573	101.00	78	560.00	NULL	11
454144	138.00	77	707.00	NULL	1
454277	118.00	76	1231.00	NULL	12
454295	180.00	75	820.00	NULL	3
454355	151.00	74	527.00	NULL	9
454379	177.00	73	1400.00	NULL	3
454389	137.00	72	1181.00	NULL	1

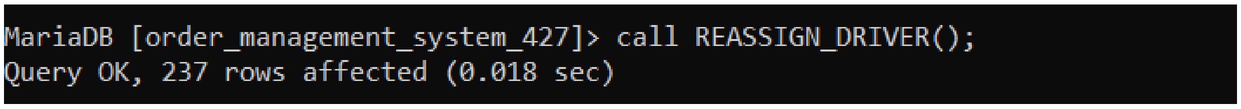
Triggers and Cursors

Create a Trigger and a Cursor. State the objective. Run and display the results.

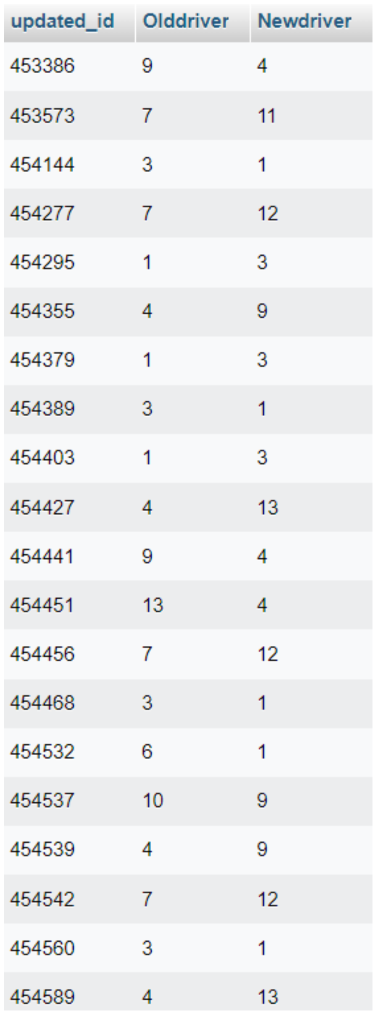
1) **Trigger to create the log table on a driver being reassigned or assigned**



```
✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0111 seconds.)

CREATE trigger trigger_update AFTER UPDATE ON orders FOR EACH ROW BEGIN INSERT into driver_log values(new.order_id, old.driver_id, new.driver_id); END;
```



```
MariaDB [order_management_system_427]> call REASSIGN_DRIVER();
Query OK, 237 rows affected (0.018 sec)
```



updated_id	Olddriver	Newdriver
453386	9	4
453573	7	11
454144	3	1
454277	7	12
454295	1	3
454355	4	9
454379	1	3
454389	3	1
454403	1	3
454427	4	13
454441	9	4
454451	13	4
454456	7	12
454468	3	1
454532	6	1
454537	10	9
454539	4	9
454542	7	12
454560	3	1
454589	4	13

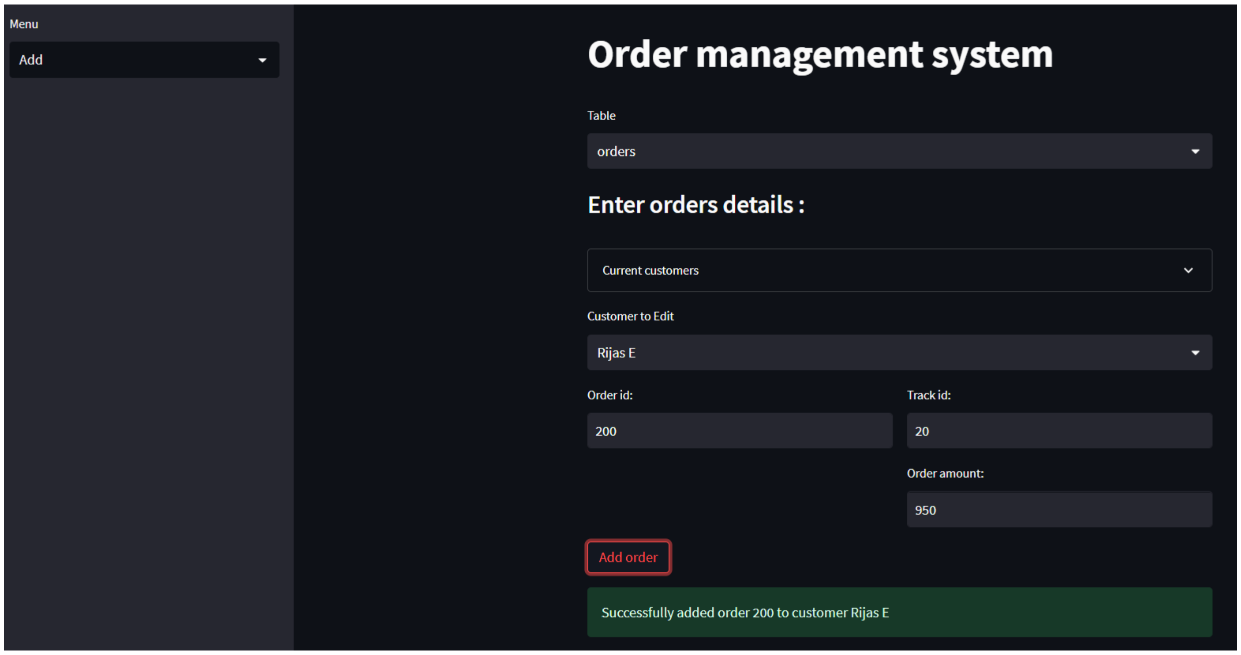
2) **Cursor to create deleted log table to see which stores information on which order is deleted**

Developing a Frontend

The frontend should support

1. Addition, Modification and Deletion of records from any chosen table
2. There should be an window to accept and run any SQL statement and display the result

ADD



Order management system

Table: orders

Enter orders details :

Current customers

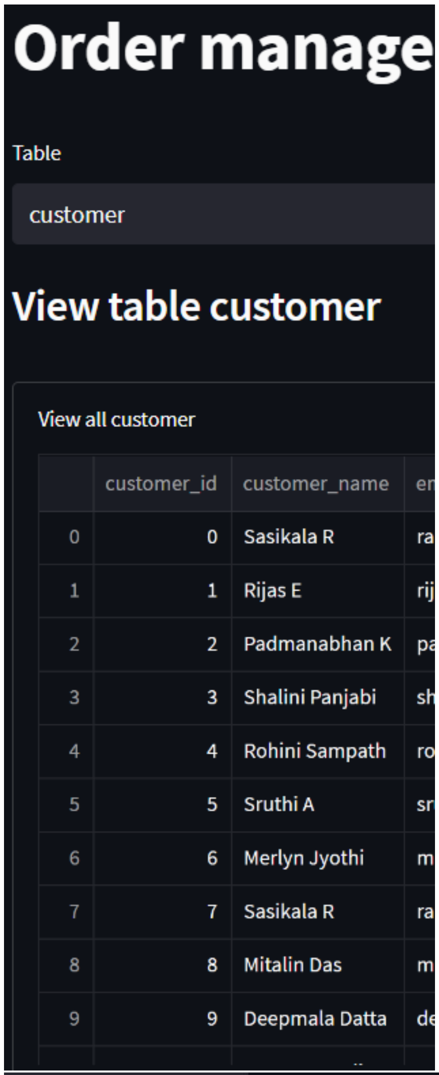
Customer to Edit: Rijas E

Order id: 200

Track id: 20

Order amount: 950

Successfully added order 200 to customer Rijas E



Order manage

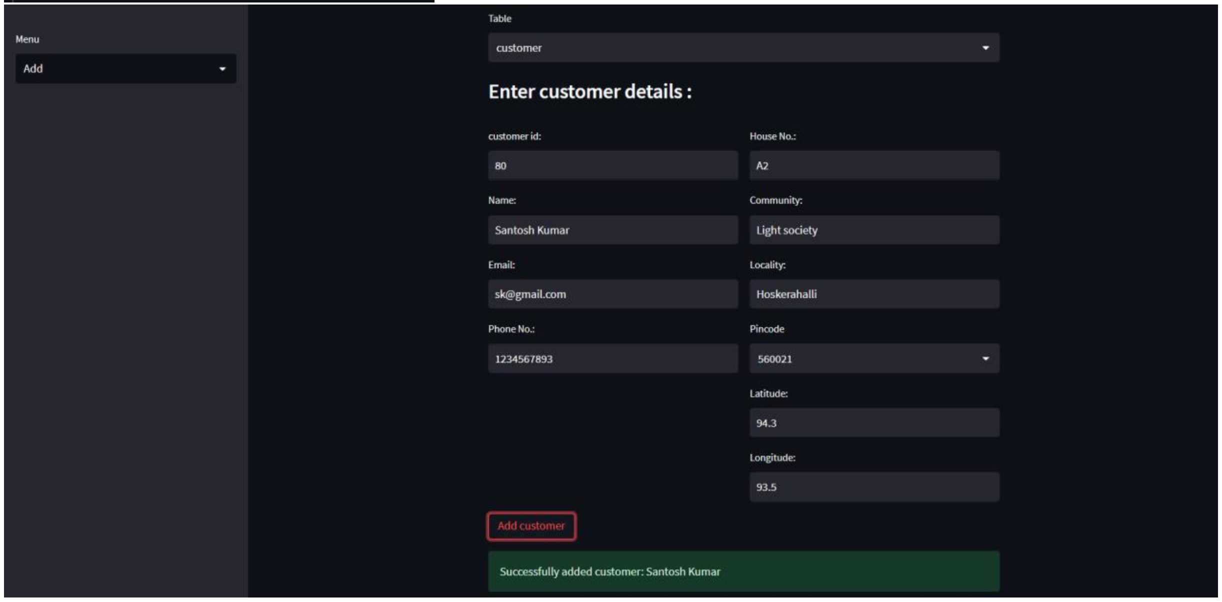
Table

customer

View table customer

View all customer

	customer_id	customer_name	er
0	0	Sasikala R	ra
1	1	Rijas E	rij
2	2	Padmanabhan K	pa
3	3	Shalini Panjabi	sh
4	4	Rohini Sampath	ro
5	5	Sruthi A	sr
6	6	Merlyn Jyothi	m
7	7	Sasikala R	ra
8	8	Mitalin Das	m
9	9	Deepmala Datta	de



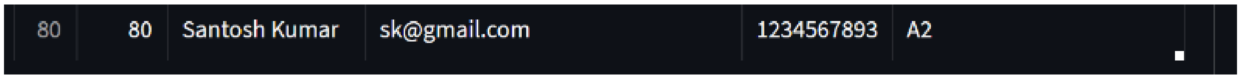
Menu: Add

Table: customer

Enter customer details :

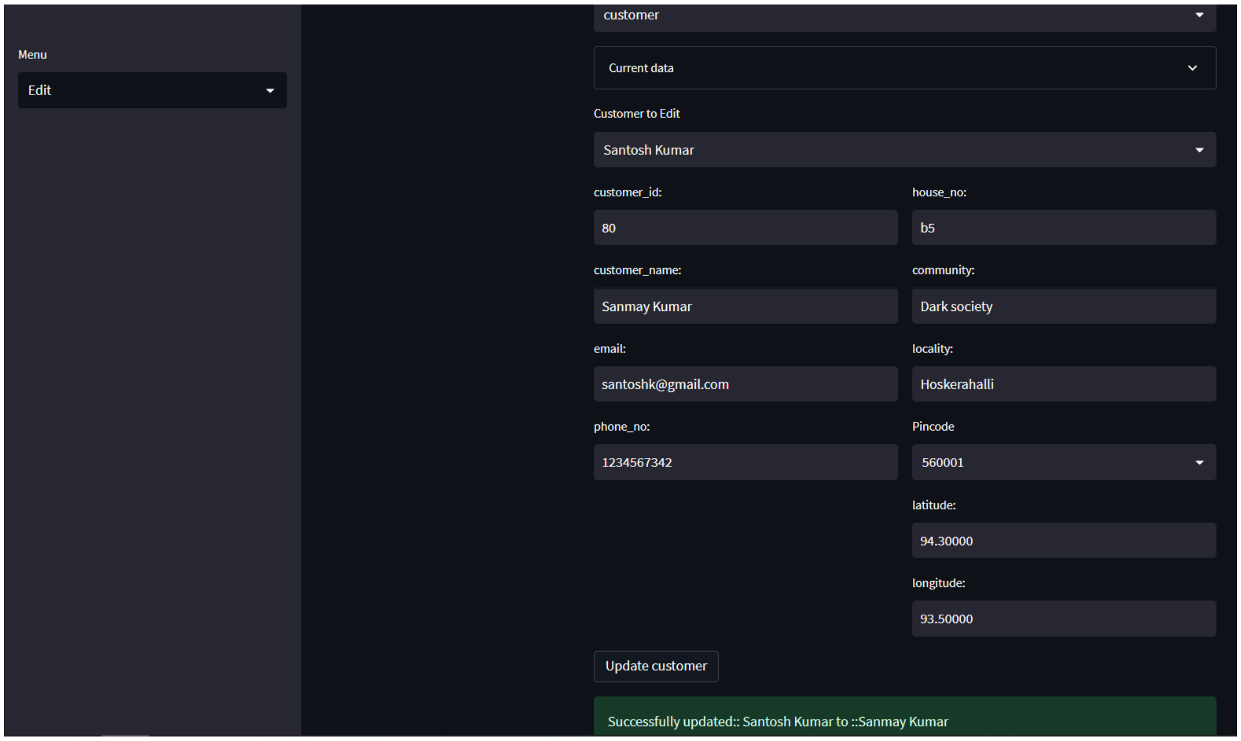
customer id: 80	House No.: A2
Name: Santosh Kumar	Community: Light society
Email: sk@gmail.com	Locality: Hoskerahalli
Phone No.: 1234567893	Pincode: 560021
	Latitude: 94.3
	Longitude: 93.5

Successfully added customer: Santosh Kumar



80	80	Santosh Kumar	sk@gmail.com	1234567893	A2

EDIT



Menu: Edit

customer

Current data

Customer to Edit: Santosh Kumar

customer_id: 80

house_no: b5

customer_name: Sanmay Kumar

community: Dark society

email: santoshk@gmail.com

locality: Hoskerahalli

phone_no: 1234567342

Pincode: 560001

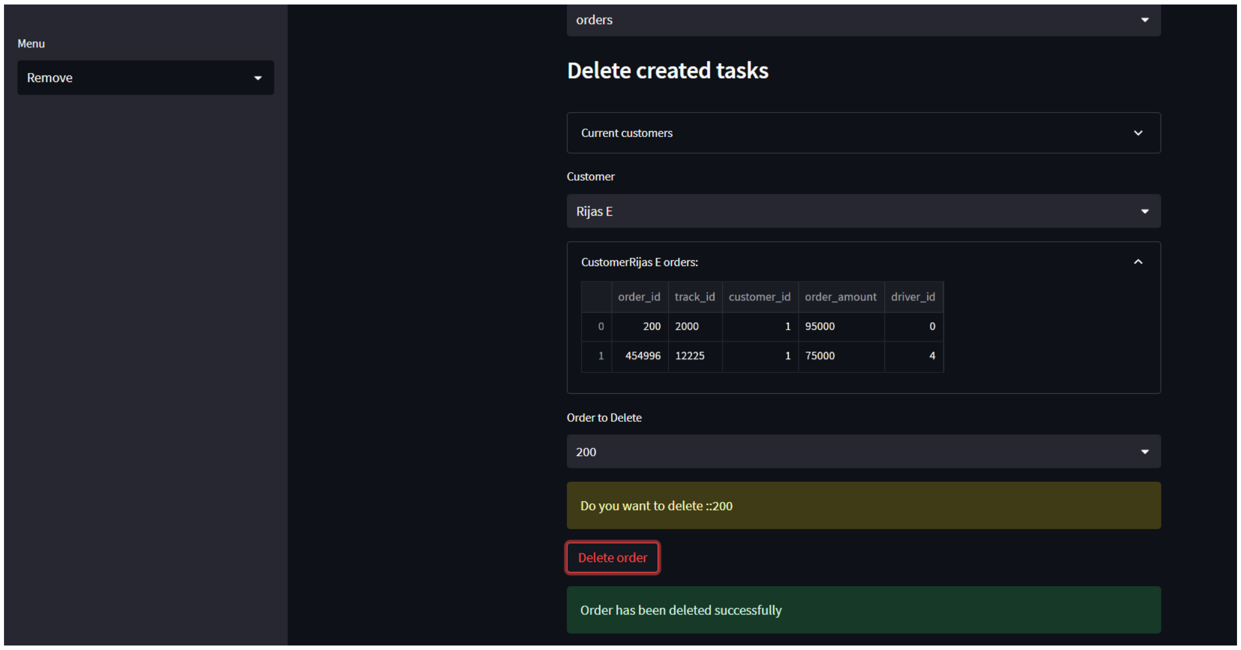
latitude: 94.30000

longitude: 93.50000

Update customer

Successfully updated:: Santosh Kumar to ::Sanmay Kumar

DELETE



Menu: Remove

orders

Delete created tasks

Current customers

Customer: Rijas E

CustomerRijas E orders:

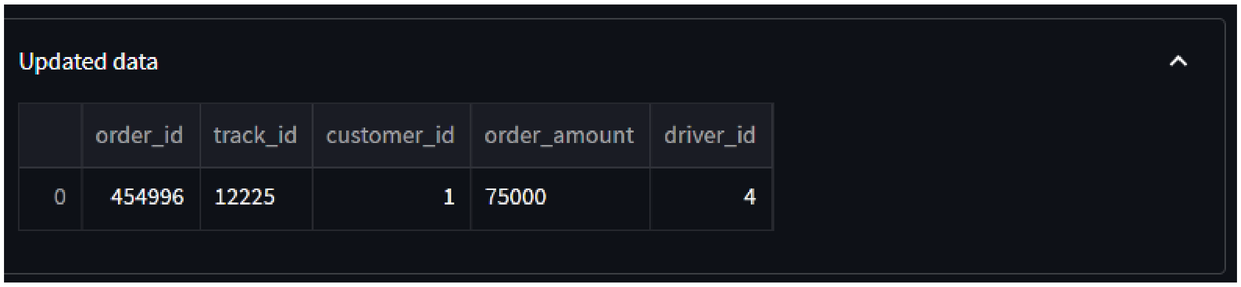
	order_id	track_id	customer_id	order_amount	driver_id
0	200	2000	1	95000	0
1	454996	12225	1	75000	4

Order to Delete: 200

Do you want to delete ::200

Delete order

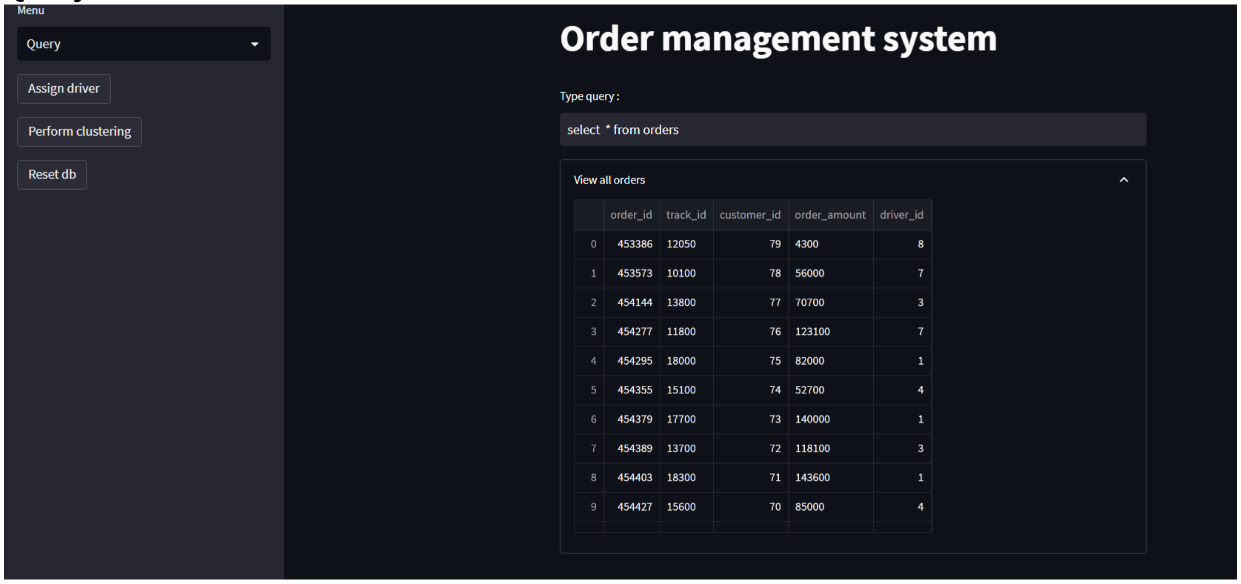
Order has been deleted successfully



Updated data

	order_id	track_id	customer_id	order_amount	driver_id
0	454996	12225	1	75000	4

Query



Order management system

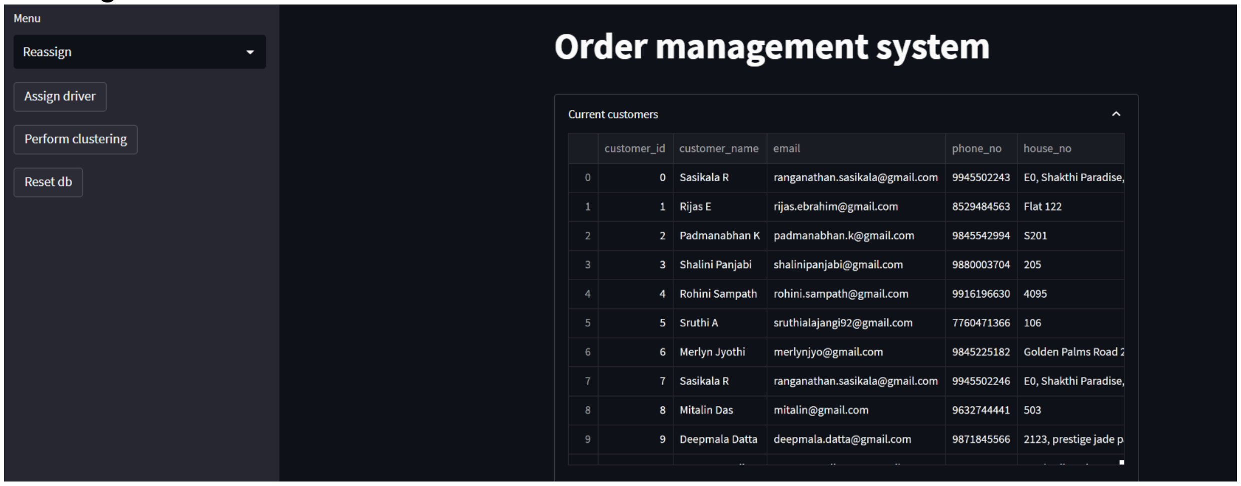
Type query :

select * from orders

View all orders

	order_id	track_id	customer_id	order_amount	driver_id
0	453386	12050	79	4300	8
1	453573	10100	78	56000	7
2	454144	13800	77	70700	3
3	454277	11800	76	123100	7
4	454295	18000	75	82000	1
5	454355	15100	74	52700	4
6	454379	17700	73	140000	1
7	454389	13700	72	118100	3
8	454403	18300	71	143600	1
9	454427	15600	70	85000	4

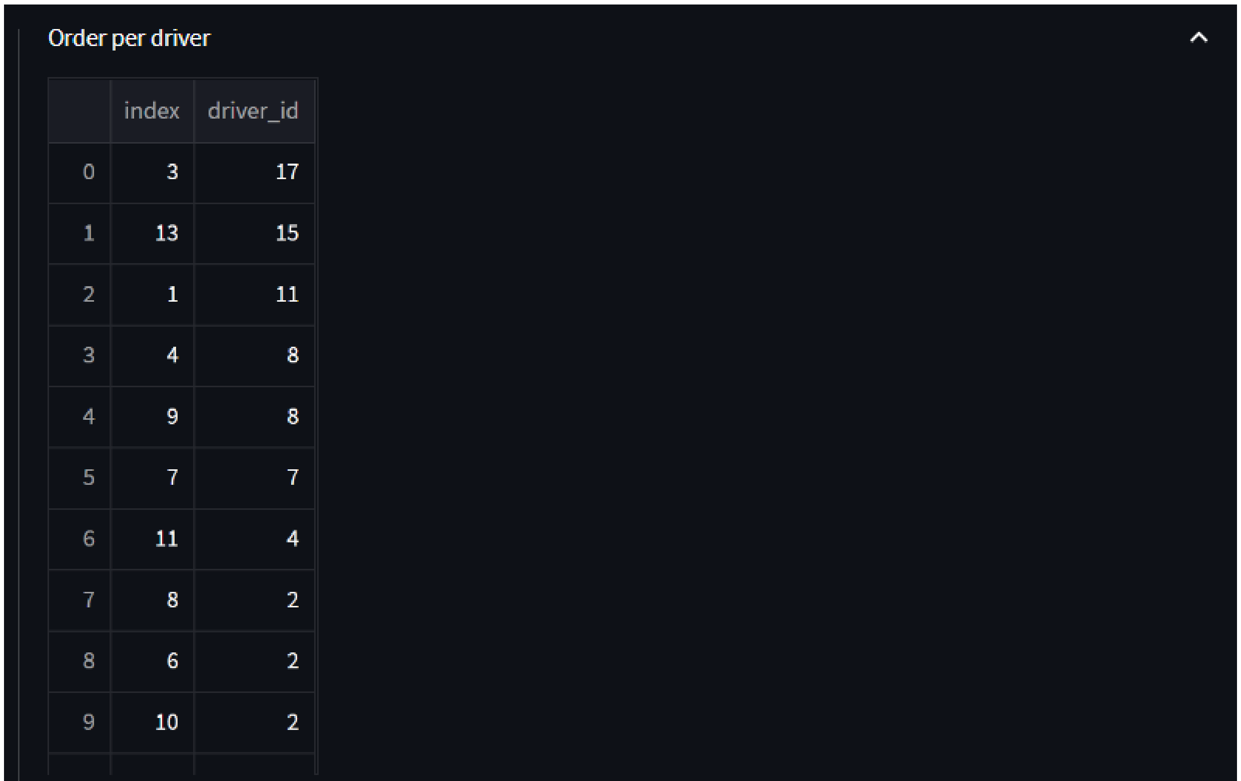
Reassign order



Order management system

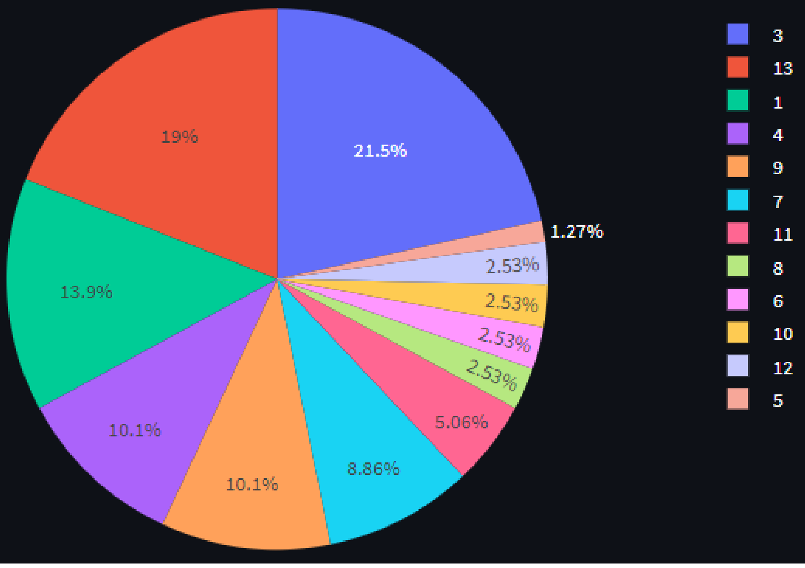
Current customers

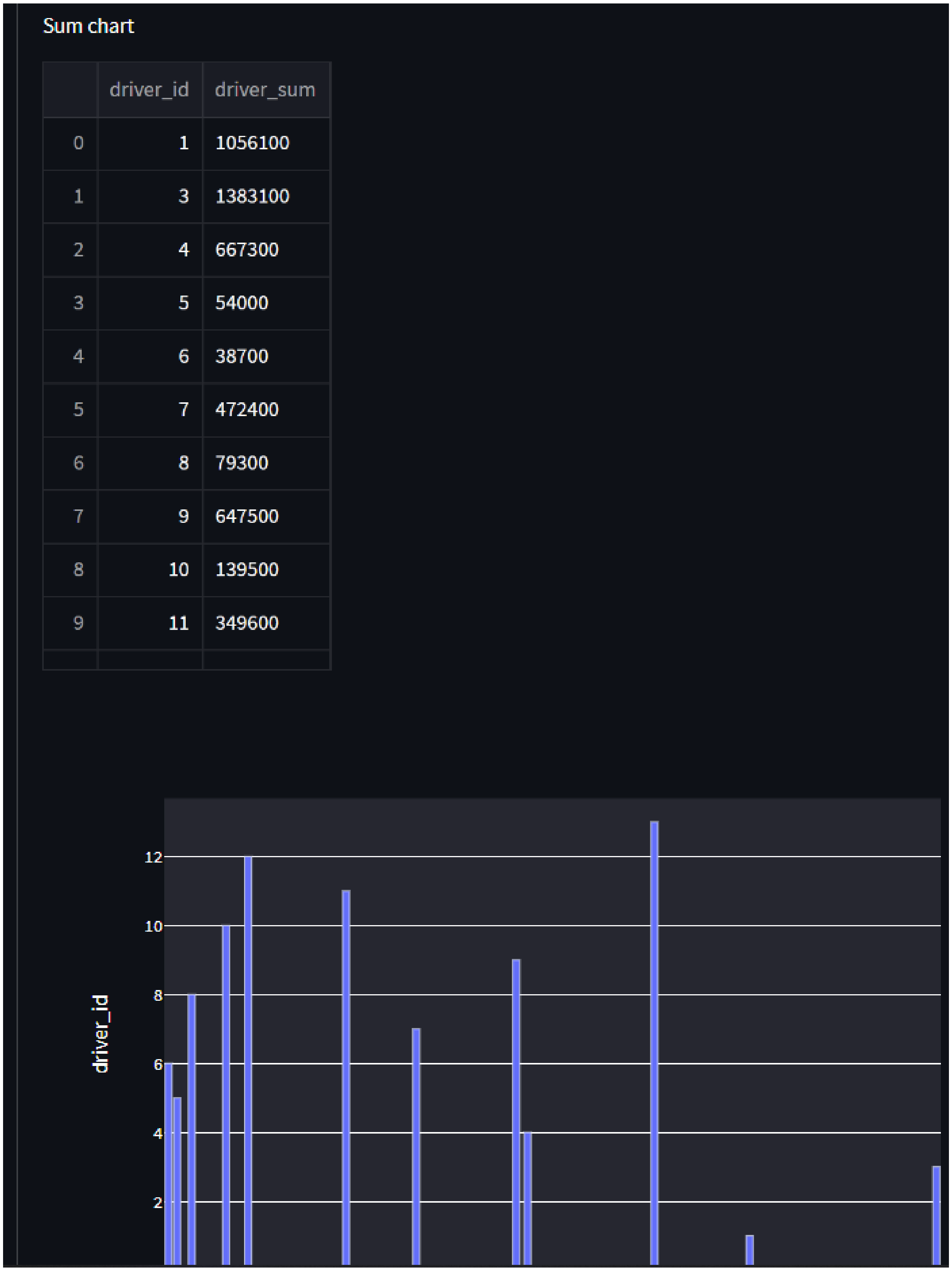
	customer_id	customer_name	email	phone_no	house_no
0	0	Sasikala R	ranganathan.sasikala@gmail.com	9945502243	E0, Shakthi Paradise,
1	1	Rijas E	rijas.ebrahim@gmail.com	8529484563	Flat 122
2	2	Padmanabhan K	padmanabhan.k@gmail.com	9845542994	S201
3	3	Shalini Panjabi	shalinipanjabi@gmail.com	9880003704	205
4	4	Rohini Sampath	rohini.sampath@gmail.com	9916196630	4095
5	5	Sruthi A	sruthialajangi92@gmail.com	7760471366	106
6	6	Merlyn Jyothi	merlynjyo@gmail.com	9845225182	Golden Palms Road 2
7	7	Sasikala R	ranganathan.sasikala@gmail.com	9945502246	E0, Shakthi Paradise,
8	8	Mitalin Das	mitalin@gmail.com	9632744441	503
9	9	Deepmala Datta	deepmala.datta@gmail.com	9871845566	2123, prestige jade p



Order per driver

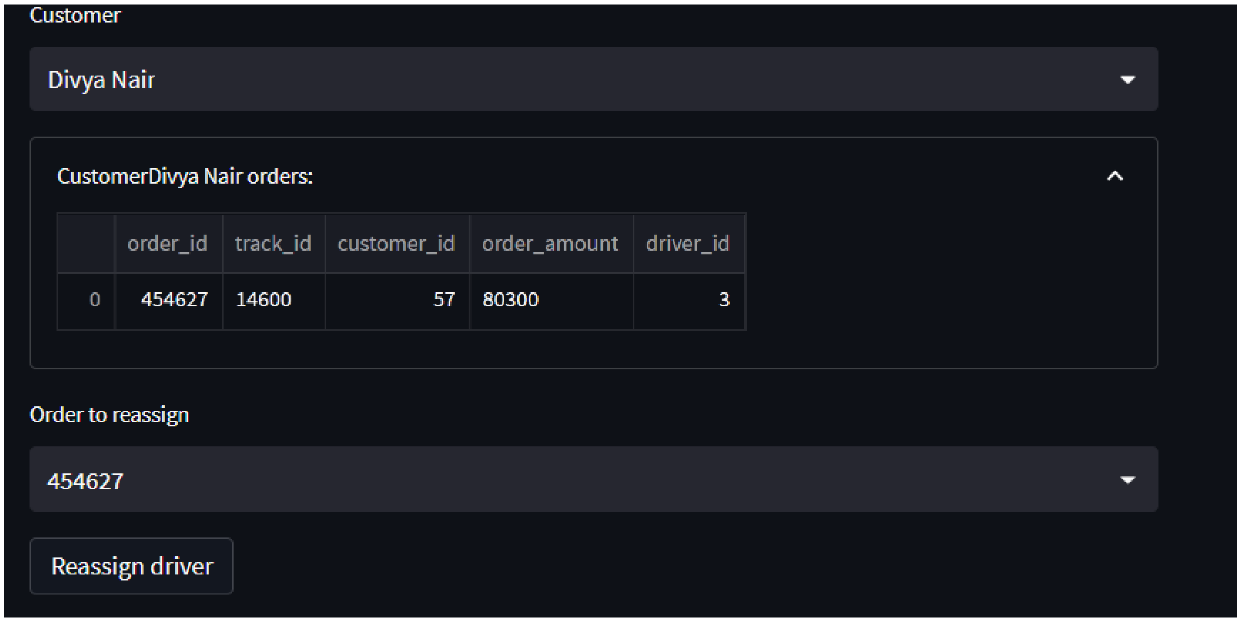
	index	driver_id
0	3	17
1	13	15
2	1	11
3	4	8
4	9	8
5	7	7
6	11	4
7	8	2
8	6	2
9	10	2





Sum chart

	driver_id	driver_sum
0	1	1056100
1	3	1383100
2	4	667300
3	5	54000
4	6	38700
5	7	472400
6	8	79300
7	9	647500
8	10	139500
9	11	349600



Customer

Divya Nair

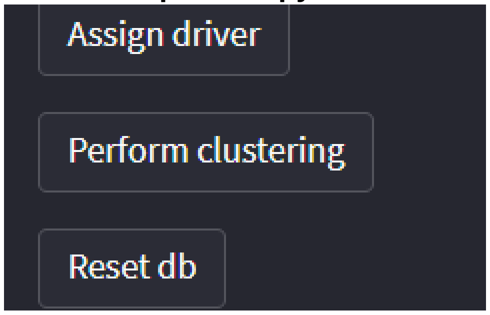
CustomerDivya Nair orders:

	order_id	track_id	customer_id	order_amount	driver_id
0	454627	14600	57	80300	3

Order to reassign

454627

Buttons to perform python codes


Assign driver
Perform clustering
Reset db