DAYANANDA SAGAR COLLEGE OF ENGINEERING

An Autonomous Institution
Affiliated to VTU
Approved by AICTE & UGC
Accredited by NBA
Accredited by NAAC with 'A' grade

(AN AUTONOMOUS INSTITUTE AFFILIATED TO VTU, BELAGAVI)

Shavige Malleshwara Hills, Kumaraswamy Layout, Bangalore-560078

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

DATABASE MANAGEMENT SYSTEMS LABORATORY MANUAL

Course Name and Course Code: DBMS Laboratory (22CG44)

Year and Semester : II year, IV semester

Name of the Faculty : Poornima D

VISION AND MISSION OF THE INSTITUTION

INSTITUTION VISION

To impact quality technical education with a focus on Research and Innovation emphasizing on Development of Sustainable and Inclusive Technology for the benefit of society.

INSTITUTION MISSION

- To provide an environment that enhances creativity and Innovation in pursuit of Excellence.
- To nurture teamwork in order to transform individuals as responsible leaders and entrepreneurs.
- To train the students to the changing technical scenario and make them to understand the importance of Sustainable and Inclusive technologies.

VISION AND MISSION OF CSE DEPARTMENT

DEPARTMENT VISION

Computer Science and Design Engineering Department shall architect the most innovative programs to deliver competitive and sustainable solutions using cutting edge technologies and implementations, for betterment of society and research.

DEPARTMENT MISSION

- To adopt the latest industry trends in teaching learning process in order to make students competitive in the job market
- To encourage forums that enable students to develop skills in multidisciplinary areas and emerging technologies

- ❖ To encourage research and innovation among students by creating an environment of learning through active participation and presentations
- ❖ To collaborate with industry and professional bodies for the students to gauge the market trends and train accordingly.
- ❖ To create an environment which fosters ethics and human values to make students responsible citizens.

COURSE OUTCOMES (CO)

Sl NO	DESCRIPTION	REVISED BLOOM'S TAXONOMY (RBT)LEVEL
1.	Implement sales database to understand the SQL Queries and Joins. Also to analyse the year wise sales data and zone wise sales data.	L4
2.	Implement the Employee database to write and understand Triggers and Views	L4
3.	Implement the Organization Database to understand Nested Queries	L4
4.	Implement the Airline Database system to Understand deletion of duplicate records along with Misc MYSQL Queries.	L4
5.	Implement MangoDB Database to create Database, collections, insert and update documents.	L4
LIST OF PROGRAMS		
S. No	Name of the Experiment	Course Outcome
1	Sales Database to understand basic SQL Queries and Joins	CO1
2	Employee Database to understand Triggers and views	CO2
3	Organization Database to understand Nested Queries	CO3
4	Airline Database system to Understand deletion of duplicate records along with Misc MYSQL Queries.	CO4
5	MangoDB Database	CO5

LAB PROGRAM 1

SALES DATABASE

For the given database schema, create the given tables with the attributes mentioned and write the SQL queries for the following.

SQL QUERIES:

QUERY 1: WRITE AN SQL QUERY TO GET THE TOTAL NUMBER OF CUSTOMERS IN SALES DATABASE

QUERY 2: WRITE AN SQL QUERY TO GET THE COMPLETE DETAILS OF TRANSACTIONS FOR MARKET CODE 'mark011'

QUERY 3: WRITE AN SQL QUERY TO GET THE TOTAL NUMBER OF TRANSACTIONS FOR MARKET CODE 'mark011'

QUERY 4: WRITE AN SQL QUERY TO GET THE DETAILS OF TRANSACTIONS INVOLVING US DOLLER CURRENCY

QUERY 5: WRITE AN SQL QUERY TO OBTAIN THE YEAR WISE SALES DATA

QUERY 6: WRITE AN SQL QUERY TO OBTAIN THE ZONE WISE SALES DATA

QUERY 7: WRITE AN SQL QUERY TO OBTAIN THE YEAR WISE SUM OF SALES

QUERY 8: WRITE AN SQL QUERY TO OBTAIN THE YEAR WISE AVG OF SALES

QUERY 9: WRITE AN SQL QUERY TO GET THE DETAILS OF CUSTOMER_CODE AND MARKET NAME

Note: in the transactions table, while inserting the data, for the columns profit_margin and *profit_margin_percentage*, null values should be inserted and the column values to calculated using below formulas.

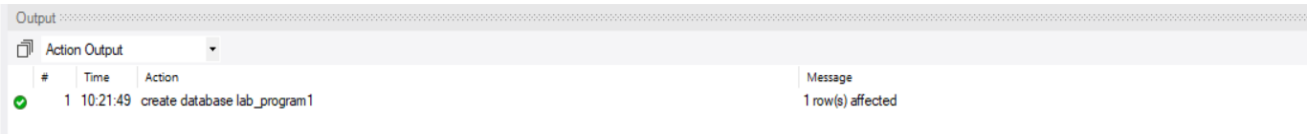
Formulas: to calculate the values of the columns profit_margin and *profit_margin_percentage*

profit_margin=cost_price-sales_amount;
profit_margin_percentage=profit_margin/cost_price*100;

SCHEMA DIAGRAM

##CREATE DATABASE NAMED lab_program1

create database lab_program1;

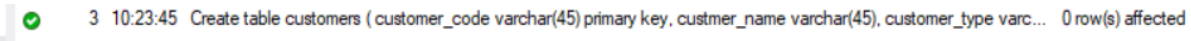


use lab_program1

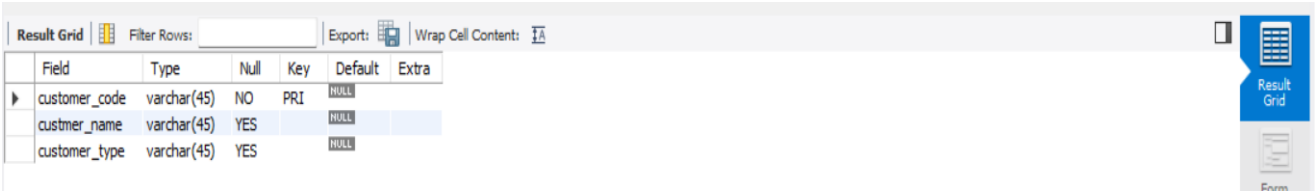
CREATE ALL THE TABLES AS PER THE SCHEMA DIAGRAM

Create table customers
(
customer_code varchar(45) primary key,
custmer_name varchar(45),
customer_type varchar(45)
);

OUTPUT



 # DISPLAY THE COLUMNS, DATATYPE AND CONSTRAINTS
Desc customers;
OUTPUT

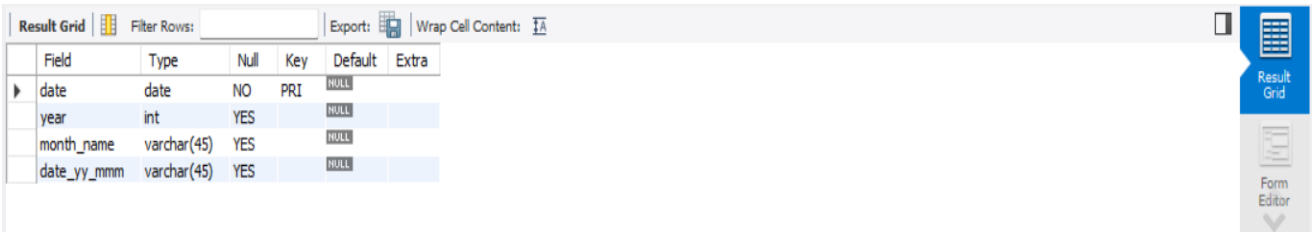


Field	Type	Null	Key	Default	Extra
customer_code	varchar(45)	NO	PRI	NULL	
custmer_name	varchar(45)	YES		NULL	
customer_type	varchar(45)	YES		NULL	

create table date
(
date date primary key,
year int,
month_name varchar(45),
date_yy_mmm varchar(45)
);

OUTPUT

 # DISPLAY THE COLUMNS, DATATYPE AND CONSTRAINTS
Desc date;



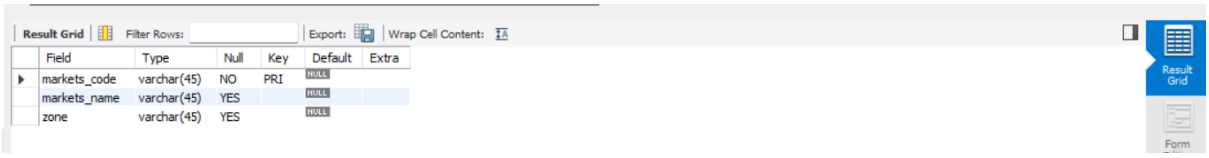
Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
year	int	YES		NULL	
month_name	varchar(45)	YES		NULL	
date_yy_mmm	varchar(45)	YES		NULL	

Create table markets
(
markets_code varchar(45) primary key,
markets_name varchar(45),
zone varchar(45)
);

OUTPUT

 # DISPLAY THE COLUMNS, DATATYPE AND CONSTRAINTS
Desc markets;

OUTPUT



Field	Type	Null	Key	Default	Extra
markets_code	varchar(45)	NO	PRI	NULL	
markets_name	varchar(45)	YES		NULL	
zone	varchar(45)	YES		NULL	

create table products
(

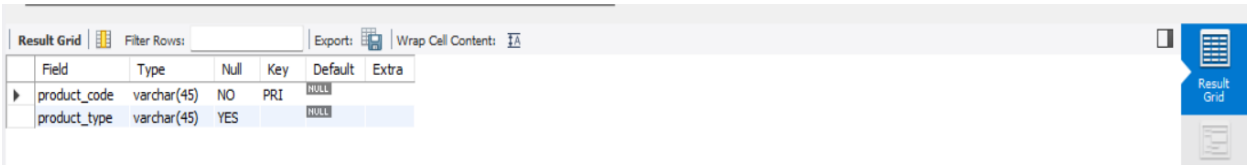
product_code varchar(45) primary key,
product_type varchar(45)
);

OUTPUT

9 10:28:09 create table products (product_code varchar(45) primary key, product_type varchar(45)) 0 row(s) affected

 # DISPLAY THE COLUMNS, DATATYPE AND CONSTRAINTS
Desc products;

OUTPUT



Field	Type	Null	Key	Default	Extra
product_code	varchar(45)	NO	PRI	NULL	
product_type	varchar(45)	YES		NULL	

create table transactions
(
product_code varchar(45) references products.product_code,
customer_code varchar(45) references customers.customer_code,
market_code varchar(45) references markets.market_code,
order_date date references date.date,
sales_qty int,
sales_amount double,
currency varchar(45),
profit_margin_percentage double,
profit_margin double,
cost_price double
);

OUTPUT:

11 10:29:28 create table transactions (product_code varchar(45) references products.product_code, customer_code varc... 0 row(s) affected

Desc transactions

OUTPUT

Field	Type	Null	Key	Default	Extra
product_code	varchar(45)	YES		NULL	
customer_code	varchar(45)	YES		NULL	
market_code	varchar(45)	YES		NULL	
order_date	date	YES		NULL	
sales_qty	int	YES		NULL	
sales_amount	double	YES		NULL	
currency	varchar(45)	YES		NULL	
profit_margin_percentage	double	YES		NULL	
profit_margin	double	YES		NULL	
cost_price	double	YES		NULL	

INSERT VALUES INTO THE RESPECTIVE TABLES.

INSERT VALUES INTO THE CUSTOMER TABLE

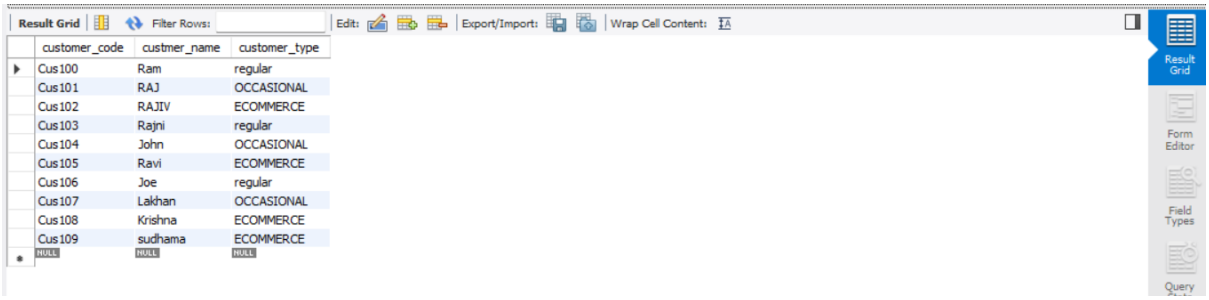
```
insert into customers values('Cus100','Ram','regular');
insert into customers values('Cus101','RAJ','OCCASIONAL');
insert into customers values('Cus102','RAJIV','ECOMMERCE');
insert into customers values('Cus103','Rajni','regular');
insert into customers values('Cus104','John','OCCASIONAL');
insert into customers values('Cus105','Ravi','ECOMMERCE');
insert into customers values('Cus106','Joe','regular');
insert into customers values('Cus107','Lakhan','OCCASIONAL');
insert into customers values('Cus108','Krishna','ECOMMERCE');
insert into customers values('Cus109','sudhama','ECOMMERCE');
```

OUTPUT

✔	13	10:31:33	insert into customers values('Cus100','Ram','regular')	1 row(s) affected	0.000 sec
✔	14	10:31:33	insert into customers values('Cus101','RAJ','OCCASIONAL')	1 row(s) affected	0.000 sec
✔	15	10:31:33	insert into customers values('Cus102','RAJIV','ECOMMERCE')	1 row(s) affected	0.000 sec
✔	16	10:31:33	insert into customers values('Cus103','Rajni','regular')	1 row(s) affected	0.000 sec
✔	17	10:31:33	insert into customers values('Cus104','John','OCCASIONAL')	1 row(s) affected	0.000 sec
✔	18	10:31:33	insert into customers values('Cus105','Ravi','ECOMMERCE')	1 row(s) affected	0.000 sec
✔	19	10:31:33	insert into customers values('Cus106','Joe','regular')	1 row(s) affected	0.000 sec
✔	20	10:31:33	insert into customers values('Cus107','Lakhan','OCCASIONAL')	1 row(s) affected	0.000 sec
✔	21	10:31:33	insert into customers values('Cus108','Krishna','ECOMMERCE')	1 row(s) affected	0.015 sec
✔	22	10:31:33	insert into customers values('Cus109','sudhama','ECOMMERCE')	1 row(s) affected	0.000 sec

select * from customers;

OUTPUT



customer_code	custmer_name	customer_type
Cus100	Ram	regular
Cus101	RAJ	OCCASIONAL
Cus102	RAJIV	ECOMMERCE
Cus103	Rajni	regular
Cus104	John	OCCASIONAL
Cus105	Ravi	ECOMMERCE
Cus106	Joe	regular
Cus107	Lakhan	OCCASIONAL
Cus108	Krishna	ECOMMERCE
Cus109	sudhama	ECOMMERCE
NULL	NULL	NULL

```
Alter table date Drop column date_yy_mmm
desc date

insert into date values('2016-06-30',2016,'june');
insert into date values('2016-07-28',2016,'july');
insert into date values('2016-06-25',2016,'june');
insert into date values('2016-07-20',2016,'july');
```

insert into date values('2016-08-15',2016,'aug');

#	Time	Action	Message	Duration / Fetch
24	10:33:48	Alter table date Drop column date_yy_mmm	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
25	10:34:14	insert into date values('2016-06-30',2016,'june')	1 row(s) affected	0.031 sec
26	10:34:14	insert into date values('2016-07-28',2016,'july')	1 row(s) affected	0.000 sec
27	10:34:14	insert into date values('2016-06-25',2016,'june')	1 row(s) affected	0.000 sec
28	10:34:14	insert into date values('2016-07-20',2016,'july')	1 row(s) affected	0.000 sec
29	10:34:14	insert into date values('2016-08-15',2016,'aug')	1 row(s) affected	0.000 sec

INSERT VALUES INTO THE DATE TABLE

insert into date values('2017-06-30',2017,'june');
insert into date values('2017-07-28',2017,'july');
insert into date values('2017-06-25',2017,'june');
insert into date values('2017-07-20',2017,'july');
insert into date values('2017-08-15',2017,'aug');
insert into date values('2017-11-30',2017,'nov');
insert into date values('2017-12-28',2017,'dec');
insert into date values('2017-11-25',2017,'nov');
insert into date values('2017-12-20',2017,'dec');
insert into date values('2017-11-15',2017,'nov');

32	10:35:03	insert into date values('2017-06-25',2017,'june')	1 row(s) affected	0.000 sec
33	10:35:03	insert into date values('2017-07-20',2017,'july')	1 row(s) affected	0.000 sec
34	10:35:03	insert into date values('2017-08-15',2017,'aug')	1 row(s) affected	0.000 sec
35	10:35:03	insert into date values('2017-11-30',2017,'nov')	1 row(s) affected	0.016 sec
36	10:35:03	insert into date values('2017-12-28',2017,'dec')	1 row(s) affected	0.000 sec
37	10:35:03	insert into date values('2017-11-25',2017,'nov')	1 row(s) affected	0.000 sec
38	10:35:03	insert into date values('2017-12-20',2017,'dec')	1 row(s) affected	0.015 sec
39	10:35:03	insert into date values('2017-11-15',2017,'nov')	1 row(s) affected	0.000 sec

insert into date values('2018-01-30',2018,'jan');
insert into date values('2018-02-28',2018,'feb');
insert into date values('2018-04-25',2018,'april');
insert into date values('2018-04-20',2018,'april');
insert into date values('2018-04-15',2018,'april');
insert into date values('2018-04-30',2018,'april');
insert into date values('2018-01-28',2018,'jan');
insert into date values('2018-02-25',2018,'feb');
insert into date values('2018-02-20',2018,'feb');
insert into date values('2018-02-15',2018,'feb');

40	10:36:37	insert into date values('2018-01-30',2018,'jan')	1 row(s) affected	0.031 sec
41	10:36:37	insert into date values('2018-02-28',2018,'feb')	1 row(s) affected	0.000 sec
42	10:36:37	insert into date values('2018-04-25',2018,'april')	1 row(s) affected	0.000 sec
43	10:36:37	insert into date values('2018-04-20',2018,'april')	1 row(s) affected	0.016 sec
44	10:36:37	insert into date values('2018-04-15',2018,'april')	1 row(s) affected	0.000 sec
45	10:36:37	insert into date values('2018-04-30',2018,'april')	1 row(s) affected	0.000 sec
46	10:36:37	insert into date values('2018-01-28',2018,'jan')	1 row(s) affected	0.000 sec
47	10:36:37	insert into date values('2018-02-25',2018,'feb')	1 row(s) affected	0.000 sec
48	10:36:37	insert into date values('2018-02-20',2018,'feb')	1 row(s) affected	0.000 sec
49	10:36:37	insert into date values('2018-02-15',2018,'feb')	1 row(s) affected	0.000 sec

select * from date;

OUTPUT

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

date	year	month_name
2016-06-25	2016	june
2016-06-30	2016	june
2016-07-20	2016	july
2016-07-28	2016	july
2016-08-15	2016	aug
2017-06-25	2017	june
2017-06-30	2017	june
2017-07-20	2017	july
2017-07-28	2017	july
2017-08-15	2017	aug
2017-11-15	2017	nov
2017-11-25	2017	nov
2017-11-30	2017	nov
2017-12-20	2017	dec
2017-12-28	2017	dec
2018-01-28	2018	jan
2018-01-30	2018	jan
2018-02-15	2018	feb
2018-02-20	2018	feb
2018-02-25	2018	feb
2018-02-28	2018	feb
2018-04-15	2018	april
2018-04-20	2018	april

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

INSERT VALUES INTO THE MARKETS TABLE

```
INSERT INTO markets values('mark500','bangalore','south');
INSERT INTO markets values('mark501','SURAT','NORTH');
INSERT INTO markets values('mark502','Dehli','NORTH');
INSERT INTO markets values('mark503','LATOOR','NORTH');
INSERT INTO markets values('mark504','CHENNAI','south');
INSERT INTO markets values('mark505','RANCHI','NORTH');
INSERT INTO markets values('mark506','MYSORE','south');
INSERT INTO markets values('mark507','BHUVANESHWAR','NORTH');
INSERT INTO markets values('mark508','TELANGANA','south');
INSERT INTO markets values('mark509','GANDHINAGAR','NORTH');
INSERT INTO markets values('mark510','HUBLI','south');
INSERT INTO markets values('mark511','JAIPUR','NORTH');
```

```
INSERT INTO markets values('mark512','SELAM','south');
INSERT INTO markets values('mark513','BHUJ','NORTH');
INSERT INTO markets values('mark514','HYDRABAD','south');
INSERT INTO markets values('mark515','JAMMU','NORTH');
INSERT INTO markets values('mark516','VIZAK','south');
INSERT INTO markets values('mark517','MADHUPUR','NORTH');
INSERT INTO markets values('mark518','TRIVENDRAM','south');
INSERT INTO markets values('mark519','PATNA','NORTH');
```

OUTPUT

✓	51	10:38:41	INSERT INTO markets values('mark500','bangalore','south')	1 row(s) affected	0.031 sec
✓	52	10:38:41	INSERT INTO markets values('mark501','SURAT','NORTH')	1 row(s) affected	0.000 sec
✓	53	10:38:41	INSERT INTO markets values('mark502','Dehli','NORTH')	1 row(s) affected	0.016 sec
✓	54	10:38:41	INSERT INTO markets values('mark503','LATOOR','NORTH')	1 row(s) affected	0.000 sec
✓	55	10:38:41	INSERT INTO markets values('mark504','CHENNAI','south')	1 row(s) affected	0.000 sec
✓	56	10:38:41	INSERT INTO markets values('mark505','RANCHI','NORTH')	1 row(s) affected	0.000 sec
✓	57	10:38:41	INSERT INTO markets values('mark506','MYSORE','south')	1 row(s) affected	0.015 sec
✓	58	10:38:41	INSERT INTO markets values('mark507','BHUVANESHWAR','NORTH')	1 row(s) affected	0.000 sec
✓	59	10:38:41	INSERT INTO markets values('mark508','TELANGANA','south')	1 row(s) affected	0.000 sec
✓	60	10:38:41	INSERT INTO markets values('mark509','GANDHINAGAR','NORTH')	1 row(s) affected	0.000 sec
✓	61	10:38:41	INSERT INTO markets values('mark510','HUBLI','south')	1 row(s) affected	0.016 sec
✓	62	10:38:41	INSERT INTO markets values('mark511','JAIPUR','NORTH')	1 row(s) affected	0.000 sec
✓	63	10:38:41	INSERT INTO markets values('mark512','SELAM','south')	1 row(s) affected	0.000 sec
✓	64	10:38:41	INSERT INTO markets values('mark513','BHUJ','NORTH')	1 row(s) affected	0.000 sec
✓	65	10:38:41	INSERT INTO markets values('mark514','HYDRABAD','south')	1 row(s) affected	0.016 sec
✓	66	10:38:41	INSERT INTO markets values('mark515','JAMMU','NORTH')	1 row(s) affected	0.000 sec
✓	67	10:38:41	INSERT INTO markets values('mark516','VIZAK','south')	1 row(s) affected	0.000 sec
✓	68	10:38:41	INSERT INTO markets values('mark517','MADHUPUR','NORTH')	1 row(s) affected	0.000 sec
✓	69	10:38:41	INSERT INTO markets values('mark518','TRIVENDRAM','south')	1 row(s) affected	0.000 sec
✓	70	10:38:41	INSERT INTO markets values('mark519','PATNA','NORTH')	1 row(s) affected	0.000 sec

select * from markets;

OUTPUT

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

markets_code	markets_name	zone
mark500	bangalore	south
mark501	SURAT	NORTH
mark502	Dehli	NORTH
mark503	LATOOR	NORTH
mark504	CHENNAI	south
mark505	RANCHI	NORTH
mark506	MYSORE	south
mark507	BHUVANESHWAR	NORTH
mark508	TELANGANA	south
mark509	GANDHINAGAR	NORTH
mark510	HUBLI	south
mark511	JAIPUR	NORTH
mark512	SELAM	south
mark513	BHUJ	NORTH
mark514	HYDRABAD	south
mark515	JAMMU	NORTH
mark516	VIZAK	south
mark517	MADHUPUR	NORTH
mark518	TRIVENDRAM	south
mark519	PATNA	NORTH
NULL	NULL	NULL

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

INSERT VALUES INTO THE PRODUCTS TABLE

```
insert into products values('prod280','clothing');
insert into products values('prod281','household');
insert into products values('prod282','cleaning');
insert into products values('prod283','clothing');
```

```
insert into products values('prod284','household');
insert into products values('prod285','cleaning');
insert into products values('prod286','clothing');
insert into products values('prod287','household');
insert into products values('prod288','cleaning');
insert into products values('prod289','clothing');
insert into products values('prod290','household');
insert into products values('prod291','cleaning');
insert into products values('prod292','clothing');
insert into products values('prod293','household');
insert into products values('prod294','cleaning');
```

OUTPUT:

72	10:40:37	insert into products values('prod280','clothing')	1 row(s) affected	0.031 sec
73	10:40:37	insert into products values('prod281','household')	1 row(s) affected	0.000 sec
74	10:40:37	insert into products values('prod282','cleaning')	1 row(s) affected	0.000 sec
75	10:40:37	insert into products values('prod283','clothing')	1 row(s) affected	0.015 sec
76	10:40:37	insert into products values('prod284','household')	1 row(s) affected	0.000 sec
77	10:40:37	insert into products values('prod285','cleaning')	1 row(s) affected	0.000 sec
78	10:40:37	insert into products values('prod286','clothing')	1 row(s) affected	0.000 sec
79	10:40:37	insert into products values('prod287','household')	1 row(s) affected	0.000 sec
80	10:40:37	insert into products values('prod288','cleaning')	1 row(s) affected	0.000 sec
81	10:40:37	insert into products values('prod289','clothing')	1 row(s) affected	0.000 sec
82	10:40:37	insert into products values('prod290','household')	1 row(s) affected	0.000 sec
83	10:40:37	insert into products values('prod291','cleaning')	1 row(s) affected	0.000 sec
84	10:40:37	insert into products values('prod292','clothing')	1 row(s) affected	0.016 sec
85	10:40:37	insert into products values('prod293','household')	1 row(s) affected	0.000 sec
86	10:40:37	insert into products values('prod294','cleaning')	1 row(s) affected	0.000 sec

select * from products;

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

product_code	product_type
prod280	clothing
prod281	household
prod282	cleaning
prod283	clothing
prod284	household
prod285	cleaning
prod286	clothing
prod287	household
prod288	cleaning
prod289	clothing
prod290	household
prod291	cleaning
prod292	clothing
prod293	household
prod294	cleaning
NULL	NULL

Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

INSERT VALUES INTO THE TRANSACTIONS TABLE

```
insert into transactions values('prod282','cus101','mark500','2017-06-
30',1,5000,'usd',null,null,4800);

insert into transactions values('prod282','cus100','mark502','2016-07-
28',3,6000,'usd',null,null,5800);

insert into transactions values('prod283','cus102','mark502','2016-06-
25',4,10000,'usd',null,null,9800);

insert into transactions values('prod283','cus103','mark503','2017-11-
30',5,10000,'usd',null,null,9800);
```

```
insert into transactions values('prod284','cus103','mark504','2018-04-20',6,5000,'usd',null,null,4800);

insert into transactions values('prod284','cus104','mark505','2018-02-20',7,5000,'usd',null,null,4800);

insert into transactions values('prod290','cus105','mark506','2018-01-28',2,5000,'usd',null,null,4800);

insert into transactions values('prod290','cus106','mark507','2018-02-15',9,6000,'usd',null,null,5800);

insert into transactions values('prod291','cus107','mark508','2018-02-15',3,5000,'usd',null,null,4800);

insert into transactions values('prod292','cus107','mark509','2016-07-28',1,7000,'usd',null,null,6800);

insert into transactions values('prod292','cus108','mark510','2016-07-28',1,9000,'usd',null,null,8800);

insert into transactions values('prod293','cus109','mark511','2018-04-25',1,8000,'usd',null,null,7800);

insert into transactions values('prod294','cus101','mark512','2018-04-25',1,4000,'usd',null,null,3800);

insert into transactions values('prod294','cus001','mark513','2018-04-25',1,5000,'usd',null,null,4800);
```

✓	88	10:42:30	insert into transactions values('prod282','cus101','mark500','2017-06-30',1,5000,'usd',null,null,4800)	1 row(s) affected	0.000 sec
✓	89	10:42:30	insert into transactions values('prod282','cus100','mark502','2016-07-28',3,6000,'usd',null,null,5800)	1 row(s) affected	0.016 sec
✓	90	10:42:30	insert into transactions values('prod283','cus102','mark502','2016-06-25',4,10000,'usd',null,null,9800)	1 row(s) affected	0.000 sec
✓	91	10:42:30	insert into transactions values('prod283','cus103','mark503','2017-11-30',5,10000,'usd',null,null,9800)	1 row(s) affected	0.000 sec
✓	92	10:42:30	insert into transactions values('prod284','cus103','mark504','2018-04-20',6,5000,'usd',null,null,4800)	1 row(s) affected	0.000 sec
✓	93	10:42:30	insert into transactions values('prod284','cus104','mark505','2018-02-20',7,5000,'usd',null,null,4800)	1 row(s) affected	0.016 sec
✓	94	10:42:30	insert into transactions values('prod290','cus105','mark506','2018-01-28',2,5000,'usd',null,null,4800)	1 row(s) affected	0.000 sec
✓	95	10:42:30	insert into transactions values('prod290','cus106','mark507','2018-02-15',9,6000,'usd',null,null,5800)	1 row(s) affected	0.000 sec
✓	96	10:42:30	insert into transactions values('prod291','cus107','mark508','2018-02-15',3,5000,'usd',null,null,4800)	1 row(s) affected	0.000 sec
✓	97	10:42:30	insert into transactions values('prod292','cus107','mark509','2016-07-28',1,7000,'usd',null,null,6800)	1 row(s) affected	0.015 sec
✓	98	10:42:30	insert into transactions values('prod292','cus108','mark510','2016-07-28',1,9000,'usd',null,null,8800)	1 row(s) affected	0.000 sec
✓	99	10:42:30	insert into transactions values('prod293','cus109','mark511','2018-04-25',1,8000,'usd',null,null,7800)	1 row(s) affected	0.000 sec
✓	100	10:42:30	insert into transactions values('prod294','cus101','mark512','2018-04-25',1,4000,'usd',null,null,3800)	1 row(s) affected	0.000 sec
✓	101	10:42:30	insert into transactions values('prod294','cus001','mark513','2018-04-25',1,5000,'usd',null,null,4800)	1 row(s) affected	0.016 sec

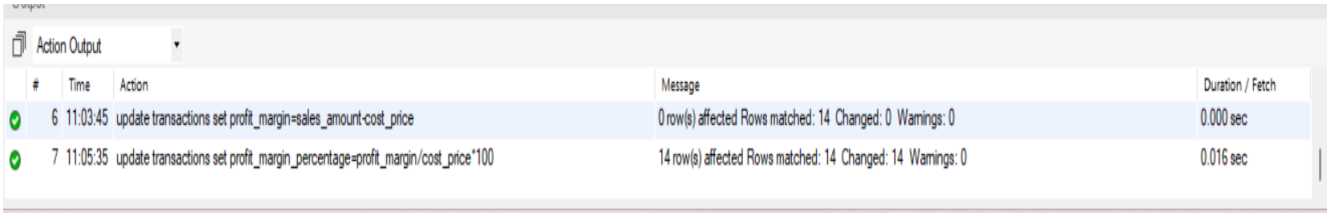
```
SET SQL_SAFE_UPDATES = 0;
```

✓	102	10:43:31	SET SQL_SAFE_UPDATES = 0	0 row(s) affected	0.000 sec

CALCULATE THE COLUMNS OF profit_margin and profit_margin_percentage.

```
update transactions
set profit_margin=sales_amount-cost_price;
```


```
update transactions
set profit_margin_percentage=profit_margin/cost_price*100;
```

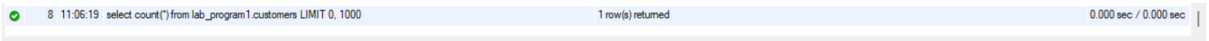


QUERY 1

WRITE AN SQL QUERY TO GET THE TOTAL NUMBER OF CUSTOMERS IN SALES DATABASE

```
select count(*) from lab_program1.customers;
```

OUTPUT

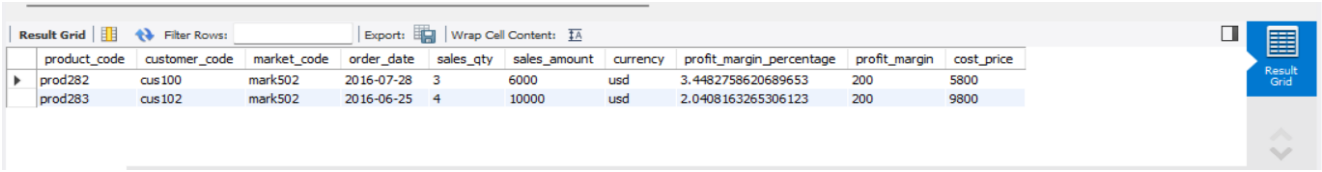


QUERY 2

WRITE AN SQL QUERY TO GET THE COMPLETE DETAILS OF TRANSACTIONS FOR MARKET CODE 'mark502'

```
select * from lab_program1.transactions
where market_code='mark502';
```

OUTPUT



product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	profit_margin_percentage	profit_margin	cost_price
prod282	cus100	mark502	2016-07-28	3	6000	usd	3.4482758620689653	200	5800
prod283	cus102	mark502	2016-06-25	4	10000	usd	2.0408163265306123	200	9800

QUERY 3

WRITE AN SQL QUERY TO GET THE TOTAL NUMBER OF TRANSACTIONS FOR MARKET CODE 'mark502'

```
select count(*) from lab_program1.transactions
where market_code='mark502'
```

OUTPUT

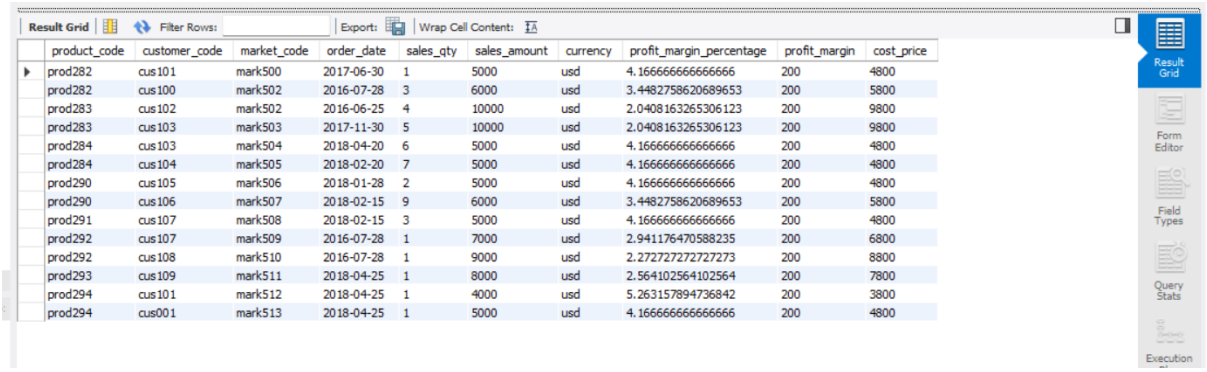
count(*)
2

QUERY 4

WRITE AN SQL QUERY TO GET THE DETAILS OF TRANSACTIONS INVOLVING US DOLLER CURRENCY

```
select * from lab_program1.transactions
where currency='USD'
```

OUTPUT

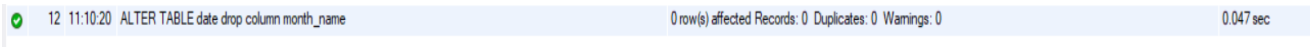


product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	profit_margin_percentage	profit_margin	cost_price
prod282	cus101	mark500	2017-06-30	1	5000	usd	4.166666666666666	200	4800
prod282	cus100	mark502	2016-07-28	3	6000	usd	3.4482758620689653	200	5800
prod283	cus102	mark502	2016-06-25	4	10000	usd	2.0408163265306123	200	9800
prod283	cus103	mark503	2017-11-30	5	10000	usd	2.0408163265306123	200	9800
prod284	cus103	mark504	2018-04-20	6	5000	usd	4.166666666666666	200	4800
prod284	cus104	mark505	2018-02-20	7	5000	usd	4.166666666666666	200	4800
prod290	cus105	mark506	2018-01-28	2	5000	usd	4.166666666666666	200	4800
prod290	cus106	mark507	2018-02-15	9	6000	usd	3.4482758620689653	200	5800
prod291	cus107	mark508	2018-02-15	3	5000	usd	4.166666666666666	200	4800
prod292	cus107	mark509	2016-07-28	1	7000	usd	2.941176470588235	200	6800
prod292	cus108	mark510	2016-07-28	1	9000	usd	2.272727272727273	200	8800
prod293	cus109	mark511	2018-04-25	1	8000	usd	2.564102564102564	200	7800
prod294	cus101	mark512	2018-04-25	1	4000	usd	5.263157894736842	200	3800
prod294	cus001	mark513	2018-04-25	1	5000	usd	4.166666666666666	200	4800

QUERY 5

WRITE AN SQL QUERY TO DELETE A COLUMN

```
ALTER TABLE date
drop column month_name;
```



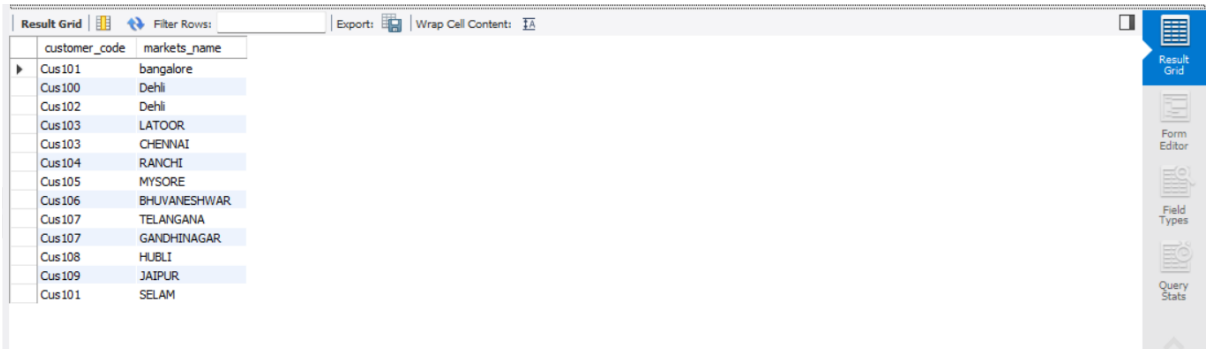
12 11:10:20 ALTER TABLE date drop column month_name — 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0 — 0.047 sec

QUERY 6

##WRITE AN SQL QUERY TO GET THE DETAILS OF CUSTOMER_CODE AND MARKET NAME

```
SELECT customers.customer_code,markets.markets_name
from customers,markets,transactions
where customers.customer_code=transactions.customer_code
and markets.markets_code=transactions.market_code;
```

OUTPUT



customer_code	markets_name
Cus101	bangalore
Cus100	Dehli
Cus102	Dehli
Cus103	LATOOR
Cus103	CHENNAI
Cus104	RANCHI
Cus105	MYSORE
Cus106	BHUVANESHWAR
Cus107	TELANGANA
Cus107	GANDHINAGAR
Cus108	HUBLI
Cus109	JAIPUR
Cus101	SELAM

SET SQL_MODE = '';

##

##year wise sales analysis

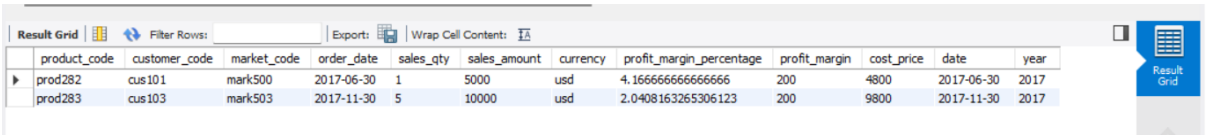
```
select lab_program1.transactions.*, lab_program1.date.*
from lab_program1.transactions INNER JOIN lab_program1.date
ON lab_program1.transactions.order_date=lab_program1.date.date
where lab_program1.date.year=2016;
```

OUTPUT

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	profit_margin_percentage	profit_margin	cost_price	date	year
prod282	cus100	mark502	2016-07-28	3	6000	usd	3.4482758620689653	200	5800	2016-07-28	2016
prod283	cus102	mark502	2016-06-25	4	10000	usd	2.0408163265306123	200	9800	2016-06-25	2016
prod292	cus107	mark509	2016-07-28	1	7000	usd	2.941176470588235	200	6800	2016-07-28	2016
prod292	cus108	mark510	2016-07-28	1	9000	usd	2.272727272727273	200	8800	2016-07-28	2016

```
select lab_program1.transactions.*, lab_program1.date.*
from lab_program1.transactions INNER JOIN lab_program1.date
ON lab_program1.transactions.order_date=lab_program1.date.date
where lab_program1.date.year=2017;
```

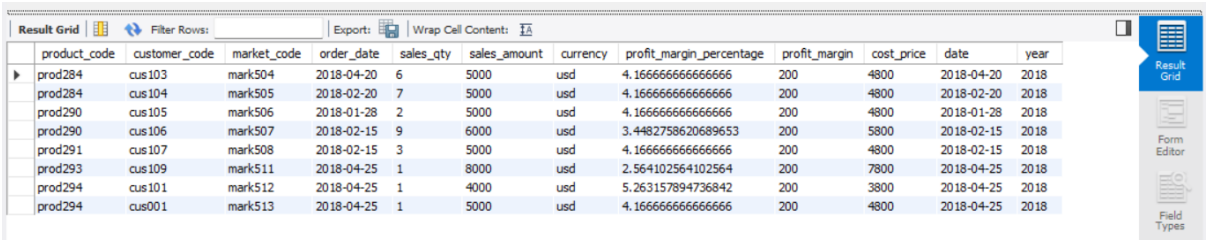
OUTPUT



product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	profit_margin_percentage	profit_margin	cost_price	date	year
prod282	cus101	mark500	2017-06-30	1	5000	usd	4.166666666666666	200	4800	2017-06-30	2017
prod283	cus103	mark503	2017-11-30	5	10000	usd	2.0408163265306123	200	9800	2017-11-30	2017

```
select lab_program1.transactions.*, lab_program1.date.*
from lab_program1.transactions INNER JOIN lab_program1.date
ON lab_program1.transactions.order_date=lab_program1.date.date
where lab_program1.date.year=2018;
```

OUTPUT



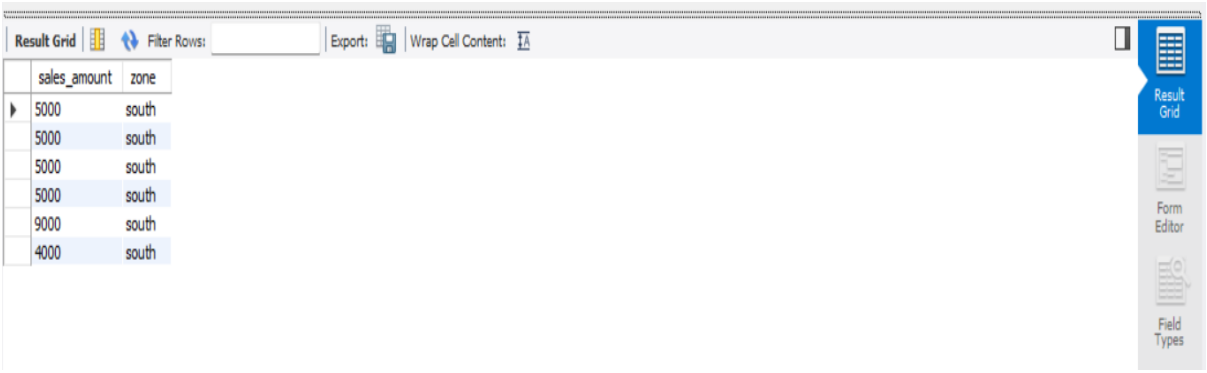
product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	profit_margin_percentage	profit_margin	cost_price	date	year
prod284	cus103	mark504	2018-04-20	6	5000	usd	4.166666666666666	200	4800	2018-04-20	2018
prod284	cus104	mark505	2018-02-20	7	5000	usd	4.166666666666666	200	4800	2018-02-20	2018
prod290	cus105	mark506	2018-01-28	2	5000	usd	4.166666666666666	200	4800	2018-01-28	2018
prod290	cus106	mark507	2018-02-15	9	6000	usd	3.4482758620689653	200	5800	2018-02-15	2018
prod291	cus107	mark508	2018-02-15	3	5000	usd	4.166666666666666	200	4800	2018-02-15	2018
prod293	cus109	mark511	2018-04-25	1	8000	usd	2.564102564102564	200	7800	2018-04-25	2018
prod294	cus101	mark512	2018-04-25	1	4000	usd	5.263157894736842	200	3800	2018-04-25	2018
prod294	cus001	mark513	2018-04-25	1	5000	usd	4.166666666666666	200	4800	2018-04-25	2018

##

zone wise data analysis

```
select lab_program1.transactions.sales_amount, lab_program1.markets.zone
from lab_program1.transactions INNER JOIN lab_program1.markets
ON lab_program1.transactions.market_code=lab_program1.markets.markets_code
where lab_program1.markets.zone='south'
```

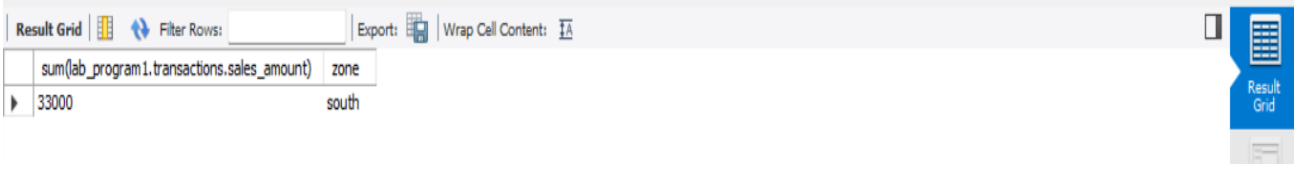
OUTPUT



sales_amount	zone
5000	south
5000	south
5000	south
5000	south
9000	south
4000	south

```
select sum(lab_program1.transactions.sales_amount), lab_program1.markets.zone
from lab_program1.transactions INNER JOIN lab_program1.markets
ON lab_program1.transactions.market_code=lab_program1.markets.markets_code
where lab_program1.markets.zone='south'
```

OUTPUT



sum(lab_program1.transactions.sales_amount)	zone
33000	south

```
select sum(lab_program1.transactions.sales_amount), lab_program1.markets.zone
from lab_program1.transactions INNER JOIN lab_program1.markets
ON lab_program1.transactions.market_code=lab_program1.markets.markets_code
where lab_program1.markets.zone='north'
```

OUTPUT

sum(lab_program1.transactions.sales_amount)	zone
57000	NORTH

LAB PROGRAM 2

EMPLOYEE DATABASE

Consider the following relational schema of Employee and Department, create the tables and by inserting the appropriate data values, the SQL Query for the following.

Department2

DEPT_ID	DEPT_NAME	DEPT_LOCATION

Emp2

emp_id	emp_name	emp_phone_no	emp_dept	job	hourly_pay	salary	Dept_id

QUERY 1- USING SAFE SQL UPDATE COMMAND WRITE AN SQL QUERY TO UPDATE THE SALARY COLUMN USING THE FORMULA *salary = hourly_pay*2080;*

QUERY 2- WRITE A TRIGGER TO UPDATE THE SALARY COLUMN AUTOMATICALLY WHEN THE HOURLY_PAY IS UPDATED. ALSO UPDATE THE HOURLY PAY OF EACH EMPLOYEE BY 50 RS AND CHECK WHETHER THE TRIGGER IS WORKING TO UPDATE THE CORRESSPONDING SALARY COLUMN.

QUERY 3- CREATE TRIGGER TO AUTOMATICALLY CALCULATE THE SALARY OF THE NEW EMPLOYEE ADDED TO THE ORGANIZATION. ALSO INSERT A NEW RECORD AND EXAMINE THE WORKING OF TRIGGER.

QUERY 4- WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME STARTS WITH 'J'.

QUERY 5- WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME CONTAINS LETTER 'a'

QUERY 6- WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME ENDS WITH 'S'

QUERY 7- WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME whose name contains exactly three characters

QUERY 8- WRITE AN SQL QUERY TO CREATE A VIEW OF EMPLOYEE NAME AND THEIR RESPECTIVE LOCATIONS.

SCHEMA DIAGRAM

DATABASE CREATION

create database lab_program2
use lab_program2

// TABLE CREATION

create table department2
(
dept_id int primary key,
dept_name varchar(30),
dept_location varchar(20)
);

Create table emp2
(
emp_id varchar(20) primary key,
emp_name varchar(20),
emp_phone_no varchar(30),
emp_dept varchar(20),
job varchar(20),
hourly_pay double,
salary double,
dept_id int,
FOREIGN KEY (dept_id) REFERENCES department2(dept_id)
);
Desc employee2
select * from employee2

OUTPUT

INSERT VALUES INTO THE TABLES

```
insert into department2 values(1,'sales','Dehli'),
(2,'claims','mumbai'),
(3,'developer','Bangalore'),
(4,'facility','Bangalore');
```

OUTPUT

```
7  14:50:10  insert into department2 values(1,'sales','Dehli'), (2,'claims','mumbai'), (3,'developer','Bangalore'), (4,'facility','Bangalore')  4 row(s) affected Records: 4  Duplicates: 0  Warnings: 0
```

```
SET SQL_SAFE_UPDATES = 0;
```

```
INSERT INTO emp2 VALUES(011,'JOE',228899887,'SALES','ASSISTANT',100,NULL,1),
(012,'JASS',228899882,'CLAIMS','MANAGER',400,NULL,2),
(013,'JOA',228899880,'DEVELOPER','TEAM LEAD',300,NULL,3),
(014,'JOHN',228899884,'FACILITY','HEL
PER',60,NULL,4);
```

OUTPUT

```
9  14:51:22  INSERT INTO emp2 VALUES(011,'JOE',228899887,'SALES','ASSISTANT',100,NULL,1), (012,'JASS',228899882,'C...  4 row(s) affected Records: 4  Duplicates: 0  Warnings: 0
```

TO DISPLAY THE DATA OF THE TABLES

```
select * from emp2;
```

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

emp_id	emp_name	emp_phone_no	emp_dept	job	hourly_pay	salary	dept_id
11	JOE	228899887	SALES	ASSISTANT	100	NULL	1
12	JASS	228899882	CLAIMS	MANAGER	400	NULL	2
13	JOA	228899880	DEVELOPER	TEAM LEAD	300	NULL	3
14	JOHN	228899884	FACILITY	HEL PER	60	NULL	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Result Grid

```
select * from department2
```

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

DEPT_ID	DEPT_NAME	DEPT_LOCATION
1	sales	Dehli
2	claims	mumbai
3	developer	Bangalore
4	facility	Bangalore
NULL	NULL	NULL

Result Grid

QUERY 1-### 40 hours in a work week and 52 weeks in an year

40*52=2080 work hours in an year

```
SET SQL_SAFE_UPDATES = 0;
update emp2
set salary = hourly_pay*2080;
```

OUTPUT

```
13  14:54:59  update emp2 set salary = hourly_pay*2080        4 row(s) affected Rows matched: 4  Changed: 4  Warnings: 0
```

select * from emp2

```
53 •  select * from emp2
```

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

emp_id	emp_name	emp_phone_no	emp_dept	job	hourly_pay	salary	dept_id
11	JOE	228899887	SALES	ASSISTANT	100	208000	1
12	JASS	228899882	CLAIMS	MANAGER	400	832000	2
13	JOA	228899880	DEVELOPER	TEAM LEAD	300	624000	3
14	JOHN	228899884	FACILITY	HEL PER	60	124800	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

emp2 3

###QUERY 2-WRITE A TRIGGER TO UPDATE THE SALARY COLUMN AUTOMATICALLY WHEN THE HOURLY_PAY IS UPDATED

```
CREATE TRIGGER HOURLY_PAY_UPDATE
BEFORE UPDATE ON EMP2
FOR EACH ROW
SET NEW.SALARY=(NEW.HOURLY_PAY*2080);
```

```
15  14:56:36  CREATE TRIGGER HOURLY_PAY_UPDATE BEFORE UPDATE ON EMP2 FOR EACH ROW SET NEW.SA...  0 row(s) affected
```

SHOW TRIGGERS

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

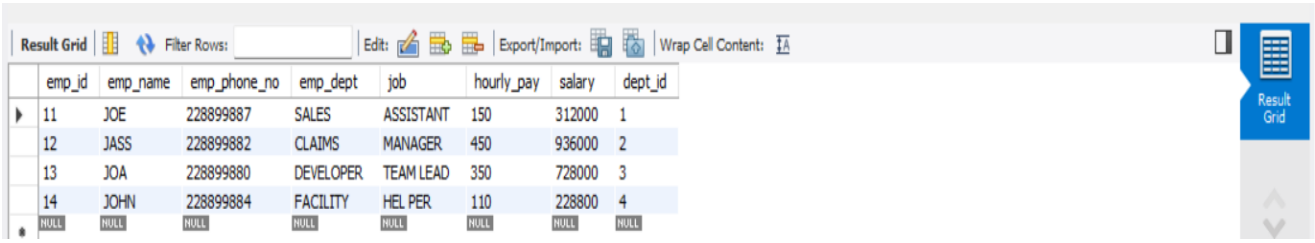
Trigger	Event	Table	Statement	Timing	Created	sql_mode	Definer
HOURLY_PAY_UPDATE	UPDATE	emp2	SET NEW.SALARY=(NEW.HOURLY_PAY*2080)	BEFORE	2024-07-19 14:56:36.80	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@localhost

LETS UPDATE THE HOURLY PAY OF EACH EMPLOYEE BY 50 RS

```
UPDATE EMP2
SET HOURLY_PAY=HOURLY_PAY+50
```

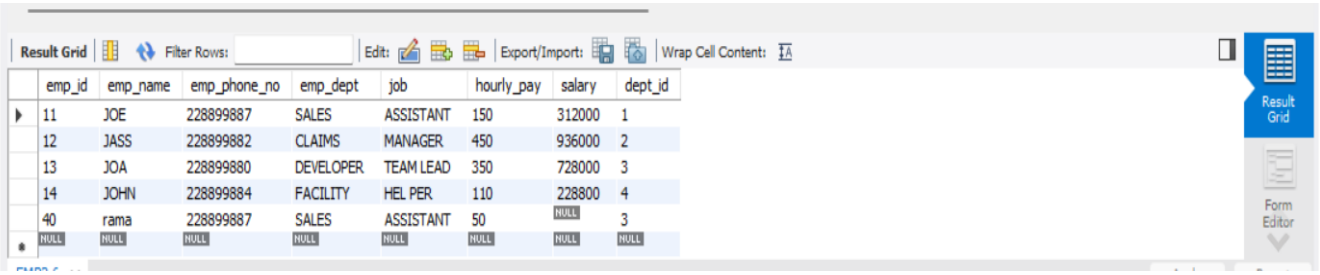
```
17  14:58:06  UPDATE EMP2 SET HOURLY_PAY=HOURLY_PAY+50        4 row(s) affected Rows matched: 4  Changed: 4  Warnings: 0
```

SELECT * FROM EMP2



emp_id	emp_name	emp_phone_no	emp_dept	job	hourly_pay	salary	dept_id
11	JOE	228899887	SALES	ASSISTANT	150	312000	1
12	JASS	228899882	CLAIMS	MANAGER	450	936000	2
13	JOA	228899880	DEVELOPER	TEAM LEAD	350	728000	3
14	JOHN	228899884	FACILITY	HEL PER	110	228800	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

INSERT INTO emp2 VALUES(40,'rama',228899887,'SALES','ASSISTANT',50,NULL,3);
SELECT * FROM EMP2



emp_id	emp_name	emp_phone_no	emp_dept	job	hourly_pay	salary	dept_id
11	JOE	228899887	SALES	ASSISTANT	150	312000	1
12	JASS	228899882	CLAIMS	MANAGER	450	936000	2
13	JOA	228899880	DEVELOPER	TEAM LEAD	350	728000	3
14	JOHN	228899884	FACILITY	HEL PER	110	228800	4
40	rama	228899887	SALES	ASSISTANT	50	NULL	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

###QUERY 3: CREATE TRIGGER TO AUTOMATICALLY CALCULATE THE SALARY OF THE NEW EMPLOYEE ADDED TO THE ORGANIZATION

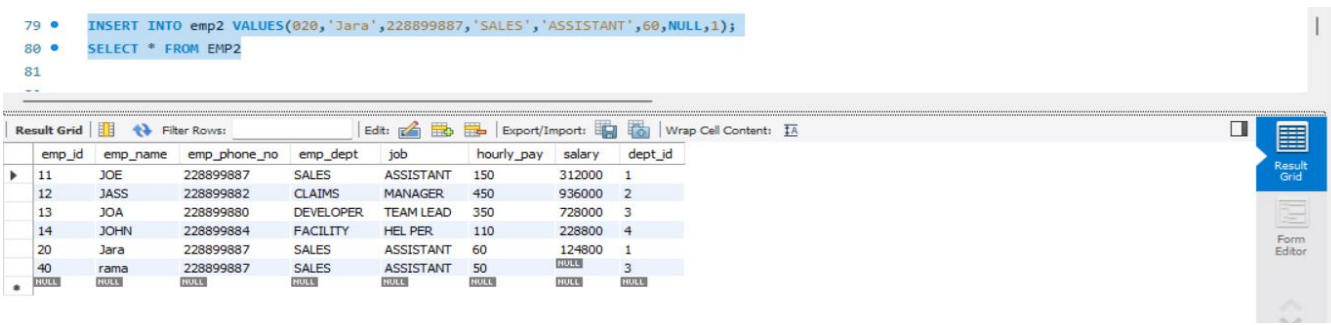
CREATE TRIGGER BEFORE_HOURLY_PAY_INSERT
BEFORE INSERT ON EMP2
FOR EACH ROW
SET NEW.SALARY=(NEW.hourly_pay*2080)

show triggers

```
71    ###CREATE TRIGGER TO AUTOMATICALLY CALCULATE THE SALARY OF THE NEW EMPLOYEE ADDED TO THE ORGANIZATION
72    CREATE TRIGGER BEFORE_HOURLY_PAY_INSERT
73    BEFORE INSERT ON EMP2
74    FOR EACH ROW
75    SET NEW.SALARY=(NEW.hourly_pay*2080);
76
```

Trigger	Event	Table	Statement	Timing	Created	sql_mode	Definer
BEFORE_HOURLY_PAY_INSERT	INSERT	emp2	SET NEW.SALARY=(NEW.hourly_pay*2080)	BEFORE	2024-07-19 15:00:55.20	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@loca
HOURLY_PAY_UPDATE	UPDATE	emp2	SET NEW.SALARY=(NEW.HOURLY_PAY*2080)	BEFORE	2024-07-19 14:56:36.80	ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLE...	root@loca

INSERT INTO emp2 VALUES(020,'Jara',228899887,'SALES','ASSISTANT',60,NULL,1);
SELECT * FROM EMP2

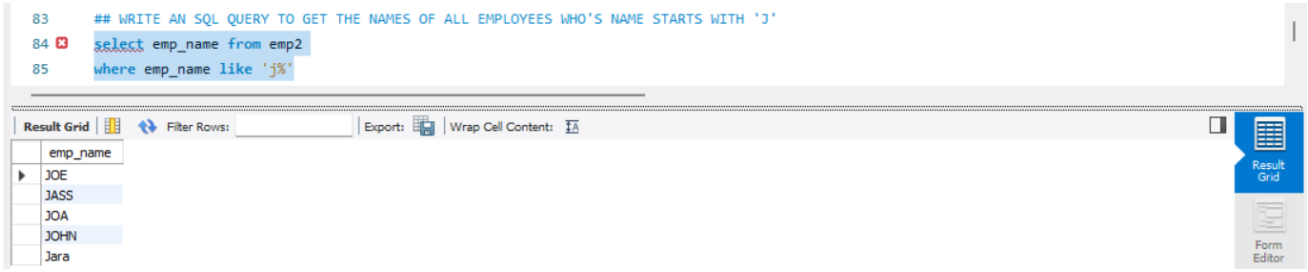


```
79    INSERT INTO emp2 VALUES(020,'Jara',228899887,'SALES','ASSISTANT',60,NULL,1);
80    SELECT * FROM EMP2
81
```

emp_id	emp_name	emp_phone_no	emp_dept	job	hourly_pay	salary	dept_id
11	JOE	228899887	SALES	ASSISTANT	150	312000	1
12	JASS	228899882	CLAIMS	MANAGER	450	936000	2
13	JOA	228899880	DEVELOPER	TEAM LEAD	350	728000	3
14	JOHN	228899884	FACILITY	HEL PER	110	228800	4
20	Jara	228899887	SALES	ASSISTANT	60	124800	1
40	rama	228899887	SALES	ASSISTANT	50	NULL	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

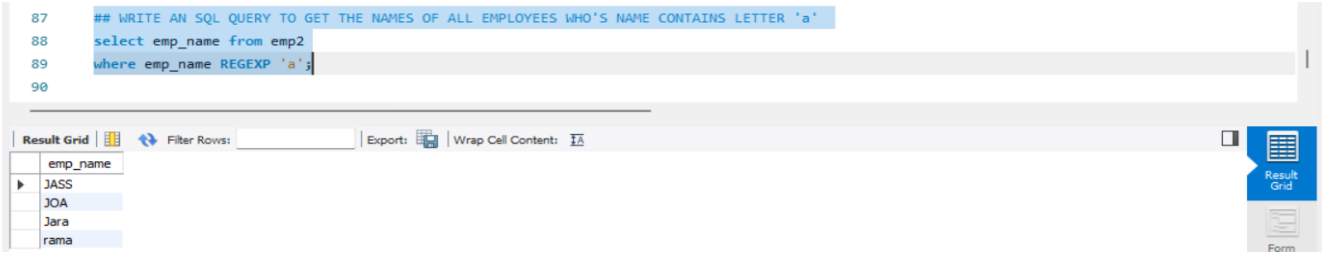
##QUERY 4: WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME STARTS WITH 'J'

select emp_name from emp2
where emp_name like 'j%'



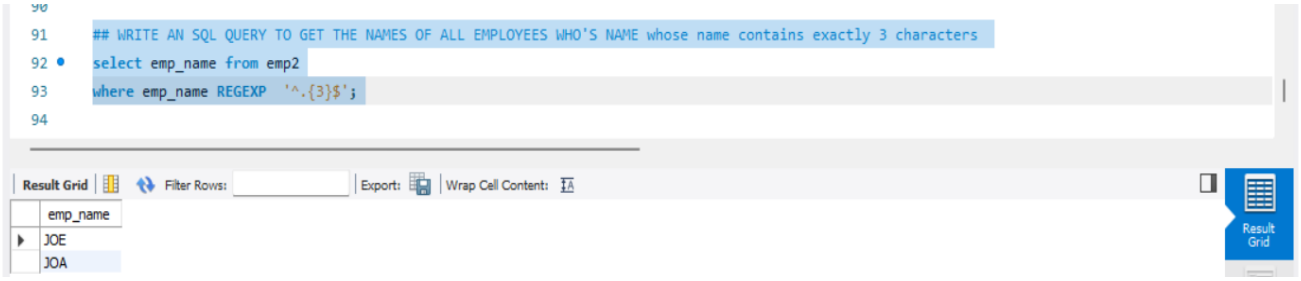
##QUERY 5: WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME CONTAINS LETTER 'a'

select emp_name from emp2
where emp_name REGEXP 'a';



####QUERY 5: WRITE AN SQL QUERY TO GET THE NAMES OF ALL EMPLOYEES WHO'S NAME whose name contains exactly THREE characters

select emp_name from emp2
where emp_name REGEXP '^.{3}$';



##QUERY 6: WRITE AN SQL QUERY TO CREATE A VIEW OF EMPLOYEE NAME AND THEIR RESPECTIVE LOCATIONS

CREATE VIEW EMP_LOCATION
AS
SELECT EMP_NAME,DEPT_LOCATION

FROM EMP2,DEPARTMENT2
WHERE DEPARTMENT2.DEPT_ID=EMP2.DEPT_ID;

```
95    ### WRITE AN SQL QUERY TO CREATE A VIEW OF EMPLOYEE NAME AND THEIR RESPECTIVE LOCATIONS
96    CREATE VIEW EMP_LOCATION
97    AS
98    SELECT EMP_NAME,DEPT_LOCATION
99    FROM EMP2,DEPARTMENT2
100   WHERE DEPARTMENT2.DEPT_ID=EMP2.DEPT_ID;
101
102   SELECT * FROM EMP_LOCATION
103
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

EMP_NAME	DEPT_LOCATION
JOE	Dehli
Jara	Dehli
JASS	mumbai
JOA	Bangalore
rama	Bangalore
JOHN	Bangalore

Result Grid
Form Editor

LAB PROGRAM 3

ORGANIZATION DATABASE

Consider the following organization database. Draw the schema diagram and Write the following Nested Queries.

EMPLOYEE(fname,minit,Lname,SSN, Birthdate,Address, Gender, Salary, SuperSSN, DNo)
DEPARTMENT(DNumber, DName, MgrSSN, MgrStartDate)
DLOCATION(DNumber,,DLocation)
PROJECT(PNumber, PName, PLocation, DNum)
WORKS_ON(ESSN, PNo, Hours)
DEPENDENT(Essn,Dependent_name,Gender, Bdate, Relationship)

Query 1 : Retrieve the name and address of all employees who work for the 'Research' department.

###Query 2: Retrieve the name of each employee who has a dependent with the same first name as the employee.

###QUERY 3: write a SQL query to find those employees whose salary matches the lowest salary of any of the departments. Return first name, last name and department ID.

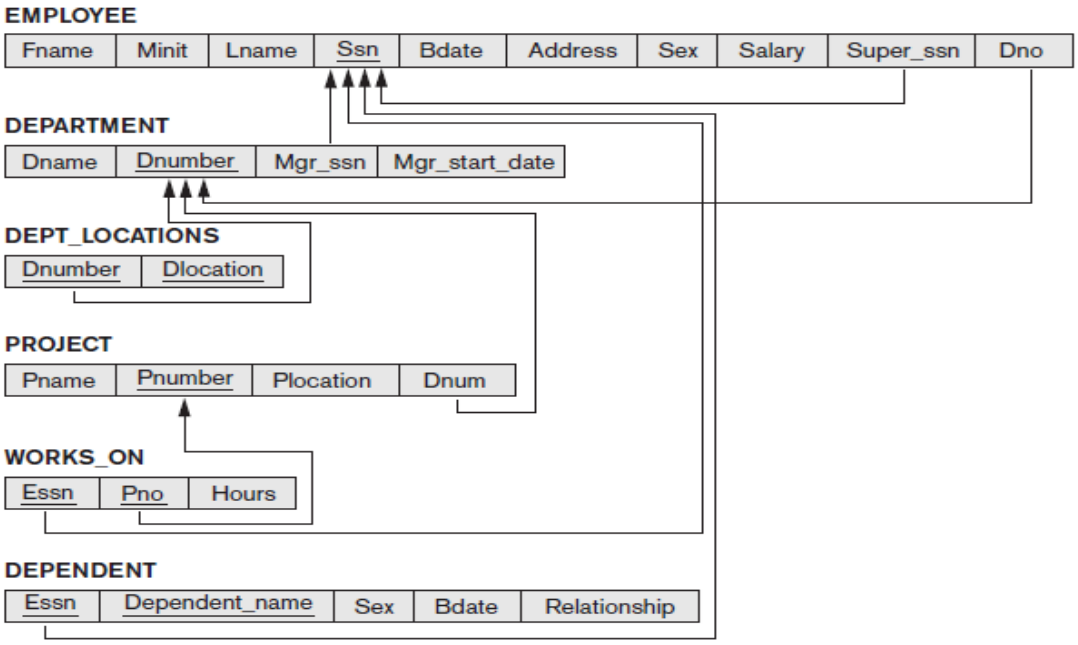
###QUERY 4: write a SQL query to find those employees who do not work in the departments of 1,2,3 (Begin and end values are included.). Return all the fields of the employees.

###QUERY 5:Write a SQL query to find those employees whose department is located at 'Stafford'. Return first name, last name AND SSN

QUERY 6: Retrieve the names of employees who have no dependents.

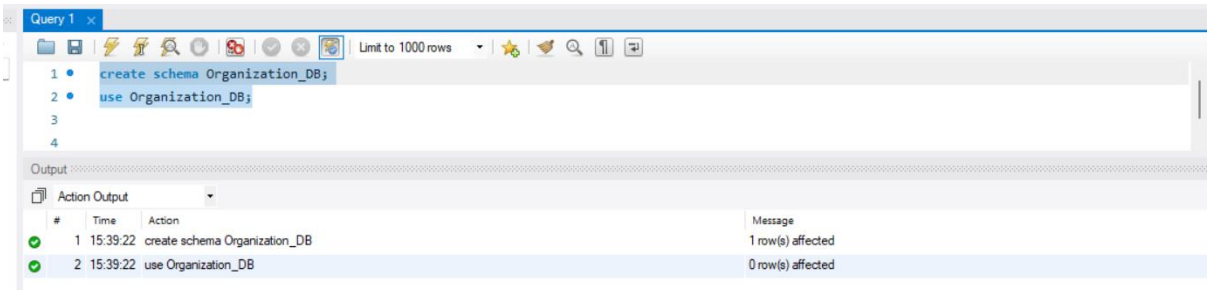
SCHEMA DIAGRAM

Referential integrity constraints displayed on the COMPANY relational database schema.



Create database

create Database Organization_DB;
use Organization_DB;



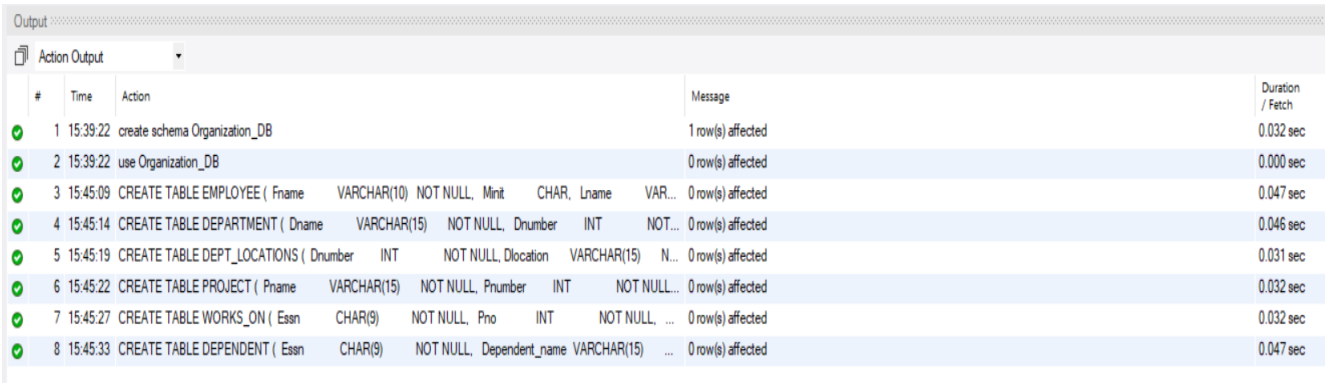
Create tables as given in the schema

```
CREATE TABLE EMPLOYEE
(
 Fname          VARCHAR(10)   NOT NULL,
 Minit          CHAR,
 Lname          VARCHAR(20)     NOT NULL,
 Ssn            CHAR(9)         NOT NULL,
 Bdate          DATE,
 Address        VARCHAR(30),
 gender           CHAR(1),
 Salary         DECIMAL(5),
 Super_ssn      CHAR(9),
 Dno            INT             NOT NULL,
PRIMARY KEY   (Ssn)
);
```

```
CREATE TABLE DEPARTMENT
(
Dname          VARCHAR(15)      NOT NULL,
 Dnumber        INT              NOT NULL,
 Mgr_ssn        CHAR(9)          NOT NULL,
 Mgr_start_date  DATE,
PRIMARY KEY (Dnumber),
UNIQUE     (Dname),
FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
);

CREATE TABLE DEPT_LOCATIONS
(
Dnumber        INT              NOT NULL,
Dlocation      VARCHAR(15)      NOT NULL,
PRIMARY KEY (Dnumber, Dlocation),
FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
 );

CREATE TABLE PROJECT
(
 Pname          VARCHAR(15)      NOT NULL,
 Pnumber        INT              NOT NULL,
 Plocation      VARCHAR(15),
 Dnum           INT              NOT NULL,
PRIMARY KEY (Pnumber),
UNIQUE     (Pname),
FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber)
 );
CREATE TABLE WORKS_ON
(
Essn           CHAR(9)          NOT NULL,
 Pno            INT              NOT NULL,
 Hours          DECIMAL(3,1)     NOT NULL,
PRIMARY KEY (Essn, Pno),
FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber)
);

CREATE TABLE DEPENDENT
(
 Essn           CHAR(9)          NOT NULL,
 Dependent_name  VARCHAR(15)      NOT NULL,
 Gender          CHAR,
```

```
 Bdate          DATE,
 Relationship   VARCHAR(8),
PRIMARY KEY (Essn, Dependent_name),
FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn)
);
```

Outputs for all the tables creation



Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	15:39:22	create schema Organization_DB	1 row(s) affected	0.032 sec
2	15:39:22	use Organization_DB	0 row(s) affected	0.000 sec
3	15:45:09	CREATE TABLE EMPLOYEE (Fname VARCHAR(10) NOT NULL, Minit CHAR, Lname VAR...	0 row(s) affected	0.047 sec
4	15:45:14	CREATE TABLE DEPARTMENT (Dname VARCHAR(15) NOT NULL, Dnumber INT NOT...	0 row(s) affected	0.046 sec
5	15:45:19	CREATE TABLE DEPT_LOCATIONS (Dnumber INT NOT NULL, Dlocation VARCHAR(15) N...	0 row(s) affected	0.031 sec
6	15:45:22	CREATE TABLE PROJECT (Pname VARCHAR(15) NOT NULL, Pnumber INT NOT NULL...	0 row(s) affected	0.032 sec
7	15:45:27	CREATE TABLE WORKS_ON (Essn CHAR(9) NOT NULL, Pno INT NOT NULL, ...	0 row(s) affected	0.032 sec
8	15:45:33	CREATE TABLE DEPENDENT (Essn CHAR(9) NOT NULL, Dependent_name VARCHAR(15) ...	0 row(s) affected	0.047 sec

```
INSERT INTO EMPLOYEE VALUES     ('John','B','Smith',123456789,'1965-01-09','731 Fondren, Houston TX','M',30000,333445555,5),

        ('Franklin','T','Wong',333445555,'1965-12-08','638 Voss, Houston TX','M',40000,888665555,5),

        ('Alicia','J','Zelaya',999887777,'1968-01-19','3321 Castle, Spring TX','F',25000,987654321,4),

        ('Jennifer','S','Wallace',987654321,'1941-06-20','291 Berry, Bellaire TX','F',43000,888665555,4),

        ('Ramesh','K','Narayan',666884444,'1962-09-15','975 Fire Oak, Humble TX','M',38000,333445555,5),

        ('Joyce','A','English',453453453,'1972-07-31','5631 Rice, Houston TX','F',25000,333445555,5),

        ('Ahmad','V','Jabbar',987987987,'1969-03-29','980 Dallas, Houston TX','M',25000,987654321,4),

        ('James','E','Borg',888665555,'1937-11-10','450 Stone, Houston TX','M',55000,null,1);

select * from employee;
```

```
INSERT INTO EMPLOYEE
VALUES      ('John','B','Smith',123456789,'1965-01-09','731 Fondren, Houston TX','M',30000,333445555,5),
            ('Franklin','T','Wong',333445555,'1965-12-08','638 Voss, Houston TX','M',40000,888665555,5),
            ('Alicia','J','Zelaya',999887777,'1968-01-19','3321 Castle, Spring TX','F',25000,987654321,4),
            ('Jennifer','S','Wallace',987654321,'1941-06-20','291 Berry, Bellaire TX','F',43000,888665555,4),
            ('Ramesh','K','Narayan',666884444,'1962-09-15','975 Fire Oak, Humble TX','M',38000,333445555,5),
            ('Joyce','A','English',453453453,'1972-07-31','5631 Rice, Houston TX','F',25000,333445555,5),
            ('Ahmad','V','Jabbar',987987987,'1969-03-29','980 Dallas, Houston TX','M',25000,987654321,4),
            ('James','E','Borg',888665555,'1937-11-10','450 Stone, Houston TX','M',55000,null,1);
select * from employee;
```

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

Fname	Minit	Lname	Ssn	Bdate	Address	gender	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1965-12-08	638 Voss, Houston TX	M	40000	888665555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston TX	F	25000	333445555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble TX	M	38000	333445555	5
James	E	Borg	888665555	1937-11-10	450 Stone, Houston TX	M	55000	NULL	1
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire TX	F	43000	888665555	4
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston TX	M	25000	987654321	4
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring TX	F	25000	987654321	4

Result Grid | Form Editor

INSERT INTO DEPARTMENT
VALUES ('Research',5,333445555,'1988-05-22'),
('Administration',4,987654321,'1995-01-01'),
('Headquarters',1,888665555,'1981-06-19');
select * from Department;

```
INSERT INTO DEPARTMENT
VALUES      ('Research',5,333445555,'1988-05-22'),
            ('Administration',4,987654321,'1995-01-01'),
            ('Headquarters',1,888665555,'1981-06-19');

select * from Department;
```

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

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Headquarters	1	888665555	1981-06-19
Administration	4	987654321	1995-01-01
Research	5	333445555	1988-05-22

Result Grid | Form

INSERT INTO PROJECT VALUES ('ProductX',1,'Bellaire',5),
('ProductY',2,'Sugarland',5),
('ProductZ',3,'Houston',5),
('Computerization',10,'Stafford',4),
('Reorganization',20,'Houston',1),
('Newbenefits',30,'Stafford',4);

```
114
115 •  INSERT INTO PROJECT
116    VALUES      ('ProductX',1,'Bellaire',5),
117                ('ProductY',2,'Sugarland',5),
118                ('ProductZ',3,'Houston',5),
119                ('Computerization',10,'Stafford',4),
120                ('Reorganization',20,'Houston',1),
121                ('Newbenefits',30,'Stafford',4);
122 •  select * from project
123
```

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

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4
NULL	NULL	NULL	NULL

Result Grid | Form Editor

INSERT INTO WORKS_ON VALUES (123456789,1,32.5),
(123456789,2,7.5),
(666884444,3,40.0),
(453453453,1,20.0),
(453453453,2,20.0),
(333445555,2,10.0),
(333445555,3,10.0),
(333445555,10,10.0),
(333445555,20,10.0),
(999887777,30,30.0),
(999887777,10,10.0),
(987987987,10,35.0),
(987987987,30,5.0),
(987654321,30,20.0),
(987654321,20,15.0),
(888665555,20,16.0);

```
124 ⊗  INSERT INTO WORKS_ON
125    VALUES      (123456789,1,32.5),
126                (123456789,2,7.5),
127                (666884444,3,40.0),
128                (453453453,1,20.0),
129                (453453453,2,20.0),
130                (333445555,2,10.0),
131                (333445555,3,10.0),
132                (333445555,10,10.0),
133                (333445555,20,10.0),
134                (999887777,30,30.0),
135                (999887777,10,10.0),
136                (987987987,10,35.0),
137                (987987987,30,5.0),
138                (987654321,30,20.0),
139                (987654321,20,15.0),
140                (888665555,20,16.0);
141 •  select * from works_on
```

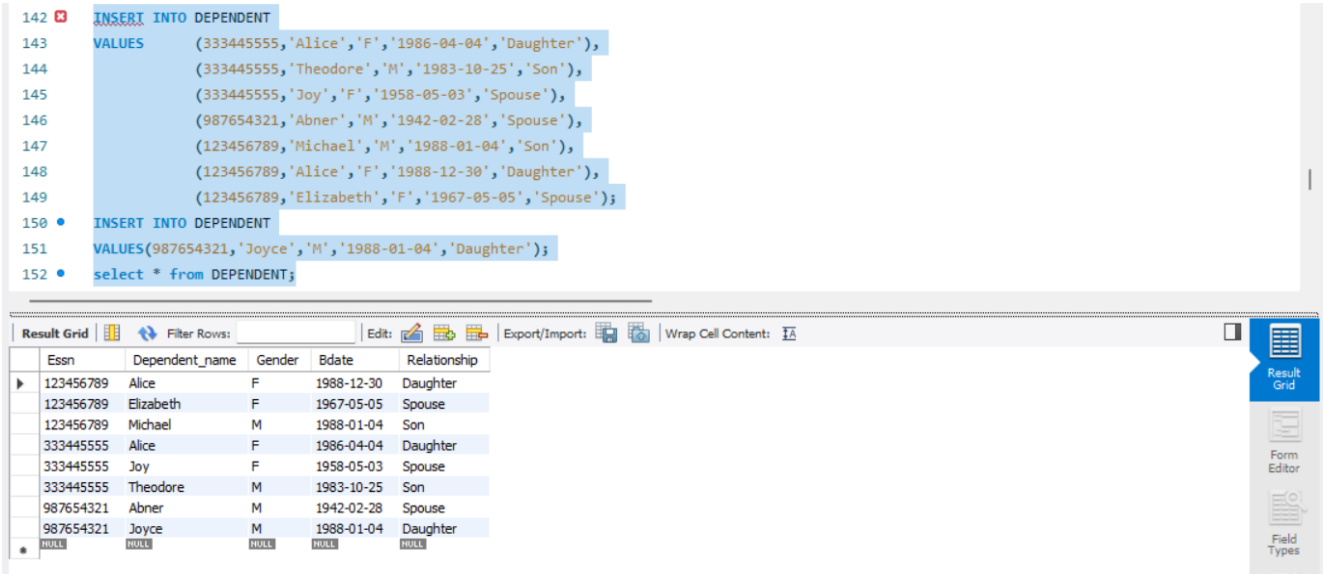
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
453453453	1	20.0
453453453	2	20.0

Result Grid | Form Editor

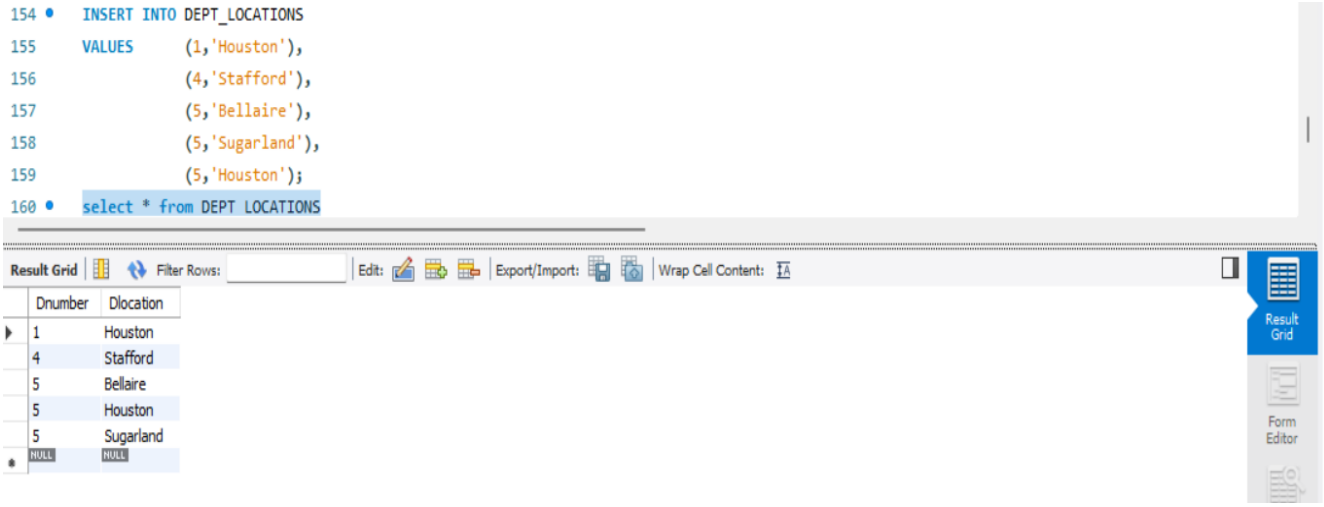
works_on 4 | Apply | Revert

```
INSERT INTO DEPENDENT
VALUES     (333445555,'Alice','F','1986-04-04','Daughter'),
           (333445555,'Theodore','M','1983-10-25','Son'),
           (333445555,'Joy','F','1958-05-03','Spouse'),
           (987654321,'Abner','M','1942-02-28','Spouse'),
           (123456789,'Michael','M','1988-01-04','Son'),
           (123456789,'Alice','F','1988-12-30','Daughter'),
           (123456789,'Elizabeth','F','1967-05-05','Spouse');
INSERT INTO DEPENDENT
VALUES(987654321,'Joyce','M','1988-01-04','Daughter');
select * from DEPENDENT;
```



Essn	Dependent_name	Gender	Bdate	Relationship
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse
123456789	Michael	M	1988-01-04	Son
333445555	Alice	F	1986-04-04	Daughter
333445555	Joy	F	1958-05-03	Spouse
333445555	Theodore	M	1983-10-25	Son
987654321	Abner	M	1942-02-28	Spouse
987654321	Joyce	M	1988-01-04	Daughter

```
INSERT INTO DEPT_LOCATIONS VALUES(1,'Houston'),
           (4,'Stafford'),
           (5,'Bellaire'),
           (5,'Sugarland'),
           (5,'Houston');
```



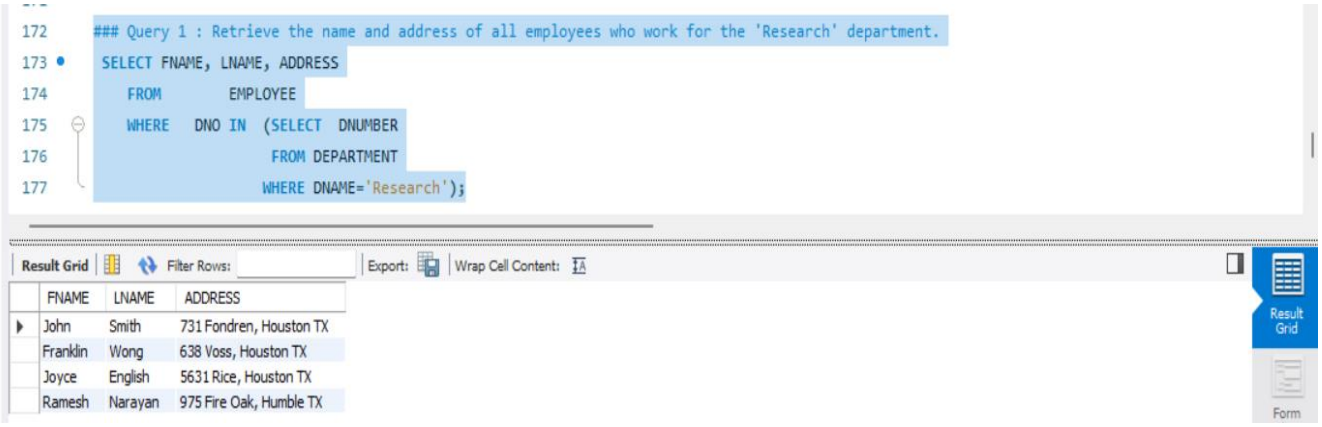
Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Houston
5	Sugarland

```
ALTER TABLE DEPARTMENT
 ADD CONSTRAINT Dep_emp FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn);
ALTER TABLE EMPLOYEE
 ADD CONSTRAINT Emp_emp FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn);
ALTER TABLE EMPLOYEE
 ADD CONSTRAINT Emp_dno FOREIGN KEY  (Dno) REFERENCES DEPARTMENT(Dnumber);
ALTER TABLE EMPLOYEE
 ADD CONSTRAINT Emp_super FOREIGN KEY  (Super_ssn) REFERENCES EMPLOYEE(Ssn);
```

22	15:55:43	ALTER TABLE DEPARTMENT ADD CONSTRAINT Dep_emp FOREIGN KEY (Mgr_ssn) REFERENCES EM...	3 row(s) affected Records: 3	Duplicates: 0	Warnings: 0	0.078 sec
23	15:55:48	ALTER TABLE EMPLOYEE ADD CONSTRAINT Emp_emp FOREIGN KEY (Super_ssn) REFERENCES EMP...	8 row(s) affected Records: 8	Duplicates: 0	Warnings: 0	0.093 sec
24	15:55:48	ALTER TABLE EMPLOYEE ADD CONSTRAINT Emp_dno FOREIGN KEY (Dno) REFERENCES DEPARTM...	8 row(s) affected Records: 8	Duplicates: 0	Warnings: 0	0.063 sec
25	15:55:48	ALTER TABLE EMPLOYEE ADD CONSTRAINT Emp_super FOREIGN KEY (Super_ssn) REFERENCES EM...	8 row(s) affected Records: 8	Duplicates: 0	Warnings: 0	0.078 sec

Query 1 : Retrieve the name and address of all employees who work for the 'Research' department.

```
SELECT FNAME, LNAME, ADDRESS
        FROM            EMPLOYEE
        WHERE DNO IN  (SELECT  DNUMBER
                                    FROM DEPARTMENT
                                   WHERE DNAME='Research');
```



```
172  ### Query 1 : Retrieve the name and address of all employees who work for the 'Research' department.
173  SELECT FNAME, LNAME, ADDRESS
174     FROM        EMPLOYEE
175     WHERE   DNO IN  (SELECT  DNUMBER
176                       FROM DEPARTMENT
177                      WHERE DNAME='Research');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

FNAME	LNAME	ADDRESS
John	Smith	731 Fondren, Houston TX
Franklin	Wong	638 Voss, Houston TX
Joyce	English	5631 Rice, Houston TX
Ramesh	Narayan	975 Fire Oak, Humble TX

Result Grid | Form

###Query 2: Retrieve the name of each employee who has a dependent with the same first name as the employee.

```
   SELECT E.FNAME
        FROM EMPLOYEE AS E
        WHERE E.SSN IN (SELECT ESSN
                                  FROM DEPENDENT
                                  WHERE ESSN=E.SSN AND
                                 Fname=Dependent_name);
```

```
###Query 2: Retrieve the name of each employee who has a dependent with the same first name as the employee.
    SELECT E.FNAME
    FROM EMPLOYEE AS E
    WHERE E.SSN IN (SELECT ESSN
                    FROM  DEPENDENT
                    WHERE ESSN=E.SSN AND
                  Fname=Dependent_name);
```

FNAME

###QUERY 3: write a SQL query to find those employees whose salary matches the lowest salary of any of the departments. Return first name, last name and department ID.

```
SELECT fname, salary, Dno FROM employee
WHERE salary IN
(SELECT MIN(salary) FROM employee
GROUP BY Dno );
```

fname	salary	Dno
Joyce	25000	5
James	55000	1
Ahmad	25000	4
Alicia	25000	4

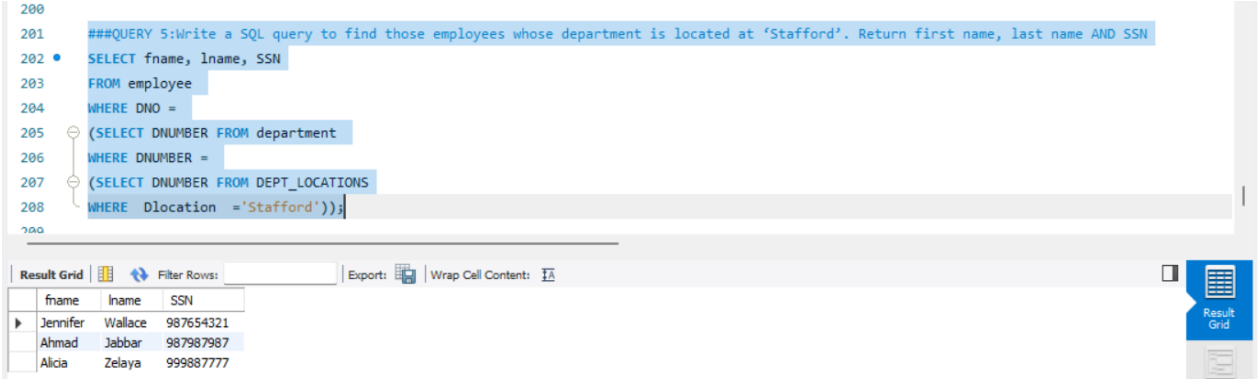
###QUERY 4: write a SQL query to find those employees who do not work in the departments of 1,2,3 (Begin and end values are included.). Return all the fields of the employees.

```
SELECT * FROM employee
WHERE Dno NOT IN
(SELECT Dnumber FROM department
WHERE
DNO BETWEEN 1 AND 3);
```

Fname	Minit	Lname	Ssn	Bdate	Address	gender	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1965-12-08	638 Voss, Houston TX	M	40000	888665555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston TX	F	25000	333445555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble TX	M	38000	333445555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire TX	F	43000	888665555	4
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston TX	M	25000	987654321	4
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring TX	F	25000	987654321	4

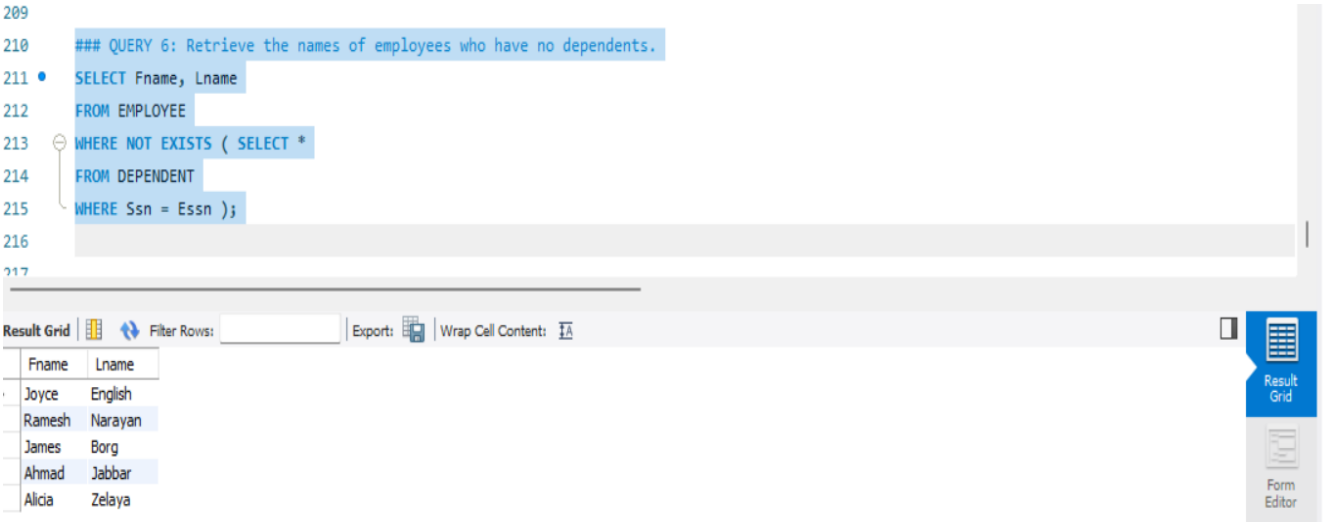
###QUERY 5:Write a SQL query to find those employees whose department is located at 'Stafford'. Return first name, last name AND SSN

```
SELECT fname, lname, SSN
FROM employee
WHERE DNO =
(SELECT DNUMBER FROM department
WHERE DNUMBER =
(SELECT DNUMBER FROM DEPT_LOCATIONS
WHERE  Dlocation  ='Stafford'));
```



QUERY 6: Retrieve the names of employees who have no dependents.

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE NOT EXISTS ( SELECT *
FROM DEPENDENT
WHERE Ssn = Essn );
```



LAB PROGRAM 4
AIRLINE DATABASE

Create an Airline database with 5 entities: Airline, Flight, Passenger, Ticket, and Airport using the following relational schema and write the following SQL Queries.

RELATIONAL SCHEMA

Airline(AirlineID, Name, Country)

Flight(FlightID, AirlineID, DepartureAirportID, ArrivalAirportID, DepartureTime, ArrivalTime)

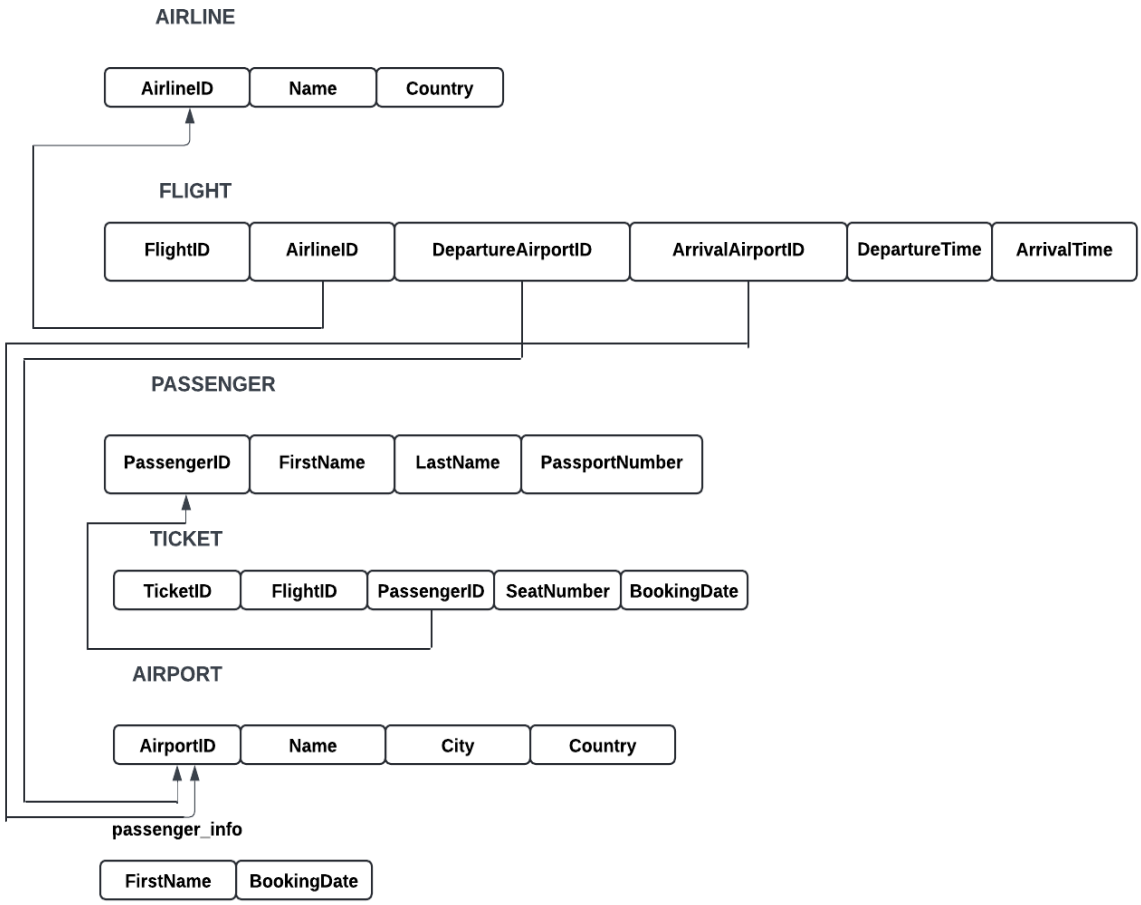
Passenger(PassengerID, FirstName, LastName, PassportNumber)

Ticket(TicketID, FlightID, PassengerID, SeatNumber, BookingDate)

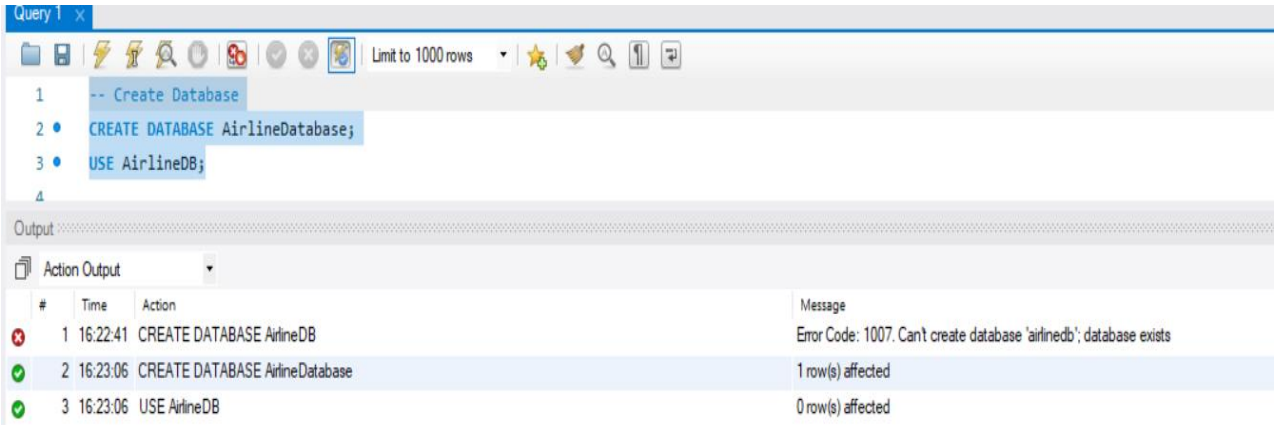
Airport(AirportID, Name, City, Country)

Passenger_info(FirstName, BookingDate)

SCHEMA DIAGRAM



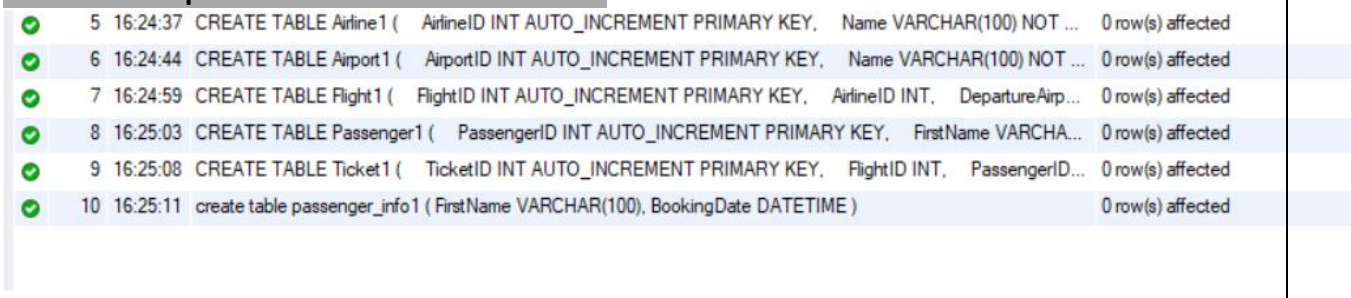
```
-- Create Database
CREATE DATABASE AirlineDB;
USE AirlineDB;
```




```
-- Create Airline Table
CREATE TABLE Airline (
   AirlineID INT AUTO_INCREMENT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   Country VARCHAR(100)
);
-- Create Airport Table
CREATE TABLE Airport (
   AirportID INT AUTO_INCREMENT PRIMARY KEY,
   Name VARCHAR(100) NOT NULL,
   City VARCHAR(100),
   Country VARCHAR(100)
);
-- Create Flight Table
CREATE TABLE Flight (
   FlightID INT AUTO_INCREMENT PRIMARY KEY,
   AirlineID INT,
   DepartureAirportID INT,
   ArrivalAirportID INT,
   DepartureTime DATETIME,
   ArrivalTime DATETIME,
   FOREIGN KEY (AirlineID) REFERENCES Airline(AirlineID),
   FOREIGN KEY (DepartureAirportID) REFERENCES Airport(AirportID),
   FOREIGN KEY (ArrivalAirportID) REFERENCES Airport(AirportID)
);
-- Create Passenger Table
CREATE TABLE Passenger (
   PassengerID INT AUTO_INCREMENT PRIMARY KEY,
   FirstName VARCHAR(100),
   LastName VARCHAR(100),
   PassportNumber VARCHAR(50) UNIQUE
);
```

```
-- Create Ticket Table
CREATE TABLE Ticket (
   TicketID INT AUTO_INCREMENT PRIMARY KEY,
   FlightID INT,
   PassengerID INT,
   SeatNumber VARCHAR(10),
   BookingDate DATETIME,
   FOREIGN KEY (SeatNumber FlightID) REFERENCES Flight(FlightID),
   FOREIGN KEY (PassengerID) REFERENCES Passenger(PassengerID)
);


create table passenger_info
(
FirstName VARCHAR(100),
BookingDate DATETIME
);
```

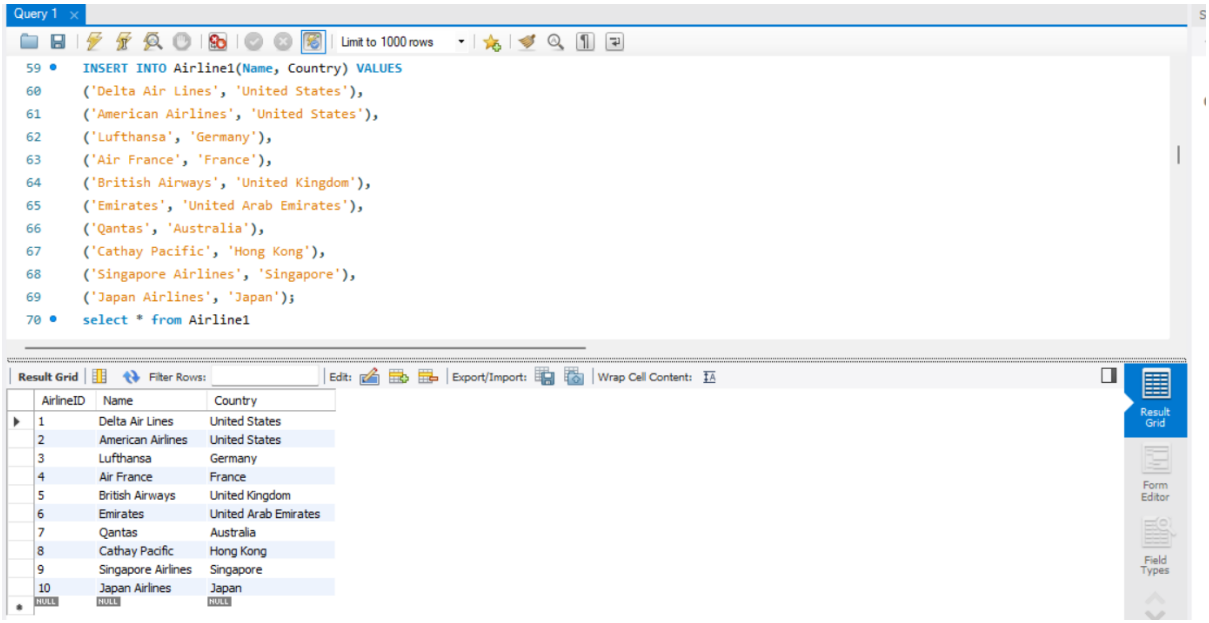
Collective outputs for create table commands



✓	5	16:24:37	CREATE TABLE Airline1 (AirlineID INT AUTO_INCREMENT PRIMARY KEY, Name VARCHAR(100) NOT ...	0 row(s) affected
✓	6	16:24:44	CREATE TABLE Airport1 (AirportID INT AUTO_INCREMENT PRIMARY KEY, Name VARCHAR(100) NOT ...	0 row(s) affected
✓	7	16:24:59	CREATE TABLE Flight1 (FlightID INT AUTO_INCREMENT PRIMARY KEY, AirlineID INT, DepartureAirp...	0 row(s) affected
✓	8	16:25:03	CREATE TABLE Passenger1 (PassengerID INT AUTO_INCREMENT PRIMARY KEY, FirstName VARCHA...	0 row(s) affected
✓	9	16:25:08	CREATE TABLE Ticket1 (TicketID INT AUTO_INCREMENT PRIMARY KEY, FlightID INT, PassengerID...	0 row(s) affected
✓	10	16:25:11	create table passenger_info1 (FirstName VARCHAR(100), BookingDate DATETIME)	0 row(s) affected

INSERTING VALUES

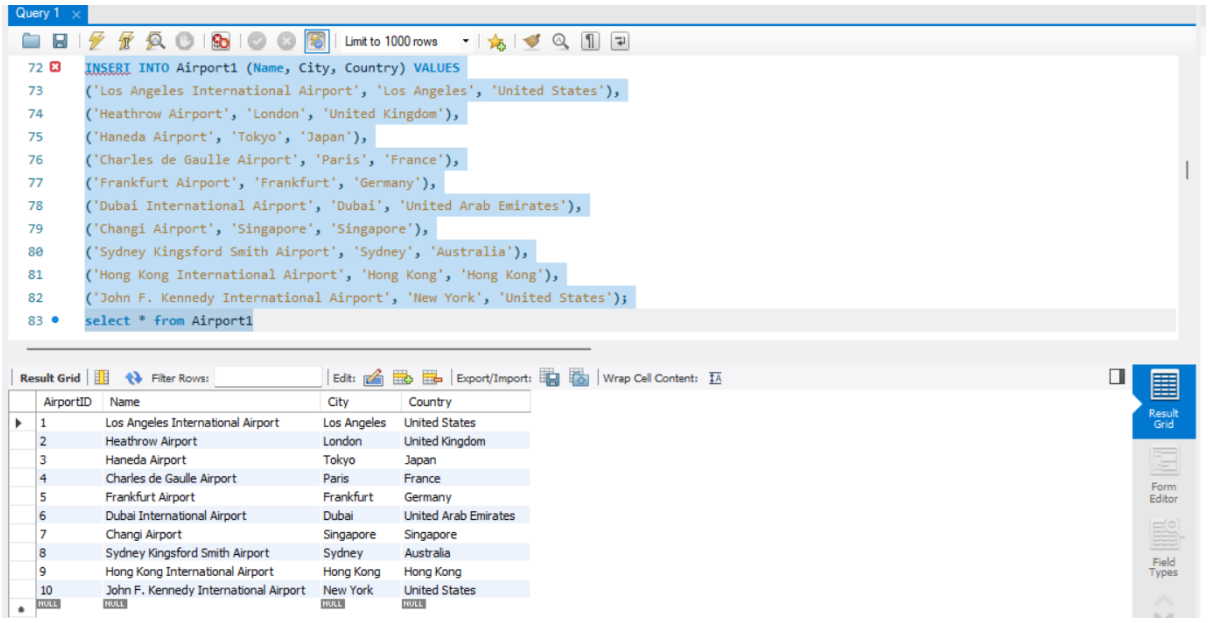
```
INSERT INTO Airline (Name, Country) VALUES
('Delta Air Lines', 'United States'),
('American Airlines', 'United States'),
('Lufthansa', 'Germany'),
('Air France', 'France'),
('British Airways', 'United Kingdom'),
('Emirates', 'United Arab Emirates'),
('Qantas', 'Australia'),
('Cathay Pacific', 'Hong Kong'),
('Singapore Airlines', 'Singapore'),
('Japan Airlines', 'Japan');
```



```
INSERT INTO Airline1(Name, Country) VALUES
('Delta Air Lines', 'United States'),
('American Airlines', 'United States'),
('Lufthansa', 'Germany'),
('Air France', 'France'),
('British Airways', 'United Kingdom'),
('Emirates', 'United Arab Emirates'),
('Qantas', 'Australia'),
('Cathay Pacific', 'Hong Kong'),
('Singapore Airlines', 'Singapore'),
('Japan Airlines', 'Japan');
select * from Airline1
```

AirlineID	Name	Country
1	Delta Air Lines	United States
2	American Airlines	United States
3	Lufthansa	Germany
4	Air France	France
5	British Airways	United Kingdom
6	Emirates	United Arab Emirates
7	Qantas	Australia
8	Cathay Pacific	Hong Kong
9	Singapore Airlines	Singapore
10	Japan Airlines	Japan

INSERT INTO Airport (Name, City, Country) VALUES
('Los Angeles International Airport', 'Los Angeles', 'United States'),
('Heathrow Airport', 'London', 'United Kingdom'),
('Haneda Airport', 'Tokyo', 'Japan'),
('Charles de Gaulle Airport', 'Paris', 'France'),
('Frankfurt Airport', 'Frankfurt', 'Germany'),
('Dubai International Airport', 'Dubai', 'United Arab Emirates'),
('Changi Airport', 'Singapore', 'Singapore'),
('Sydney Kingsford Smith Airport', 'Sydney', 'Australia'),
('Hong Kong International Airport', 'Hong Kong', 'Hong Kong'),
('John F. Kennedy International Airport', 'New York', 'United States');



```
INSERT INTO Airport1 (Name, City, Country) VALUES
('Los Angeles International Airport', 'Los Angeles', 'United States'),
('Heathrow Airport', 'London', 'United Kingdom'),
('Haneda Airport', 'Tokyo', 'Japan'),
('Charles de Gaulle Airport', 'Paris', 'France'),
('Frankfurt Airport', 'Frankfurt', 'Germany'),
('Dubai International Airport', 'Dubai', 'United Arab Emirates'),
('Changi Airport', 'Singapore', 'Singapore'),
('Sydney Kingsford Smith Airport', 'Sydney', 'Australia'),
('Hong Kong International Airport', 'Hong Kong', 'Hong Kong'),
('John F. Kennedy International Airport', 'New York', 'United States');
select * from Airport1
```

AirportID	Name	City	Country
1	Los Angeles International Airport	Los Angeles	United States
2	Heathrow Airport	London	United Kingdom
3	Haneda Airport	Tokyo	Japan
4	Charles de Gaulle Airport	Paris	France
5	Frankfurt Airport	Frankfurt	Germany
6	Dubai International Airport	Dubai	United Arab Emirates
7	Changi Airport	Singapore	Singapore
8	Sydney Kingsford Smith Airport	Sydney	Australia
9	Hong Kong International Airport	Hong Kong	Hong Kong
10	John F. Kennedy International Airport	New York	United States

INSERT INTO Flight (AirlineID, DepartureAirportID, ArrivalAirportID, DepartureTime, ArrivalTime) VALUES
(1, 1, 2, '2024-07-01 08:00:00', '2024-07-01 16:00:00'),
(2, 3, 4, '2024-07-02 09:00:00', '2024-07-02 17:00:00'),
(3, 5, 6, '2024-07-03 10:00:00', '2024-07-03 18:00:00'),
(4, 7, 8, '2024-07-04 11:00:00', '2024-07-04 19:00:00'),
(5, 9, 10, '2024-07-05 12:00:00', '2024-07-05 20:00:00'),
(6, 1, 3, '2024-07-06 13:00:00', '2024-07-06 21:00:00'),
(7, 2, 4, '2024-07-07 14:00:00', '2024-07-07 22:00:00'),
(8, 5, 7, '2024-07-08 15:00:00', '2024-07-08 23:00:00'),
(9, 6, 8, '2024-07-09 16:00:00', '2024-07-09 00:00:00'),
(10, 9, 1, '2024-07-10 17:00:00', '2024-07-10 01:00:00');

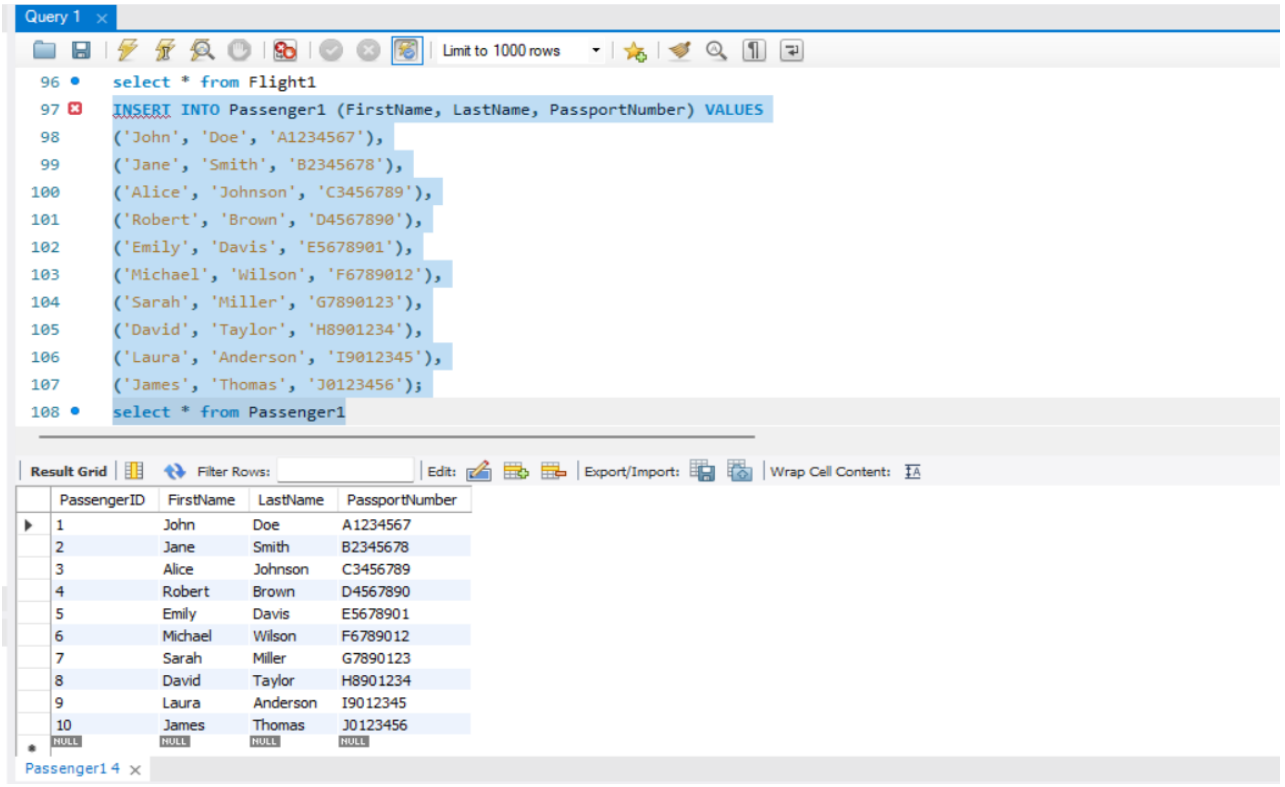
```
84
85  INSERT INTO Flight1 (AirlineID, DepartureAirportID, ArrivalAirportID, DepartureTime, ArrivalTime) VALUES
86  (1, 1, 2, '2024-07-01 08:00:00', '2024-07-01 16:00:00'),
87  (2, 3, 4, '2024-07-02 09:00:00', '2024-07-02 17:00:00'),
88  (3, 5, 6, '2024-07-03 10:00:00', '2024-07-03 18:00:00'),
89  (4, 7, 8, '2024-07-04 11:00:00', '2024-07-04 19:00:00'),
90  (5, 9, 10, '2024-07-05 12:00:00', '2024-07-05 20:00:00'),
91  (6, 1, 3, '2024-07-06 13:00:00', '2024-07-06 21:00:00'),
92  (7, 2, 4, '2024-07-07 14:00:00', '2024-07-07 22:00:00'),
93  (8, 5, 7, '2024-07-08 15:00:00', '2024-07-08 23:00:00'),
94  (9, 6, 8, '2024-07-09 16:00:00', '2024-07-09 00:00:00'),
95  (10, 9, 1, '2024-07-10 17:00:00', '2024-07-10 01:00:00');
96  select * from Flight1
```

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

FlightID	AirlineID	DepartureAirportID	ArrivalAirportID	DepartureTime	ArrivalTime
1	1	1	2	2024-07-01 08:00:00	2024-07-01 16:00:00
2	2	3	4	2024-07-02 09:00:00	2024-07-02 17:00:00
3	3	5	6	2024-07-03 10:00:00	2024-07-03 18:00:00
4	4	7	8	2024-07-04 11:00:00	2024-07-04 19:00:00
5	5	9	10	2024-07-05 12:00:00	2024-07-05 20:00:00
6	6	1	3	2024-07-06 13:00:00	2024-07-06 21:00:00
7	7	2	4	2024-07-07 14:00:00	2024-07-07 22:00:00
8	8	5	7	2024-07-08 15:00:00	2024-07-08 23:00:00
9	9	6	8	2024-07-09 16:00:00	2024-07-09 00:00:00
10	10	9	1	2024-07-10 17:00:00	2024-07-10 01:00:00
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid | Form Editor | Field Types

INSERT INTO Passenger (FirstName, LastName, PassportNumber) VALUES
('John', 'Doe', 'A1234567'),
('Jane', 'Smith', 'B2345678'),
('Alice', 'Johnson', 'C3456789'),
('Robert', 'Brown', 'D4567890'),
('Emily', 'Davis', 'E5678901'),
('Michael', 'Wilson', 'F6789012'),
('Sarah', 'Miller', 'G7890123'),
('David', 'Taylor', 'H8901234'),
('Laura', 'Anderson', 'I9012345'),
('James', 'Thomas', 'J0123456');



INSERT INTO Ticket (FlightID, PassengerID, SeatNumber, BookingDate) VALUES
(1, 1, '12A', '2024-06-01 10:00:00'),
(2, 2, '14B', '2024-06-02 11:00:00'),
(3, 3, '16C', '2024-06-03 12:00:00'),
(4, 4, '18D', '2024-06-04 13:00:00'),
(5, 5, '20E', '2024-06-05 14:00:00'),
(6, 6, '22F', '2024-06-06 15:00:00'),
(7, 7, '24G', '2024-06-07 16:00:00'),
(8, 8, '26H', '2024-06-08 17:00:00'),
(9, 9, '28I', '2024-06-09 18:00:00'),
(10, 10, '30J', '2024-06-10 19:00:00');

```
109  INSERT INTO Ticket (FlightID, PassengerID, SeatNumber, BookingDate) VALUES
110  (1, 1, '12A', '2024-06-01 10:00:00'),
111  (2, 2, '14B', '2024-06-02 11:00:00'),
112  (3, 3, '16C', '2024-06-03 12:00:00'),
113  (4, 4, '18D', '2024-06-04 13:00:00'),
114  (5, 5, '20E', '2024-06-05 14:00:00'),
115  (6, 6, '22F', '2024-06-06 15:00:00'),
116  (7, 7, '24G', '2024-06-07 16:00:00'),
117  (8, 8, '26H', '2024-06-08 17:00:00'),
118  (9, 9, '28I', '2024-06-09 18:00:00'),
119  (10, 10, '30J', '2024-06-10 19:00:00');
120  select * from ticket
```

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

TicketID	FlightID	PassengerID	SeatNumber	BookingDate
11	1	1	12A	2024-06-01 10:00:00
12	2	2	14B	2024-06-02 11:00:00
13	3	3	16C	2024-06-03 12:00:00
14	4	4	18D	2024-06-04 13:00:00
15	5	5	20E	2024-06-05 14:00:00
16	6	6	22F	2024-06-06 15:00:00
17	7	7	24G	2024-06-07 16:00:00
18	8	8	26H	2024-06-08 17:00:00
19	9	9	28I	2024-06-09 18:00:00
20	10	10	30J	2024-06-10 19:00:00
21	1	1	12A	2024-06-01 10:00:00

ticket 5 | Apply | Revert

INSERT INTO passenger_info (FirstName, BookingDate) VALUES
('John Doe', '2024-07-01 12:00:00'),
('Jane Smith', '2024-07-02 13:00:00'),
('Alice Johnson', '2024-07-03 14:00:00'),
('Bob Brown', '2024-07-04 15:00:00'),
('John Doe', '2024-07-01 12:00:00'), -- Duplicate
('Jane Smith', '2024-07-02 13:00:00'), -- Duplicate
('Alice Johnson', '2024-07-03 14:00:00'), -- Duplicate
('Bob Brown', '2024-07-04 15:00:00'), -- Duplicate
('Charlie Davis', '2024-07-05 16:00:00'),
('Eva Green', '2024-07-06 17:00:00');

Query 1 | Limit to 1000 rows

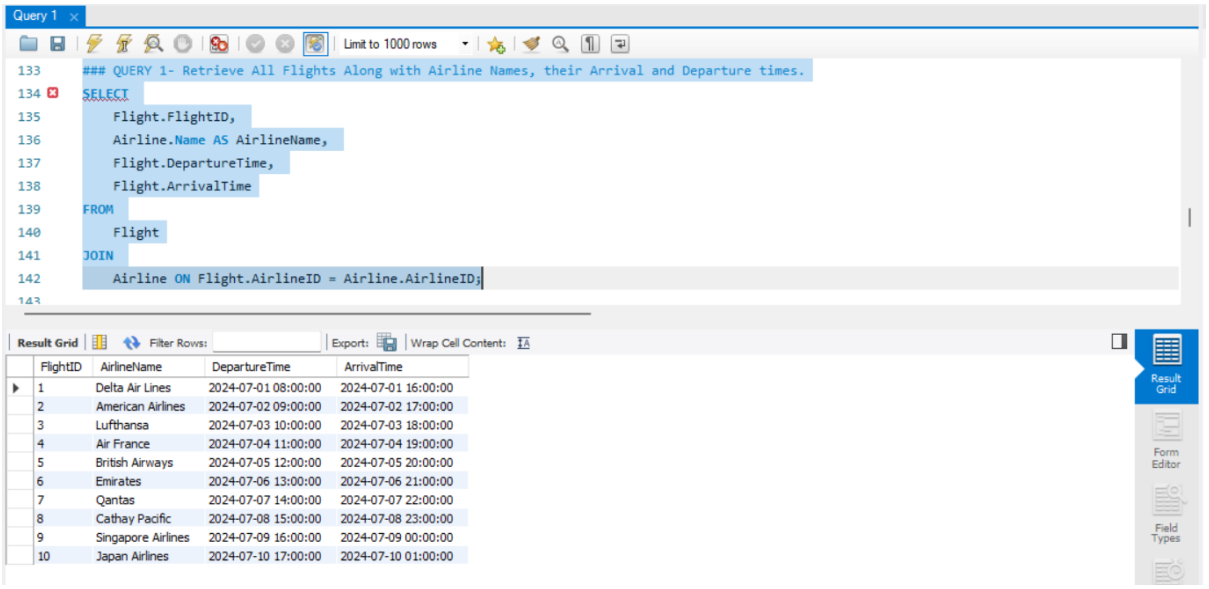
```
121  INSERT INTO passenger_info1 (FirstName, BookingDate) VALUES
122  ('John Doe', '2024-07-01 12:00:00'),
123  ('Jane Smith', '2024-07-02 13:00:00'),
124  ('Alice Johnson', '2024-07-03 14:00:00'),
125  ('Bob Brown', '2024-07-04 15:00:00'),
126  ('John Doe', '2024-07-01 12:00:00'),  -- Duplicate
127  ('Jane Smith', '2024-07-02 13:00:00'), -- Duplicate
128  ('Alice Johnson', '2024-07-03 14:00:00'), -- Duplicate
129  ('Bob Brown', '2024-07-04 15:00:00'), -- Duplicate
130  ('Charlie Davis', '2024-07-05 16:00:00'),
131  ('Eva Green', '2024-07-06 17:00:00');
132  select * from passenger_info1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

FirstName	BookingDate
John Doe	2024-07-01 12:00:00
Jane Smith	2024-07-02 13:00:00
Alice Johnson	2024-07-03 14:00:00
Bob Brown	2024-07-04 15:00:00
John Doe	2024-07-01 12:00:00
Jane Smith	2024-07-02 13:00:00
Alice Johnson	2024-07-03 14:00:00
Bob Brown	2024-07-04 15:00:00
Charlie Davis	2024-07-05 16:00:00
Eva Green	2024-07-06 17:00:00

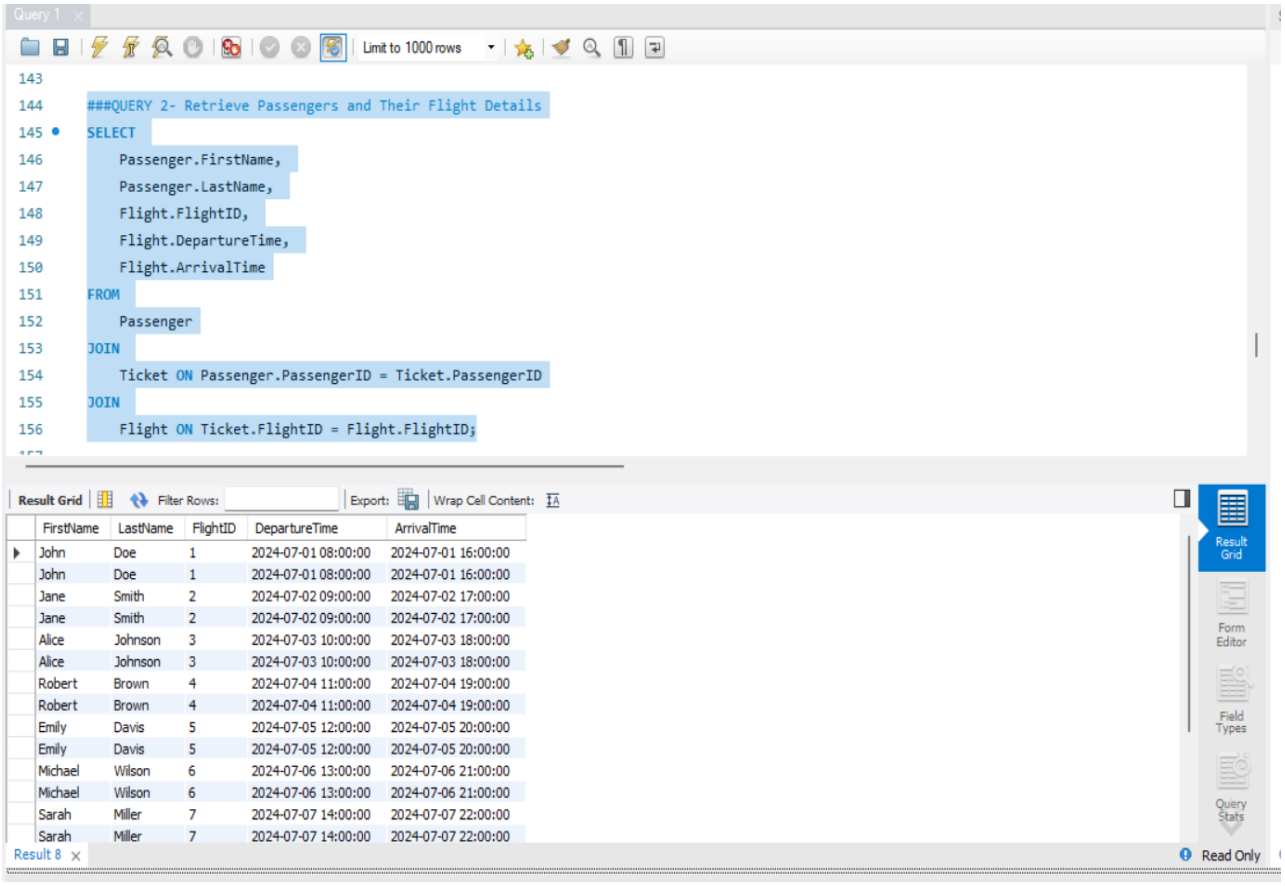
QUERY 1- Retrieve All Flights Along with Airline Names, their Arrival and Departure times.

```
SELECT
   Flight.FlightID,
   Airline.Name AS AirlineName,
   Flight.DepartureTime,
   Flight.ArrivalTime
FROM
   Flight
JOIN
   Airline ON Flight.AirlineID = Airline.AirlineID;
```



QUERY 2- Retrieve Passengers and Their Flight Details

```
SELECT
   Passenger.FirstName,
   Passenger.LastName,
   Flight.FlightID,
   Flight.DepartureTime,
   Flight.ArrivalTime
FROM
   Passenger
JOIN
   Ticket ON Passenger.PassengerID = Ticket.PassengerID
JOIN
   Flight ON Ticket.FlightID = Flight.FlightID;
```



```
143
144    ###QUERY 2- Retrieve Passengers and Their Flight Details
145 •  SELECT
146        Passenger.FirstName,
147        Passenger.LastName,
148        Flight.FlightID,
149        Flight.DepartureTime,
150        Flight.ArrivalTime
151    FROM
152        Passenger
153    JOIN
154        Ticket ON Passenger.PassengerID = Ticket.PassengerID
155    JOIN
156        Flight ON Ticket.FlightID = Flight.FlightID;
```

Result Grid

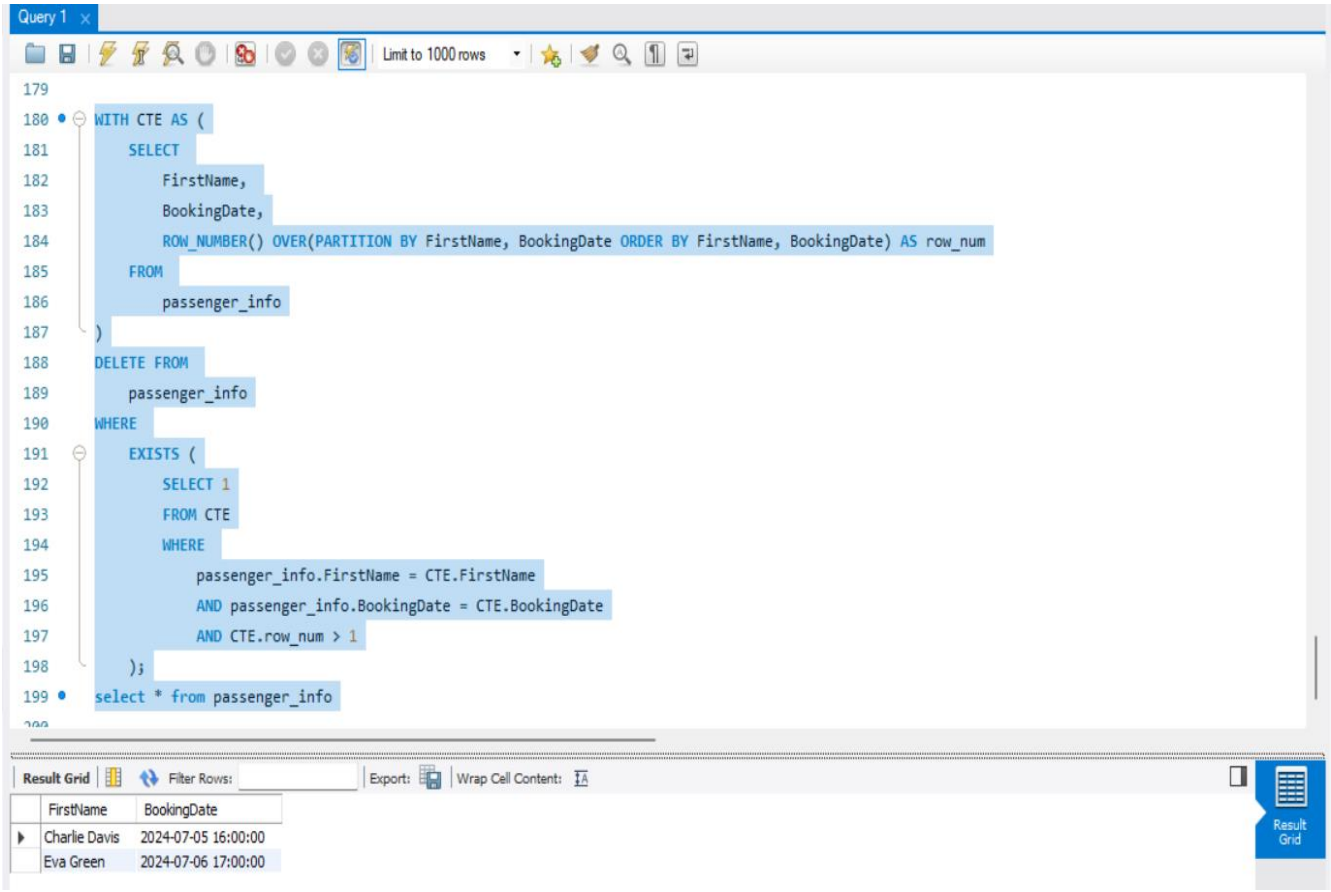
FirstName	LastName	FlightID	DepartureTime	ArrivalTime
John	Doe	1	2024-07-01 08:00:00	2024-07-01 16:00:00
John	Doe	1	2024-07-01 08:00:00	2024-07-01 16:00:00
Jane	Smith	2	2024-07-02 09:00:00	2024-07-02 17:00:00
Jane	Smith	2	2024-07-02 09:00:00	2024-07-02 17:00:00
Alice	Johnson	3	2024-07-03 10:00:00	2024-07-03 18:00:00
Alice	Johnson	3	2024-07-03 10:00:00	2024-07-03 18:00:00
Robert	Brown	4	2024-07-04 11:00:00	2024-07-04 19:00:00
Robert	Brown	4	2024-07-04 11:00:00	2024-07-04 19:00:00
Emily	Davis	5	2024-07-05 12:00:00	2024-07-05 20:00:00
Emily	Davis	5	2024-07-05 12:00:00	2024-07-05 20:00:00
Michael	Wilson	6	2024-07-06 13:00:00	2024-07-06 21:00:00
Michael	Wilson	6	2024-07-06 13:00:00	2024-07-06 21:00:00
Sarah	Miller	7	2024-07-07 14:00:00	2024-07-07 22:00:00
Sarah	Miller	7	2024-07-07 14:00:00	2024-07-07 22:00:00

###QUERY 3- Write an SQL query to identify the duplicate records in the passenger_info table.

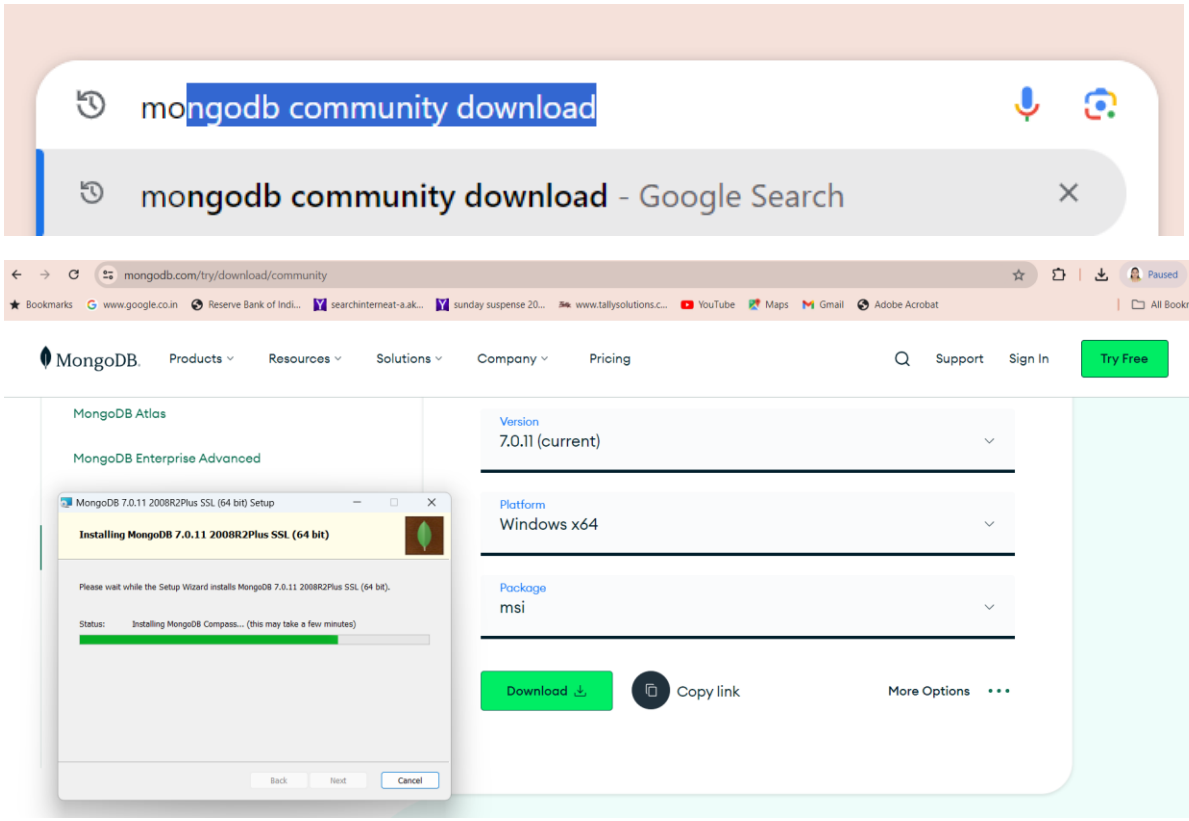
```
WITH CTE AS (
   SELECT
      FirstName,
      BookingDate,
      ROW_NUMBER() OVER(PARTITION BY FirstName, BookingDate ORDER BY
FirstName, BookingDate) AS row_num
   FROM
      passenger_info
)
SELECT
   FirstName,
   BookingDate,
   row_num
FROM
   CTE
WHERE
   row_num > 1;
```

###QUERY 4- Write an SQL query to delete the duplicate records in the passenger_info table.

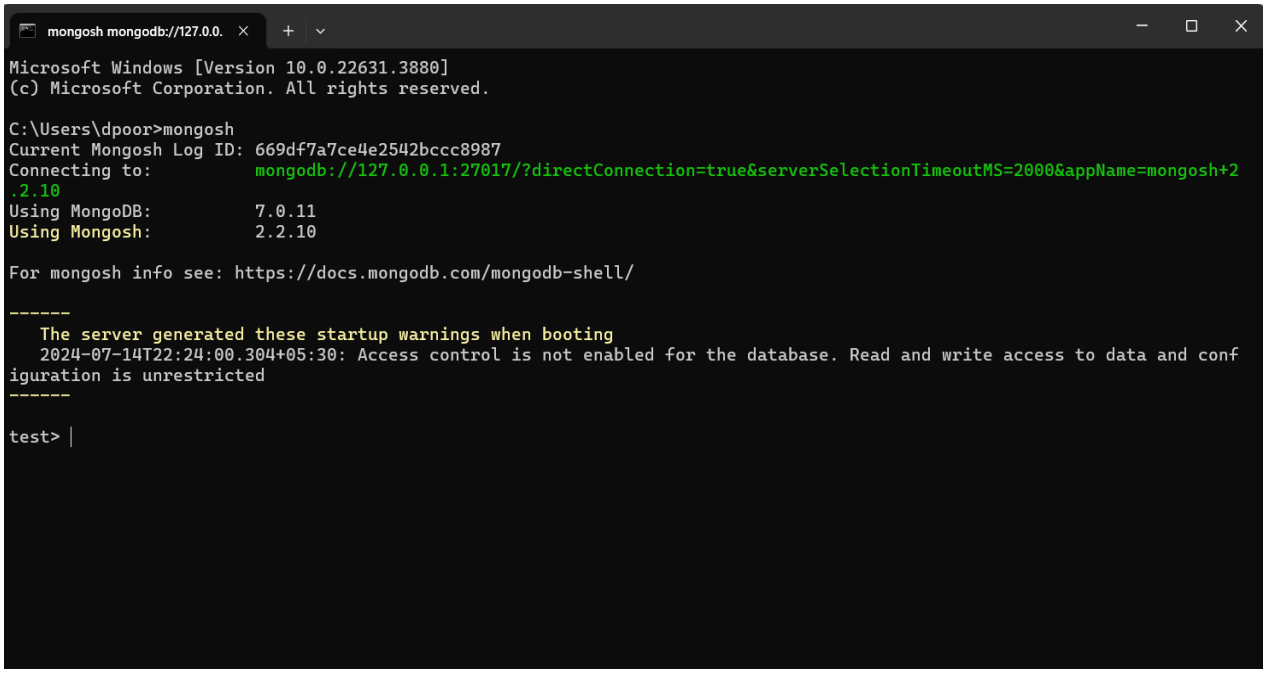
```
SET SQL_SAFE_UPDATES = 0;

WITH CTE AS (
   SELECT
      FirstName,
      BookingDate,
      ROW_NUMBER() OVER(PARTITION BY FirstName, BookingDate ORDER BY FirstName,
BookingDate) AS row_num
   FROM
      passenger_info
)
DELETE FROM
   passenger_info
WHERE
   EXISTS (
      SELECT 1
      FROM CTE
      WHERE
         passenger_info.FirstName = CTE.FirstName
         AND passenger_info.BookingDate = CTE.BookingDate
         AND CTE.row_num > 1
   );
select * from passenger_info
```


Query 1
Limit to 1000 rows
179
180 WITH CTE AS (
181 SELECT
182 FirstName,
183 BookingDate,
184 ROW_NUMBER() OVER(PARTITION BY FirstName, BookingDate ORDER BY FirstName, BookingDate) AS row_num
185 FROM
186 passenger_info
187)
188 DELETE FROM
189 passenger_info
190 WHERE
191 EXISTS (
192 SELECT 1
193 FROM CTE
194 WHERE
195 passenger_info.FirstName = CTE.FirstName
196 AND passenger_info.BookingDate = CTE.BookingDate
197 AND CTE.row_num > 1
198);
199 select * from passenger_info
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
FirstName
BookingDate
Charlie Davis
2024-07-05 16:00:00
Eva Green
2024-07-06 17:00:00
Result Grid

LAB PROGRAM 5
MONGODB Installation Screenshots



After Installation, Go to Command Prompt and type “mongosh” command



Command to create Database in MongoDB

>> use mydatabase

Output

```
test> use mydatabase
switched to db mydatabase
mydatabase>
```

QUERY1: Command to create new collections, update the documents

>> db.createCollection('csd')

Output

```
mydatabase> db.createCollection('CSD')
{ ok: 1 }
mydatabase> db.CSD.insertOne({name:"Dr. Suma V"})
{
  acknowledged: true,
  insertedId: ObjectId('6683a84a65cd3ef7bacc898a')
}
```

Command for updating the documents

```
mydatabase> db.CSD.updateOne({name:"Dr. suma V"},{$set:{age:29}})
```

OUTPUT

```
mydatabase> db.CSD.updateOne({name:"Dr. suma V"},{$set:{age:29}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
```

QUERY 2: Create a new collection named "empDetails" and insert 1 document for an employee first name, Last name, DOB, Email and PhoneNo

```
mydatabase> db.createCollection("empDetails")
```

```
mydatabase> db.empDetails.insertOne(
...     {
...             First_Name: "Radhika",
...             Last_Name: "Sharma",
...             Date_Of_Birth: "1995-09-26",
...             e_mail: "radhika_sharma.123@gmail.com",
...             phone: "9848022338"
...     })
{
```

OUTPUT

```
{
  acknowledged: true,
  insertedId: ObjectId('6683ae1465cd3ef7bacc898b')
}
```

QUERY 3: For the same collection "empDetails" add 2 to 3 documents capturing the different employees information for the same attributes.

```
mydatabase> db.empDetails.insertMany(
...     [
...             {
...                     First_Name: "Radhika",
...                     Last_Name: "Sharma",
...                     Date_Of_Birth: "1995-09-26",
...                     e_mail: "radhika_sharma.123@gmail.com",
...                     phone: "9000012345"
...             },
...             {
...                     First_Name: "Rachel",
...                     Last_Name: "Christopher",
...                     Date_Of_Birth: "1990-02-16",
...                     e_mail: "Rachel_Christopher.123@gmail.com",
...                     phone: "9000054321"
...             },
...             {
...                     First_Name: "Fathima",
...                     Last_Name: "Sheik",
...                     Date_Of_Birth: "1990-02-16",
...                     e_mail: "Fathima_Sheik.123@gmail.com",
...                     phone: "9000054321"
...             }
...     ]
... )
```

QUERY 4. Write a MongoDB command to display the contents of "empDetails" collection

Command:

```
mydatabase> db.empDetails.find();
```

Output:

```
[
  {
    _id: ObjectId('6683ae1465cd3ef7bacc898b'),
    First_Name: 'Radhika',
    Last_Name: 'Sharma',
    Date_Of_Birth: '1995-09-26',
    e_mail: 'radhika_sharma.123@gmail.com',
    phone: '9848022338'
  },
  {
    _id: ObjectId('6683aeb765cd3ef7bacc898c'),
    First_Name: 'Radhika',
    Last_Name: 'Sharma',
    Date_Of_Birth: '1995-09-26',
    e_mail: 'radhika_sharma.123@gmail.com',
    phone: '9000012345'
  },
  {
    _id: ObjectId('6683aeb765cd3ef7bacc898d'),
    First_Name: 'Rachel',
    Last_Name: 'Christopher',
    Date_Of_Birth: '1990-02-16',
    e_mail: 'Rachel_Christopher.123@gmail.com',
    phone: '9000054321'
  },
  {
    _id: ObjectId('6683aeb765cd3ef7bacc898e'),
    First_Name: 'Fathima',
    Last_Name: 'Sheik',
    Date_Of_Birth: '1990-02-16',
    e_mail: 'Fathima_Sheik.123@gmail.com',
    phone: '9000054321'
  }
]
mydatabase>
```

QUERY 5: Write a mongoDB command to update the phone number of an employee using objectID and display the collection to check if the phone no. is updated or not.

Command:

```
mydatabase> db.empDetails.update({_id:ObjectId("6683aeb765cd3ef7bacc898d")},{$set:{phone:"8889990008"}});
```

OUTPUT:

```
mydatabase> db.empDetails.update({_id:ObjectId("6683aeb765cd3ef7bacc898d")},{$set:{phone:"8889990008"}});
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
mydatabase> db.empDetails.find()
[
  {
    _id: ObjectId('6683ae1465cd3ef7bacc898b'),
    First_Name: 'Radhika',
    Last_Name: 'Sharma',
    Date_Of_Birth: '1995-09-26',
    e_mail: 'radhika_sharma.123@gmail.com',
    phone: '9848022338'
  },
  {
    _id: ObjectId('6683aeb765cd3ef7bacc898c'),
    First_Name: 'Radhika',
    Last_Name: 'Sharma',
    Date_Of_Birth: '1995-09-26',
    e_mail: 'radhika_sharma.123@gmail.com',
    phone: '9000012345'
  },
  {
    _id: ObjectId('6683aeb765cd3ef7bacc898d'),
    First_Name: 'Rachel',
    Last_Name: 'Christopher',
    Date_Of_Birth: '1990-02-16',
    e_mail: 'Rachel_Christopher.123@gmail.com',
    phone: '8889990008'
  },
  {
    _id: ObjectId('6683aeb765cd3ef7bacc898e'),
    First_Name: 'Fathima',
    Last_Name: 'Sheik',
    Date_Of_Birth: '1990-02-16',
    e_mail: 'Fathima_Sheik.123@gmail.com',
    phone: '9000054321'
  }
]
mydatabase>
```

ANACONDA PART

Installation link for Anaconda

https://www.anaconda.com/download

Inside Jupiter Notebook→ Commands to connect anaconda with MongoDB

```
[1]: pip install mymongo
```

```
[7]: from pymongo import MongoClient
[11]: client=MongoClient("mongodb://localhost:27017/")
[13]: db=client['day_six']
[17]: collection=db["products"]
```

Command to insert many documents

```
[19]: collection.insert_many(documents)

documents = [
{ "name": "Laptop", "price": 800, "category": "electronics" },
{ "name": "Keyboard", "price": 20, "category": "electronics" },
{ "name": "Chair", "price": 50, "category": "furniture" },
{ "name": "Desk", "price": 100, "category": "furniture" },
{ "name": "Mouse", "price": 25, "category": "electronics" },
{ "name": "Monitor", "price": 150, "category": "electronics" }
]
```

QUERY 1: FIND ALL THE PRODUCTS WITH CATEGORY “ELECTRONICS”

```
[26]: #find products with category-NOT IN FUNCTION------------------------------
qi={"category":{"$nin":["furniture"]}}
result=collection.find(qi)
for i in result:
    print(i)
```

QUERY 2: FIND ALL THE PRODUCTS WITH CATEGORY “FURNITURE”

```
[21]: #find products with category- IN FUNCTION--------------------------
qi={"category":{"$in":["furniture"]}}
result=collection.find(qi)
for i in result:
    print(i)
```

QUERY 3: FIND ALL THE PRODUCTS WITH PRODUCT PRICE IS NOT EQUAL TO 800

```
: ### find the products whose price is not equal to 800rs---------------------------------
qno={"price":{"$not":{"$eq":800}}}
result=collection.find(qno)
for i in result:
    print(i)
```

QUERY 4: FIND ALL THE PRODUCTS WITH PRODUCT PRICE IS GREATER THAN 100

```
### find the products whose price is greater than or equal to 100 rs--------------------------
gte={"price":{"$gte":100}}
result=collection.find(gte)
for i in result:
    print(i)
```