NIKHIL M ABRAHAM

USN: 1BM19CS101

DATABASE MANAGEMENT SYSTEM
19CS4PCDBM

LAB REPORT

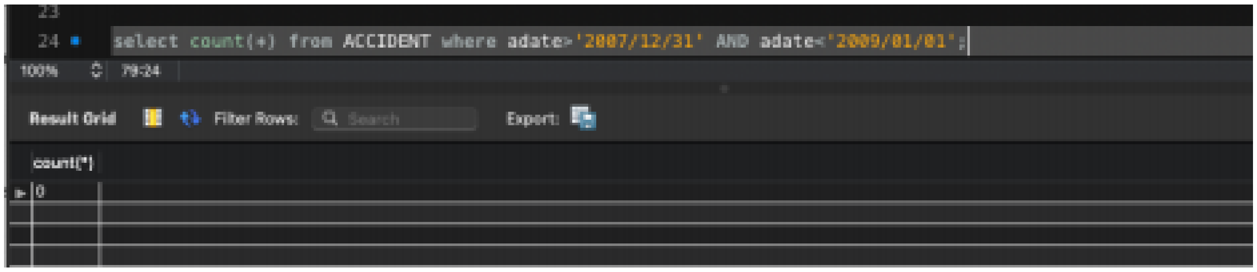
Lab-1 : Insurance Database

```
show databases;
create database Insurance;
use Insurance;
create table PERSON(driver_id varchar(30) primary key, name varchar(30), address varchar(30));
create table CAR(Regno varchar(30) primary key, model varchar(30), year int);
create table ACCIDENT(report_number int primary key, adate date, location varchar(30)); create
table OWNS(driver_id varchar(30), Regno varchar(30), primary key(driver_id,Regno), foreign
key(driver_id) references PERSON(driver_id), foreign key(Regno) references CAR(Regno)); create
table PARTICIPATED(driver_id varchar(30), Regno varchar(30), report_number int,
damage_amount int, primary key(driver_id, Regno), foreign key(driver_id, Regno) references
OWNS(driver_id, Regno));
show tables;
insert into PERSON values('05C','Bob','Jamaica');
insert into CAR values('1E','Hundai','2015');
insert into ACCIDENT values('5','1999/09/02','Hubli');
insert into OWNS values('05C','1E');
insert into PARTICIPATED values('01A','1A',1,96000);
select * from CAR;

-- a. Update the damage amount for the car with a specific Regno in the accident with report number 12 to
-- 25000.

update PARTICIPATED set damage_amount=25000 where Regno='01A' AND report_number=1;
select * from PARTICIPATED;


-- Add a new accident to the database.

insert into ACCIDENT values('6','1967/09/02','Kerala');
select * from ACCIDENT;
-- Find the total number of people who owned cars that involved in accidents in 2008.
```

select count from ACCIDENT where adate>'2007/12/31' AND adate<'2009/01/01'; -- Find

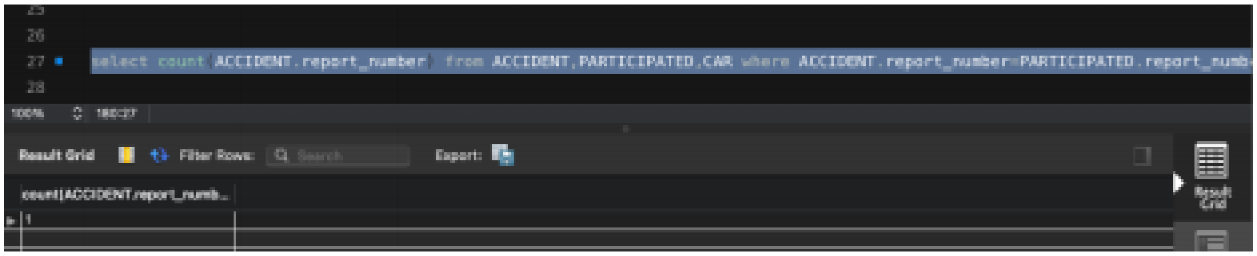
the number of accidents in which cars belonging to a specific model were involved.

select count(ACCIDENT.report_number) from ACCIDENT,PARTICIPATED,CAR where ACCIDENT.report_number=PARTICIPATED.report_number AND PARTICIPATED.Regno=CAR.Regno AND CAR.model='Chevy';



```
select count(*) from ACCIDENT where adate>'2007/12/31' AND adate<'2009/01/01';
```

```
count(*)
0
```



```
select count(ACCIDENT.report_number) from ACCIDENT,PARTICIPATED,CAR where ACCIDENT.report_number=PARTICIPATED.report_numb
```

```
count(ACCIDENT.report_numb...
1
```

Lab-2 : Book Database

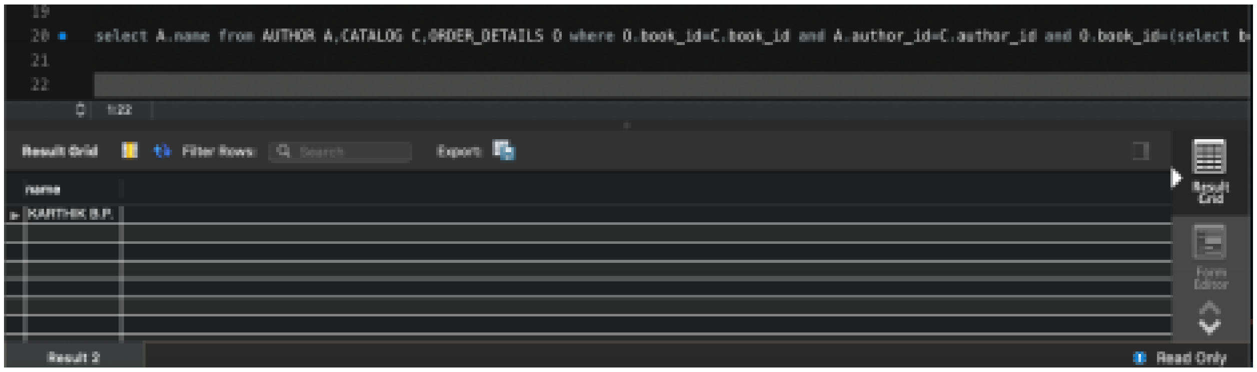
create database Book;
use Book;
create table AUTHOR(author_id int primary key, name varchar(30), city varchar(20), country varchar(20));
create table PUBLISHER(publisher_id int primary key, name varchar(30), city varchar(20), country varchar(20));
create table CATALOG(book_id int, title varchar(30), author_id int, publisher_id int, category_id int, year int, price int, primary key(book_id), foreign key(author_id) references AUTHOR(author_id), foreign key(publisher_id) references PUBLISHER(publisher_id), foreign key(category_id) references CATEGORY(category_id));
create table CATEGORY(category_id int, description varchar(50), primary key(category_id)); create table ORDER_DETAILS(order_no int primary key, book_id int, quantity int, foreign key(book_id) references CATALOG(book_id));

insert into AUTHOR values(1005,'WILLIAMS STALLINGS','LAS VEGAS','USA');
insert into PUBLISHER values(5,'MGH','NEW YORK','USA');
insert into CATEGORY values(1005,'OPERATING SYSTEMS');
insert into CATALOG values(17,'COBOL Handbook',1005,4,1001,2000,658);
insert into ORDER_DETAILS values(2,17,10);

```
select A.name,C.title,C.price from AUTHOR A,CATALOG C where C.author_id=A.author_id and
C.year>=2000 and A.name=(select A.name from AUTHOR  A,CATALOG C where
A.author_id=C.author_id group by C.author_id having count(*)>=2);
```

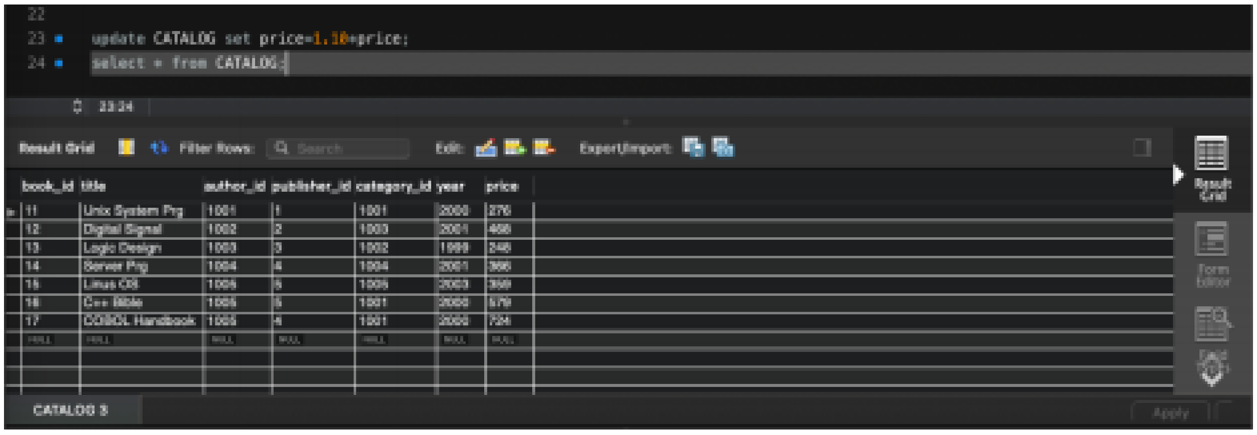
```
select A.name from AUTHOR A,CATALOG C,ORDER_DETAILS  O where O.book_id=C.book_id and
A.author_id=C.author_id and O.book_id=(select book_id from ORDER_DETAILS where quantity=(select
max(quantity) from ORDER_DETAILS));
```

```
update CATALOG set price=1.10*price;
select * from CATALOG;
```



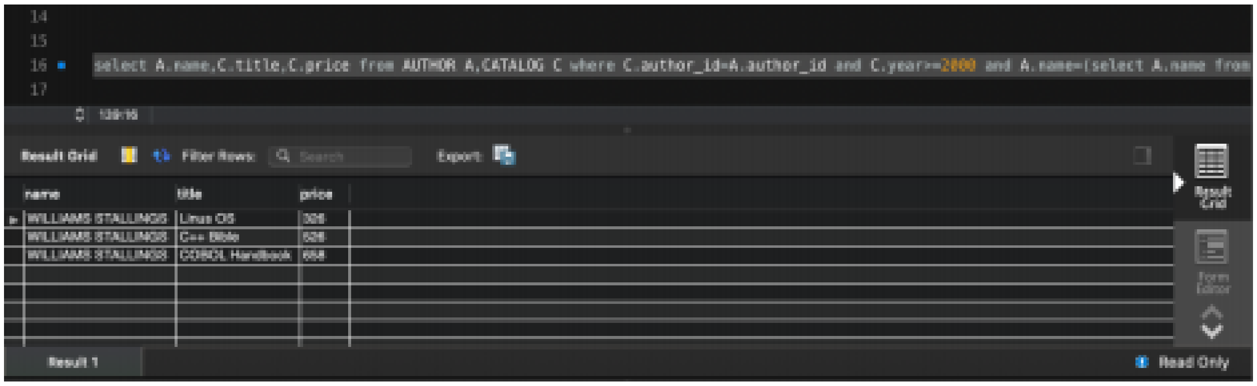
```
select A.name from AUTHOR A,CATALOG C,ORDER_DETAILS O where O.book_id=C.book_id and A.author_id=C.author_id and O.book_id=(select b
```

name
KARTHIK B.P.



```
update CATALOG set price=1.10*price;
select * from CATALOG;
```

book_id	title	author_id	publisher_id	category_id	year	price
11	Unix System Prg	1001	1	1001	2000	276
12	Digital Signal	1002	2	1003	2001	468
13	Logic Design	1003	3	1002	1999	248
14	Server Prg	1004	4	1004	2001	396
15	Linux OS	1005	5	1005	2003	369
16	C++ Bible	1005	5	1001	2000	579
17	COBOL Handbook	1005	4	1001	2000	724
NULL	NULL	NULL	NULL	NULL	NULL	NULL



```
select A.name,C.title,C.price from AUTHOR A,CATALOG C where C.author_id=A.author_id and C.year>=2000 and A.name=(select A.name from
```

```
name               title            price
WILLIAMS STALLINGS Linux OS         326
WILLIAMS STALLINGS C++ Bible        526
WILLIAMS STALLINGS COBOL Handbook   658
```

Lab-3 : Orders Database

create database order_processing1;
use order_processing1;
create table customer(cust int primary key,cname varchar(20),city varchar(20));
create table order_(order_no int primary key,odate date,cust int ,ord_amt int,
foreign key(cust) references customer(cust) on delete cascade);
create table item(item_no int primary key,unit_price int);
create table order_item(order_no int,item_no int ,qty int,
foreign key(order_no) references order_(order_no)on delete cascade,
foreign key(item_no) references item(item_no)on delete cascade);
create table warehouse(warehouse_no int primary key,city varchar(20));
create table shipment(order_no int,warehouse_no int ,ship_date date, foreign
key(order_no) references order_(order_no) on delete cascade, foreign
key(warehouse_no) references warehouse(warehouse_no) on delete cascade); show
tables;
drop table order_item;
insert into customer values(771,"pushpa k","bangalore");
insert into customer values(772,"suman","mumbai");
insert into customer values(773,"sourav","calicut");
insert into customer values(774,"laila","hyderabad");
insert into customer values(775,"faizal","bangalore");
select * from customer;

insert into order_ values(111,'2002-01-22',771,18000);
insert into order_ values(112,'2002-07-30',774,6000);
insert into order_ values(113,'2003-04-03',775,9000);
insert into order_ values(114,'2003-11-03',775,29000);
insert into order_ values(115,'2003-12-10',773,29000);
insert into order_ values(116,'2004-08-19',772,56000);
insert into order_ values(117,'2004-09-10',771,20000);
insert into order_ values(118,'2004-11-20',775,29000);
insert into order_ values(119,'2005-02-13',774,29000);
insert into order_ values(120,'2005-10-13',775,29000);

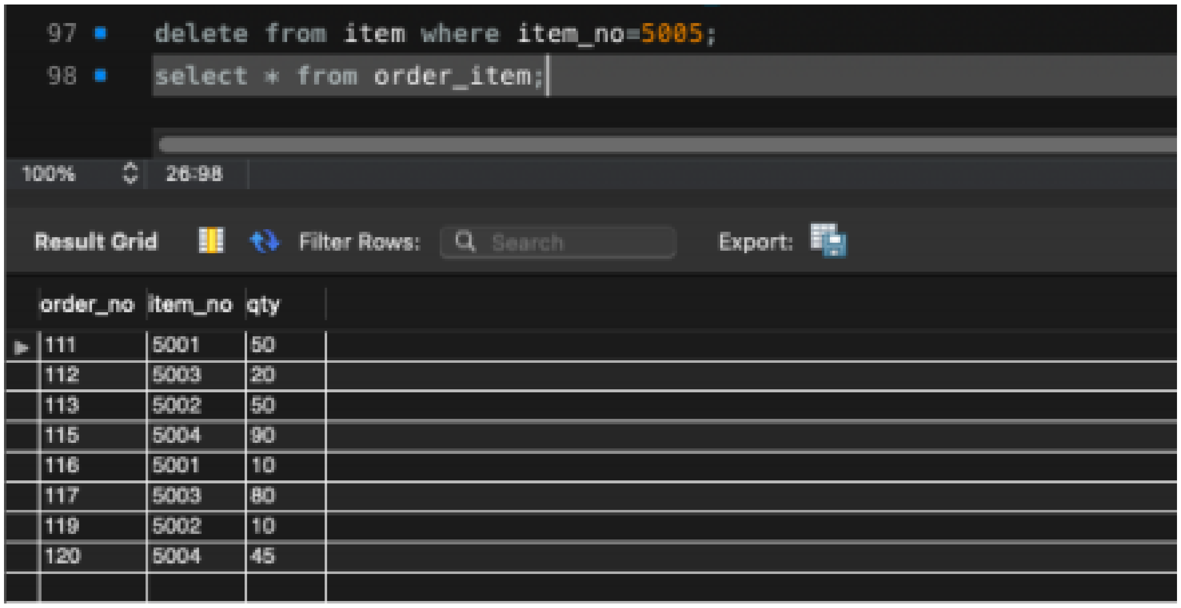
```
select * from order_;

insert into item values(5001,503);
insert into item values(5002,750);
insert into item values(5003,150);
insert into item values(5004,600);
insert into item values(5005,890);
select * from item;

insert into order_item values(111,5001,50);
insert into order_item values(112,5003,20);
insert into order_item values(113,5002,50);
insert into order_item values(114,5005,60);
insert into order_item values(115,5004,90);
insert into order_item values(116,5001,10);
insert into order_item values(117,5003,80);
insert into order_item values(118,5005,50);
insert into order_item values(119,5002,10);
insert into order_item values(120,5004,45);
select * from order_item;

insert into warehouse values(1,"delhi");
insert into warehouse values(2,"bombay");
insert into warehouse values(3,"chennai");
insert into warehouse values(4,"bangalore");
insert into warehouse values(5,"bangalore");
insert into warehouse values(6,"delhi");
insert into warehouse values(7,"bombay");
insert into warehouse values(8,"chennai");
insert into warehouse values(9,"delhi");
insert into warehouse values(10,"bangalore");
select * from warehouse;

insert into shipment values(111,1,'2002-02-10');
insert into shipment values(112,5,'2002-09-10');
insert into shipment values(113,8,'2003-02-10');
insert into shipment values(114,3,'2003-12-10');
insert into shipment values(115,9,'2004-01-19');
insert into shipment values(116,1,'2004-09-20');
insert into shipment values(117,5,'2004-09-10');
insert into shipment values(118,7,'2004-11-30');
insert into shipment values(119,7,'2005-04-30');
insert into shipment values(120,6,'2005-12-21');
select * from shipment;


-- iii) Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the middle column is the total
```

-- numbers of orders by the customer and the last column is the average order amount for that -- customer.
select c.cname,count(o.order_no) as total_orders,avg(o.ord_amt) as average_amount from customer c,order_ o
where c.cust=o.cust group by o.cust;

-- iv) List the order# for orders that were shipped from all
-- warehouses that the company has in a specific city.
select s.order_no from shipment s,warehouse w
where s.warehouse_no=w.warehouse_no and w.city="delhi";

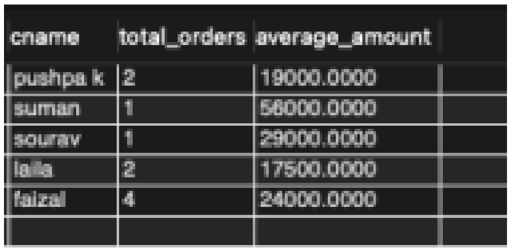
-- select s.order_no from shipment s where s.warehouse_no in(select w.warehouse_no from warehouse w where w.city="delhi");

-- v) Demonstrate how you delete item# 10 from the ITEM table and
-- make that field null in theORDER_ITEM table.
delete from item where item_no=5005;
select * from order_item;

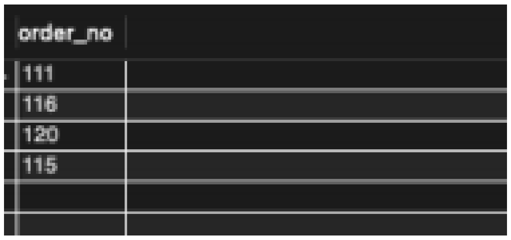


```
delete from item where item_no=5005;
select * from order_item;
```

```
order_no  item_no  qty
111       5001     50
112       5003     20
113       5002     50
115       5004     90
116       5001     10
117       5003     80
119       5002     10
120       5004     45
```



```
cname     total_orders  average_amount
pushpa k  2             19000.0000
suman     1             56000.0000
sourav    1             29000.0000
laila     2             17500.0000
faizal    4             24000.0000
```



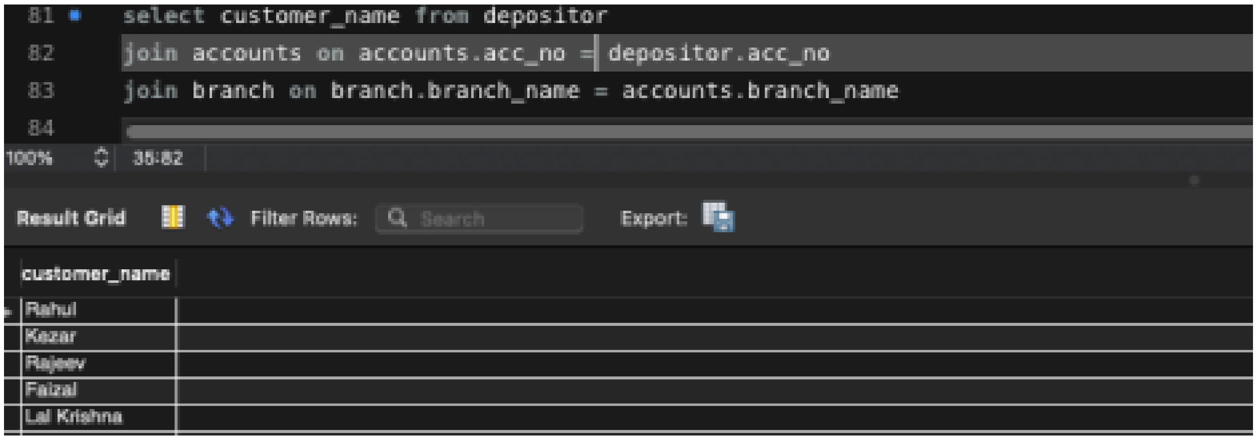
```
order_no
111
116
120
115
```

Lab-4 : Bank Database

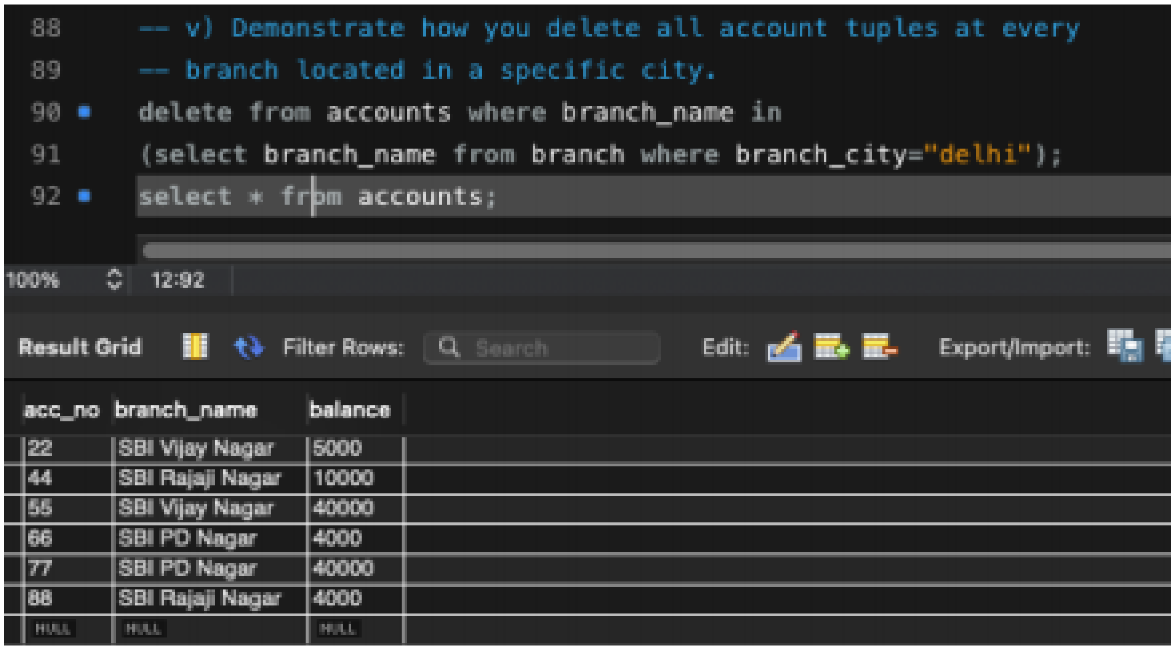
```
create database banking_enterprise;

use banking_enterprise;

create table branch(branch_name varchar(20) primary key,branch_city varchar(20),assets real);

create table accounts(acc_no int primary key,branch_name varchar(20),balance real, foreign
key(branch_name)
references branch(branch_name) on delete cascade);

create table customer(customer_name varchar(20) primary key,customer_street
varchar(20),customer_city varchar(20));

create table depositor(customer_name varchar(20),acc_no int,
foreign key(customer_name) references customer(customer_name) on delete cascade,
foreign key(acc_no) references accounts(acc_no) on delete cascade);

create table loan(loan_number int primary key,branch_name varchar(20),amount int,
foreign key(branch_name) references branch(branch_name) on delete cascade);

create table borrower(customer_name varchar(20),loan_number int, foreign
key(customer_name) references customer(customer_name) on delete cascade, foreign
key(loan_number) references loan(loan_number) on delete cascade); show tables;


insert into branch values("SBI PD Nagar","Bangalore",200000);
insert into branch values("SBI Rajaji Nagar","Bangalore",500000);
insert into branch values("SBI Jayanagar","Delhi",660000);
insert into branch values("SBI Vijay Nagar","Chennai",870000);
insert into branch values("SBI Hosakerehalli","Bangalore",550000);
select * from branch;

insert into accounts values(11,"SBI Hosakerehalli",5000);
insert into accounts values(22,"SBI Vijay Nagar",5000);
insert into accounts values(33,"SBI Jayanagar",5000);
insert into accounts values(44,"SBI Rajaji Nagar",10000);
insert into accounts values(55,"SBI Vijay Nagar",40000);
insert into accounts values(66,"SBI PD Nagar",4000);
insert into accounts values(77,"SBI PD Nagar",40000);
insert into accounts values(88,"SBI Rajaji Nagar",4000);
select * from accounts;
```

```sql
insert into customer values("Kezar","MG road","Bangalore");
insert into customer values("Lal Krishna","ST MKS road","Bangalore");
insert into customer values("Rahul","Augsten road","Bangalore");
insert into customer values("Lallu","V S road","Bangalore");
insert into customer values("Faizal","Resedency road","Bangalore");
insert into customer values("Rajeev","Dicknsn road","Bangalore");
select * from customer;

insert into depositor values("Rahul",11);
insert into depositor values("Lallu",22);
insert into depositor values("Rahul",33);
insert into depositor values("Faizal",44);
insert into depositor values("Lallu",55);
insert into depositor values("Kezar",66);
insert into depositor values("Rajeev",77);
insert into depositor values("Lal Krishna",88);
select * from depositor;


insert into loan values(10011,"SBI Jayanagar",10000);
insert into loan values(10012,"SBI Vijay Nagar",5000);
insert into loan values(10013,"SBI Hosakerehalli",20000);
insert into loan values(10014,"SBI PD Nagar",15000);
insert into loan values(10015,"SBI Rajaji Nagar",25000);
select * from loan;

insert into borrower values("Kezar",10011);
insert into borrower values("Lal Krishna",10012);
insert into borrower values("Rahul",10013);
insert into borrower values("Lallu",10014);
insert into borrower values("Lal Krishna",10015);
select * from borrower;

-- iii) Find all the customers who have at least two accounts at the Main branch.
select d.customer_name from depositor d,accounts a where d.acc_no=a.acc_no and
a.branch_name="SBI Vijay Nagar"
group by d.customer_name having count(d.customer_name>=2);

-- iv) Find all the customers who have an account at all the
-- branches located in a specific city.
select customer_name from depositor
join accounts on accounts.acc_no = depositor.acc_no
join branch on branch.branch_name = accounts.branch_name
where branch.branch_city = "Bangalore"
GROUP BY depositor.customer_name;
```

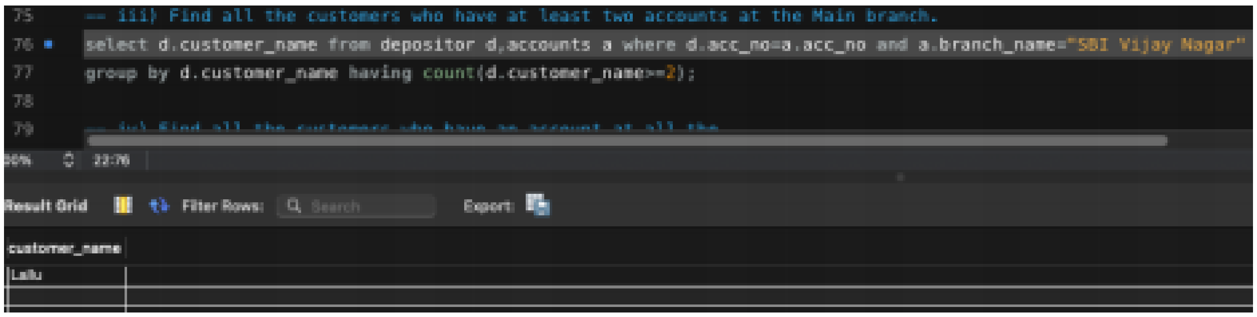
-- v) Demonstrate how you delete all account tuples at every
-- branch located in a specific city.
delete from accounts where branch_name in
(select branch_name from branch where branch_city="delhi");
select * from accounts;



```
select customer_name from depositor
join accounts on accounts.acc_no = depositor.acc_no
join branch on branch.branch_name = accounts.branch_name

100%    35:82
Result Grid    Filter Rows: Search    Export:
customer_name
Rahul
Kezar
Rajeev
Faizal
Lal Krishna
```



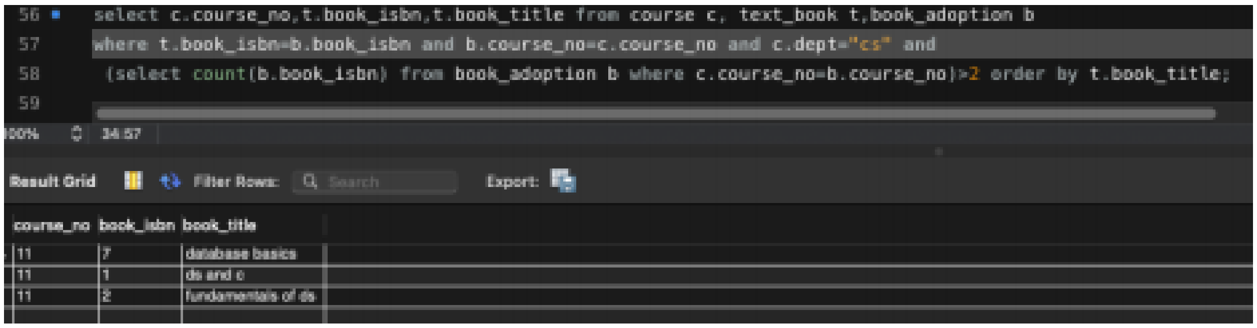
```
-- v) Demonstrate how you delete all account tuples at every
-- branch located in a specific city.
delete from accounts where branch_name in
(select branch_name from branch where branch_city="delhi");
select * from accounts;

100%    12:92
Result Grid    Filter Rows: Search    Edit:    Export/Import:
acc_no  branch_name      balance
22      SBI Vijay Nagar  5000
44      SBI Rajaji Nagar 10000
55      SBI Vijay Nagar  40000
66      SBI PD Nagar     4000
77      SBI PD Nagar     40000
88      SBI Rajaji Nagar 4000
NULL    NULL             NULL
```



```
-- iii) Find all the customers who have at least two accounts at the Main branch.
select d.customer_name from depositor d,accounts a where d.acc_no=a.acc_no and a.branch_name="SBI Vijay Nagar"
group by d.customer_name having count(d.customer_name>=2);

-- iv) Find all the customers who have an account at all the

80%    22:76
Result Grid    Filter Rows: Search    Export:
customer_name
Lallu
```

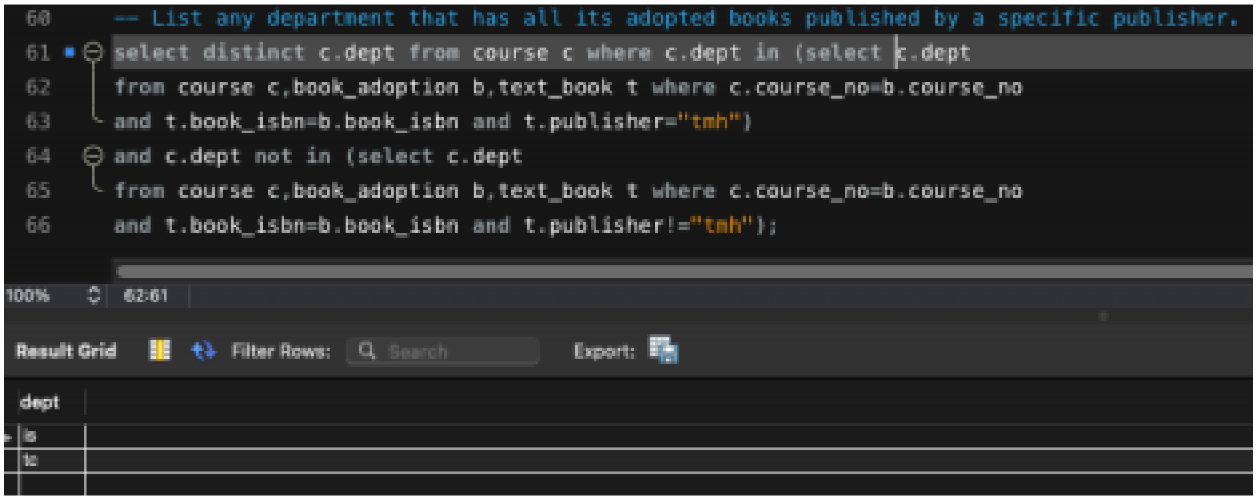
Lab-5 : Student Enroll Database

```
create database Student_Enrollment;
use Student_enrollment;
create table student(regno varchar(10) primary key,name varchar(10),major varchar(10),bdate date);
create table course(course_no int primary key,cname varchar(10),dept varchar(10));
create table enroll(regno varchar(10),course_no int,sem int, marks int,
foreign key(regno) references student(regno) on delete cascade,
foreign key(course_no) references course(course_no) on delete cascade);
create table text_book(book_isbn int primary key,book_title varchar(20),publisher varchar(10),author
varchar(10));
create table book_adoption(course_no int,sem int,book_isbn int ,
foreign key(course_no) references course(course_no) on delete cascade,
foreign key(book_isbn) references text_book(book_isbn) on delete cascade);

insert into student(regno,name,major,bdate) values
("cs01","ram","ds",'1986-03-12'),
("is02","smith","usp",'1987-12-23'),
("ec03","ahmed","sns",'1985-04-17'),
("cs03","sneha","dbms",'1987-01-01'),
("tc05","akhila","ec",'1986-10-06');
select * from student;

insert into course(course_no,cname,dept) values
(11,"ds","cs"),
(22,"usp","is"),
(33,"sns","ec"),
(44,"dbms","cs"),
(55,"ec","tc");
select * from course;

insert into enroll(regno,course_no,sem,marks) values
("cs01",11,4,85),
("is02",22,6,80),
("ec03",33,2,80),
("cs03",44,6,75),
("tc05",55,2,80);
select * from enroll;

insert into text_book(book_isbn,book_title,publisher,author) values
(1,"ds and c","princeton","padma"),
(2,"fundamentals of ds","princeton","godse"),
(3,"fundamentals of dbms","princeton","navathe"),
(4,"sql","princeton","foley"),
(5,"electronic circuits","tmh","elmarsi"),
(6,"adv unix program","tmh","stevens");
select * from text_book;

insert into book_adoption(course_no,sem,book_isbn) values
(11,4,1),(11,4,2),(44,6,3),(44,6,4),(55,2,5),(22,6,6);
select * from book_adoption;

-- Demonstrate how you add a new text book to the database and make this book be adopted by some
department.
insert into text_book values(7,"database basics","princeton","shawn");
insert into book_adoption values(11,4,7);

-- Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order
 -- for courses offered by the ‘CS’ department that use more than two books.
select c.course_no,t.book_isbn,t.book_title from course c, text_book t,book_adoption b where
t.book_isbn=b.book_isbn and b.course_no=c.course_no and c.dept="cs" and (select
count(b.book_isbn) from book_adoption b where c.course_no=b.course_no)>2 order by
t.book_title;

-- List any department that has all its adopted books published by a specific publisher.
select distinct c.dept from course c where c.dept in (select c.dept
from course c,book_adoption b,text_book t where c.course_no=b.course_no
and t.book_isbn=b.book_isbn and t.publisher="tmh")
and c.dept not in (select c.dept
from course c,book_adoption b,text_book t where c.course_no=b.course_no
and t.book_isbn=b.book_isbn and t.publisher!="tmh");
```



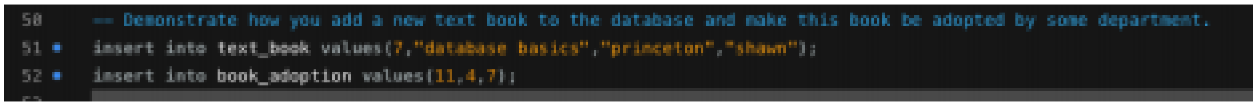
```
select c.course_no,t.book_isbn,t.book_title from course c, text_book t,book_adoption b
where t.book_isbn=b.book_isbn and b.course_no=c.course_no and c.dept="cs" and
 (select count(b.book_isbn) from book_adoption b where c.course_no=b.course_no)>2 order by t.book_title;
```

```
course_no book_isbn book_title
11        7         database basics
11        1         ds and c
11        2         fundamentals of ds
```



```
-- List any department that has all its adopted books published by a specific publisher.
select distinct c.dept from course c where c.dept in (select c.dept
from course c,book_adoption b,text_book t where c.course_no=b.course_no
and t.book_isbn=b.book_isbn and t.publisher="tmh")
and c.dept not in (select c.dept
from course c,book_adoption b,text_book t where c.course_no=b.course_no
and t.book_isbn=b.book_isbn and t.publisher!="tmh");
```

```
dept
is
te
```



```
-- Demonstrate how you add a new text book to the database and make this book be adopted by some department.
insert into text_book values(7,"database basics","princeton","shawn");
insert into book_adoption values(11,4,7);
```

Lab-6 : Movie Database

CREATE DATABASE MOVIE;

USE MOVIE;

CREATE TABLE ACTOR(ACT_ID INT PRIMARY KEY ,ACT_NAME VARCHAR(30),ACT_GENDER VARCHAR(30));

CREATE TABLE DIRECTOR(DIR_ID INT,DIR_NAME VARCHAR(30),PHONE_NO LONG,PRIMARY KEY(DIR_ID));

CREATE TABLE MOVIES(MOVIE_ID INT,MOVIE_TITLE VARCHAR(30),MOVIE_YEAR INT,MOVIE_LANG VARCHAR(30),DIR_ID INT,

PRIMARY KEY(MOVIE_ID),

```
    FOREIGN KEY(DIR_ID) REFERENCES DIRECTOR(DIR_ID) ON UPDATE CASCADE);


CREATE TABLE MOVIE_CAST(ACT_ID INT,MOVIE_ID INT,ROLE VARCHAR(30),

      FOREIGN KEY(ACT_ID) REFERENCES ACTOR(ACT_ID) ON DELETE CASCADE ON UPDATE
CASCADE,

        FOREIGN KEY(MOVIE_ID) REFERENCES MOVIES(MOVIE_ID) ON DELETE CASCADE ON
UPDATE CASCADE);


CREATE TABLE RATING(MOVIE_ID INT,RATING_STARS INT CHECK (RATING_STARS<=5),

    FOREIGN KEY(MOVIE_ID) REFERENCES MOVIES(MOVIE_ID) ON UPDATE CASCADE);


INSERT INTO ACTOR(ACT_ID,ACT_NAME,ACT_GENDER) VALUES

(1, 'Tom Cruise','MALE' ),

(2, 'Leonardo','MALE'),

(3, 'Robert Downey', 'MALE'),

 (4, 'Jennifer Lawrence','FEMALE'),

(5, 'Emma Stone','FEMALE');

select * from ACTOR;


INSERT INTO DIRECTOR(DIR_ID, DIR_NAME, PHONE_NO) VALUES

(1, 'Steven Spielberg', 99988776600),

(2, 'Christopher', 9988776611),
(3, 'Alfred Hitchcock', 9988776622),

(4, 'Tim Burton', 9988776633),

(5, 'James Cameron', 9988776644);

select * from DIRECTOR;

INSERT INTO MOVIES(MOVIE_ID,MOVIE_TITLE,MOVIE_YEAR,MOVIE_LANG,DIR_ID) VALUES

(1,'War of the Worlds', 2005, 'ENG', 1),

(2,'Titanic', 1997, 'ENG', 1),

(3,'Iron Man', 2008, 'ENG', 2),

(4,'Red Sparrow', 2018, 'ENG', 3),

(5,'Spider Man',2015, 'ENG', 4),

(6, 'Avatar', 2009, 'ENG', 5),

(7,'Mission Impossible',2017,'ENG',3);

select * from MOVIES;


INSERT INTO MOVIE_CAST(ACT_ID, MOVIE_ID,ROLE) VALUES

(1, 1, 'LEAD'),

(1, 7, 'LEAD'),

(2, 2, 'LEAD'),

(3, 3, 'LEAD'),

(4, 4, 'LEAD'),

(5, 5, 'LEAD'),

 (5,6,'CO-STAR');
 select * FROM MOVIE_CAST;



INSERT INTO RATING(MOVIE_ID, RATING_STARS) VALUES

(1, 3),

(2, 4),

(3, 5),

(4, 3),

(5, 4),

(6, 4),

(7, 5);


SELECT * FROM RATING;

-- 3. List the titles of all movies directed by ‘Hitchcock’.

SELECT M.MOVIE_TITLE FROM MOVIES M,DIRECTOR D WHERE M.DIR_ID=D.DIR_ID

AND D.DIR_NAME='Alfred Hitchcock';



-- 4. Find the movie names where one or more actors acted in two or more movies.

SELECT M.MOVIE_TITLE FROM ACTOR A,MOVIE_CAST C,MOVIES M WHERE A.ACT_ID=C.ACT_ID
AND

C.MOVIE_ID=M.MOVIE_ID AND A.ACT_ID IN(SELECT ACT_ID FROM MOVIE_CAST GROUP BY
ACT_ID HAVING COUNT(*)>=2);
```

-- 5. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

```
SELECT A.ACT_NAME FROM ACTOR A

JOIN MOVIE_CAST MC ON A.ACT_ID=MC.ACT_ID

JOIN MOVIES M ON MC.MOVIE_ID=M.MOVIE_ID

WHERE M.MOVIE_YEAR NOT BETWEEN 2000 AND 2015;
```

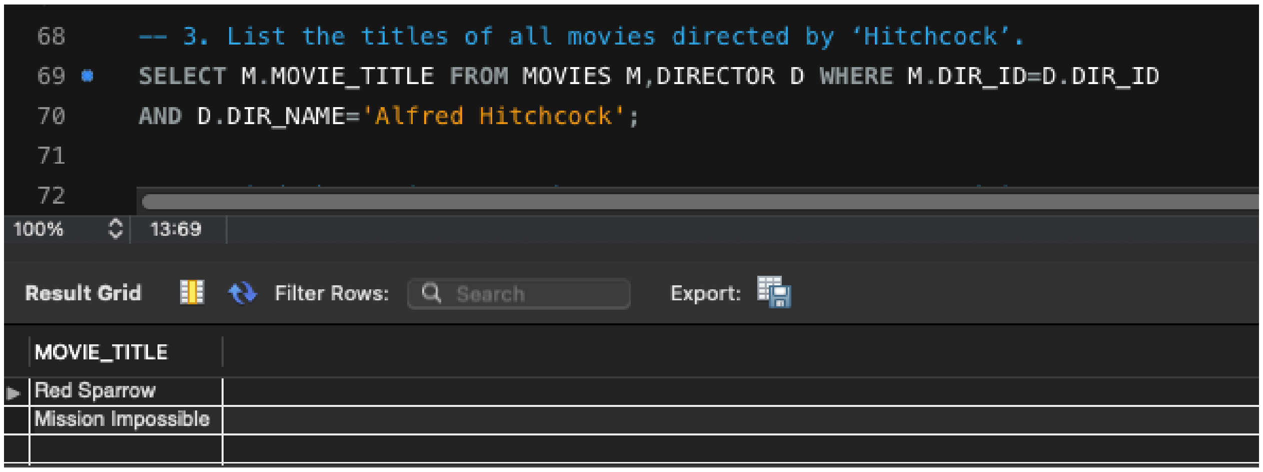
-- 6. Find the title of movies and number of stars for each movie that has at least one rating and find the highest

-- number of stars that movie received. Sort the result by movie title.

```
SELECT M.MOVIE_TITLE, MAX(R.RATING_STARS) AS MAXIMUM_RATING FROM MOVIES M, RATING R

WHERE M.MOVIE_ID = R.MOVIE_ID GROUP BY M.MOVIE_TITLE HAVING COUNT(R.RATING_STARS>=1) ORDER BY M.MOVIE_TITLE;
```

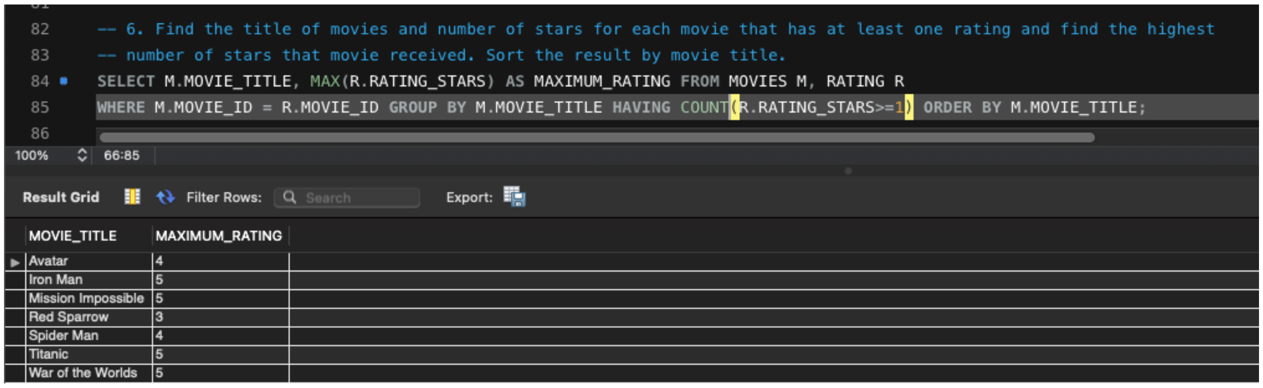
-- 7. Update rating of all movies directed by ‘Steven Spielberg’ to 5.

```
UPDATE RATING SET RATING_STARS = 5 WHERE MOVIE_ID IN

(SELECT M.MOVIE_ID FROM MOVIES M, DIRECTOR D WHERE M.DIR_ID = D.DIR_ID

AND D.DIR_NAME='Steven Spielberg');

SELECT * FROM RATING;
```



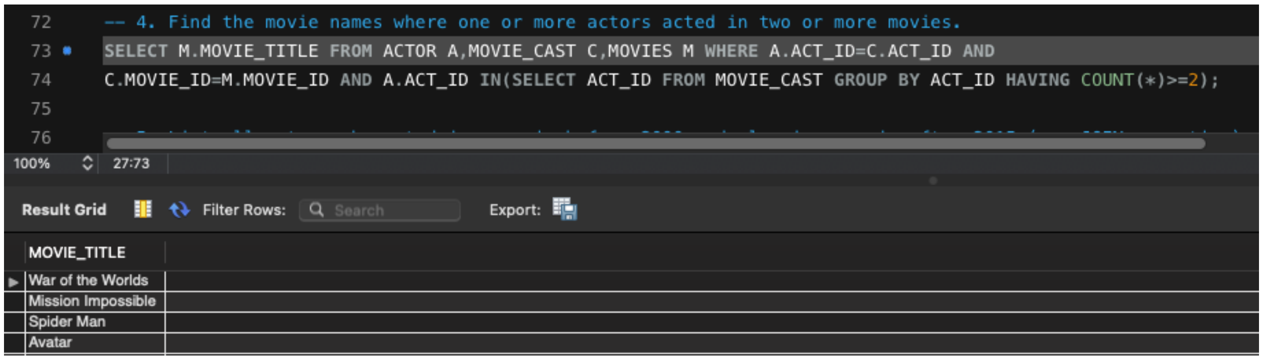
```sql
-- 3. List the titles of all movies directed by ‘Hitchcock’.
SELECT M.MOVIE_TITLE FROM MOVIES M,DIRECTOR D WHERE M.DIR_ID=D.DIR_ID
AND D.DIR_NAME='Alfred Hitchcock';
```

MOVIE_TITLE
Red Sparrow
Mission Impossible



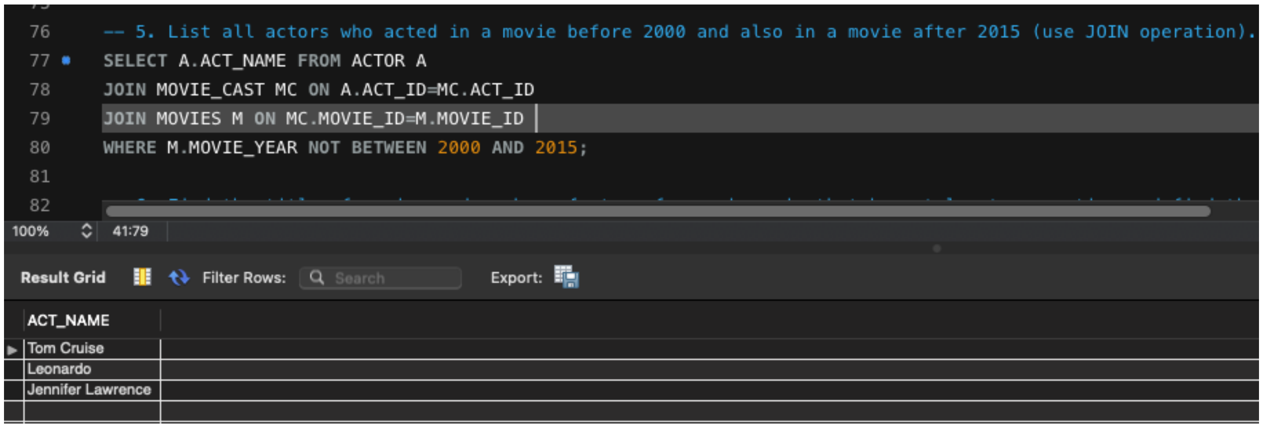
```sql
-- 6. Find the title of movies and number of stars for each movie that has at least one rating and find the highest
-- number of stars that movie received. Sort the result by movie title.
SELECT M.MOVIE_TITLE, MAX(R.RATING_STARS) AS MAXIMUM_RATING FROM MOVIES M, RATING R
WHERE M.MOVIE_ID = R.MOVIE_ID GROUP BY M.MOVIE_TITLE HAVING COUNT(R.RATING_STARS>=1) ORDER BY M.MOVIE_TITLE;
```

MOVIE_TITLE	MAXIMUM_RATING
Avatar	4
Iron Man	5
Mission Impossible	5
Red Sparrow	3
Spider Man	4
Titanic	5
War of the Worlds	5



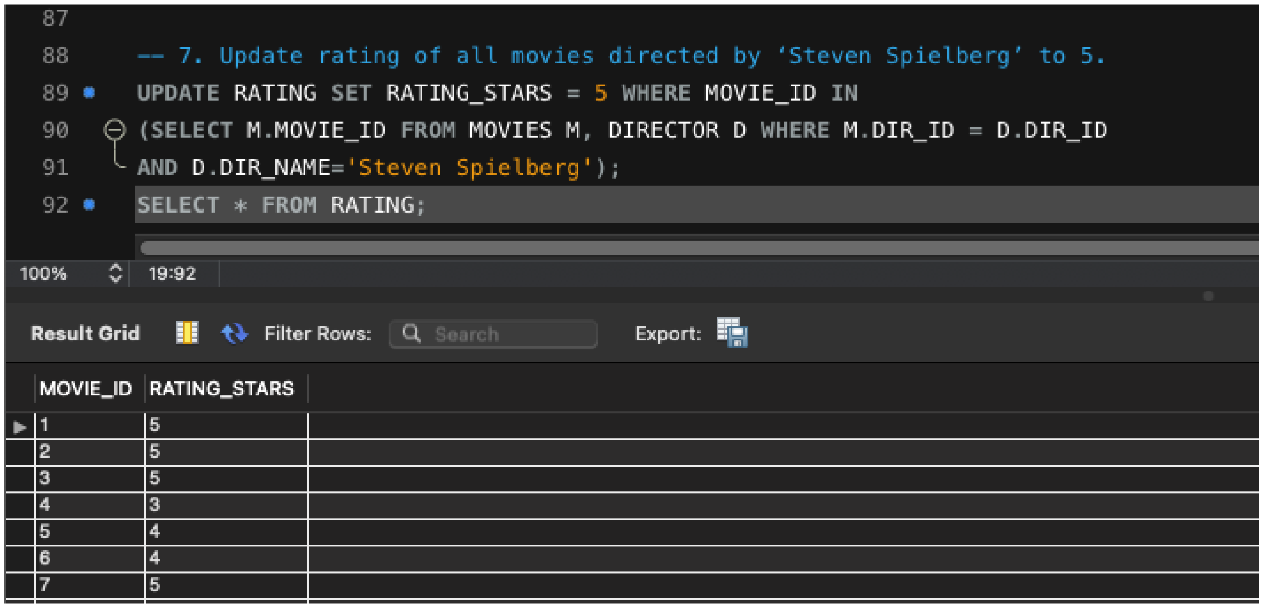
```sql
-- 4. Find the movie names where one or more actors acted in two or more movies.
SELECT M.MOVIE_TITLE FROM ACTOR A,MOVIE_CAST C,MOVIES M WHERE A.ACT_ID=C.ACT_ID AND
C.MOVIE_ID=M.MOVIE_ID AND A.ACT_ID IN(SELECT ACT_ID FROM MOVIE_CAST GROUP BY ACT_ID HAVING COUNT(*)>=2);
```

MOVIE_TITLE
War of the Worlds
Mission Impossible
Spider Man
Avatar



```sql
-- 5. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
SELECT A.ACT_NAME FROM ACTOR A
JOIN MOVIE_CAST MC ON A.ACT_ID=MC.ACT_ID
JOIN MOVIES M ON MC.MOVIE_ID=M.MOVIE_ID
WHERE M.MOVIE_YEAR NOT BETWEEN 2000 AND 2015;
```

ACT_NAME
Tom Cruise
Leonardo
Jennifer Lawrence



```
-- 7. Update rating of all movies directed by 'Steven Spielberg' to 5.
UPDATE RATING SET RATING_STARS = 5 WHERE MOVIE_ID IN
(SELECT M.MOVIE_ID FROM MOVIES M, DIRECTOR D WHERE M.DIR_ID = D.DIR_ID
AND D.DIR_NAME='Steven Spielberg');
SELECT * FROM RATING;
```

100% 19:92

Result Grid Filter Rows: Search Export:

MOVIE_ID	RATING_STARS
1	5
2	5
3	5
4	3
5	4
6	4
7	5

Lab-7 : Airlines Database

create database AIRLINE;

use AIRLINE;

create table flights(flno int ,from_city varchar(20),to_city varchar(20),distance int,

departs time, arrives time ,price int);

create table aircraft(a_id int primary key ,a_name varchar(20),cruisingrange int);

create table employee(e_id int primary key ,e_name varchar(20),salary int);

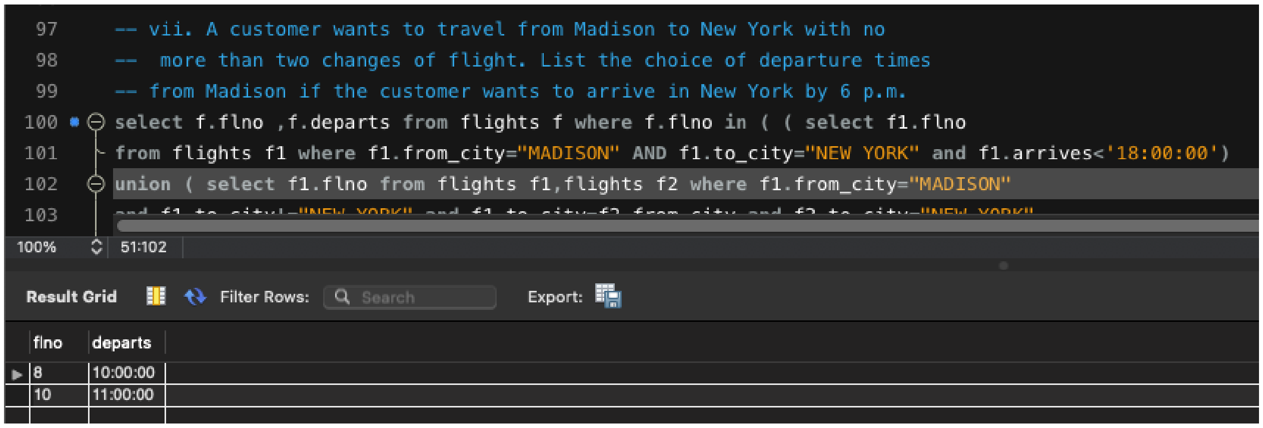
create table certified(e_id int,a_id int,

foreign key(a_id) references aircraft(a_id) on delete cascade,

foreign key(e_id) references employee(e_id) on delete cascade);

```
insert into flights(flno,from_city,to_city,distance,departs,arrives,price)values

(1,'BANGALORE','MANGALORE',360,'10:45:00','12:00:00',10000),

(2,'BANGALORE','DELHI',5000,'12:15:00','04:30:00',25000),

(3,'BANGALORE','MUMBAI',3500,'02:15:00','05:25:00',30000),

(4,'DELHI','MUMBAI',4500,'10:15:00','12:05:00',35000),

(5,'DELHI','FRANKFURT',18000,'07:15:00','05:30:00',90000),

(6,'Mumbai','Delhi',1200,'10:30:00','12:30:00',28000),

(7,'BANGALORE','FRANKFURT',17000,'12:00:00','06:30:00',99000),

(8,'MADISON','NEW YORK', 19000, '10:00:00', '17:00:00', 100000),

(9,'MADISON','NEW YORK', 29000, '10:00:00', '18:30:00', 100000),

(10,'MADISON','LONDON', 30000, '11:00:00', '14:00:00', 55000),

(12,'LONDON','NEW YORK', 30000, '14:05:00', '17:50:00', 50000),

(11,'LONDON','NEW YORK', 31000, '14:06:00', '18:05:00', 51000),

(12,'LONDON','BERLIN', 15000, '14:06:00', '16:05:00', 17000);

select * from flights;


insert into aircraft(a_id,a_name,cruisingrange)values

(111,'AIRBUS',1000),

(222,'BOEING',5000),

(333,'JET01',5000),

(444,'DOUGLAS',8000),
(555,'ANTONOV',500),

(666,'VICKERS',800),

(777,'FOKKER',1000);

select * from aircraft;


insert into employee(e_id,e_name,salary)values (10,'DANNY',80000),

(1,'ARJUN',30000),

(2,'ARPITH',85000),

(3,'BHOOMI',50000),

(4,'HENRY',45000),

(5,'JOMIE',90000),

(6,'ANOSH',75000),

(7,'RICK',100000),

(8,'JANE',70000),

(9,'SOFIE',80000);

select * from employee;


insert into certified(e_id,a_id) values (9,222),

(1,111),

(2,777),

(2,333),

(3,555),
(4,222),

(5,666),

(5,222),

(6,333),

(6,111),

(7,111),

(8,444),

(9,555),

(9,333);

select * from certified;




-- i. Find the names of aircraft such that all pilots certified to

-- operate them have salaries more than Rs.80,000.

select distinct a.a_name from aircraft a,certified c,employee e

where a.a_id=c.a_id and c.e_id=e.e_id and e.salary>80000;


-- ii. For each pilot who is certified for more than three aircrafts, find the

--  eid and the maximum cruising range of the aircraft for which she or he is certified.

select e.e_id,max(a.cruisingrange) from aircraft a,employee e,certified c

where a.a_id=c.a_id and e.e_id=c.e_id group by e.e_id having count(e.e_id)>3;
```

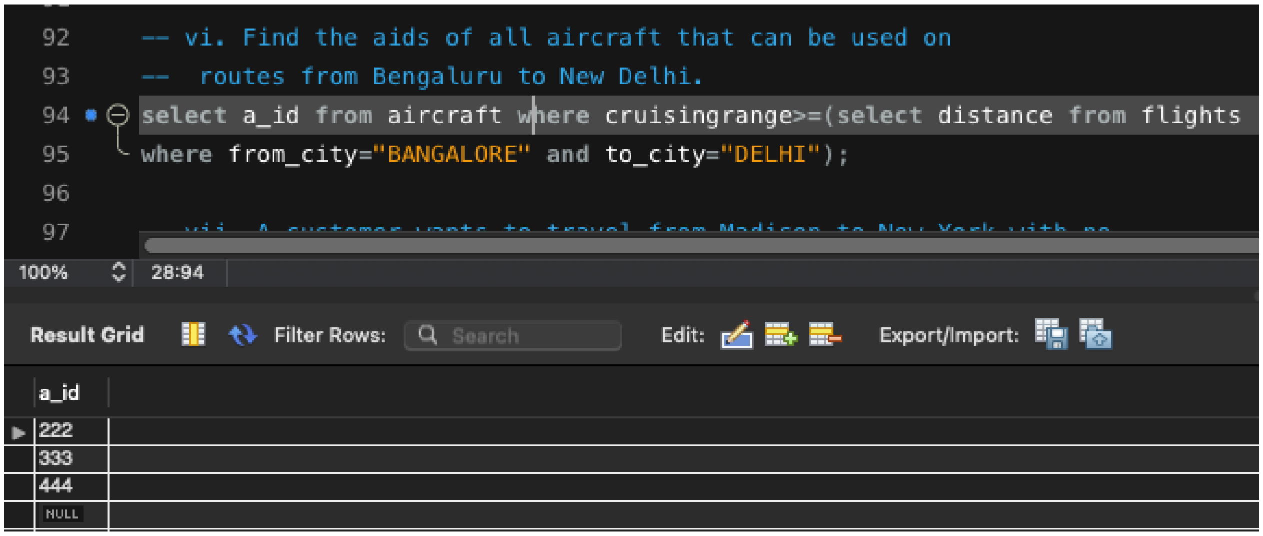
```
-- iii. Find the names of pilots whose salary is less than the price of the

-- cheapest route from Bengaluru to Frankfurt.

select e.e_name from employee e where e.e_id in(select e_id from certified)

and salary<(select min(price) from flights where from_city="BANGALORE" and

to_city="FRANKFURT");


-- iv. For all aircraft with cruising range over 1000 Kms, find the name of the

-- aircraft and the average salary of all pilots certified for this aircraft.

select a.a_name,avg(e.salary) from aircraft a,employee e,certified c

where a.a_id=c.a_id and e.e_id=c.e_id and a.cruisingrange>1000 group by a.a_name;


-- v. Find the names of pilots certified for some Boeing aircraft.

select e.e_name from aircraft a,employee e,certified c

where a.a_id=c.a_id and e.e_id=c.e_id and a.a_name="BOEING";


-- vi. Find the aids of all aircraft that can be used on

--  routes from Bengaluru to New Delhi.

select a_id from aircraft where cruisingrange>=(select distance from flights

where from_city="BANGALORE" and to_city="DELHI");


-- vii. A customer wants to travel from Madison to New York with no

--  more than two changes of flight. List the choice of departure times
-- from Madison if the customer wants to arrive in New York by 6 p.m.

select f.flno ,f.departs from flights f where f.flno in ( ( select f1.flno

from flights f1 where f1.from_city="MADISON" AND f1.to_city="NEW YORK" and f1.arrives<'18:00:00')

union ( select f1.flno from flights f1,flights f2 where f1.from_city="MADISON"

and f1.to_city!="NEW YORK" and f1.to_city=f2.from_city and f2.to_city="NEW YORK"

and f2.departs>f1.arrives and f2.arrives<'18:00:00'));



-- viii. Print the name and salary of every non-pilot whose

-- salary is more than the average salary for pilots.

select e_name from employee where e_id not in(select e_id from certified)

and salary>(select avg(salary) from employee where e_id in(select e_id from certified));
```



```
-- vii. A customer wants to travel from Madison to New York with no
--  more than two changes of flight. List the choice of departure times
-- from Madison if the customer wants to arrive in New York by 6 p.m.
select f.flno ,f.departs from flights f where f.flno in ( ( select f1.flno
from flights f1 where f1.from_city="MADISON" AND f1.to_city="NEW YORK" and f1.arrives<'18:00:00')
union ( select f1.flno from flights f1,flights f2 where f1.from_city="MADISON"
and f1.to_city!="NEW YORK" and f1.to_city=f2.from_city and f2.to_city="NEW YORK"
```

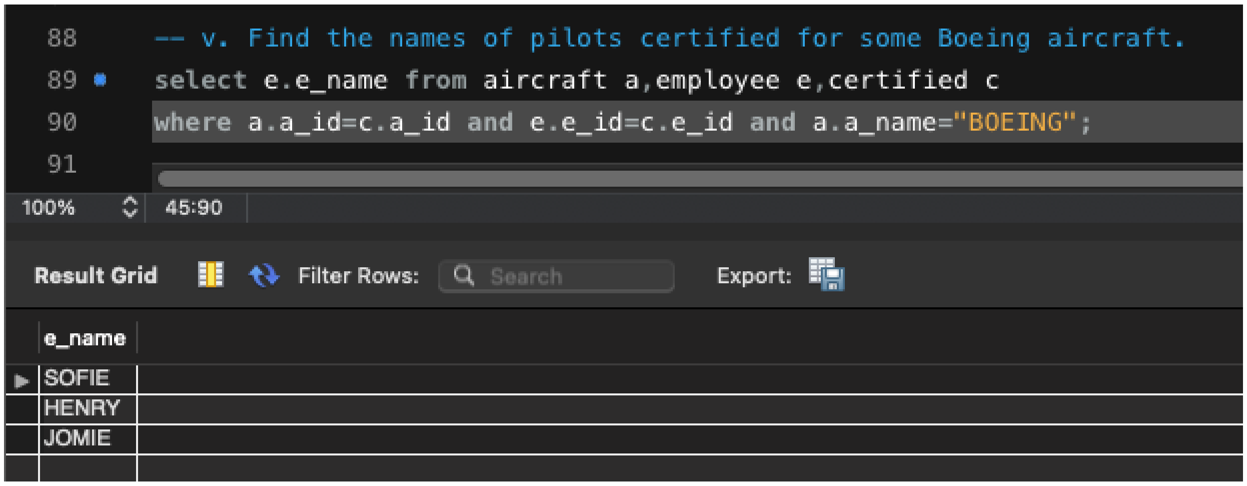
100% 51:102

```
flno  departs
8     10:00:00
10    11:00:00
```



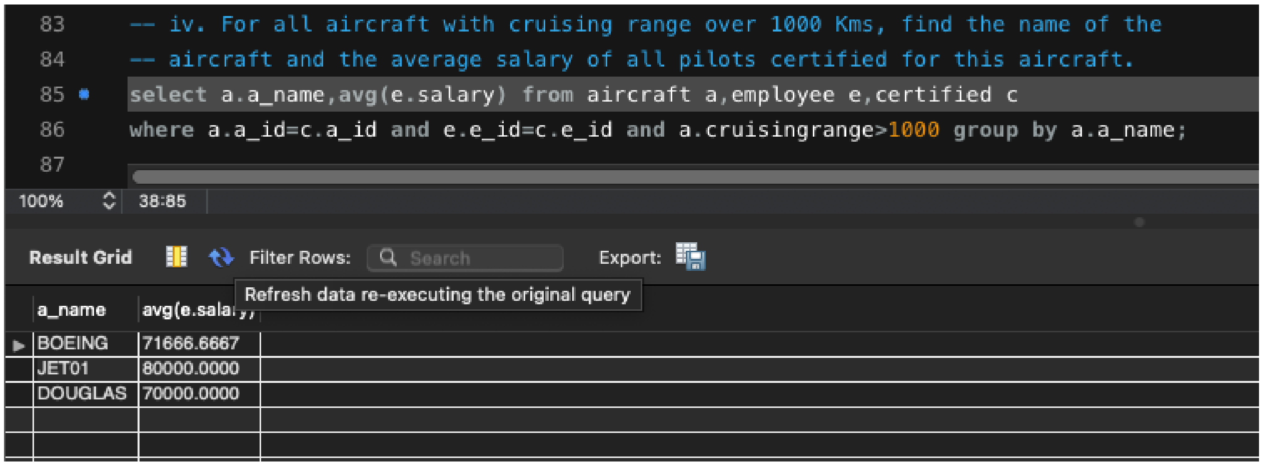
```
-- vi. Find the aids of all aircraft that can be used on
--  routes from Bengaluru to New Delhi.
select a_id from aircraft where cruisingrange>=(select distance from flights
where from_city="BANGALORE" and to_city="DELHI");
```

```
a_id
222
333
444
NULL
```



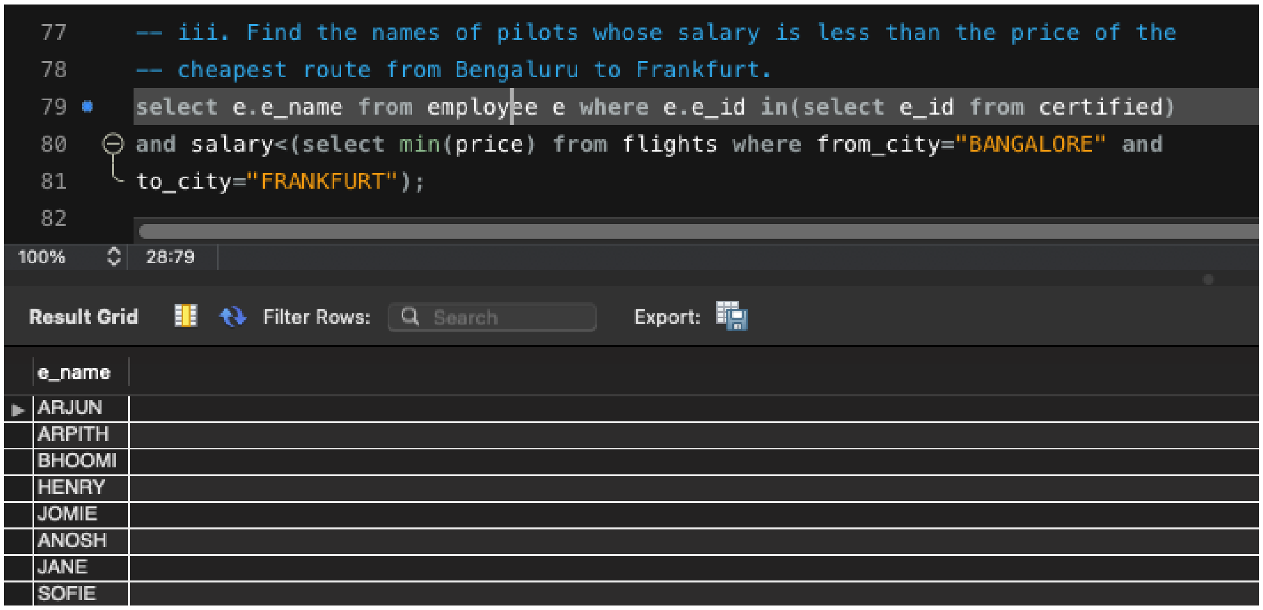
```
-- v. Find the names of pilots certified for some Boeing aircraft.
select e.e_name from aircraft a,employee e,certified c
where a.a_id=c.a_id and e.e_id=c.e_id and a.a_name="BOEING";
```

```
e_name
SOFIE
HENRY
JOMIE
```



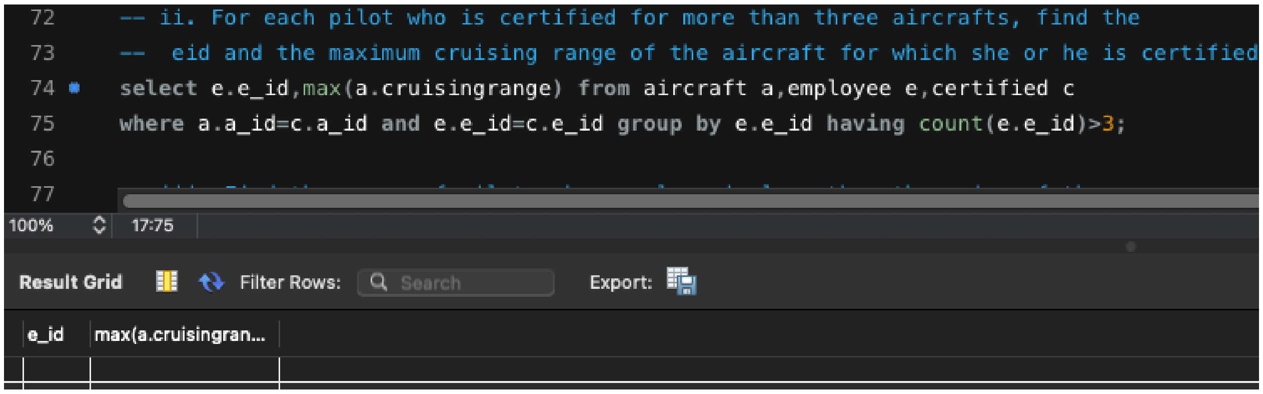
```
-- iv. For all aircraft with cruising range over 1000 Kms, find the name of the
-- aircraft and the average salary of all pilots certified for this aircraft.
select a.a_name,avg(e.salary) from aircraft a,employee e,certified c
where a.a_id=c.a_id and e.e_id=c.e_id and a.cruisingrange>1000 group by a.a_name;
```

```
a_name    avg(e.salary)
BOEING    71666.6667
JET01     80000.0000
DOUGLAS   70000.0000
```



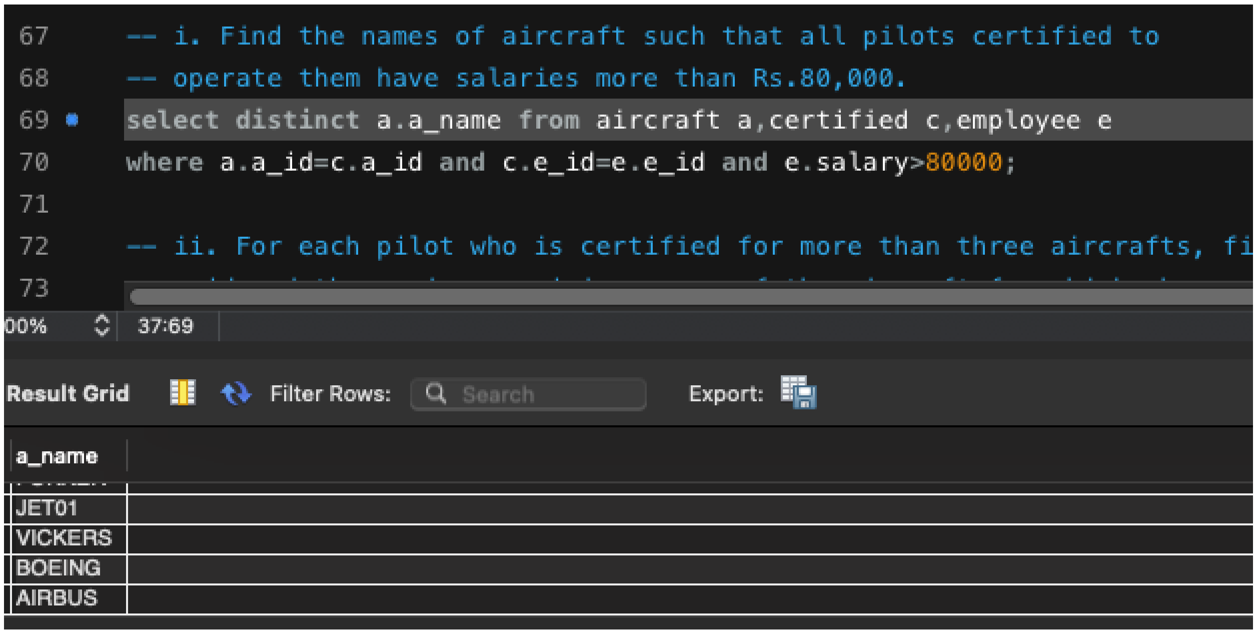
```sql
-- iii. Find the names of pilots whose salary is less than the price of the
-- cheapest route from Bengaluru to Frankfurt.
select e.e_name from employee e where e.e_id in(select e_id from certified)
and salary<(select min(price) from flights where from_city="BANGALORE" and
to_city="FRANKFURT");
```

```
e_name
ARJUN
ARPITH
BHOOMI
HENRY
JOMIE
ANOSH
JANE
SOFIE
```



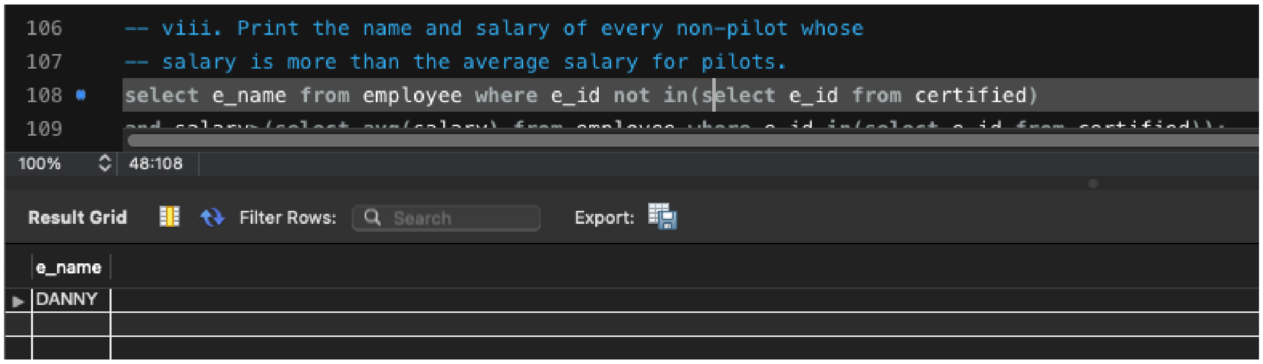
```sql
-- ii. For each pilot who is certified for more than three aircrafts, find the
--  eid and the maximum cruising range of the aircraft for which she or he is certified
select e.e_id,max(a.cruisingrange) from aircraft a,employee e,certified c
where a.a_id=c.a_id and e.e_id=c.e_id group by e.e_id having count(e.e_id)>3;
```

```
e_id    max(a.cruisingran...
```



```sql
-- i. Find the names of aircraft such that all pilots certified to
-- operate them have salaries more than Rs.80,000.
select distinct a.a_name from aircraft a,certified c,employee e
where a.a_id=c.a_id and c.e_id=e.e_id and e.salary>80000;

-- ii. For each pilot who is certified for more than three aircrafts, fi
```

```
a_name
JET01
VICKERS
BOEING
AIRBUS
```



```sql
-- viii. Print the name and salary of every non-pilot whose
-- salary is more than the average salary for pilots.
select e_name from employee where e_id not in(select e_id from certified)
and salary>(select avg(salary) from employee where e_id in(select e_id from certified));
```

```
e_name
DANNY
```