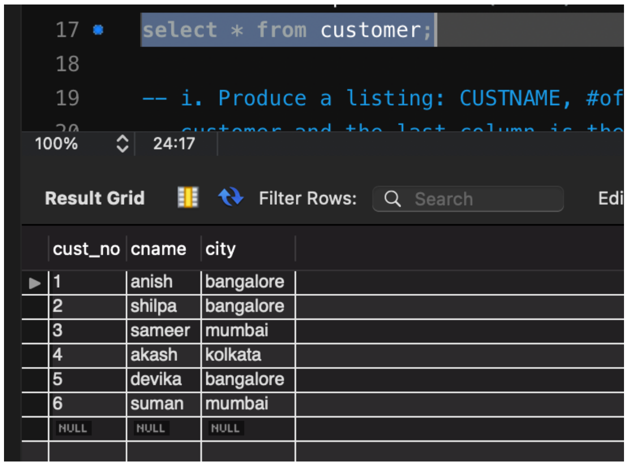
# DBMS RECORD
# (6-10 PROGRAMS)

RAHUL PRAKASHA
1BM18CS078
DBMS LAB
19CS4PCDBM

## PROGRAM NO 6. ORDER PROCESSING

```
CREATE DATABASE ORDER_PROCESSING;
USE ORDER_PROCESSING;
create table customer(cust_no int, cname varchar(20), city varchar(20), primary key(cust_no));
create table orders(order_no int, odate date, cust_no int, ord_amt int, primary key(order_no),
foreign key(cust_no) references customer(cust_no));
create table item(item_no int, unit_price int, primary key(item_no));
create table order_item(order_no int, item_no int , qty int, foreign key(order_no) references
orders(order_no), foreign key(item_no) references item(item_no) on delete set NULL);
create table warehouse(warehouse_no int, city varchar(20), primary key(warehouse_no));
create table shipment(order_no int, warehouse_no int, shit_date date, foreign key(order_no)
references orders(order_no), foreign key(warehouse_no) references
warehouse(warehouse_no));
show tables;

insert into customer values('1','anish','bangalore');
insert into orders values('100','20-03-21','1','6000');
insert into item values('10','100');
insert into order_item values('100','10','1');
insert into warehouse values('1001','bangalore');
insert into shipment values('100','1001','25-03-21');

-- i. 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(*) as NO_OF_ORDERS, avg(O.ord_amt) as AVG_ORDER_AMT
from customer C, orders O
where (C.cust_no = O.cust_no) group by cname;

-- ii. List the order# for orders that were shipped from all warehouses that the company has in a
specific city.

select * from orders where order_no in (
select order_no from shipment where warehouse_no in (
select warehouse_no from warehouse where city='bangalore'));

-- v. Demonstrate how you delete item# 10 from the ITEM table and make that field null in the
ORDER_ITEM table.

delete from item where item_no = 10;
```

OUTPUT:

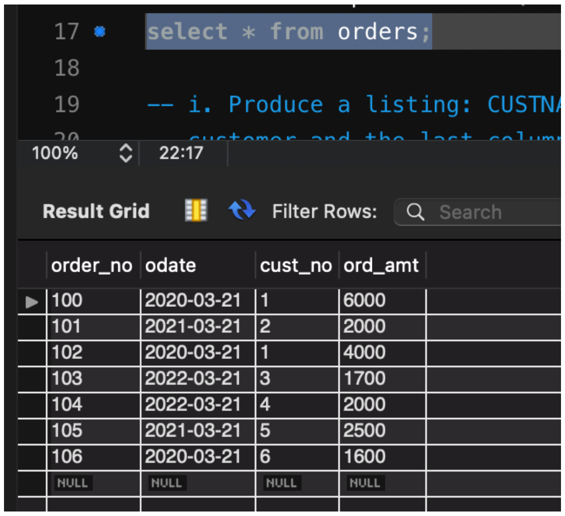


```
select * from customer;

-- i. Produce a listing: CUSTNAME, #of
```

100% 24:17

Result Grid Filter Rows: Search Edi

| cust_no | cname | city |
|---|---|---|
| 1 | anish | bangalore |
| 2 | shilpa | bangalore |
| 3 | sameer | mumbai |
| 4 | akash | kolkata |
| 5 | devika | bangalore |
| 6 | suman | mumbai |
| NULL | NULL | NULL |

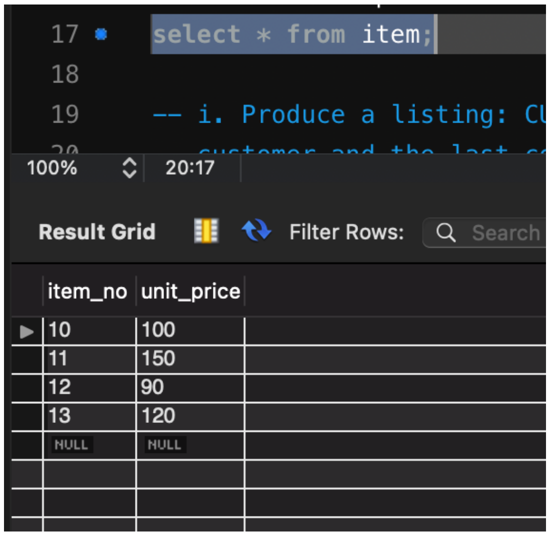


```
select * from orders;

-- i. Produce a listing: CUSTNA
```

100% 22:17

Result Grid Filter Rows: Search

| order_no | odate | cust_no | ord_amt |
|---|---|---|---|
| 100 | 2020-03-21 | 1 | 6000 |
| 101 | 2021-03-21 | 2 | 2000 |
| 102 | 2020-03-21 | 1 | 4000 |
| 103 | 2022-03-21 | 3 | 1700 |
| 104 | 2022-03-21 | 4 | 2000 |
| 105 | 2021-03-21 | 5 | 2500 |
| 106 | 2020-03-21 | 6 | 1600 |
| NULL | NULL | NULL | NULL |

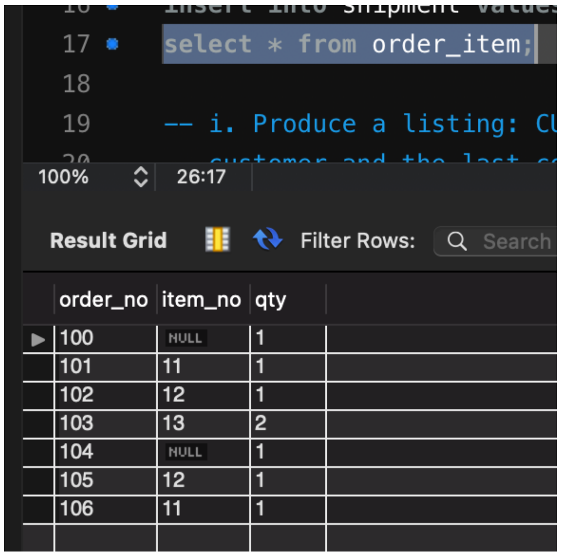


```
select * from item;

-- i. Produce a listing: CU
```

100% 20:17

Result Grid Filter Rows: Search

| item_no | unit_price |
|---|---|
| 10 | 100 |
| 11 | 150 |
| 12 | 90 |
| 13 | 120 |
| NULL | NULL |

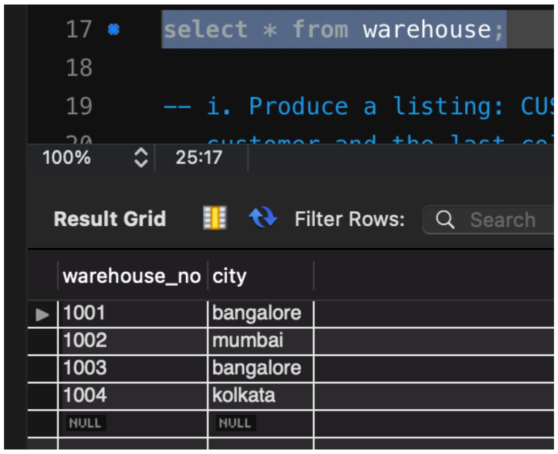


```
select * from order_item;

-- i. Produce a listing: CU
```

100% 26:17

Result Grid Filter Rows: Search

| order_no | item_no | qty |
|---|---|---|
| 100 | NULL | 1 |
| 101 | 11 | 1 |
| 102 | 12 | 1 |
| 103 | 13 | 2 |
| 104 | NULL | 1 |
| 105 | 12 | 1 |
| 106 | 11 | 1 |

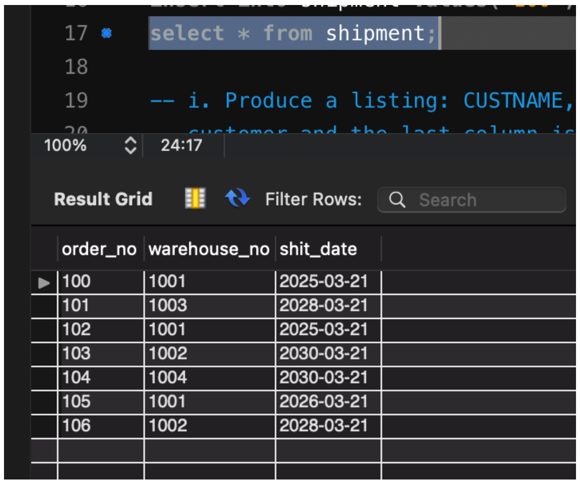


```
select * from warehouse;

-- i. Produce a listing: CUS
```

100% 25:17

Result Grid Filter Rows: Search

| warehouse_no | city |
|---|---|
| 1001 | bangalore |
| 1002 | mumbai |
| 1003 | bangalore |
| 1004 | kolkata |
| NULL | NULL |



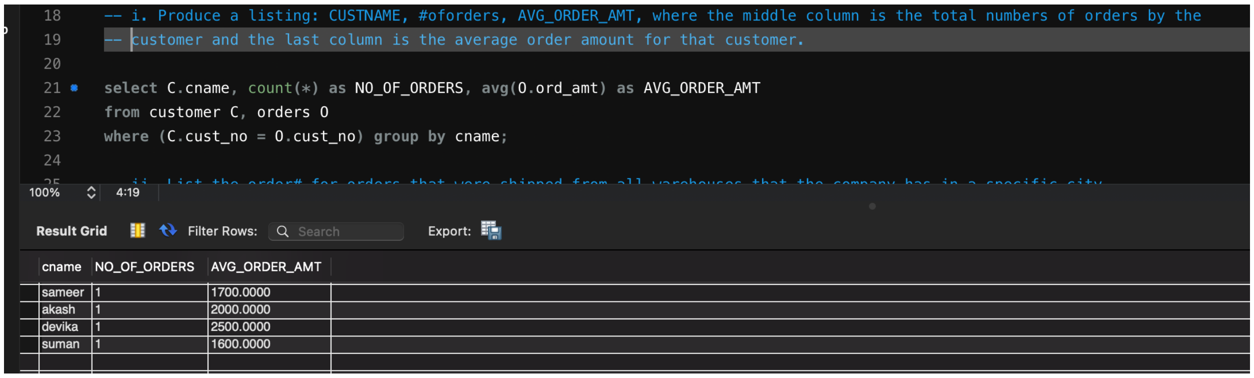
```
select * from shipment;

-- i. Produce a listing: CUSTNAME,
```

100% 24:17

Result Grid Filter Rows: Search

| order_no | warehouse_no | shit_date |
|---|---|---|
| 100 | 1001 | 2025-03-21 |
| 101 | 1003 | 2028-03-21 |
| 102 | 1001 | 2025-03-21 |
| 103 | 1002 | 2030-03-21 |
| 104 | 1004 | 2030-03-21 |
| 105 | 1001 | 2026-03-21 |
| 106 | 1002 | 2028-03-21 |

QUERIES:

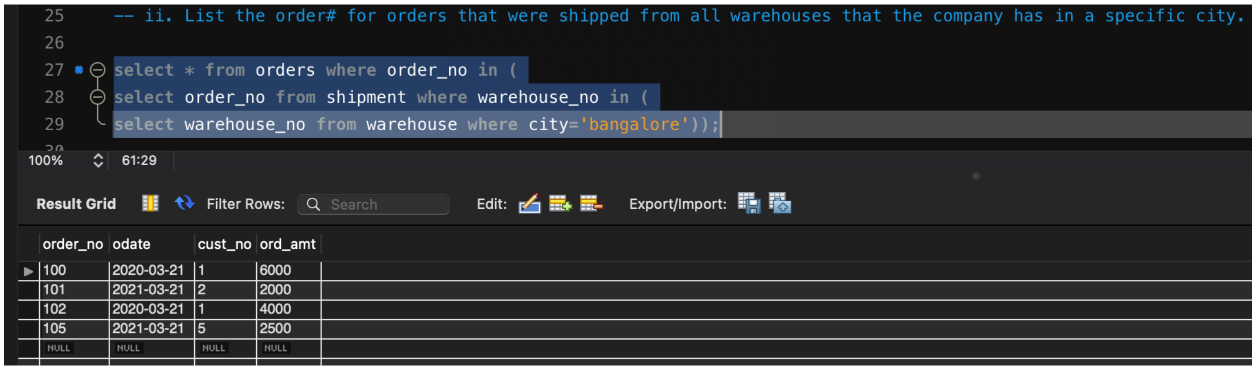


```sql
-- i. Produce a listing: CUSTNAME, #oforders, AVG_ORDER_AMT, where the middle column is the total numbers of orders by the
-- customer and the last column is the average order amount for that customer.

select C.cname, count(*) as NO_OF_ORDERS, avg(O.ord_amt) as AVG_ORDER_AMT
from customer C, orders O
where (C.cust_no = O.cust_no) group by cname;

-- ii. List the order# for orders that were shipped from all warehouses that the company has in a specific city
```

100%    4:19

Result Grid    Filter Rows: Search    Export:

```
cname   NO_OF_ORDERS  AVG_ORDER_AMT
sameer  1             1700.0000
akash   1             2000.0000
devika  1             2500.0000
suman   1             1600.0000
```



```sql
-- ii. List the order# for orders that were shipped from all warehouses that the company has in a specific city.

select * from orders where order_no in (
select order_no from shipment where warehouse_no in (
select warehouse_no from warehouse where city='bangalore'));
```

100%    61:29

Result Grid    Filter Rows: Search    Edit:    Export/Import:

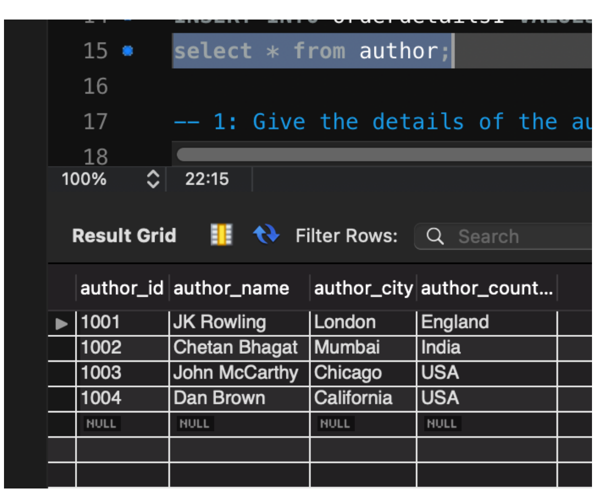
```
order_no  odate       cust_no  ord_amt
100       2020-03-21  1        6000
101       2021-03-21  2        2000
102       2020-03-21  1        4000
105       2021-03-21  5        2500
NULL      NULL        NULL     NULL
```

PROGRAM NO 7. BOOK DEALER

```
create database book;
use book;
CREATE TABLE author(author_id INT,author_name VARCHAR(20),author_city
VARCHAR(20),author_country VARCHAR(20),PRIMARY KEY(author_id));
CREATE TABLE publisher(publisher_id INT,publisher_name VARCHAR(20),publisher_city
VARCHAR(20),publisher_country VARCHAR(20),PRIMARY KEY(publisher_id));
CREATE TABLE category(category_id INT,description VARCHAR(30),PRIMARY KEY(category_id) );
CREATE TABLE catalogue(book_id INT,book_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 orderdetails1(order_id INT,book_id INT,quantity INT,PRIMARY
KEY(order_id),FOREIGN KEY(book_id) REFERENCES catalogue(book_id));
show tables;

INSERT INTO author VALUES(1001,'JK Rowling','London','England');
INSERT INTO publisher VALUES(2001,'Bloomsbury','London','England');
INSERT INTO category VALUES(3001,'Fiction');
INSERT INTO catalogue VALUES(4001,'HP and Goblet Of Fire',1001,2001,3001,2002,600);
INSERT INTO orderdetails1 VALUES(5001,4001,5);
select * from publisher;

-- 1: Give the details of the authors who have 2 or more books in the catalog and the price of
the books is greater than
-- the average price of the books in the catalog and the year of publication is after 2000

 SELECT * FROM author
                    WHERE author_id IN
      (SELECT author_id FROM catalogue WHERE
      year>2000 AND price>
      (SELECT AVG(price) FROM catalogue)
      GROUP BY author_id HAVING COUNT(*)>1);

-- 2: Find the author of the book which has maximum sales.

 SELECT author_name FROM author a,catalogue c WHERE a.author_id=c.author_id AND
book_id IN (SELECT book_id FROM orderdetails1 WHERE
 quantity= (SELECT MAX(quantity) FROM orderdetails1));

-- 3: Demonstrate how you increase the price of books published by a specific publisher1 by
10%.
```

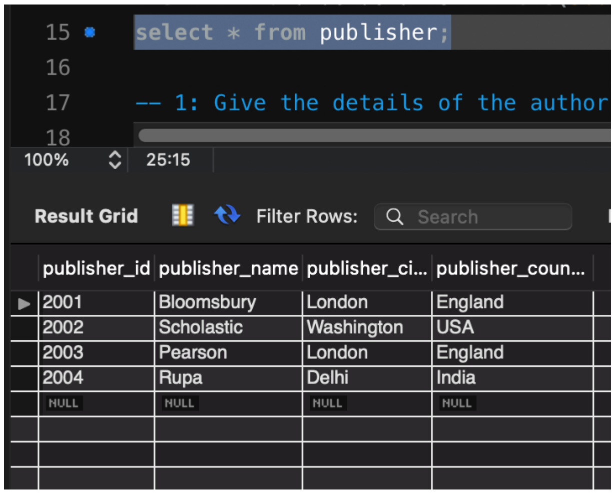
```
UPDATE catalogue SET price=1.1*price
WHERE publisher_id IN
(SELECT publisher_id FROM publisher WHERE
publisher_name='pearson');

select * from catalogue;
```

OUTPUT:



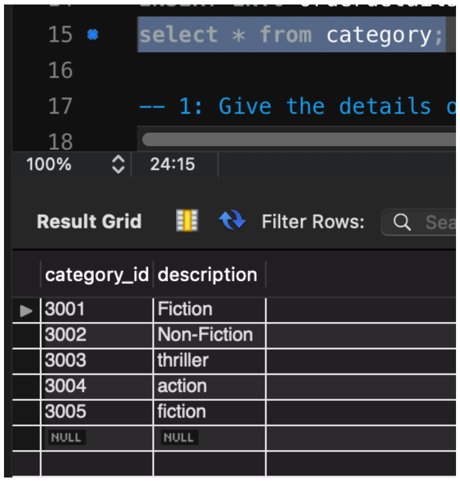
```
select * from author;

-- 1: Give the details of the au
```

| author_id | author_name | author_city | author_count... |
|---|---|---|---|
| 1001 | JK Rowling | London | England |
| 1002 | Chetan Bhagat | Mumbai | India |
| 1003 | John McCarthy | Chicago | USA |
| 1004 | Dan Brown | California | USA |
| NULL | NULL | NULL | NULL |



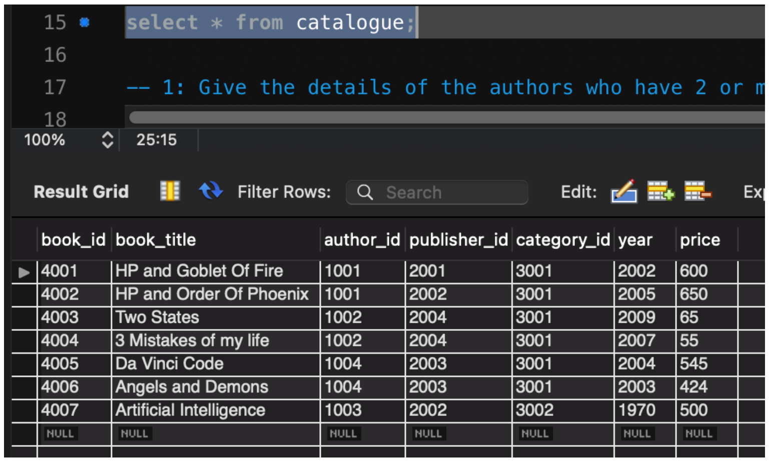
```
select * from publisher;

-- 1: Give the details of the author
```

| publisher_id | publisher_name | publisher_ci... | publisher_coun... |
|---|---|---|---|
| 2001 | Bloomsbury | London | England |
| 2002 | Scholastic | Washington | USA |
| 2003 | Pearson | London | England |
| 2004 | Rupa | Delhi | India |
| NULL | NULL | NULL | NULL |



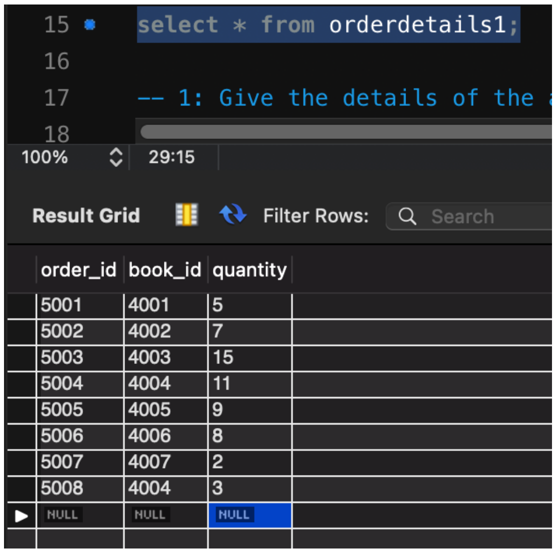
```
select * from category;

-- 1: Give the details o
```

| category_id | description |
|---|---|
| 3001 | Fiction |
| 3002 | Non-Fiction |
| 3003 | thriller |
| 3004 | action |
| 3005 | fiction |
| NULL | NULL |



```
select * from catalogue;

-- 1: Give the details of the authors who have 2 or m
```

| book_id | book_title | author_id | publisher_id | category_id | year | price |
|---|---|---|---|---|---|---|
| 4001 | HP and Goblet Of Fire | 1001 | 2001 | 3001 | 2002 | 600 |
| 4002 | HP and Order Of Phoenix | 1001 | 2002 | 3001 | 2005 | 650 |
| 4003 | Two States | 1002 | 2004 | 3001 | 2009 | 65 |
| 4004 | 3 Mistakes of my life | 1002 | 2004 | 3001 | 2007 | 55 |
| 4005 | Da Vinci Code | 1004 | 2003 | 3001 | 2004 | 545 |
| 4006 | Angels and Demons | 1004 | 2003 | 3001 | 2003 | 424 |
| 4007 | Artificial Intelligence | 1003 | 2002 | 3002 | 1970 | 500 |
| NULL | NULL | NULL | NULL | NULL | NULL | NULL |



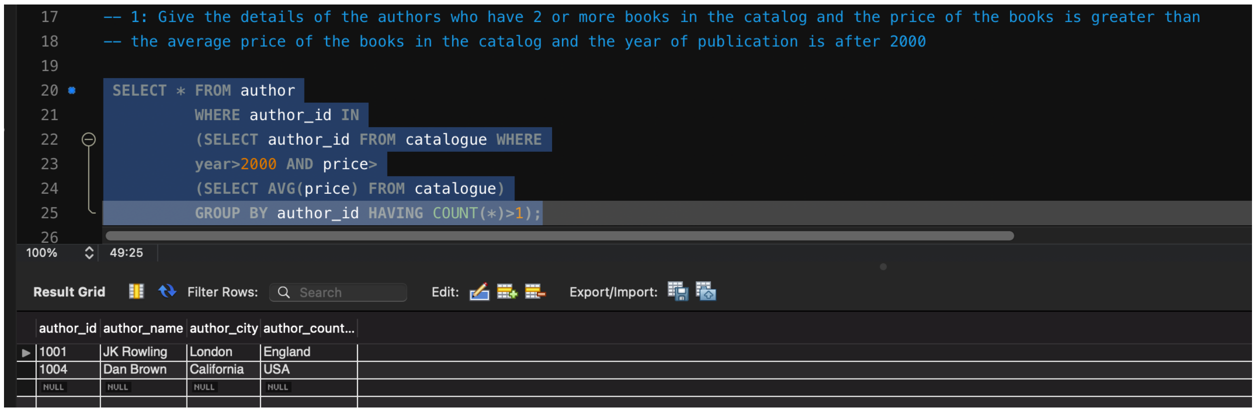
```
select * from orderdetails1;

-- 1: Give the details of the a
```

100% 29:15

Result Grid Filter Rows: Search

```
order_id book_id quantity
5001     4001    5
5002     4002    7
5003     4003    15
5004     4004    11
5005     4005    9
5006     4006    8
5007     4007    2
5008     4004    3
NULL     NULL    NULL
```

QUERIES:

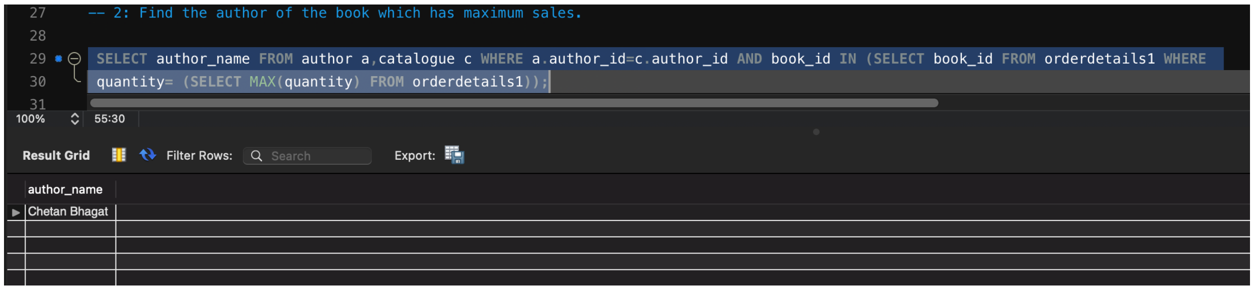


```
-- 1: Give the details of the authors who have 2 or more books in the catalog and the price of the books is greater than
-- the average price of the books in the catalog and the year of publication is after 2000

SELECT * FROM author
        WHERE author_id IN
        (SELECT author_id FROM catalogue WHERE
        year>2000 AND price>
        (SELECT AVG(price) FROM catalogue)
        GROUP BY author_id HAVING COUNT(*)>1);
```

100% 49:25

Result Grid Filter Rows: Search Edit: Export/Import:

```
author_id author_name author_city author_count...
1001      JK Rowling  London      England
1004      Dan Brown   California  USA
NULL      NULL        NULL        NULL
```



```
-- 2: Find the author of the book which has maximum sales.

SELECT author_name FROM author a,catalogue c WHERE a.author_id=c.author_id AND book_id IN (SELECT book_id FROM orderdetails1 WHERE
quantity= (SELECT MAX(quantity) FROM orderdetails1));
```

100% 55:30

Result Grid Filter Rows: Search Export:

```
author_name
Chetan Bhagat
```

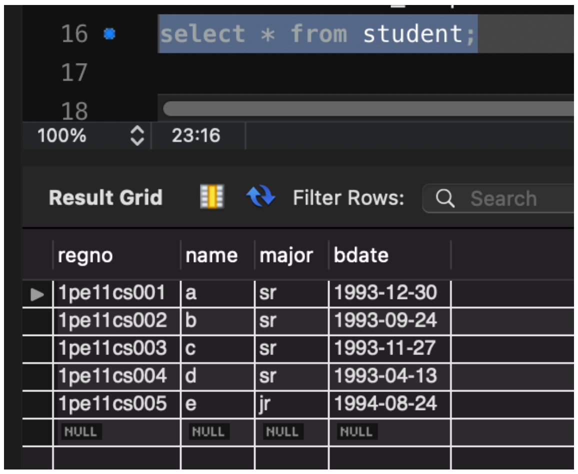
PROGRAM NO 8. STUDENT ENROLLMENT

```
Create database student_enrollment;
use student_enrollment;

CREATE TABLE student(regno VARCHAR(15),name VARCHAR(20),major VARCHAR(20),bdate
DATE,PRIMARY KEY (regno) );
CREATE TABLE course(courseno INT,cname VARCHAR(20),dept VARCHAR(20),PRIMARY KEY
(courseno) );
CREATE TABLE enroll(regno VARCHAR(15),courseno INT,sem INT,marks INT,PRIMARY KEY
(regno,courseno),FOREIGN KEY (regno) REFERENCES student (regno),FOREIGN KEY (courseno)
REFERENCES course (courseno));
CREATE TABLE text(book_isbn INT(5),book_title VARCHAR(20),publisher VARCHAR(20),author
VARCHAR(20),PRIMARY KEY (book_isbn) );
CREATE TABLE book_adoption(courseno INT,sem INT,book_isbn INT,PRIMARY KEY
(courseno,book_isbn),FOREIGN KEY (courseno) REFERENCES course (courseno),FOREIGN KEY
(book_isbn) REFERENCES text(book_isbn) );
show tables;

INSERT INTO student VALUES('1pe11cs001','a','sr',19931230);
INSERT INTO course VALUES (111,'OS','CSE');
INSERT INTO text VALUES(10,'DATABASE SYSTEMS','PEARSON','SCHIELD');
INSERT INTO enroll VALUES ('1pe11cs001',115,3,100);
INSERT INTO book_adoption VALUES(111,5,900);
select * from book_adoption;

-- 1. 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.courseno,t.book_isbn,t.book_title
   FROM course c,book_adoption ba,text t
   WHERE c.courseno=ba.courseno
   AND ba.book_isbn=t.book_isbn
   AND c.dept='CSE'
   AND 2<(
   SELECT COUNT(book_isbn)
   FROM book_adoption b
   WHERE c.courseno=b.courseno)
   ORDER BY t.book_title;
```

```
-- 2. Demonstrate how you add a new text book to the database and make this book be
adopted
-- by some department.
        INSERT INTO text
        VALUES (11,'DATABASE SYSTEMS','GRB','SCHIELD');
        INSERT INTO book_adoption (courseno,sem,book_isbn) VALUES(111,5,11);
  select * from book_adoption;
```

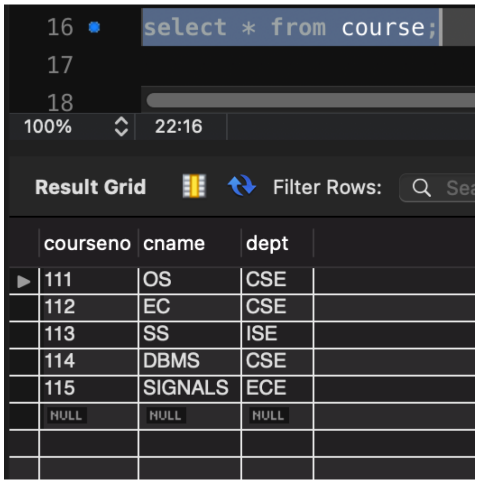
```
-- 3. List any department that has all its adopted books published by a specific publisher.
  SELECT DISTINCT c.dept
  FROM course c
  WHERE c.dept IN
  ( SELECT c.dept
  FROM course c,book_adoption b,text t
  WHERE c.courseno=b.courseno
  AND t.book_isbn=b.book_isbn
  AND t.publisher='PEARSON')
  AND c.dept NOT IN
  (SELECT c.dept
  FROM course c,book_adoption b,text t
  WHERE c.courseno=b.courseno
  AND t.book_isbn=b.book_isbn
  AND t.publisher != 'PEARSON');
```

OUTPUT:



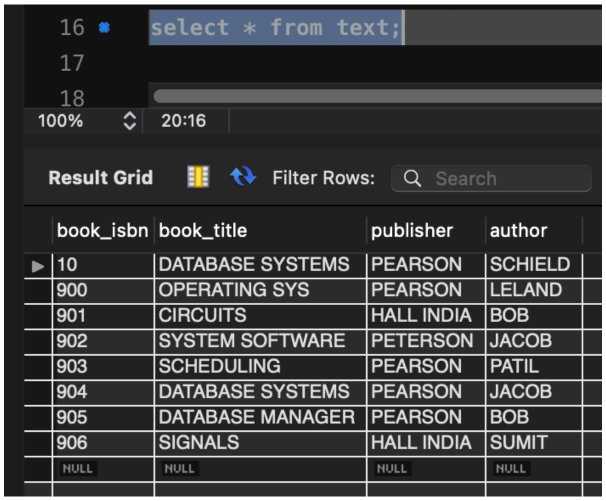
```
select * from student;
```

| regno | name | major | bdate |
|---|---|---|---|
| 1pe11cs001 | a | sr | 1993-12-30 |
| 1pe11cs002 | b | sr | 1993-09-24 |
| 1pe11cs003 | c | sr | 1993-11-27 |
| 1pe11cs004 | d | sr | 1993-04-13 |
| 1pe11cs005 | e | jr | 1994-08-24 |
| NULL | NULL | NULL | NULL |



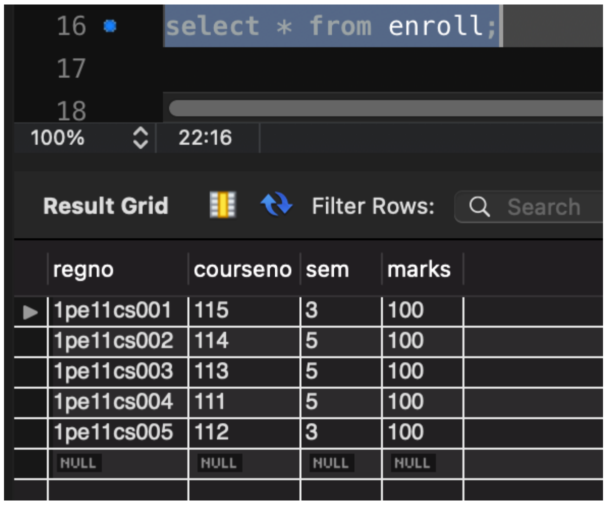
```
select * from course;
```

| courseno | cname | dept |
|---|---|---|
| 111 | OS | CSE |
| 112 | EC | CSE |
| 113 | SS | ISE |
| 114 | DBMS | CSE |
| 115 | SIGNALS | ECE |
| NULL | NULL | NULL |



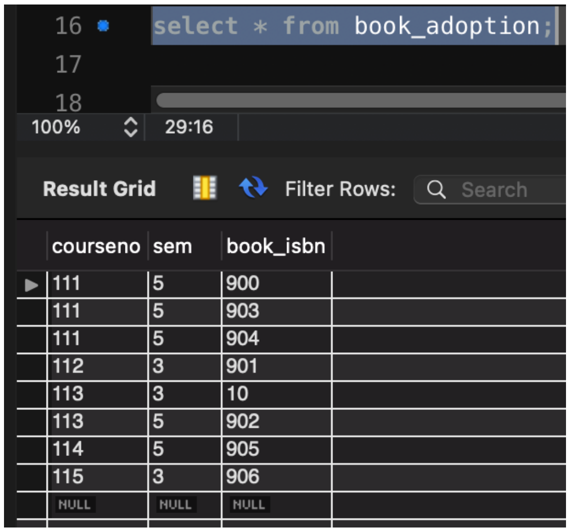
```
select * from text;
```

| book_isbn | book_title | publisher | author |
|---|---|---|---|
| 10 | DATABASE SYSTEMS | PEARSON | SCHIELD |
| 900 | OPERATING SYS | PEARSON | LELAND |
| 901 | CIRCUITS | HALL INDIA | BOB |
| 902 | SYSTEM SOFTWARE | PETERSON | JACOB |
| 903 | SCHEDULING | PEARSON | PATIL |
| 904 | DATABASE SYSTEMS | PEARSON | JACOB |
| 905 | DATABASE MANAGER | PEARSON | BOB |
| 906 | SIGNALS | HALL INDIA | SUMIT |
| NULL | NULL | NULL | NULL |



```
select * from enroll;
```

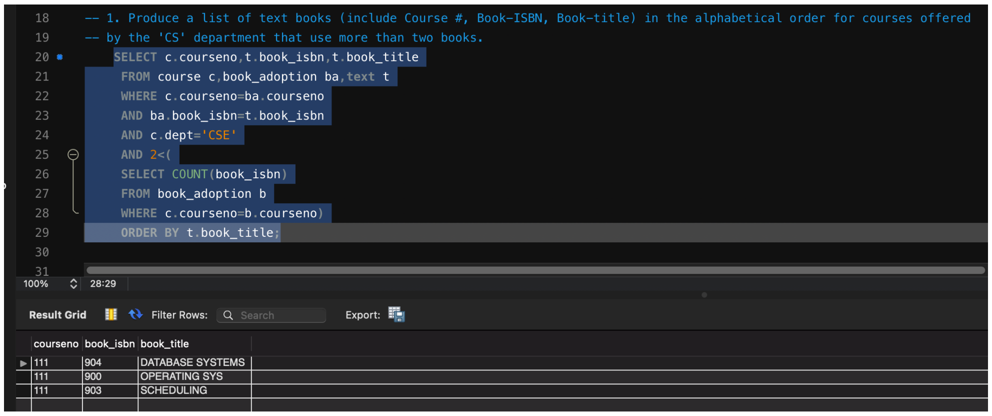
| regno | courseno | sem | marks |
|---|---|---|---|
| 1pe11cs001 | 115 | 3 | 100 |
| 1pe11cs002 | 114 | 5 | 100 |
| 1pe11cs003 | 113 | 5 | 100 |
| 1pe11cs004 | 111 | 5 | 100 |
| 1pe11cs005 | 112 | 3 | 100 |
| NULL | NULL | NULL | NULL |



```
select * from book_adoption;
```

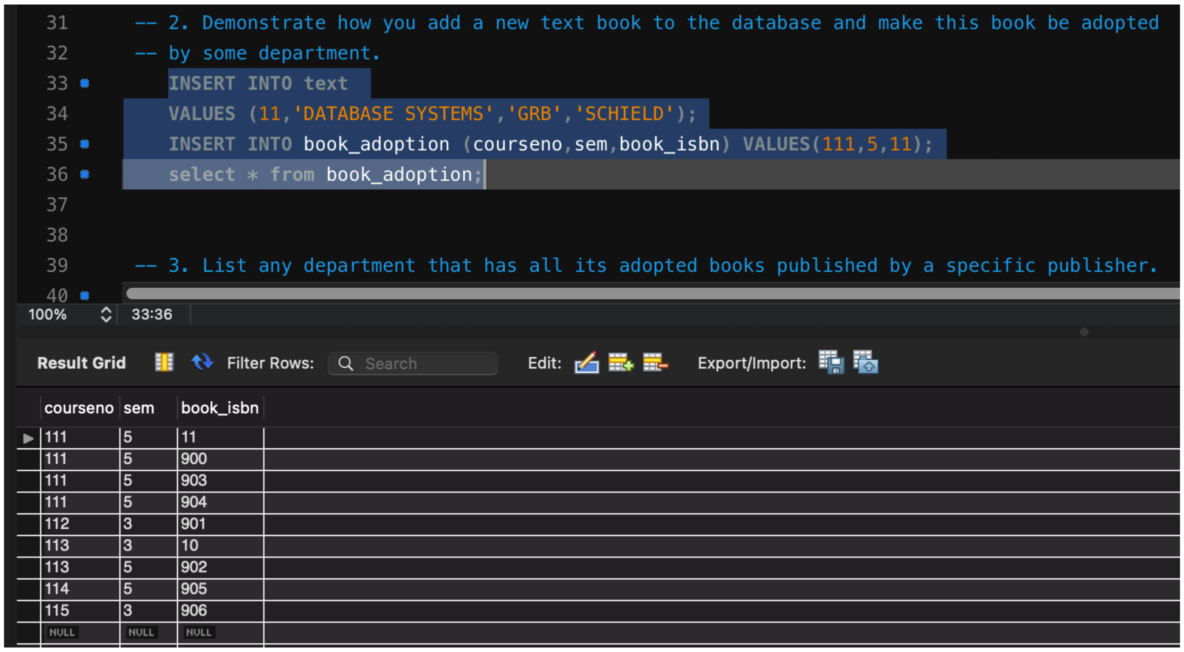
| courseno | sem | book_isbn |
|---|---|---|
| 111 | 5 | 900 |
| 111 | 5 | 903 |
| 111 | 5 | 904 |
| 112 | 3 | 901 |
| 113 | 3 | 10 |
| 113 | 5 | 902 |
| 114 | 5 | 905 |
| 115 | 3 | 906 |
| NULL | NULL | NULL |

QUERIES:



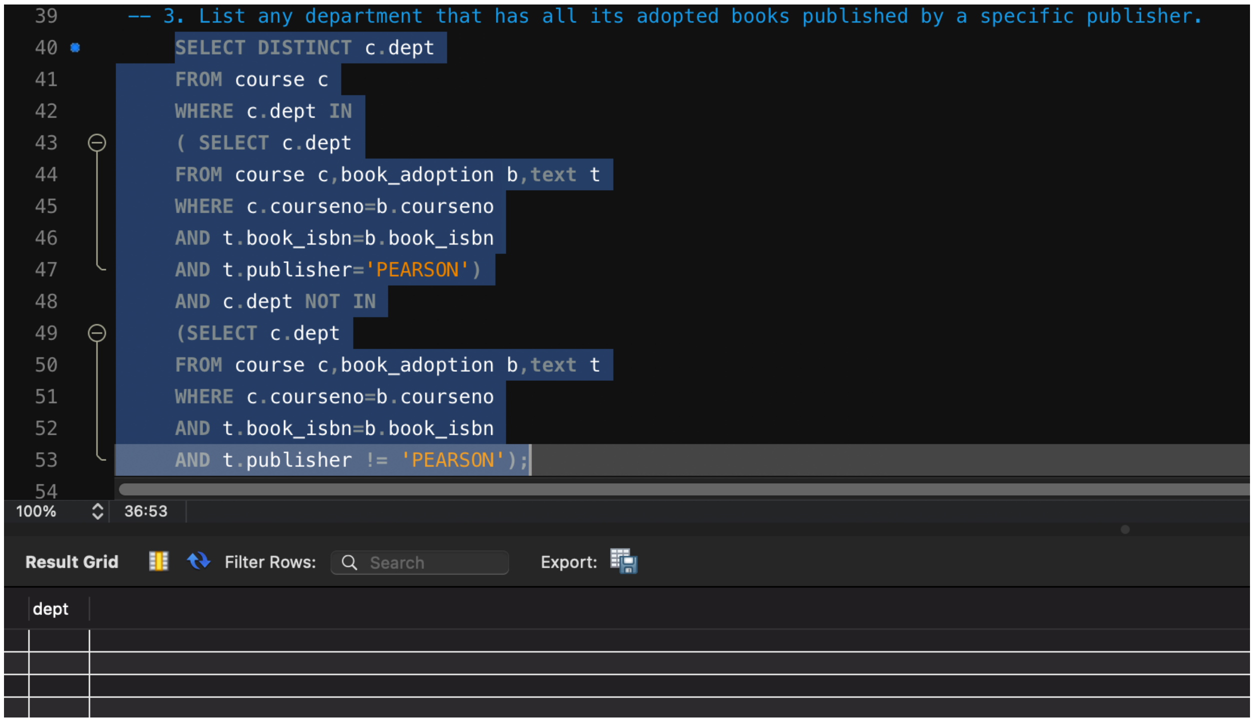
```
-- 1. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered
-- by the 'CS' department that use more than two books.
    SELECT c.courseno,t.book_isbn,t.book_title
     FROM course c,book_adoption ba,text t
     WHERE c.courseno=ba.courseno
     AND ba.book_isbn=t.book_isbn
     AND c.dept='CSE'
     AND 2<(
     SELECT COUNT(book_isbn)
     FROM book_adoption b
     WHERE c.courseno=b.courseno)
     ORDER BY t.book_title;
```

| courseno | book_isbn | book_title |
|---|---|---|
| 111 | 904 | DATABASE SYSTEMS |
| 111 | 900 | OPERATING SYS |
| 111 | 903 | SCHEDULING |



```
-- 2. Demonstrate how you add a new text book to the database and make this book be adopted
-- by some department.
    INSERT INTO text
    VALUES (11,'DATABASE SYSTEMS','GRB','SCHIELD');
    INSERT INTO book_adoption (courseno,sem,book_isbn) VALUES(111,5,11);
    select * from book_adoption;


-- 3. List any department that has all its adopted books published by a specific publisher.
```

```
courseno sem book_isbn
111      5   11
111      5   900
111      5   903
111      5   904
112      3   901
113      3   10
113      5   902
114      5   905
115      3   906
NULL     NULL NULL
```



```
-- 3. 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 t
    WHERE c.courseno=b.courseno
    AND t.book_isbn=b.book_isbn
    AND t.publisher='PEARSON')
    AND c.dept NOT IN
    (SELECT c.dept
    FROM course c,book_adoption b,text t
    WHERE c.courseno=b.courseno
    AND t.book_isbn=b.book_isbn
    AND t.publisher != 'PEARSON');
```

```
dept
```

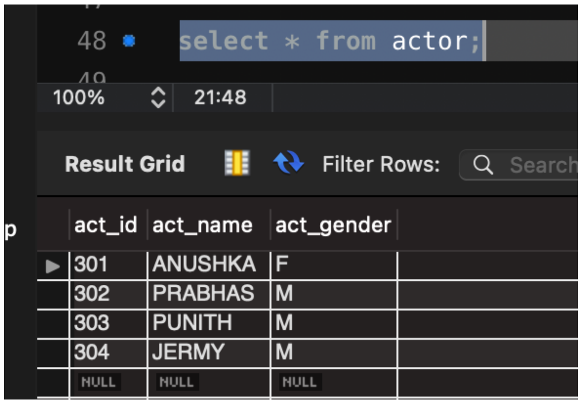
PROGRAM NO 9. MOVIE DATABASE

```
create database Movie;
use Movie;

create table actor (act_id integer,act_name varchar(30),act_gender varchar(2),primary
key(act_id));
create table director (dir_id integer,dir_name varchar(30),dir_phone decimal(10),primary
key(dir_id));
create table movie (mov_id integer, mov_Title varchar(30), mov_Year integer, mov_Lang
varchar(20), dir_id integer,primary key(mov_id),foreign key(dir_id) references director(dir_id));
create table movie_cast (act_id integer ,mov_id integer, role varchar(15),primary
key(act_id,mov_id),foreign key(act_id)references actor(act_id),foreign key(mov_id)references
movie(mov_id));
create table rating (mov_id integer, rev_Stars integer,primary key(mov_id),foreign
key(mov_id)references movie(mov_id));
show tables;

INSERT INTO ACTOR VALUES (301,'ANUSHKA','F');
INSERT INTO DIRECTOR VALUES (60,'RAJAMOULI', 8751611001);
INSERT INTO MOVIE VALUES (1001,'BAHUBALI-2', 2017, 'TELAGU', 60);
INSERT INTO MOVIE_CAST VALUES (301, 1002, 'HEROINE');
INSERT INTO RATING VALUES (1001, 4);
select * from rating;

/*1. List the titles of all movies directed by ‘Hitchcock’. */
select mov_title from movie where dir_id in
 (select dir_id from director where dir_name='Hitchcock');

/*2. Find the movie names where one or more actors acted in two or more movies */
select mov_title from movie m, movie_cast mv where m.mov_id=mv.mov_id and act_id
in( select act_id from movie_cast group by act_id having count( act_id)>1) group
by mov_title having count(*)>1;

/*3. List all actors who acted in a movie before 2000 and also in a movie after */
select act_name from actor a join movie_cast c on a.act_id=c.act_id join movie m
on c.mov_id=m.mov_id where m.mov_year not between 2000 and 2015;

/*4. 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 mov_title ,max(rev_stars) from movie inner join rating using(mov_id)
group by mov_title having max(rev_stars)>0 order by mov_title;

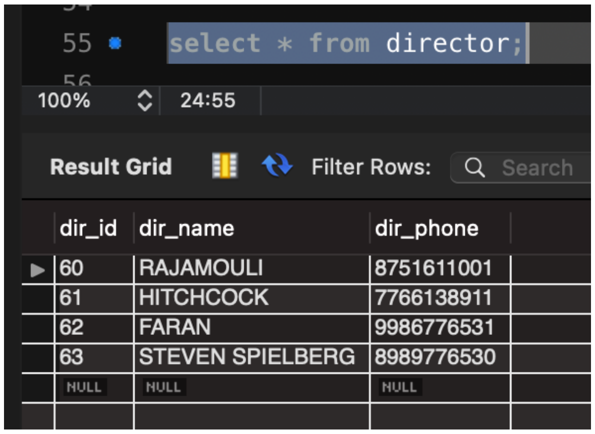
/*5. Update rating of all movies directed by 'Steven Spielberg' to 5. */
update rating
set rev_stars=5 where mov_id
in(select mov_id from movie where dir_id
in(select dir_id from director where dir_name='Steven Spielberg'));

OUTPUT:



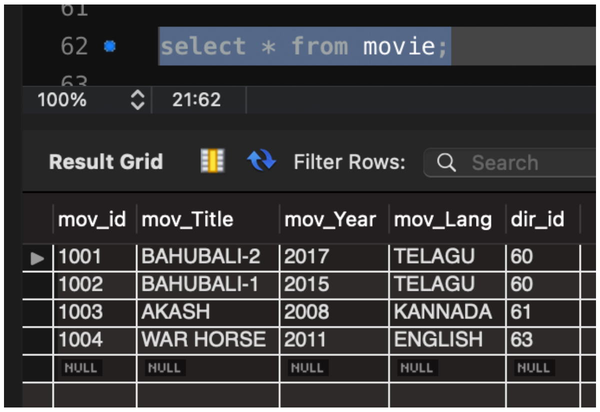
```
select * from actor;
```

| act_id | act_name | act_gender |
|---|---|---|
| 301 | ANUSHKA | F |
| 302 | PRABHAS | M |
| 303 | PUNITH | M |
| 304 | JERMY | M |
| NULL | NULL | NULL |



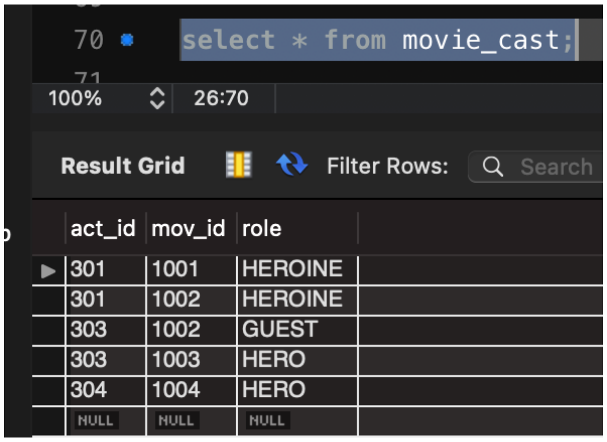
```
select * from director;
```

| dir_id | dir_name | dir_phone |
|---|---|---|
| 60 | RAJAMOULI | 8751611001 |
| 61 | HITCHCOCK | 7766138911 |
| 62 | FARAN | 9986776531 |
| 63 | STEVEN SPIELBERG | 8989776530 |
| NULL | NULL | NULL |



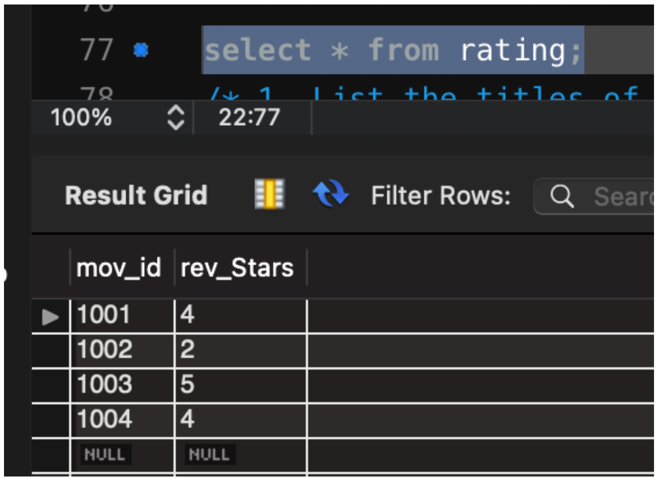
```
select * from movie;
```

| mov_id | mov_Title | mov_Year | mov_Lang | dir_id |
|---|---|---|---|---|
| 1001 | BAHUBALI-2 | 2017 | TELAGU | 60 |
| 1002 | BAHUBALI-1 | 2015 | TELAGU | 60 |
| 1003 | AKASH | 2008 | KANNADA | 61 |
| 1004 | WAR HORSE | 2011 | ENGLISH | 63 |
| NULL | NULL | NULL | NULL | NULL |



```
select * from movie_cast;
```

| act_id | mov_id | role |
|---|---|---|
| 301 | 1001 | HEROINE |
| 301 | 1002 | HEROINE |
| 303 | 1002 | GUEST |
| 303 | 1003 | HERO |
| 304 | 1004 | HERO |
| NULL | NULL | NULL |

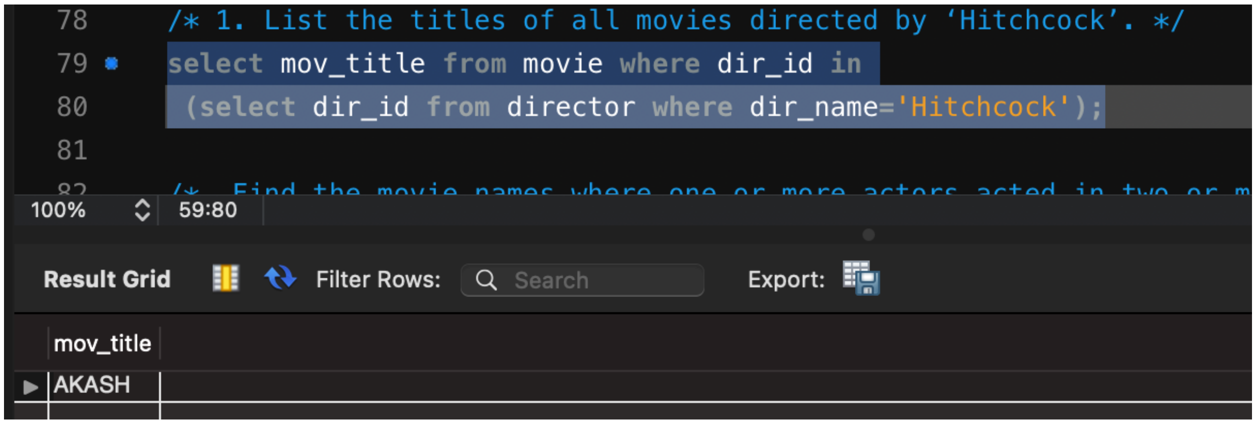


```
select * from rating;
```

```
100%    22:77
Result Grid    Filter Rows:
```

| mov_id | rev_Stars |
|---|---|
| 1001 | 4 |
| 1002 | 2 |
| 1003 | 5 |
| 1004 | 4 |
| NULL | NULL |

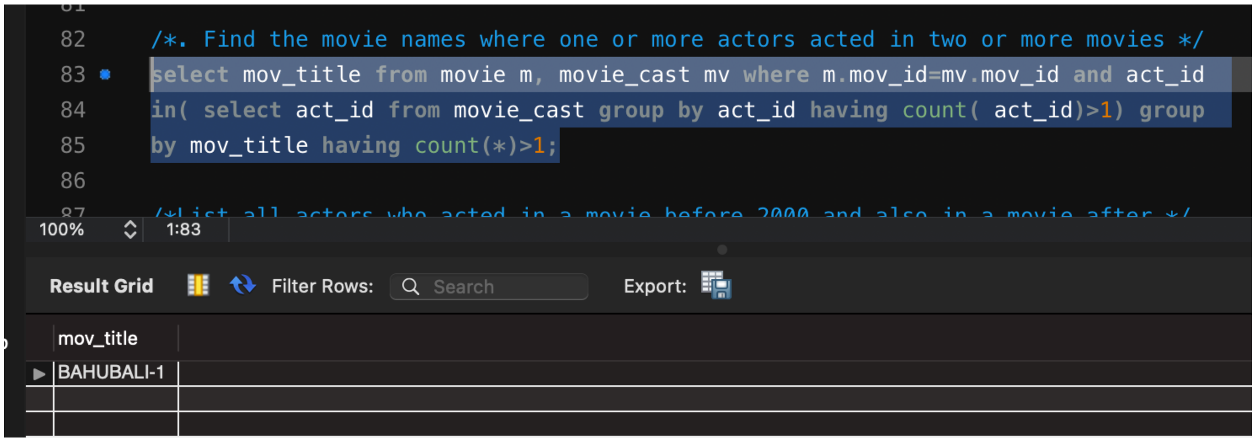
QUERIES:



```
/* 1. List the titles of all movies directed by 'Hitchcock'. */
select mov_title from movie where dir_id in
 (select dir_id from director where dir_name='Hitchcock');
```

```
100%    59:80
Result Grid    Filter Rows:  Search    Export:
```

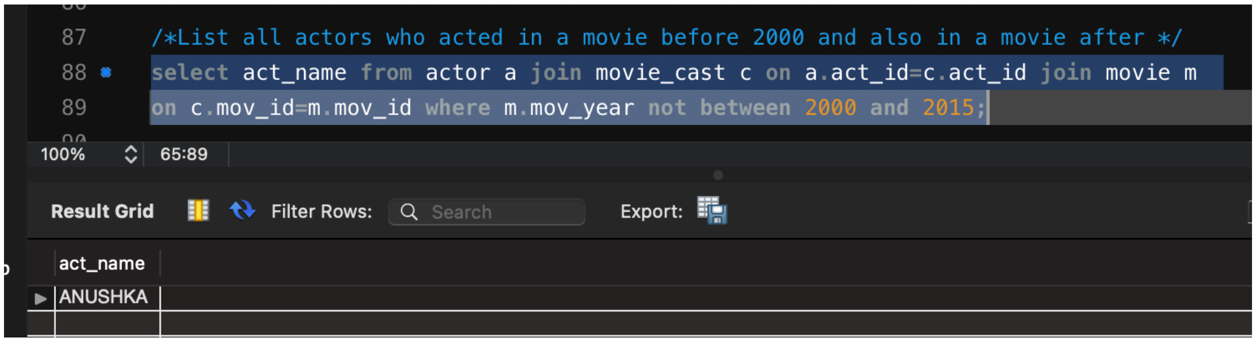
| mov_title |
|---|
| AKASH |



```
/*. Find the movie names where one or more actors acted in two or more movies */
select mov_title from movie m, movie_cast mv where m.mov_id=mv.mov_id and act_id
in( select act_id from movie_cast group by act_id having count( act_id)>1) group
by mov_title having count(*)>1;
```

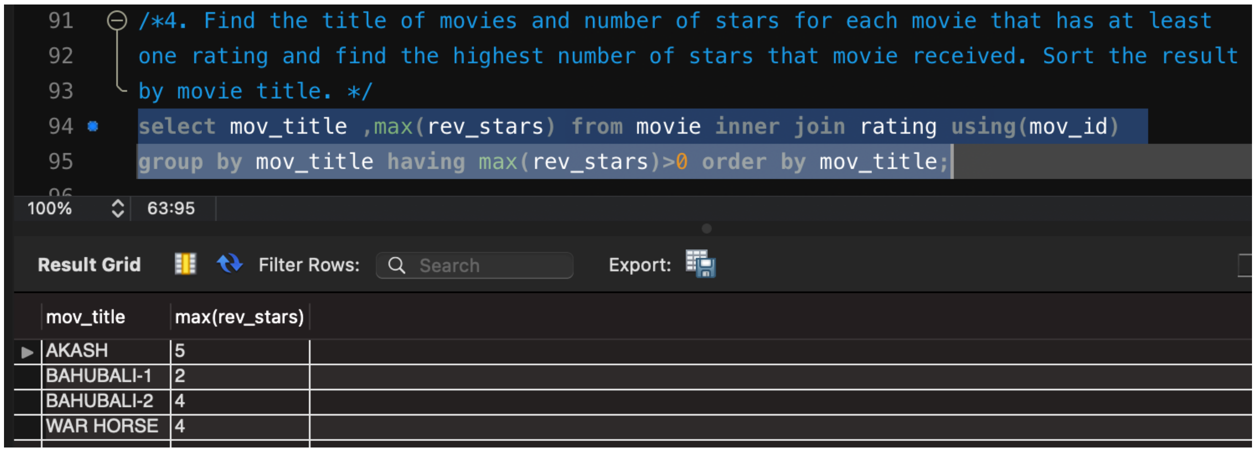
```
100%    1:83
Result Grid    Filter Rows:  Search    Export:
```

| mov_title |
|---|
| BAHUBALI-1 |



```
/*List all actors who acted in a movie before 2000 and also in a movie after */
select act_name from actor a join movie_cast c on a.act_id=c.act_id join movie m
on c.mov_id=m.mov_id where m.mov_year not between 2000 and 2015;
```

```
act_name
ANUSHKA
```



```
/*4. 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 mov_title ,max(rev_stars) from movie inner join rating using(mov_id)
group by mov_title having max(rev_stars)>0 order by mov_title;
```

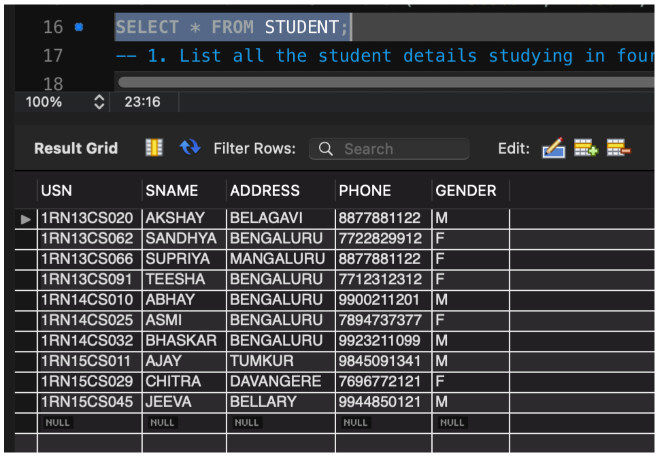
```
mov_title    max(rev_stars)
AKASH        5
BAHUBALI-1   2
BAHUBALI-2   4
WAR HORSE    4
```

## PROGRAM NO 10. COLLEGE DATABASE

```
CREATE DATABASE COLLEGE;
USE COLLEGE;
DROP DATABASE COLLEGE;
CREATE TABLE STUDENT (USN VARCHAR (10) PRIMARY KEY, SNAME VARCHAR (25), ADDRESS
VARCHAR (25), PHONE VARCHAR(15), GENDER VARCHAR (1));
CREATE TABLE SEMSEC (SSID VARCHAR (5) PRIMARY KEY, SEM INT, SEC VARCHAR (1));
CREATE TABLE CLASS (USN VARCHAR (10), SSID VARCHAR (5), PRIMARY KEY (USN, SSID),
FOREIGN KEY (USN) REFERENCES STUDENT (USN), FOREIGN KEY (SSID) REFERENCES SEMSEC
(SSID));
CREATE TABLE SUBJECT ( SUBCODE VARCHAR (8), TITLE VARCHAR (20), SEM INT, CREDITS INT,
PRIMARY KEY (SUBCODE));
CREATE TABLE IAMARKS ( USN VARCHAR (10), SUBCODE VARCHAR (8), SSID VARCHAR (5), TEST1
INT, TEST2 INT, TEST3 INT, FINALIA INT, PRIMARY KEY (USN, SUBCODE, SSID), FOREIGN KEY (USN)
REFERENCES STUDENT (USN), FOREIGN KEY (SUBCODE) REFERENCES SUBJECT (SUBCODE),
FOREIGN KEY (SSID) REFERENCES SEMSEC (SSID));
SHOW TABLES;

INSERT INTO STUDENT VALUES ('1RN13CS020','AKSHAY','BELAGAVI', 8877881122,'M');
INSERT INTO SEMSEC VALUES ('CSE8A', 8,'A');
INSERT INTO CLASS VALUES ('1RN13CS020','CSE8A');
INSERT INTO SUBJECT VALUES ('10CS81','ACA', 8, 4);
INSERT INTO IAMARKS VALUES ('1RN13CS091','10CS81','CSE8C', 15, 16, 18,16);
SELECT * FROM SUBJECT;
-- 1. List all the student details studying in fourth semester ‘C’ section.

SELECT S.*, SS.SEM, SS.SEC
FROM STUDENT S, SEMSEC SS, CLASS C WHERE S.USN = C.USN AND
SS.SSID = C.SSID AND
SS.SEM = 4 AND SS.SEc='C';

-- 2. Compute the total number of male and female students in each semester and in each
section

SELECT SS.SEM, SS.SEC, S.GENDER, COUNT(S.GENDER) AS COUNT FROM STUDENT S, SEMSEC SS,
CLASS C
WHERE S.USN = C.USN AND
SS.SSID = C.SSID
GROUP BY SS.SEM, SS.SEC, S.GENDER ORDER BY SEM;
```

```
-- 3. Create a view of Test1 marks of student USN ‘1BI15CS101’ in all subjects.

CREATE VIEW STU_TEST1_MARKS_VIEW AS
SELECT TEST1, SUBCODE
FROM IAMARKS
WHERE USN = '1RN13CS091';

select * from STU_TEST1_MARKS_VIEW ;

-- 4. Calculate the FinalIA (average of best two test marks) and update the corresponding table
for all students.
UPDATE IAMARKS
SET FINALIA = (TEST1 + TEST2 + TEST3) / 3;

SELECT * FROM IAMARKS;

-- 5. Categorize students based on the following criterion:
-- If FinalIA = 17 to 20 then CAT = ‘Outstanding’
-- If FinalIA = 12 to 16 then CAT = ‘Average’
-- If FinalIA< 12 then CAT = ‘Weak’
-- Give these details only for 8th semester A, B, and C section students.

SELECT S.USN,S.SNAME,S.ADDRESS,S.PHONE,S.GENDER, (CASE
WHEN IA.FINALIA BETWEEN 17 AND 20 THEN 'OUTSTANDING'
WHEN IA.FINALIA BETWEEN 12 AND 16 THEN 'AVERAGE'
ELSE 'WEAK' END) AS CAT
FROM STUDENT S, SEMSEC SS, IAMARKS IA, SUBJECT SUB WHERE S.USN = IA.USN AND
SS.SSID = IA.SSID AND
SUB.SUBCODE = IA.SUBCODE AND
SUB.SEM = 8;
```

OUTPUT:

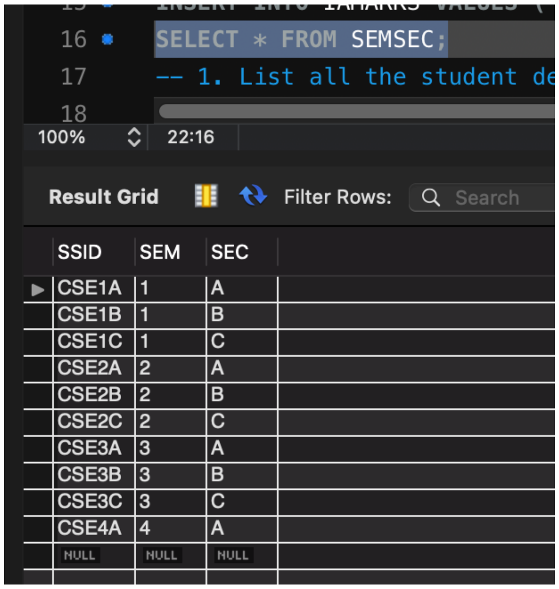


```
SELECT * FROM STUDENT;
-- 1. List all the student details studying in four
```

100% 23:16

Result Grid Filter Rows: Search Edit:

| USN | SNAME | ADDRESS | PHONE | GENDER |
|---|---|---|---|---|
| 1RN13CS020 | AKSHAY | BELAGAVI | 8877881122 | M |
| 1RN13CS062 | SANDHYA | BENGALURU | 7722829912 | F |
| 1RN13CS066 | SUPRIYA | MANGALURU | 8877881122 | F |
| 1RN13CS091 | TEESHA | BENGALURU | 7712312312 | F |
| 1RN14CS010 | ABHAY | BENGALURU | 9900211201 | M |
| 1RN14CS025 | ASMI | BENGALURU | 7894737377 | F |
| 1RN14CS032 | BHASKAR | BENGALURU | 9923211099 | M |
| 1RN15CS011 | AJAY | TUMKUR | 9845091341 | M |
| 1RN15CS029 | CHITRA | DAVANGERE | 7696772121 | F |
| 1RN15CS045 | JEEVA | BELLARY | 9944850121 | M |
| NULL | NULL | NULL | NULL | NULL |

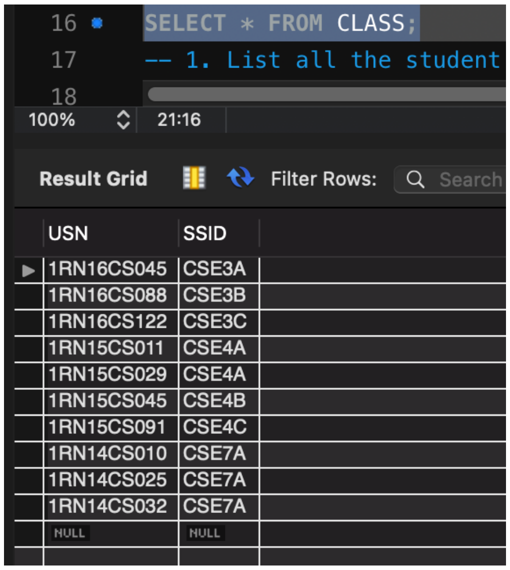


```
INSERT INTO IAMARKS VALUES (
SELECT * FROM SEMSEC;
-- 1. List all the student de
```

100% 22:16

Result Grid Filter Rows: Search

| SSID | SEM | SEC |
|---|---|---|
| CSE1A | 1 | A |
| CSE1B | 1 | B |
| CSE1C | 1 | C |
| CSE2A | 2 | A |
| CSE2B | 2 | B |
| CSE2C | 2 | C |
| CSE3A | 3 | A |
| CSE3B | 3 | B |
| CSE3C | 3 | C |
| CSE4A | 4 | A |
| NULL | NULL | NULL |

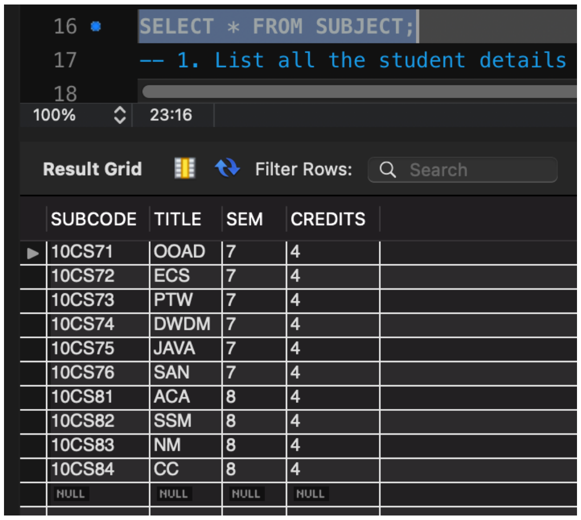


```
SELECT * FROM CLASS;
-- 1. List all the student
```

100% 21:16

Result Grid Filter Rows: Search

| USN | SSID |
|---|---|
| 1RN16CS045 | CSE3A |
| 1RN16CS088 | CSE3B |
| 1RN16CS122 | CSE3C |
| 1RN15CS011 | CSE4A |
| 1RN15CS029 | CSE4A |
| 1RN15CS045 | CSE4B |
| 1RN15CS091 | CSE4C |
| 1RN14CS010 | CSE7A |
| 1RN14CS025 | CSE7A |
| 1RN14CS032 | CSE7A |
| NULL | NULL |

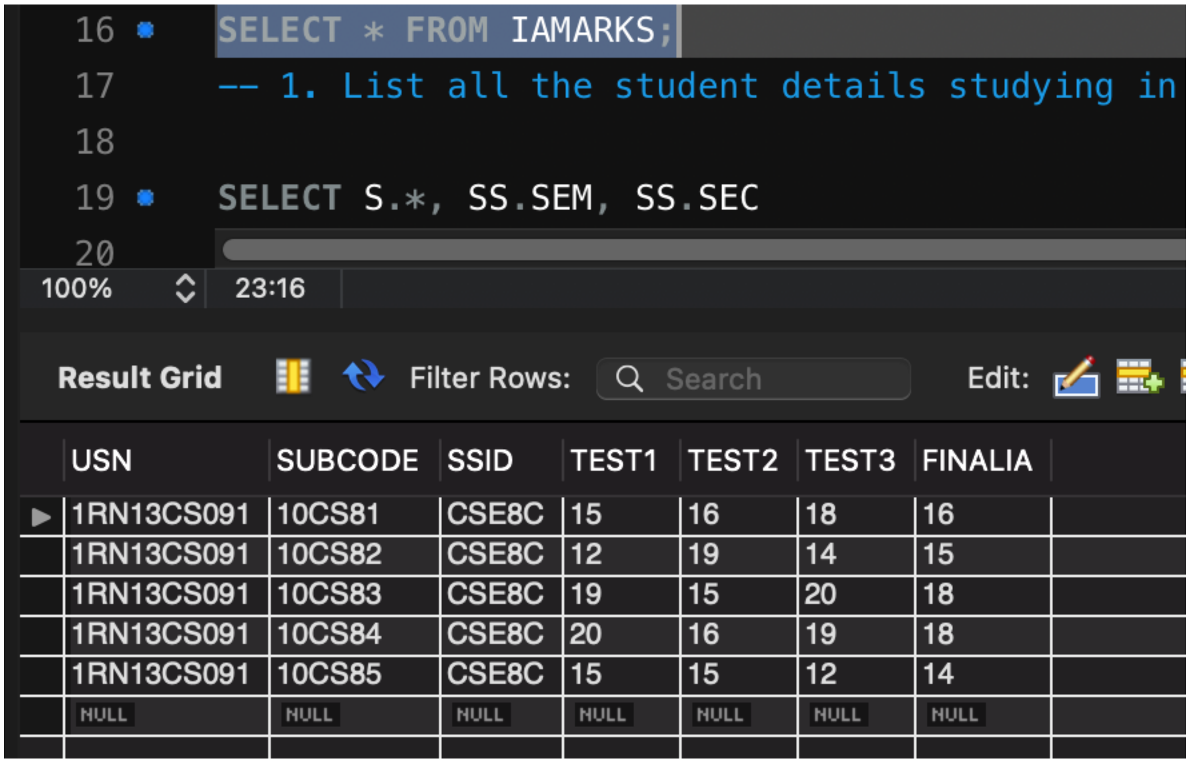


```
SELECT * FROM SUBJECT;
-- 1. List all the student details
```

100% 23:16

Result Grid Filter Rows: Search

| SUBCODE | TITLE | SEM | CREDITS |
|---|---|---|---|
| 10CS71 | OOAD | 7 | 4 |
| 10CS72 | ECS | 7 | 4 |
| 10CS73 | PTW | 7 | 4 |
| 10CS74 | DWDM | 7 | 4 |
| 10CS75 | JAVA | 7 | 4 |
| 10CS76 | SAN | 7 | 4 |
| 10CS81 | ACA | 8 | 4 |
| 10CS82 | SSM | 8 | 4 |
| 10CS83 | NM | 8 | 4 |
| 10CS84 | CC | 8 | 4 |
| NULL | NULL | NULL | NULL |



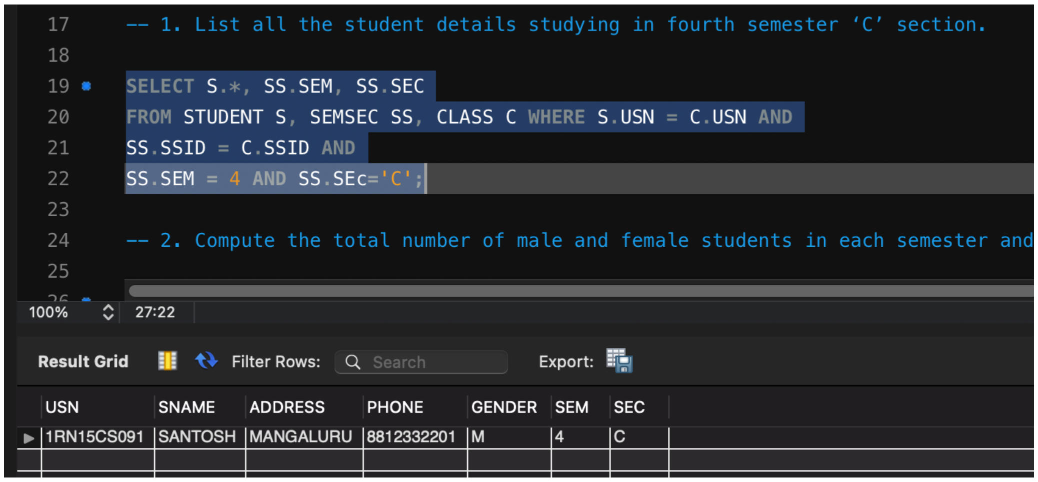
```
SELECT * FROM IAMARKS;
-- 1. List all the student details studying in

SELECT S.*, SS.SEM, SS.SEC
```

```
100%    23:16
```

Result Grid  Filter Rows: Search  Edit:

| | USN | SUBCODE | SSID | TEST1 | TEST2 | TEST3 | FINALIA |
|---|---|---|---|---|---|---|---|
| ▶ | 1RN13CS091 | 10CS81 | CSE8C | 15 | 16 | 18 | 16 |
| | 1RN13CS091 | 10CS82 | CSE8C | 12 | 19 | 14 | 15 |
| | 1RN13CS091 | 10CS83 | CSE8C | 19 | 15 | 20 | 18 |
| | 1RN13CS091 | 10CS84 | CSE8C | 20 | 16 | 19 | 18 |
| | 1RN13CS091 | 10CS85 | CSE8C | 15 | 15 | 12 | 14 |
| | NULL | NULL | NULL | NULL | NULL | NULL | NULL |

QUERIES:

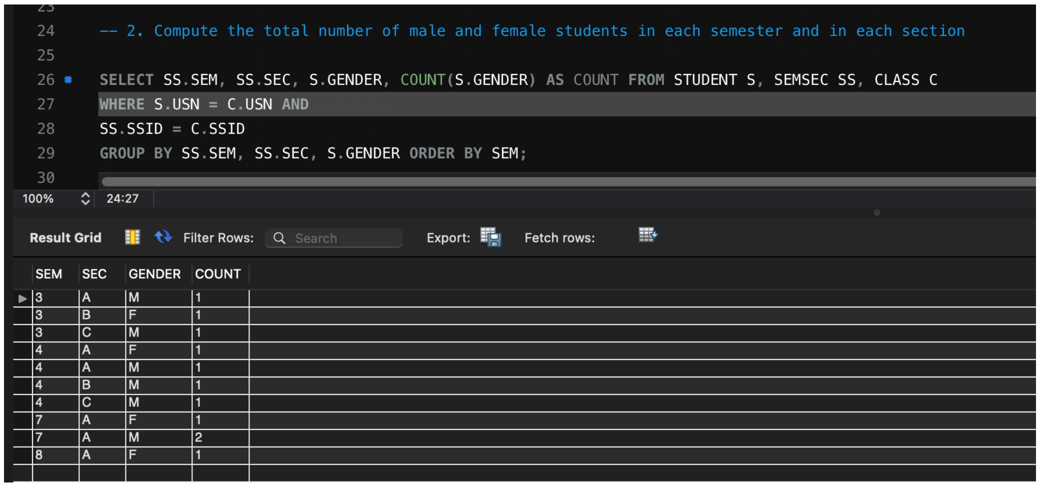


```
-- 1. List all the student details studying in fourth semester 'C' section.

SELECT S.*, SS.SEM, SS.SEC
FROM STUDENT S, SEMSEC SS, CLASS C WHERE S.USN = C.USN AND
SS.SSID = C.SSID AND
SS.SEM = 4 AND SS.SEc='C';

-- 2. Compute the total number of male and female students in each semester and
```

```
100%    27:22
```

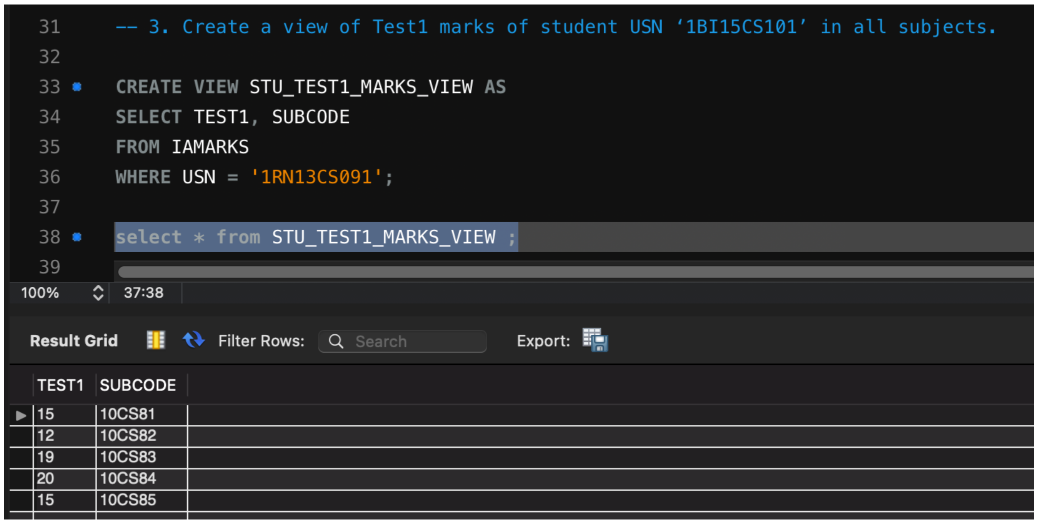
Result Grid  Filter Rows: Search  Export:

| | USN | SNAME | ADDRESS | PHONE | GENDER | SEM | SEC |
|---|---|---|---|---|---|---|---|
| ▶ | 1RN15CS091 | SANTOSH | MANGALURU | 8812332201 | M | 4 | C |



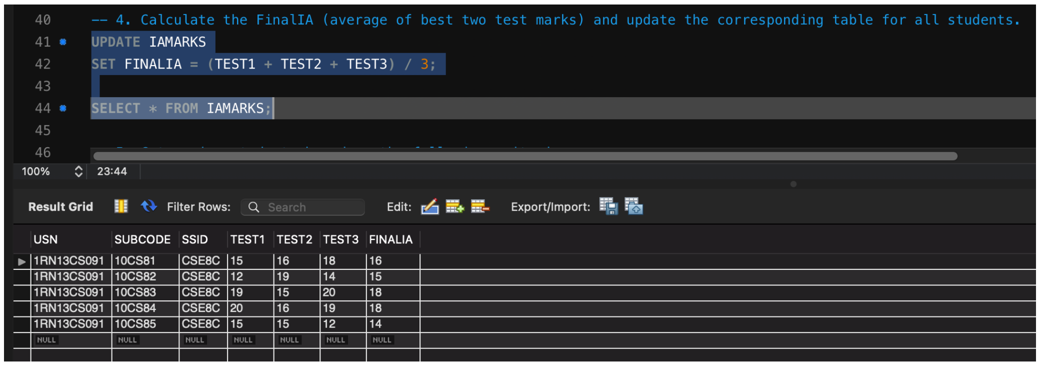
```
-- 2. Compute the total number of male and female students in each semester and in each section

SELECT SS.SEM, SS.SEC, S.GENDER, COUNT(S.GENDER) AS COUNT FROM STUDENT S, SEMSEC SS, CLASS C
WHERE S.USN = C.USN AND
SS.SSID = C.SSID
GROUP BY SS.SEM, SS.SEC, S.GENDER ORDER BY SEM;
```

```
SEM  SEC  GENDER  COUNT
3    A    M       1
3    B    F       1
3    C    M       1
4    A    F       1
4    A    M       1
4    B    M       1
4    C    M       1
7    A    F       1
7    A    M       2
8    A    F       1
```



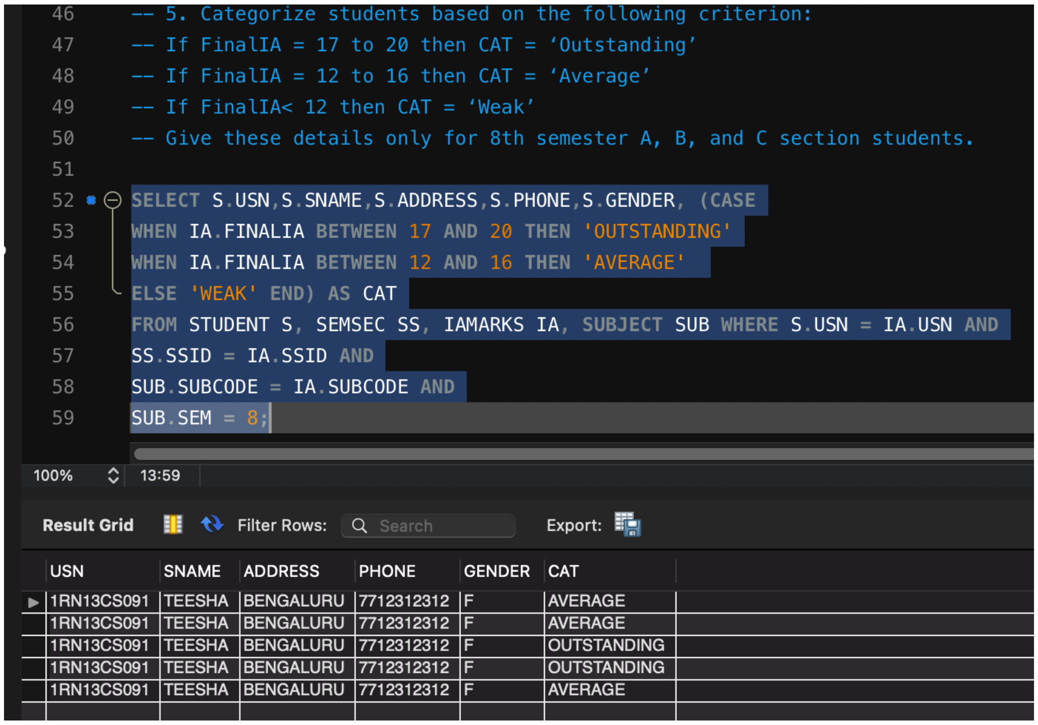
```
-- 3. Create a view of Test1 marks of student USN ‘1BI15CS101’ in all subjects.

CREATE VIEW STU_TEST1_MARKS_VIEW AS
SELECT TEST1, SUBCODE
FROM IAMARKS
WHERE USN = '1RN13CS091';

select * from STU_TEST1_MARKS_VIEW ;
```

```
TEST1  SUBCODE
15     10CS81
12     10CS82
19     10CS83
20     10CS84
15     10CS85
```



```
-- 4. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.
UPDATE IAMARKS
SET FINALIA = (TEST1 + TEST2 + TEST3) / 3;

SELECT * FROM IAMARKS;
```

```
USN         SUBCODE  SSID   TEST1  TEST2  TEST3  FINALIA
1RN13CS091  10CS81   CSE8C  15     16     18     16
1RN13CS091  10CS82   CSE8C  12     19     14     15
1RN13CS091  10CS83   CSE8C  19     15     20     18
1RN13CS091  10CS84   CSE8C  20     16     19     18
1RN13CS091  10CS85   CSE8C  15     15     12     14
NULL        NULL     NULL   NULL   NULL   NULL   NULL
```



```
-- 5. Categorize students based on the following criterion:
-- If FinalIA = 17 to 20 then CAT = ‘Outstanding’
-- If FinalIA = 12 to 16 then CAT = ‘Average’
-- If FinalIA< 12 then CAT = ‘Weak’
-- Give these details only for 8th semester A, B, and C section students.

SELECT S.USN,S.SNAME,S.ADDRESS,S.PHONE,S.GENDER, (CASE
WHEN IA.FINALIA BETWEEN 17 AND 20 THEN 'OUTSTANDING'
WHEN IA.FINALIA BETWEEN 12 AND 16 THEN 'AVERAGE'
ELSE 'WEAK' END) AS CAT
FROM STUDENT S, SEMSEC SS, IAMARKS IA, SUBJECT SUB WHERE S.USN = IA.USN AND
SS.SSID = IA.SSID AND
SUB.SUBCODE = IA.SUBCODE AND
SUB.SEM = 8;
```

100% 13:59

Result Grid Filter Rows: Search Export:

| USN | SNAME | ADDRESS | PHONE | GENDER | CAT |
|---|---|---|---|---|---|
| 1RN13CS091 | TEESHA | BENGALURU | 7712312312 | F | AVERAGE |
| 1RN13CS091 | TEESHA | BENGALURU | 7712312312 | F | AVERAGE |
| 1RN13CS091 | TEESHA | BENGALURU | 7712312312 | F | OUTSTANDING |
| 1RN13CS091 | TEESHA | BENGALURU | 7712312312 | F | OUTSTANDING |
| 1RN13CS091 | TEESHA | BENGALURU | 7712312312 | F | AVERAGE |