**BRANCH(branch-name:string, branch-city:string, assets:real)**

**ACCOUNT(accno:int, branch-name:string, balance:real)**

**DEPOSITOR(customer-name:string, accno:int)**

**CUSTOMER(customer-name:string, customer-street:string, customer-city:string)**

**LOAN(loan-number:int, branch-name:string, amount:real)**

**BORROWER(customer-name:string, loan-number:int)**

1. Find the loan number of those loans with loan amount between 50000 and 100000.

select loanno from loan where amount between 100000 and 500000;

2.For all customers, who have a loan from the bank find their names, loan numbers & loan amount.

select c.customername,l.loanno,l.amount from borrower c,loan l

where l.loanno=c.loanno;

3.Find the customer names, loan numbers loan amount for all loans at Brooklyn Branch.

select b.customername,l.loanno,l.amount from borrower b,loan l

where l.branchname='mgr' and l.loanno=b.loanno;

4.Find the names of all branches that have assets greater than at least one branch located in Frankfurt.

select branchname from branch where assets > any (select assets

from branch where branchcity='hyderabad');

5.List in alphabetical order all customers who have a loan at Perryridge branch.

select b.customername from borrower b,loan l where l.branchname='mgr' and l.loanno=b.loanno order by b.customername;

6.Find the names of all customers who street address includes the substring “Main”.

select customername from customer1 where customerstreet like '%road%';

7.List the loan numbers in descending order of amount

select loanno,amount from loan order by amount desc;

8.Find all customers having a loan, an account, or both at the bank(union, & union all).

(select d.customername from depositor d) union (select customername from borrower);

9.To find all customers who have both a loan and an account at the bank.

select unique borrower.customername from borrower,depositor where borrower.customername=depositor.customername;

10.To find all customers who have an account but no loan at the bank.

select customername from depositor where customername not in(select customername

from borrower);

11.Find the average account balance at the SOMAJIGUDA branch.

SELECT AVG(BALANCE) FROM ACCOUNT WHERE BRANCHNAME='SOMAJIGUDA';

12.Find the average account balance at each branch

SELECT AVG(BALANCE),BRANCHNAME FROM ACCOUNT GROUP BY BRANCHNAME;

13.Find the number of depositor for each branch.

SELECT A.BRANCHNAME,COUNT(\*) FROM ACCOUNT A,DEPOSITOR D

WHERE A.ACCNO=D.ACCNO GROUP BY A.BRANCHNAME;

14.Find the branch where the average account balance is more than Rs.13000 (group by, having)

SELECT BRANCHNAME FROM ACCOUNT HAVING AVG(BALANCE)>13000

GROUP BY BRANCHNAME;

15.Find the average balance for each customer who lives in Horriron & has at least 3 account.

SELECT AVG(A.BALANCE) FROM CUSTOMER1 C,ACCOUNT A,DEPOSITOR D

WHERE C.CUSTOMERCITY='BANGALORE' AND A.ACCNO=D.ACCNO

HAVING COUNT(\*)>=3 GROUP BY D.CUSTOMERNAME;

**16. Find bank accounts with a balance under $700:**

SELECT account\_number, balance FROM account WHERE balance < 700;

**17. Find bank accounts with a balance under $700 with increasing order of bank balance:**

SELECT account\_number, balance FROM account WHERE balance < 700 ORDER BY balance;

“Retrieve a list of all bank branch details, ordered by branch city, with each city’s branches listed in reverse order of holdings.” SELECT \* FROM branch ORDER BY branch\_city ASC, assets DESC;

**18. “Find the average loan amount for each branch.**

” SELECT branch\_name, AVG(amount) AS avg\_amt FROM loan GROUP BY branch\_name;

**19. “Find all customers with more than one loan**.

” SELECT customer\_name, COUNT(\*) AS num\_loans FROM borrower GROUP BY customer\_name HAVING COUNT(\*) > 1;

20.“**Select all customer names from depositor relation, that also appear somewhere in borrower relation.”**

SELECT DISTINCT customer\_name FROM depositor WHERE customer\_name IN ( SELECT customer\_name FROM borrower)

**21.find the largest loan at each branch**

SELECT branch\_name, MAX(amount) FROM loan GROUP BY branch\_name

**22. “Find customers with an account but not a loan.**”

SELECT DISTINCT customer\_name FROM depositor d WHERE NOT EXISTS ( SELECT \* FROM borrower b WHERE b.customer\_name = d.customer\_name);

**23. “Find customers with an account but not a loan**.”

SELECT DISTINCT customer\_name FROM depositor d WHERE NOT EXISTS ( SELECT \* FROM borrower b WHERE b.customer\_name = d.customer\_name);

**24. “Find all branches with assets greater than at least one branch in Brooklyn.”**

SELECT branch\_name FROM branch WHERE assets > SOME ( SELECT assets FROM branch WHERE branch\_name='Brooklyn');

**25. “Find branches with assets greater than all branches in Brooklyn.”**

SELECT branch\_name FROM branch WHERE assets > ALL ( SELECT assets FROM branch WHERE branch\_name='Brooklyn');

**Or**

SELECT branch\_name FROM branch WHERE assets > (SELECT MAX(assets) FROM branch WHERE branch\_name='Brooklyn');

**26. “Find all cities with more than two customers living in the city.”**

SELECT customer\_city, COUNT(\*) AS num\_customers FROM customer GROUP BY customer\_city HAVING COUNT(\*) > 2;

**Or, can write:**

SELECT customer\_city, num\_customers FROM (SELECT customer\_city, COUNT(\*) FROM customer GROUP BY customer\_city) AS counts (customer\_city, num\_customers) WHERE num\_customers > 2;

**27. Add 2% interest to all bank account balances with a balance of $500 or less.**

UPDATE account SET balance = balance \* 1.02 WHERE balance <= 500;

28. Delete all account tuples at every branch located at BANGALORE

DELETE FROM ACCOUNT WHERE BRANCHNAME IN( SELECT BRANCHNAME

FROM BRANCH WHERE BRANCHCITY='BANGALORE');

29. Find customers who hare borrowers are account holder.

SELECT CUSTOMERNAME FROM BORROWER WHERE CUSTOMERNAME IN (SELECT CUSTOMERNAME FROM DEPOSITOR);

30. Find all customers who have an both an account & loan at the bank

(SELECT CUSTOMERNAME FROM DEPOSITOR) INTERSECT

(SELECT CUSTOMERNAME FROM BORROWER);

**BOOK DEALER DATABASE**

**AUTHOR (author-id:int, name:string, city:string, country:string)**

**PUBLISHER (publisher-id:int, name:string, city:string, country:string)**

**CATALOG (book-id:int, title:string,author-id:int, publisher-id:int, category-id:int, year:int, price:int)**

**CATEGORY (category-id:int, description:string)**

**ORDER-DETAILS (order-no:int, book-id:int, quantity:int)**

1. Get the titles of all the books that are not in the FICTION category

select c.title from catalog c,category ca

where ca.description!='fiction' and ca.categoryid=c.categoryid;

-----------------------------------------------------------------------------------------------------------------

1. Get the names of all books whose price is greater than the maximum of the category averages.

select title from catalog where price > (select max(avg(price)) from catalog

group by categoryid);

-----------------------------------------------------------------------------------------------------------------

1. Get the names of all books that are in the book table and for which an order is placed.

select c.title from catalog c,orderdetails o where c.bookid=o.bookid;

-----------------------------------------------------------------------------------------------------------------

1. Get the names of all books that are not in the order table.

select c.title from catalog c,orderdetails o where c.bookid not exists

(select bookid from orderdetails);

-----------------------------------------------------------------------------------------------------------------

1. Get the title, author name and publisher name where publisher name has an underscore.

select p.name,a.name,c.title from catalog c,author a,publisher p

where instr(p.name,'\_')>0 and c.authorid=a.authorid and p.publisherid=c.publisherid;

-----------------------------------------------------------------------------------------------------------------

1. Get all the book names where price is null.

select c.title from catalog c where c.price is null;

-----------------------------------------------------------------------------------------------------------------

1. Get the publisher, the average,maximum and minimum book prices of all the publisher other that “McGraw-Hill”

select p.name,max(c.price),avg(c.price),min(c.price) from catalog c,publisher p

where p.publisherid=c.publisherid and p.name!=' McGraw-Hill ' group by p.name;

-----------------------------------------------------------------------------------------------------------------

1. Get publisher, the average, maximum and minimum book prices of all publisher who have more that 2 books listed in catalog table.

select p.name,avg(c.price),max(c.price),min(c.price) from catalog c,publisher p

where p.publisherid=c.publisherid group by c.publisherid,p.name having count(\*)>2;

-----------------------------------------------------------------------------------------------------------------

1. Get all the books details whose price is greater than 300 in descending order of price.

select \* from catalog where price > 300 order by price desc.

-----------------------------------------------------------------------------------------------------------------

1. Get the titles of all the books in the CATALOG table whose price is greater than the average price.

select title from catalog where price > (select avg(price) from catalog);

-----------------------------------------------------------------------------------------------------------------

1. Get the names of all books for which an order has been placed.

select c.title from catalog c,orderdetails o where o.bookid=c.bookid;

-----------------------------------------------------------------------------------------------------------------

1. get the title and price of all the books whose price is less than the average price of the books.

select title,price from catalog where price < (select avg(price) from catalog);

-----------------------------------------------------------------------------------------------------------------

1. Get the names of all authors who have more than two books in the catalog.

select a.name,count(\*) from author a,catalog c where a.authorid=c.authorid

group by a.name having count(\*)>2;

-----------------------------------------------------------------------------------------------------------------

1. Get the title and price of all the books whose price is greater than the maximum of the category averages.

select c.title,c.price from catalog c where c.price > (select max(avg(price))

from catalog group by categoryid);

-----------------------------------------------------------------------------------------------------------------

1. Get the details of the authors whose books are being sold from the book house

select a.name from author a,catalog c,orderdetails o

where o.bookid=c.bookid and c.authorid=a.authorid;

-----------------------------------------------------------------------------------------------------------------

1. Get the title and price of all the books whose price is greater than the average price of the books in the “Business”category.

select c.title,c.price from catalog c where c.price > (select avg(c.price)

from catalog c,category co where co.description='business' and co.categoryid=c.categoryid);

-----------------------------------------------------------------------------------------------------------------

1. Get the details of the titles whose price is greater than the average price and whose year of publishing is greater than the average year of publishing.

select c.title from catalog c where c.price > (select avg(price) from catalog)

and c.year > (select avg(year) from catalog);

-----------------------------------------------------------------------------------------------------------------

1. Get the title,year and price of all the books in the ascending order of year of publishing for which an order is placed.

select c.title,c.year,c.price from catalog c,orderdetails o where o.bookid=c.bookid

order by c.year asc;

-----------------------------------------------------------------------------------------------------------------

1. Get the book details and category details of all books whose price is greater than 1000.

select c.title,c.categoryid from catalog c where c.price > 1000;

-----------------------------------------------------------------------------------------------------------------

1. Find out the titles that have the same price.

select distinct t1.title from catalog t1,catalog t2 where t1.price = t2.price and t1.title != t2.title;

-----------------------------------------------------------------------------------------------------------------

1. Get the details of all authors and publishers in India ordered by name.

(select name from author where country='india') union

(select name from publisher where country='india') order by name;

-----------------------------------------------------------------------------------------------------------------

1. Find the author of the book which has maximum sales.

select name, authored from author where authored in( select authored from catlog where bookid in

(select bookid from odetails where quantity in (select max(quantity) from odetails));

-----------------------------------------------------------------------------------------------------------------

23. Demonstrate how you increase the price of books published by a specific publisher by 10%.

Select \* from catlog; **(before updation)**

update catlog set price= price +(price \*0.10) where publisherid =’& pubid’;

**[OR]**

update catalog set price=price\*1.10 where publisherid=44;

**MOVIE DATABASE**

**list all the information of the actors who played a role in the movie 'Annie Hall'**

SELECT \*

FROM actor

WHERE act\_id IN(

SELECT act\_id

FROM movie\_cast

WHERE mov\_id IN (

SELECT mov\_id

FROM movie

WHERE mov\_title='Annie Hall'

));

**find the name of the director (first and last names) who directed a movie that casted a role for 'Eyes Wide Shut'.**

SELECT dir\_fname, dir\_lname

FROM director

WHERE dir\_id in (

SELECT dir\_id

FROM movie\_direction

WHERE mov\_id in(

SELECT mov\_id

FROM movie\_cast WHERE role = ANY (

SELECT role

FROM movie\_cast

WHERE mov\_id IN (

SELECT mov\_id

FROM movie

WHERE mov\_title='Eyes Wide Shut'))));

**list all the movies which released in the country other than UK.**

SELECT mov\_title, mov\_year, mov\_time,

mov\_dt\_rel AS Date\_of\_Release,

mov\_rel\_country AS Releasing\_Country

FROM movie

WHERE mov\_rel\_country<>'UK';

**find the movie title, year, date of release, director and actor for those movies which reviewer is unknown.**

SELECT mov\_title, mov\_year, mov\_dt\_rel, dir\_fname, dir\_lname,

act\_fname, act\_lname

FROM movie a, movie\_direction b, director c,

rating d, reviewer e, actor f, movie\_cast g

WHERE a.mov\_id=b.mov\_id

AND b.dir\_id=c.dir\_id

AND a.mov\_id=d.mov\_id

AND d.rev\_id=e.rev\_id

AND a.mov\_id=g.mov\_id

AND g.act\_id=f.act\_id

AND e.rev\_name IS NULL;

**find the titles of all movies directed by the director whose first and last name are Woddy Allen.**

SELECT mov\_title

FROM movie

WHERE mov\_id=(

SELECT mov\_id

FROM movie\_direction

WHERE dir\_id=(

SELECT dir\_id

FROM director

WHERE dir\_fname='Woody' AND dir\_lname='Allen'

));

**to find all the years which produced at least one movie and that received a rating of more than 3 stars.**

SELECT DISTINCT mov\_year

FROM movie

WHERE mov\_id IN (

SELECT mov\_id

FROM rating

WHERE rev\_stars>3)

ORDER BY mov\_year;

**find the titles of all movies that have no ratings.**

SELECT DISTINCT mov\_title

FROM movie

WHERE mov\_id IN (

SELECT mov\_id

FROM movie

WHERE mov\_id NOT IN (

SELECT mov\_id FROM Rating));

**find the names of all reviewers who have ratings with a NULL value**.

SELECT DISTINCT rev\_name

FROM reviewer

WHERE rev\_id IN (

SELECT rev\_id

FROM rating

WHERE rev\_stars IS NULL);

**Return the reviewer name, movie title, and stars for those movies which reviewed by a reviewer and must be rated. Sort the result by reviewer name, movie title, and number of stars**

SELECT rev\_name, mov\_title, rev\_stars

FROM reviewer, rating, movie

WHERE reviewer.rev\_id=rating.rev\_id

AND movie.mov\_id=rating.mov\_id

AND reviewer.rev\_name IS NOT NULL

AND rating.rev\_stars IS NOT NULL

ORDER BY rev\_name, mov\_title, rev\_stars;

**find the reviewer's name and the title of the movie for those reviewers who rated more than one movies.**

SELECT rev\_name, mov\_title, rev\_stars

FROM reviewer, rating, movie

WHERE reviewer.rev\_id=rating.rev\_id

AND movie.mov\_id=rating.mov\_id

AND reviewer.rev\_name IS NOT NULL

AND rating.rev\_stars IS NOT NULL

ORDER BY rev\_name, mov\_title, rev\_stars;

**find the movie title, and the highest number of stars that movie received and arranged the result according to the group of a movie and the movie title appear alphabetically in ascending order.**

SELECT mov\_title, MAX(rev\_stars)

FROM movie, rating

WHERE movie.mov\_id=rating.mov\_id

AND rating.rev\_stars IS NOT NULL

GROUP BY mov\_title

ORDER BY mov\_title;

**find the names of all reviewers who rated the movie American Beauty.**

SELECT DISTINCT reviewer.rev\_name

FROM reviewer, rating, movie

WHERE reviewer.rev\_id = rating.rev\_id

AND movie.mov\_id = rating.mov\_id

AND movie.mov\_title = 'American Beauty';

**find the titles of all movies which have been reviewed by anybody except by Paul** **Monks**

SELECT movie.mov\_title

FROM movie

WHERE movie.mov\_id IN(

SELECT mov\_id

FROM rating

WHERE rev\_id NOT IN (

SELECT rev\_id

FROM reviewer

WHERE rev\_name='Paul Monks'));

**Return the reviewer name, movie title, and number of stars for those movies which rating is the lowest one**

SELECT reviewer.rev\_name, movie.mov\_title, rating.rev\_stars

FROM reviewer, movie, rating

WHERE rating.rev\_stars = (

SELECT MIN(rating.rev\_stars)

FROM rating

)

AND rating.rev\_id = reviewer.rev\_id

AND rating.mov\_id = movie.mov\_id;

**find the titles of all movies directed by James Cameron.**

SELECT mov\_title

FROM movie

WHERE mov\_id IN (

SELECT mov\_id

FROM movie\_direction

WHERE dir\_id IN (

SELECT dir\_id

FROM director

WHERE dir\_fname = 'James' AND dir\_lname='Cameron'

));

**find the name of those movies where one or more actors acted in two or more movies.**

SELECT mov\_title

FROM movie

WHERE mov\_id IN (

SELECT mov\_id

FROM movie\_cast

WHERE act\_id IN (

SELECT act\_id

FROM actor

WHERE act\_id IN (

SELECT act\_id

FROM movie\_cast GROUP BY act\_id

HAVING COUNT(act\_id)>1)));