consider the example:

Branch (Branch_name , Branch_city , Assets)

customer (customer_id , customer_name, customer_street, customer_city)

Account (Account_number , Branch_name , ~~Amount~~ Balance)

Loan (~~Depositor~~ Loan_number , Branch_name , Amount) .

Depositor (customer_id , Account_number)

Borrower (customer_id , Loan_number)

Write SQL queries for the following :

1. Find all customers who have account but no loan in bank.

```
select customer_name
from customer
where customer_id in (select customer_id
from depositor) and
customer_id not in (select customer_id
from borrower );
```

2. Delete all loans between 10,000 and 25,000

```
Delete ~~amount~~
from loan
where amount between 10,000 and 25,00
```

3. Find the name of all customers who have a loan at Utara branch.

```
select customer name
from customer
where customer_id in(
```

```
select customer_id from borrower b, loan l
where b.loan-number = l.loan-number
and l.branch_name = 'uttara branch'
);
```

4 Find all customers who have both loan and account.

```
select customer-name
from customer-name
where customer-id in (
select customer-id from depositor) and
customer-id in (select customer-id from
borrower );
```

5. Delete all loans with an amount in the range of 0 to 500;

```
Delete from loan
where amount between 0 and 500;
```

6. Find the average account balance at each bran

```
select branch-name, avg (balance) as average-
balance from account
~~where~~ Group by branch.name;
```

7 Find the name of all branches in the loan relation.

```
select distinct branch-name
from loan;
```

8. Deduct 3% service charge from account balance for those who have both loan and account, otherwise deduct 5% charge:

```
update account
set balance = case
when account_number in (
select d.account-number
from depositor d, borrower b
where d.customer-id = b.customer-id)
then
balance - balance * 0.03
Else
balance - balance * 0.05
End;
```

9. Increase all balances by 5 percent.

```
update account
set balance = balance * 1.05
```

10. Find the number of depositor for each branch.

```
select branch_name, count (distinct
d.customer_id) as no-of-depositor
from depositord, account a
where d.account.number = a.account.number
Group by branch-name;
```

11. Find the largest account balance in the bank.

```
select max(balance) as largest_balance
from account;
```

12 Delete all ~~Sajid~~ Asif's accounts record.

```
Delete from account
where account_number in (
select account_number from
depositor d, customer c
where d.customer_id = c.customer_id
and customer_name = 'Asif' );
```

13. Find the second highest account balance in the bank.

```
select max (balance) as second_highest_balance
from account
where Balance < (select max (balance) from
                 account) ;
```

or,

```
select branch_name, balance as second_highest_balance
from (
select branch_name, balance,
Dense_rank() over (order by balance Desc) as
rank from account ) as ranked_balances
where rank = 2;
```

14. Find all customers who live in the same city as the brance where they have an account but no loan

```
select c.customer-name as customer
from customer c
where customer_id in (
select customer-id from
depositord, account a
where ~~d.cc~~ d.account_number = a.account_
number) and
customer_id not in (select customer-id
from borrower);
```

15. Find branches that have no accounts.

```
select branch-name
from branch
where branch_name not in (
select distinct branch_name from account);
```

16. Find branches where total loan ~~number~~ amount is greater than 50,000

```
select branch_name
from loan Group by branch_name
~~where sum~~
Having sum(amount) > 50000;
```

17. Retrieve distinct customers who have either an account or an loan.

```
select distinct customer_name
from customer
where customer_id in (select customer_id
from depositor) or
customer_id in (select customer_id from
borrower);
```

18. Retrieve the branch_name with the maximum number of accounts

```
select branch_name from account
Group by branch_name
order by count (account_number) Desc
limit 1;
```

19. Find the maximum amount issued in each branch

```
select branch_name, max (amount) as max_amount
from loan
Group by branch_name;
```

20. Find customers whose names start with 'A':

```
select customer_name
from customer
where customer_name like 'A%';
```

consider the following

employee (employee_id, employee name, street, city)

works (employee_id, company_name, salary)

company (company_name, city)

manages (employee id, manager id)

write sql queries.

1. Find the company name that has the most employees.

```
select company_name
from works group by company_name
order by count(employee id) desc
limit 1;
```

2. Find the average salaries of each company.

```
select company_name, avg(salary) as
average_salary from works
group by company_name;
```

3. Find all employees who live in city, but their company is not in city

```
select e.employee_name
from employee e
join works w on e.employee_id = w.employee id
join company c on w.company_name = c.company_name
where e.employee.city = 'Barishal' and
  e.company_city != 'Barishal';
```

4. Find the name of all employees who work for first bank corporation.

```
select e.employee_name
from employee e
join works w on e.employee_id = w.employee_id
where w.company_name = 'First bank Co.';
```

5. Find the name, cities, salaries of all employees who work IFIC.

```
select e.employee_name, e.employee_city,
w.salary
from employee e
join works w on e.employee_id = w.employee_id
where w.company_name = 'first bank Co.';
```

6. Find the names of employees who donot work for first bank Co.

```
select e.employee_name
from employee e
join works w on e.employee_id = w.employee_id
where w.company_name != 'first bank Co';
```

7. Find total salaries of each company.

```
select company_name, sum(salary) as
total_salary from works
Group by company_name;
```

10. Find employee_name, company_name, and employee_city of the employee who earns the second highest salary.

```
select employee_name, company_name, employee_city
from employee e
join works w on e.employee_id = w.employee_id
where w.salary = (select max(salary) from
                  works where
                  salary < (select max(salary)
                  from works));
```

11. ৭ নং এর মত 12. ৯ নং এর মত

13. Find companies with more than 5 employees.

```
select company_name
From works Group by company_name
Having count(employee_id) > 5;
```

14. same as 8

15. Delete all records where an employee has no manager.

```
Delete from manages
where manager_id is null;
```

16. Assign a new manager to employees who currently have no manager.

```
update manages
set manager_id = (select managerid from manag
                  where manager_id is not null
                  limit 1)
where manager_id is null;
```

8. Give all employees of first bank co. a 20% salary raise

```
update works
set salary = salary * 1.2
where company_name = 'First bank co'
```

9. Find the second highest salary.

```
Select max(salary) as second_highest_salary
from works
where salary < (select max(salary) from
                works);
```

or

```
select employee_name, salary as second_highest
From (
select employee_name, salary
Dense_rank() over (order by salary desc)
as rank from works) as ranked_salary
where rank = 2
```