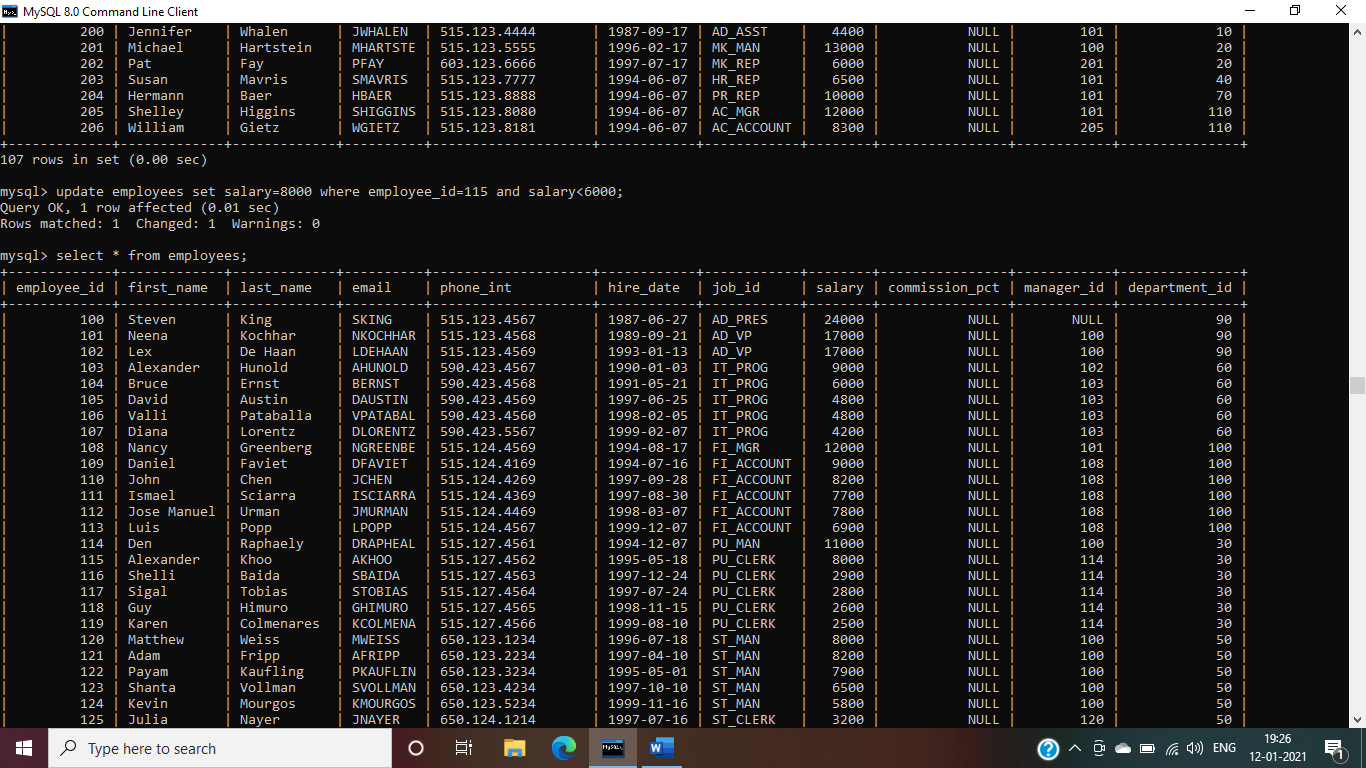
12/01/2020

**DML Operation**

1. Change salary of employee 115 to 8000 if the existing salary is less than 6000.

Ans-update employees set salary=8000 where employee\_id=115 and salary<6000;



2. Insert a new employee into employees with all the required details.

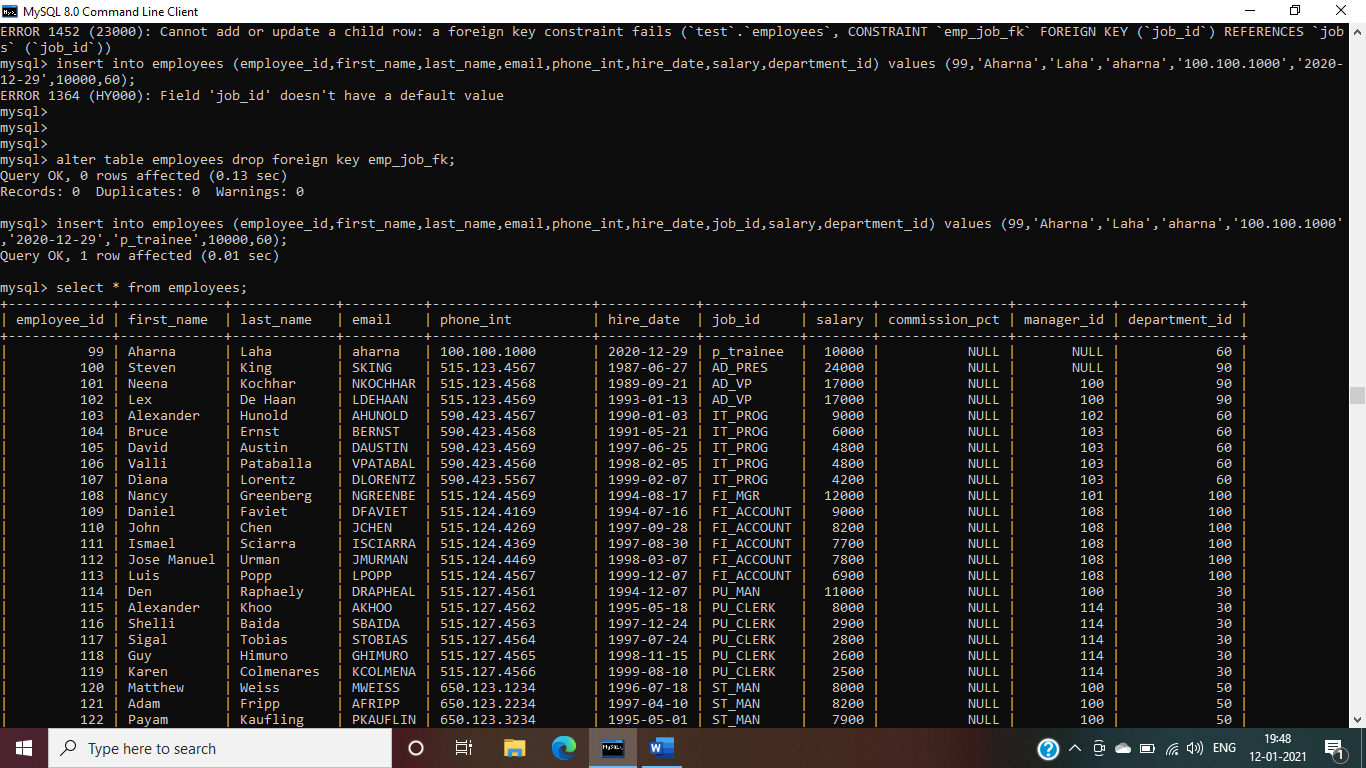
Ans-

Drop foreign key-

alter table employees drop foreign key emp\_job\_fk;

Insert values-

insert into employees (employee\_id,first\_name,last\_name,email,phone\_int,hire\_date,job\_id,salary,department\_id) values (99,'Aharna','Laha','aharna','100.100.1000','2020-12-29','p\_trainee',10000,60);



3. Delete department 20.

Ans-delete from departments where department\_id=20;

4. Change job ID of employee 110 to IT\_PROG if the employee belongs to department 10 and the existing job ID does not start with IT.

Ans-update employees set job\_id= ‘IT\_PROG’ where employee\_id=110 and department\_id=10 and not job\_id like ’IT%’;

5. Insert a row into departments table with manager ID 120 and location ID in any location ID for city Tokyo.

Ans-insert into departments(99,’Games’,120,1200);

6. Display job title, employee ID, number of days between ending date and starting date for all jobs in department 30 from job history.

Ans-select employee\_id,job\_title,end\_date-start\_date as days from job\_history natural join jobs where department\_id=30;

**DDL Assignments with Constraints**

Table ---> Customer

custId, firstName,lastName,age,city, mobileNumber, dob

Add the Constraints

custId is Primary Key

firstName not null

age must be greater than 21

mobile must be unique

Ans-create table customer (custid int primary key,firstname varchar(10) not null,age int check(age>21),city varchar(10),mobile\_number int unique,dob varchar(10));

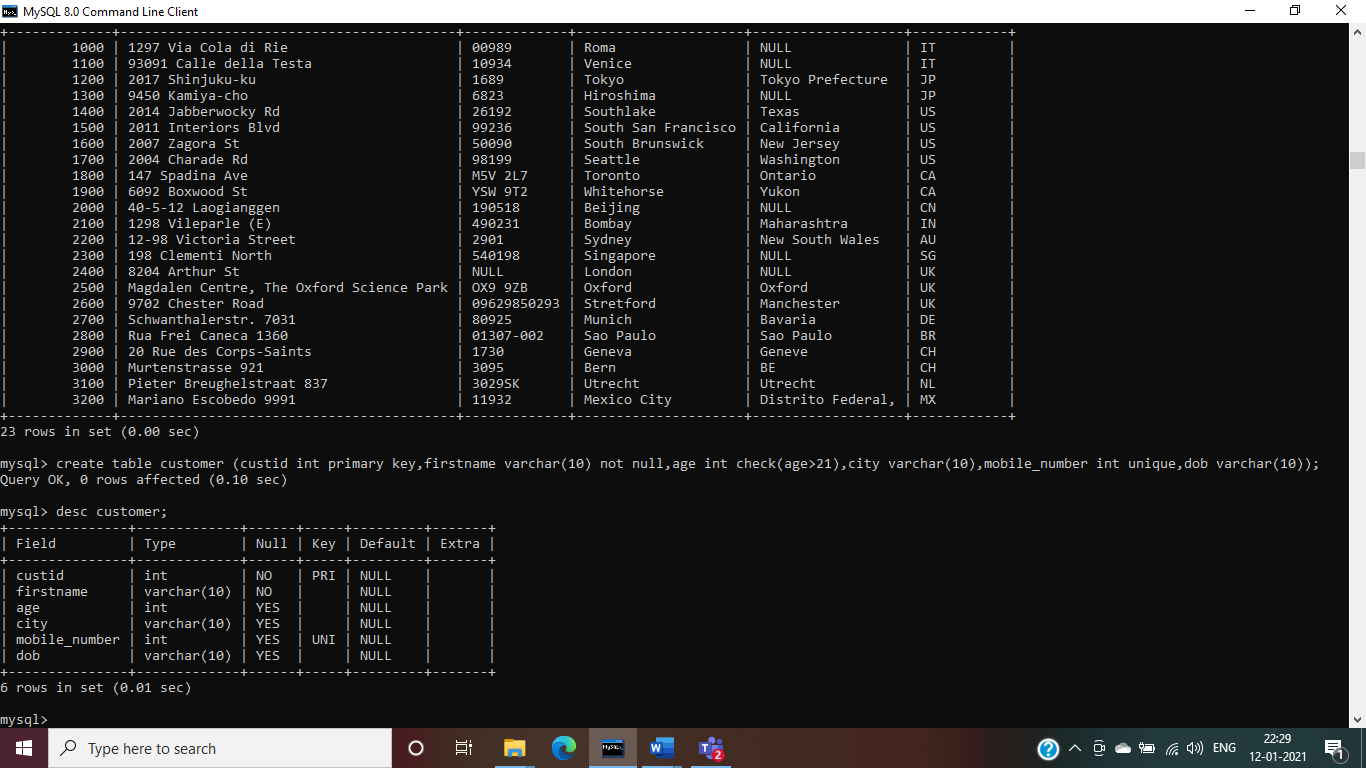


Table ----> Branch

branchId, branchName, city

Add the Constraints

branchId is Primary Key

branchName not null

city not null

Ans-create table branch (branchid int primary key,branch\_name varchar(15) not null,city varchar(20) not null);

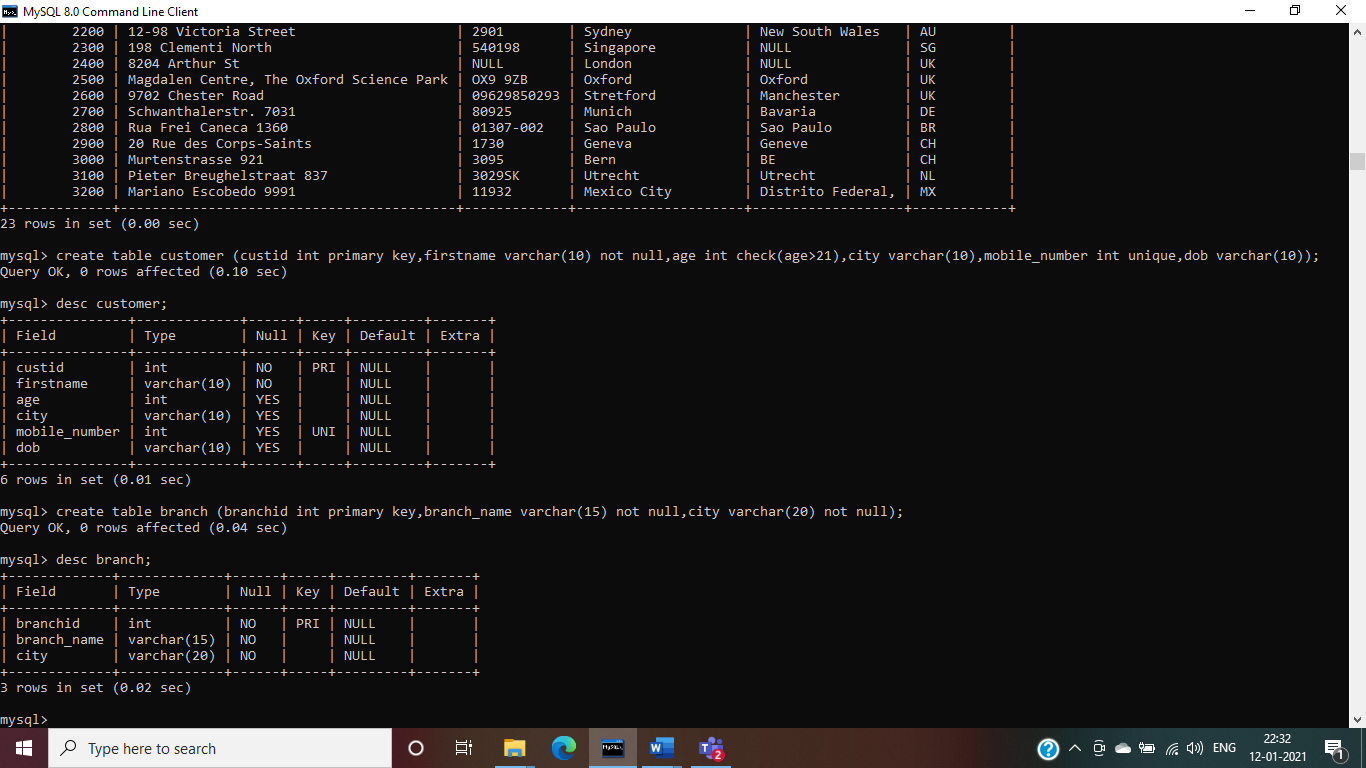


Table -----> Account

accountNumber, openingBalance, typeOfAccount, status,BankId,CustId

Add the Constraints

accountNumber is primary key

openingBalance must be greater than 5000

typeOfAccount must be saving/current

BankId is foreign key refer to BranchId(Primary key) Branch table

CustId is foreign key refer to Customer(Primary key) Customer table

Ans-

create table account (account\_number int primary key,opening\_balance int check(opening\_balance>5000),type\_of\_account enum('savings','current'),branchid int,custid int);

alter table account add constraint b\_fk foreign key(branchid) references branch(branchid);

alter table account add constraint c\_fk foreign key(custid) references customer(custid);

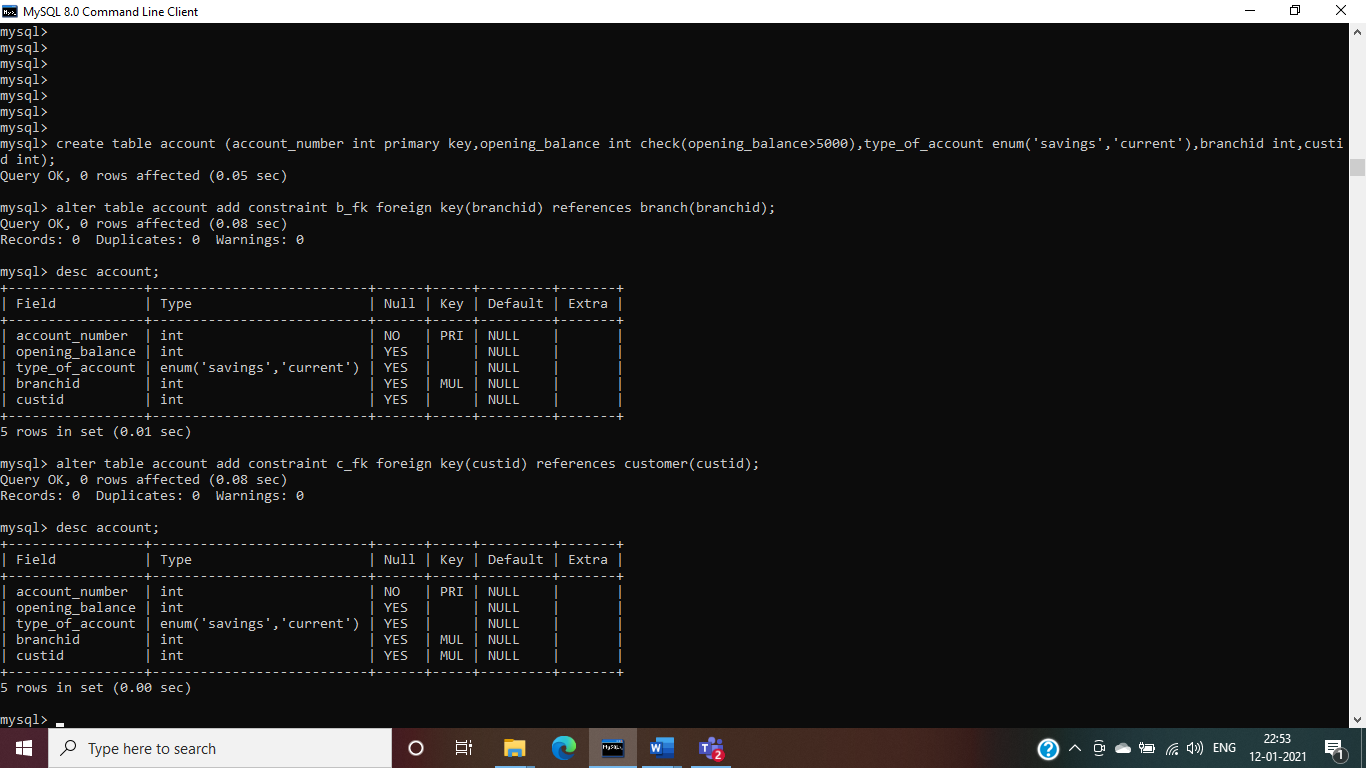


Table ----> Transaction

transactionId, transactionDate, MediumOfTransaction, TransactionAmount

Add the Constraints

transactionId is primary key

Ans-create table transaction (transaction\_id int primary key,transaction\_date varchar(10),medium\_of\_transaction varchar(10),transaction\_amount float);

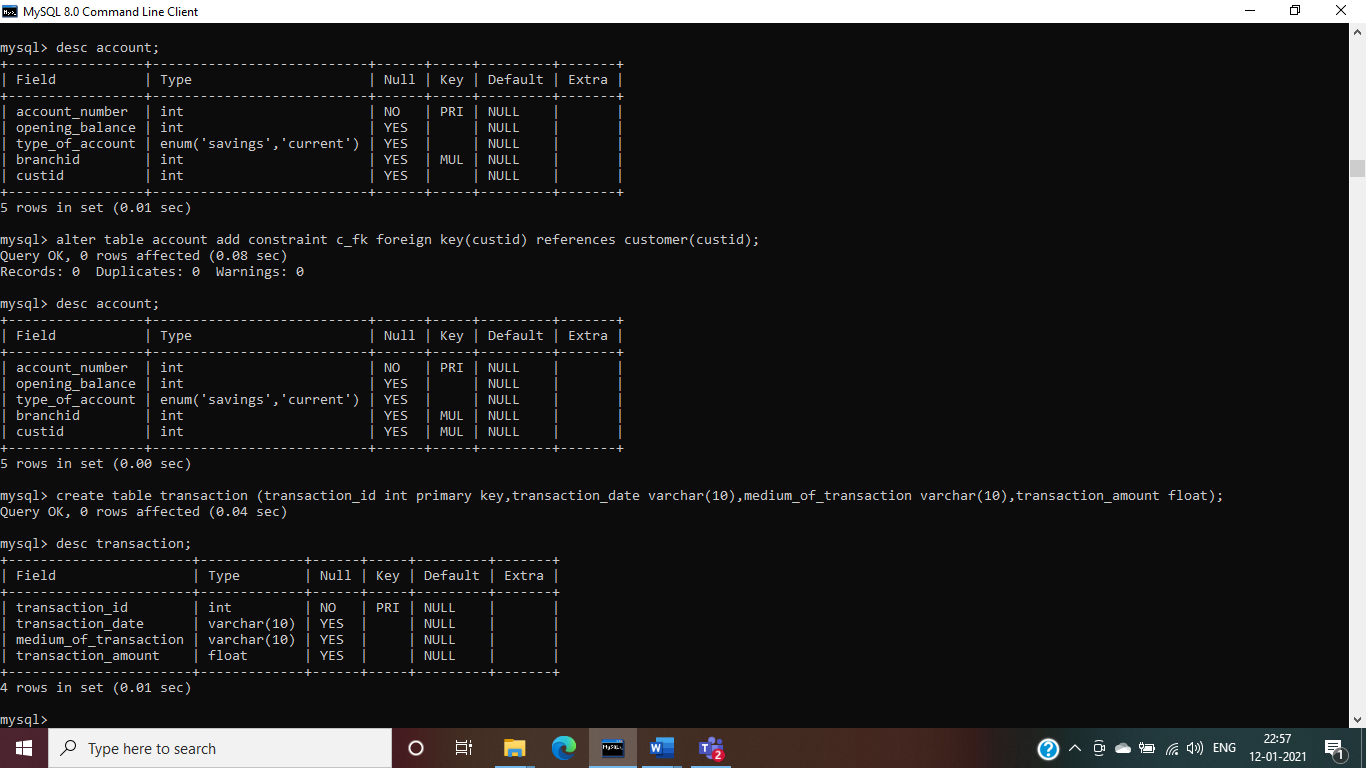


Table ----> Loan

LoanId, loanAmount, customerId and bankdId

Add the Constraints

loadId is primary key

loanAmount must be +ve

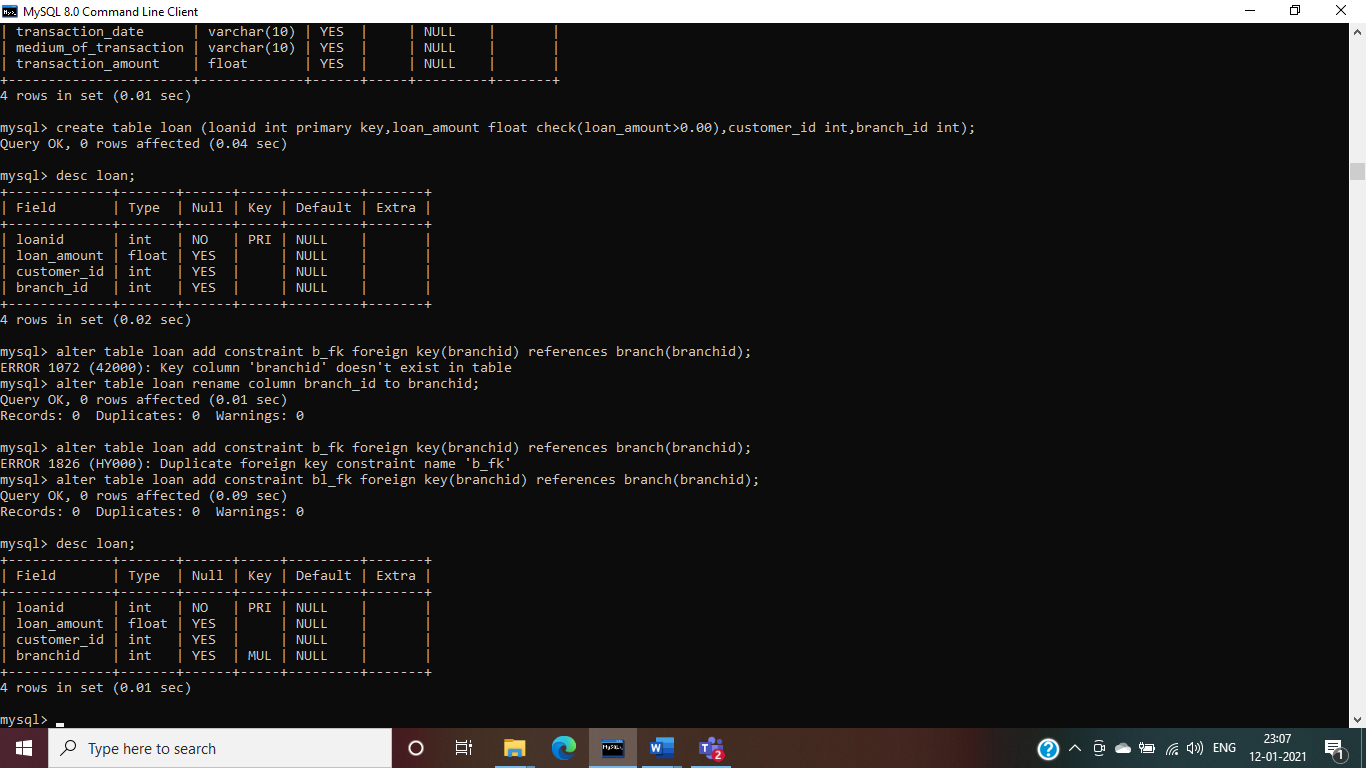
BankId is foreign key refer to BranchId(Primary key) Branch table

Ans-

create table loan (loanid int primary key,loan\_amount float check(loan\_amount>0.00),customer\_id int,branch\_id int);

alter table loan rename column branch\_id to branchid;

alter table loan add constraint bl\_fk foreign key(branchid) references branch(branchid);



**Sub Query**

1. Display details of departments managed by ‘John’.

Ans- select \* from departments where manager\_id in (select employee\_id from employees where first\_name=’John’);

2. Display employees who did not do any job in the past.

Ans-select \* from employees where employee\_id not in (select employee\_id from job\_history);

3. Display job title and average salary for employees who did a job in the past.

Ans-select job\_title ,avg(salary) from jobs natural join employees group by job\_title where employee\_id in (select employee\_id from job\_history);

4. Display country name, city, and number of departments where department has more than 5 employees.

Ans-select country\_name,city, count(department\_id) from countries join locations using (country\_id) join departments using (location\_id) where department\_id in (select department\_id from employees group by department\_id having count(department\_id)>5) group by country\_name,city;

5. Display details of manager who manages more than 5 employees.

Ans-select first\_name from employees where employee\_id in (select manager\_id from employees group by manager\_id having count(\*)>5);

6. Display details of current job for employees who worked as IT Programmers in the past.

Ans-select \* from jobs where job\_id in (select job\_id from employees where employee\_id in (select employee\_id from job\_history where job\_id = ‘IT\_PROG’));

7. Display the details of employees drawing the highest salary in the department.

Ans-select department\_id , first\_name,salary from employees outer where salary=(select max(salary) from employees where department\_id = outer.department\_id);

8. Display third highest salary of all employees

Ans- select salary from employees main where 2=(select count (distinct salary) from employees where salary> main.salary);