use test;

use studentb;

show tables;

desc student\_table;

select \* from student\_table1;

insert into student\_table1 values(10,'ravi','20220522','10c');

insert into student\_table values(10,'ravi','2022-05-22','10c');

insert into studentb.employee values('jamie@example.com',5079,'Jamie','Smith');

insert into studentb.employee values(jamie@example.com,5079,Jamie,Smith);

select \* from studentb.employee;

use apidatabase;

drop database apidatabase;

show tables;

desc employee;

desc student;

select \* from apidatabase.employee;

update apidatabase.employee set empname = 'Astha1';

create database if not exists RaviBank;

use ravibank;

create table if not exists bank\_details4(age int(4),job varchar(30),marital varchar(30),education varchar(30),

`default` varchar(30),balance int(4),housing varchar(30),loan varchar(30),contact varchar(30),

`day` int ,`month` varchar(15),duration int ,campaign int,pdays int ,previous int, poutcome varchar(30),y varchar(30));

desc bank\_details;

-- -load data local in file bank\_details(

insert into bank\_details values(58,"management","married","tertiary","no",2143,"yes","no","unknown",5,"may",261,1,-1,0,"unknown","no");

select \* from bank\_details4;

insert into bank\_details values

(44,"technician","single","secondary","no",29,"yes","no","unknown",5,"may",151,1,-1,0,"unknown","no"),

(33,"entrepreneur","married","secondary","no",2,"yes","yes","unknown",5,"may",76,1,-1,0,"unknown","no"),

(47,"blue-collar","married","unknown","no",1506,"yes","no","unknown",5,"may",92,1,-1,0,"unknown","no"),

(33,"unknown","single","unknown","no",1,"no","no","unknown",5,"may",198,1,-1,0,"unknown","no"),

(35,"management","married","tertiary","no",231,"yes","no","unknown",5,"may",139,1,-1,0,"unknown","no"),

(28,"management","single","tertiary","no",447,"yes","yes","unknown",5,"may",217,1,-1,0,"unknown","no"),

(42,"entrepreneur","divorced","tertiary","yes",2,"yes","no","unknown",5,"may",380,1,-1,0,"unknown","no"),

(58,"retired","married","primary","no",121,"yes","no","unknown",5,"may",50,1,-1,0,"unknown","no"),

(43,"technician","single","secondary","no",593,"yes","no","unknown",5,"may",55,1,-1,0,"unknown","no"),

(41,"admin.","divorced","secondary","no",270,"yes","no","unknown",5,"may",222,1,-1,0,"unknown","no"),

(29,"admin.","single","secondary","no",390,"yes","no","unknown",5,"may",137,1,-1,0,"unknown","no"),

(53,"technician","married","secondary","no",6,"yes","no","unknown",5,"may",517,1,-1,0,"unknown","no"),

(58,"technician","married","unknown","no",71,"yes","no","unknown",5,"may",71,1,-1,0,"unknown","no"),

(57,"services","married","secondary","no",162,"yes","no","unknown",5,"may",174,1,-1,0,"unknown","no"),

(51,"retired","married","primary","no",229,"yes","no","unknown",5,"may",353,1,-1,0,"unknown","no"),

(45,"admin.","single","unknown","no",13,"yes","no","unknown",5,"may",98,1,-1,0,"unknown","no"),

(57,"blue-collar","married","primary","no",52,"yes","no","unknown",5,"may",38,1,-1,0,"unknown","no"),

(60,"retired","married","primary","no",60,"yes","no","unknown",5,"may",219,1,-1,0,"unknown","no"),

(33,"services","married","secondary","no",0,"yes","no","unknown",5,"may",54,1,-1,0,"unknown","no"),

(28,"blue-collar","married","secondary","no",723,"yes","yes","unknown",5,"may",262,1,-1,0,"unknown","no"),

(56,"management","married","tertiary","no",779,"yes","no","unknown",5,"may",164,1,-1,0,"unknown","no"),

(32,"blue-collar","single","primary","no",23,"yes","yes","unknown",5,"may",160,1,-1,0,"unknown","no"),

(25,"services","married","secondary","no",50,"yes","no","unknown",5,"may",342,1,-1,0,"unknown","no"),

(40,"retired","married","primary","no",0,"yes","yes","unknown",5,"may",181,1,-1,0,"unknown","no"),

(44,"admin.","married","secondary","no",-372,"yes","no","unknown",5,"may",172,1,-1,0,"unknown","no"),

(39,"management","single","tertiary","no",255,"yes","no","unknown",5,"may",296,1,-1,0,"unknown","no"),

(52,"entrepreneur","married","secondary","no",113,"yes","yes","unknown",5,"may",127,1,-1,0,"unknown","no"),

(46,"management","single","secondary","no",-246,"yes","no","unknown",5,"may",255,2,-1,0,"unknown","no"),

(36,"technician","single","secondary","no",265,"yes","yes","unknown",5,"may",348,1,-1,0,"unknown","no"),

(57,"technician","married","secondary","no",839,"no","yes","unknown",5,"may",225,1,-1,0,"unknown","no");

select \* from bank\_details;

select `default`, age , `day`,`month` from bank\_details where age = 41;

select \* from bank\_details where job = 'retired' and balance > 30;

select \* from bank\_DetAIls where education = 'primary' or balance < 100;

select \* from bank\_details;

SELECT MARITAL,EDUCATION,JOB,HOUSING FROM BANK\_DETAILS;

SELECT DISTINCT MARITAL,EDUCATION,JOB,HOUSING FROM BANK\_DETAILS;

select count(\*) from bank\_details;

select \* from bank\_details order by age;

-- Bulk Upload of Data

SHOW VARIABLES LIKE "secure\_file\_priv";

--Make # Secure File Priv. in ini.my file to spaces which is stored in location shown by above command

--secure-file-priv = ""

load data infile 'D://bank.csv' into table bank\_details4

fields terminated by ';' Enclosed by '"' IGNORE 1 ROWS;

-- Aggregate Function

-- Select sum of all the Balance

select sum(balance) from bank\_details ;

-- Select average Balance

select avg(balance) from bank\_details;

-- Select minimum Balance

select min(balance) from bank\_details;

-- Select maximum Balance

select max(balance) from bank\_details;

-- Details of person having minimum Balance

select \* from bank\_details where balance = (select min(balance) from bank\_details);

-- Details of person having minimum Balance

select \* from bank\_details where balance = (select max(balance) from bank\_details);

select \* from bank\_details;

-- group by clause

select marital, count(\*) as count\_of\_records from bank\_details group by marital;

-- group by clause

select marital, count(\*) as count\_of\_records, sum(balance) as sum\_of\_balance, avg(balance) as average\_balance from bank\_details group by marital;

-- Grouping based on Balance with average balance > 300

select marital, count(\*) as count\_of\_records, sum(balance) as sum\_of\_balance, avg(balance) as average\_balance from bank\_details group by marital

having avg(balance) > 300;

select balance from bank\_details where job = 'unknown';

-- group by clause

select marital, education, max(Age), sum(balance), count(housing) from bank\_details group by marital, education;

select marital, education, max(Age), sum(balance), count(housing) from bank\_details group by marital;

-- Update transaction

-- set balance = 0 where job == unknown

update bank\_details set balance = 0 where job = 'unknown';

select \* from bank\_details;

update bank\_details set `default`= null where `default` = 'no';

-- Creating a Procedure

-- Creating Procedure for select clause

delimiter &&

create procedure select\_pre()

BEGIN

select \* from bank\_details;

END &&

call select\_pre();

-- Creating Procedure for minimum balance

delimiter &&

create procedure select\_min\_balance()

BEGIN

select \* from bank\_details where balance = (select min(balance) from bank\_details);

END &&

call select\_min\_balance();

-- Creating Procedure for update balance

delimiter &&

create procedure update\_balance(IN default\_value varchar(30) )

BEGIN

update bank\_details set education = 'primary' where `default`= default\_value;

END &&

call update\_balance('yes');

call update\_balance('null');

-- Creating Procedure with ifelse statment

delimiter &&

create procedure update\_balance4(IN default\_value varchar(30) )

BEGIN

IF default\_value ='null'

THEN

update bank\_details set education = 'secondary\_ravi' where `default` IS NULL;

ELSE

update bank\_details set education = 'primary\_ravi' where `default`= default\_value;

end if;

END &&

call update\_balance2('yes');

call update\_balance4('null');

--Procedure for balance print

create procedure balance\_print()

BEGIN

DECLARE @bal INT;

SELECT @bal = SUM(balance \* day)

FROM bank\_Details b1 INNER JOIN bank\_Details2 b2 ON B1.AGE= B2.AGE

WHERE YEAR(job) = 'management';

SELECT @bal;

IF @bal > 10000000

BEGIN

PRINT 'Great! The sales amount in 2018 is greater than 10,000,000';

END

ELSE

BEGIN

PRINT 'Sales amount in 2017 did not reach 10,000,000';

END

END

-- Views

-- in case we need data where age is 58

select \* from (select job,age,education, y from bank\_details) as a

where a.age = 58;

-- Creating view by selecting 4 fields whcih are required further

create view bank\_view2 as select job,age,education, y from bank\_details;

select \* from bank\_view2;

-- selecting data based on age

select \* from bank\_view2 where age = 58;

-- JOIN operations

--Create new table bank\_Details2

create table if not exists bank\_details1(age int(4),job varchar(30),marital varchar(30),education varchar(30),

`default` varchar(30),balance int(4),housing varchar(30),loan varchar(30),contact varchar(30),

`day` int ,`month` varchar(15),duration int ,campaign int,pdays int ,previous int, poutcome varchar(30),y varchar(30));

DELETE FROM BANK\_details1;

select \* from bank\_details1;

-- INSERTING DATA INTO BANK\_DETAILS1 from BANK\_DETAILS1

INSERT INTO BANK\_DETAILS1

SELECT \* FROM BANK\_DETAILS

where age = 58;

-- INNER JOIN

select B1.age,B1.job,B1.marital,B2.age,B2.job,B2.marital from bank\_details B1 INNER join bank\_details1 B2 ON B1.AGE= B2.AGE;

-- LEFT JOIN

select B1.age,B1.job,B1.marital,B2.age,B2.job,B2.marital from bank\_details B1 LEFT join bank\_details1 B2 ON B1.AGE= B2.AGE;

-- RIGHT JOIN

select B1.age,B1.job,B1.marital,B2.age,B2.job,B2.marital from bank\_details B1 RIGHT join bank\_details1 B2 ON B1.AGE= B2.AGE;