---TASK 1

create or replace DATABASE DEMODATABASE;

USE DEMODATABASE;

CREATE OR REPLACE TABLE SHOPPING\_HISTORY(

PRODUCT VARCHAR NOT NULL,

QUANTITY INT NOT NULL,

UNIT\_PRICE INT NOT NULL

);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('MILK',10,350);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('BREAD',5,250);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('TOMATO',10,30);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('POTATO',50,270);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('HOENY',1,400);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('BUTTER',3,90);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('WATCH',1,1350);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('CLOTHES',10,3500);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('BISCUIT',11,650);

INSERT INTO SHOPPING\_HISTORY(PRODUCT,QUANTITY,UNIT\_PRICE) VALUES ('JUICE',20,900);

SELECT \* FROM SHOPPING\_HISTORY;

SELECT PRODUCT, SUM(UNIT\_PRICE) FROM SHOPPING\_HISTORY

GROUP BY PRODUCT

ORDER BY PRODUCT DESC;

---task 2

create or replace table phones(

name varchar(20) not null unique,

phone\_number int not null unique);

insert into phones (name, phone\_number) values('yogesh',8790987092);

insert into phones (name, phone\_number) values('jayesh',8790657093);

insert into phones (name, phone\_number) values('prakash',8792387094);

insert into phones (name, phone\_number) values('prashant',8723987095);

insert into phones (name, phone\_number) values('pranav',8791287096);

create or replace table calls(

id int not null unique,

caller int not null,

callee int not null,

duration int not null

);

select \* from calls;

delete from calls;

insert into calls(id, caller, callee, duration) values(1,'8790987092','8790657093', 12);

insert into calls(id, caller, callee, duration) values(2,'8792387094','8723987095', 10);

insert into calls(id, caller, callee, duration) values(3,'8790987093','8790657092', 6);

insert into calls(id, caller, callee, duration) values(4,'8792387094','8790657094', 5);

insert into calls(id, caller, callee, duration) values(5,'8790987096','8790657092', 9);

SELECT name, duration

FROM calls LEFT OUTER JOIN phones ON calls.caller = phones.phone\_number

where duration >= 10

group by name,duration

order by name;

---Task 3

CREATE TABLE transactions(

Amount INTEGER NOT NULL,

Date DATE NOT NULL);

INSERT INTO transactions (Amount,Date) VALUES (1000, ‘2020-01-06’ );

INSERT INTO transactions (Amount,Date) VALUES (-10, ‘2020-01-14’ );

INSERT INTO transactions (Amount,Date) VALUES (-75, ‘2020-01-20’ );

INSERT INTO transactions (Amount,Date) VALUES (-5, ‘2020-01-25’ );

INSERT INTO transactions (Amount,Date) VALUES (-4, ‘2020-01-29’ );

INSERT INTO transactions (Amount,Date) VALUES (2000, ‘2020-03-10’ );

INSERT INTO transactions (Amount,Date) VALUES (-75, ‘2020-03-12’ );

INSERT INTO transactions (Amount,Date) VALUES (-20, ‘2020-03-15’ );

INSERT INTO transactions (Amount,Date) VALUES (40, ‘2020-03-15’ );

INSERT INTO transactions (Amount,Date) VALUES (-50, ‘2020-03-17’ );

INSERT INTO transactions (Amount,Date) VALUES (200, ‘2020-10-10’ );

INSERT INTO transactions (Amount,Date) VALUES (-200, ‘2020-10-10’);

WITH monthly\_payments AS (

SELECT

DATE\_TRUNC('month', date) AS month,

COUNT(\*) AS num\_payments,

SUM(CASE WHEN amount < 0 THEN amount ELSE 0 END) AS total\_payments

FROM transactions

WHERE amount < 0

GROUP BY month

HAVING COUNT(\*) >= 3 AND SUM(CASE WHEN amount < 0 THEN amount ELSE 0 END) <= -100

),

monthly\_fees AS (

SELECT

DATE\_TRUNC('month', date) AS month,

-5 AS fee

FROM transactions

WHERE amount > 0

GROUP BY month

),

balance\_changes AS (

SELECT

DATE\_TRUNC('month', date) AS month,

SUM(amount) AS balance\_change

FROM transactions

GROUP BY month

),

running\_balance AS (

SELECT

month,

SUM(balance\_change) OVER (ORDER BY month) AS balance

FROM balance\_changes

)

SELECT

COALESCE(SUM(fee), 0) + COALESCE(MAX(balance), 0) AS balance

FROM monthly\_fees

LEFT JOIN running\_balance ON monthly\_fees.month = running\_balance.month

WHERE running\_balance.month IS NULL OR running\_balance.month IN (SELECT month FROM monthly\_payments);