----------------------------------------------------------- TASK1-------------------------------------------------------------

use warehouse SHOP1WARE;

CREATE DATABASE SHOPHISTORY;

USE SHOPHISTORY;

CREATE OR REPLACE TABLE SHOPPING\_HISTORY(

PRODUCT VARCHAR NOT NULL,

QUANTITY INTEGER NOT NULL,

UNIT\_PRICE INTEGER NOT NULL

);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'MILK', 3, 10);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'BREAD', 5, 17);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'WHEAT', 2, 15);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'APPLE', 10, 7);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'NOODLES', 3, 10);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'MANGO', 9, 10);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'JUICE', 3, 4);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'VEGITABLES', 8, 6);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'SUGAR', 3, 10);

INSERT INTO SHOPPING\_HISTORY (PRODUCT,QUANTITY,UNIT\_PRICE) VALUES( 'RICE', 7, 12);

SELECT \* FROM SHOPPING\_HISTORY;

-- TOTAL PRICE OF EACH PRODUCT IN DESCENDING ALPHABETICAL ORDER OF PRODUCTS

SELECT PRODUCT, (QUANTITY \* UNIT\_PRICE) AS TOTAL\_PRICE FROM SHOPPING\_HISTORY ORDER BY PRODUCT DESC;

----------------------------------------TASK-2--------------------------------------------------------------------

CREATE OR REPLACE TABLE PHONE(

NAME VARCHAR(10) NOT NULL UNIQUE,

PHONE\_NUMBER INTEGER NOT NULL UNIQUE,

);

CREATE OR REPLACE TABLE CALLS(

ID INTEGER NOT NULL,

CALLER INTEGER NOT NULL,

CALLEE INTEGER NOT NULL,

DURATION INTEGER NOT NULL,

UNIQUE(ID)

);

SELECT \* FROM PHONE;

SELECT \* FROM CALLS;

SELECT NAME FROM

(SELECT TAB1.NAME, (TAB1.SM + TAB2.SM1 ) AS MINUTES FROM

(SELECT NAME, CALLER, SUM(DURATION) AS SM FROM (SELECT P.NAME, C.CALLER, C.CALLEE, C.DURATION FROM PHONE P LEFT JOIN CALLS C ON P.PHONE\_NUMBER = C.CALLER )TABLE1 GROUP BY NAME, CALLER ORDER BY NAME )TAB1

LEFT JOIN

(SELECT NAME, CALLEE, SUM(case when DURATION IS NULL then 0 when DURATION IS NOT NULL then DURATION end) AS SM1 FROM (SELECT P.NAME, C.CALLER, C.CALLEE, C.DURATION FROM PHONE P LEFT JOIN CALLS C ON P.PHONE\_NUMBER = C.CALLEE)TABLE2 GROUP BY NAME, CALLEE ORDER BY NAME)TAB2 ON TAB1.NAME= TAB2.NAME)A

WHERE MINUTES>=10;

----------------------------------------TASK-2--------------------------------------------------------------------

CREATE OR REPLACE TABLE PHONE(

NAME VARCHAR(10) NOT NULL UNIQUE,

PHONE\_NUMBER INTEGER NOT NULL UNIQUE,

);

CREATE OR REPLACE TABLE CALLS(

ID INTEGER NOT NULL,

CALLER INTEGER NOT NULL,

CALLEE INTEGER NOT NULL,

DURATION INTEGER NOT NULL,

UNIQUE(ID)

);

SELECT \* FROM PHONE;

SELECT \* FROM CALLS;

SELECT NAME FROM

(SELECT TAB1.NAME, (TAB1.SM + TAB2.SM1 ) AS MINUTES FROM

(SELECT NAME, CALLER, SUM(DURATION) AS SM FROM

(SELECT P.NAME, C.CALLER, C.CALLEE, C.DURATION FROM PHONE P LEFT JOIN CALLS C ON P.PHONE\_NUMBER = C.CALLER )TABLE1

GROUP BY NAME, CALLER ORDER BY NAME )TAB1

LEFT JOIN

(SELECT NAME, CALLEE, SUM(case when DURATION IS NULL then 0 when DURATION IS NOT NULL then DURATION end) AS SM1 FROM

(SELECT P.NAME, C.CALLER, C.CALLEE, C.DURATION FROM PHONE P LEFT JOIN CALLS C ON P.PHONE\_NUMBER = C.CALLEE)TABLE2

GROUP BY NAME, CALLEE ORDER BY NAME)TAB2

ON TAB1.NAME= TAB2.NAME)A WHERE MINUTES>=10;

-------------------------------------------------------TASK3----------------------------------------------------

CREATE OR REPLACE TABLE TRANSACTIONS(

AMOUNT INTEGER NOT NULL,

DATE DATE NOT NULL

);

SELECT \* FROM TRANSACTIONS;

SELECT (SUM(ACCOUNT) - (SUM(CREDIT)+(60- 5\* COUNT(case when CREDIT> 100 AND COUNT\_CREDIT>=3 then 1 end)))) AS BALANCE

FROM (SELECT MONTH(DATE) AS MON, SUM(case when AMOUNT< 0 then (AMOUNT \* -1) end) AS CREDIT,

COUNT(case when AMOUNT< 0 then (AMOUNT \* -1) end) AS COUNT\_CREDIT , SUM(case when AMOUNT>0 THEN (AMOUNT) end) ACCOUNT

FROM TRANSACTIONS GROUP BY MON) B;