**SQL ASSIGNMENT**

**TASK1:**

create database test;

create table if not exists shopping\_history (product varchar(30), Quantity integer not null , unit\_price integer not null);

insert into shopping\_history values ('milk',3,10),

('bread',7,3),

('bread',5,2);

select \* from shopping\_history;

ALTER TABLE shopping\_history

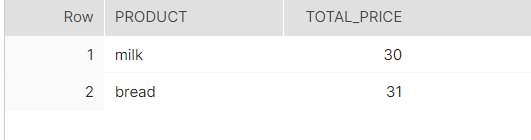
ADD COLUMN Total\_Price smallint;

UPDATE shopping\_history

SET Total\_Price = Quantity \* unit\_price ;

select product, sum(Total\_Price) as total\_price from shopping\_history group by product;

**Output:**

****

**TASK 2:**

create database test;

use test;

create table phones (name varchar (20) not null unique,phone\_number integer not null unique);

insert into phones values ('Jack',1234),

('Lena',3333),

('Mark',9999),

('Anna',7582);

select \* from phones;

create table calls (id integer not null,caller integer not null, callee integer not null, duration integer not null,unique (id));

insert into calls values(25,1234,7582,8),

(7,9999,7582,1),

(18,9999,3333,4),

(2,7582,3333,3),

(3,3333,1234,1),

(21,3333,1234,1);

select \* from calls;

CREATE TABLE total\_caller

(select caller, sum(duration) as total\_duration\_caller FROM calls GROUP BY caller);

CREATE TABLE total\_callee

(SELECT callee, sum(duration) as total\_duration\_callee FROM calls GROUP BY callee);

SELECT \* FROM total\_caller;

SELECT \* FROM total\_callee;

CREATE TABLE total\_duration(

SELECT p.name, p.phone\_number , cr.caller , cr.total\_duration\_caller , ce.callee , ce.total\_duration\_callee from phones p

LEFT JOIN total\_caller cr on p.phone\_number = cr.caller

LEFT JOIN total\_callee ce on p.phone\_number = ce.callee);

select \* from total\_duration;

ALTER TABLE total\_duration

ADD COLUMN TOTAL\_DURATION INT;

set SQL\_SAFE\_UPDATES = 0;

UPDATE TOTAL\_DURATION

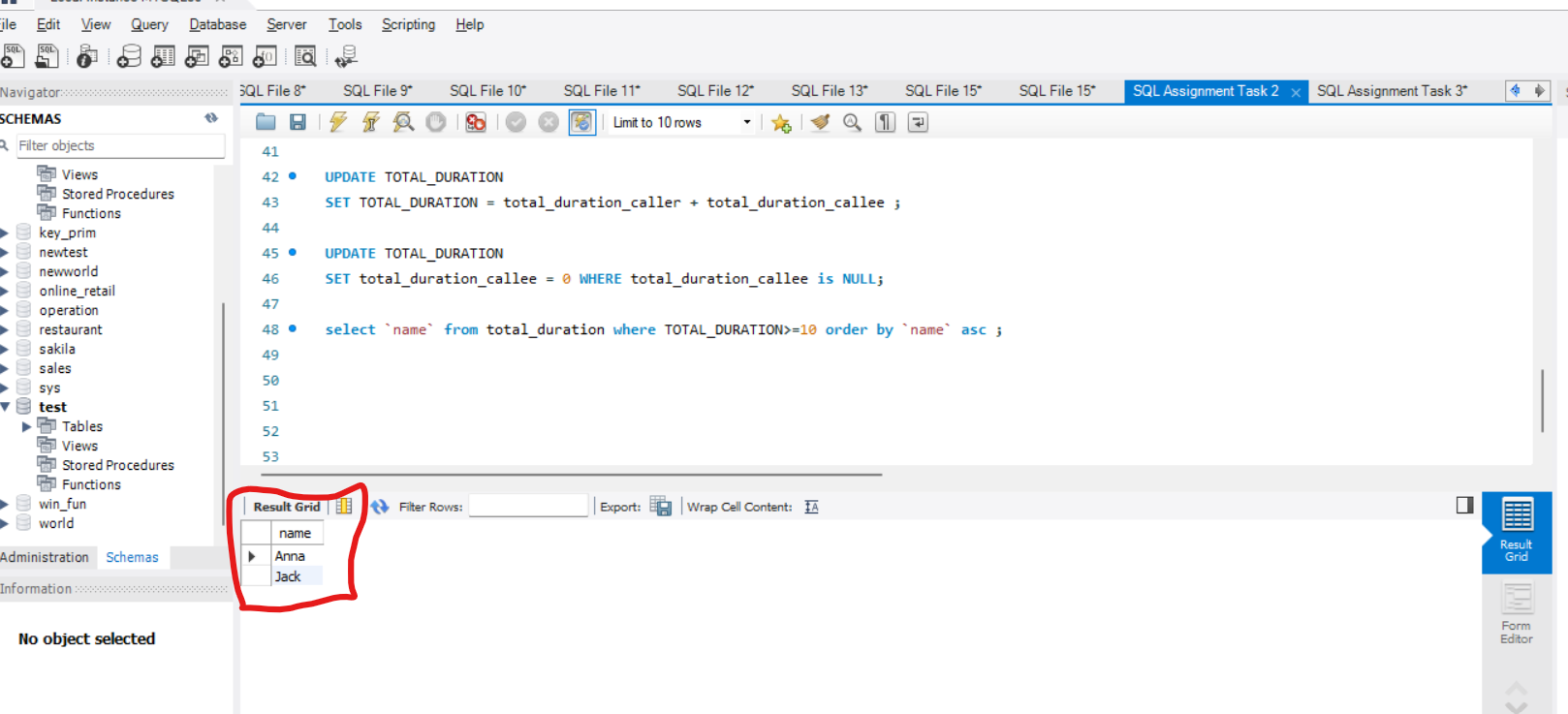
SET TOTAL\_DURATION = total\_duration\_caller + total\_duration\_callee ;

UPDATE TOTAL\_DURATION

SET total\_duration\_callee = 0 WHERE total\_duration\_callee is NULL;

select `name` from total\_duration where TOTAL\_DURATION>=10 order by `name` asc ;

**Output:**

****

**TASK 3(1):**

use test;

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');

select \* from transactions;

SET SQL\_SAFE\_UPDATES = 0;

ALTER TABLE transactions

ADD COLUMN mnth INT;

UPDATE transactions

SET mnth = EXTRACT(MONTH FROM `Date`);

ALTER TABLE transactions

ADD COLUMN comments varchar(30);

UPDATE transactions

SET comments = CASE

WHEN amount > 0 THEN 'Credit'

ELSE 'Debit'

END;

select \* from transactions limit 12;

CREATE TABLE if not exists transactions\_amount

SELECT mnth,

CASE

WHEN sum(amount) < -100 AND

count(amount) >= 3 THEN

sum(amount)

ELSE

sum(amount)-5

END AS amount

FROM transactions WHERE comments = 'Debit' GROUP BY mnth

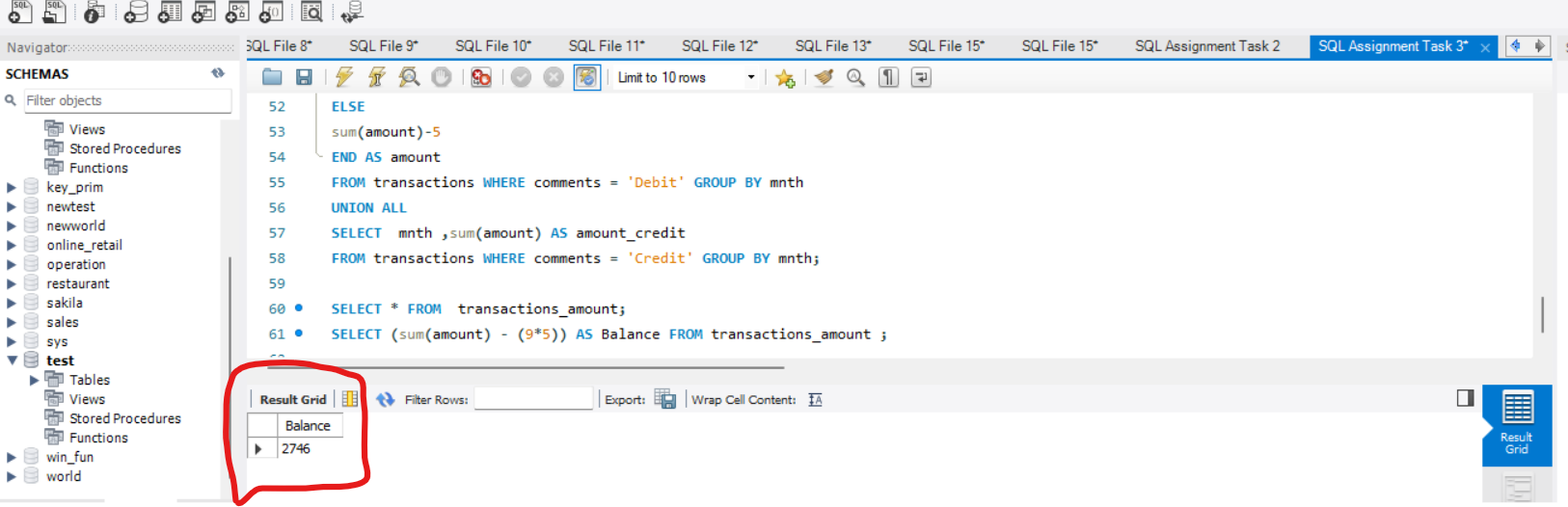
UNION ALL

SELECT mnth ,sum(amount) AS amount\_credit

FROM transactions WHERE comments = 'Credit' GROUP BY mnth;

SELECT \* FROM transactions\_amount;

SELECT (sum(amount) - (9\*5)) AS Balance FROM transactions\_amount ;

****

**TASK 3(2):**

use test;

CREATE TABLE new\_transactions (

Amount INTEGER NOT NULL,`Date` DATE NOT NULL);

INSERT INTO new\_transactions (Amount, `Date`) VALUES (1, '2020-06-09');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (35, '2020-02-20');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-50, '2020-02-03');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-1, '2020-02-26');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-200, '2020-08-01');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-44,'2020-02-07');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-5, '2020-02-25');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (1, '2020-06-29');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (1, '2020-06-29');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-100, '2020-12-29');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-100, '2020-12-30');

INSERT INTO new\_transactions (Amount, `Date`) VALUES (-100, '2020-12-31');

select \* from new\_transactions;

select count(\*) from new\_transactions;

SET SQL\_SAFE\_UPDATES = 0;

ALTER TABLE new\_transactions

ADD COLUMN mnth INT;

UPDATE new\_transactions

SET mnth = EXTRACT(MONTH FROM `Date`);

ALTER TABLE new\_transactions

ADD COLUMN new\_comments varchar(30);

UPDATE new\_transactions

SET new\_comments = CASE

WHEN amount > 0 THEN 'Credit'

ELSE 'Debit'

END;

select \* from new\_transactions limit 12;

select sum(Amount) from new\_transactions;

CREATE TABLE if not exists my\_transactions\_amount

SELECT mnth,

CASE

WHEN sum(Amount) <= -100 AND

count(Amount) >= 3 THEN

sum(Amount)

ELSE

sum(Amount)-5

END AS Amount

FROM new\_transactions WHERE new\_comments = 'Debit' GROUP BY mnth

UNION ALL

SELECT mnth ,sum(Amount) AS amount\_credit

FROM new\_transactions WHERE new\_comments = 'Credit' GROUP BY mnth;

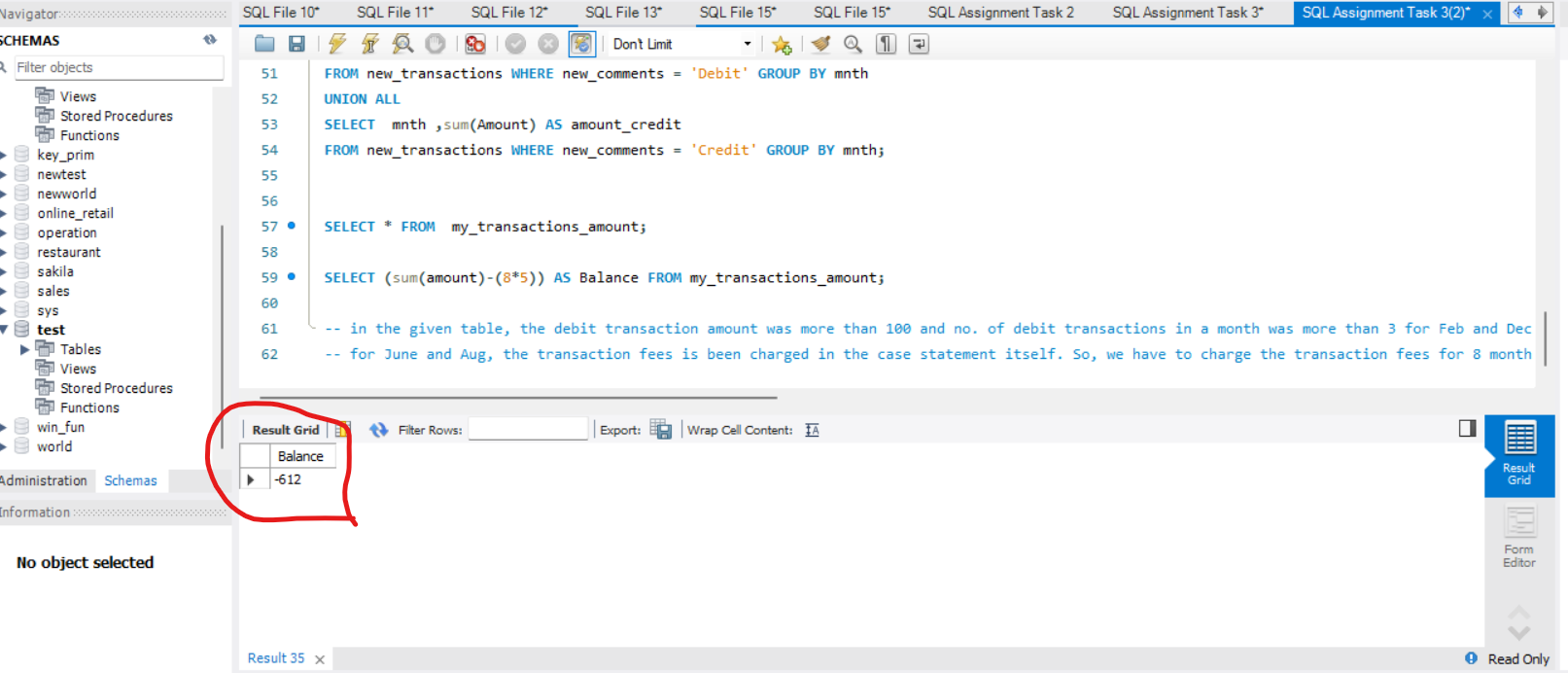
SELECT \* FROM my\_transactions\_amount;

SELECT (sum(amount)-(8\*5)) AS Balance FROM my\_transactions\_amount;

# in the given table, the debit transaction amount was more than 100 and no. of debit transactions in a month was more than 3 for Feb and Dec

for June and Aug, the transaction fees is been charged in the case statement itself. So, we have to charge the transaction fees for 8 months only.

Output:

****