```sql
use mydb_banking_system;

describe customer;
describe account;
describe transaction;


-- INSERTION -------------------------------------------------------------------------------------------------------------------
insert into customer(customer_first_name, customer_last_name, customer_dob) values
('ronald', 'weasley', '2001-02-10'),
('hermione', 'granger', '2002-11-15'),
('draco', 'malfoy', '2000-05-06');

select * from customer;

insert into account(account_type, account_balance, customer_id) values
('current', 120000, 2),
('zero_balance', 100000, 3),
('savings', 30000, 3),
('zero_balance', 40000, 1);

select * from account;

insert into transaction(transaction_type, transaction_amount, transaction_date, account_id) values
('deposit', 20000, '2024-02-02', 5),
('withdrawal', 8000, '2024-02-02', 6),
('transfer', 7000, '2024-02-05', 7);

select * from transaction;

-- ---------------------------------------------------------------------------------------------------------------------------

-- Task 2
/*
2. Write SQL queries for the following tasks:
1. Write a SQL query to retrieve the name, account type and email of all customers.
priority: customer
criteria: account
*/
select c.customer_first_name, c.customer_last_name, a.account_type
from customer c left join account a on c.customer_id = a.customer_id;
/* output
'ronald','weasley','zero_balance'
'hermione','granger','current'
'draco','malfoy','zero_balance'
'draco','malfoy','savings'

*/
```

```sql
-- 2. Write a SQL query to list all transaction corresponding customer.
select *
from transaction t left join account a on a.account_id = t.account_id
left join customer c on c.customer_id = a.customer_id;
/* output
'4','deposit','20000','2024-02-02','5','5','current','120000','2','2','hermione','granger','2002-11-15'
'5','withdrawal','8000','2024-02-02','6','6','zero_balance','101000','3','3','draco','malfoy','2000-05-06'
'6','transfer','7000','2024-02-05','7','7','savings','30000','3','3','draco','malfoy','2000-05-06'

*/


-- 3. Write a SQL query to increase the balance of a specific account by a certain amount.
update account
set account_balance = account_balance + 1000
where account_id = 6;

select * from account;
/* output
'5','current','120000','2'
'6','zero_balance','102000','3'
'7','savings','30000','3'
'8','zero_balance','40000','1'

*/


-- 4. Write a SQL query to Combine first and last names of customers as a full_name.
select c.customer_id, concat(c.customer_first_name,' ', c.customer_last_name) as 'Customer_full_name'
from customer c;
/* output
'1','ronald weasley'
'2','hermione granger'
'3','draco malfoy'

*/


-- 5. Write a SQL query to remove accounts with a balance of zero where the account type is savings.
delete
from account
where account_balance = 0
and account_type = 'savings';
/* output
*/


-- 6. Write a SQL query to Find customers living in a specific city.
-- NOT possible without 'city' column
-- 7. Write a SQL query to Get the account balance for a specific account.
select account_id, account_balance
from account
where account_id = 5;
/* output
'5','120000'

*/


-- 8. Write a SQL query to List all current accounts with a balance greater than $1,000.
select a.*
from account a
where a.account_type = 'current'
and a.account_balance > 1000;
/* output
'5','current','120000','2'

*/


-- 9. Write a SQL query to Retrieve all transactions for a specific account.
select a.account_id, t.*
from transaction t join account a on a.account_id = t.account_id
where a.account_id = 4;
/* output
*/


-- 10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.
-- NOT Possible without required columns(Interest Rate);


-- 11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit.
select *
from account
where account_balance < 3000;
/* output
*/


-- 12. Write a SQL query to Find customers not living in a specific city.
-- NOT possible without 'city' column



-- Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:
-- 1. Write a SQL query to Find the average account balance for all customers.
-- projection: account
-- criteria: customer
select c.customer_id, avg(a.account_balance)
from customer c join account a on c.customer_id = a.customer_id
group by c.customer_id;
/* output
'1','40000'
'2','120000'
'3','66000'

*/


-- 2. Write a SQL query to Retrieve the top 10 highest account balances.
select *
from account
order by account_balance desc
limit 10;
/* output
'5','current','120000','2'
'6','zero_balance','102000','3'
'8','zero_balance','40000','1'
'7','savings','30000','3'

*/


-- 3. Write a SQL query to Calculate Total Deposits for All Customers in specific date.
-- project: transaction
-- criteria: customer
select c.customer_id, sum(t.transaction_amount)
from customer c left join account a on c.customer_id = a.customer_id
left join transaction t on t.account_id = a.account_id
where t.transaction_date = '2024-02-02'
and t.transaction_type = 'deposit'
group by c.customer_id;
/* output
'2','20000'

*/


-- 4. Write a SQL query to Find the Oldest and Newest Customers.
select *
from customer
order by customer_dob;
/* output
'3','draco','malfoy','2000-05-06'
'1','ronald','weasley','2001-02-10'
'2','hermione','granger','2002-11-15'
```

```
*/


-- 5. Write a SQL query to Retrieve transaction details along with the account type.
select t.*, a.account_type
from transaction t join account a on t.account_id = a.account_id;
/* output
'4','deposit','20000','2024-02-02','5','current'
'5','withdrawal','8000','2024-02-02','6','zero_balance'
'6','transfer','7000','2024-02-05','7','savings'

*/


-- 6. Write a SQL query to Get a list of customers along with their account details.
-- projection: customer
-- criteria: account
select *
from customer c left join account a on c.customer_id = a.customer_id;
/* output
'1','ronald','weasley','2001-02-10','8','zero_balance','40000','1'
'2','hermione','granger','2002-11-15','5','current','120000','2'
'3','draco','malfoy','2000-05-06','6','zero_balance','102000','3'
'3','draco','malfoy','2000-05-06','7','savings','30000','3'

*/


-- 7. Write a SQL query to Retrieve transaction details along with customer information for a specific account.
select a.account_id, t.*, c.*
from customer c left join account a on c.customer_id = a.customer_id
left join transaction t on t.account_id = a.account_id
where a.account_id = 7;

select * from account;
/* output
'7','6','transfer','7000','2024-02-05','7','3','draco','malfoy','2000-05-06'

*/


-- 8. Write a SQL query to Identify customers who have more than one account.
-- projection: customer
-- criteria: account
select customer_id
from account
group by customer_id
having count(*) > 1;
/* output
3
```

```
*/


-- 9. Write a SQL query to Calculate the difference in transaction amounts between deposits and withdrawals.
select (select sum(transaction_amount)
from transaction
where transaction_type = 'deposit')
-
(select sum(transaction_amount)
from transaction
where transaction_type = 'withdrawal') as 'Difference';
/* output
12000
*/


-- 10. Write a SQL query to Calculate the average daily balance for each account over a specified period.
-- 11. Calculate the total balance for each account type.
select account_type, sum(account_balance)
from account
group by account_type;
/* output
'current','120000'
'zero_balance','142000'
'savings','30000'

*/


-- 12. Identify accounts with the highest number of transactions order by descending order.
-- projection: accounts
-- criteria: transaction
select a.account_id, count(*) as num_of_transaction
from account a join transaction t on a.account_id = t.account_id
group by a.account_id
order by num_of_transaction desc;
/* output
'5','1'
'6','1'
'7','1'

*/


-- 13. List customers with high aggregate account balances, along with their account types.
-- projection: customer
-- criteria: account
select c.customer_id, sum(account_balance) as aggregate, a.account_type
from customer c join account a on c.customer_id = a.customer_id
group by c.customer_id
```

```
order by aggregate desc;
/* output
*/


-- 14. Identify and list duplicate transactions based on transaction amount, date, and account


-- DOUBT 10,13,14



-- Task 4: : Subquery and its type:
-- 1. Retrieve the customer(s) with the highest account balance.
select c.*, a.account_balance
from customer c join account a on c.customer_id = a.customer_id
order by a.account_balance desc
limit 1;
/* output
'2','hermione','granger','2002-11-15','120000'
*/

-- 2. Calculate the average account balance for customers who have more than one account.
select c.customer_id, avg(a.account_balance)
from customer c join account a on c.customer_id = a.customer_id
group by c.customer_id
having count(a.account_id) > 1;
/* output
'3','65500'
*/


-- 3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.
select a.account_id
from account a join transaction t on a.account_id = t.account_id
where t.transaction_amount > (select avg(transaction_amount)
                                                        from transaction);
/* output
5
*/


-- 4. Identify customers who have no recorded transactions.
select c.customer_id
from customer c left join account a on c.customer_id = a.customer_id
left join transaction t on t.account_id = a.account_id
where t.transaction_id is null;
/* output
1
*/
```

```
-- 5. Calculate the total balance of accounts with no recorded transactions.
select sum(a.account_balance) as total_balance
from account a
left join transaction t on a.account_id = t.account_id
where t.transaction_id is null;
/* output
40000
*/


-- 6. Retrieve transactions for accounts with the lowest balance.

-- 7. Identify customers who have accounts of multiple types.
select c.customer_id
from customer c
join account a on c.customer_id = a.customer_id
group by c.customer_id
having count(distinct a.account_type) > 1;
/* output
3
*/

-- 8. Calculate the percentage of each account type out of the total number of accounts.

-- 9. Retrieve all transactions for a customer with a given customer_id.
select t.*
FROM transaction t
join account a on t.account_id = a.account_id
where a.customer_id = 2;
/* output
'4','deposit','20000','2024-02-02','5'
*/


-- 10. Calculate the total balance for each account type, including a subquery within the SELECT clause.
select account_type, sum(account_balance)
from account
group by account_type;
/* output
'current','120000'
'zero_balance','141000'
'savings','30000'
*/
```