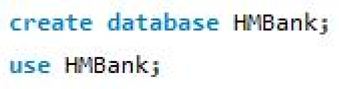
Banking System

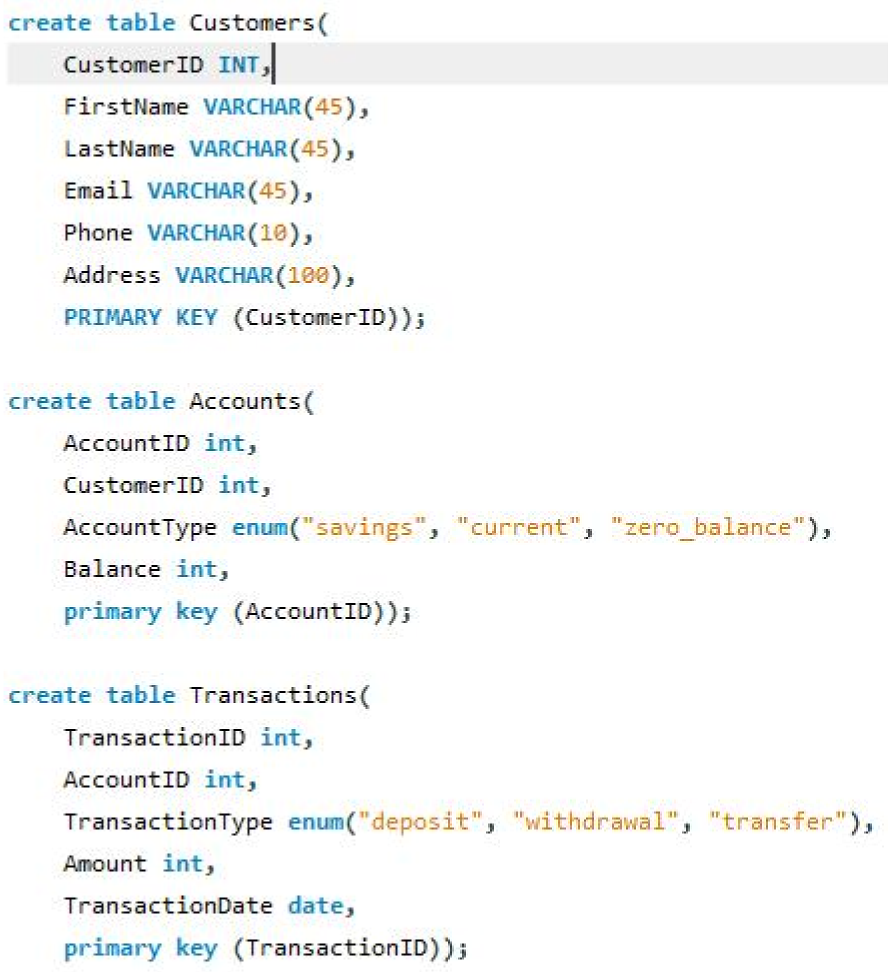
Tasks 1: Database Design:

1. Create the database named "HMBank"



```
create database HMBank;
use HMBank;
```

2. Define the schema for the Customers, Accounts, and Transactions tables based on the provided schema.

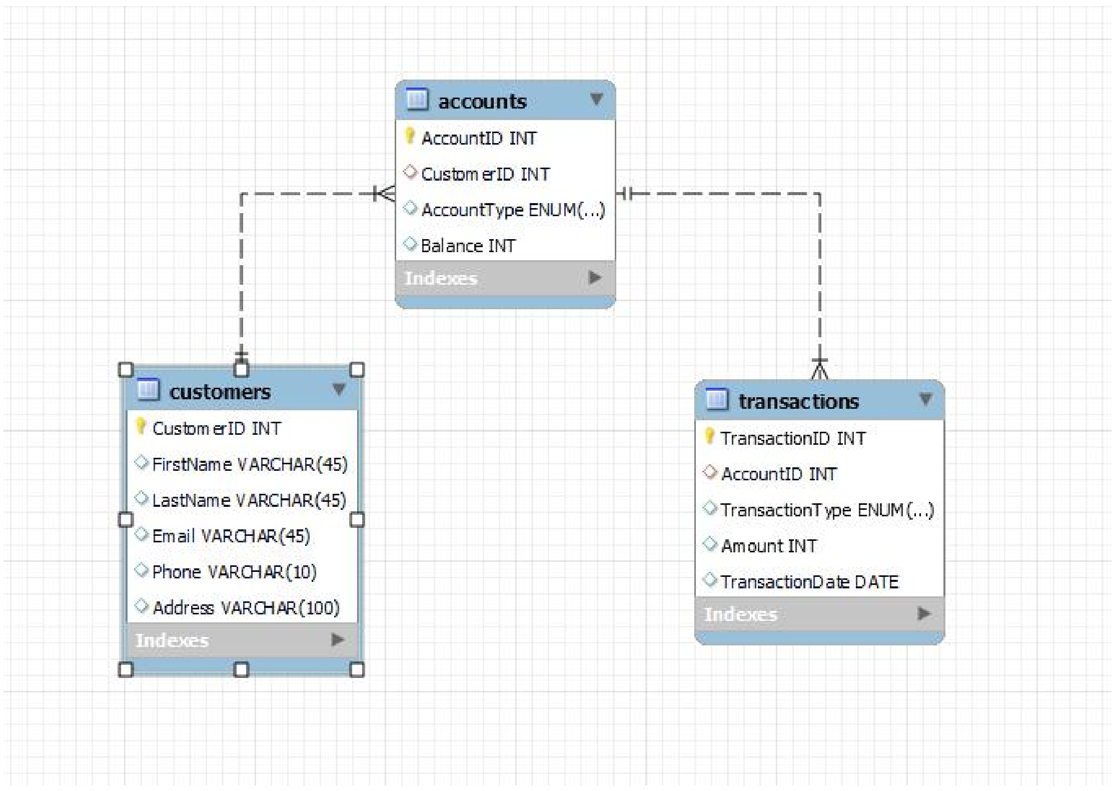


```
create table Customers(
    CustomerID INT,
    FirstName VARCHAR(45),
    LastName VARCHAR(45),
    Email VARCHAR(45),
    Phone VARCHAR(10),
    Address VARCHAR(100),
    PRIMARY KEY (CustomerID));

create table Accounts(
    AccountID int,
    CustomerID int,
    AccountType enum("savings", "current", "zero_balance"),
    Balance int,
    primary key (AccountID));

create table Transactions(
    TransactionID int,
    AccountID int,
    TransactionType enum("deposit", "withdrawal", "transfer"),
    Amount int,
    TransactionDate date,
    primary key (TransactionID));
```

3. Create an ERD (Entity Relationship Diagram) for the database.



4. Create appropriate Primary Key and Foreign Key constraints for referential integrity.

```
ALTER TABLE Accounts ADD FOREIGN KEY(CustomerID) REFERENCES Customers(CustomerID);
ALTER TABLE Transactions ADD FOREIGN KEY(AccountID) REFERENCES Accounts(AccountID);
```

5. Write SQL scripts to create the mentioned tables with appropriate data types, constraints, and relationships.
• Customers
• Accounts
• Transactions

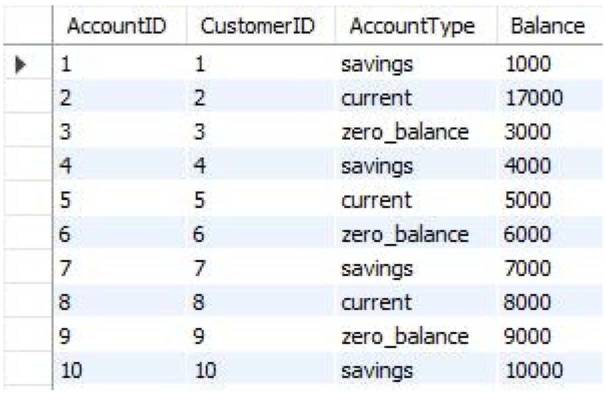
Tasks 2: Select, Where, Between, AND, LIKE:

1. Insert at least 10 sample records into each of the following tables.
• Customers
• Accounts
• Transactions

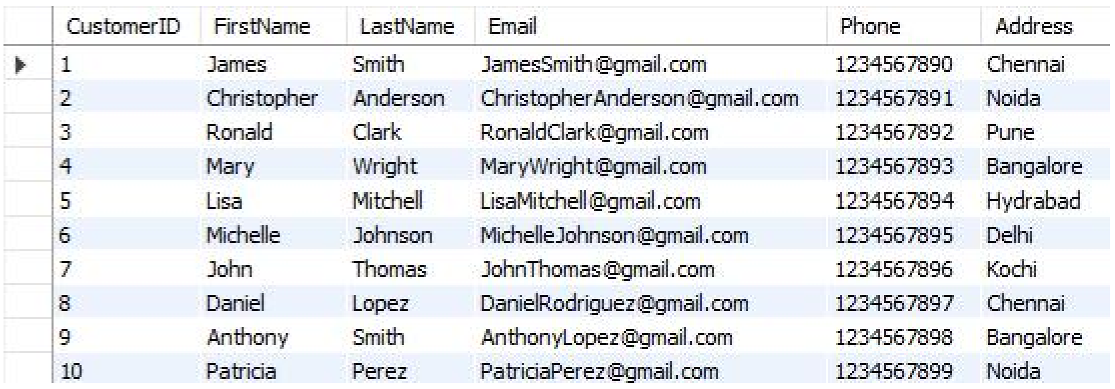
```sql
insert into Customers values
    (1, "James", "Smith", "JamesSmith@gmail.com", "1234567890", "Chennai"),
    (2, "Christopher", "Anderson", "ChristopherAnderson@gmail.com", "1234567891", "Noida"),
    (3, "Ronald", "Clark", "RonaldClark@gmail.com", "1234567892", "Pune"),
    (4, "Mary", "Wright", "MaryWright@gmail.com", "1234567893", "Bangalore"),
    (5, "Lisa", "Mitchell", "LisaMitchell@gmail.com", "1234567894", "Hydrabad"),
    (6, "Michelle", "Johnson", "MichelleJohnson@gmail.com", "1234567895", "Delhi"),
    (7, "John", "Thomas", "JohnThomas@gmail.com", "1234567896", "Kochi"),
    (8, "Daniel", "Lopez", "DanielRodriguez@gmail.com","1234567897", "Chennai"),
    (9, "Anthony", "Smith", "AnthonyLopez@gmail.com", "1234567898", "Bangalore"),
    (10, "Patricia", "Perez", "PatriciaPerez@gmail.com", "1234567899", "Noida");
```

```sql
insert into Accounts values
    (1, 1, "savings", 1000),
    (2, 2, "current", 2000),
    (3, 3, "zero_balance", 3000),
    (4, 4, "savings", 4000),
    (5, 5, "current", 5000),
    (6, 6, "zero_balance", 6000),
    (7, 7, "savings", 7000),
    (8, 8, "current", 8000),
    (9, 9, "zero_balance", 9000),
    (10, 10, "savings", 10000);
```

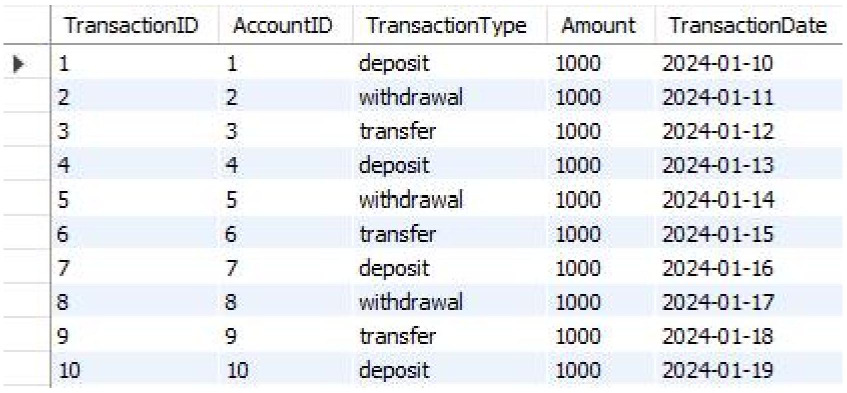
```sql
insert into Transactions values
    (1, 1, "deposit", 1000, "2024-1-10"),
    (2, 2, "withdrawal", 1000, "2024-1-11"),
    (3, 3, "transfer", 1000, "2024-1-12"),
    (4, 4, "deposit", 1000, "2024-1-13"),
    (5, 5, "withdrawal", 1000, "2024-1-14"),
    (6, 6, "transfer", 1000, "2024-1-15"),
    (7, 7, "deposit", 1000, "2024-1-16"),
    (8, 8, "withdrawal", 1000, "2024-1-17"),
    (9, 9, "transfer", 1000, "2024-1-18"),
    (10, 10, "deposit", 1000, "2024-1-19");
```



AccountID	CustomerID	AccountType	Balance
1	1	savings	1000
2	2	current	17000
3	3	zero_balance	3000
4	4	savings	4000
5	5	current	5000
6	6	zero_balance	6000
7	7	savings	7000
8	8	current	8000
9	9	zero_balance	9000
10	10	savings	10000



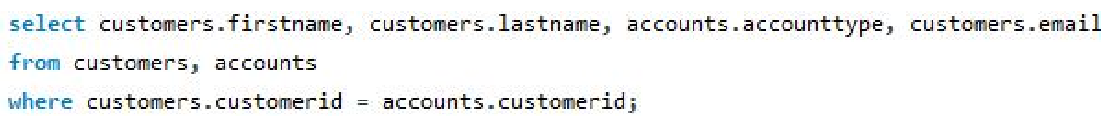
	CustomerID	FirstName	LastName	Email	Phone	Address
▶	1	James	Smith	JamesSmith@gmail.com	1234567890	Chennai
	2	Christopher	Anderson	ChristopherAnderson@gmail.com	1234567891	Noida
	3	Ronald	Clark	RonaldClark@gmail.com	1234567892	Pune
	4	Mary	Wright	MaryWright@gmail.com	1234567893	Bangalore
	5	Lisa	Mitchell	LisaMitchell@gmail.com	1234567894	Hydrabad
	6	Michelle	Johnson	MichelleJohnson@gmail.com	1234567895	Delhi
	7	John	Thomas	JohnThomas@gmail.com	1234567896	Kochi
	8	Daniel	Lopez	DanielRodriguez@gmail.com	1234567897	Chennai
	9	Anthony	Smith	AnthonyLopez@gmail.com	1234567898	Bangalore
	10	Patricia	Perez	PatriciaPerez@gmail.com	1234567899	Noida



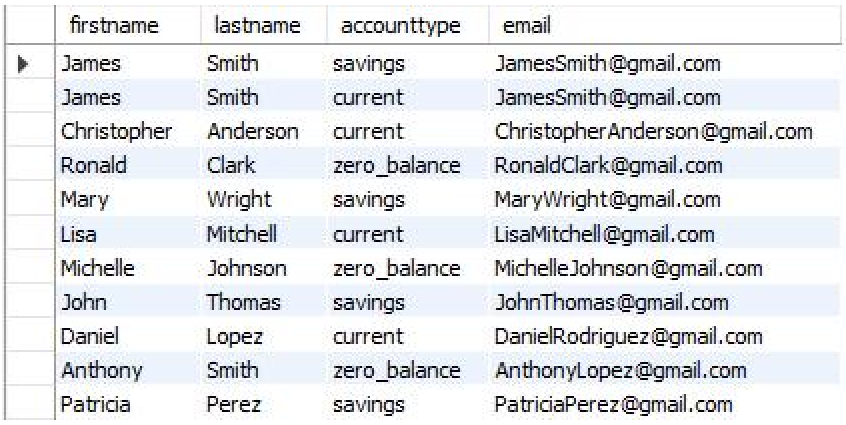
	TransactionID	AccountID	TransactionType	Amount	TransactionDate
▶	1	1	deposit	1000	2024-01-10
	2	2	withdrawal	1000	2024-01-11
	3	3	transfer	1000	2024-01-12
	4	4	deposit	1000	2024-01-13
	5	5	withdrawal	1000	2024-01-14
	6	6	transfer	1000	2024-01-15
	7	7	deposit	1000	2024-01-16
	8	8	withdrawal	1000	2024-01-17
	9	9	transfer	1000	2024-01-18
	10	10	deposit	1000	2024-01-19

2. Write SQL queries for the following tasks:

1. Write a SQL query to retrieve the name, account type and email of all customers.

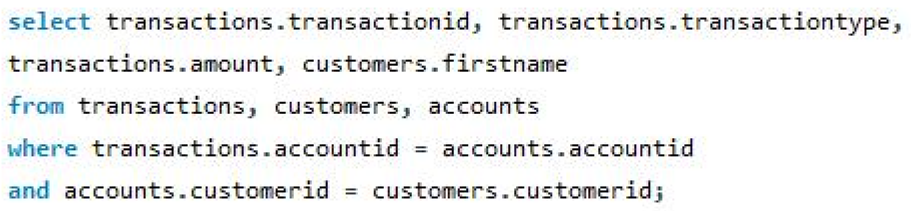


```sql
select customers.firstname, customers.lastname, accounts.accounttype, customers.email
from customers, accounts
where customers.customerid = accounts.customerid;
```

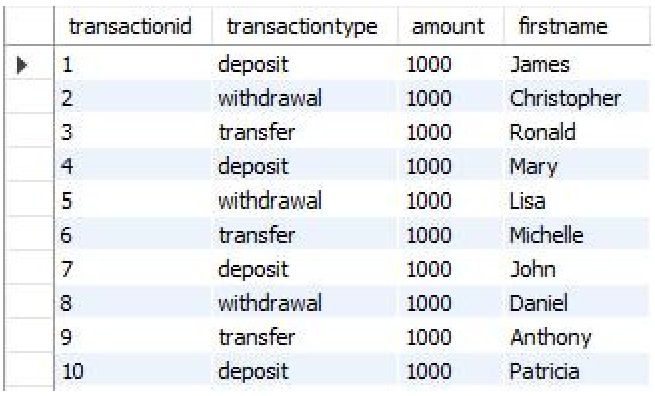


	firstname	lastname	accounttype	email
▶	James	Smith	savings	JamesSmith@gmail.com
	James	Smith	current	JamesSmith@gmail.com
	Christopher	Anderson	current	ChristopherAnderson@gmail.com
	Ronald	Clark	zero_balance	RonaldClark@gmail.com
	Mary	Wright	savings	MaryWright@gmail.com
	Lisa	Mitchell	current	LisaMitchell@gmail.com
	Michelle	Johnson	zero_balance	MichelleJohnson@gmail.com
	John	Thomas	savings	JohnThomas@gmail.com
	Daniel	Lopez	current	DanielRodriguez@gmail.com
	Anthony	Smith	zero_balance	AnthonyLopez@gmail.com
	Patricia	Perez	savings	PatriciaPerez@gmail.com

2. Write a SQL query to list all transaction corresponding customer.

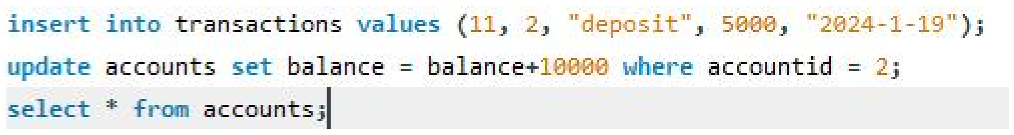


```
select transactions.transactionid, transactions.transactiontype,
transactions.amount, customers.firstname
from transactions, customers, accounts
where transactions.accountid = accounts.accountid
and accounts.customerid = customers.customerid;
```

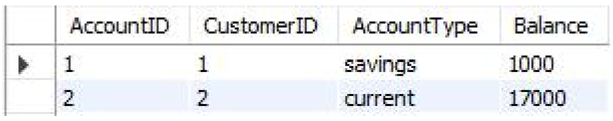


transactionid	transactiontype	amount	firstname
1	deposit	1000	James
2	withdrawal	1000	Christopher
3	transfer	1000	Ronald
4	deposit	1000	Mary
5	withdrawal	1000	Lisa
6	transfer	1000	Michelle
7	deposit	1000	John
8	withdrawal	1000	Daniel
9	transfer	1000	Anthony
10	deposit	1000	Patricia

3. Write a SQL query to increase the balance of a specific account by a certain amount.

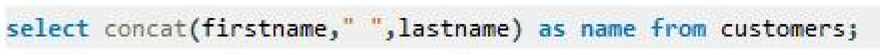


```
insert into transactions values (11, 2, "deposit", 5000, "2024-1-19");
update accounts set balance = balance+10000 where accountid = 2;
select * from accounts;
```

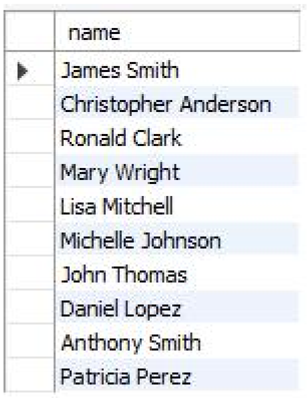


AccountID	CustomerID	AccountType	Balance
1	1	savings	1000
2	2	current	17000

4. Write a SQL query to Combine first and last names of customers as a full_name.



```
select concat(firstname," ",lastname) as name from customers;
```



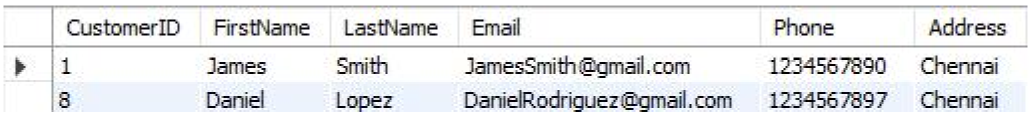
name
James Smith
Christopher Anderson
Ronald Clark
Mary Wright
Lisa Mitchell
Michelle Johnson
John Thomas
Daniel Lopez
Anthony Smith
Patricia Perez

5. Write a SQL query to remove accounts with a balance of zero where the account type is savings.

```
delete from accounts where balance = 0 and accounttype = "savings";
```

6. Write a SQL query to Find customers living in a specific city.

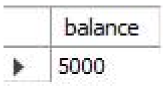
```
select * from customers where address = "Chennai";
```



CustomerID	FirstName	LastName	Email	Phone	Address
1	James	Smith	JamesSmith@gmail.com	1234567890	Chennai
8	Daniel	Lopez	DanielRodriguez@gmail.com	1234567897	Chennai

7. Write a SQL query to Get the account balance for a specific account.

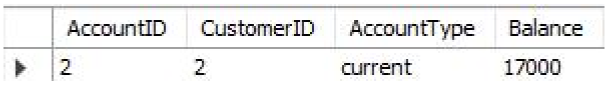
```
select balance from accounts where accountid = 5;
```



balance
5000

8. Write a SQL query to List all current accounts with a balance greater than $1,000.

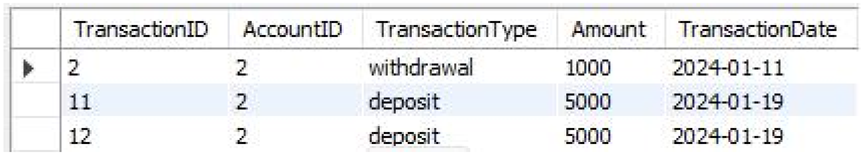
```
select * from accounts where accounttype = "current" and balance > 10000;
```



AccountID	CustomerID	AccountType	Balance
2	2	current	17000

9. Write a SQL query to Retrieve all transactions for a specific account.

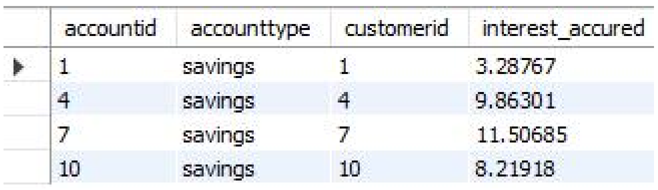
```
select * from transactions where accountid = 2;
```



TransactionID	AccountID	TransactionType	Amount	TransactionDate
2	2	withdrawal	1000	2024-01-11
11	2	deposit	5000	2024-01-19
12	2	deposit	5000	2024-01-19

10. Write a SQL query to Calculate the interest accrued on savings accounts based on a given interest rate.

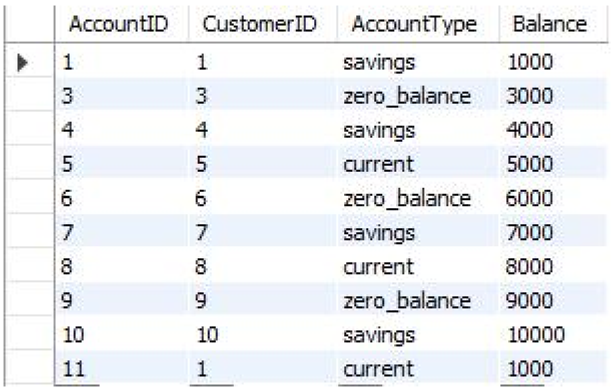
```
select accounts.accountid, accounttype, customerid,
balance*0.1*(datediff(now(), transactions.transactiondate)/365) as interest_accured
from accounts, transactions
where accounts.accountid = transactions.accountid and accounts.accounttype="savings";
```



accountid	accounttype	customerid	interest_accured
1	savings	1	3.28767
4	savings	4	9.86301
7	savings	7	11.50685
10	savings	10	8.21918

11. Write a SQL query to Identify accounts where the balance is less than a specified overdraft limit.

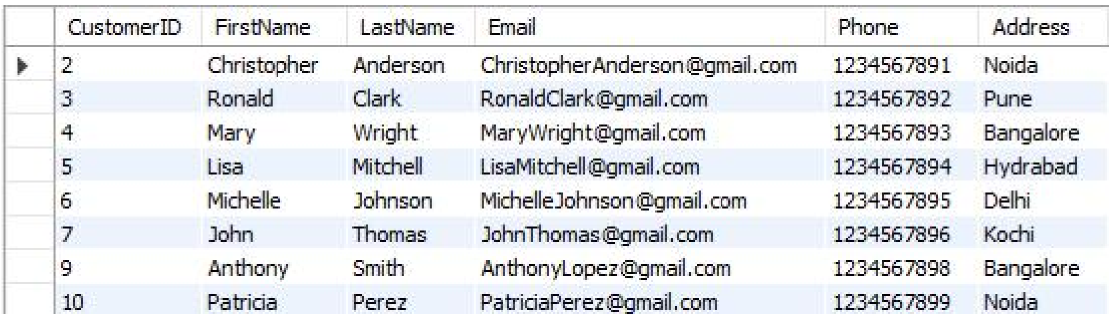
```
set @overdraftlimit = 15000;
select * from accounts where balance < (select @overdraftlimit);
```



AccountID	CustomerID	AccountType	Balance
1	1	savings	1000
3	3	zero_balance	3000
4	4	savings	4000
5	5	current	5000
6	6	zero_balance	6000
7	7	savings	7000
8	8	current	8000
9	9	zero_balance	9000
10	10	savings	10000
11	1	current	1000

12. Write a SQL query to Find customers not living in a specific city.

```
select * from customers where address != "Chennai";
```

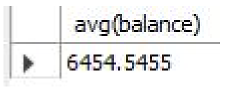


CustomerID	FirstName	LastName	Email	Phone	Address
2	Christopher	Anderson	ChristopherAnderson@gmail.com	1234567891	Noida
3	Ronald	Clark	RonaldClark@gmail.com	1234567892	Pune
4	Mary	Wright	MaryWright@gmail.com	1234567893	Bangalore
5	Lisa	Mitchell	LisaMitchell@gmail.com	1234567894	Hydrabad
6	Michelle	Johnson	MichelleJohnson@gmail.com	1234567895	Delhi
7	John	Thomas	JohnThomas@gmail.com	1234567896	Kochi
9	Anthony	Smith	AnthonyLopez@gmail.com	1234567898	Bangalore
10	Patricia	Perez	PatriciaPerez@gmail.com	1234567899	Noida

Tasks 3: Aggregate functions, Having, Order By, GroupBy and Joins:

1. Write a SQL query to Find the average account balance for all customers.

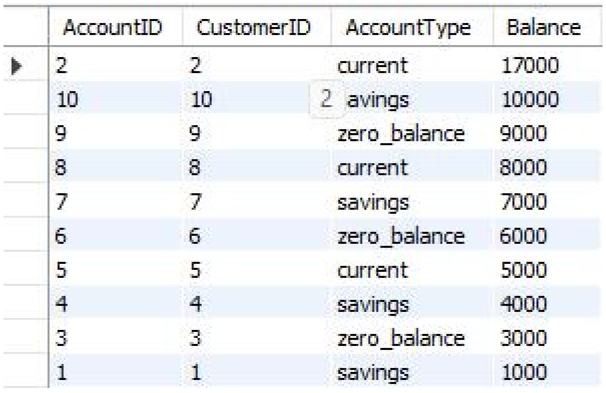
```
select avg(balance) from accounts;
```



avg(balance)
6454.5455

2. Write a SQL query to Retrieve the top 10 highest account balances.

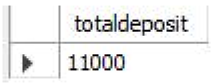
```
select * from accounts order by balance desc limit 10;
```



AccountID	CustomerID	AccountType	Balance
2	2	current	17000
10	10	2 avings	10000
9	9	zero_balance	9000
8	8	current	8000
7	7	savings	7000
6	6	zero_balance	6000
5	5	current	5000
4	4	savings	4000
3	3	zero_balance	3000
1	1	savings	1000

3. Write a SQL query to Calculate Total Deposits for All Customers in specific date.

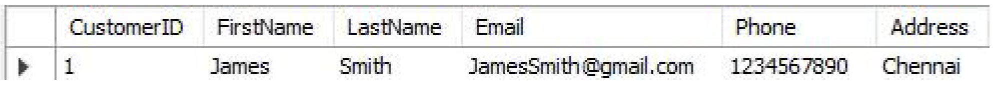
```
select sum(amount) as totaldeposit from transactions where transactiondate = "2024-1-19";
```



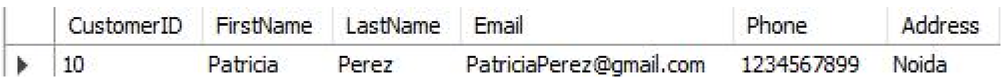
totaldeposit
11000

4. Write a SQL query to Find the Oldest and Newest Customers.

```
select * from customers limit 1;
select * from customers order by customerid desc limit 1;
```



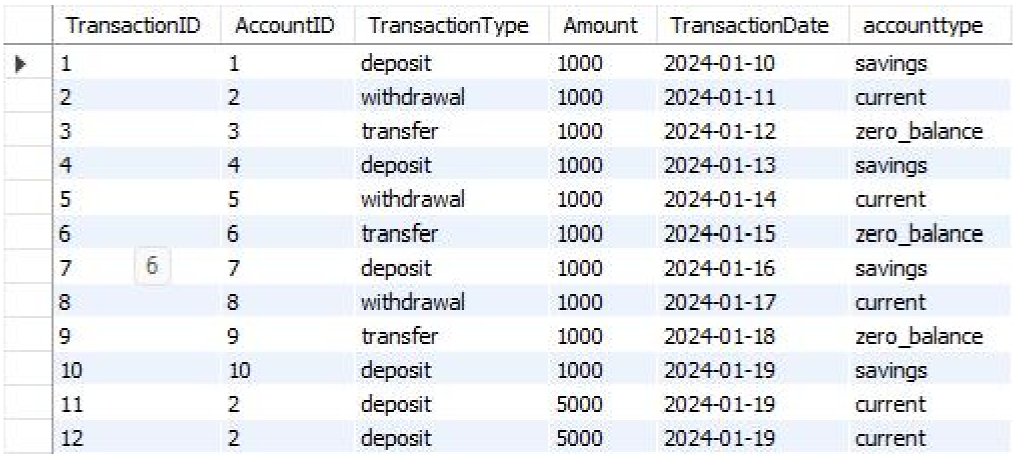
CustomerID	FirstName	LastName	Email	Phone	Address
1	James	Smith	JamesSmith@gmail.com	1234567890	Chennai



CustomerID	FirstName	LastName	Email	Phone	Address
10	Patricia	Perez	PatriciaPerez@gmail.com	1234567899	Noida

5. Write a SQL query to Retrieve transaction details along with the account type.

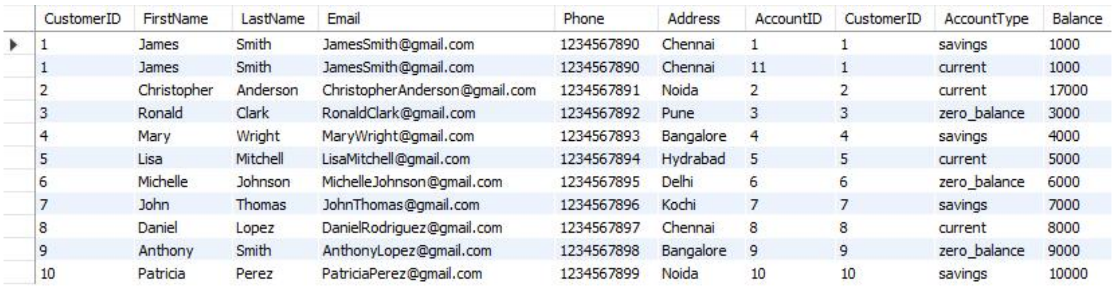
```
select transactions.*, accounts.accounttype
from transactions join accounts
on transactions.accountid = accounts.accountid
order by transactions.transactionid;
```



	TransactionID	AccountID	TransactionType	Amount	TransactionDate	accounttype
▶	1	1	deposit	1000	2024-01-10	savings
	2	2	withdrawal	1000	2024-01-11	current
	3	3	transfer	1000	2024-01-12	zero_balance
	4	4	deposit	1000	2024-01-13	savings
	5	5	withdrawal	1000	2024-01-14	current
	6	6	transfer	1000	2024-01-15	zero_balance
	7	7	deposit	1000	2024-01-16	savings
	8	8	withdrawal	1000	2024-01-17	current
	9	9	transfer	1000	2024-01-18	zero_balance
	10	10	deposit	1000	2024-01-19	savings
	11	2	deposit	5000	2024-01-19	current
	12	2	deposit	5000	2024-01-19	current

6. Write a SQL query to Get a list of customers along with their account details.

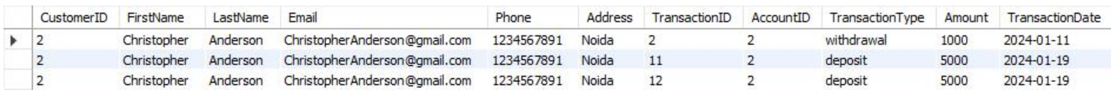
```
select * from customers join accounts
on customers.customerid = accounts.customerid;
```



	CustomerID	FirstName	LastName	Email	Phone	Address	AccountID	CustomerID	AccountType	Balance
▶	1	James	Smith	JamesSmith@gmail.com	1234567890	Chennai	1	1	savings	1000
	1	James	Smith	JamesSmith@gmail.com	1234567890	Chennai	11	1	current	1000
	2	Christopher	Anderson	ChristopherAnderson@gmail.com	1234567891	Noida	2	2	current	17000
	3	Ronald	Clark	RonaldClark@gmail.com	1234567892	Pune	3	3	zero_balance	3000
	4	Mary	Wright	MaryWright@gmail.com	1234567893	Bangalore	4	4	savings	4000
	5	Lisa	Mitchell	LisaMitchell@gmail.com	1234567894	Hydrabad	5	5	current	5000
	6	Michelle	Johnson	MichelleJohnson@gmail.com	1234567895	Delhi	6	6	zero_balance	6000
	7	John	Thomas	JohnThomas@gmail.com	1234567896	Kochi	7	7	savings	7000
	8	Daniel	Lopez	DanielRodriguez@gmail.com	1234567897	Chennai	8	8	current	8000
	9	Anthony	Smith	AnthonyLopez@gmail.com	1234567898	Bangalore	9	9	zero_balance	9000
	10	Patricia	Perez	PatriciaPerez@gmail.com	1234567899	Noida	10	10	savings	10000

7. Write a SQL query to Retrieve transaction details along with customer information for a specific account.

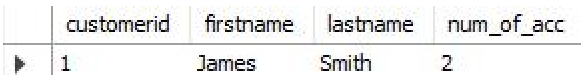
```
select customers.*, transactions.*
from customers join accounts
on customers.customerid = accounts.customerid join transactions
on accounts.accountid = transactions.accountid
where accounts.accountid = 2;
```



	CustomerID	FirstName	LastName	Email	Phone	Address	TransactionID	AccountID	TransactionType	Amount	TransactionDate
▶	2	Christopher	Anderson	ChristopherAnderson@gmail.com	1234567891	Noida	2	2	withdrawal	1000	2024-01-11
	2	Christopher	Anderson	ChristopherAnderson@gmail.com	1234567891	Noida	11	2	deposit	5000	2024-01-19
	2	Christopher	Anderson	ChristopherAnderson@gmail.com	1234567891	Noida	12	2	deposit	5000	2024-01-19

8. Write a SQL query to Identify customers who have more than one account.

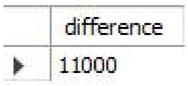
```
select customers.customerid, firstname, lastname, count(accountid)
as num_of_acc from customers join accounts
on customers.customerid = accounts.customerid
group by customers.customerid, firstname, lastname
having count(accountid)>1;
```



customerid	firstname	lastname	num_of_acc
1	James	Smith	2

9. Write a SQL query to Calculate the difference in transaction amounts between deposits and withdrawals.

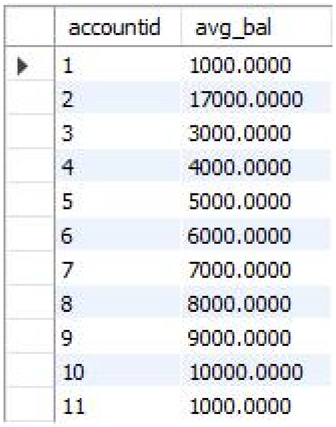
```sql
select (select sum(amount) from transactions
where transactiontype="deposit") - (select sum(amount)
from transactions where transactiontype="withdrawal") as difference;
```



difference
11000

10. Write a SQL query to Calculate the average daily balance for each account over a specified period.

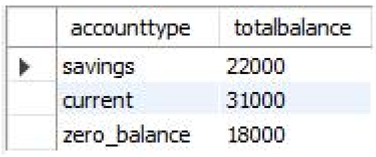
```sql
delimiter @@
create procedure avgbal(in val1 date, val2 date)
    begin
        select accountid,
        (datediff(val1, val2))*balance/datediff(val1, val2) as avg_bal
        from accounts;
    end@@
delimiter ;
call avgbal("2024-01-16", "2024-01-12");
```



accountid	avg_bal
1	1000.0000
2	17000.0000
3	3000.0000
4	4000.0000
5	5000.0000
6	6000.0000
7	7000.0000
8	8000.0000
9	9000.0000
10	10000.0000
11	1000.0000

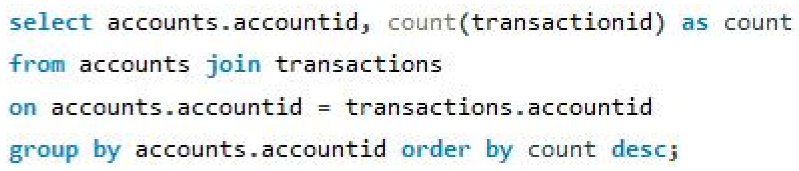
11. Calculate the total balance for each account type.

```sql
select accounttype, sum(balance) as totalbalance
from accounts group by accounttype;
```

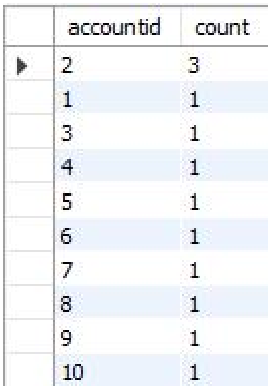


accounttype	totalbalance
savings	22000
current	31000
zero_balance	18000

12. Identify accounts with the highest number of transactions order by descending order.

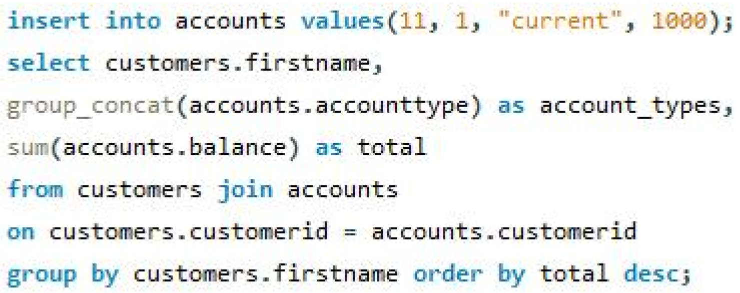


```
select accounts.accountid, count(transactionid) as count
from accounts join transactions
on accounts.accountid = transactions.accountid
group by accounts.accountid order by count desc;
```

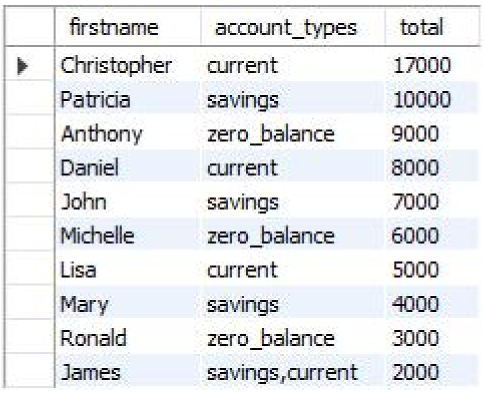


accountid	count
2	3
1	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1

13. List customers with high aggregate account balances, along with their account types.

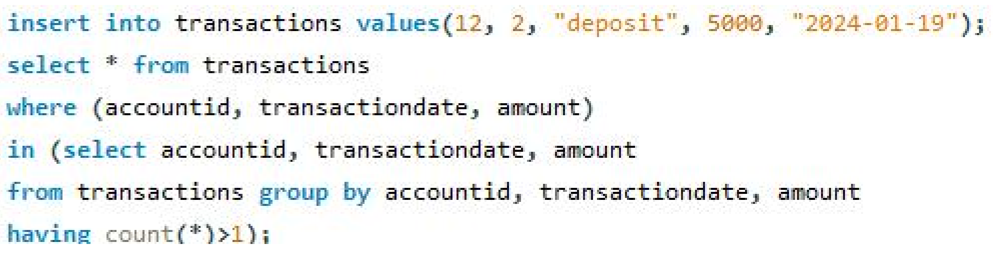


```
insert into accounts values(11, 1, "current", 1000);
select customers.firstname,
group_concat(accounts.accounttype) as account_types,
sum(accounts.balance) as total
from customers join accounts
on customers.customerid = accounts.customerid
group by customers.firstname order by total desc;
```

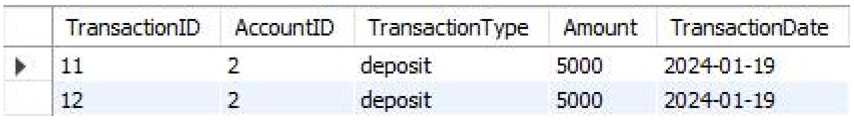


firstname	account_types	total
Christopher	current	17000
Patricia	savings	10000
Anthony	zero_balance	9000
Daniel	current	8000
John	savings	7000
Michelle	zero_balance	6000
Lisa	current	5000
Mary	savings	4000
Ronald	zero_balance	3000
James	savings,current	2000

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



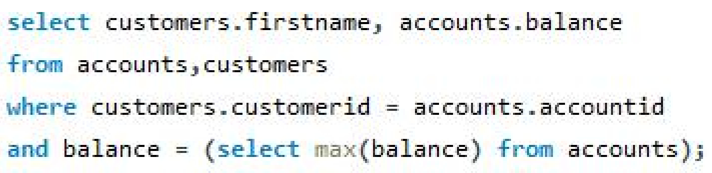
```
insert into transactions values(12, 2, "deposit", 5000, "2024-01-19");
select * from transactions
where (accountid, transactiondate, amount)
in (select accountid, transactiondate, amount
from transactions group by accountid, transactiondate, amount
having count(*)>1);
```



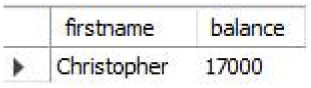
TransactionID	AccountID	TransactionType	Amount	TransactionDate
11	2	deposit	5000	2024-01-19
12	2	deposit	5000	2024-01-19

Tasks 4: Subquery and its type:

1. Retrieve the customer(s) with the highest account balance.

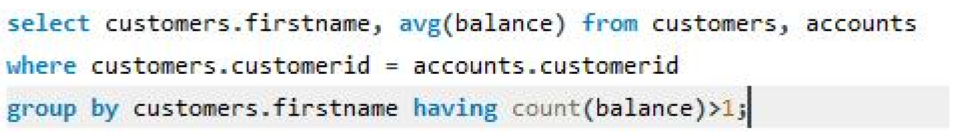


```
select customers.firstname, accounts.balance
from accounts,customers
where customers.customerid = accounts.accountid
and balance = (select max(balance) from accounts);
```

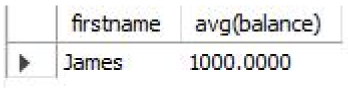


firstname	balance
Christopher	17000

2. Calculate the average account balance for customers who have more than one account.

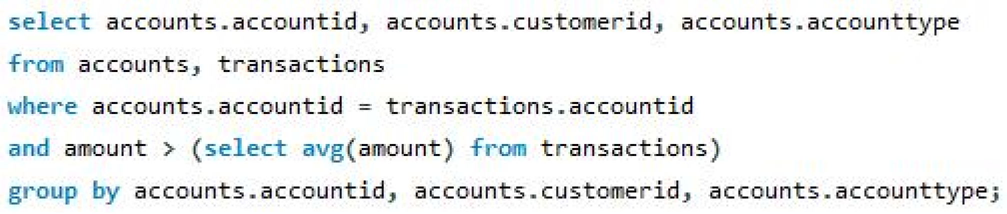


```
select customers.firstname, avg(balance) from customers, accounts
where customers.customerid = accounts.customerid
group by customers.firstname having count(balance)>1;
```

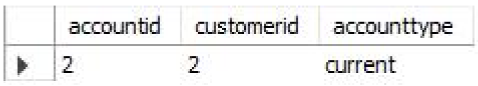


firstname	avg(balance)
James	1000.0000

3. Retrieve accounts with transactions whose amounts exceed the average transaction amount.



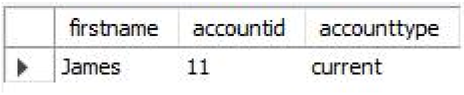
```
select accounts.accountid, accounts.customerid, accounts.accounttype
from accounts, transactions
where accounts.accountid = transactions.accountid
and amount > (select avg(amount) from transactions)
group by accounts.accountid, accounts.customerid, accounts.accounttype;
```



accountid	customerid	accounttype
2	2	current

4. Identify customers who have no recorded transactions.

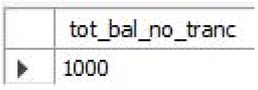
```sql
select customers.firstname, accounts.accountid, accounts.accounttype
from customers, accounts where customers.customerid = accounts.customerid
and accounts.accountid in (select accounts.accountid from accounts
where accounts.accountid not in (select accountid from transactions));
```



	firstname	accountid	accounttype
▶	James	11	current

5. Calculate the total balance of accounts with no recorded transactions.

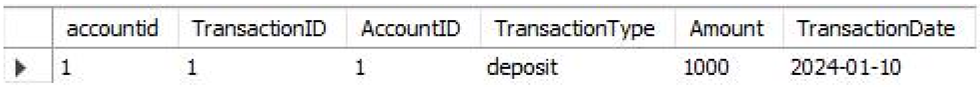
```sql
select sum(balance) as tot_bal_no_tranc from accounts
where accountid in (select accounts.accountid from accounts
where accounts.accountid not in (select accountid from transactions));
```



	tot_bal_no_tranc
▶	1000

6. Retrieve transactions for accounts with the lowest balance.

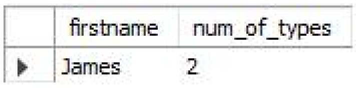
```sql
select accounts.accountid, transactions.* from accounts, transactions
where accounts.accountid = transactions.accountid
and accounts.balance = (select min(balance) from accounts);
```



	accountid	TransactionID	AccountID	TransactionType	Amount	TransactionDate
▶	1	1	1	deposit	1000	2024-01-10

7. Identify customers who have accounts of multiple types.

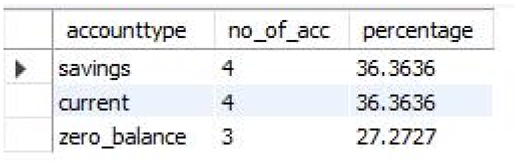
```sql
select firstname, count(accounttype) as num_of_types
from (select customers.firstname, accounts.accounttype
from customers, accounts where customers.customerid = accounts.customerid
group by customers.firstname, accounts.accounttype) as subquery
group by firstname having count(accounttype)>1;
```



	firstname	num_of_types
▶	James	2

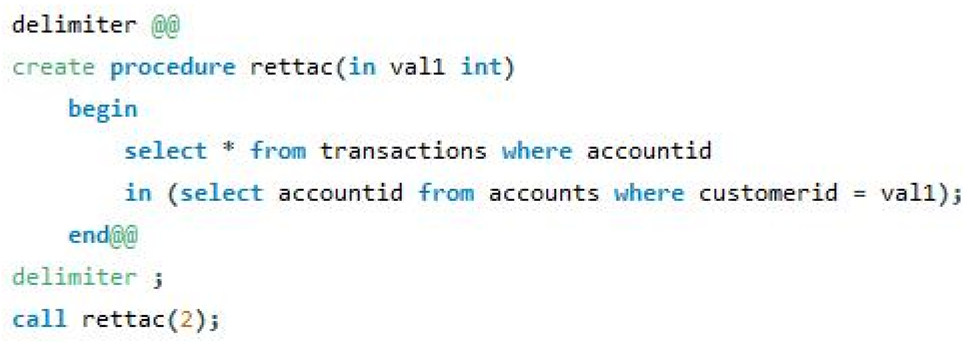
8. Calculate the percentage of each account type out of the total number of accounts.

```sql
select accounttype,
count(*) as no_of_acc, count(*)*100/(select count(*) from accounts)
as percentage from accounts group by accounttype;
```

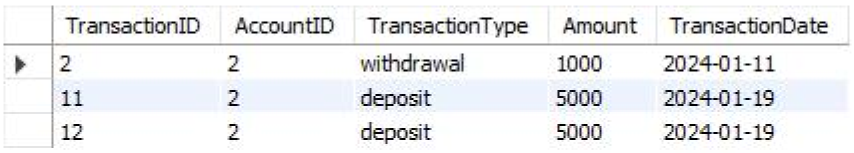


accounttype	no_of_acc	percentage
savings	4	36.3636
current	4	36.3636
zero_balance	3	27.2727

9. Retrieve all transactions for a customer with a given customer_id.

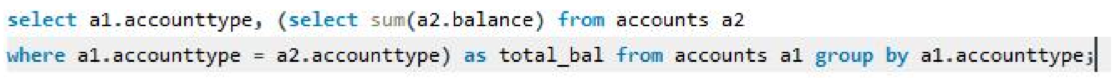


```
delimiter @@
create procedure rettac(in val1 int)
    begin
        select * from transactions where accountid
        in (select accountid from accounts where customerid = val1);
    end@@
delimiter ;
call rettac(2);
```

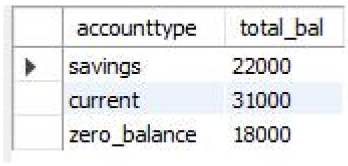


TransactionID	AccountID	TransactionType	Amount	TransactionDate
2	2	withdrawal	1000	2024-01-11
11	2	deposit	5000	2024-01-19
12	2	deposit	5000	2024-01-19

10. Calculate the total balance for each account type, including a subquery within the SELECT clause.



```
select a1.accounttype, (select sum(a2.balance) from accounts a2
where a1.accounttype = a2.accounttype) as total_bal from accounts a1 group by a1.accounttype;
```



accounttype	total_bal
savings	22000
current	31000
zero_balance	18000