Project Summary

ABC bank is one of the largest banks in the world. We recently completed a marketing campaign on bank credit and deposit products to customers. The management team is inclined to establish a database to store the entire marketing campaign data for the purpose of structuring the marketing campaign to properly fit our stakeholders' needs. With the data, we can establish a better marketing strategy to enhance revenues. During this project, we will set business rules between entities, a glossary, and data questions to improve our marketing campaign.

Stakeholder list

CEO and managers:
    Upper management makes decisions to help the company succeed.
sales team
    Customer facing team. Order product and service for customers.
Customer:
    Clients who do business with the bank.
Engineers:
    Department which handles technical issues and builds the database.
Investor:
    People who invest funds in the company.
Government:
    Government(regulation, interest rate, tax) has a big impact on the bank.
Suppliers:
    Provide office supplies for the bank.
Business partner:
    People or groups who have business relationships with the bank.

Glossary

Term-deposit(CD) is a type of deposit agreed with a financial institution which is locked within a certain time period and the financial institution would guarantee a fixed **interest rate** as return.

Account info: In account info, account type has credit product or deposit product.

Credit product: credit product means any product customer needs to make payment to the bank as such(loans and credit card).

Deposit product: deposit product is a type of account that allows customers to place funds in financial institutions for safekeeping, and banks might offer an interest rate of return depending on which deposit product.

Department: department **description** includes all of the department such as investment, online, branch, business in the bank

Transaction: This entity includes what type of transaction such as withdraw or deposit customer made with the date and amount of transaction.

Customer: Customer is a person who has an account with either credit or debit with the bank.

Employee: employee is a person who works in the bank in any department, **tenure** means how many years have this employee worked in the bank.

Manager: manager is an employee who manages other employees.

Loans: Loan **type** includes mortgage, auto, business and personal.

Terms: Term for loans means how many months the customer has to pay it off, term for saving product(term-deposit) means how many months until the agreement ends.

office: office includes the address, operation hours and rent expense for this office.

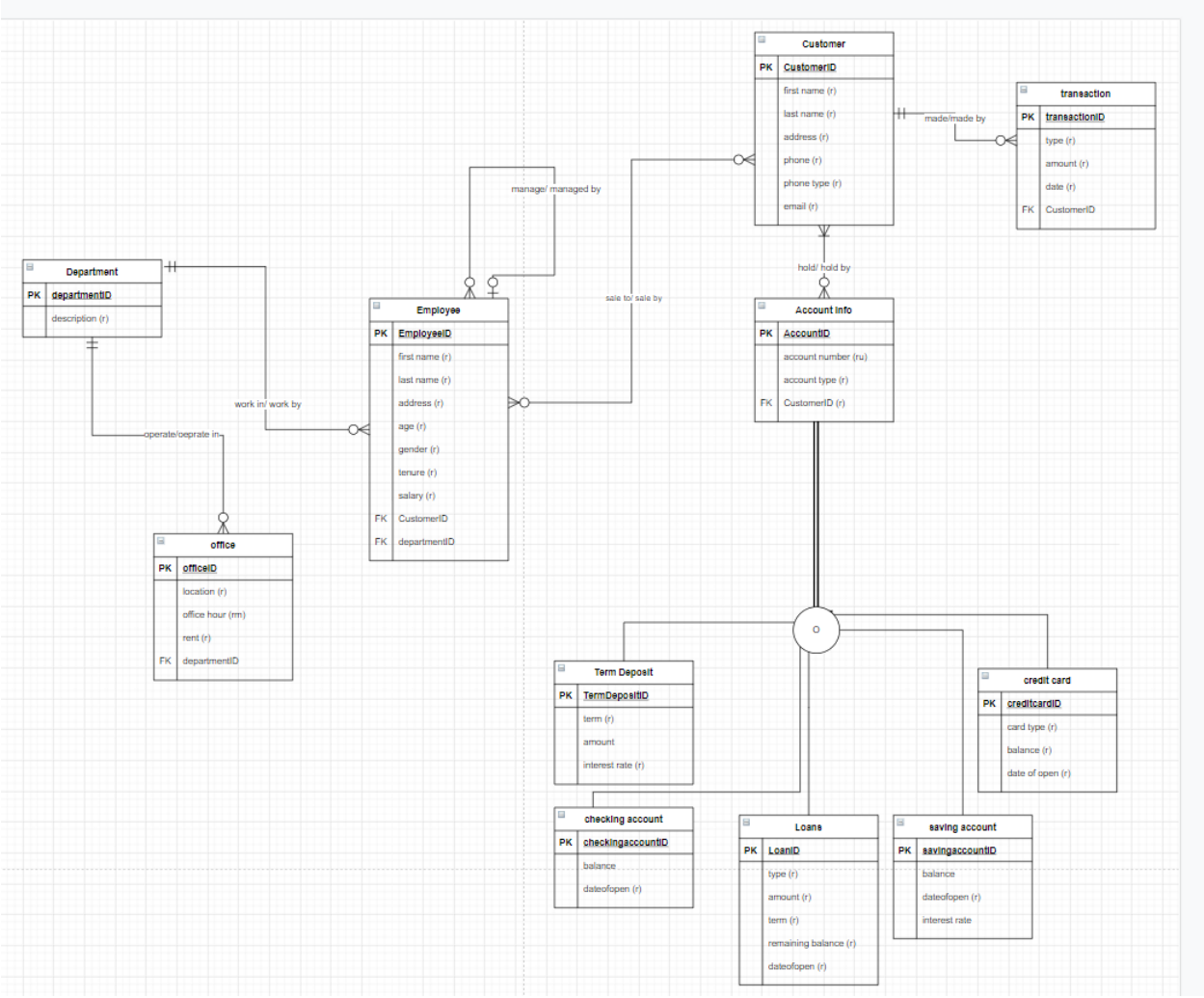
Account Info: Account info includes account type(credit or debit), the account number, when the account opened and the balance in the account.

Business rule

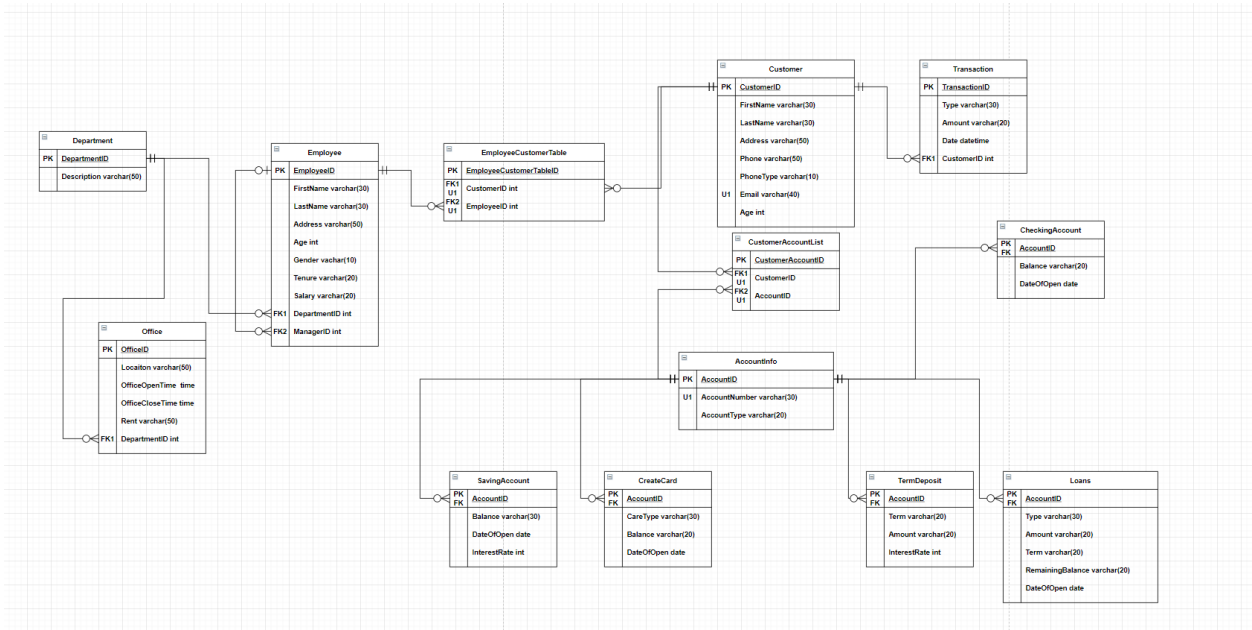
- Each customer can have one or more credit and deposit products(term deposit, credit card, loans)
- Each customer can make none or more than one transaction.
- Each employee can open an account for none or more than one customer.
- Account has to have at least one or more than one customer as account owner
- Employee can on can only work in one department
- Every office only has one department.

- Employees only work in one department.
- Each customer must have one or more either credit or deposit accounts with the bank.
- Each transaction can only be made by one customer.
- Employees can either have only one manager or no manager.
- Manager can either have zero, one, or many employees.
- Term deposit's term can be from 1 month to 5 years on a monthly basis.
- Customers can only do transactions when an account is opened.

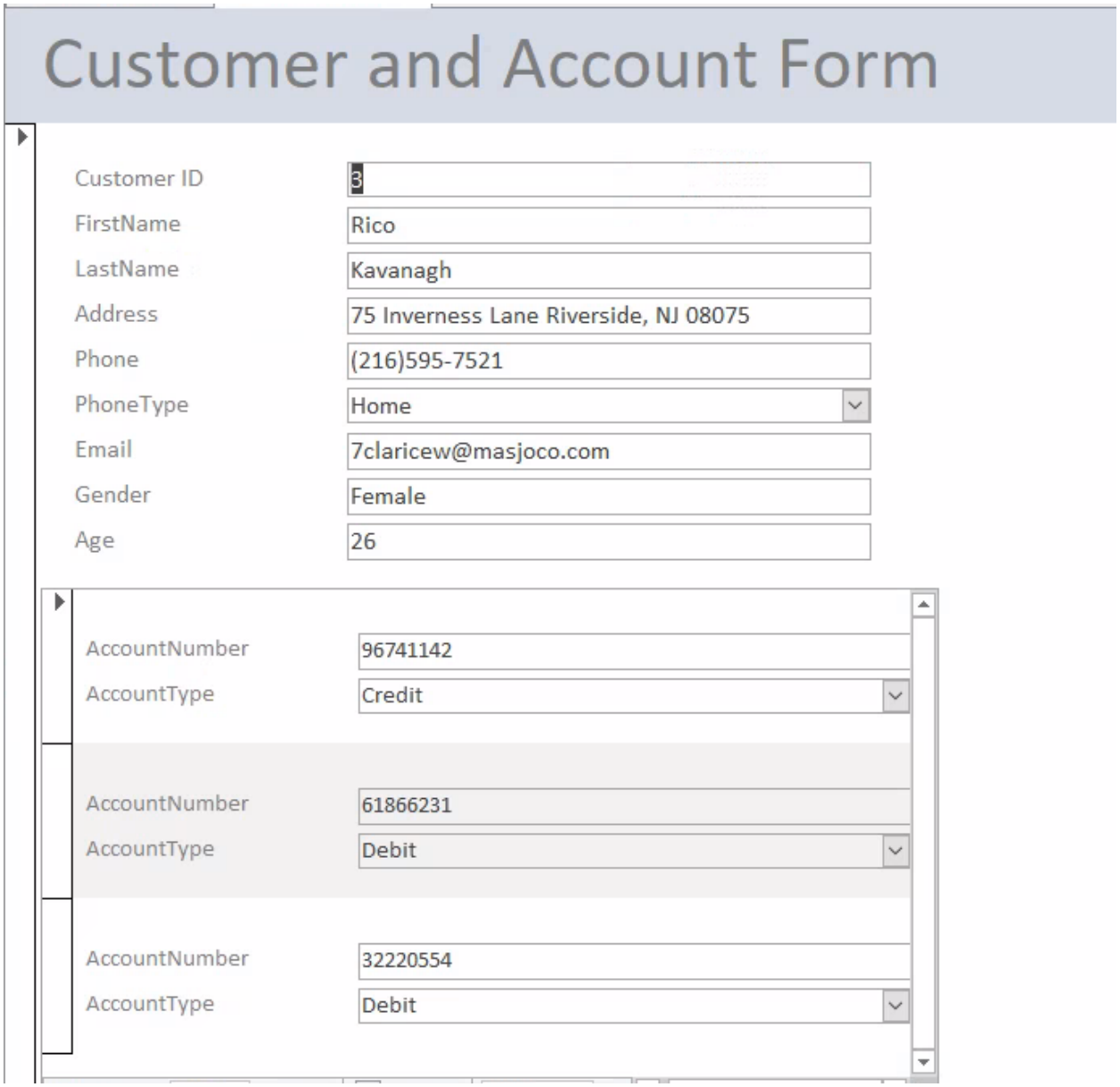
Conceptual Model



Logical Model



Data Entry Form



Data questions

Q1: What are the top 3 best performance employees?

shows the top 3 employees helped most customers.

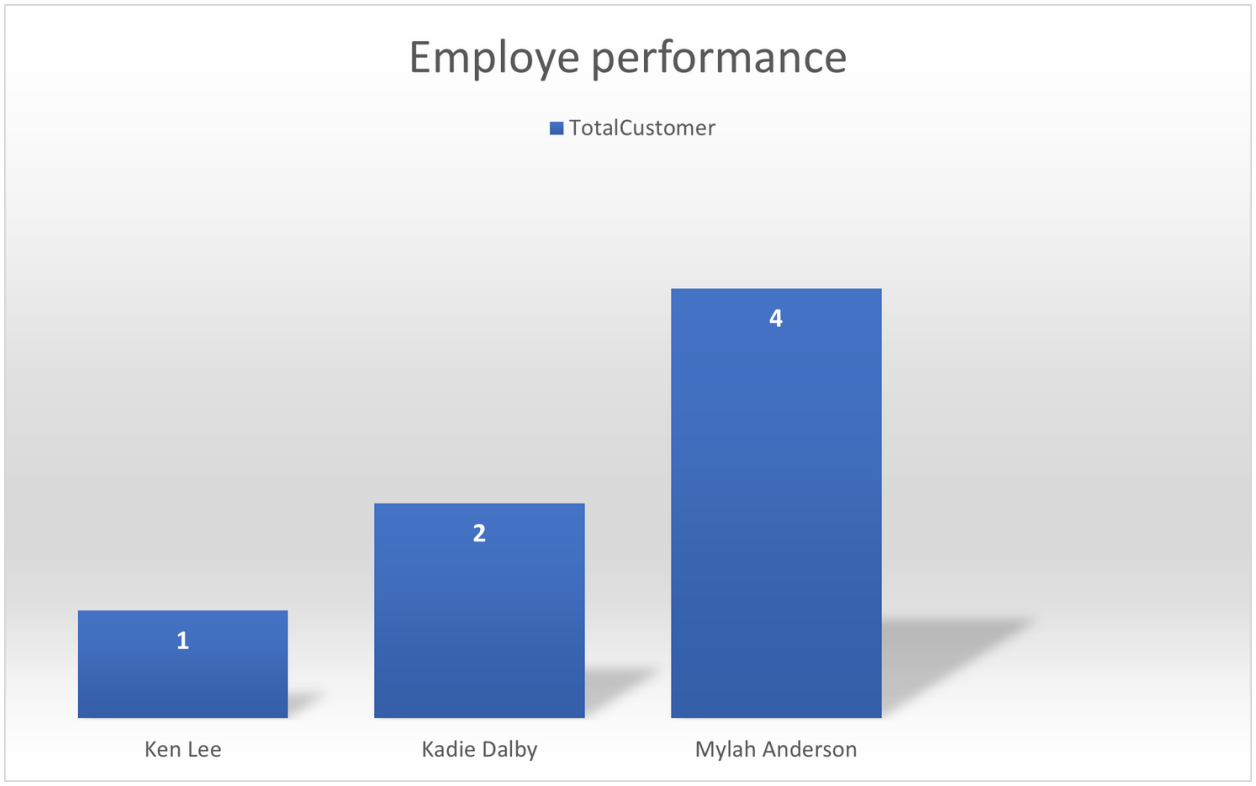
```
-- show each employee helped how many customer?
CREATE OR ALTER VIEW EmployeePerformance AS
SELECT CONCAT(e.FirstName, ' ', e.LastName) as EmployeeName,  COUNT(*) as TotalCustomer, DENSE_RANK() OVER(ORDER BY COUNT(*) DESC) as rank
FROM Employee e
join EmployeeCustomerTable ec
on e.EmployeeID = ec.EmployeeID
GROUP BY CONCAT(e.FirstName, ' ', e.LastName)
go

-- show the result of top 3 best performance employees
CREATE OR ALTER VIEW question1 AS
SELECT * FROM EmployeePerformance WHERE rank <= 3
go
SELECT * FROM question1 ORDER BY TotalCustomer DESC
GO
```

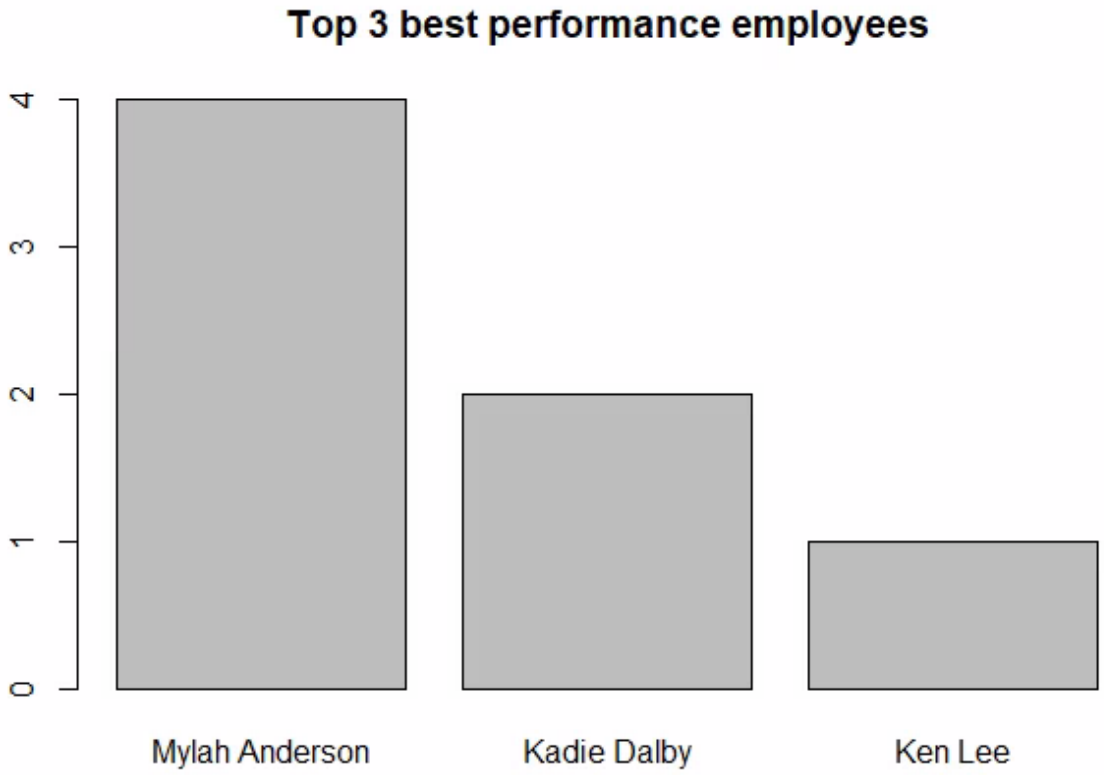
Top 3 employee performance

Employee Name	Total Customer
Mylah Anderson	4
Kadie Dalby	2
Ken Lee	1

Monday, September 13, 2021 Page 1 of 1



#using R to create this graph



Q2: How many customers have both a credit and debit product with the bank?

shows customers have both credit and debit products with the bank.

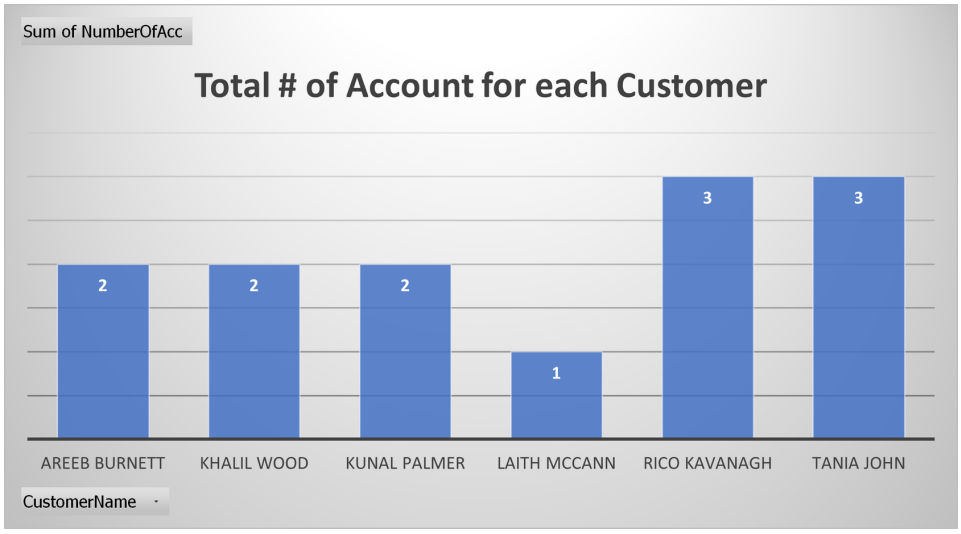
```
--#2:
-- show which customer have both debit and credit account, 4 customers have both account out of 6 customers
CREATE OR ALTER VIEW both_acc AS
SELECT CustomerName, SUM(NumberOfAcc) as TotalAcc
FROM TotalCustomerAcc
GROUP BY CustomerName
HAVING COUNT(*) > 1
GO
SELECT * FROM both_acc
GO
```

Customer with both credit and debit product

Customer Name	Total # of Account
Khalil Wood	2
Kunal Palmer	2
Rico Kavanagh	3
Tania John	3

Monday, September 13, 2021 Page 1 of 1

This graph shows all of the customers having any accounts with the bank



Q3: What is top 3 employee salary

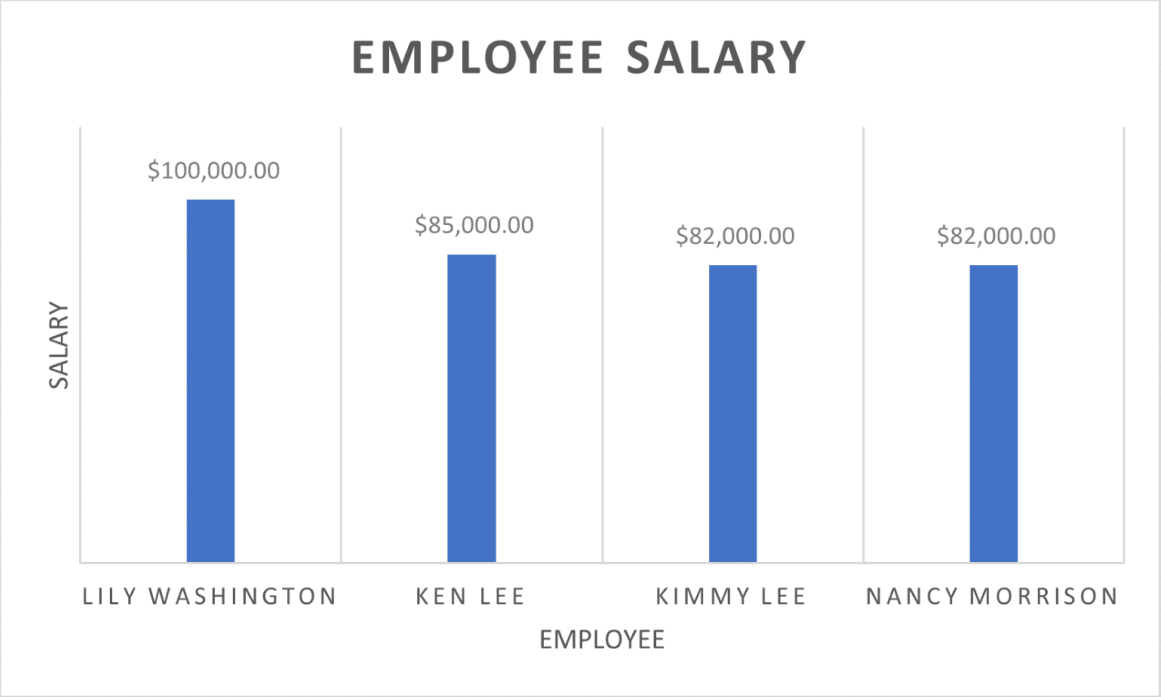
shows the top 3 employee salaries.

```
--3: top 3 salary employee
-- return the top 3 salary employees. using dense rank windown function to create rank for each employee based on their salary. if tie, they share the same rank and not skip the next rank level.
CREATE OR ALTER VIEW question3 AS
SELECT EmployeeName, Salary
FROM
(SELECT CONCAT(FirstName, ' ', LastName) as EmployeeName, DENSE_RANK() OVER(ORDER BY CAST(Salary AS int) DESC) as rank, Salary
FROM employee) t1
WHERE rank <= 3
GO
SELECT * FROM question3
GO
```

Top 3 Employee Salary

Employee Name	Salary
Lily Washington	100000
Ken Lee	85000
Kimmy Lee	82000
Nancy Morrison	82000

Monday, September 13, 2021 Page 1 of 1



Q4:What is the expense for each department monthly?

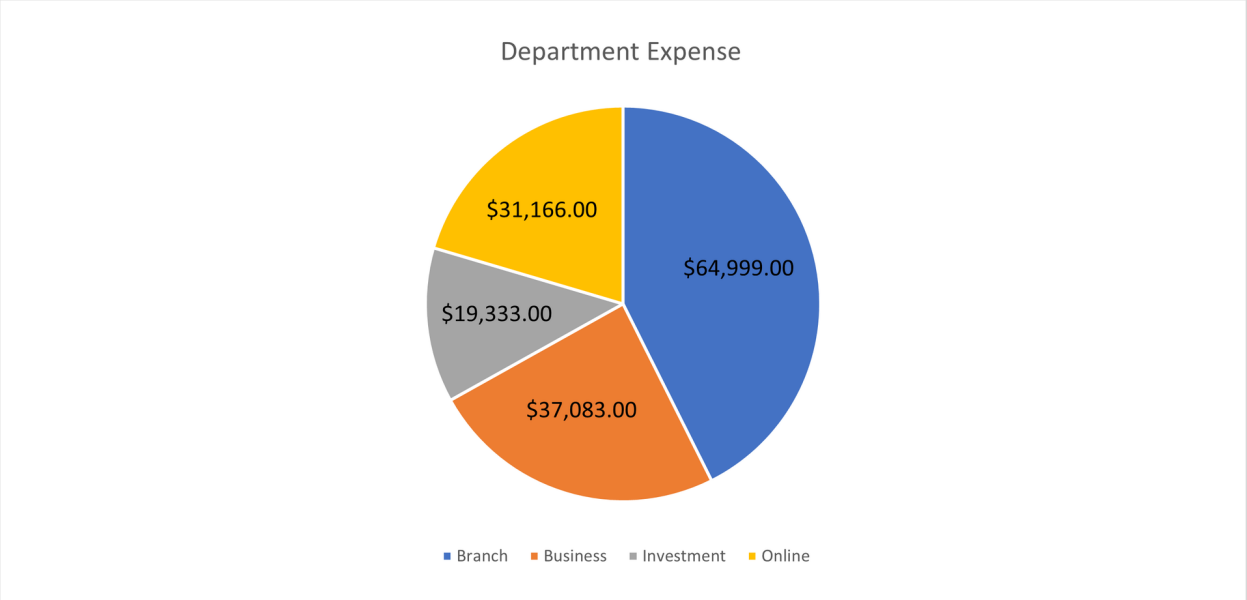
shows the expense for each department.

```
--4: what is the expense for different department monthly
CREATE OR ALTER VIEW question4 AS
SELECT Description as Department, SUM(MonthlySalary+rent) as Expense FROM
(SELECT CONCAT(FirstName, ' ', LastName) as EmployeeName, CAST(Salary AS int)/12 as MonthlySalary, rent, Description
FROM employee e
JOIN Department d on e.DepartmentID = d.DepartmentID
JOIN Office o on d.DepartmentID = o.DepartmentID) t1
GROUP BY Description
GO
SELECT * FROM question4
GO
```

Department expense

Department	Expense
Branch	$64,999.00
Business	$37,083.00
Investment	$19,333.00
Online	$31,166.00

Monday, September 13, 2021 Page 1 of 1



Q5: How many deposit transactions are made each week after opening a deposit account ?

shows how many transactions were made for each customer.

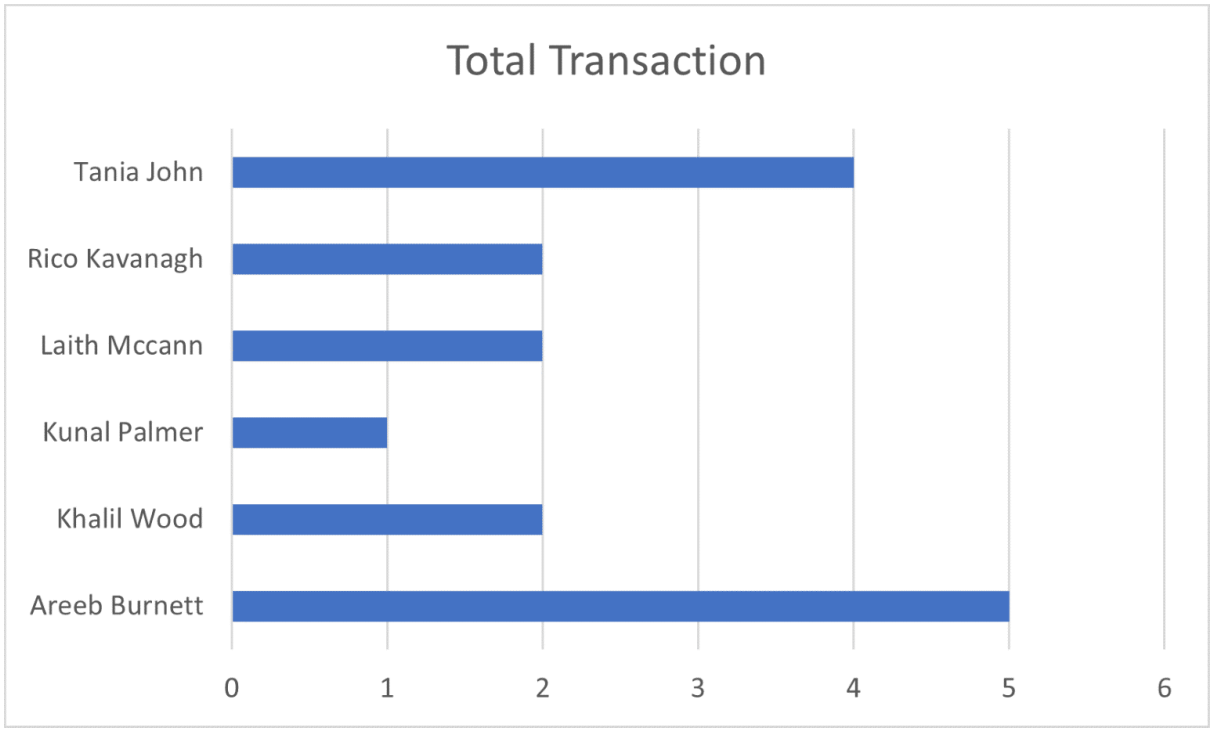
```
--5: How many deposit transaction are made after opening account

CREATE OR ALTER VIEW question5 AS
SELECT CONCAT(FirstName, ' ', LastName) AS CustomerName, COUNT(*) AS TotalTrans FROM BankTransaction bt
JOIN Customer c
ON bt.CustomerID = c.CustomerID
GROUP BY CONCAT(FirstName, ' ', LastName)
GO
SELECT * FROM question5
GO
```

Total trasanction made after opening account

Customer Name	Total # of Transactions
Areeb Burnett	5
Khalil Wood	2
Kunal Palmer	1
Laith Mccann	2
Rico Kavanagh	2
Tania John	4

Monday, September 13, 2021 Page 1 of 1



Raw Data Sample

Reflection

Before taking this class, the only thing I knew about SQL databases was SQL. I didn't know that a lot of pre-work is required before we actually build the database. During this class, I learned the importance of setting the conceptual and logical models correctly in the first place. When I was doing my logical model, I forgot to include account number info in my transaction table. So when I inserted data into the tables, I realized that I needed to edit it by adding back the attribute. Also, I realized that business rule was one of the key elements when we were building a database. When I was creating my conceptual model, I had looked back at business rules to complete it. After resolving these problems, I couldn't believe that I actually built a database. If I could have spent more time on this project I would create more tables for accounts, department and office so I can have more data to analyze for any business problems.

Summary

During this project, I learned how to build a database from scratch. Also, I learned how to use SELECT, INSERT and different FUNCTION, VIEW and STORED PROCEDURE statements to maintain the database. Moreover, it is essential to connect the database to other software like Excel, Access and RStudio to create visualization, report and data entry form to present the data to show our findings. Different softwares have different purposes, I am glad that I have the opportunity to try with these softwares.

Physical Database Design

```
-- drop all procedures
DROP PROCEDURE IF EXISTS ChangeAccountNumber;
go

-- drop all the views
DROP VIEW IF EXISTS TotalCustomerAcc;
go

DROP VIEW IF EXISTS EmployeePerformance;
go
DROP VIEW IF EXISTS question1;
go
DROP VIEW IF EXISTS question2;
go
DROP VIEW IF EXISTS both_acc;
go
DROP VIEW IF EXISTS question3;
go
DROP VIEW IF EXISTS question4;
go
DROP VIEW IF EXISTS question5;
go

-- drop all tables in reverse order of their dependencies
DROP TABLE IF EXISTS CheckingAccount;
go

DROP TABLE IF EXISTS Loans;
go

DROP TABLE IF EXISTS TermDeposit;
go
```

```
DROP TABLE IF EXISTS CreditCard;
go

DROP TABLE IF EXISTS SavingAccount;
go

DROP TABLE IF EXISTS CustomerAccountList;
go

DROP TABLE IF EXISTS BankTransaction;
go

DROP TABLE IF EXISTS AccountInfo;
go

DROP TABLE IF EXISTS Office;
go

DROP TABLE IF EXISTS EmployeeCustomerTable;
go

DROP TABLE IF EXISTS Employee;
go

DROP TABLE IF EXISTS Department;
go

DROP TABLE IF EXISTS Customer;
go

-- create all tables in order of their dependencies
--Create Customer table
CREATE TABLE Customer
```

```
(
	-- Columns in Customer table
	CustomerID int NOT NULL IDENTITY,
	FirstName varchar(30) NOT NULL,
	LastName varchar(30) NOT NULL,
	Address varchar(50) NOT NULL,
	Phone varchar(50) NOT NULL,
	PhoneType varchar(50) NOT NULL,
	Email varchar(40) NOT NULL,
	Gender varchar(10) NOT NULL,
	Age int,
	-- Constraint for Customer table
	CONSTRAINT Customer_PK PRIMARY KEY (CustomerID),
	CONSTRAINT Customer_U1 UNIQUE (Email)
);
go

--Create Department table
CREATE TABLE Department
(
	-- Columns for Department table
	DepartmentID int NOT NULL IDENTITY,
	Description varchar(50),
	-- Constraint for Department table
	CONSTRAINT Department_PK PRIMARY KEY (DepartmentID),
);
go


--Create Employee table
CREATE TABLE Employee
(
	--Columns for Employee table
	EmployeeID int NOT NULL IDENTITY,
```

```
	FirstName varchar(30) NOT NULL,
	LastName varchar(30) NOT NULL,
	Address varchar(50),
	Age int,
	Gender varchar(10),
	Tenure varchar(20),
	Salary varchar(20),
	DepartmentID int,
	ManagerID int,
	-- Constraint for Employee table
	CONSTRAINT Employee_PK PRIMARY KEY (EmployeeID),
	CONSTRAINT Employee_FK1 FOREIGN KEY (DepartmentID) REFERENCES
Department (DepartmentID),
	CONSTRAINT Employee_FK2 FOREIGN KEY (ManagerID) REFERENCES Employee
(EmployeeID)
);
go

--Create EmployeeCustomerTable
CREATE TABLE EmployeeCustomerTable
(
	--Columns for EmployeeCustomer table
	EmployeeCustomerTableID int NOT NULL IDENTITY,
	CustomerID int NOT NULL,
	EmployeeID int NOT NULL,
	--Constraint for EmployeeCustomer table
	CONSTRAINT EmployeeCustomerTable_PK PRIMARY KEY
(EmployeeCustomerTableID),
	CONSTRAINT EmployeeCustomerTable_FK1 FOREIGN KEY (CustomerID)
REFERENCES Customer (CustomerID),
	CONSTRAINT EmployeeCustomerTable_FK2 FOREIGN KEY (EmployeeID)
REFERENCES Employee (EmployeeID)
);
go
```

```
--Create Office table
CREATE TABLE Office
(
	--Columns for Office table
	OfficeID int NOT NULL IDENTITY,
	Location varchar(50) NOT NULL,
	OfficeOpenTime time NOT NULL,
	OfficeCloseTIme time NOT NULL,
	Rent varchar(50),
	DepartmentID int,
	--Constraint for Office table
	CONSTRAINT Office_PK PRIMARY KEY (OfficeID),
	CONSTRAINT Office_FK1 FOREIGN KEY (DepartmentID) REFERENCES Department
(DepartmentID),
);
go

--Create AccountInfo table
CREATE TABLE AccountInfo
(
	--columns for Accountinfo table
	AccountID int NOT NULL IDENTITY,
	AccountNumber varchar(30) NOT NULL,
	AccountType varchar(20) NOT NULL,
	--Constraint for Accountinfo table
	CONSTRAINT AccountInfo_PK PRIMARY KEY (AccountID),
	CONSTRAINT AccountInfo_U1 UNIQUE (AccountNumber)
);
go

--Create Banktransaction table
CREATE TABLE BankTransaction
(
```

```
	--columns for Banktransaction table
	TransactionID int NOT NULL IDENTITY,
	Type varchar(30) NOT NULL,
	Amount varchar(20) NOT NULL,
	Date datetime,
	CustomerID int NOT NULL,
	AccountID int NOT NULL,
	--Constraint for Banktransaction table
	CONSTRAINT Transaction_PK PRIMARY KEY (TransactionID),
	CONSTRAINT Transaction_FK1 FOREIGN KEY (CustomerID) REFERENCES
Customer (CustomerID),
	CONSTRAINT Transaction_FK2 FOREIGN KEY (AccountID) REFERENCES
AccountInfo (AccountID)
);
go


--Create CustomerAccount Table
CREATE TABLE CustomerAccountList
(
	--columns for CustomerAccount table
	CustomerAccountID int NOT NULL IDENTITY,
	CustomerID int NOT NULL,
	AccountID int NOT NULL,
	--Constraint for CustomerAccount table
	CONSTRAINT CustomerAccountList_PK PRIMARY KEY (CustomerAccountID),
	CONSTRAINT CustomerAccountList_FK1 FOREIGN KEY (CustomerID) REFERENCES
Customer (CustomerID),
	CONSTRAINT CustomerAccountList_FK2 FOREIGN KEY (AccountID) REFERENCES
AccountInfo (AccountID)
);
go

--Create SavingAccount table
```

```
CREATE TABLE SavingAccount
(
	--columns for SavingAccount table
	AccountID int NOT NULL,
	Balance varchar(30) NOT NULL,
	DateOfOpen date NOT NULL,
	InterestRate varchar(30) NOT NULL,
	--Constraint for SavingAccount table
	CONSTRAINT AccountInfo_PK2 PRIMARY KEY (AccountID),
	CONSTRAINT SavingAccount_FK1 FOREIGN KEY (AccountID) REFERENCES
AccountInfo (AccountID)
);
go

--Create CreditCard table
CREATE TABLE CreditCard
(
	--columns for CreditCard table
	AccountID int NOT NULL,
	CardType varchar(30) NOT NULL,
	Balance varchar(20) NOT NULL,
	DateOfOpen date NOT NULL,
	--Constraint for CreditCard table
	CONSTRAINT AccountInfo_PK3 PRIMARY KEY (AccountID),
	CONSTRAINT CreditCard_FK1 FOREIGN KEY (AccountID) REFERENCES
AccountInfo (AccountID)
);
go

--Create TermDeposit table
CREATE TABLE TermDeposit
(
	--columns for TermDeposit table
	AccountID int NOT NULL,
```

```
	Term varchar(30) NOT NULL,
	Amount varchar(20) NOT NULL,
	InterestRate varchar(30) NOT NULL,
	--Constraint for TermDeposit table
	CONSTRAINT AccountInfo_PK4 PRIMARY KEY (AccountID),
	CONSTRAINT TermDeposit_FK1 FOREIGN KEY (AccountID) REFERENCES
AccountInfo (AccountID)
);
go

--Create loans table
CREATE TABLE Loans
(
	--columns for loans table
	AccountID int NOT NULL,
	Type varchar(30) NOT NULL,
	Amount varchar(20) NOT NULL,
	Term varchar(20) NOT NULL,
	RemainingBalance varchar(20),
	DateOfOpen date,
	--Constraint for loans table
	CONSTRAINT AccountInfo_PK5 PRIMARY KEY (AccountID),
	CONSTRAINT Loans_FK1 FOREIGN KEY (AccountID) REFERENCES AccountInfo
(AccountID)
);
go

-- Create Checking Account table
CREATE TABLE CheckingAccount
(
	--columns for Checking Account table
	AccountID int NOT NULL,
	Balance varchar(20),
	DateOfOpen date,
```

```
	--Constraint for CheckingAccount table
	CONSTRAINT AccountInfo_PK6 PRIMARY KEY (AccountID),
	CONSTRAINT CheckingAccount_FK1 FOREIGN KEY (AccountID) REFERENCES
AccountInfo (AccountID)
);
go

-- insert records into tables; note using scalar SELECT statements to avid hard-coding
IDENTITY values
INSERT dbo.Customer (FirstName, LastName, Address, Phone, PhoneType, Email, Gender,
Age)
VALUES	('Tania','John','44 Prince St. Dyersburg, TN 38024', '(559)891-4794', 'Cell',
'wphotoshop.ahmedo@kalayya.com', 'Female', '35'),
		('Laith','Mccann','30 South Hanover Street Richardson, TX 75080',
'(252)467-5273', 'Cell', 'gbrandon.stewa@halumail.com', 'Female', '41'),
		('Rico','Kavanagh','75 Inverness Lane Riverside, NJ 08075', '(216)595-7521',
'Cell', '7claricew@masjoco.com', 'Female', '26'),
		('Khalil','Wood','2 Brookside Street Dearborn, MI 48124', '(330)571-6856', 'Home',
'pjoymu@neaeo.com', 'Male', '54'),
		('Areeb','Burnett','44 Summerhouse Ave. Vernon Rockville, CT 06066',
'(773)890-7810', 'Home', 'malaam.os@vietkevin.com', 'Male', '76'),
		('Kunal','Palmer','674 West Glen Eagles Court Sacramento, CA 95820',
'(478)287-4012', 'Home', 'faliahmed205g@hotmail.red', 'Male', '64'),
		('Christopher','Quinn','6827 West Glen Eagles Court Sacramento, CA 95820',
'(478)281-4212', 'Home', 'feliahmed205g@hotmail.red', 'Male', '45')
		;
go


-- insert records into Deparment table
INSERT dbo.Department (Description)
VALUES	('Investment'), ('Online'),
		('Business'), ('Branch');
go
```

```
--insert records into Employee table
INSERT dbo.Employee(FirstName, LastName, Address, Age, Gender, Tenure, Salary,
DepartmentID, ManagerID)
VALUES          ('Kadie', 'Dalby', '8699 East Proctor St. Taylors, SC 29687', '34', 'Male', '5',
'80000', (SELECT DepartmentID FROM Department WHERE Description = 'Branch'), NULL),
                ('Oscar', 'Daly', '27 East Buttonwood Ave. Morton Grove, IL 60053', '35', 'Male',
'6', '75000', (SELECT DepartmentID FROM Department WHERE Description = 'Business'),
NULL),
                ('Mylah', 'Anderson', '8699 East Proctor St. Taylors, SC 29687', '32', 'Male', '6',
'75000', (SELECT DepartmentID FROM Department WHERE Description = 'Branch'), NULL),
                ('Lisa', 'White', '8699 East Proctor St. Taylors, SC 29687', '42', 'Male', '2', '60000',
(SELECT DepartmentID FROM Department WHERE Description = 'Branch'), NULL),
                ('Ken', 'Lee', '8699 East Proctor St. Taylors, SC 29687', '35', 'Male', '3', '85000',
(SELECT DepartmentID FROM Department WHERE Description = 'Branch'), NULL),
                ('Joe', 'Hayes', '829 Miller St. Oak Forest, IL 60452', '43', 'Male', '3', '76000',
(SELECT DepartmentID FROM Department WHERE Description = 'Online'), NULL),
                ('Lily', 'Washington', '2 Lyme St. Ankeny, IA 50023', '46', 'Male', '3', '100000',
(SELECT DepartmentID FROM Department WHERE Description = 'Investment'), NULL),
                ('Kimmy', 'Lee', '27 East Buttonwood Ave. Morton Grove, IL 60053', '25', 'Male',
'3', '82000', (SELECT DepartmentID FROM Department WHERE Description = 'Business'),
NULL),
                ('Nancy', 'Morrison', '829 Miller St. Oak Forest, IL 60452', '28', 'Male', '3', '82000',
(SELECT DepartmentID FROM Department WHERE Description = 'Online'), NULL)
                ;
go

-- Update the managerID. Kadie Dalby is Mylah Anderson's manager
UPDATE Employee
SET ManagerID = (SELECT EmployeeID FROM Employee WHERE FirstName = 'Kadie' and
LastName = 'Dalby')
WHERE FirstName = 'Mylah' and LastName = 'Anderson'
GO
```

```
-- Update the managerID for business department
UPDATE Employee
SET ManagerID = (SELECT EmployeeID FROM Employee WHERE FirstName = 'Kimmy' and
LastName = 'Lee')
WHERE FirstName = 'Oscar' and LastName = 'Daly'
GO

-- Update the managerID for Online department
UPDATE Employee
SET ManagerID = (SELECT EmployeeID FROM Employee WHERE FirstName = 'Nancy' and
LastName = 'Morrison')
WHERE FirstName = 'Joe' and LastName = 'Hayes'
GO

--insert records into EmployeeCustomerTable
INSERT dbo.EmployeeCustomerTable(CustomerID, EmployeeID)
VALUES      ((SELECT CustomerID FROM Customer WHERE FirstName = 'Tania' and
LastName = 'John'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Mylah' and
LastName = 'Anderson')),
            ((SELECT CustomerID FROM Customer WHERE FirstName = 'Laith' and
LastName = 'Mccann'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Mylah'
and LastName = 'Anderson')),
            ((SELECT CustomerID FROM Customer WHERE FirstName = 'Rico' and
LastName = 'Kavanagh'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Kadie'
and LastName = 'Dalby')),
            ((SELECT CustomerID FROM Customer WHERE FirstName = 'Khalil' and
LastName = 'Wood'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Mylah' and
LastName = 'Anderson')),
            ((SELECT CustomerID FROM Customer WHERE FirstName = 'Areeb' and
LastName = 'Burnett'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Kadie'
and LastName = 'Dalby')),
```

```
                ((SELECT CustomerID FROM Customer WHERE FirstName = 'Kunal' and
LastName = 'Palmer'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Mylah'
and LastName = 'Anderson')),
                ((SELECT CustomerID FROM Customer WHERE FirstName = 'Christopher' and
LastName = 'Quinn'), (SELECT EmployeeID FROM Employee WHERE FirstName = 'Ken' and
LastName = 'Lee'))
;
go

--insert records into Office table
INSERT dbo.Office (Location, OfficeOpenTime, OfficeCloseTIme, Rent, DepartmentID)
VALUES      ('779 Hilltop Rd. Woodside, SC 29687', '08:00:00', '18:00:00', '10000', (SELECT
DepartmentID FROM Department WHERE Description = 'Branch')),
                ('745 West Sutor Dr. Hyde Park, MI 48136', '08:00:00', '18:00:00', '12000',
(SELECT DepartmentID FROM Department WHERE Description = 'Business')),
                ('9226 Front St. Lansing, MI 48910', '08:00:00', '18:00:00', '9000', (SELECT
DepartmentID FROM Department WHERE Description = 'Online')),
                ('158 Lyme Lane Grandville, MI 49418', '08:00:00', '18:00:00', '11000', (SELECT
DepartmentID FROM Department WHERE Description = 'Investment'))
                ;
go

-- return scope_identity() after a insert statement
CREATE OR ALTER FUNCTION dbo.ReturnIdentity()
RETURNS int
AS BEGIN
        DECLARE @return int
        set @return = SCOPE_IDENTITY()
        RETURN @return
END
go

-- Insert data to super/sub type and use ReturnIdentity function to insert AccountID in sub type
INSERT dbo.AccountInfo (AccountNumber, AccountType)
```

```
VALUES        ('23133526', 'Debit')
INSERT dbo.SavingAccount (AccountID, Balance, DateOfOpen, InterestRate)
VALUES        (dbo.ReturnIdentity(), '1030', '06/25/2021', 0.01);

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES        ('87381716', 'Debit')
INSERT dbo.SavingAccount (AccountID, Balance, DateOfOpen, InterestRate)
VALUES        (dbo.ReturnIdentity(), '2000', '06/25/2021', 0.01);

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES        ('96741142', 'Credit')
INSERT dbo.CreditCard (AccountID, CardType, Balance, DateOfOpen)
VALUES        (dbo.ReturnIdentity(), 'Personal', '0', '06/25/2021');

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES        ('27382698', 'Debit')
INSERT dbo.CheckingAccount (AccountID, Balance, DateOfOpen)
VALUES        (dbo.ReturnIdentity(), '1900', '06/25/2021');

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES        ('47237567', 'Debit')
INSERT dbo.CheckingAccount (AccountID, Balance, DateOfOpen)
VALUES        (dbo.ReturnIdentity(), '1020', '06/25/2021');

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES        ('77461251', 'Credit')
INSERT dbo.CreditCard (AccountID, CardType, Balance, DateOfOpen)
VALUES        (dbo.ReturnIdentity(), 'Business', '0', '06/25/2021');

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES        ('94734247', 'Credit')
INSERT dbo.Loans (AccountID, Type, Amount, Term, RemainingBalance, DateOfOpen)
VALUES        (dbo.ReturnIdentity(), 'Personal', '10000', '12', '10000', '06/25/2021');
```

```
INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES      ('61866231', 'Debit')
INSERT dbo.TermDeposit (AccountID, Term, Amount, InterestRate)
VALUES      (dbo.ReturnIdentity(), '12', '5000', 1.15);

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES      ('33115295', 'Credit')
INSERT dbo.Loans (AccountID, Type, Amount, Term, RemainingBalance, DateOfOpen)
VALUES      (dbo.ReturnIdentity(), 'Personal', '20000', '12', '20000', '06/25/2021')

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES      ('32220554', 'Debit')
INSERT dbo.TermDeposit (AccountID, Term, Amount, InterestRate)
VALUES      (dbo.ReturnIdentity(), '24', '30000', 2.00);

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES      ('20337370', 'Debit')
INSERT dbo.TermDeposit (AccountID, Term, Amount, InterestRate)
VALUES      (dbo.ReturnIdentity(), '36', '50000', 2.75);

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES      ('47827759', 'Debit')
INSERT dbo.TermDeposit (AccountID, Term, Amount, InterestRate)
VALUES      (dbo.ReturnIdentity(), '24', '20000', 2.15);

INSERT dbo.AccountInfo (AccountNumber, AccountType)
VALUES      ('38527312', 'Debit')
INSERT dbo.TermDeposit (AccountID, Term, Amount, InterestRate)
VALUES      (dbo.ReturnIdentity(), '12', '10000', 1.15);
go

--insert records into BankTransaction Table
INSERT dbo.BankTransaction (Type, Amount, Date, CustomerID, AccountID)
```

```
VALUES        ('deposit', '1030', '06/26/2021', (SELECT CustomerID from Customer where
FirstName = 'Tania' and LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE
AccountNumber = '23133526')),
              ('deposit', '2030', '06/27/2021', (SELECT CustomerID from Customer where
FirstName = 'Tania' and LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE
AccountNumber = '23133526')),
              ('deposit', '3030', '06/28/2021', (SELECT CustomerID from Customer where
FirstName = 'Tania' and LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE
AccountNumber = '23133526')),
              ('deposit', '2000', '06/26/2021', (SELECT CustomerID from Customer where
FirstName = 'Laith' and LastName = 'Mccann'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '87381716')),
              ('deposit', '4200', '06/27/2021', (SELECT CustomerID from Customer where
FirstName = 'Laith' and LastName = 'Mccann'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '87381716')),
              ('deposit', '1900', '06/26/2021', (SELECT CustomerID from Customer where
FirstName = 'Khalil' and LastName = 'Wood'), (SELECT AccountID FROM AccountInfo WHERE
AccountNumber = '27382698')),
              ('deposit', '100', '06/27/2021', (SELECT CustomerID from Customer where
FirstName = 'Khalil' and LastName = 'Wood'), (SELECT AccountID FROM AccountInfo WHERE
AccountNumber = '27382698')),
              ('deposit', '1020', '06/28/2021', (SELECT CustomerID from Customer where
FirstName = 'Areeb' and LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '47237567')),
              ('deposit', '1000', '06/27/2021', (SELECT CustomerID from Customer where
FirstName = 'Areeb' and LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '47237567')),
              ('deposit', '1320', '06/26/2021', (SELECT CustomerID from Customer where
FirstName = 'Areeb' and LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '47237567')),
              ('deposit', '1520', '06/28/2021', (SELECT CustomerID from Customer where
FirstName = 'Areeb' and LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '47237567')),
```

```
                ('deposit', '5000', '06/26/2021', (SELECT CustomerID from Customer where
FirstName = 'Rico' and LastName = 'Kavanagh'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '61866231')),
                ('deposit', '30000', '06/28/2021', (SELECT CustomerID from Customer where
FirstName = 'Rico' and LastName = 'Kavanagh'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '32220554')),
                ('deposit', '50000', '06/27/2021', (SELECT CustomerID from Customer where
FirstName = 'Areeb' and LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '20337370')),
                ('deposit', '20000', '06/26/2021', (SELECT CustomerID from Customer where
FirstName = 'Kunal' and LastName = 'Palmer'), (SELECT AccountID FROM AccountInfo
WHERE AccountNumber = '47827759')),
                ('deposit', '10000', '06/30/2021', (SELECT CustomerID from Customer where
FirstName = 'Tania' and LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE
AccountNumber = '38527312'))
                ;
go

--insert records into CustomerAccountTable
INSERT dbo.CustomerAccountList (CustomerID, AccountID)
VALUES          ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Tania' AND
LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'23133526')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Laith' AND
LastName = 'Mccann'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'87381716')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Rico' AND
LastName = 'Kavanagh'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'96741142')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Khalil' AND
LastName = 'Wood'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'27382698')),
```

```
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Areeb' AND
LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'47237567')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Kunal' AND
LastName = 'Palmer'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'77461251')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Tania' AND
LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'94734247')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Rico' AND
LastName = 'Kavanagh'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'61866231')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Khalil' AND
LastName = 'Wood'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'33115295')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Rico' AND
LastName = 'Kavanagh'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'32220554')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Areeb' AND
LastName = 'Burnett'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'20337370')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Kunal' AND
LastName = 'Palmer'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'47827759')),
                ((SELECT c.CustomerID FROM Customer c WHERE FirstName = 'Tania' AND
LastName = 'John'), (SELECT AccountID FROM AccountInfo WHERE AccountNumber =
'38527312'))
;
go

-- If customer's account has be compromised, we will need to change the account number
CREATE OR ALTER PROCEDURE ChangeAccountNumber (@CurrentAccountNumber
varchar(30), @NewAccountNumber varchar(30))
AS
```

```
BEGIN
	UPDATE AccountInfo SET AccountNumber = @NewAccountNumber
	WHERE  AccountNumber = @CurrentAccountNumber
END
GO
-- all of the account number before changing the account number
select * from AccountInfo
GO
-- changing the account number 23133526 to 12345678
EXEC ChangeAccountNumber '23133526', '12345678'
GO
-- all of the account number after changing the account number
select * from AccountInfo
GO
-- 5 data questions
--1: Top 3 best performance employees in different branches
-- show each employee helped how many customer.
CREATE OR ALTER VIEW EmployeePerformance AS
SELECT CONCAT(e.FirstName, ' ', e.LastName) as EmployeeName,  COUNT(*) as
TotalCustomer, DENSE_RANK() OVER(ORDER BY COUNT(*) DESC) as rank
FROM Employee e
join EmployeeCustomerTable ec
on e.EmployeeID = ec.EmployeeID
GROUP BY CONCAT(e.FirstName, ' ', e.LastName)
go

-- show the result of top 3 best performance employees
CREATE OR ALTER VIEW question1 AS
SELECT * FROM EmployeePerformance WHERE rank <= 3
go
SELECT * FROM question1 ORDER BY TotalCustomer DESC
GO

-- Create view to show how many account customer have in different type.
```

```
CREATE OR ALTER VIEW TotalCustomerAcc AS
SELECT DISTINCT CONCAT(Customer.FirstName, ' ', Customer.LastName) as
CustomerName, Customer.CustomerID
        , COUNT(AccountType) OVER(PARTITION BY Customer.CustomerID, AccountType) as
NumberOfAcc, AccountType
FROM Customer
JOIN CustomerAccountList ca
ON Customer.CustomerID = ca.CustomerID
JOIN AccountInfo ai
ON ai.AccountID = ca.AccountID
go
select * from TotalCustomerAcc
go

--#2:
-- show which customer have both debit and credit account, 4 customers have both account out
of 6 customers
CREATE OR ALTER VIEW both_acc AS
SELECT CustomerName, SUM(NumberOfAcc) as TotalAcc
FROM TotalCustomerAcc
GROUP BY CustomerName
HAVING COUNT(*) > 1
GO
SELECT * FROM both_acc
GO

-- show how many accounts does each customer have
CREATE OR ALTER VIEW question2 AS
SELECT CustomerName, SUM(NumberOfAcc) as TotalAcc from TotalCustomerAcc
GROUP BY CustomerName
GO
SELECT * FROM question2 ORDER BY TotalAcc DESC
GO
```

```
--3: top 3 salary employee
-- return the top 3 salary employees. using dense rank windown function to create rank for each employee based on their salary. if tie, they share the same rank and not skip the next rank level.
CREATE OR ALTER VIEW question3 AS
SELECT EmployeeName, Salary
FROM
(SELECT CONCAT(FirstName, ' ', LastName) as EmployeeName, DENSE_RANK()
OVER(ORDER BY CAST(Salary AS int) DESC) as rank, Salary
FROM employee) t1
WHERE rank <= 3
GO
SELECT * FROM question3
GO

--4: what is the expense for different department monthly
CREATE OR ALTER VIEW question4 AS
SELECT Description as Department, SUM(MonthlySalary+rent) as Expense FROM
(SELECT CONCAT(FirstName, ' ', LastName) as EmployeeName, CAST(Salary AS int)/12 as MonthlySalary, rent, Description
FROM employee e
JOIN Department d on e.DepartmentID = d.DepartmentID
JOIN Office o on d.DepartmentID = o.DepartmentID) t1
GROUP BY Description
GO
SELECT * FROM question4
GO
--5: How many deposit transaction are made after opening account

CREATE OR ALTER VIEW question5 AS
SELECT CONCAT(FirstName, ' ', LastName) AS CustomerName, COUNT(*) AS TotalTrans
FROM BankTransaction bt
JOIN Customer c
ON bt.CustomerID = c.CustomerID
GROUP BY CONCAT(FirstName, ' ', LastName)
```

```
GO
SELECT * FROM question5
GO
```