DBMS Project

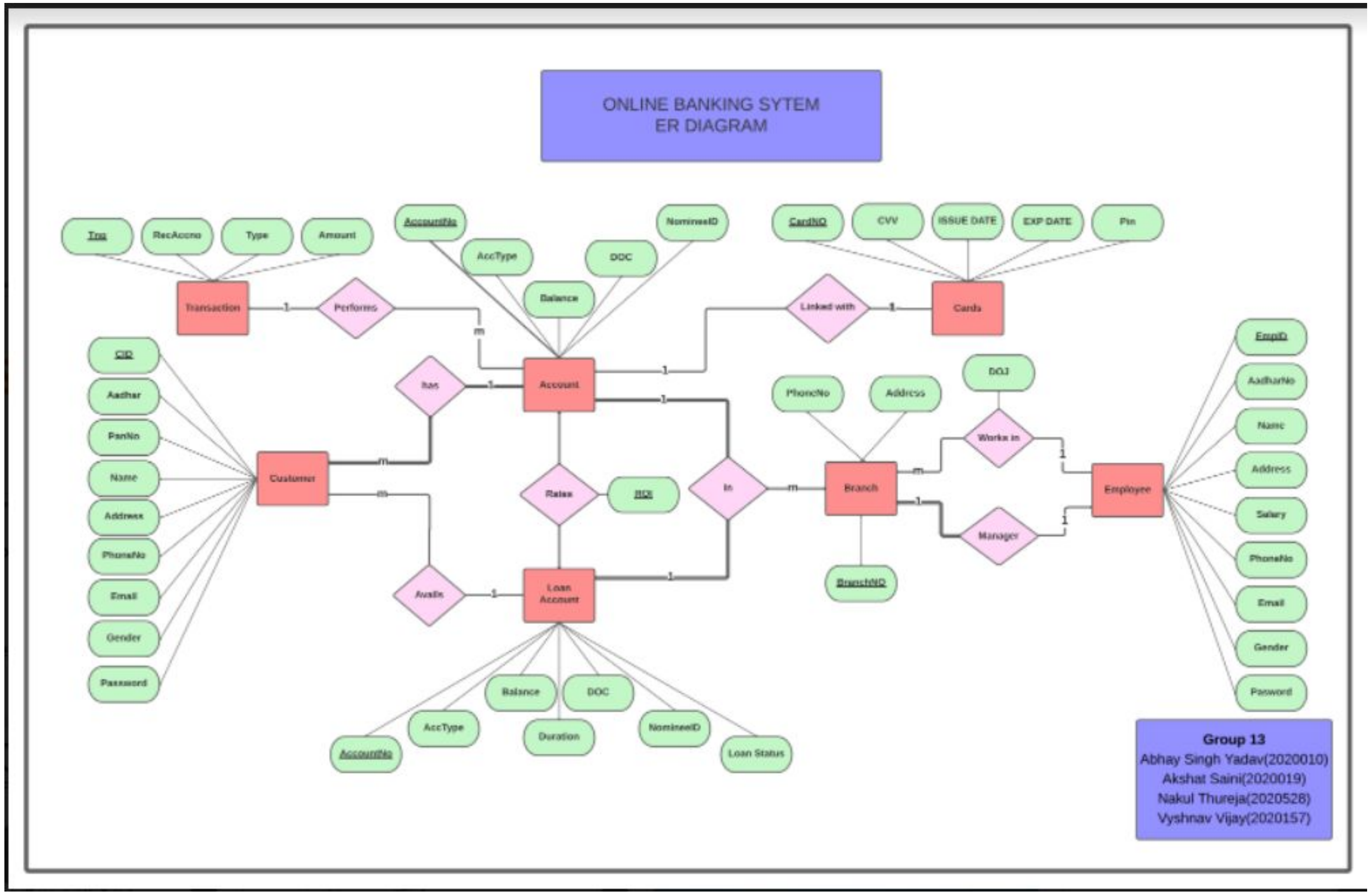
Updated Scope of Project

- We have added password for Customer and Employees to allow them to login into the database and have given grants according to their usage.
- Also we have added the Foreign Key Constraints missing from the Mid-Term Submission

ER Diagram (Updated)

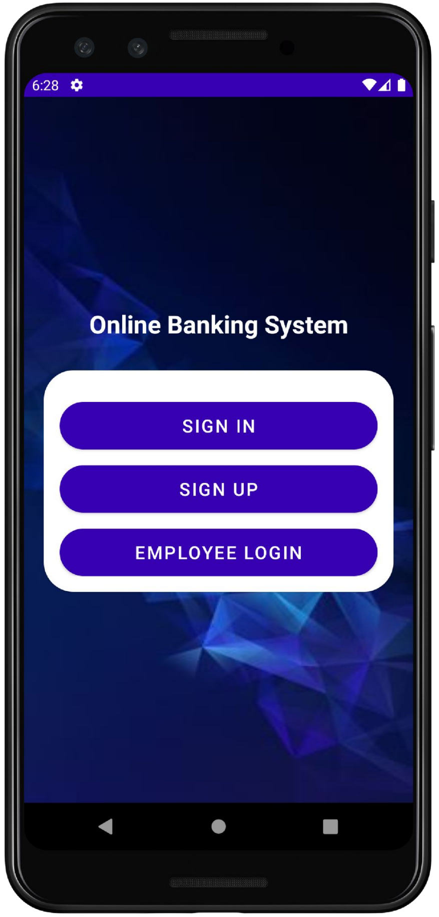
PDF LINK: https://drive.google.com/file/d/1N18PilVDM77IYO1XSCrreaCqn3lkr0LB/view?usp=sharing

(download for better resolution)



Project

- Android Application using Android Studio and Kotlin.
- Connected to mssql server using jdbc connector.
- Optimized and Implemented Embedded SQL Queries.



View and Grants

Specific view and grants have been provided for the different stakeholders in the app. Separate logins have also been provided for each user by using triggers.

- Admin - Has access to everything throughout the database with all grants
- Customer - Views are limited to the tables related to the Customer only, additionally they only have access to their own data. Only Select grant has been given so that they can see their own details only. Any updation or change by the Customer is directed through the admin.

```
"Create or Alter View customer_view as Select * from Customer where CID = $id"
"Grant Select on Customer_view to U$id"
```

Similar to the above queries grants and views have been created for every customer when they login

View and Grants

- Manager - Views are provided to all the tables, but they only have access to their own branch data. Only Select and Update grants have been give. Any deletion or insertion is directed through the admin.

```
"Create or Alter View manager_accounts_view as Select * from Accounts where
branchno = $branch"
"Grant Select,Update on manager_Accounts_view to M$id"
```

Similar to the above queries grants and views have been created for every manager when they login

Triggers

Triggers have been used for necessary data management in the application.

Triggers used are as follows -

Triggers

Trigger to create logins for new customers

Similar triggers for other stakeholders also

```
GO
CREATE TRIGGER login_create ON Customer
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE
        @SQL nvarchar(MAX),
        @Username nchar(20),
        @Password nchar(4)
    SELECT @Username = INSERTED.[CID], @Password =
INSERTED.[pass] FROM INSERTED
    SET @SQL = 'CREATE LOGIN U' + @Username + ' WITH
PASSWORD = '''+@Password+''', CHECK_EXPIRATION = OFF';
    EXECUTE(@SQL);
    SET @SQL = 'CREATE User U' + @Username + ' for login
U'+ @Username;
    EXECUTE(@SQL)
END
```

Triggers

Trigger to update balance of customers when a transaction is requested

```
GO
Create trigger trans_update on Transactions
AFTER INSERT
AS
Begin
       Set NOCOUNT ON;
       DECLARE
              @SQL nvarchar(MAX),
              @Sender bigint,
              @Receiver bigint,
              @Amount bigint,
              @initamount bigint,
              @initamount2 bigint
       SELECT @SENDER = INSERTED.[SenderAccNo], @Receiver = INSERTED.[ReceiverAccNo],
@Amount = INSERTED.[Amount] FROM INSERTED
       IF @Receiver > 0
    BEGIN
       SELECT @initamount = Accounts.[Balance] FROM Accounts WHERE AccNo = @Sender
       SELECT @initamount = Accounts.[Balance] FROM Accounts WHERE AccNo = @Receiver
    SET @SQL = 'Update Accounts set Balance = '+@initamount-@Amount+' where AccNo = '+
@Sender
       EXECUTE(@SQL)
       SET @SQL = 'Update Accounts set Balance = '+@initamount2+@Amount+' where AccNo =
'+@Receiver
       EXECUTE(@SQL)
       SET @SQL = 'Create or Alter View transactions_view as Select * from Transactions
where SenderAccNo = ' + @Sender +' or ReceiverAccNo = ' + @Receiver;
       Execute(@SQL)
    END
       ELSE
       SET @SQL = 'Create or Alter View transactions_view as Select * from Transactions
where SenderAccNo = ' + @Sender +' or ReceiverAccNo = ' + @Receiver;
       Execute(@SQL)
END
```

Triggers

Trigger to delete employee data from WORKS table, when employee is terminated

```
GO
CREATE TRIGGER Employee_Delete ON WORKS
AFTER DELETE
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE
        @SQL nvarchar(MAX),
        @ID nvarchar(15)
    SELECT @ID = DELETED.[EmpID] FROM DELETED
    SET @SQL = 'Delete from EMPLOYEE where Empid = 
'+@ID;
    Execute(@SQL)
END
```

Triggers

Trigger to allow updation of view for dynamic viewing

(similar triggers have been implemented for other tables also)

```
GO
CREATE TRIGGER availer ON Accounts
AFTER UPDATE
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE
        @SQL nvarchar(MAX),
        @Username nvarchar(4)
    SELECT @Username = INSERTED.[pin] FROM INSERTED
    SET @SQL = 'Create or Alter View cards_view as
Select * from Cards where Pin = '+@username;
    Execute(@SQL)
END
```

SQL Queries

In the following slides we have noted down some of the major queries running in our app.

Note that all the queries are in embedded form and uses some other variables also from the code. So the isolated queries here may not always be clear.

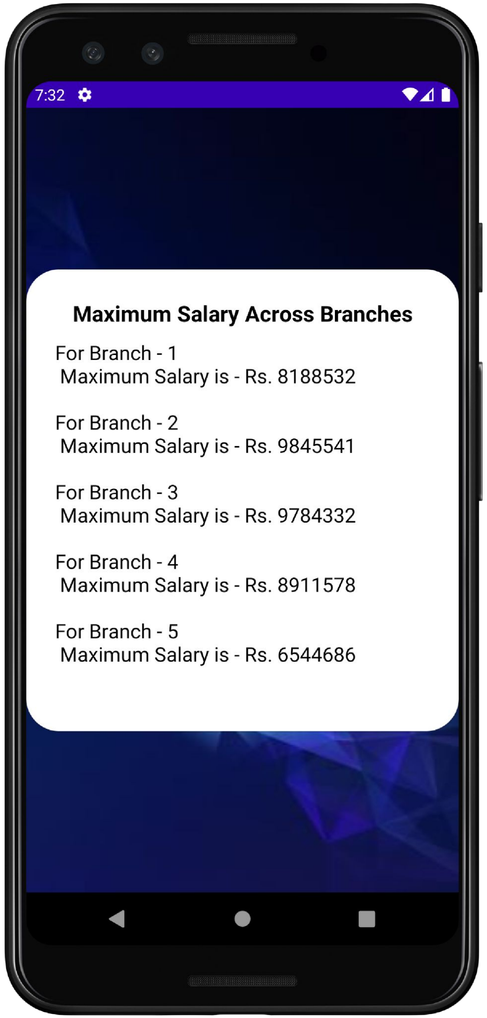
The common syntax followed for any embedded query is -

```
val connectionhelper: ConnectionHelper = ConnectionHelper()

val connect: Connection = connectionhelper.connectionclass(id, pass)

val query: String = "<SQL Query Here>”

val st: Statement = connect.createStatement()

val rs: ResultSet = st.executeQuery(query)
```

SQL Queries - 1

```
"Select
B.BranchNo,Max(E.salary) as
MaxSalary from Branch
B,Employee E,Works W where
W.BranchNo = B.BranchNo and
E.EmpID = W.EmpID
group by B.BranchNo"
```

(Query to print the maximum salary of employee branch wise)


7:32

Maximum Salary Across Branches

For Branch - 1
 Maximum Salary is - Rs. 8188532

For Branch - 2
 Maximum Salary is - Rs. 9845541

For Branch - 3
 Maximum Salary is - Rs. 9784332

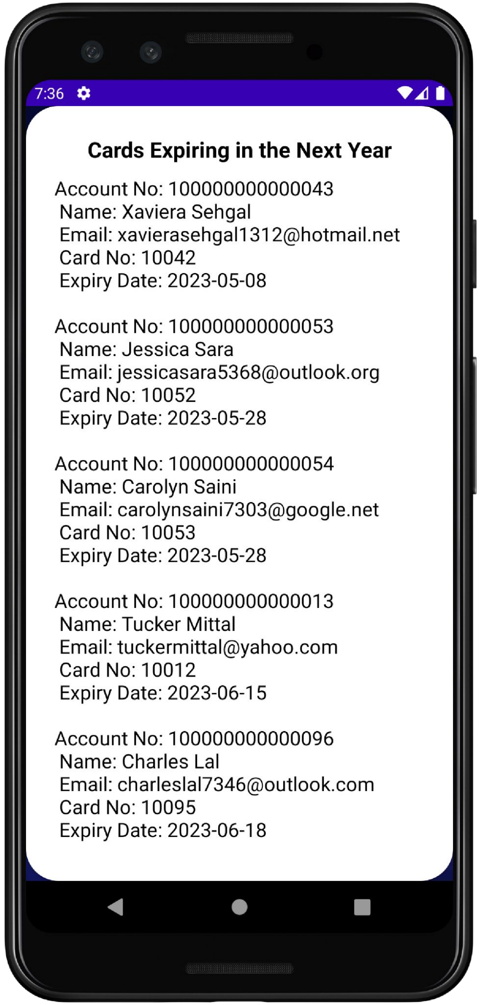
For Branch - 4
 Maximum Salary is - Rs. 8911578

For Branch - 5
 Maximum Salary is - Rs. 6544686

SQL Queries - 2

```
"Select
A.AccNo,c.name,c.Email,cd.CardNo,cd.ExpDate
from Accounts A,Customer C, Cards Cd
where c.cid = a.cid and A.AccNo = CD.AccNo
and Cd.ExpDate between '$expfinal' and
'$expfinal2'order by cd.ExpDate"
```

(Query to print details of customer whose card is expiring in the next year)



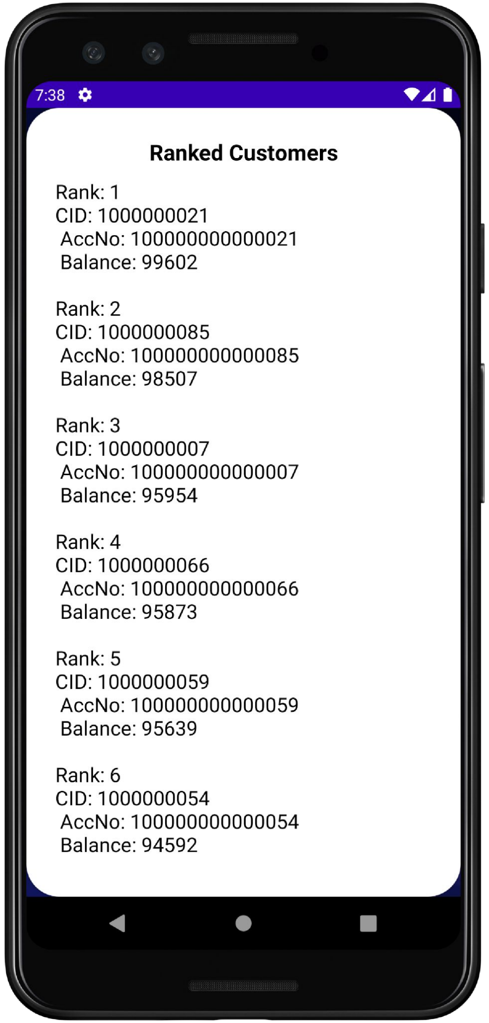
Cards Expiring in the Next Year

```
Account No: 100000000000043
 Name: Xaviera Sehgal
 Email: xavierasehgal1312@hotmail.net
 Card No: 10042
 Expiry Date: 2023-05-08

Account No: 100000000000053
 Name: Jessica Sara
 Email: jessicasara5368@outlook.org
 Card No: 10052
 Expiry Date: 2023-05-28

Account No: 100000000000054
 Name: Carolyn Saini
 Email: carolynsaini7303@google.net
 Card No: 10053
 Expiry Date: 2023-05-28

Account No: 100000000000013
 Name: Tucker Mittal
 Email: tuckermittal@yahoo.com
 Card No: 10012
 Expiry Date: 2023-06-15

Account No: 100000000000096
 Name: Charles Lal
 Email: charleslal7346@outlook.com
 Card No: 10095
 Expiry Date: 2023-06-18
```

SQl Queries - 3

```
"SELECT CID, AccNo, Balance,
DENSE_RANK() OVER(ORDER BY Balance
DESC) Rank FROM Accounts ORDER BY
Rank"
```

(Query to Rank Customers based on their Account Balance)



7:38

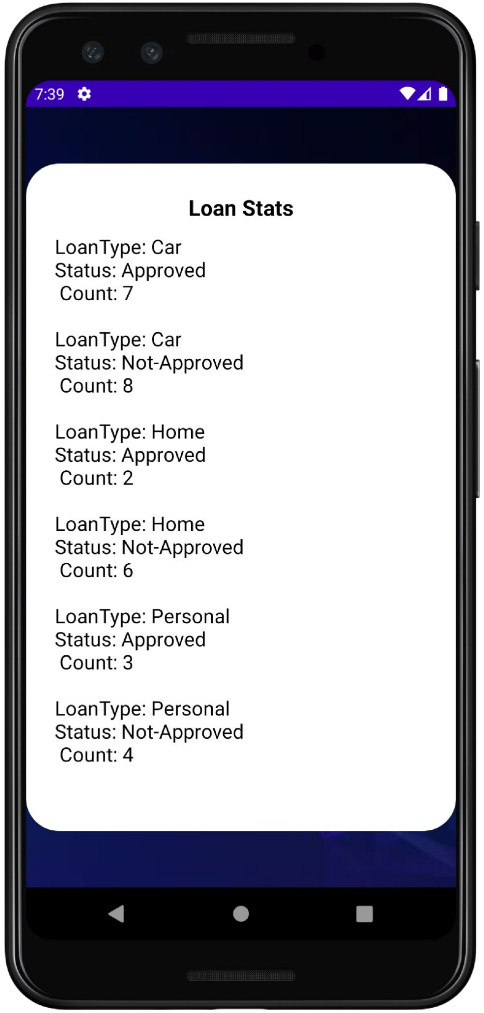
Ranked Customers

```
Rank: 1
CID: 1000000021
 AccNo: 100000000000021
 Balance: 99602

Rank: 2
CID: 1000000085
 AccNo: 100000000000085
 Balance: 98507

Rank: 3
CID: 1000000007
 AccNo: 100000000000007
 Balance: 95954

Rank: 4
CID: 1000000066
 AccNo: 100000000000066
 Balance: 95873

Rank: 5
CID: 1000000059
 AccNo: 100000000000059
 Balance: 95639

Rank: 6
CID: 1000000054
 AccNo: 100000000000054
 Balance: 94592
```

SQL Queries - 4

```
"select
L.LoanType,L.status,count(*) from
Loan L group by status,LoanType
order by LoanType,status"
```

(Query to print number of loan applications grouped by type and status)



Loan Stats

LoanType: Car
Status: Approved
Count: 7

LoanType: Car
Status: Not-Approved
Count: 8

LoanType: Home
Status: Approved
Count: 2

LoanType: Home
Status: Not-Approved
Count: 6

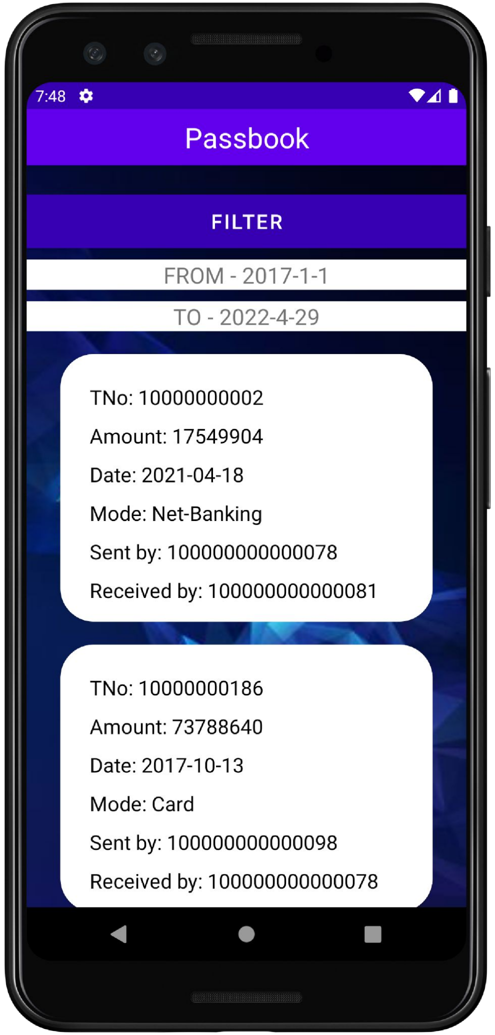
LoanType: Personal
Status: Approved
Count: 3

LoanType: Personal
Status: Not-Approved
Count: 4

SQL Queries - 5

```
"Select
TNo,amount,DOT,TransactionType,SenderAccNo
,ReceiverAccno from transactions_view
where (SenderAccNo = $acc_no or
ReceiverAccNo = $acc_no) and (DOT >=
'$datefrom' and DOT <= '$dateto') ORDER BY
DOT DESC"
```

(Query to filter out transactions based on dates)



TNo: 10000000002
Amount: 17549904
Date: 2021-04-18
Mode: Net-Banking
Sent by: 100000000000078
Received by: 100000000000081

TNo: 10000000186
Amount: 73788640
Date: 2017-10-13
Mode: Card
Sent by: 100000000000098
Received by: 100000000000078

SQL Queries - 6

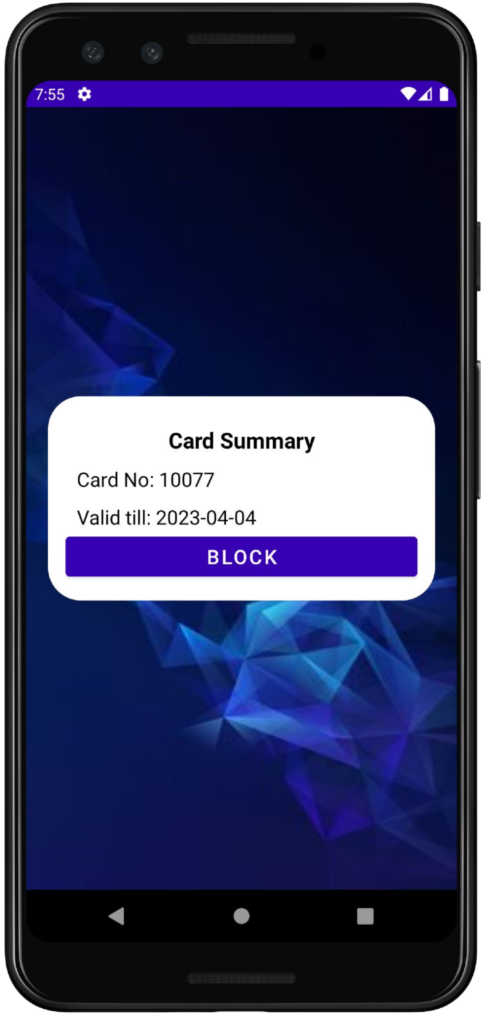
```
"Select CardNo,ExpDate from Cards_view where
Cards_view.AccNo = $account"
```

```
"INSERT INTO
Cards(CardNo,cvv,AccNo,IssueDATE,ExpDATE,pin) " +
"VALUES
(10$counter,'$rnds',$accNo,'$today','$expfinal',$pass)"
```

```
"Delete from Cards where CardNo = $cardNo"
```

Set of Queries to Handle Cards -

- Select to View Information
- Delete to Block Card
- Insert to get a new Card Issued (possible after blocking)


7:55

Card Summary

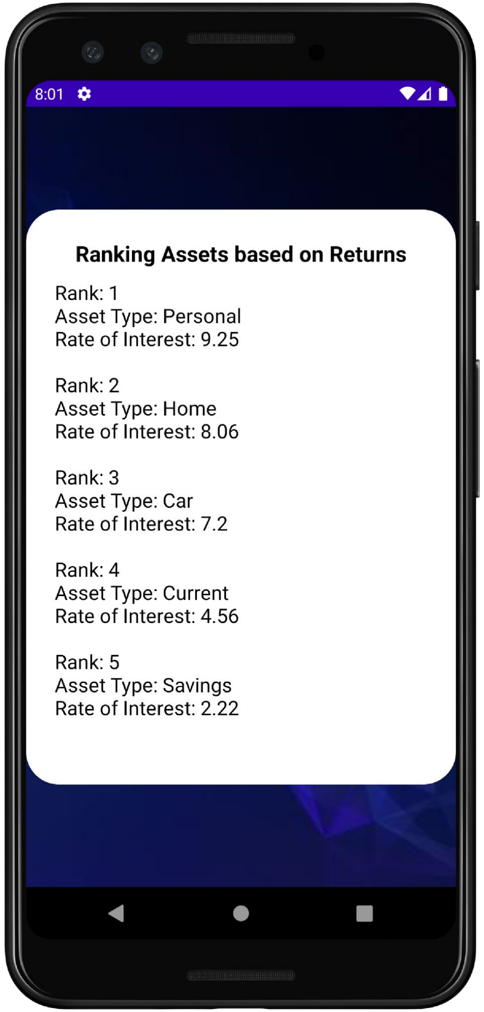
Card No: 10077

Valid till: 2023-04-04

SQL Queries - 7

```
"SELECT Type, ROI, DENSE_RANK()
OVER(ORDER BY ROI DESC) Rank FROM
Rates ORDER BY Rank"
```

(Query to print ranking of assets of the bank with respect to their Rate of Interest)



Ranking Assets based on Returns

Rank: 1
Asset Type: Personal
Rate of Interest: 9.25

Rank: 2
Asset Type: Home
Rate of Interest: 8.06

Rank: 3
Asset Type: Car
Rate of Interest: 7.2

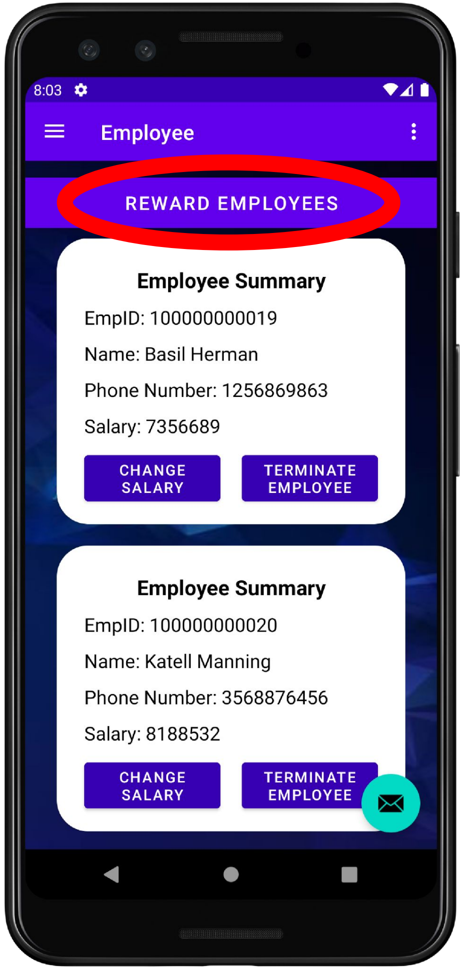
Rank: 4
Asset Type: Current
Rate of Interest: 4.56

Rank: 5
Asset Type: Savings
Rate of Interest: 2.22

SQl Queries - 8

```
"Update
manager_employee_view
set Salary =
1.05*Salary"
```

(Query to provide a bonus raise in the salaries of Employees. Here a separate query for View is maintained such that manager is not included in the employees)



8:03

Employee

REWARD EMPLOYEES

Employee Summary

EmpID: 100000000019

Name: Basil Herman

Phone Number: 1256869863

Salary: 7356689

CHANGE SALARY | TERMINATE EMPLOYEE

Employee Summary

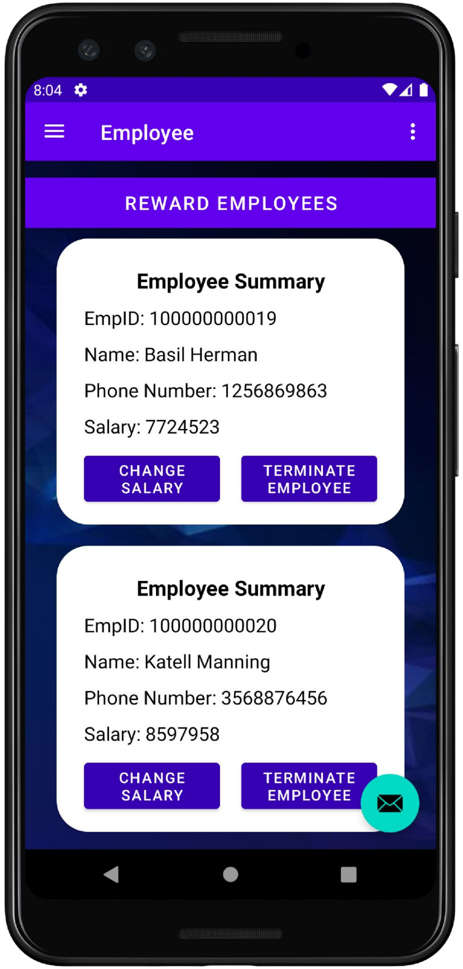
EmpID: 100000000020

Name: Katell Manning

Phone Number: 3568876456

Salary: 8188532

CHANGE SALARY | TERMINATE EMPLOYEE



8:04

Employee

REWARD EMPLOYEES

Employee Summary

EmpID: 100000000019

Name: Basil Herman

Phone Number: 1256869863

Salary: 7724523

CHANGE SALARY | TERMINATE EMPLOYEE

Employee Summary

EmpID: 100000000020

Name: Katell Manning

Phone Number: 3568876456

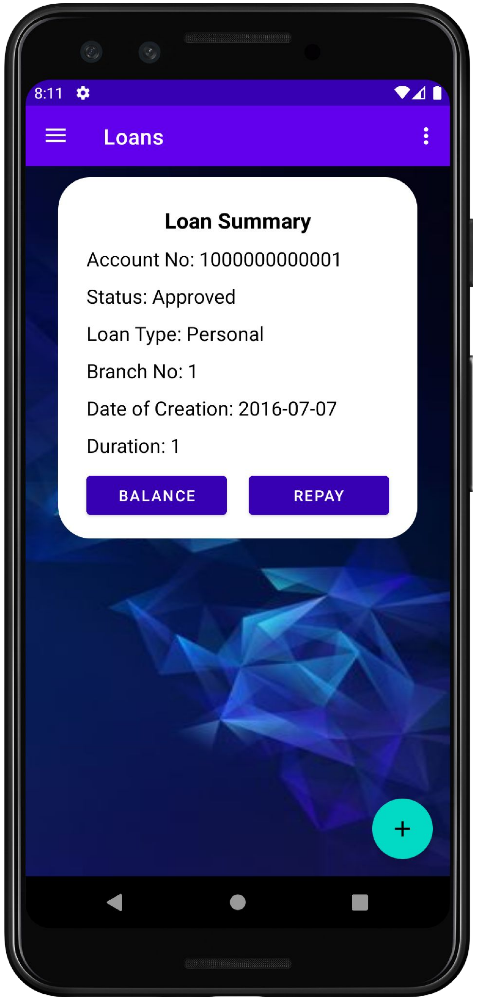
Salary: 8597958

CHANGE SALARY | TERMINATE EMPLOYEE

SQL Queries - 9

```
"Select LoanID, LoanType, BranchNo,
DOC, Duration, Status Loan_view where
Loan_view.CID = $id
```

(Query to find out Loans taken by the Customer)



Loan Summary

Account No: 1000000000001

Status: Approved

Loan Type: Personal

Branch No: 1

Date of Creation: 2016-07-07

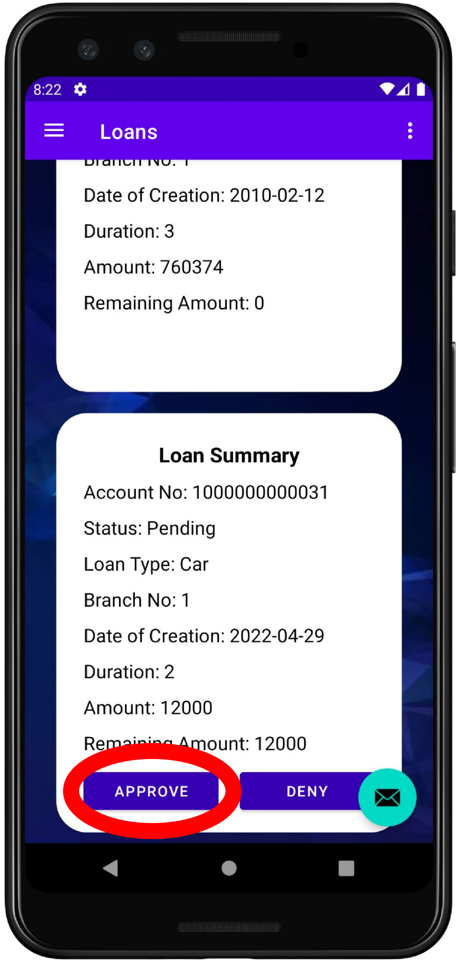
Duration: 1

SQL Queries - 10

```
if(t==0){

query = "Update manager_Loan_view
set Status = 'Not-Approved' where
LoanID = $acc_no"}

Else{

query = "Update manager_Loan_view
set Status = 'Approved' where
LoanID = $acc_no"}
```

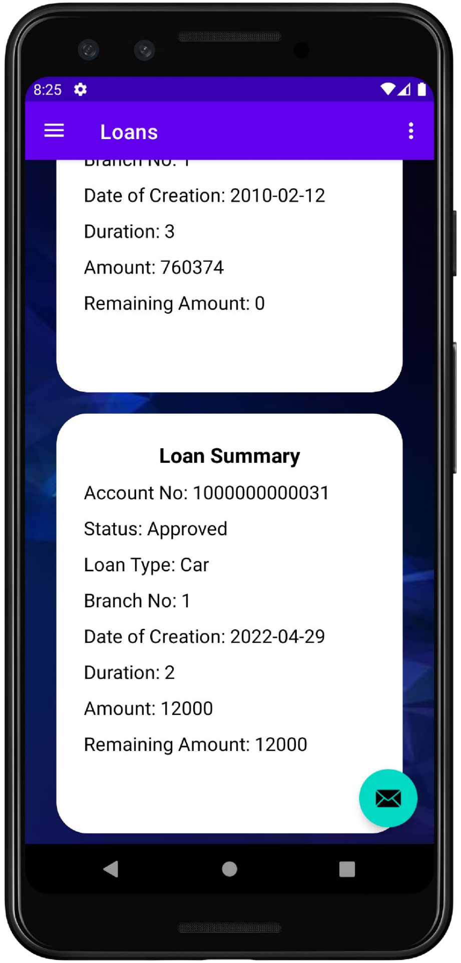
(Query used by the Manager to Approve or Deny a Loan)
Only available for pending loans.

Here t is a value obtained from the click on the app



Branch No: 1
Date of Creation: 2010-02-12
Duration: 3
Amount: 760374
Remaining Amount: 0

Loan Summary
Account No: 1000000000031
Status: Pending
Loan Type: Car
Branch No: 1
Date of Creation: 2022-04-29
Duration: 2
Amount: 12000
Remaining Amount: 12000



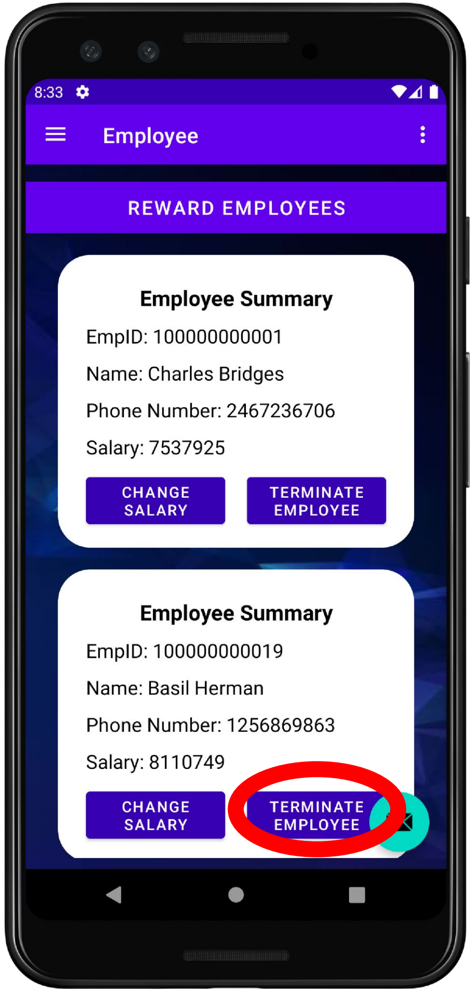
Branch No: 1
Date of Creation: 2010-02-12
Duration: 3
Amount: 760374
Remaining Amount: 0

Loan Summary
Account No: 1000000000031
Status: Approved
Loan Type: Car
Branch No: 1
Date of Creation: 2022-04-29
Duration: 2
Amount: 12000
Remaining Amount: 12000

SQl Queries - 11

```
"Delete from Works where
empID = $empID and Empid
not in (Select ManagerID
from Branch)"
```

(Query to terminate an Employee such that it is not a manager, works with a Trigger to delete other instances of Employee also to keep in check with Foreign Key Constraint)



8:33

Employee

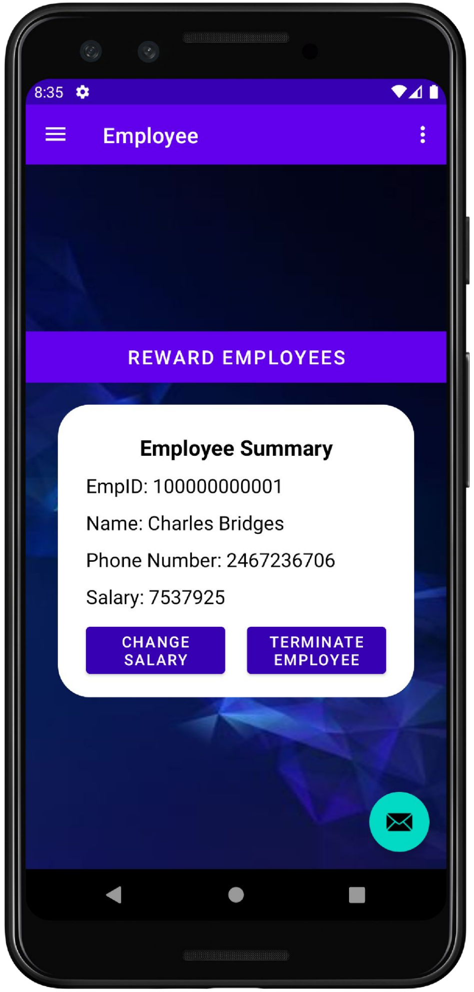
REWARD EMPLOYEES

Employee Summary
EmpID: 100000000001
Name: Charles Bridges
Phone Number: 2467236706
Salary: 7537925

CHANGE SALARY | TERMINATE EMPLOYEE

Employee Summary
EmpID: 100000000019
Name: Basil Herman
Phone Number: 1256869863
Salary: 8110749

CHANGE SALARY | TERMINATE EMPLOYEE



8:35

Employee

REWARD EMPLOYEES

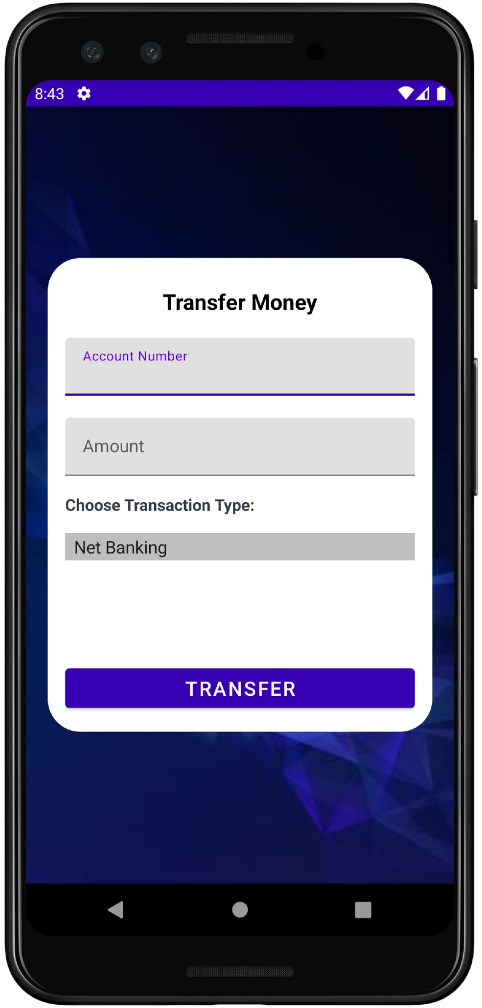
Employee Summary
EmpID: 100000000001
Name: Charles Bridges
Phone Number: 2467236706
Salary: 7537925

CHANGE SALARY | TERMINATE EMPLOYEE

SQl Queries - 12

```
"SELECT CAST( GETDATE() AS Date )"
"select count(*) from Transactions"
"select Balance from Accounts where Accno =
$acc_no"
"select Balance from Accounts where Accno =
$recAcc"
"INSERT INTO Transactions
(Tno,TransactionType,SenderAccNo,Amount,DOT,Receiv
erAccNo) VALUES
(10000000$counter,'$type',$acc_no,$amountcut,'$tod
ay',$recAcc)"
```

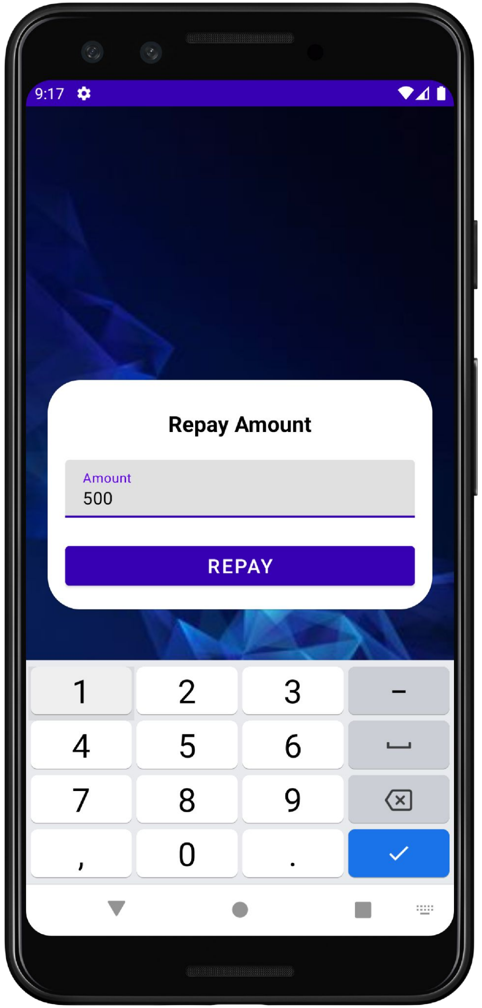
(Single Query of Creating a Transfer broken down into a series of queries to optimize the overall process)
(Works with triggers then to deduct and add the balance in the respective accounts)



SQl Queries - 13

```
"Update loan_view set RemainingAmount = $amount
where LoanID = $acc_no"
"INSERT INTO
Loan(LoanID,CID,BranchNo,LoanType,duration,DOC,Tota
lAmount,RemainingAmount,Status,NomineeID) VALUES
(1000000000$counter, $id, $branch_num,
'$loan_type', $duration, '$today', $amount,
$amount, 'Pending', $nominee)"
```

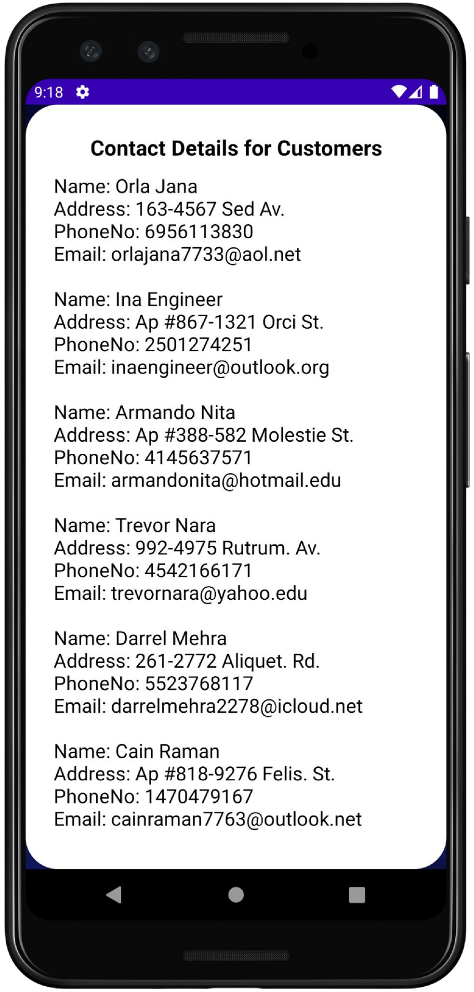
(Queries to get a new Loan and Repay a loan in sql server)



SQL Queries - 14

```
"SELECT Name,Address,PhoneNo,Email from Customer
Inner Join Accounts ON Customer.CID =
Accounts.CID and Accounts.BranchNo = $branch"
```

(Query to get contact details for the customer in the manager’s branch)


9:18

Contact Details for Customers

Name: Orla Jana
Address: 163-4567 Sed Av.
PhoneNo: 6956113830
Email: orlajana7733@aol.net

Name: Ina Engineer
Address: Ap #867-1321 Orci St.
PhoneNo: 2501274251
Email: inaengineer@outlook.org

Name: Armando Nita
Address: Ap #388-582 Molestie St.
PhoneNo: 4145637571
Email: armandonita@hotmail.edu

Name: Trevor Nara
Address: 992-4975 Rutrum. Av.
PhoneNo: 4542166171
Email: trevornara@yahoo.edu

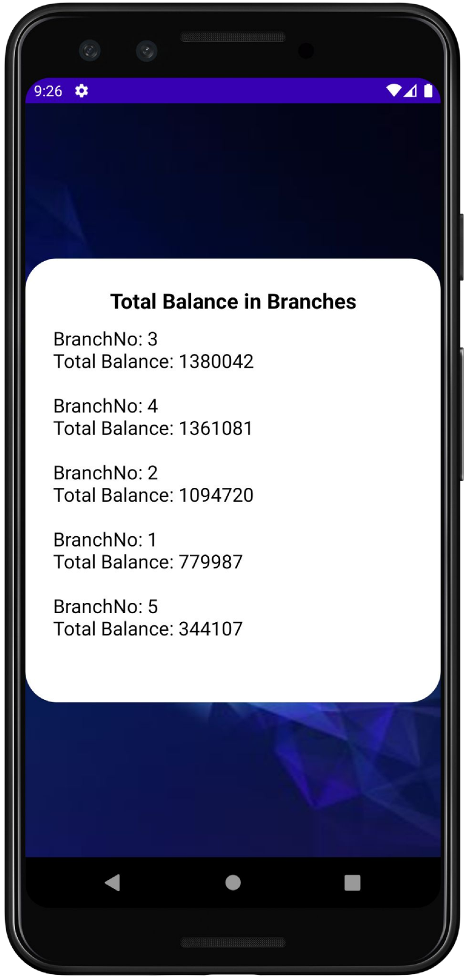
Name: Darrel Mehra
Address: 261-2772 Aliquet. Rd.
PhoneNo: 5523768117
Email: darrelmehra2278@icloud.net

Name: Cain Raman
Address: Ap #818-9276 Felis. St.
PhoneNo: 1470479167
Email: cainraman7763@outlook.net

SQl Queries - 15

```
 "Select SUM(balance) as totalbalance,BranchNo
from Accounts group by branchNO order by
totalbalance DESC"
```

(Query to get total balance across Branches ordered in a descending manner)


9:26

Total Balance in Branches

```
BranchNo: 3
Total Balance: 1380042

BranchNo: 4
Total Balance: 1361081

BranchNo: 2
Total Balance: 1094720

BranchNo: 1
Total Balance: 779987

BranchNo: 5
Total Balance: 344107
```

Indexing

Judging from our use case and scope we have defined the following Indexes for our database.The columns that are frequently searched against, have been included in these indexes.

```
CREATE INDEX index_customer ON Customer (CID,Pass)

CREATE INDEX index_employee ON Employee (EmpID,Password)

CREATE INDEX index_cards ON Cards (expdate,AccNo)

CREATE INDEX index_transactions ON Transactions
(DOT,SenderAccNo,ReceiverAccno)

CREATE INDEX index_loan ON Loan (CID)
```

Optimizing Queries

Following the rules of Query Optimization we have optimized all our SQL Queries.

One such example here is -

```
Select
T.Tno,T.TransactionType,T.SenderAccNo as
AccountNo,T.amount as Withdrwal,"" as
Deposit ,T.DOT from Transactions T,Accounts
A
where 1000000007 = A.cid and T.SenderAccNo
= A.AccNo
and T.DOT >= date_sub(current_date,
INTERVAL 10 YEAR)
UNION
Select
T.Tno,T.TransactionType,T.ReceiverAccNo as
AccountNo,"" as Withdrwal,T.amount as
Deposit,T.DOT from Transactions T,Accounts
A
where 1000000007 = A.cid and
T.ReceiverAccNo = A.AccNo
and T.DOT >= date_sub(current_date,
INTERVAL 10 YEAR)
order by DOT;
```

```
"Select
TNo,amount,DOT,TransactionType,SenderAccNo,
ReceiverAccno from transactions_view where
(SenderAccNo = $acc_no or ReceiverAccNo =
$acc_no) and (DOT >= '$datefrom' and DOT <=
'$dateto') ORDER BY DOT DESC"
```

Filtering out Transactions based on Dates

Optimizing Queries

Another example -

```
"Select Salary,EmpId from
manager_employee_view where Empid
not in (Select ManagerID from
Branch)"
FOR EACH EMPID
"Update manager_employee_view SET
salary = $newsalary where Empid =
$empID"
```

$\rightarrow$

```
"Update manager_employee_view set
Salary = 1.05*Salary"
```

Providing salary benefits to Employee

Contribution of Members

Thank You

Group 13

Abhay Singh Yadav(2020010)
Akshat Saini(2020019)
Nakul Thureja(2020528)
Vyshnav Vijay(2020157)