CUSTOMER CARE DATABASE

FOR

A BANKING SYSTEM

BY: S10_T2

PREPARED BY:

201801015 – SHANTANU TYAGI

201801030 – NIKHIL MEHTA

201801162 – VATSAL GUJARATI

201801408 – ARKAPRABHA BANERJEE

MENTORED BY:

MAYANK PATEL JR.

4 NOVEMBER 2020

INDEX

FINAL VERSION OF SRS

INTRODUCTION

PURPOSE

The primary purpose of this document is to provide support information and an overview regarding the Customer Database Project for Financial Institutions. It attempts to explain the primary functionality and features of the aforementioned product from a broad perspective. Customer Care services are primarily required to cater to various kinds of queries and issues which the customer of that particular service might have. For services which operate on a huge scale it is imperative that a proper database is present which contains all the relevant information pertaining to that service and also possesses the capability to efficiently fetch and modify data with proper provisions for validation and login for the end users. This shall serve as a comprehensive piece of documentation with regard to each of these functionalities and User Classes in detail.

INTENDED AUDIENCE AND READING SUGGESTIONS

While the software requirement specification (SRS) document is written for a more general audience, this document is intended for:

- Developers who can review project's capabilities and more easily understand where their efforts should be targeted to improve or add more features to it (design and code the application – it sets the guidelines for future development).

- Project testers can use this document as a base for their testing strategy as some bugs are easier to find using a requirements document. This way testing becomes more methodically organized and efficient as well.

- End users of this application who wish to read about what this project can do. Most of the end users are often in the dark regarding certain functionalities of any system. This document shall strive to educate them regarding the same.

- Developers and Testers are encouraged to have a pre-requisite knowledge regarding Database Design and SQL functionalities/queries. The project shall primarily be open-source in nature, hence this pre-requisite knowledge shall enable them to understand the existing flaws and provide feasible solutions to it as well.

This document need not be read sequentially. Readers are encouraged to jump to any section they find relevant.

PRODUCT SCOPE

The primary purpose of this Customer Care Database is to meet the expectations of the customers with regards to an array of services/queries spanning over a wide variety of Financial Institutions. In addition to that it also aims to provide customers with constructive solutions for a user-friendly and hassle-free experience for their desired query/service. This product also aims to understand the queries of the customers and ensure that they enjoy a cost-effective and flawless experience with respect to their service. It furthermore enables the Service Providers to gain better insights with regards to the usability of their products which in turn helps them improve their services/products and makes them more efficient.

OVERALL DESCRIPTION

PRODUCT PERSPECTIVE

Users can get information regarding their account/balance after validating their account and can furthermore transfer money to other valid accounts. In addition to that customers can also block their existing accounts in case of extraordinary events. New users can also create accounts. Other general queries within the purview of the system shall also be catered to. Administrators would have access to the entire database in order to maintain information integrity throughout the database. Service Providers would have a lower sense of privilege with regards to the Admin and can only modify/provide information when prompted by the user. End Users can only view information pertaining to their own account after validation. Auditors can view the entire database and report faulty transactions to the admin and the bank as well. The owner also has admin privileges to facilitate policy changes for customers.

BACKGROUND READINGS

Description:

- **Book**:

Database System Concepts by Abraham Silberschatz: We read part 1-Relational languages, and part 2- Database design from the book. We learned now the relational model remains the primary data model for commercial data-processing applications. It attained its primary position because of its simplicity, which eases the job of the programmer, compared to earlier data models such as the network model or the hierarchical model. We also learned about important concepts, logic and different terminologies that will be useful while working on this project. Continuing the reading we were introduced to database design using ER model and how can it be transformed into a set of relation schemas and how some of the constraints can be captured in this design. The various features of the E-R model offer the database

designer numerous choices in how to best represent the enterprise being modelled. Concepts and objects may, in certain cases, be represented by entities, relationships, or attributes. Aspects of the overall structure of the enterprise may be best described by using weak entity sets, generalization, specialization, or aggregation. Often, the designer must weigh the merits of a simple, compact model versus those of a more precise, but more complex one. UML is a popular modelling language. UML class diagrams are widely used for modelling classes, as well as for general-purpose data modelling.

- **Websites**:

We read numerous blogs and articles on individual database concepts but more importantly we had to understand what kinds of customer care services do most banks offer so that we could decide what features we wanted to include and also helped us the existing problems and possible solutions and how can we implement them. In order to properly understand customer care solutions, we looked up for various companies and start-ups providing such services and how they are trying to optimize and utilizing the queries to improve their services.

- **Videos:**

PostgreSQL Tutorial For Beginners | Learn PostgreSQL | Introduction to PostgreSQL | Edureka: Since we will be working on PostgreSQL, we had to take a basic course to strengthen our fundamentals. It covered all beginner topics from commands, keys, entity, constraints, operators, triggers and functions.

How to Design Your First Database: This video covered the rules to follow when designing databases, as well as general design principles.

References:

- https://www.geeksforgeeks.org/how-to-write-a-good-srs-for-your-project/
- https://medium.com/@vincetran_28429/software-requirements-specification-srs-document-fd9ab103b18
- https://www.geeksforgeeks.org/introduction-of-er-model/
- https://www.tutorialspoint.com/dbms/er_model_to_relational_model.htm
- https://nptel.ac.in/courses/106/106/106106093/
- https://www.creditmantri.com/customer-care/
- https://www.slideshare.net/AshwinkumarDinoriya/banking-database
- https://www.youtube.com/watch?v=-VO7YjQeG6Y
- https://www.youtube.com/playlist?list=PLQVJk9oC5JKohoyVILfdxOOzyl6w-yOur

COMBINED REQUIREMENTS:

- Problem analysis
- Determine the purpose of the database
- Determining data to be stored
- Find and organize the information required
- Determining data relationships
- Logically structuring data
- ER diagram
- Divide the information into tables
- Physical Schema Design
- Specify constraints
- Set up the table relationships
- Apply the normalization rules

CUSTOMER CARE DATABASE: INTERVIEW PLAN

Interview 1

System: Alphabet Consultancy
Project Reference: SF/SJ/20XX/XX
Interviewer*: 1)* Arkaprabha Banerjee
Designation: Backend Developer for Customer Care Database
Contact Details: 7016570121
Organization Details: CEO at Alphabet Consultancy

Interviewee: 1) Harish Tyagi
Designation: Software Auditor
Date: 27th September 2020 **Time:** 10:30 AM
Duration: 30 minutes **Place:** Online Zoom meeting
Purpose of the Interview: Preliminary meeting to understand how the existing customer care services can be improved by better encryption.
Agenda: To discuss and deliberate upon the existing flaws in the system in terms of security so as to provide comprehensive suggestions to the aforementioned database project.
Documents to be interviewed: Any documents relating to current customer care services.

Interview 2

System: Alphabet Consultancy
Project Reference: SF/SJ/20XX/XX
***Interviewer:** 1)* Arkaprabha Banerjee
Designation: Backend Developer for Customer Care Database
Contact Details: 7016570121
Organization Details: CEO at Alphabet Consultancy

Interviewee: 2) Durgadas Mahato
Designation: Retired army personnel
Date: 27th September 2020 **Time:** 11:00 AM

Duration: 38 minutes **Place:** Online Zoom meeting
Purpose of the Interview: Preliminary meeting was to understand how frequently elderly people use customer care services and their requirements.
Agenda: To discuss and deliberate upon the frequency of these services which are accessed so as to improve the customer experience.
Documents to be interviewed: Few documents relating to the frequency of the current banking customer care availing services.

Interview 3

System: Alphabet Consultancy
Project Reference: SF/SJ/20XX/XX
Interviewer: *1)* Arkaprabha Banerjee
Designation: Backend Developer for Customer Care Database
Contact Details: 7016570121
Organization Details: CEO at Alphabet Consultancy

Interviewee: 3) Alpesh Sharma
Designation: Banker at Goldman Sachs
Date: 27th September 2020 **Time:** 12:00 PM
Duration: 35 minutes **Place:** Online Zoom meeting
Purpose of the Interview: Preliminary meeting was to understand the time constraints of middle age/working people.
Agenda: To know and deliberate upon the time constraints that people might have with respect to different age groups.
Documents to be interviewed: Documents showing his working hours and general statistics regarding the end users and the bank.

Interview 4

System: Alphabet Consultancy
Project Reference: SF/SJ/20XX/XX
Interviewer: *1)* Arkaprabha Banerjee
Designation: Backend Developer for Customer Care Database

Contact Details: 7016570121
Organization Details: CEO at Alphabet Consultancy

Interviewee: 4) Abhishek Mehra
Designation: System Administrator at Juniper Networks.
Date: 27th September 2020 **Time:** 1:00 PM
Duration: 40 minutes **Place:** Online Zoom meeting
Purpose of the Interview: To understand how existing banking customer care services can be improved.
Agenda: To discuss upon the current flaws of the banking customer care services so as to improve upon them and enhance the user experience.
Documents to be interviewed: Any documents related to the improvements which the existing customer care databases should have.

Interview 5

System: Alphabet Consultancy
Project Reference: SF/SJ/20XX/XX
***Interviewer:** 1)* Arkaprabha Banerjee
Designation: Backend Developer for Customer Care Database
Contact Details: 7016570121
Organization Details: CEO at Alphabet Consultancy

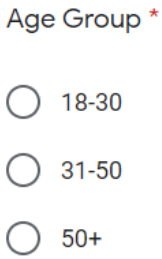
Interviewee: 5) Nikhil Shah
Designation: Service provider at TCS.
Date: 27th September 2020 **Time:** 5:00 PM
Duration: 40 minutes **Place:** Online Zoom meeting
Purpose of the Interview: To understand how a constructive feedback loop can be put in place to improve the existing services.
Agenda: To discuss upon the current flaws in the feedback system of the banking customer care services.
Documents to be interviewed: Any documents related to the improvements, which the existing customer care databases should have.

COMBINED REQUIREMENTS:

- Prioritizing issues for middle-aged and elderly people as they primarily face a lot of time constraints as opposed to the younger generation.
- Issues which demand immediate attention (Blocking or suspending accounts) should be prioritized irrespective of the age group.
- Recurring Issues should be solved more efficiently at later stages.
- Need to implement better encryption thus increasing reliability.
- The database shall strive to attain a constructive feedback loop with respect to its customers in order to foster a more holistic environment to improve the current system.

QUESTIONNAIRE:

Question 1



Purpose of asking this question: We exactly wanted to know which age group of people use the customer care services because then it will help us know for which age group should our database be inclined and accordingly modify the schema.

Question 2

How frequently do you use customer care services for banking purposes? *

- ◯ Almost everyday
- ◯ Weekly
- ◯ Monthly

Purpose of asking this question: Our motto behind asking this question was to understand the frequency at which people use the customer care services so that we can understand what flaws are there in our database because only then we'll know why people aren't using our customer care service.

Question 3

How much time did it take for your issue to get resolved? *

- ◯ Within 2 days
- ◯ Around one week
- ◯ More than a week

Purpose of this question: After the response of this question, we understood that how long does it take for an issue to get resolved because

for example if an issue takes too long to get resolved then it is possible that the user would not be happy with the customer care service. According to the time it takes for an issue to get resolved, we'll modify and improve our database accordingly.

Question 4

Do you think your recurring issues tend to get solved faster every time you encounter them *

- ◯ Yes
- ◯ No

Purpose of this question: We wanted to understand that if a person lodges a complaint along the previous lines, how long does it take to get resolved. It is possible that a new complaint which gets registered might get stored somewhere in the database which when tried to access again, takes a very long time. So, if this is the case then we'll design our database in such a manner so that if we complaint along the previous lines, it should not take too long for that complaint to get resolved.

Question 5

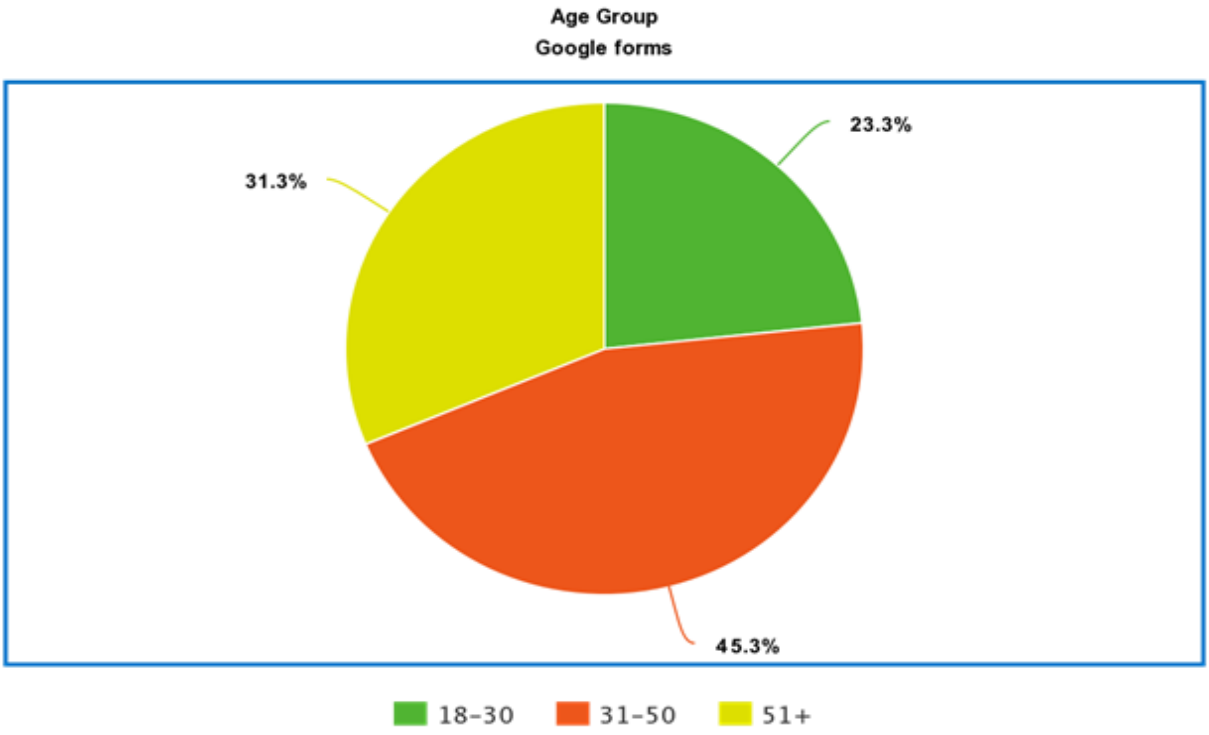
What improvements would you suggest for the existing customer care services?

Your answer

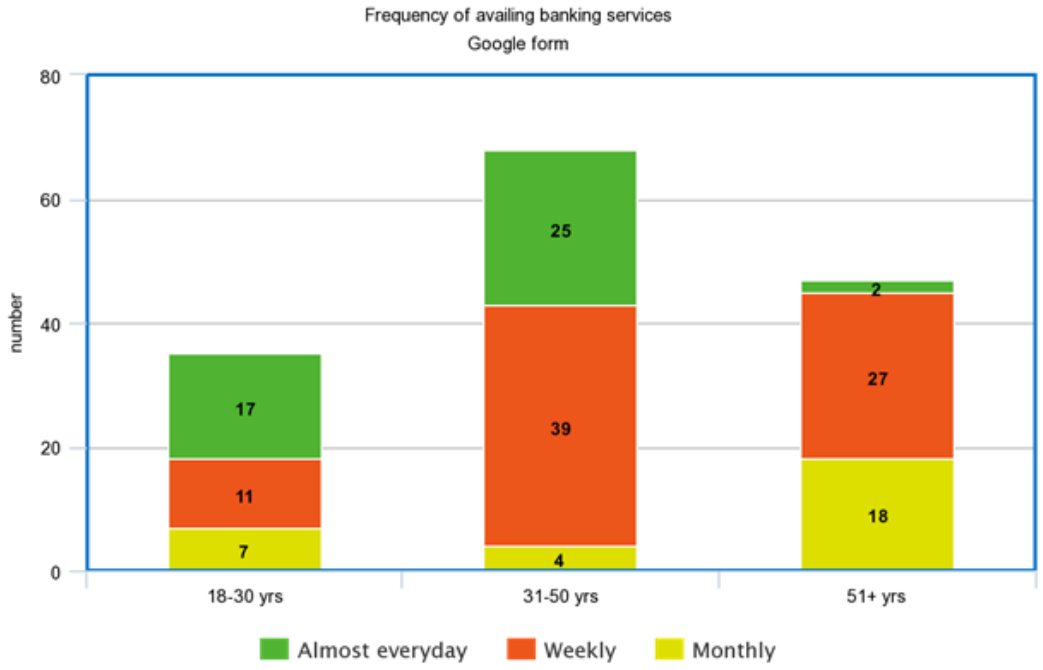
Purpose of this question: The most important component of our customer care database will be the user. If the user is not happy then we'll have to modify our database accordingly. So, suggestions from the users were taken and we tried to understand what difficulties the users are facing and how can we improve upon them.

OBSERVATIONS

- The first graph represents the age distribution of the people who filled up the survey form. This demography distribution shall help us to understand the rest of the survey responses better and furthermore help in generating a constructive feedback loop to understand the existing flaws and implement thee required functionalities.



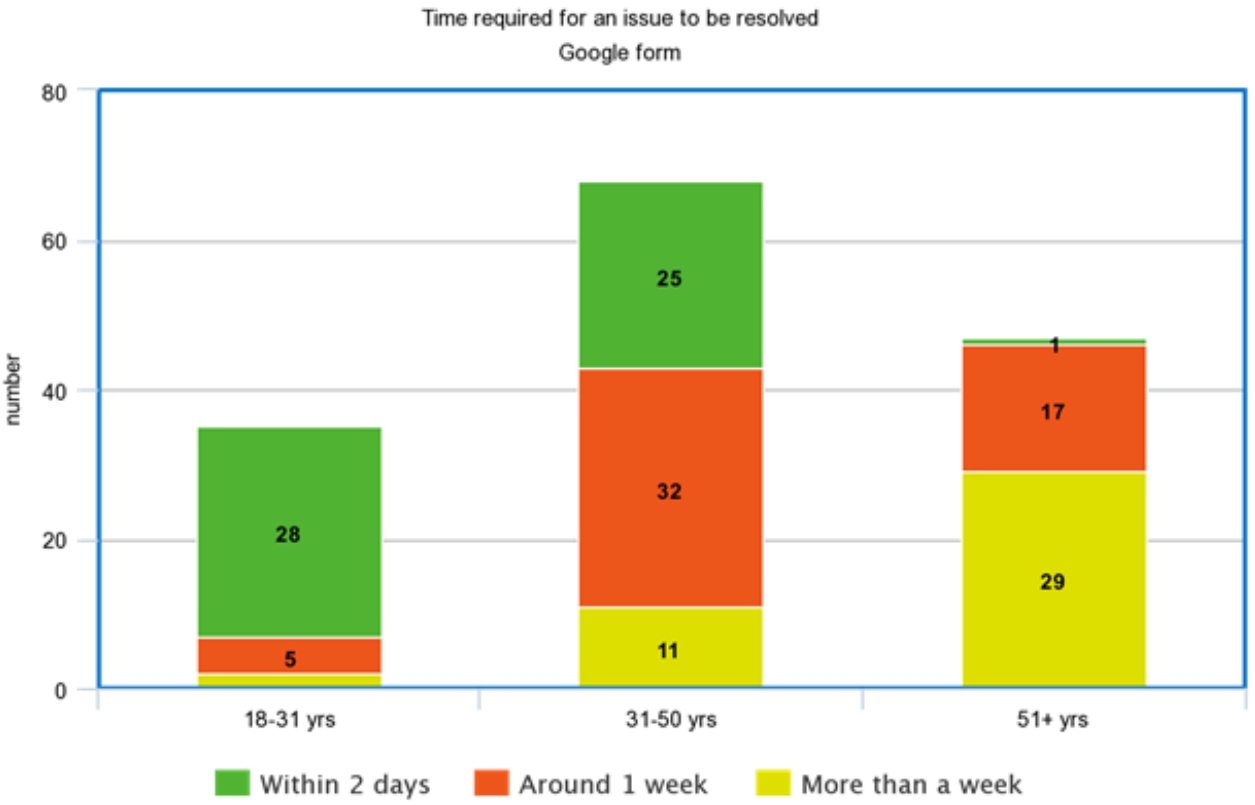
- The second graph represents the response to Q2 present in the form. The following observations have been made:
    1. Most of the people in the age bracket of 18-30 years, tend to require these services almost every day. This can be explained by that fact that most of these people tend to have a lot of bank transactions via digital forms. Hence its essential that they stay updated to avoid faulty/erroneous transactions
    2. We observe that people in the age range of 31-50 years primarily use these services on a weekly basis. This can be attributed to the fact that they are working professionals and can't devote time for such issues on a daily basis. Time is an important factor for this class of people.



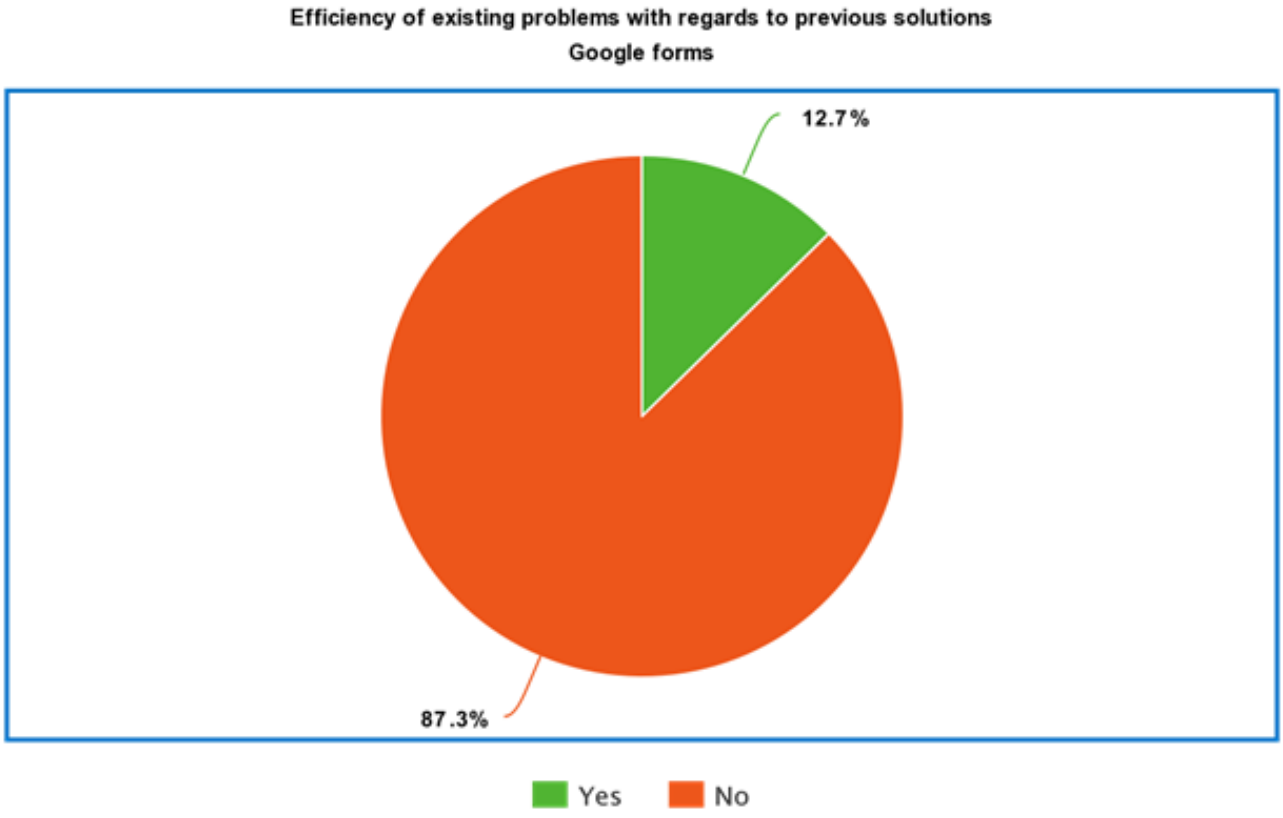
    3. For elderly people, most of them have contact the customer care in a weekly or monthly. Since they are not involved in frequent transactions, hence their interaction on a daily basis with customer care services is quite diminished.

- This graph primarily represents the time required to solve issues pertaining to the banking services offered:

    1. For the younger audience most of their queries were solved within a day or two. This is primarily due to the fact that most of their issues are less complex in nature and hence can be done faster.
    2. For the middle aged there is an equitable distribution in all 3 categories. This is because their issues span over a wide range of services. Still there remains a considerable number of issues which take up a large amount of time (around a week). This is worrisome especially since they have a time constraint imposed upon them.
    3. For the elderly, majority of their issues take a large amount of time to get solved. The complexity of their issues may have a role to play in this scenario.



- About 87.3% of the people seem to think that the existing service does not seem to get more efficient when they have an issue which they had also previously faced. This is a major point to work upon as it affects all the age groups.



FACT FINDING CHART

Objective	Technique	Subject(s)	Time Commitment
To know, how often do people need banking services	Interview	One elderly person, a middle aged person	2x1 hour each
To know how long does it take for one's complaint to get registered and get resolved	Interview, Survey	Google forms, a few middle aged people	1 day for google form, Number of people x ½ hour each
To know that if any complaint lodged is along the previous lines, how long does it take to get resolved.	Survey	Google forms	1 day
To know how can the existing banking services could be improved	Interview	People from all the age group	2 days
The study of schema of bank databases and how are customers catered	Background reading	Online websites, journals	1 day
To find how people prefer the banking services, i.e. online or offline.	Interview and survey	Google forms, people from all the age group	1 day

LIST REQUIREMENTS

- A major requirement especially among elderly and middle-aged people was with regards to efficiency. Recurring Issues which have been already solved in

the past should typically be solved faster and shouldn't take the same amount of time as that of the previous ones.

- Issues pertaining to middle-aged and elder people should typically be prioritized in cases where a significant time is already being invested in solving the issue. This is primarily to facilitate the time crunch faced by middle-aged people and to accommodate the elderly as well.
- The System must have provisions to accommodate cancelling cards/accounts in extreme situations on an urgent priority basis irrespective of the age group.
- The database shall strive to attain a constructive feedback loop with respect to its customers in order to foster a more holistic environment to improve upon the current system.
- The system shall provide storage of all databases on redundant computers with automatic switchover.
- The system shall provide for replication of databases to off-site storage locations.
- The system shall provide RAID V Disk Stripping on all database storage disks.
- The system's back-end servers shall only be accessible to authenticated administrators
- The system's back-end databases shall be encrypted

USER CLASSES AND CHARACTERISTICS

There are basically five categories of users: -

- Administrator: These people are involved in managing the database from an overall point of view.

- Service Provider: These are responsible for providing services to the customers by accessing the relevant data from the database.

- End user: These are the people who avail the services.
- Auditor – They have full access to view the database and the admin menus in order to conduct timely audits to maintain data integrity.
- Owner Bank: The bank using the database can update it's information.

OPERATING ENVIRONMENT

Recommended Operating Systems: -

- **Windows:** 7 or newer
- **MAC:** OS X v10.7 or higher
- **Linux:** Ubuntu

Hardware Requirements: -

- We strongly recommend a computer fewer than 5 years old.
- Processor: Minimum 1 GHz; Recommended 2GHz or more.
- Hard Drive: Minimum 32 GB; Recommended 64 GB or more.
- Memory (RAM): Minimum 1 GB; Recommended 4 GB or above.
- Some classes require a camera and microphone.

Recommended Operating Browsers: -

- Safari version 7 and above.
- Chrome version 44 and above.
- Firefox version 40 and above.

PRODUCT FUNCTIONS

This section provides the functional overview of the customer care database. Various functional modules that can be accessed by the user are:

1. Login:

This module allows valid customers to access the functionalities provided by the bank. Customer logins by entering customer id and the login pin.

2. Get balance information:

This module maintains the balance details of a particular account. This system must be networked to the bank's computer. The updated database of every customer is maintained with bank. Hence the balance information of every account is available in the database and can be displayed to the customer.

3. Customer info:

This module allows the customer to view and update the profile of their account. It also allows them to view their account status, load information and transaction details.

4. Transfer Money:

This module allows the customers to transfer funds from one account to another within the same bank.

5. General Grievances:

This module allows the customer to suspend their account and block their cards and allows new users to create a new account.

PRIVILEGES

- **Administrator:** His role includes capacity planning, installation configuration, database design, data recovery etc. These are exclusive tasks and are only to be performed by the administrator. No one else except the admin has the access to person these tasks.
- **Service Provider:** The service provider primarily has access to look into the database. This allows them to help the customers with resolving their queries and providing them with services. Also, the service provider can modify certain fields of the database to cater to the immediate needs of the user provided he/she has provided the login credentials.
- **End User:** They will not be having any access, i.e. they cannot modify or look up the entire database. They can just avail the services and will only be allowed to look into their own data, i.e. their account number, bank balance, etc.
- **Auditor:** They shall have complete access to view the entire database and the changes incorporated by the Service Provider and the Administrator as well. Their primary role is to maintain data integrity and look for faulty transactions. In case of an issue, it shall notify the same to the Administrator and the Bank as well.
- **Owner Bank:** They shall have complete edit and view access to the entire database. This is primarily to facilitate policy changes for its customers or update information regarding certain queries.

ASSUMPTIONS

- We have assumed that the customer care services possess the complete data regarding the customer's bank accounts and are not dependent on the banks to explicitly provide that information.
- We have also assumed that the service providers/admin can block any account or any debit/credit card if the customer wishes so. There is no need for a confirmation from the bank regarding this issue.
- The bank has given explicit authority to the customer care center to read and modify the information of the customers.

BUSINESS CONSTRAINTS

- Scalability of the aforementioned database could be an issue if the user pool is extremely large. Distributed Database Systems need to be used for that.
- The data needs to be stored on a server. For large amounts of data, storing it on cloud servers could be an expensive affair.
- For highly scaled systems, there should be dedicated personnel to maintain and handle the complete system.

FINAL NOUN ANALYSIS

TABLES

ALL NOUNS

Noun	Noun	Noun	Noun	Noun
Customer Care services	various kinds	queries	issues	the customer
that particular service	services	a huge scale	it	a proper database
all the relevant information	that service	the capability	data	proper provisions
validation	login	the end users	The primary purpose	this Customer Care Database
the expectations	the customers	regards	an array	services/queries
a wide variety	Financial Institutions	addition	it	customers
constructive solutions	a user-friendly and hassle-free experience	their desired query/service	This product	the customers
they	a cost-effective and flawless experience	respect	their service	It
the Service Providers	better insights	regards	the usability	their products
turn	them	their services/products	them	Users
information	their account/balance	their account	money	other valid accounts
addition	customers	their existing accounts	case	extraordinary events
New users	accounts	Other general queries	the purview	the system
Administrators	access	the entire database	order	information integrity
the database	Service Providers	a lower sense	privilege	regards
the Admin	information	the user	End Users	information
their own account	validation	Auditors	the entire database	faulty transactions
the admin	the bank	The owner	admin privileges	policy changes
customers	Every User class	different pages	respect	their functionality
five categories	users	His role	capacity planning	installation configuration
database design	exclusive tasks	the administrator	addition	they

the following functions	The service provider	access	the database	them
the customers	their queries	them	services	the service provider
certain fields	the database	the immediate needs	the user	he
she	the login credentials	the functions	a service provider	They
any access	the database	information	their own account	they
the entire database	the actions	a user	access	They
complete access	the entire database	the changes	the Service Provider	the Administrator
Their primary role	data integrity	faulty transactions	case	an issue
it	the Administrator	the Bank	the functions	an auditor
They	complete edit	access	the entire database	policy changes
its customers	information	certain queries	Scalability	the aforementioned database
an issue	the user pool	Distributed Database Systems	The data	a server
large amounts	data	it	cloud servers	an expensive affair
complete reconfiguration	the complete system	This section	the functional overview	the customer care database
Various functional modules	the user	The customer	an obligation	secrecy
regard	Username	Password	the Bank	The bank
valid Username	Password	a valid session	none	the customer
The customer	secrecy	regard	Username	Password
the Bank	The bank	valid Username	Password	a valid session
none	the customer	The customer	User ID	password
any other person	Any loss	the customer	non-compliance	this condition
his/her own risk	responsibility	the Bank	any manner	The login page
all the people	different page	different user classes	their functionality	This module
the balance details	a particular account	The updated database	every customer	bank
the balance information	every account	the database	the customer	This module
the customer	the profile	their account	It	them
their account status	load information	transaction details	This module	the customers
funds	one account	the same bank	the customer	a sufficient enough balance
online payment	bills	you	the unique bill number	the vendor
Customers	the bills	their account	A secure way	the billing
Online shopping	them	the easiest way	their items	the moment
the bank balance	the billing amount	It	the services	you
recurring payments	the internet connectivity	the customer care	you	recurring payment
you	It	your account	the moment	you
it	the moment	the bank balance	the billing amount	card payment
we	large amounts	no card payment machine	you	a cheque

it	a few payments	the cheque book	the bank	a new cheque book
you	it	it	you	a few days
This module	the customer	their account	their cards	new users
a new account	A customer	more than one bank account	a bank	this case
the customer	which account	money	these operations	customers
their owned bank accounts	it	the administrations	the system	It
the customer	his history	transactions	past 1-year transactions	It
him	the opportunity	his bank balance	needs	Bank staff
a record	it	transactions	the branch	it
the bank staff	the balance	a specific person	its record	the customer care services
the complete data	the customers bank accounts	the banks	that information	We
the service providers/admin	any account	any debit/credit card	the customer	no need
a confirmation	the bank	this issue	The bank	explicit authority
the customer care center	the information	the customers	highly scaled systems	the complete system

ALL VERBS

Verb	Verb	Verb	Verb	Verb
require	cater	may	operate	contain
pertain	possess	fetch	modify	meet
span	aim	provide	desire	aim
understand	query	ensure	enjoy	enable
gain	help	improve	make	can
regard	validate	can	transfer	can
block	exist	can	create	shall
cater	would	maintain	would	can
modify	provide	prompt	can	view
pertain	can	view	report	facilitate
log	will	redirect	include	perform
can	perform	follow	look	allow
help	resolve	provide	can	modify
cater	provide	provide	follow	can
perform	will	have	pertain	can
modify	look	follow	shall	view
incorporate	maintain	look	shall	notify
follow	can	perform	shall	view
facilitate	update	need	store	store
could	may	require	could	need

use	provide	can	access	maintain
register	presuppose	login	use	initiate
maintain	register	presuppose	login	use
initiate	should	keep	should	divulge
sustain	will	will	would	remain
log	will	come	accord	maintain
maintain	can	display	allow	view
update	allow	view	allow	transfer
can	type	pay	will	shop
pay	will	provide	will	provide
buy	sell	will	may	happen
use	would	require	recur	would
allow	set	recur	need	pay
will	deduct	cancel	will	prefer
need	pay	a	will	pay
go	issue	can	order	will
deliver	allow	suspend	block	allow
create	can	prompt	decide	use
debit	credit	can	add	own
will	approve	will	view	save
will	provide	maintain	will	search
update	need	will	will	check
update	assume	possess	regard	provide
assume	can	block	wish	regard
give	read	modify	scale	should
dedicate	maintain	handle	modify	open

TRUNCATED NOUNS

Nouns
Service Providers
Customers
the Bank
Users
The service provider
the vendor
Auditors
Administrators
credit card
debit card
online payment
Online shopping
Account details
recurring payment

REJECTED NOUNS

Noun	Reject Reason	Noun	Reject Reason	Noun	Reject Reason
all the people	General	various kinds	Vague	the customer care database	General
a cheque	Duplicates	valid Username	Attributes	issues	General
him	General	information	General	card payment	Associations
General	General	better insights	Irrelevant	certain fields	Vague
services/ products	General	customers	Duplicates	their account status	Attributes
new users	Duplicates	every customer	Duplicates	Other general queries	General
New users	Duplicates	this Customer Care Database	General	load information	General
the end users	Duplicates	we	General	the bank	Duplicates
the admin	Duplicates	the Service Provider	Attributes	Some classes	General
the customer care centre	General	the following functions	Irrelevant	a confirmation	Attributes
Financial Institutions	General	them	General	she	General
cloud servers	Irrelevant	a few payments	Attributes	that particular service	general
a server	General	the Service Providers	Attributes	constructive solutions	Associations
the user pool	General	Processor	Irrelevant	the purview	Vague
Bank staff	Attributes	1 GB	Irrelevant	the system	General
the customer	Duplicates	the opportunity	General	their service	Attributes
a bank	Duplicates	immediate needs	Attributes	a lower sense	Vague
A customer	Duplicates	aforementioned database	Vague	the services	Attributes
the administrations	Duplicates	faulty transactions	Attributes	validation	Attributes
an auditor	Duplicates	complete reconfiguration	Attributes	the moment	Vague
the complete data	General	other valid accounts	Attributes	any other person	Duplicates
a user	Duplicates	the balance details	Attributes	the profile	Attributes
responsibility	Vague	a new cheque book	Irrelevant	regard	Vague
large amounts	General	4 GB	Irrelevant	no need	Vague
The data	General	needs	General	transaction details	Attributes
the customer care	Associations	the Admin	Attributes	data	Attributes
which account	Vague	respect	Irrelevant	The login page	Attributes

these operations	Vague	access	Vague	case	Vague		
the login credentials	Attributes	microphone	Irrelevant	the information	Attributes		
services	General	their cards	Attributes	certain queries	Attributes		
a particular account	Vague	This module	Attributes	Password	Attributes		
their own account	General	Hard Drive	Irrelevant	the complete system	General		
a valid session	Associations	one account	General	the internet connectivity	Irrelevant		
1 GHz	Irrelevant	complete access	Attributes	database design	General		
his bank balance	Attributes	the bank staff	Irrelevant	the entire database	General		
Various functional modules	Vague	the database	General	the balance information	Attributes		
This product	Vague	a specific person	General	data integrity	Associations		
money	Associations	all the relevant information	Vague	Every User class	Vague		
secrecy	Associations	privilege	Irrelevant	that information	Vague		
any manner	Vague	services/queries	Duplicates	the customer care services	Associations		
the customers	Duplicates	the billing	Attributes	this condition	Vague		
information integrity	Associations	The owner	Duplicates	their account	Attributes		
the banks	Duplicates	transactions	Attributes	the functional overview	Vague		
their functionality	Vague	Minimum	Irrelevant	they	General		
password	Attributes	its record	Vague	different page	Vague		
The primary purpose	Vague	any access	Vague	the unique bill number	Attributes		
an obligation	Irrelevant	a service provider	Duplicates	a sufficient enough balance	Attributes		
extraordinary events	Vague	a user-friendly and hassle-free experience	Associations	a wide variety	Vague	past 1-year transactions	Attributes
The customer	Duplicates	Scalability	Associations	It	Duplicate	every account	Vague
complete edit	Associations	They	General	a few days	General	a record	General
the Administrator	Duplicate	its customers	Duplicates	accounts	Attributes	the expectations	Vague
funds	General	RAM	Irrelevant	queries	Duplicate	installation configuration	Vague
explicit authority	Vague	A secure way	Associations	We	General	this case	Vague

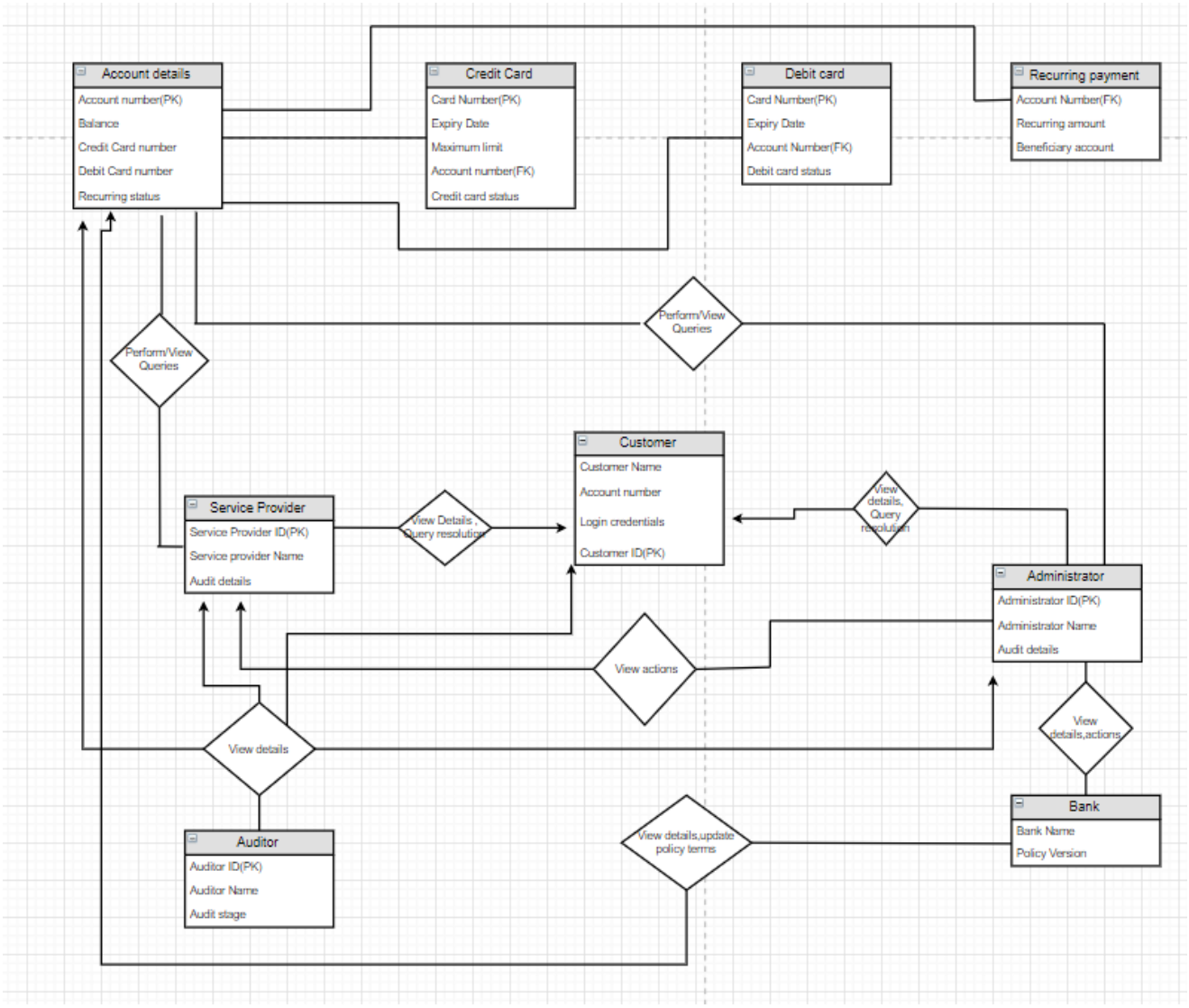
His role	Irrelevant	their items	Vague	the changes	General	recurring payments	Duplicates
admin privileges	Attributes	the same bank	Duplicates	the service providers/admin	Duplicate	their existing accounts	Attributes
the easiest way	Vague	any account	General	none	General	the bank balance	Attributes
their products	Vague	the usability	Associations	Their primary role	Vague	the bills	Attributes
This section	General	the functions	Vague	policy changes	Attributes	exclusive tasks	General
highly scaled systems	Irrelevant	Any loss	Vague	bank	Duplicate	regards	Vague
The bank	General	Customer Care services	Duplicates	a proper database	General	your account	Attributes
you	General	the user	Duplicates	Memory	Irrelevant	no card payment machine	Irrelevant
the branch	Attributes	capacity planning	General	Distributed Database Systems	General	proper provisions	General
an array	Vague	the customer's bank accounts	Attributes	Username	Duplicate	he	General
a new account	Attributes	The updated database	Associations	the administrator	Duplicate	different user classes	General
User ID	Attributes	a camera	Irrelevant	the balance	Attributes	different pages	Vague
login	Attributes	his history	General	End Users	Duplicate	their account/balance	Attributes
bills	Attributes	more than one bank account	Irrelevant	the capability	Irrelevant	that service	General
the billing amount	Attributes	the cheque book	Irrelevant	their desired query/service	Duplicate	an expensive affair	Vague
his/her own risk	Irrelevant	their queries	General	64 GB	Irrelevant	turn	Vague
it	General	the service provider	Duplicates	cost-effective and flawless experience	Associations	this issue	Attributes

TRUNCATED VERBS

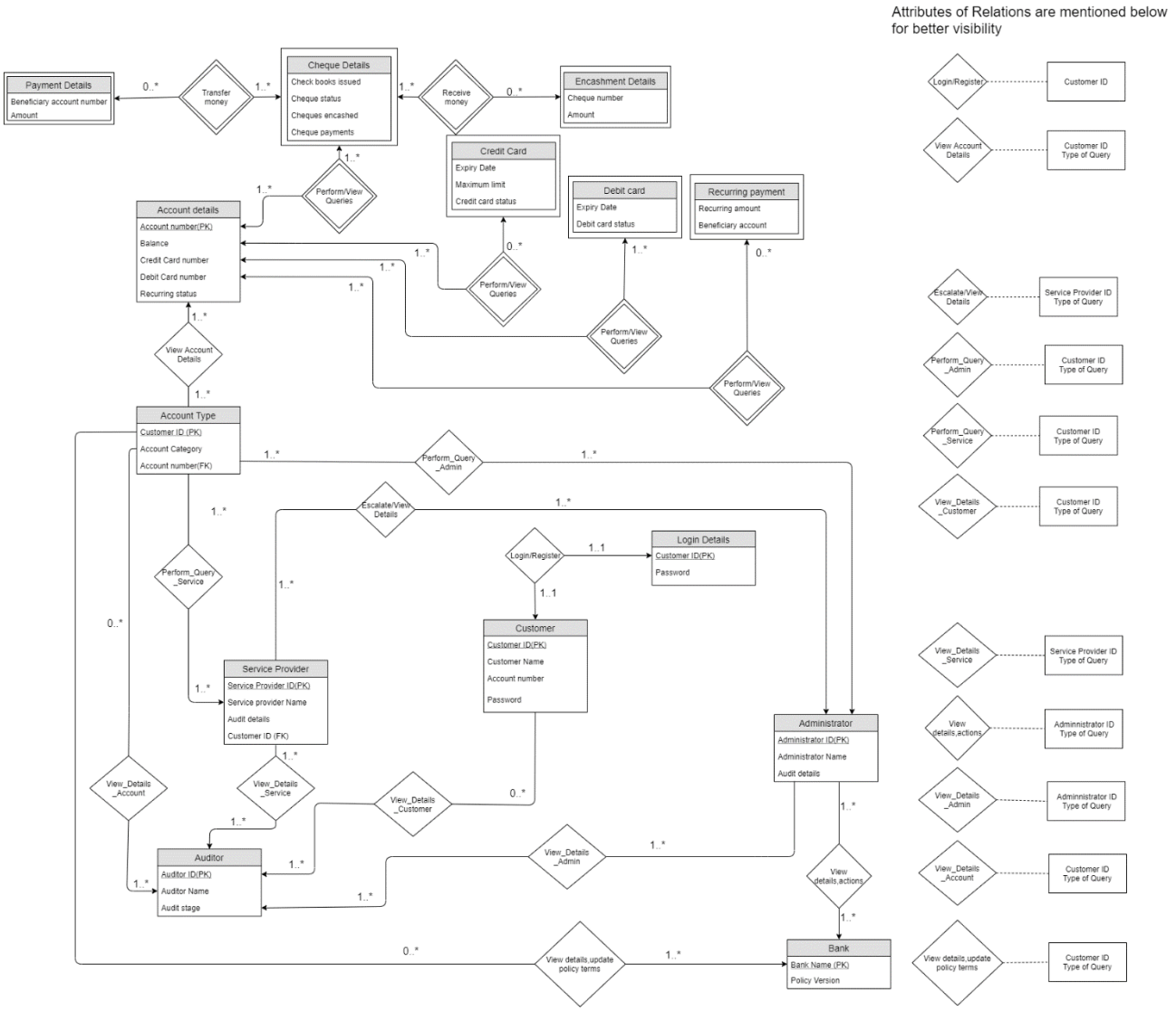
Verb	Verb	Verb	Verb	Verb
span	enjoy	cancel	make	can
accord	come	pay	own	add
remain	buy	pertain	prefer	order
facilitate	type	follow	recur	meet
keep	happen	resolve	debit	approve
notify	presuppose	exist	desire	fetch
scale	perform	query	understand	recommend
deliver	require	maintain	incorporate	shall
view	transfer	regard	login	wish
aim	register	create	need	have
improve	redirect	enable	use	access
update	issue	contain	sell	possess
ensure	sustain	cater	modify	validate
â€™	block	suspend	prompt	may
assume	initiate	operate	read	should
credit	display	deduct	go	include
shop	report	help	search	give
log	look	save	handle	would
check	dedicate	will	set	allow
store	could	decide	gain	provide

FINAL ER DIAGRAM ALL VERSIONS

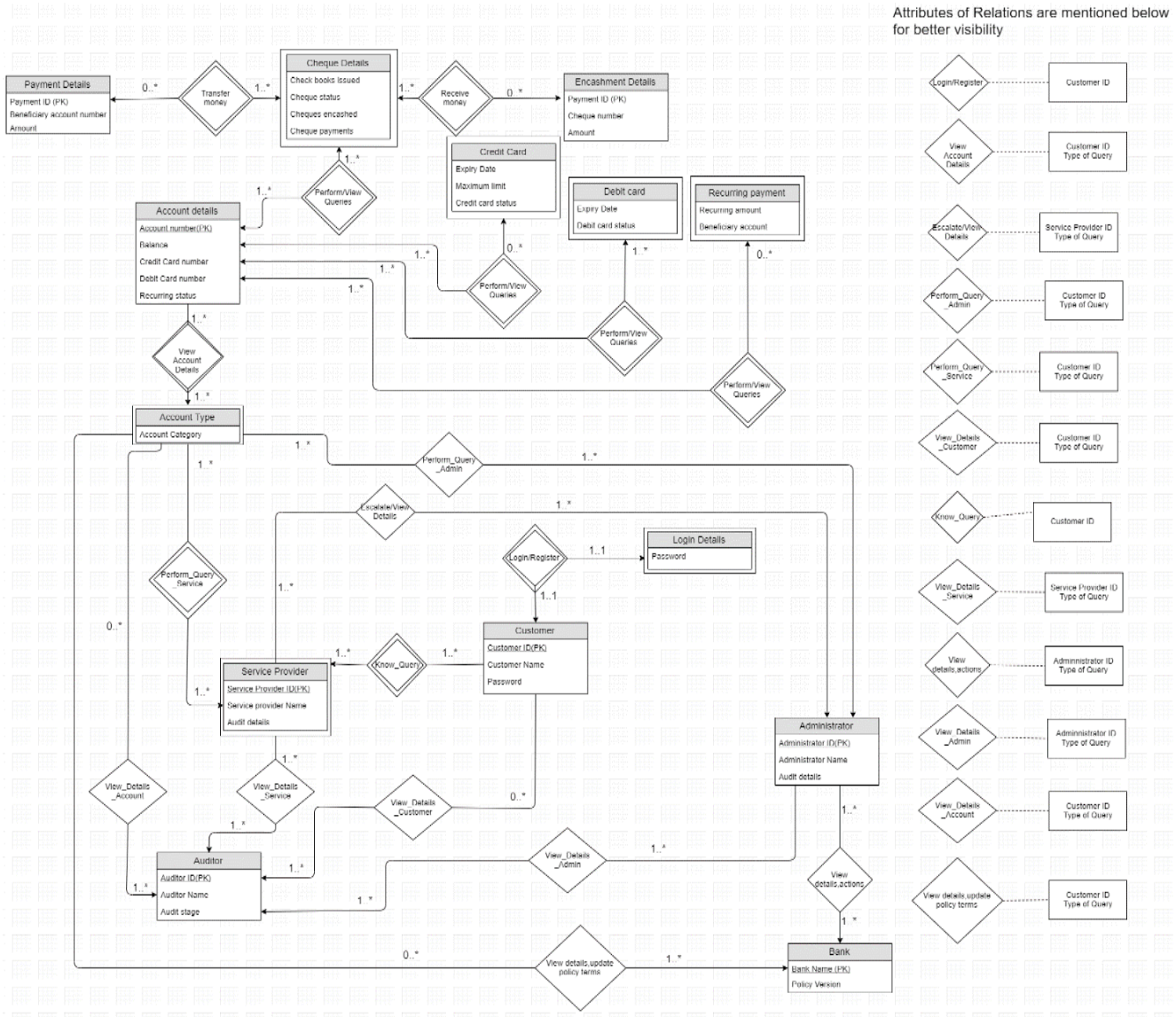
VERSION 1



VERSION 2

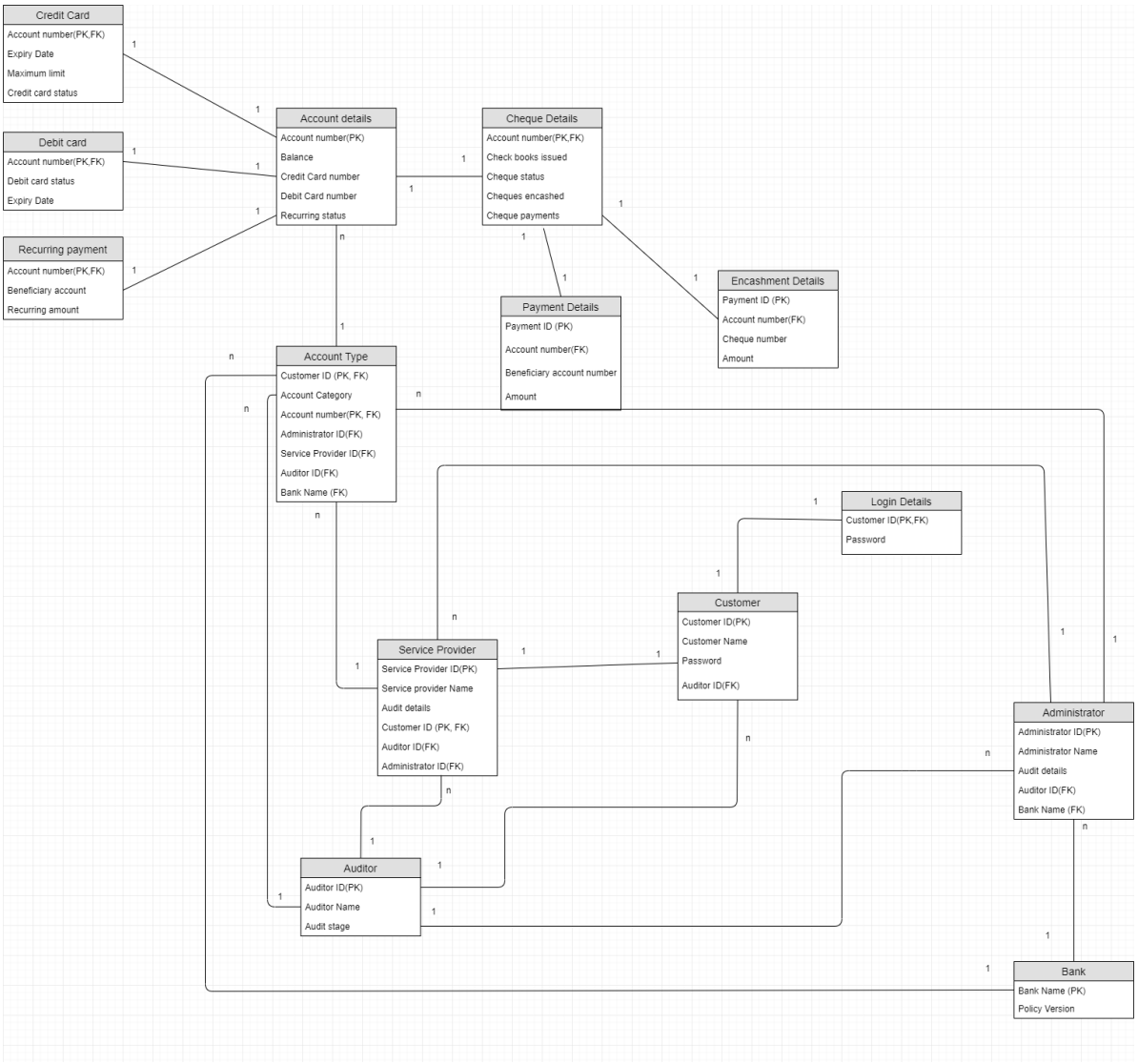

Attributes of Relations are mentioned below for better visibility
Payment Details
Beneficiary account number
Amount
Transfer money
Cheque Details
Check books issued
Cheque status
Cheques encashed
Cheque payments
Receive money
Encashment Details
Cheque number
Amount
Credit Card
Expiry Date
Maximum limit
Credit card status
Debit card
Expiry Date
Debit card status
Recurring payment
Recurring amount
Beneficiary account
Perform/View Queries
Account details
Account number(PK)
Balance
Credit Card number
Debit Card number
Recurring status
View Account Details
Account Type
Customer ID (PK)
Account Category
Account number(FK)
Perform_Query_Admin
Escalate/View Details
Perform_Query_Service
Login/Register
Login Details
Customer ID(PK)
Password
Customer
Customer ID(PK)
Customer Name
Account number
Password
Service Provider
Service Provider ID(PK)
Service provider Name
Audit details
Customer ID (FK)
Administrator
Administrator ID(PK)
Administrator Name
Audit details
View_Details_Account
View_Details_Service
View_Details_Customer
View_Details_Admin
Auditor
Auditor ID(PK)
Auditor Name
Audit stage
View details,actions
View details,update policy terms
Bank
Bank Name (PK)
Policy Version
Login/Register
Customer ID
View Account Details
Customer ID Type of Query
Escalate/View Details
Service Provider ID Type of Query
Perform_Query_Admin
Customer ID Type of Query
Perform_Query_Service
Customer ID Type of Query
View_Details_Customer
Customer ID Type of Query
View_Details_Service
Service Provider ID Type of Query
View details,actions
Adminnistrator ID Type of Query
View_Details_Admin
Adminnistrator ID Type of Query
View_Details_Account
Customer ID Type of Query
View details,update policy terms
Customer ID Type of Query

VERSION 3


Attributes of Relations are mentioned below for better visibility
Payment Details
Payment ID (PK)
Beneficiary account number
Amount
Transfer money
Cheque Details
Check books issued
Cheque status
Cheques encashed
Cheque payments
Receive money
Encashment Details
Payment ID (PK)
Cheque number
Amount
Credit Card
Expiry Date
Maximum limit
Credit card status
Debit card
Expiry Date
Debit card status
Recurring payment
Recurring amount
Beneficiary account
Perform/View Queries
Account details
Account number(PK)
Balance
Credit Card number
Debit Card number
Recurring status
View Account Details
Account Type
Account Category
Perform_Query_Admin
Escalate/View Details
Perform_Query_Service
Login/Register
Login Details
Password
Customer
Customer ID(PK)
Customer Name
Password
Know_Query
Service Provider
Service Provider ID(PK)
Service provider Name
Audit details
Administrator
Administrator ID(PK)
Administrator Name
Audit details
View_Details_Account
View_Details_Service
View_Details_Customer
View_Details_Admin
Auditor
Auditor ID(PK)
Auditor Name
Audit stage
View details,actions
View details,update policy terms
Bank
Bank Name (PK)
Policy Version
Login/Register
Customer ID
View Account Details
Customer ID Type of Query
Escalate/View Details
Service Provider ID Type of Query
Perform_Query_Admin
Customer ID Type of Query
Perform_Query_Service
Customer ID Type of Query
View_Details_Customer
Customer ID Type of Query
Know_Query
Customer ID
View_Details_Service
Service Provider ID Type of Query
View details,actions
Adminnistrator ID Type of Query
View_Details_Admin
Adminnistrator ID Type of Query
View_Details_Account
Customer ID Type of Query
View details,update policy terms
Customer ID Type of Query

CONVERSION OF FINAL ER DIAGRAM TO RELATIONAL MODEL

RELATIONAL SCHEMA



NORMALISATION AND SCHEMA REFINEMENT

For the removal of redundancies, for most of the cases, i.e. many of the tables had a single primary key and so there were no redundancies in that. For two tables that are Encashment details, and Account details, we removed the redundancies while removing and modifying the DDL and schema accordingly. We analysed each and every table this way.
There are primarily three kinds of anomalies we need to look upon while removing the redundancy. These are insertion anomaly, deletion anomaly, and updating anomaly.
For any table, a primary key can never be NULL and so if the table is not reduced to its highest form, it may cause insertion anomaly wherein no new data could be inserted in the database unless there's a corresponding primary key associated to it or not. Similarly, if we want to update any data, it may so happen that due to redundancy we may end up updating the same piece of information in thousands of rows. If our table has redundancy, then deletion of one tuple may delete the corresponding data even though it was not supposed to be.

NORMALISATION

Credit Card: (BCNF)

- Primary Key : Account Number
- Foreign Key : Account Number
- Functional Dependency

  Account Number →Maximum Limit ,Credit Card Status, Expiry Date

Since we have atomic attributes, hence it is in first normal form. Furthermore we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Debit Card (BCNF)

- Primary Key : Account Number
- Foreign Key : Account Number
- Functional Dependency

  Account Number →Debit Card Status, Expiry Date

Since we have atomic attributes, hence it is in first normal form. Furthermore we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well

Recurring Payment (BCNF)

- Primary Key : Account Number
- Foreign Key : Account Number
  Functional Dependency

  Account Number →Beneficiary account, Recurring amount

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Login Details (BCNF)

- Primary Key : Customer ID
- Foreign Key : Customer ID
- Functional Dependency

  Customer ID → Password

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well

Payment Details (BCNF)

- Primary Key : Payment ID
- Foreign Key : Account Number
- Functional Dependency

  Payment ID → Account Number, Beneficiary Account Number, Amount

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well

Account details (BCNF)

- Primary Key : Account Number
- Account Number is Primary Key
- Credit Card Number and Debit Card Number are Candidate keys
- Functional Dependency

  Account Number → Balance, Credit Card number, Debit Card number, Recurring Status
  Credit Card number→ Balance, Debit Card number, Recurring Status, Account Number
  Debit Card number→ Balance, Recurring Status, Account Number, Credit Card number

  (Account Number, Credit Card number) - 1
  (Account Number, Debit Card number,) -2

(Account Number, Balance Recurring Status) - 3

We have applied Heath's theorem for transforming a non-BCNF table to a BCNF table.
Let us take an example for explaining the heath's theorem where we have Initialized S = {R}
While S has a relation R' that is not in BCNF do:
Pick a FD: X->Y that holds in R' and violates BCNF
Add the relation XY to S
Update R' = R'-Y
Return S
So, now if s={ABCDE}
S = {ACDE, AB} // Pick FD: A->B which violates BCNF
S = {ACE, AB, CD} // Pick FD: C->D which violates BCNF
// Return S as all relations are in BCNF

Bank (BCNF)

- PRIMARY KEY:- Bank Name
- Functional Dependency

Bank Name → Policy Version

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Auditor (BCNF)

- Primary Key : Auditor ID
- Functional Dependency

Auditor ID →Auditor Name, Audit Stage

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Encashment Details (BCNF)

- Primary Key : Payment ID
- Foreign Key : Account Number
- Functional Dependency

Payment ID→ Cheque Number, Amount, Account_number
Cheque Number → Payment ID, Amount, , Account_number

(Payment ID, Cheque Number, Account_number) -1
(Payment ID, Amount)-2

We have applied Heath's theorem for transforming a non-BCNF table to a BCNF table.
Let us take an example for explaining the heath's theorem where we have Initialized S = {R}

While S has a relation R' that is not in BCNF do:

Pick a FD: X->Y that holds in R' and violates BCNF

Add the relation XY to S

Update R' = R'-Y

Return S

So, now if s={ABCDE}

S = {ACDE, AB} // Pick FD: A->B which violates BCNF

S = {ACE, AB, CD} // Pick FD: C->D which violates BCNF

// Return S as all relations are in BCNF

Cheque details (BCNF)

- Primary Key : Account Number
- Foreign Key : Account Number
- Functional Dependency

Account Number → Cheques encashed, Cheque payments, Check books issued, Cheque Status

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Service Provider (BCNF)

- PRIMARY KEY:- Service Provider ID, Customer ID
- FOREIGN KEY:- Customer ID, Administrator ID, Auditor ID
- Functional Dependency

(Service Provider ID, Customer ID) →Service Provider Name, Audit Details, Administrator ID, Auditor ID

Since we have atomic attributes, hence it is in first normal form. Furthermore, in none of the dependencies the non-prime attributes don't depend on a proper subset and hence there's no partial dependency and therefore it's in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Administrator (BCNF)

- PRIMARY KEY:- Administrator ID
- FOREIGN KEY:- Bank Name, Auditor ID
- Functional Dependency

Administrator ID →Administrator Name, Bank Name, Auditor ID,Audit Details

Since we have atomic attributes, hence it is in first normal form. Furthermore, we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Account Type (BCNF)

- Primary Key : Account Number,Customer ID
- Foreign Key : Account Number, Administrator ID, Service Provider ID, Auditor ID, Bank Name
- Functional Dependency

(Account Number,Customer ID)→ Account Category,Administrator ID, Service Provider ID, Bank Name , Auditor ID

Since we have atomic attributes, hence it is in first normal form. Furthermore, in none of the dependencies the non-prime attributes don't depend on a proper subset and hence there's no partial dependency and therefore it's in second normal form. Since there is one candidate key, hence it is in BCNF as well.

Customer (BCNF)

- Primary Key : Customer ID
- Foreign Key : Auditor ID
- Functional Dependency

Customer ID →Customer Name , Password, Auditor ID

Since we have atomic attributes, hence it is in first normal form. Furthermore we have a single attribute primary key, hence it is in second normal form. Since there is one candidate key, hence it is in BCNF as well.

SQL: FINAL DDL SCRIPTS, INSERT STATEMENTS, 40 SQL QUERIES, SNAPSHOTS OF OUTPUT OF EACH QUERY

FINAL DDL SCRIPTS

```
CREATE TABLE Auditor(
        Auditor_ID CHAR(20) NOT NULL,
        Auditor_Name CHAR(20),
        Audit_stage INT,
        PRIMARY KEY (Auditor_ID))

CREATE TABLE Customer(
        Customer_ID CHAR(20) NOT NULL,
        Customer_Name CHAR(20),
        Account_number CHAR(20),
        Pass_word CHAR(20),
```

```
        Auditor_ID CHAR(20),
        PRIMARY KEY (Customer_ID),
        FOREIGN KEY (Auditor_ID) REFERENCES Auditor
        ON DELETE CASCADE)

CREATE TABLE Login_Details(
        Customer_ID CHAR(20) NOT NULL,
        Pass_word CHAR(20),
        PRIMARY KEY(Customer_ID),
        FOREIGN KEY (Customer_ID) REFERENCES Customer
        ON DELETE CASCADE)

CREATE TABLE Bank(
        Bank_Name CHAR(20) NOT NULL,
        Policy_Version numeric(4,2),
        PRIMARY KEY(Bank_Name))

CREATE TABLE Administrator(
        Audit_Details NUMERIC(4,2),
        Auditor_ID CHAR(20),
        Administrator_ID CHAR(20) NOT NULL,
        Administrator_Name CHAR(20),
        BANK_NAME CHAR (20),
        PRIMARY KEY(Administrator_ID),
        FOREIGN KEY (Bank_Name) REFERENCES Bank ON DELETE CASCADE,
        FOREIGN KEY (Auditor_ID) REFERENCES Auditor ON DELETE CASCADE)

CREATE TABLE  Service_Provider(
        Service_Provider_ID CHAR(20) NOT NULL,
        Service_Provider_Name CHAR(20),
        Audit_Details NUMERIC(4,2),
        Auditor_ID CHAR(20),
        Administrator_ID CHAR(20),
        Customer_ID CHAR(20),
        PRIMARY KEY(Service_Provider_ID, Customer_ID),
        FOREIGN KEY (Customer_ID ) REFERENCES customer ON DELETE CASCADE,
        FOREIGN KEY (Administrator_ID) REFERENCES Administrator ON DELETE CASCADE,
        FOREIGN KEY (Auditor_ID) REFERENCES Auditor ON DELETE CASCADE)

CREATE TABLE Account_Details_1(
        Account_Number CHAR(20) NOT NULL,
        Credit_Card_number INTEGER,
        PRIMARY KEY (Account_Number))

CREATE TABLE Account_Details_2(
        Account_Number CHAR(20) NOT NULL,
```

```
    Debit_Card_number INTEGER,
    PRIMARY KEY (Account_Number))

CREATE TABLE Account_Details_3(
    Account_Number CHAR(20) NOT NULL,
    Balance NUMERIC(8,2),
    Recurring_Status BOOLEAN,
    PRIMARY KEY (Account_Number))

CREATE TABLE Account_type(
    Customer_ID CHAR(20) NOT NULL,
    Account_Category CHAR(20),
    Account_number CHAR(20),
    Administrator_ID CHAR(20),
    Service_Provider_ID CHAR(20),
    Auditor_ID CHAR(20),
    Bank_Name CHAR(20),
    PRIMARY KEY (Customer_ID, Account_number),
    FOREIGN KEY (Account_number) REFERENCES Account_Details_1 ON DELETE CASCADE,
    FOREIGN KEY (Administrator_ID) REFERENCES Administrator ON DELETE CASCADE,
    FOREIGN KEY (Service_Provider_ID, Customer_ID) REFERENCES Service_Provider ON DELETE
    CASCADE,
    FOREIGN KEY (Auditor_ID) REFERENCES Auditor ON DELETE CASCADE,
    FOREIGN KEY (Bank_Name) REFERENCES Bank ON DELETE CASCADE)

CREATE TABLE Credit_card(
    Account_Number CHAR(20) NOT NULL,
    Expiry_Date DATE,
    Maximum_Limit INTEGER,
    Credit_Card_Status BOOLEAN,
    PRIMARY KEY (Account_Number),
    FOREIGN KEY (Account_Number) REFERENCES Account_Details_1
    ON DELETE CASCADE)

CREATE TABLE Debit_card(
    Account_Number CHAR(20) NOT NULL,
    Expiry_Date DATE,
    Debit_Card_Status BOOLEAN,
    PRIMARY KEY (Account_Number),
    FOREIGN KEY (Account_Number) REFERENCES Account_Details_2
    ON DELETE CASCADE)

CREATE TABLE Recurring_payment(
    Account_Number CHAR(20) NOT NULL,
    Beneficiary_account CHAR(20),
    Recurring_amount INTEGER,
```

```
        PRIMARY KEY (Account_Number),
        FOREIGN KEY (Account_Number) REFERENCES Account_Details_3
        ON DELETE CASCADE)

CREATE TABLE Cheque_details(
        Account_Number CHAR(20) NOT NULL,
        Cheques_encashed INTEGER,
        Cheque_payments NUMERIC(8,2),
        Check_books_issues INTEGER,
        Cheque_Status BOOLEAN,
        PRIMARY KEY (Account_Number),
        FOREIGN KEY (Account_Number) REFERENCES Account_Details_1
        ON DELETE CASCADE)

CREATE TABLE Encashment_Details_1(
        Payment_ID CHAR(20) NOT NULL,
        Account_Number CHAR(20) NOT NULL,
        Cheque_Number CHAR(20),
        PRIMARY KEY (Payment_ID),
        FOREIGN KEY (Account_Number) REFERENCES Cheque_Details
        ON DELETE CASCADE)

CREATE TABLE Encashment_Details_2(
        Payment_ID CHAR(20) NOT NULL,
        Amount INTEGER,
        PRIMARY KEY (Payment_ID))

CREATE TABLE Payment_Details(
        Account_Number CHAR(20) ,
        Payment_ID CHAR (20) NOT NULL,
        Beneficiary_Account_Number CHAR(20),
        Amount INTEGER,
        PRIMARY KEY (Payment_ID),
        FOREIGN KEY (Account_Number) REFERENCES Cheque_Details
        ON DELETE CASCADE)
```

INSERT STATEMENTS

Auditor CSV insertion

```
COPY auditor(Auditor_ID,Auditor_Name,Audit_stage)
FROM 'C:\Users\Public\auditor.csv'
DELIMITER '|'
CSV HEADER;
```

Customer CSV insertion

```
COPY customer(Customer_ID,Customer_Name,Account_number,Pass_word,Auditor_ID)
FROM 'C:\Users\Public\Customer.csv'
DELIMITER ','
CSV HEADER;
```

Login details CSV insertion

```
COPY login_details(Customer_ID,Pass_word)
FROM 'C:\Users\Public\login.csv'
DELIMITER ','
CSV HEADER;
```

Bank

```
COPY Bank(Bank_name,Policy_version)
FROM 'C:\Users\Public\bank.csv'
DELIMITER ','
CSV HEADER;
```

Administrator

```
COPY Administrator(Audit_Details,Auditor_ID,Administrator_ID,Administrator_Name,BANK_NAME)
FROM 'C:\Users\Public\administrator.csv'
DELIMITER ','
CSV HEADER;
```

Service_Provider

```
COPY Service_Provider(
Service_Provider_ID,Service_Provider_Name,Audit_Details,Auditor_ID,Administrator_ID,Customer_ID)
FROM 'C:\Users\Public\service_provider.csv'
DELIMITER ','
CSV HEADER;
```

Account_Details_1

```
COPY Account_Details_1(Account_Number,Credit_Card_number)
FROM 'C:\Users\Public\account_details_1.csv'
DELIMITER ','
CSV HEADER;
```

Account_Details_2

```
COPY Account_Details_2(Account_Number,Debit_Card_number)
FROM 'C:\Users\Public\account_details_2.csv'
DELIMITER ','
CSV HEADER;
```

Account_Details_3

```
COPY Account_Details_3(Account_Number,Balance,Recurring_Status)
FROM 'C:\Users\Public\account_details_3.csv'
DELIMITER ','
CSV HEADER;
```

Account_type

```
COPY Account_type(
Customer_ID,Account_Category,Account_number,Administrator_ID,Service_Provider_ID,Auditor_ID,Bank_Name)
FROM 'C:\Users\Public\Account_type.csv'
DELIMITER ','
CSV HEADER;
```

Credit_Card

```
COPY Credit_card(Account_Number,Expiry_Date,Maximum_Limit,Credit_Card_Status)
FROM 'C:\Users\Public\credit_card.csv'
DELIMITER ','
CSV HEADER;
```

Debit_card

```
COPY Debit_card(Account_Number,Expiry_Date,Debit_Card_Status)
FROM 'C:\Users\Public\debit_card.csv'
DELIMITER ','
CSV HEADER;
```

Recurring_payment

```
COPY Recurring_payment(Account_Number,Beneficiary_account,Recurring_amount)
FROM 'C:\Users\Public\recurring_payment.csv'
DELIMITER ','
CSV HEADER;
```

Cheque_details

```
COPY Cheque_details(Account_Number,Cheques_encashed,Cheque_payments,Check_books_issues,Cheque_Status)
FROM 'C:\Users\Public\cheque_details.csv'
DELIMITER ','
CSV HEADER;
```

Encashment_details_2

```
COPY encashment_details_2(Payment_ID, Amount)
FROM 'C:\Users\Public\encashment_details_2.csv'
DELIMITER ','
CSV HEADER;
```

Encashment_details_1

```
COPY encashment_details_1(Payment_ID,Account_Number,Cheque_Number)
FROM 'C:\Users\Public\encashment_details_1.csv'
DELIMITER ','
CSV HEADER;
```

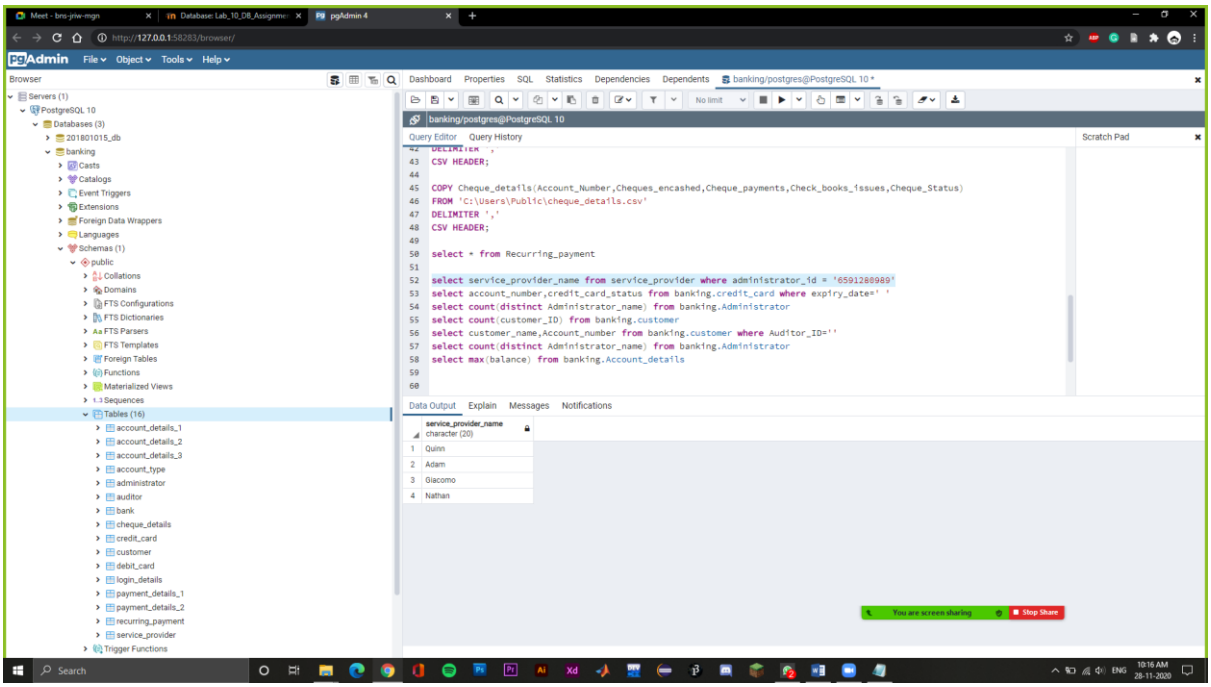
Payment_details

```
COPY payment_details(Account_Number,Payment_ID,Beneficiary_Account_Number,Amount)
FROM 'C:\Users\Public\payment_details.csv'
DELIMITER ','
CSV HEADER;
```

40 SQL QUERIES

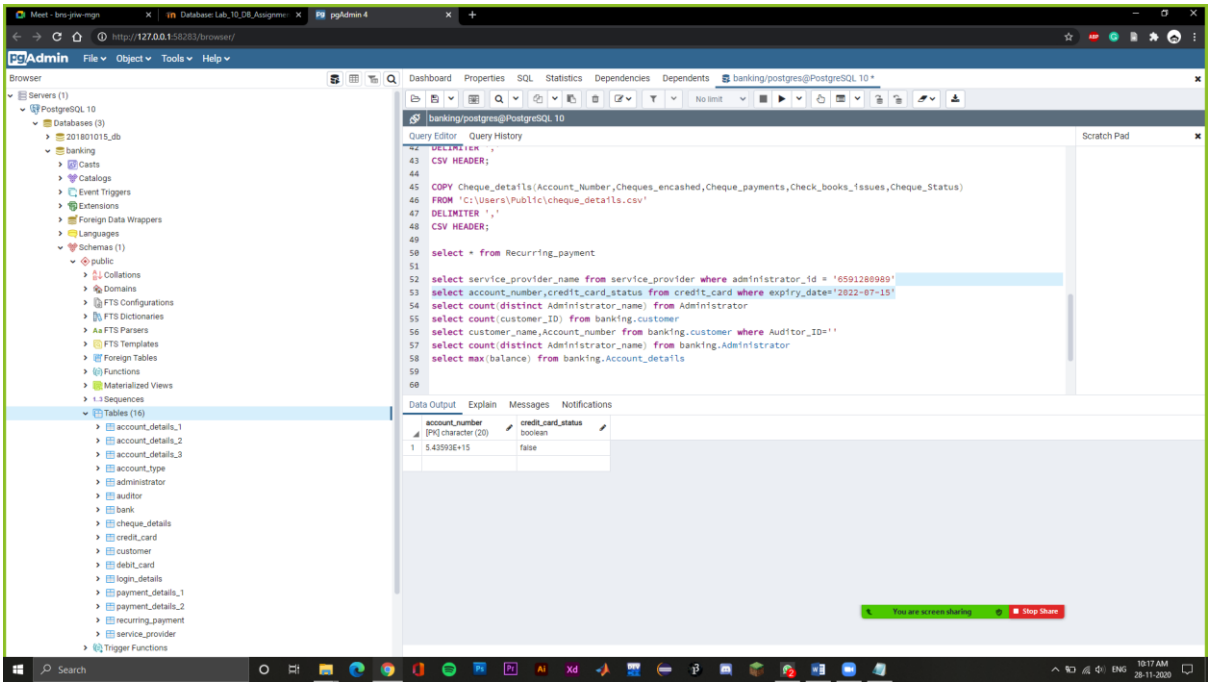
select service_provider_name from service_provider where administrator_id = '6591280989'

-> Here, we are selecting the name of the service provider for the administrator id 6591280989.



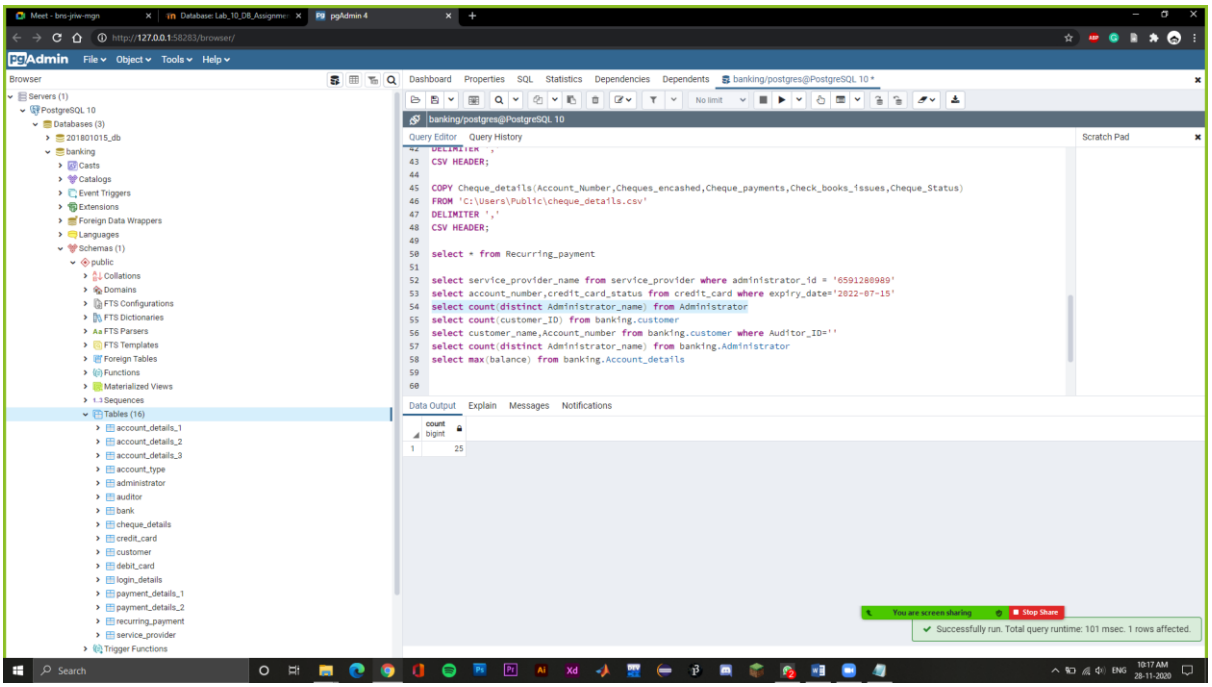
select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'

-> Get the account number and the credit card status for the credit card whose expiry date is 15-07-2022.



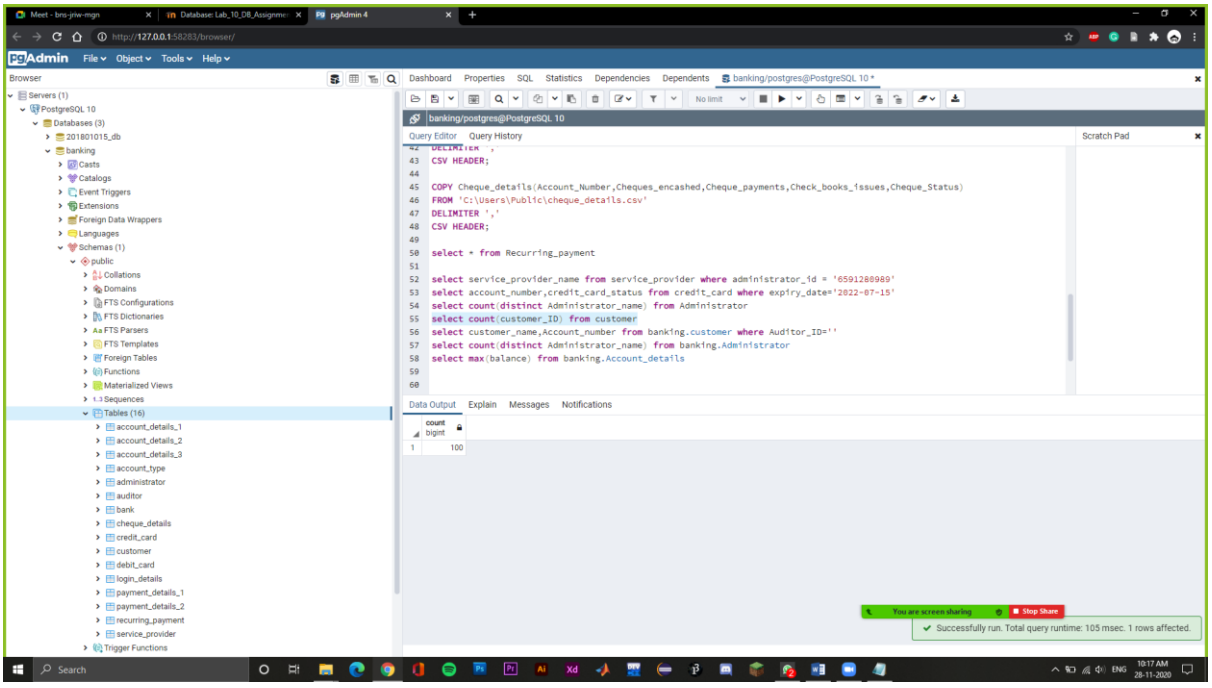
select count(distinct Administrator_name) from Administrator

-> Here, we are giving the total number of distinct administrators.



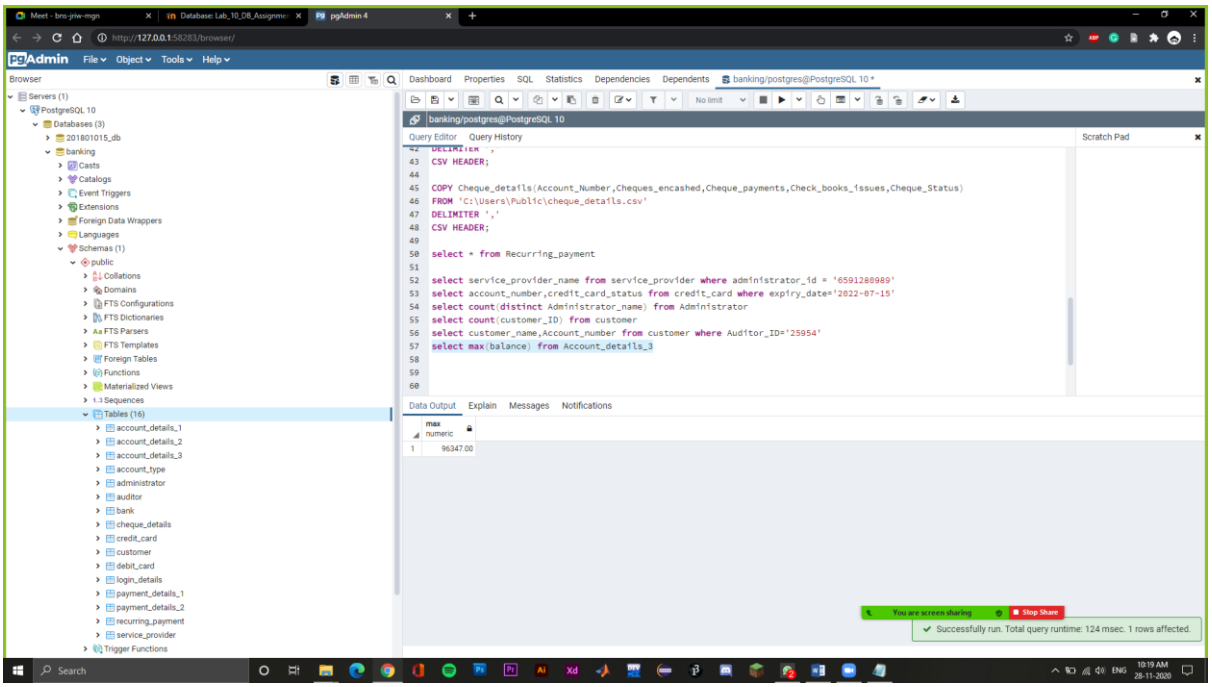
select count(customer_ID) from customer

-> Here, we are calculating the total number of customers.



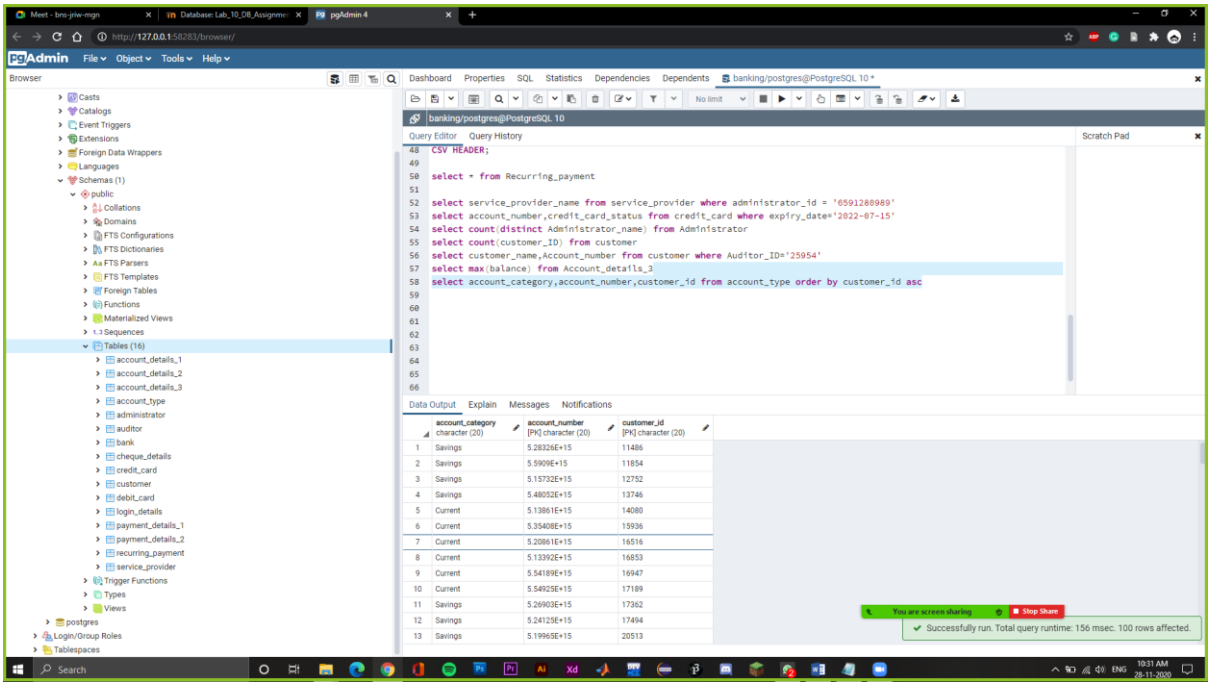
select max(balance) from Account_details_3

-> Here, we are printing the maximum balance of all the bank accounts in the bank.



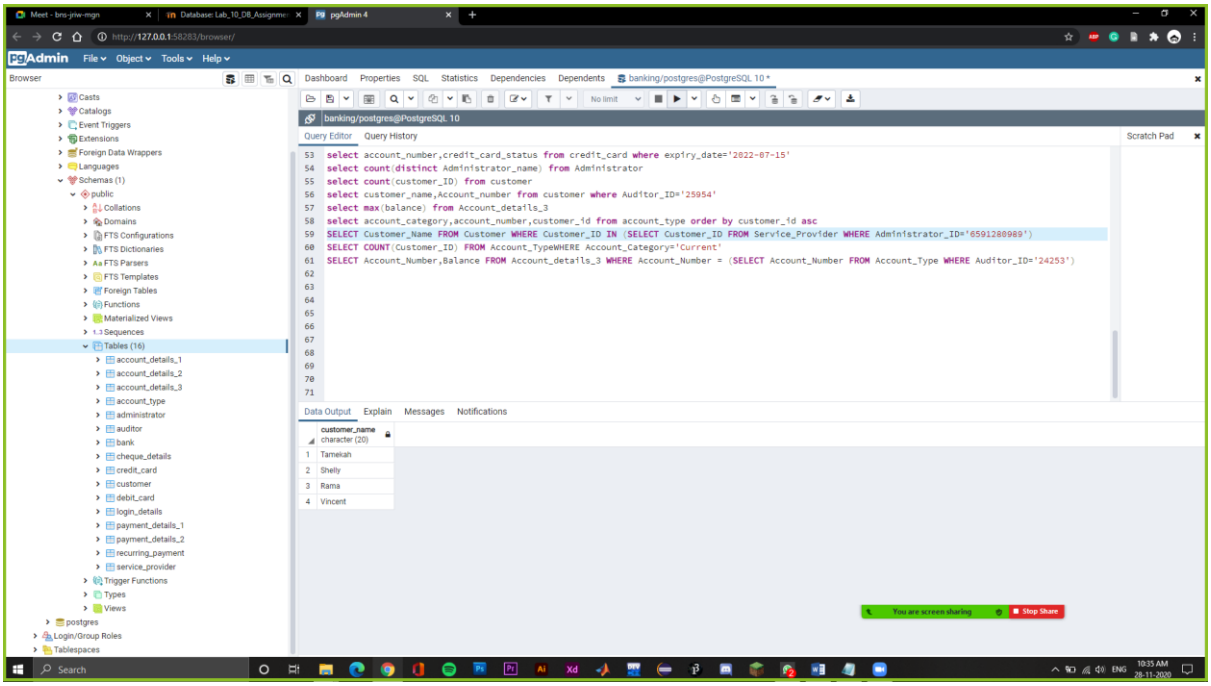
select account_category,account_number,customer_id from account_type order by customer_id asc

-> Here, we are printing the account category, account number,customer id and arranging them on the basics of customer id from smallest to largest order.



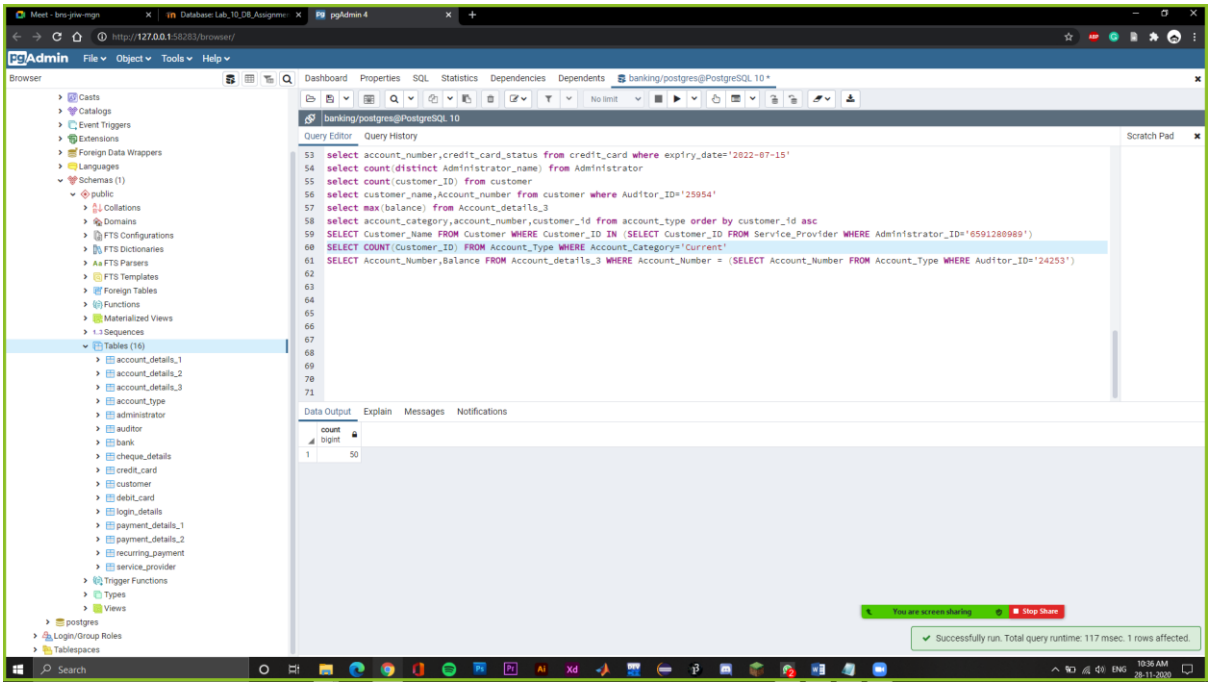
SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591280989')

-> Show customer name who have administrator id as 6591280989



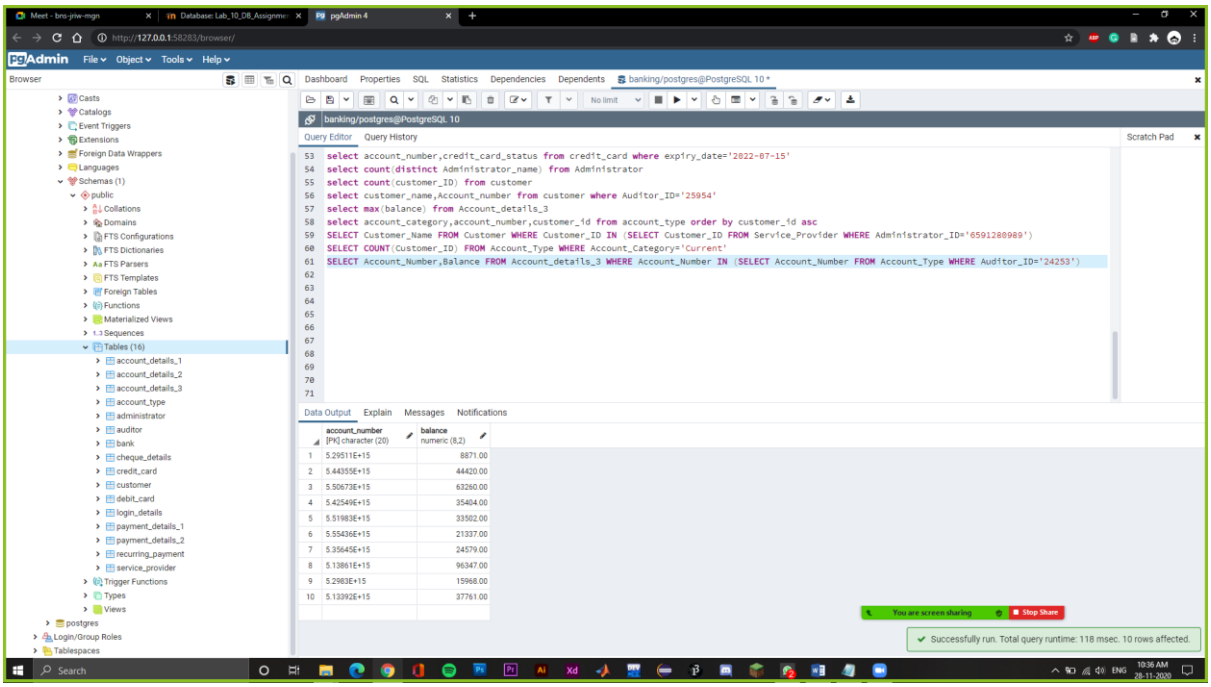
SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Category='Current'

->Show number of customers who have account category as current.



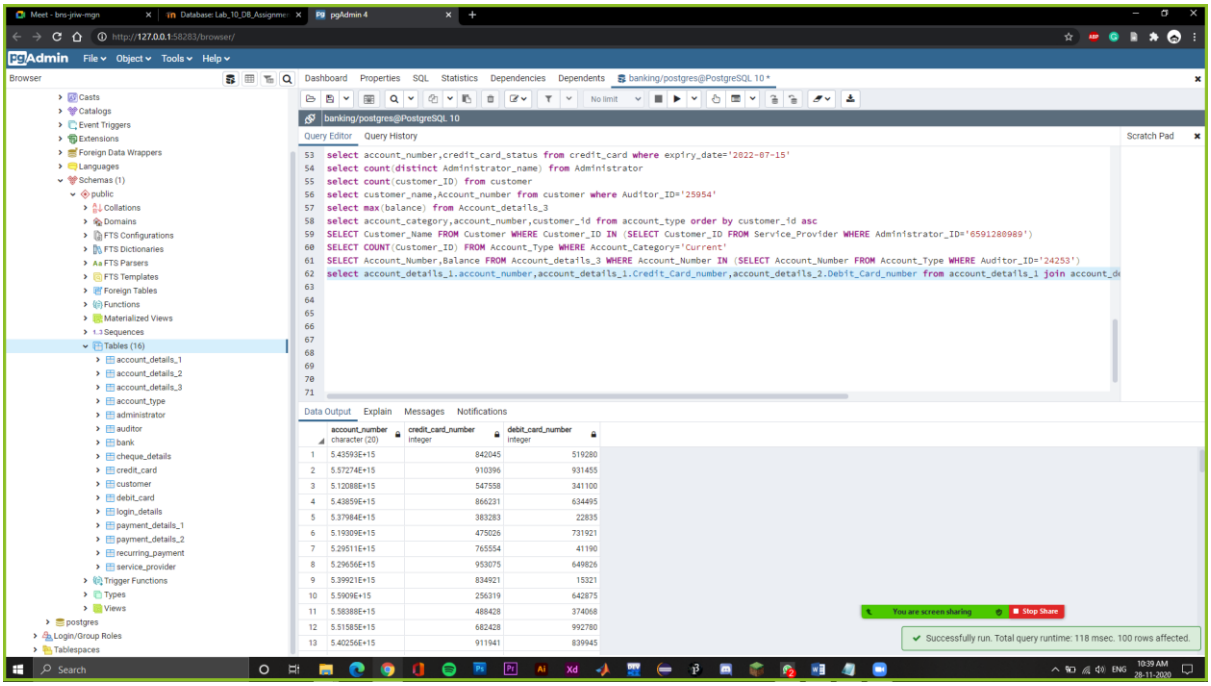
SELECT Account_Number,Balance FROM Account_details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')

-> Show account number and balance of customers who have auditor id as 24253



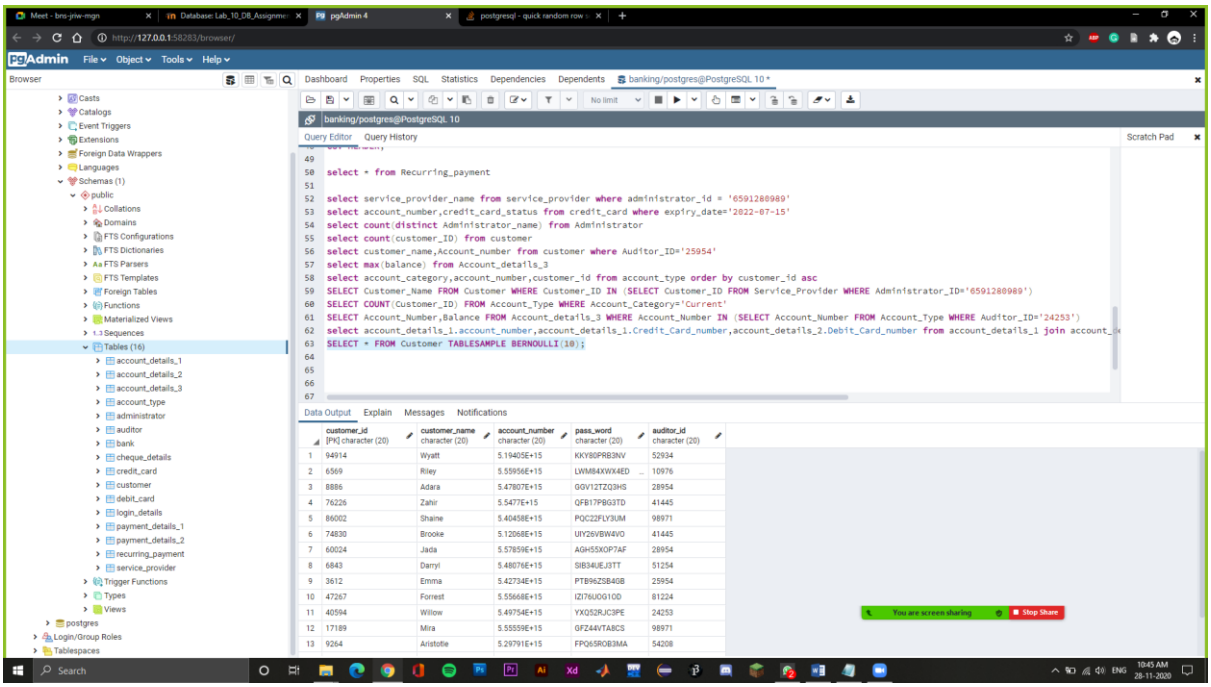
Select account_details_1.account_number, account_details_1.Credit_Card_number, account_details_2.Debit_Card_number from account_details_1 join account_details_2 on account_details_1.account_number=account_details_2.account_number

-> Show account number, credit card number and debit card number of all the customers



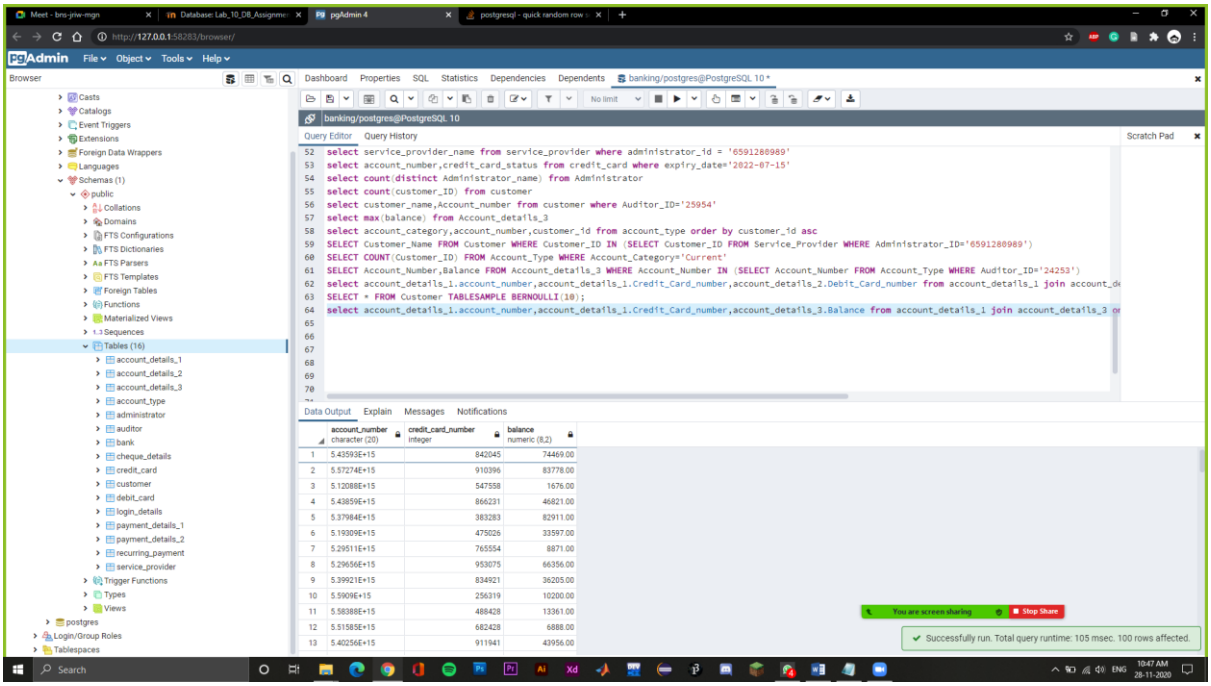
SELECT * FROM Customer TABLESAMPLE BERNOULLI(10);

->10 random values from customer based on Bernoulli distribution



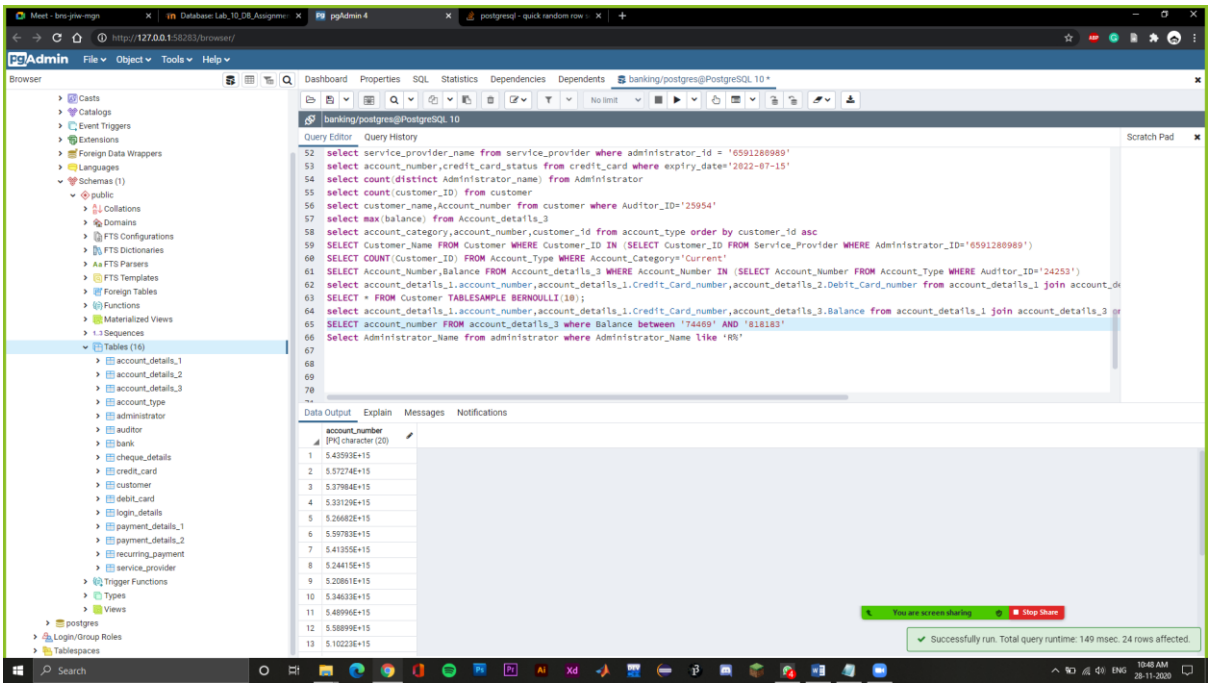
select account_details_1.account_number,account_details_1.Credit_Card_number,account_details_3.Balance from account_details_1 join account_details_3 on account_details_1.account_number=account_details_3.account_number

->Show account number, credit card number and balance of all the customers



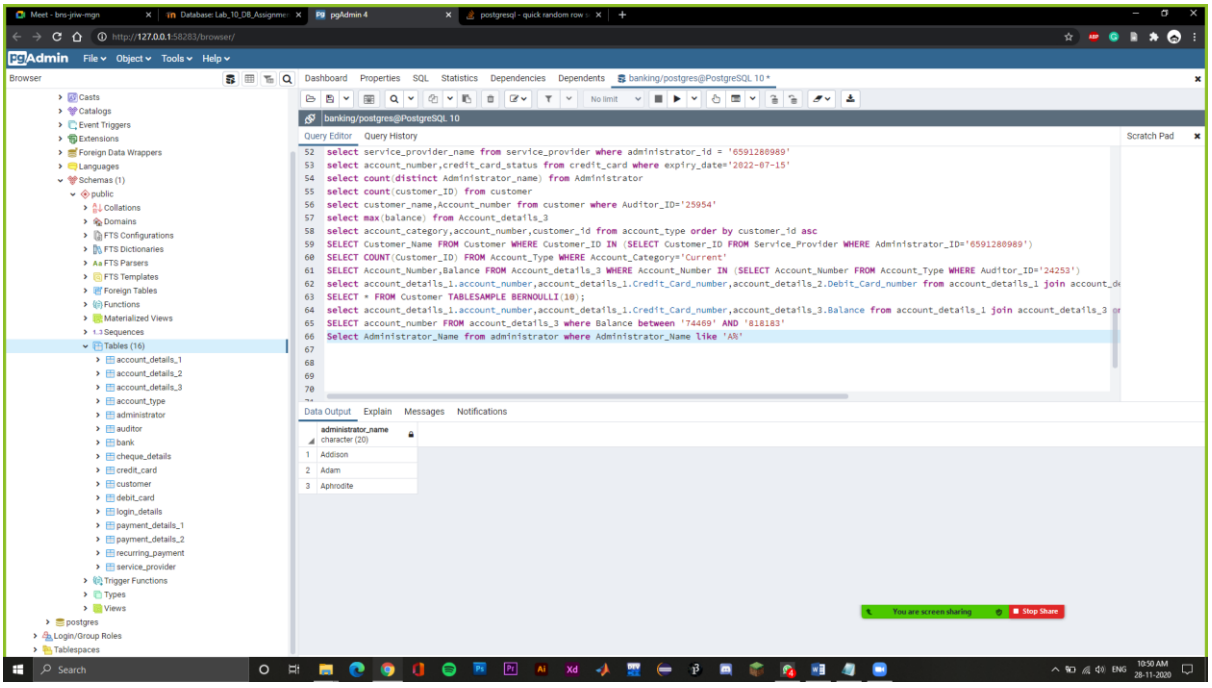
SELECT account_number FROM account_details_3 where Balance between '74469' AND '818183'

-> Showing the account number which has a balance between 74469 and 818183.



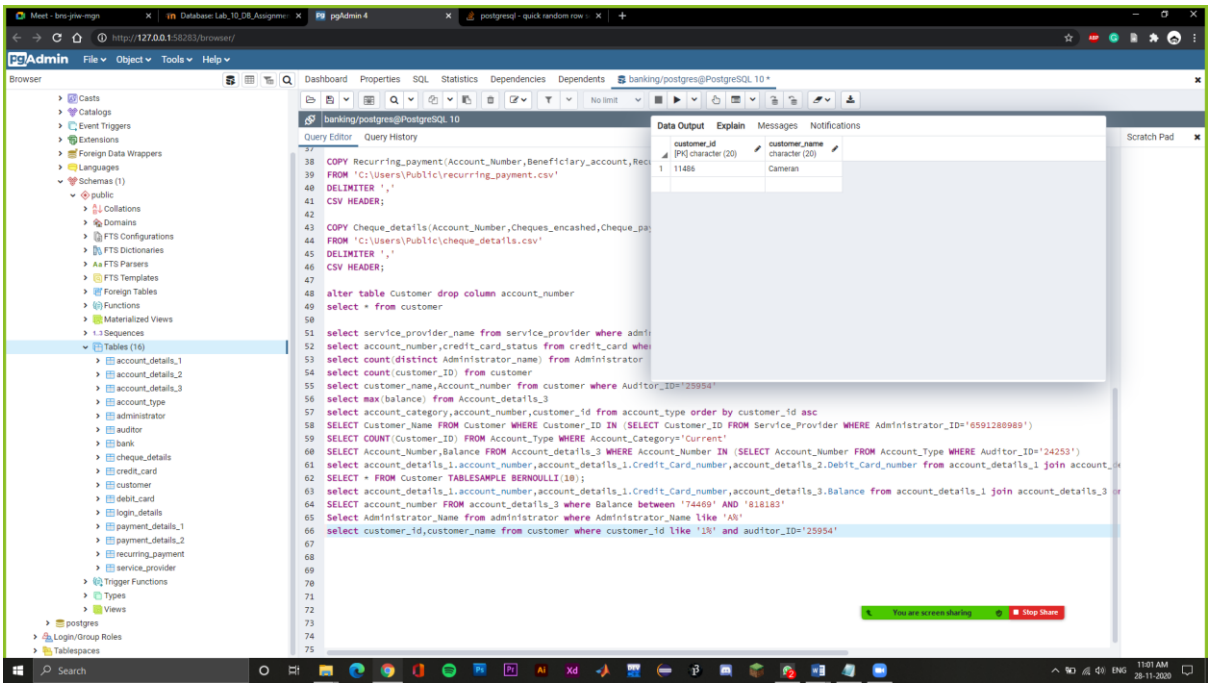
Select Administrator_Name from administrator where Administrator_Name like 'A%'

-> Showing the administrator name starting from A.



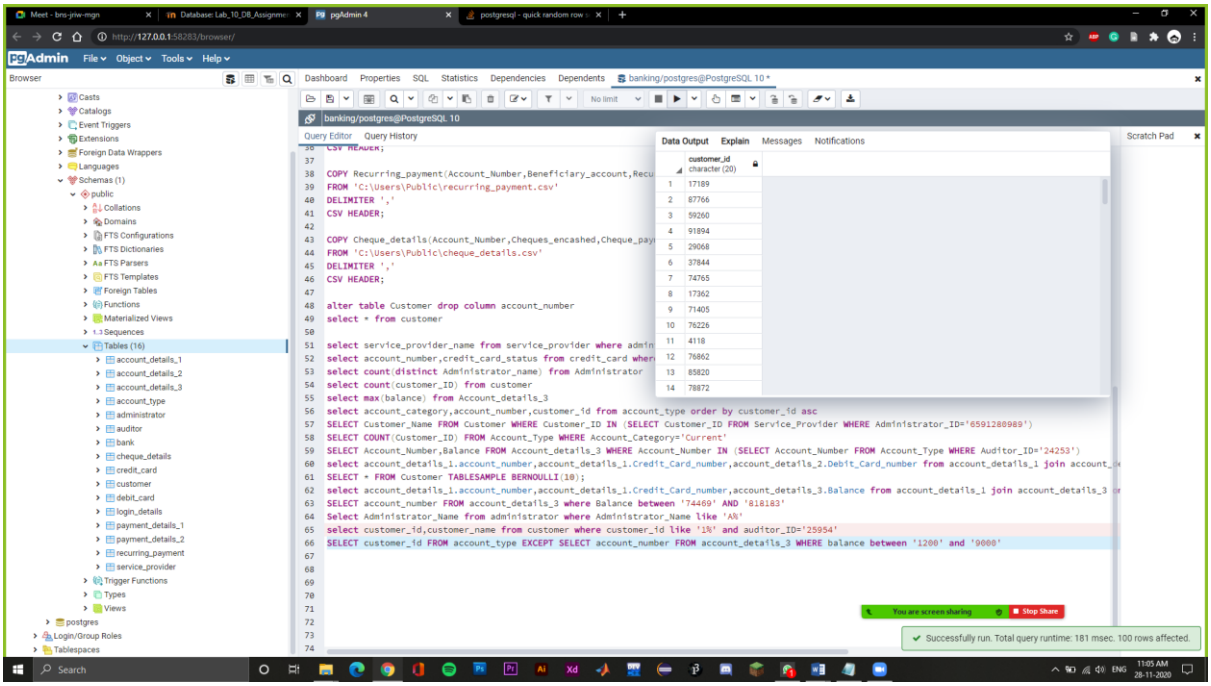
select customer_id,customer_name from customer where customer_id like '1%' and auditor_ID='25954'

-> Showing customer id and customer name which has auditor id 25954 and customer id starts from 1.



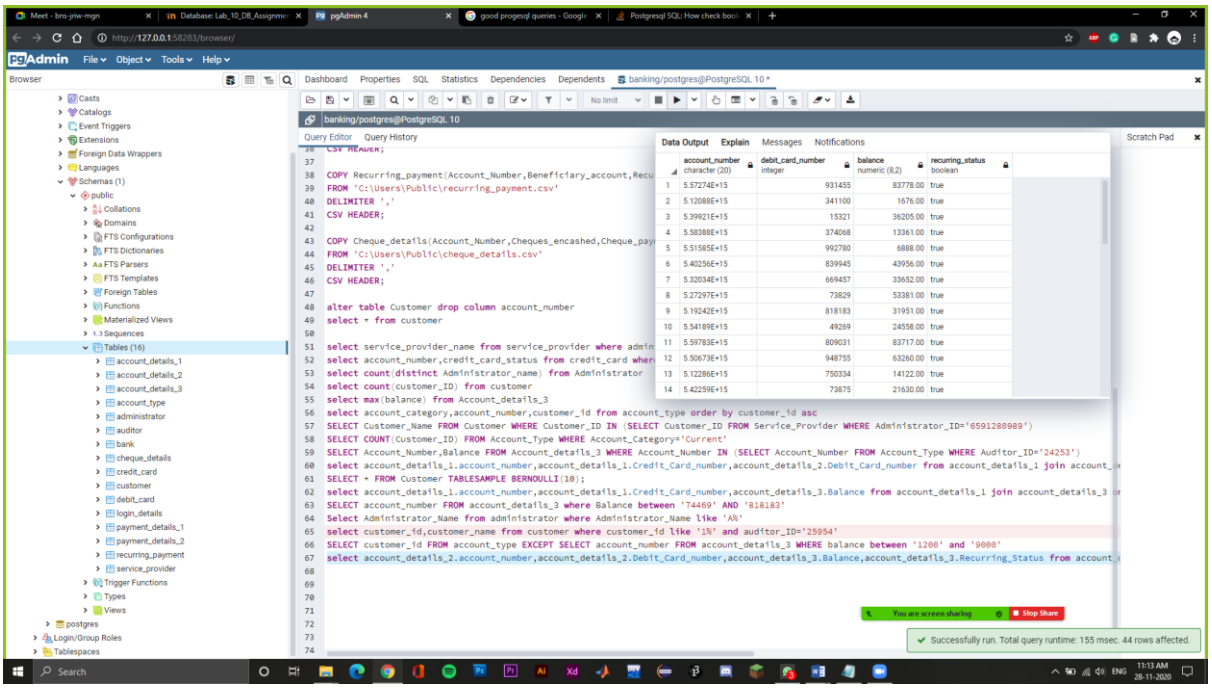
SELECT customer_id FROM account_type EXCEPT SELECT account_number FROM account_details_3 WHERE balance between '1200' and '9000'

-> Show customer id except for customer having balance between 1200 and 9000



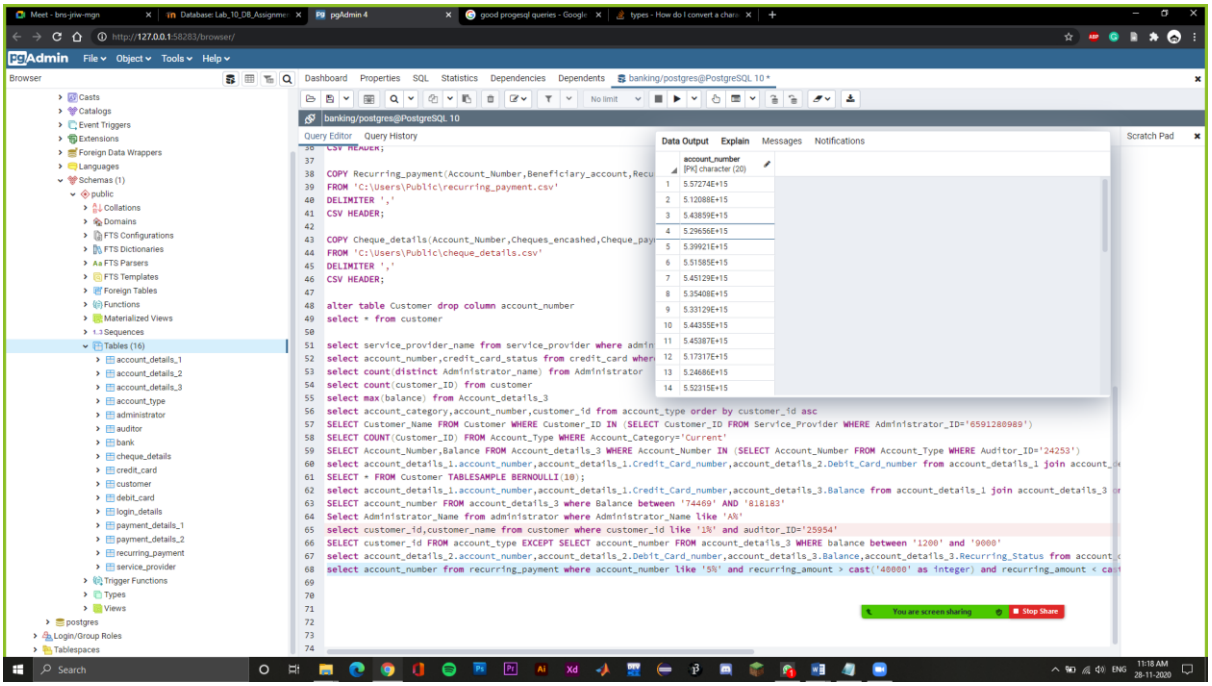
select account_details_2.account_number, account_details_2.Debit_Card_number, account_details_3.Balance,account_details_3.Recurring_Status from account_details_2 join account_details_3 on account_details_2.account_number=account_details_3.account_number and Recurring_Status = True

-> Show account number, debit card number, Balance and Recurring status of all the customers who have recurring customer as true



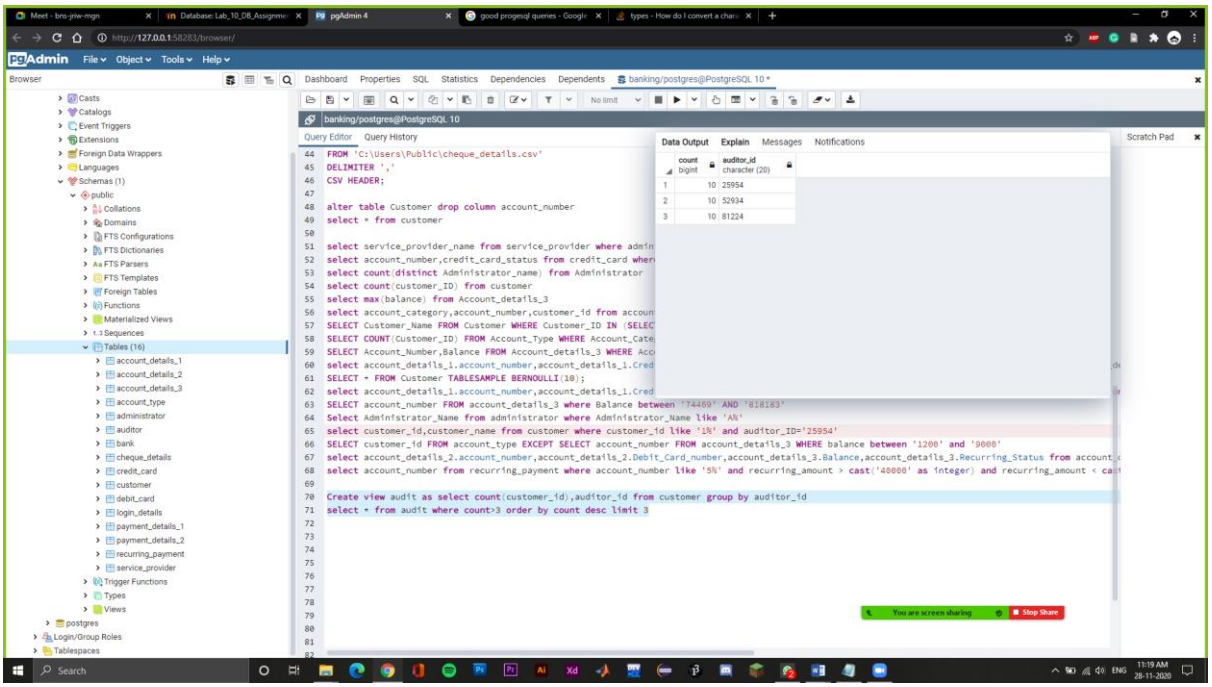
select account_number from recurring_payment where account_number like '5%' and recurring_amount > cast('40000' as integer) and recurring_amount < cast('80000' as integer)

-> Showing account number which has recurring payment between 40000 and 80000.



Create view audit as select count(customer_id), auditor_id from customer group by auditor_id
select * from audit where count > 3 order by count desc limit 3

->Show top 3 auditors who are auditing highest number of customer accounts



select account_number, balance from account_details_3 where balance=(select max(balance) from account_details_3 where balance < (select max(balance) from account_details_3))

->Show second highest balance among all the accounts present