

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 DECEMBER 2020

INDEX

1. Final version of SRS.
2. Final Noun Analysis.
3. Final ER-Diagrams all versions.
4. Conversion of Final ER-Diagram to Relational Model.
5. Normalization and Schema Refinement.
6. Final DDL Scripts, Insert statements, 40 SQL Queries, Snapshots of output of each query.

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=PLQVJk9oC5JKohoyVILfdxOOzyl6wyOur>

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

Age Group *

- 18-30
- 31-50
- 50+

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 complain along the previous lines, it should not take too long for that complaint to get resolved.

Question 5

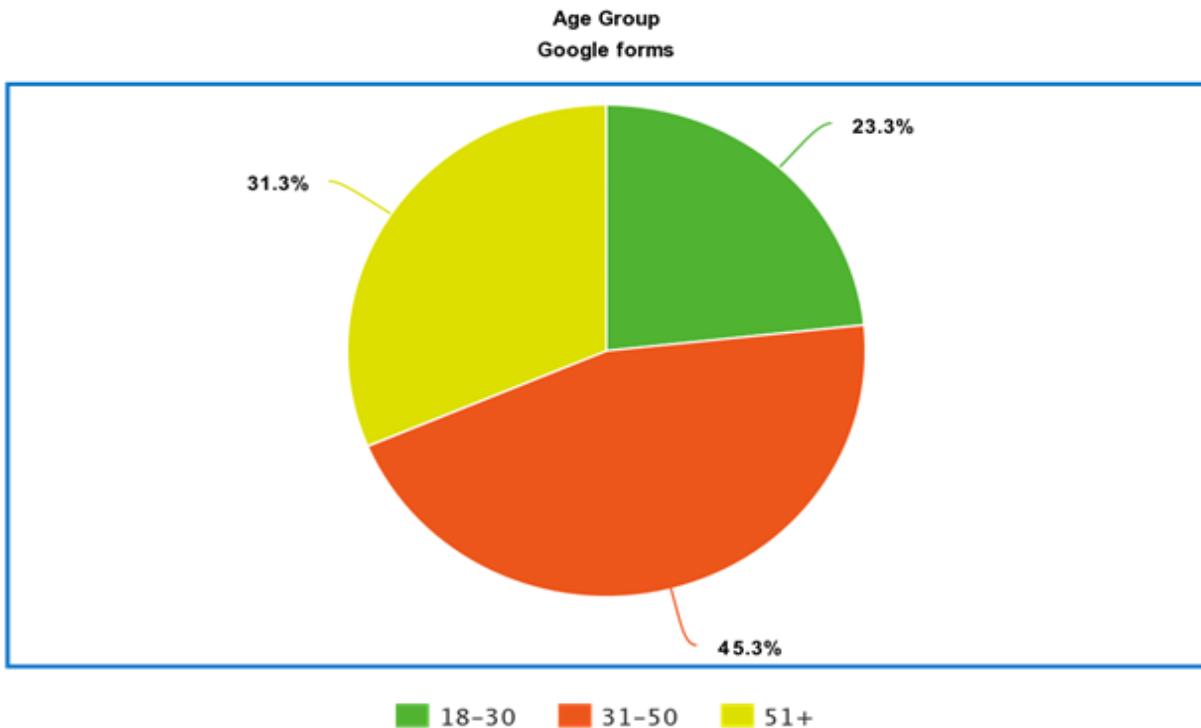
What improvements would you suggest for the existing customer care services?

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.

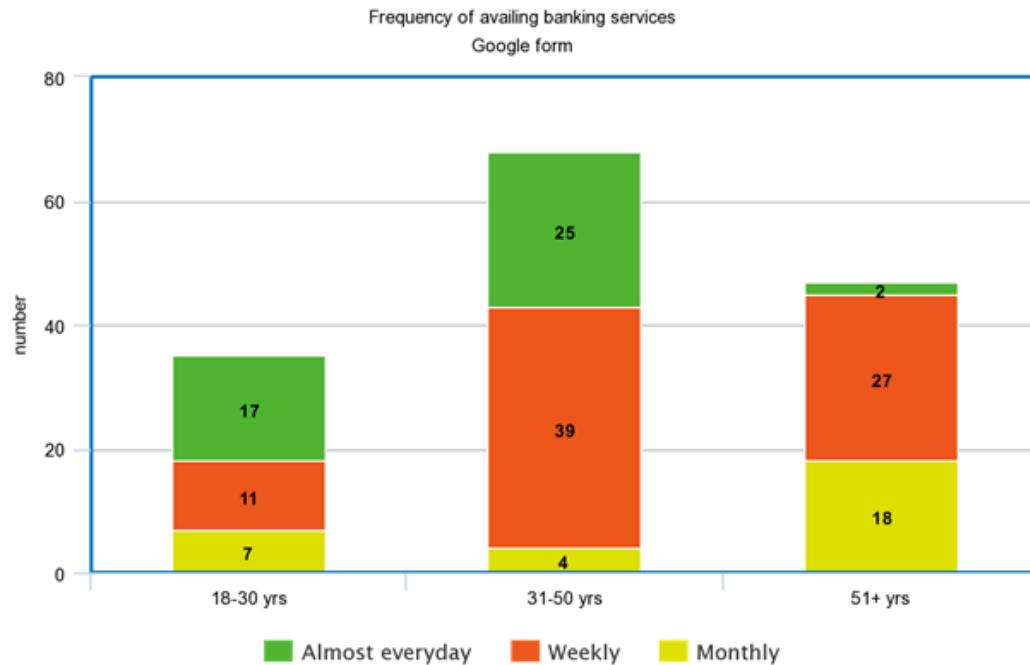
OBSERVATIONS

- 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 the required functionalities.



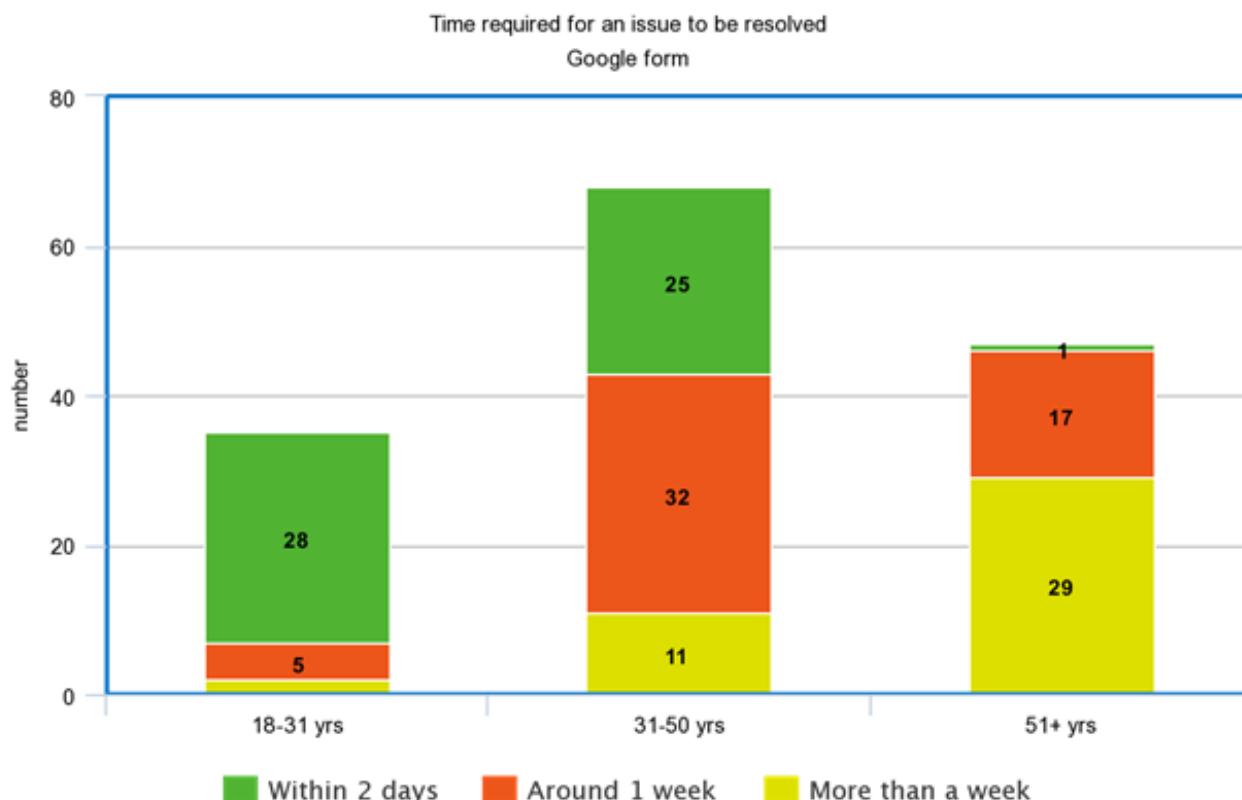
- The second graph represents the response to Q2 present in the form. The following observations have been made:

- 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
- 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.

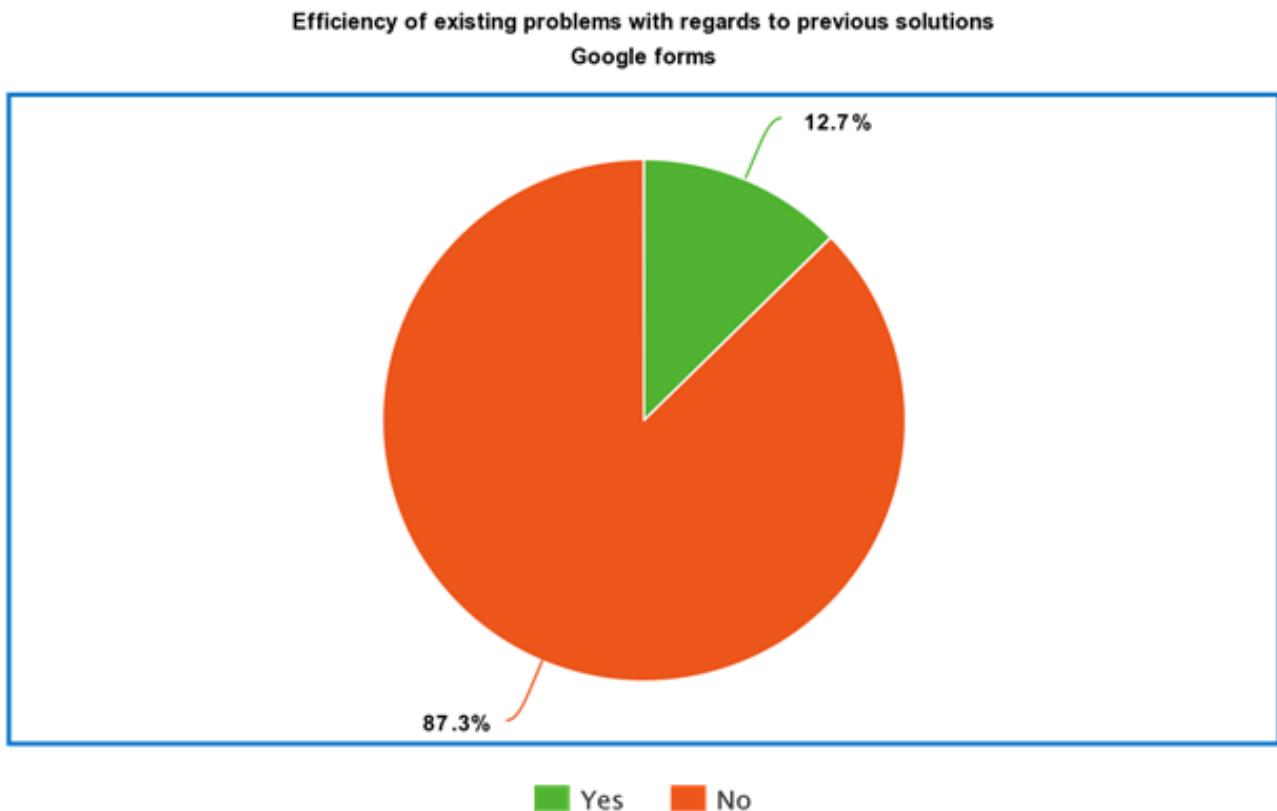


- For elderly people, most of them have contact the customer care in a weekly or monthly. Since they are not involved in frequent transactions, hence their interaction on a daily basis with customer care services is quite diminished.

- 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 perform 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
The customer	Duplicates	Scalability	Associations	It	Duplicate
complete edit	Associations	They	General	a few days	General
the Administrator	Duplicate	its customers	Duplicates	accounts	Attributes
funds	General	RAM	Irrelevant	queries	Duplicate
explicit authority	Vague	A secure way	Associations	We	General
				this case	Vague

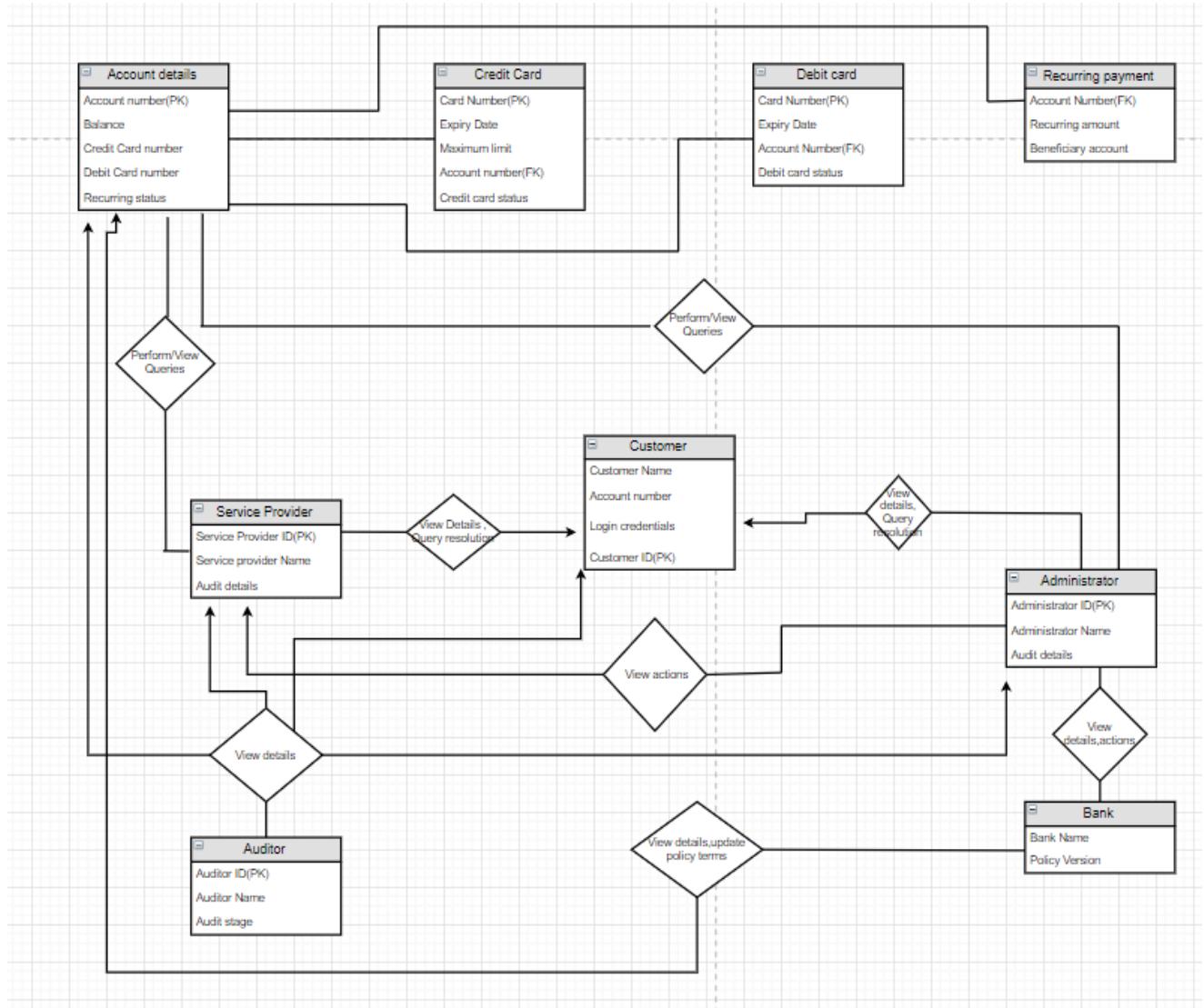
His role	Irrelevant	their items	Vague	the changes	General	recurring payments	Duplicates
admin privileges	Attributes	the same bank	Duplicates	the service providers/admin	Duplicate	their existing accounts	Attributes
the easiest way	Vague	any account	General	none	General	the bank balance	Attributes
their 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

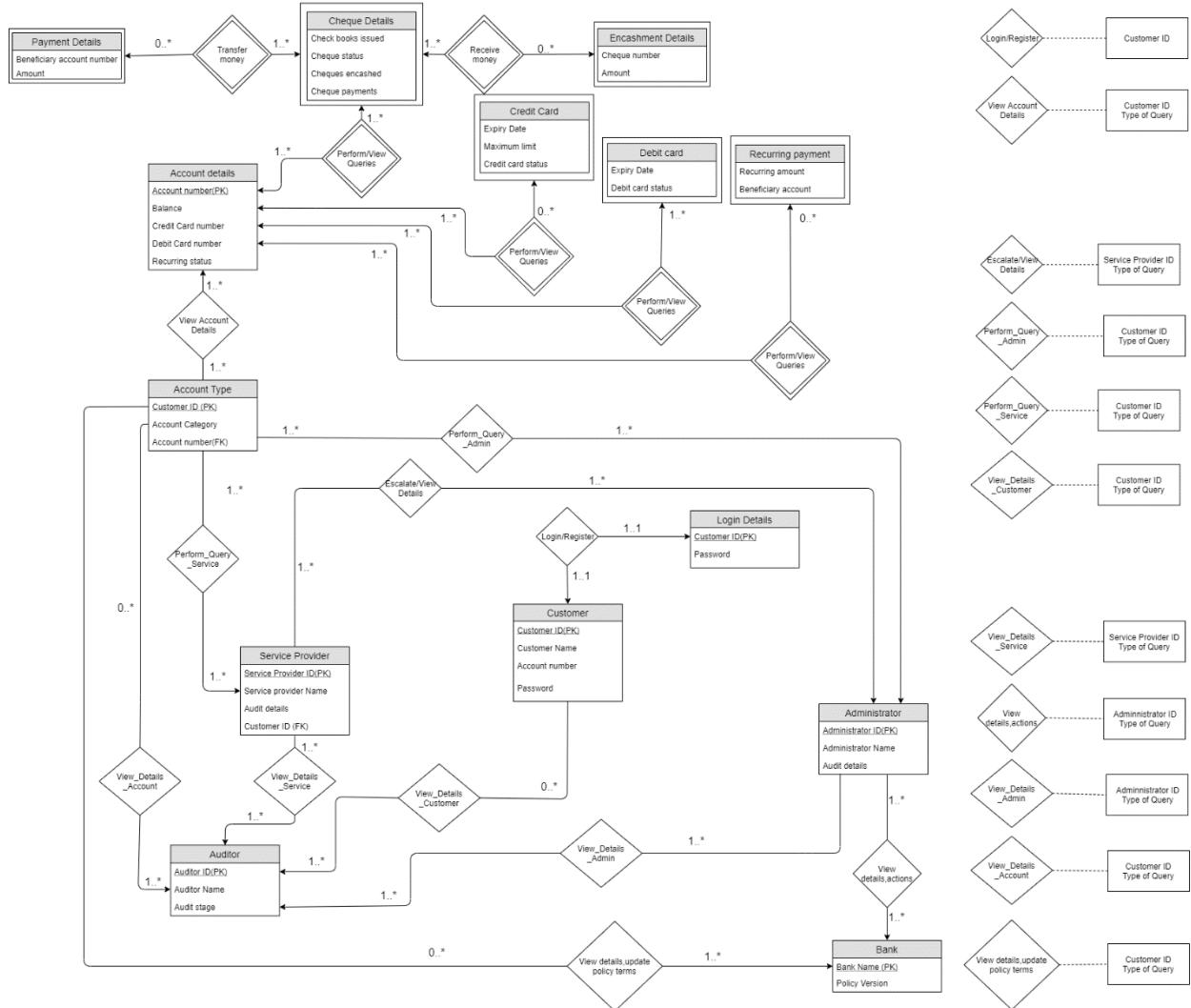
FINAL ER DIAGRAM ALL VERSIONS

VERSION 1

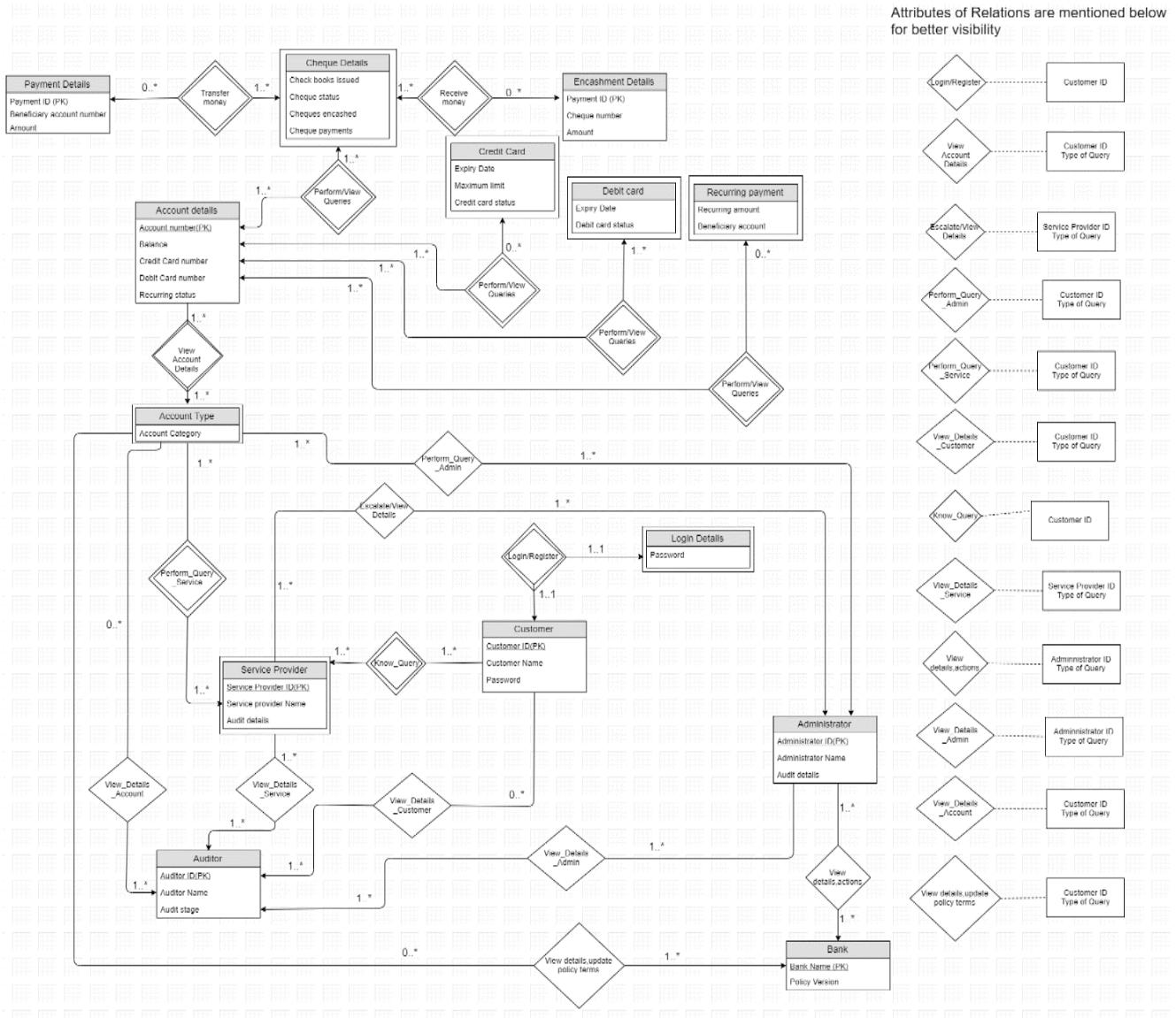


VERSION 2

Attributes of Relations are mentioned below for better visibility

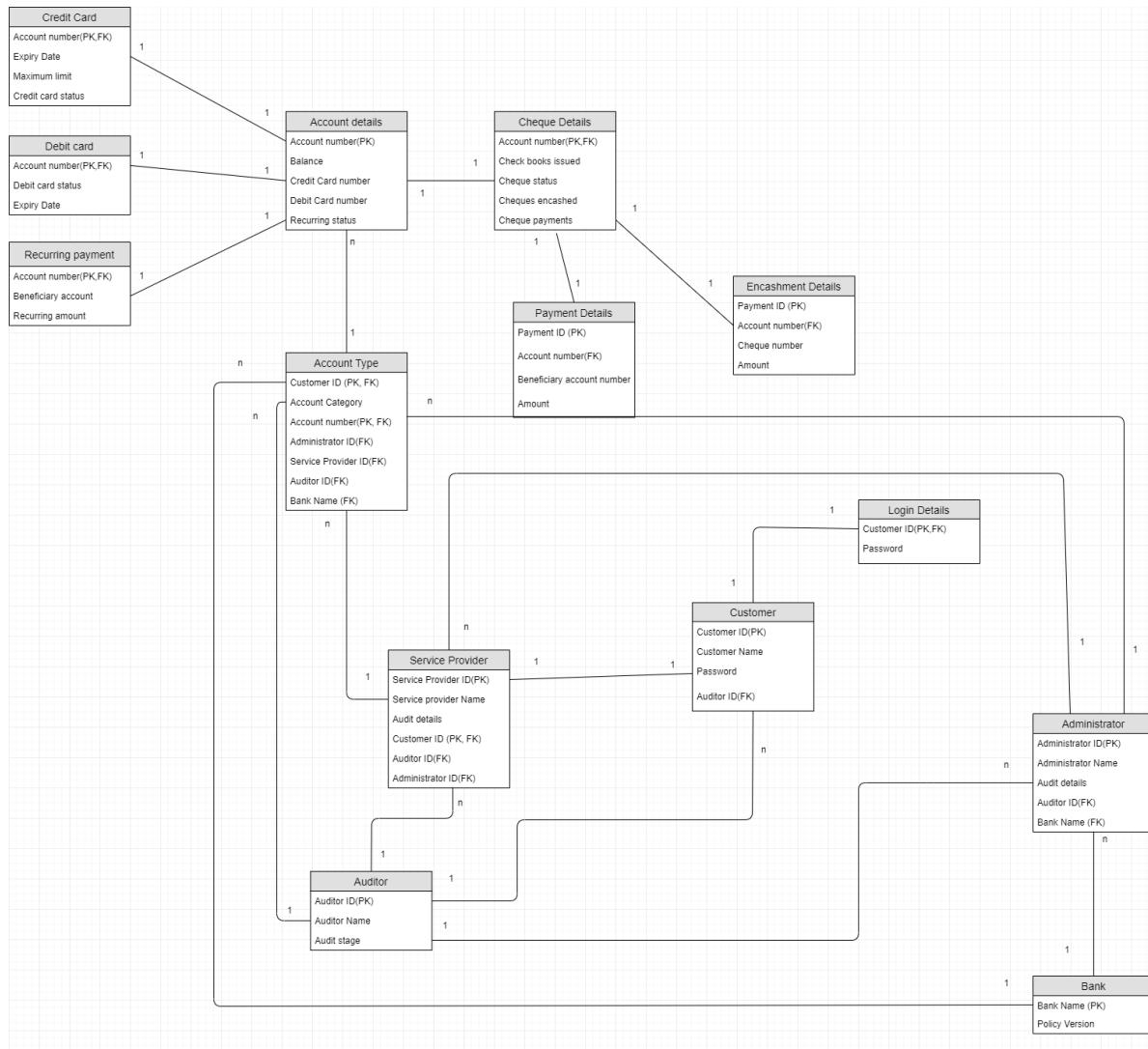


VERSION 3



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

Since we have multiple candidate keys with no partial and transitive dependencies, we decompose the relation into further sub-relations to obtain the highest normal form i.e. 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

Since we have multiple candidate keys with no partial and transitive dependencies, we decompose the relation into further sub-relations to obtain the highest normal form i.e. 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

$(\text{Service Provider ID}, \text{Customer ID}) \rightarrow \text{Service Provider Name, Audit Details, Administrator ID, Auditor ID}$

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

$\text{Administrator ID} \rightarrow \text{Administrator Name, Bank Name, Auditor ID, Audit Details}$

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

$(\text{Account Number}, \text{Customer ID}) \rightarrow \text{Account Category, Administrator ID, Service Provider ID, Bank Name, Auditor ID}$

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

$\text{Customer ID} \rightarrow \text{Customer Name, Password, Auditor ID}$

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;

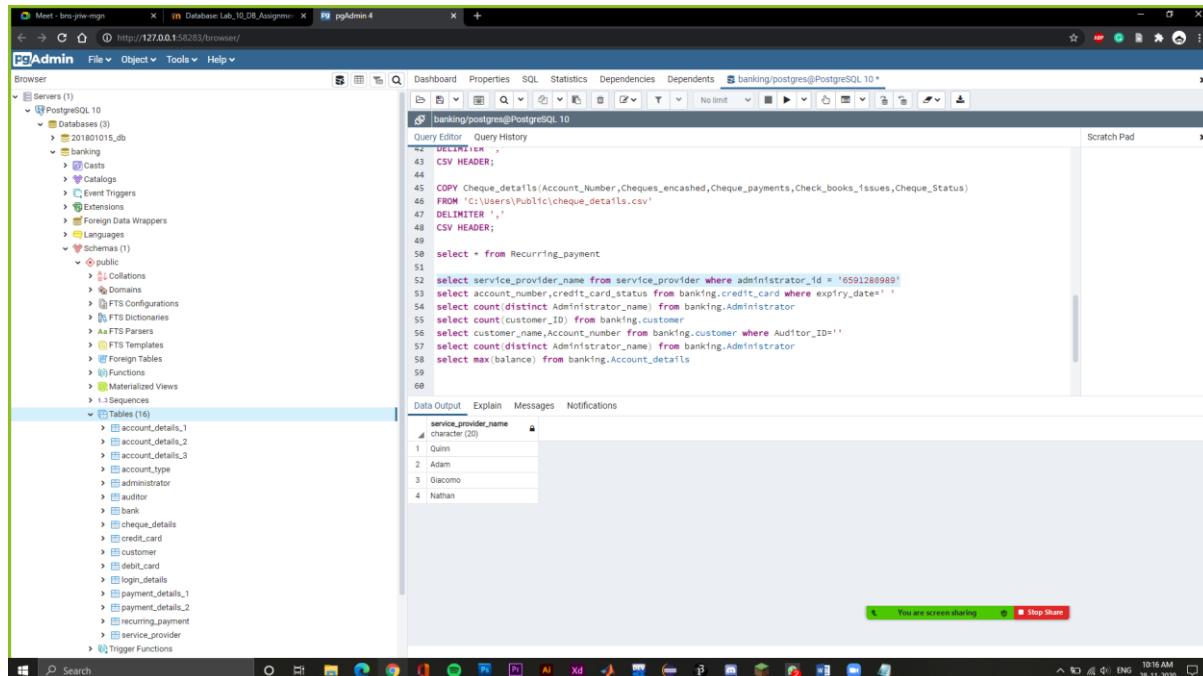
Payment_details
COPY payment_details(Account_Number,Payment_ID,Beneficiary_Account_Number,Amount)
FROM 'C:\Users\Public\payment_details.csv'
DELIMITER ','
CSV HEADER;

```

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.

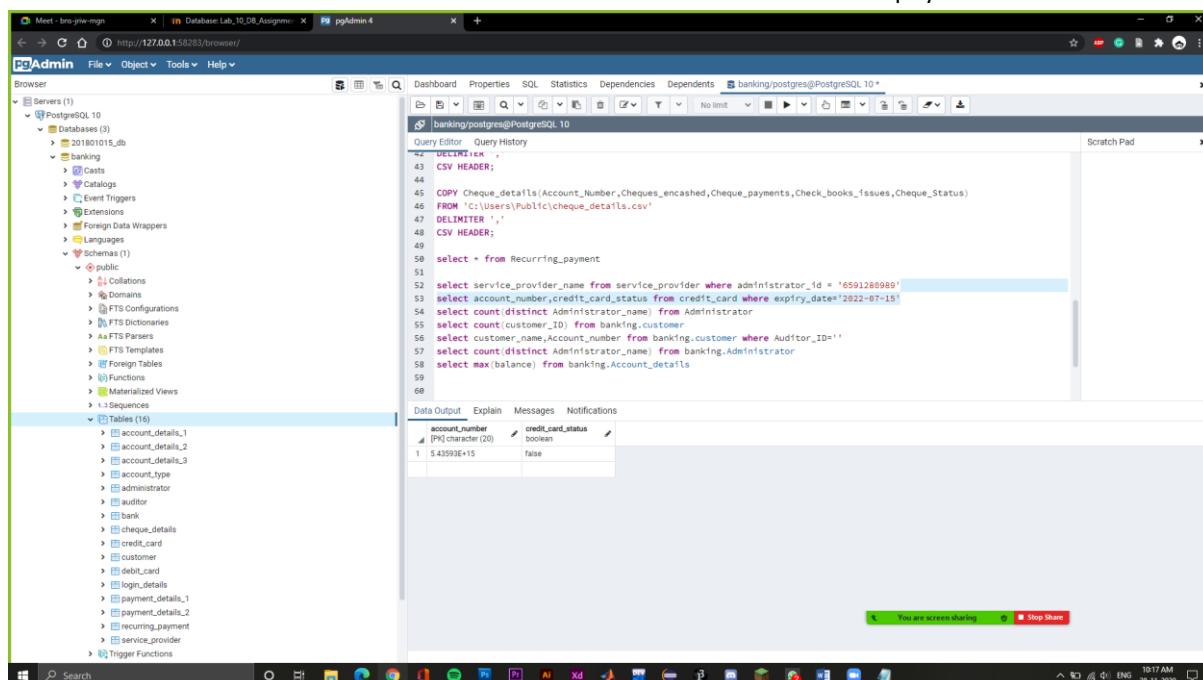


```
DELIMITER ;
CSV HEADER;
COPY Cheque_details(Account_Number,Cheques_encashed,Cheque_payments,Check_books_issues,Cheque_Status)
FROM 'C:\Users\Public\cheque_details.csv'
DELIMITER ',';
CSV HEADER;
select * from Recurring_payment
select service_provider_name from service_provider where administrator_id = '6591280989'
select account_number,credit_card_status from banking.credit_card where expiry_date=' '
select count(distinct Administrator_name) from banking.Administrator
select count(customer_ID) from banking.customer
select customer_name,Account_number from banking.customer where Auditor_ID=''
select count(distinct Administrator_name) from banking.Administrator
select max(balance) from banking.Account_details
```

service_provider_name
Quinn
Adam
Giacomo
Nathan

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.



```
DELIMITER ;
CSV HEADER;
COPY Cheque_details(Account_Number,Cheques_encashed,Cheque_payments,Check_books_issues,Cheque_Status)
FROM 'C:\Users\Public\cheque_details.csv'
DELIMITER ',';
CSV HEADER;
select * from Recurring_payment
select service_provider_name from service_provider where administrator_id = '6591280989'
select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
select count(distinct Administrator_name) from Administrator
select count(customer_ID) from banking.customer
select customer_name,Account_number from banking.customer where Auditor_ID=''
select count(distinct Administrator_name) from banking.Administrator
select max(balance) from banking.Account_details
```

account_number	credit_card_status
54359E+15	false

```
select count(distinct Administrator_name) from Administrator
```

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

The screenshot shows the pgAdmin 4 interface with a database connection named 'banking/postgres@PostgreSQL 10'. In the left sidebar, under 'Tables (16)', there is a table named 'Administrator'. In the main query editor window, the following SQL code is run:

```
DELIMITER ;
CSV HEADER;
COPY Cheque_details(Account_Number,Cheques_EnCashed,Cheque_Payments,Check_Books_Issues,Cheque_Status)
FROM 'C:/Users/Public/Cheque_Details.csv'
DELIMITER ;
CSV HEADER;
select * from Recurring_payment
select service_provider_name from service_provider where administrator_id = '6591288989'
select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
select count(distinct Administrator_name) from Administrator
select count(customer_ID) from banking.customer
select customer_name,Account_number from banking.customer where Auditor_ID='1'
select count(distinct Administrator_name) from banking.Administrator
select max(balance) from banking.Account_Details
;
```

The results of the query are displayed in the 'Data Output' tab, showing a single row with a count of 25.

```
select count(customer_ID) from customer
```

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

The screenshot shows the pgAdmin 4 interface with a database connection named 'banking/postgres@PostgreSQL 10'. In the left sidebar, under 'Tables (16)', there is a table named 'Customer'. In the main query editor window, the same SQL code as the previous screenshot is run:

```
DELIMITER ;
CSV HEADER;
COPY Cheque_Details(Account_Number,Cheques_EnCashed,Cheque_Payments,Check_Books_Issues,Cheque_Status)
FROM 'C:/Users/Public/Cheque_Details.csv'
DELIMITER ;
CSV HEADER;
select * from Recurring_payment
select service_provider_name from service_provider where administrator_id = '6591288989'
select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
select count(distinct Administrator_name) from Administrator
select count(customer_ID) from customer
select customer_name,Account_number from banking.customer where Auditor_ID='1'
select count(distinct Administrator_name) from banking.Administrator
select max(balance) from banking.Account_Details
;
```

The results of the query are displayed in the 'Data Output' tab, showing a single row with a count of 100.

select max(balance) from Account_details_3

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

The screenshot shows the pgAdmin 4 interface with a PostgreSQL 10 database selected. In the left sidebar, under 'Tables (16)', several tables are listed, including account_details_1, account_details_2, account_details_3, account_type, administrator, auditor, bank, cheque_details, credit_card, customer, debit_card, login_details, payment_details_1, payment_details_2, recurring_payment, service_provider, and trigger_functions. The main window displays a query editor with the following SQL code:

```
42 CSV HEADER;
43
44
45 COPY Cheque_details(Account_Number,Cheques_encahsed,Cheque_payments,Check_books_Issues,Cheque_Status)
46 FROM 'C:/Users/Public/Cheque_Details.csv'
47 DELIMITER ','
48 CSV HEADER;
49
50 select * from Recurring_payment
51
52 select service_provider_name from service_provider where administrator_id = '6591288989'
53 select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
54 select count(distinct administrator_name) from Administrator
55 select count(customer_ID) from customer
56 select customer_name,Account_number from customer where Auditor_ID='25954'
57 select max(balance) from Account_details_3
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
718
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
878
879
880
881
882
883
884
885
886
887
888
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
918
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
978
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1095
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1295
1296
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1395
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1495
1496
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1595
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1695
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1895
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2095
2096
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2195
2196
2197
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
223
```

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

```

53 select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
54 select count(distinct Administrator_name) from Administrator
55 select count(customer_ID) from customer
56 select customer_name,Account_number from customer where Auditor_ID='25954'
57 select max(balance) from Account_details_3
58 select account_category,account_number,customer_id from account_type order by customer_id asc
59 SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591280989')
60 SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Category='Current'
61 SELECT Account_Number,Balance FROM Account_details_3 WHERE Account_Number = (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
62
63
64
65
66
67
68
69
70
71
    
```

customer_name
Tameehan
Shelly
Rama
Vincent

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

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

```

53 select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
54 select count(distinct Administrator_name) from Administrator
55 select count(customer_ID) from customer
56 select customer_name,Account_number from customer where Auditor_ID='25954'
57 select max(balance) from Account_details_3
58 select account_category,account_number,customer_id from account_type order by customer_id asc
59 SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591280989')
60 SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Category='Current'
61 SELECT Account_Number,Balance FROM Account_details_3 WHERE Account_Number = (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
62
63
64
65
66
67
68
69
70
71
    
```

count
50

Successfully run. Total query runtime: 117 msec. 1 rows affected.

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_Number,Balance FROM Account_Details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')

```

account_number	balance
5.29515e+15	8871.00
5.44355e+15	44420.00
5.50674e+15	63260.00
5.45245e+15	35404.00
5.51983e+15	33502.00
5.53494e+15	21337.00
5.35645e+15	24579.00
5.13861e+15	96347.00
5.29832e+15	15968.00
5.13992e+15	37761.00

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 account_number,Credit_Card_number,Debit_Card_number FROM account_details_1 JOIN account_details_2 ON account_details_1.account_number=account_details_2.account_number

```

account_number	Credit_Card_number	Debit_Card_number
5.45953e+15	840045	519280
5.57274e+15	910396	931455
5.12088e+15	547558	341100
5.43859e+15	866231	634495
5.37984e+15	383283	22835
5.193909e+15	475026	731921
5.29511e+15	765554	41190
5.29656e+15	930375	649826
5.39921e+15	834921	15321
5.59094e+15	256319	642875
5.58388e+15	488428	374068
5.51585e+15	682428	992780
5.40256e+15	911941	839945

SELECT * FROM Customer TABLESAMPLE BERNOUILLI(10);

->10 random values from customer based on Bernoulli distribution

```

49
50      select * from Recurring_payment
51
52      select service_provider_name from service_provider where administrator_id = '6591288989'
53      select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
54      select count(distinct Administrator_name) from Administor
55      select count(customer_ID) from customer
56      select customer_name,Account_number,customer where Auditor_ID='25954'
57      select max(balance) from Account_details_3
58      select account_category,account_number,customer_id from account_type order by customer_id asc
59      SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591288989')
60      SELECT Account_Number,Balance FROM Account_details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
61      SELECT Account_number,Balance FROM Account_details_3 WHERE Account_number=account_number,account_details_1.Credit_Card_number,account_details_2.Debit_Card_number from account_details_1 join account_details_2 on account_number=account_number
62      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_number=account_number
63      SELECT * FROM Customer TABLESAMPLE BERNOUILLI(10);
64
65
66
67

```

customer_id	customer_name	account_number	pass_word	auditor_id
1 94914	Wyatt	5.19405E+15	KKV0P9PB3W	52934
2 65669	Riley	5.59950E+15	LVM8MXKX4ED	10976
3 88866	Adara	5.47807E+15	G0V1ZT2U3HS	28954
4 76226	Zahir	5.54771E+15	QFB17P9G3TD	41445
5 86002	Shane	5.40486E+15	PQCC2FLYJUM	98771
6 74830	Brooke	5.1208E+15	UY2SBWV4W0	41445
7 60024	Jada	5.57859E+15	AGH55X0P7AF	28954
8 6843	Darryl	5.48076E+15	SIB34UEJTT	51254
9 3612	Emmy	5.42734E+15	PTB62S2B4GB	25954
10 47267	Forrest	5.55668E+15	I27HU0010D	81224
11 40594	Willow	5.49754E+15	YXQ2RJC3PE	24253
12 17189	Mira	5.55359E+15	FQ24VTA0CS	98771
13 9264	Aristote	5.29791E+15	FP065R0B3MA	54208

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

```

52      select service_provider_name from service_provider where administrator_id = '6591288989'
53      select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
54      select count(distinct Administrator_name) from Administor
55      select count(customer_ID) from customer
56      select customer_name,Account_number,customer where Auditor_ID='25954'
57      select max(balance) from Account_details_3
58      select account_category,account_number,customer_id from account_type order by customer_id asc
59      SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591288989')
60      SELECT COUNT(Customer_ID) FROM Customer WHERE Account_Type='Current'
61      SELECT Account_Number,Balance FROM Account_details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
62      select account_details_1.account_number,account_number,account_details_1.Credit_Card_number,account_details_3.Balance from account_details_1 join account_details_3 on account_number=account_number
63      SELECT * FROM Customer TABLESAMPLE BERNOUILLI(10);
64
65
66
67

```

account_number	credit_card_number	balance
1 5.4593E+15	842045	74469.00
2 5.57274E+15	910396	83778.00
3 5.1208E+15	547558	1676.00
4 5.4839E+15	866231	46821.00
5 5.3794E+15	383283	82911.00
6 5.19309E+15	475026	33597.00
7 5.29511E+15	765554	8871.00
8 5.29656E+15	950307	66356.00
9 5.39921E+15	834921	36205.00
10 5.5906E+15	256319	10200.00
11 5.58388E+15	488428	13361.00
12 5.51595E+15	682428	6888.00
13 5.40256E+15	911941	43956.00

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.

The screenshot shows the pgAdmin 4 interface with a query editor containing the following SQL code:

```
S2 select service_provider_name from service_provider where administrator_id = '6591288989'
S3 select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
S4 select count(distinct Administrator_name) from Administrator
S5 select count(customer_id) from customer
S6 select max(balance) from Account_details_3
S7 select account_category,account_number,customer_id from account_type order by customer_id asc
S8 SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591288989')
S9 SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Category='Current'
S10 SELECT Account_Number,Balance FROM Account_Details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
S11 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
S12 select account_number FROM account_details_3 where Balance between '74469' AND '818183'
S13 Select Administrator_Name from administrator where Administrator_Name like 'A%'
```

The Data Output tab shows the results of the query:

account_number
1. 5.4359E+15
2. 5.57274E+15
3. 5.37984E+15
4. 5.33196E+15
5. 5.26682E+15
6. 5.59782E+15
7. 5.41356E+15
8. 5.24415E+15
9. 5.20861E+15
10. 5.34633E+15
11. 5.48996E+15
12. 5.38899E+15
13. 5.10223E+15

Select Administrator_Name from administrator where Administrator_Name like 'A%'

-> Showing the administrator name starting from A.

The screenshot shows the pgAdmin 4 interface with a query editor containing the following SQL code:

```
S2 select service_provider_name from service_provider where administrator_id = '6591288989'
S3 select account_number,credit_card_status from credit_card where expiry_date='2022-07-15'
S4 select count(distinct Administrator_name) from Administrator
S5 select count(customer_id) from customer
S6 select max(balance) from Account_details_3
S7 select account_category,account_number,customer_id from account_type order by customer_id asc
S8 SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591288989')
S9 SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Category='Current'
S10 SELECT Account_Number,Balance FROM Account_Details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
S11 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
S12 select account_number FROM account_details_3 where Balance between '74469' AND '818183'
S13 Select Administrator_Name from administrator where Administrator_Name like 'A%'
```

The Data Output tab shows the results of the query:

administrator_name
1. Addison
2. Adam
3. Aphrodite

```
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.

The screenshot shows the pgAdmin 4 interface with a database connection named 'banking/postgres@PostgreSQL 10'. In the left sidebar, under 'Tables (16)', the 'account_type' table is selected. The main area displays a SQL query:

```
38 COPY Recurring_payment(Account_Number,Beneficiary_account,Recu
39 FROM 'C:\Users\Public\recurring_payment.csv'
40 DELIMITER ',';
41 CSV HEADER;
42
43 COPY Cheque_details(Account_Number,Cheques_enveloped,Cheque_pa
44 FROM 'C:\Users\Public\cheque_details.csv'
45 DELIMITER ',';
46 CSV HEADER;
47
48 alter table Customer drop column account_number
49 select * from customer
50
51 select service_provider_name from service_provider where adm
52 select account_number,credit_card_status from credit_card wh
53 select count(distinct Administrator_Name) from Administrator
54 select count(customer_ID) from customer
55 select customer_name,Account_Number from customer where Auditor_ID='25954'
56 select max(balance) from Account_Details_3
57 select account_category,account_number,customer_id from account_type order by customer_id asc
58 SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591288989')
59 SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Cat
60 SELECT Account_Number,Balance FROM Account_Details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
61 select account_details_1.account_number,account_details_1.Credit_Card_number,account_details_2.B
62 SELECT + FROM Customer TABLESAMPLE BERNOULLI(10);
63 select account_details_1.account_number,account_details_1.Credit_Card_number,account_details_3.Balanc
64 SELECT account_number FROM account_details_3 WHERE Balance between '74469' AND '818183'
65 Select Administrator_Name from administrator where Administrator_Name like 'AN'
66 select customer_id,customer_name from customer where customer_id like '1%' and auditor_ID='25954'
```

The 'Data Output' tab shows the results of the query:

customer_id	customer_name
11486	Cameran

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

The screenshot shows the pgAdmin 4 interface with a database connection named 'banking/postgres@PostgreSQL 10'. In the left sidebar, under 'Tables (16)', the 'account_type' table is selected. The main area displays a SQL query:

```
38 COPY Recurring_payment(Account_Number,Beneficiary_account,Recu
39 FROM 'C:\Users\Public\recurring_payment.csv'
40 DELIMITER ',';
41 CSV HEADER;
42
43 COPY Cheque_details(Account_Number,Cheques_enveloped,Cheque_pa
44 FROM 'C:\Users\Public\cheque_details.csv'
45 DELIMITER ',';
46 CSV HEADER;
47
48 alter table Customer drop column account_number
49 select * from customer
50
51 select service_provider_name from service_provider where adm
52 select account_number,credit_card_status from credit_card wh
53 select count(distinct Administrator_Name) from Administrator
54 select count(customer_ID) from customer
55 select customer_name,Account_Number from customer where Auditor_ID='25954'
56 select max(balance) from Account_Details_3
57 select account_category,account_number,customer_id from account_type order by customer_id asc
58 SELECT Customer_Name FROM Customer WHERE Customer_ID IN (SELECT Customer_ID FROM Service_Provider WHERE Administrator_ID='6591288989')
59 SELECT COUNT(Customer_ID) FROM Account_Type WHERE Account_Cat
60 SELECT Account_Number,Balance FROM Account_Details_3 WHERE Account_Number IN (SELECT Account_Number FROM Account_Type WHERE Auditor_ID='24253')
61 select account_details_1.account_number,account_details_1.Credit_Card_number,account_details_2.B
62 SELECT + FROM Customer TABLESAMPLE BERNOULLI(10);
63 select account_details_1.account_number,account_details_1.Credit_Card_number,account_details_3.Balanc
64 SELECT account_number FROM account_details_3 WHERE Balance between '74469' AND '818183'
65 Select Administrator_Name from administrator where Administrator_Name like 'AN'
66 SELECT customer_id FROM account_type EXCEPT SELECT account_number FROM account_details_3 WHERE balance between '1200' and '9000'
```

The 'Data Output' tab shows the results of the query:

customer_id
17189
67766
59260
91894
29068
37844
74765
817362
71405
76226
4118
76862
69820
78872

```
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

account_number	debit_card_number	balance	recurring_status
1 5.57274E+15	931455	83778.00	true
2 5.12088E+15	341100	1676.00	true
3 5.39921E+15	15321	36205.00	true
4 5.58088E+15	374068	13361.00	true
5 5.51585E+15	992780	6888.00	true
6 5.40256E+15	859945	43956.00	true
7 5.32034E+15	669437	33652.00	true
8 5.27297E+15	73829	53381.00	true
9 5.19242E+15	818183	31951.00	true
10 5.54189E+15	49269	24558.00	true
11 5.59783E+15	809691	82717.00	true
12 5.50078E+15	948755	63260.00	true
13 5.12286E+15	750334	14122.00	true
14 5.42259E+15	73875	21630.00	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.

account_number
1 5.57274E+15
2 5.12088E+15
3 5.43859E+15
4 5.29656E+15
5 5.39921E+15
6 5.51585E+15
7 5.45129E+15
8 5.35408E+15
9 5.33129E+15
10 5.44355E+15
11 5.45897E+15
12 5.17317E+15
13 5.24686E+15
14 5.52315E+15

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

The screenshot shows the pgAdmin 4 interface with two tabs open: 'Meet - bns-prw-mgn' and 'Database Lab_10_DB_Assignment'. The 'Query Editor' tab contains the following SQL code:

```
CREATE VIEW audit AS
SELECT auditor_id, COUNT(*) AS count
FROM (
    SELECT customer_id, auditor_id
    FROM customer
    GROUP BY customer_id, auditor_id
) AS subquery
ORDER BY count DESC
LIMIT 3;
```

The 'Data Output' tab shows the results of the query:

count	auditor_id
1	10 25954
2	10 52934
3	10 81224

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

The screenshot shows the pgAdmin 4 interface with two tabs open: 'Meet - bns-prw-mgn' and 'Database Lab_10_DB_Assignment'. The 'Query Editor' tab contains the following SQL code:

```
CREATE VIEW second_highest_balance AS
SELECT account_number, balance
FROM (
    SELECT account_number, balance
    FROM account_details_3
    ORDER BY balance DESC
) AS subquery
LIMIT 1;
```

The 'Data Output' tab shows the results of the query:

account_number	balance
52215E+15	90620.00