**DATABASE MANAGEMENT SYSTEMS**

**Name: Abhirami.b**

**Roll.no : cb.sc.i5das21003**

**Github URL of the project page:**

**1.Section-1**

**Application Name : BANK MANAGEMENT**

**Description:** Bank management refers to the strategic planning, organization, and oversight of all activities within a bank to ensure its smooth and effective operation. It involves a comprehensive set of functions and responsibilities aimed at achieving the bank's objectives, maintaining financial stability, and ensuring regulatory compliance. The key components of bank management encompass various aspects of planning, decision-making, risk management, and customer service.

**Statergic planning**

Long-term vision: Developing a clear vision and mission for the bank.

Goal setting: Establishing measurable objectives and performance targets.

Market analysis: Assessing economic trends and competition to inform strategic decisions.

**Risk Management:**

Credit risk: Evaluating the creditworthiness of borrowers to minimize loan default.

Market risk: Managing exposure to financial market fluctuations.

Operational risk: Addressing potential disruptions in day-to-day operations.

Regulatory compliance: Ensuring adherence to legal and regulatory requirements.

**Asset and Liability Management (ALM):**

Balancing assets and liabilities to maintain liquidity and profitability.

Monitoring interest rate risk to optimize the bank's net interest margin.

Investment decisions: Strategically allocating resources to maximize returns.

**Customer Relationship Management (CRM):**

Customer service: Providing a positive and efficient experience for clients.

Product development: Introducing new financial products and services.

Customer retention: Building and maintaining long-term relationships with clients.

**Regulatory Compliance:**

Adhering to local and international banking regulations.

Implementing anti-money laundering (AML) and know your customer (KYC) measures.

Ensuring data security and privacy in accordance with regulations.

**Human Resource Management:**

**Talent acquisition**: Recruiting skilled professionals for various roles.

**Training and development**: Continuously enhancing employee skills.

**Employee engagement**: Fostering a positive work environment and culture.

Bank management is a dynamic field that is continually evolving in response to changes in the financial landscape, technology, and regulatory requirements. Effective bank management is crucial for the institution's success, financial stability, and the trust of its customers and stakeholders

**[About the application]**

**Report:**

**List the report names with its purpose**

|  |  |
| --- | --- |
| **Report Name** | **Purpose** |
| Operational efficency | **Streamlined Processes:** Streamlines operational processes for increased efficiency.  **Error Reduction:** Reduces the likelihood of errors and enhances overall operational performance. |
| Financial services | Builds and maintains customer trust through reliable and secure financial services |
| **C**omplaint resolution work flow | Streamlines the process of addressing and resolving customer complaints |
| Perform metric dashboard | Offers a real-time dashboard for monitoring key performance indicators related to strategic goals. |
| **P**roduct recommandation | Recommends tailored financial products based on customer profiles and preferences |

**Technologies**

**SQL Based Application:**

|  |  |
| --- | --- |
| **Front End** | HTML,CSS |
| **Back End** | LIVE SQL |
| **Editor** | VISUAL STUDIO CODE |
| **Language** | JAVA SCRIPT |
| **Framework** | NODE.js |

**Why? What?**

**[[Write why this application is required]]**

A bank management application is indispensable for the efficient and secure functioning of financial institutions. With its ability to streamline operations, automate processes, and provide real-time data-driven insights, the application significantly enhances overall efficiency. It plays a crucial role in risk management by identifying, assessing, and mitigating various risks such as credit, market, and operational risks. Additionally, the application ensures regulatory compliance, assisting banks in adhering to complex local and international banking regulations, anti-money laundering laws, and know your customer requirements. By facilitating optimal asset and liability management, the application aids in maintaining liquidity and making strategic investment decisions. The inclusion of a customer relationship management module enables personalized services, product recommendations, and efficient complaint resolution, contributing to enhanced customer satisfaction. Moreover, the application supports strategic planning, providing tools for long-term vision, scenario analysis, and performance monitoring. It also addresses human resource management needs, including talent acquisition and development. Overall, a bank management application is a vital tool that not only ensures the smooth operation of a bank but also positions it to adapt, innovate, and thrive in a dynamic and competitive financial landscape.

List of similar applications :

|  |  |
| --- | --- |
| Application Name | URL |
| KVB | https://www.kvb.co.in/ |
| SBI | https://www.onlinesbi.sbi/ |
| BANK OF INDIA | https://bankofindia.co.in/ |
| SOUTH INDIA | https://www.southindianbank.com/ |

**2.Section-2**

**DDL,DML,TCL operations**

**Table Details**

**Other than user Table there should be five master and six transaction tables**

**Master Table**

|  |  |
| --- | --- |
| **Table Name** | **Purpose** |
| **customer** | **The customer table serves as a fundamental repository within a business database, particularly in sectors such as banking and finance. Its primary purpose lies in systematically organizing and storing critical information pertaining to each customer or client associated with the business. By storing unique identifiers, personal details, and authentication information, the customer table facilitates effective differentiation between customers and ensures the security of sensitive data. Linked to account information, this table plays a pivotal role in efficient account management, recording transaction history, balances, and statuses.** |
| **Account** | **The account table stands as a cornerstone within the database infrastructure of financial institutions, particularly in the realm of banks. Its central purpose lies in the systematic organization and storage of information pertaining to individual accounts held by customers. With unique identifiers, it facilitates accurate and efficient account identification. Linked to the customer table, it establishes a vital association between account information and individual clients, enabling personalized services and effective customer relationship management. This table comprehensively records essential details such as account types, characteristics, balances, and transaction histories, offering real-time insights into the financial activities associated with each account.** |
| **Branch** | **The branch table within a database plays a pivotal role in managing the complex structure of organizations with multiple physical locations, such as banks or retail networks. Serving as a comprehensive repository, its primary purpose is to systematically organize and store essential information related to each branch or location affiliated with the organization. This includes unique identifiers for accurate branch identification, detailed location information for operational and regulatory purposes, and contact details facilitating effective communication. The table captures operating hours, enabling customers and staff to know when services are available** |
| **Product** | **The product table in a database serves as a fundamental component, particularly within the context of financial institutions or businesses offering various products and services. Its primary purpose is to systematically organize and store information related to each product offered by the organization. The product table includes unique identifiers for accurate product identification, detailed descriptions, and specifications. It records essential details such as pricing, terms and conditions, and any associated fees. By maintaining this structured information, the product table facilitates effective product management, pricing strategies, and marketing efforts. It forms the basis for generating reports and analyses, allowing businesses to assess product performance, identify trends, and make informed decisions to enhance their product portfolio.** |
|  |  |

**TRANSACTION**

|  |  |
| --- | --- |
| **Deposit** | **The deposit table in a database is a fundamental component, particularly within financial institutions, designed to systematically organize and store information related to customer deposits. This table includes unique identifiers for each deposit account, detailed account information, and transaction history. It captures data on various deposit types, such as savings or checking accounts, and records details of each deposit, including date, amount, and any associated fees. The deposit table is essential for monitoring account balances, facilitating accurate financial reporting, and supporting customer inquiries. Additionally, it plays a crucial role in compliance by documenting transaction details required for regulatory reporting and audits. This table is central to managing customer funds, ensuring transparency, and enhancing overall banking operations.** |
| **Product bill payment** | **The product bill payment table is a specialized database component focused on organizing and storing information related to bill payments for various products and services offered by a business. It includes unique identifiers for each bill payment transaction, customer details, and particulars of the payment, such as amount, date, and payment method. This table facilitates efficient tracking of customer payments, ensuring timely and accurate processing. It supports customer relationship management by recording payment histories and streamlining the resolution of any payment-related issues. The product bill payment table is crucial for financial reconciliation, enabling businesses to maintain accurate records of revenue collection and assess the performance of their payment processes.** |
| **Loan payment** | **The loan payment table within a database is dedicated to managing and storing information pertinent to loan repayments. It includes unique identifiers for each loan payment transaction, details about the loan, payment amounts, dates, and any associated fees or interest. The table helps in monitoring the repayment status of loans, tracking outstanding balances, and ensuring compliance with agreed-upon terms. It is instrumental in risk management by assessing borrower repayment patterns and identifying potential defaults. Additionally, the loan payment table aids in financial reporting, providing a comprehensive overview of loan portfolio performance and supporting regulatory compliance through the documentation of loan repayment transactions.** |
| **Withdrawal** | **The withdrawal table is a critical component within a financial institution's database, designed to organize and store information related to customer withdrawals. It includes unique identifiers for each withdrawal transaction, customer details, and particulars of the withdrawal, such as amount, date, and location. This table is essential for monitoring customer account activity, managing liquidity, and ensuring accurate financial reporting. It supports compliance by documenting withdrawal transactions for regulatory reporting and auditing purposes. The withdrawal table plays a crucial role in customer service, providing a record of transactions for customer inquiries and dispute resolution.** |
| **Transfer** | **The transfer table in a database is focused on systematically organizing and storing information related to fund transfers between accounts. It includes unique identifiers for each transfer transaction, details of the accounts involved, transfer amounts, dates, and any associated fees. This table facilitates the tracking of funds moving within the financial institution, supporting efficient reconciliation and ensuring accurate financial reporting. Additionally, it plays a crucial role in customer service, providing a record of transfer transactions for customer inquiries and dispute resolution. The transfer table is integral to risk management, helping monitor and analyze patterns of fund movements to identify any irregularities or potential fraudulent activities.** |

**For Each master table:<<Five table other than user table>>**

**Table Name:customer**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Datatype** | **Constraint** |
| **Customer\_id** | **int** | **Primary key** |
| **Customer\_name** | **Varchar(20)** | **Not null** |
| **balance** | **int** | **Not null** |
| **Phone\_number** | **int** | **Not null** |

**Table Name:Accounts**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **Constraints** |
| **Account number** | **int** | **Primary key** |
| **Account title** | **Varchar(25)** | **unique** |
| **Account type** | **Varchar(25)** | **unique** |
| **Postal address** | **Varchar(255)** | **unique** |
| **Phone number** | **int** | **unique** |
| **Email** | **Varchar(255)** | **notnull** |
| **Occupation** | **Varchar(255)** | **Not null** |

**Table Name:Branch**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **Constraints** |
| **Branch id** | **int** | **Primary key** |
| **location** | **Varchar(20)** | **Not null** |
| **Manager\_details** | **Varchar(30)** | **Not null** |
| **Operating\_hours** | **int** | **Not null** |

**Table Name:Product**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **Constraints** |
| **Product id** | **int** | **Primary key** |
| **Product name** | **Varchar(20)** | **Not null** |
| **description** | **Varchar(50)** | **Not null** |
| **terms** | **int** | **Not null** |
| **Interest rate** | **float** | **unique** |

**Table Name:loan**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Constraints** |
| **Loan id** | **int** | **Primary key** |
| **Loan amount** | **int** | **Not null** |
| **Loan date** | **date** | **Not null** |
| **Repayment schedule** | **int** | **Not null** |

**TRANSACTION TABLE**

**Table Name:Deposit**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Constraints** |
| **Deposit id** | **int** | **Primary key** |
| **Account number** | **int** | **Foreign key** |
| **Deposit date** | **date** | **Not null** |
| **amount** | **int** | **Not null** |

**Table Name:loan\_payment**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **Constraints** |
| **Payment id** | **int** | **primary** |
| **Loan id** | **int** | **Foreign key** |
| **Payment amount** | **int** | **Not null** |
| **payment date** | **date** | **Not null** |

**Table Name:product\_bill\_payment**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **constraints** |
| **Bill payment id** | **int** | **Primary key** |
| **Product id** | **int** | **Foreign key** |
| **Payment amount** | **int** | **Not null** |
| **paymentdate** | **date** | **Not null** |
| **Payment type** | **Varchar(25)** | **Not null** |

**Table Name:withdrawal**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **Constraints** |
| **Withdrawal id** | **int** | **Primary key** |
| **Account number** | **int** | **Foreign key** |
| **Withdrawal date** | **date** | **Not null** |
| **Amount** | **int** | **Not null** |

**Table Name:Transfer**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Data type** | **Constraints** |
| **Transfer id** | **int** | **Primary key** |
| **source** | **Varchar(20)** | **Not null** |
| **Customer id** | **int** | **Foreign key** |
| **destination** | **Varchar(255)** | **Not null** |
| **amount** | **int** | **Not null** |
| **Transfer date** | **date** | **Not null** |

**Table Name:Transaction**

|  |  |  |
| --- | --- | --- |
| **Column name** | **Datatype** | **constraints** |
| **Transaction id** | **int** | **Primary key** |
| **Account number** | **int** | **Foreign key** |
| **Transaction date** | **date** | **Not null** |
| **description** | **Varchar(50)** | **Not null** |
| **Amount** | **int** | **Not null** |
| **Transaction type** | **Varchar(20)** | **Not null** |

**Operations (DDL,DML,TCL):**

**1.Create**

* create table customers(customer\_id int PRIMARY KEY,customer\_name varchar(25) not null,balance int not null , phone\_number int not null)
* CREATE TABLE Accounts (

AccountNumber INT PRIMARY KEY,

AccountTitle VARCHAR(255),

AccountType VARCHAR(255),

PostalAddress VARCHAR(255),

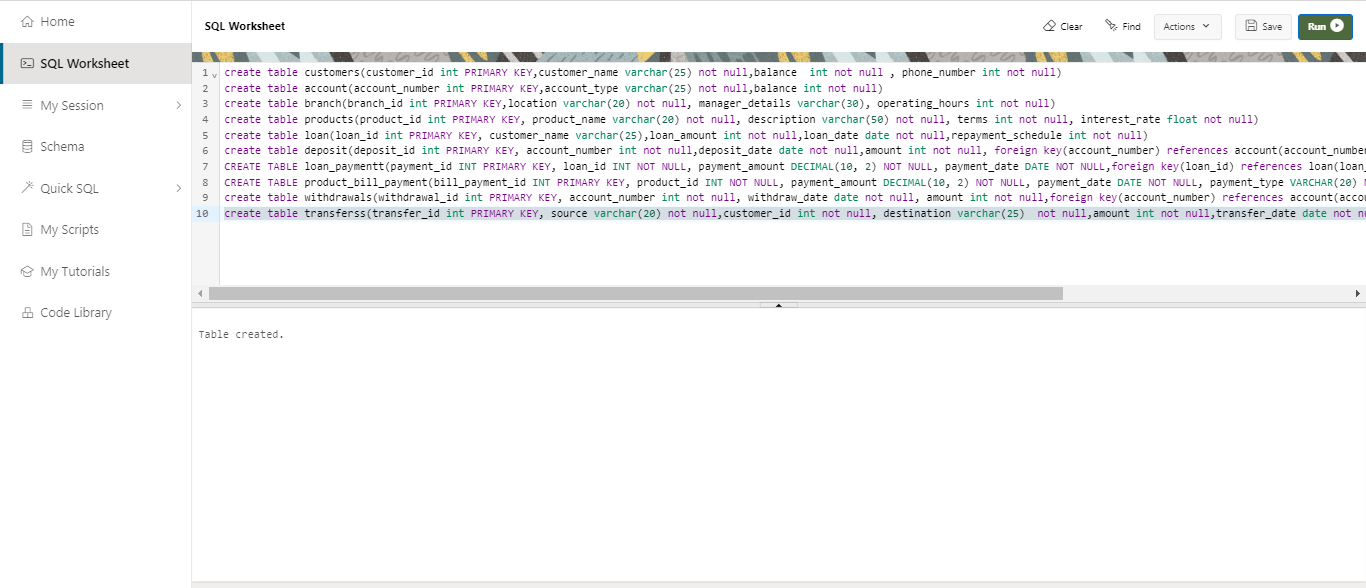
PhoneNumber VARCHAR(15),

EmailAddress VARCHAR(255),

Occupation VARCHAR(255)

);

* create table branch(branch\_id int PRIMARY KEY,location varchar(20) not null, manager\_details varchar(30), operating\_hours int not null)
* create table products(product\_id int PRIMARY KEY, product\_name varchar(20) not null, description varchar(50) not null, terms int not null, interest\_rate float not null)
* create table loan(loan\_id int PRIMARY KEY, customer\_name varchar(25),loan\_amount int not null,loan\_date date not null,repayment\_schedule int not null)



TRANSACTION TABLE -CREATION

\*create table deposit(deposit\_id int PRIMARY KEY, account\_number int not null,deposit\_date date not null,amount int not null, foreign key(account\_number) references account(account\_number))

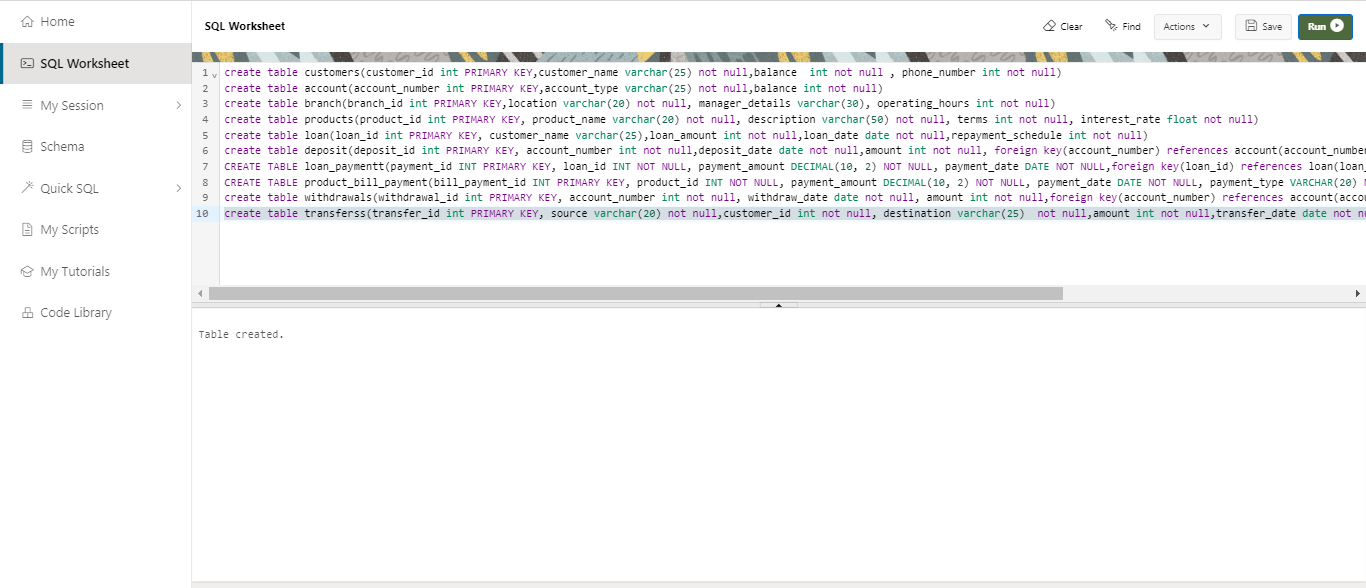
\*CREATE TABLE loan\_paymentz(payment\_id INT PRIMARY KEY, loan\_id INT NOT NULL, payment\_amount int NOT NULL, payment\_date DATE NOT NULL,foreign key(loan\_id) references loan(loan\_id))

\*CREATE TABLE product\_bill\_paymentt (bill\_payment\_id INT PRIMARY KEY, product\_id INT NOT NULL, payment\_amount DECIMAL(10, 2) NOT NULL, payment\_date DATE NOT NULL, payment\_type VARCHAR(20) NOT NULL,foreign key(product\_id) references product(product\_id))

\*create table withdrawals(withdrawal\_id int PRIMARY KEY, AccountNumber int not null, withdraw\_date date not null, amount int not null,foreign key(AccountNumber) references Account(AccountNumber))

\*create table transfer(transfer\_id int PRIMARY KEY, source varchar(20) not null,customer\_id int not null, destination not null,amount int not null,transfer\_date date not null, foreign key(customer\_id) references customers(customer\_id))

\*CREATE TABLE Transactionn (TransactionID INT PRIMARY KEY, AccountNumber INT,TransactionDate DATE,Description VARCHAR(255),Amount float,TransactionType VARCHAR(20),FOREIGN KEY (AccountNumber) REFERENCES Accounts(AccountNumber))



**2.insert**

MASTER TABLE-INSERTION

CUSTOMER TABLE INSERTION 1

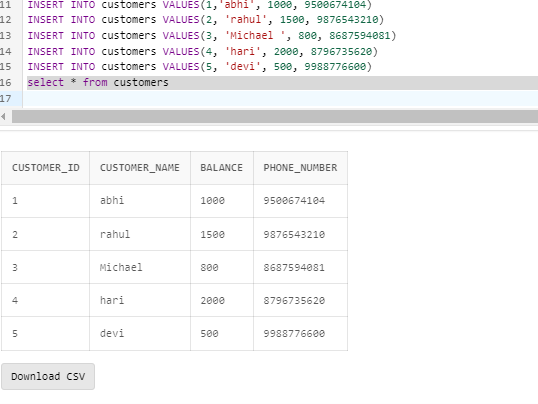
INSERT INTO customers VALUES(1,'abhi', 1000, 9500674104)

INSERT INTO customers VALUES(2, 'rahul', 1500, 9876543210)

INSERT INTO customers VALUES(3, 'Michael ', 800, 8687594081)

INSERT INTO customers VALUES(4, 'hari', 2000, 8796735620)

INSERT INTO customers VALUES(5, 'devi', 500, 9988776600)



ACCOUNTS TABLE INSERTION 2

INSERT INTO Accounts VALUES

(1001, 'abhi', 'Savings', '123 Main Street, Cityville', '(555) 123-4567', 'abhi.doe@email.com', 'Engineer')

INSERT INTO Accounts VALUES

(1002, 'rahul', 'Checking', '456 Elm Street, Townsville', '(555) 987-6543', 'rahul.smith@email.com', 'Teacher')

INSERT INTO Accounts VALUES

(1003, 'Micheal ', 'Business', '789 main Road, Metropolis', '(555) 111-2222', 'micheal.johnson@email.com', 'Entrepreneur')

INSERT INTO Accounts VALUES

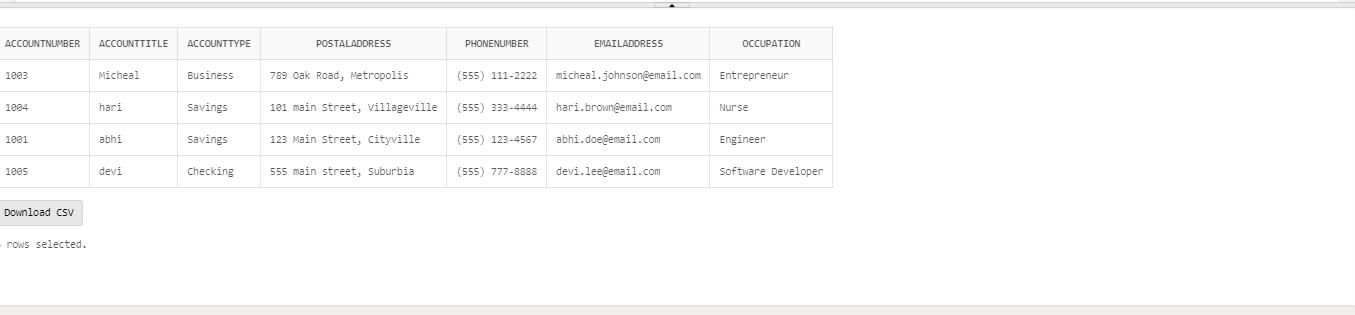
(1004, 'hari', 'Savings', '101 main Street, Villageville', '(555) 333-4444', 'hari.brown@email.com', 'Nurse')

INSERT INTO Accounts VALUES

(1005, 'devi', 'Checking', '555 main street, Suburbia', '(555) 777-8888', 'devi.lee@email.com', 'Software Developer')

select \* from Accounts

INSERT INTO account VALUES(1000000011, 'Savings', 1000)

BRANCH TABLE INSERTION 3

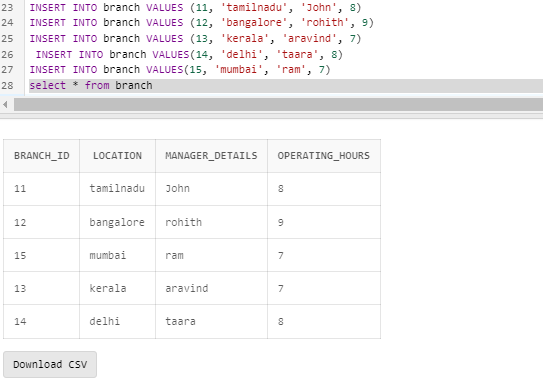
INSERT INTO branch VALUES (11, 'tamilnadu', 'John', 8)

INSERT INTO branch VALUES (12, 'bangalore', 'rohith', 9)

INSERT INTO branch VALUES (13, 'kerala', 'aravind', 7)

INSERT INTO branch VALUES(14, 'delhi', 'taara', 8)

INSERT INTO branch VALUES(15, 'mumbai', 'ram', 7)



PRODUCT TABLE INSERTION 4

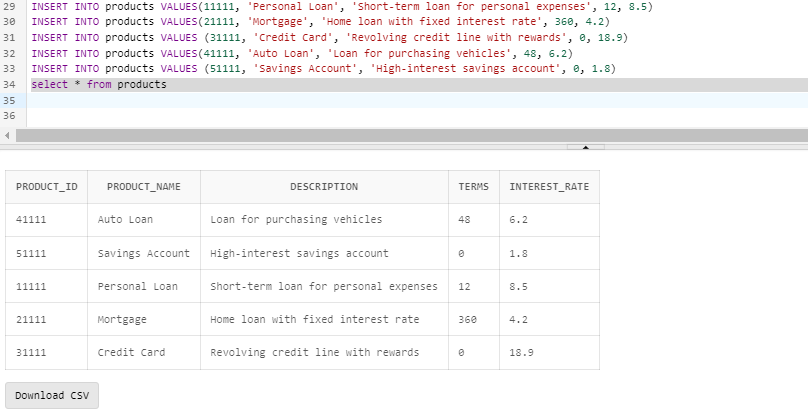
INSERT INTO products VALUES(11111, 'Personal Loan', 'Short-term loan for personal expenses', 12, 8.5)

INSERT INTO products VALUES(21111, 'Mortgage', 'Home loan with fixed interest rate', 360, 4.2)

INSERT INTO products VALUES (31111, 'Credit Card', 'Revolving credit line with rewards', 0, 18.9)

INSERT INTO products VALUES(41111, 'Auto Loan', 'Loan for purchasing vehicles', 48, 6.2)

INSERT INTO products VALUES (51111, 'Savings Account', 'High-interest savings account', 0, 1.8)



LOAN TABLE TRANSACTION 5

INSERT INTO loan VALUES

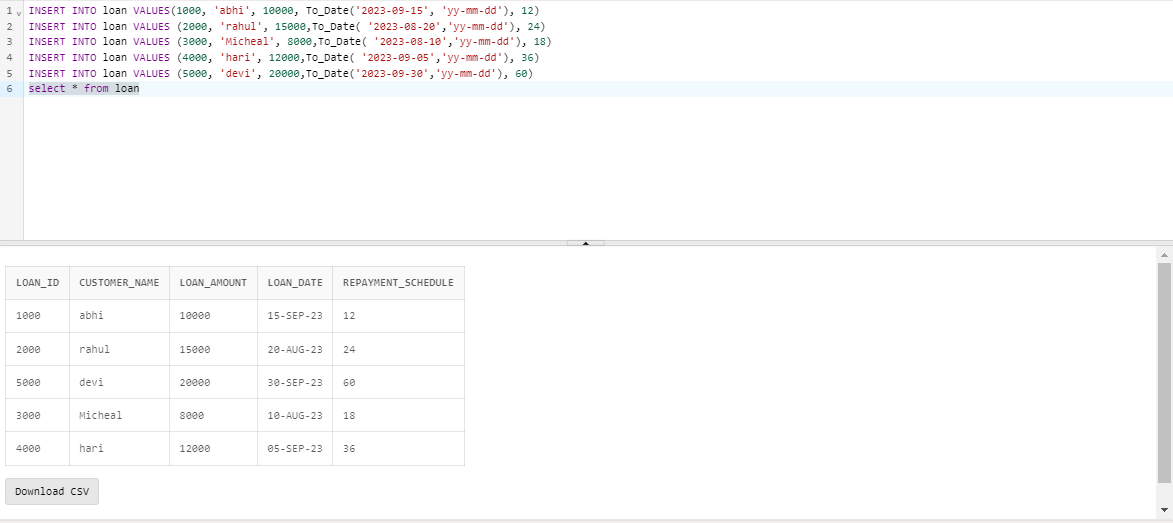
INSERT INTO loan VALUES(1000, 'abhi', 10000, To\_Date('2023-09-15', 'yy-mm-dd'), 12)

INSERT INTO loan VALUES (2000, 'rahul', 15000,To\_Date( '2023-08-20','yy-mm-dd'), 24)

INSERT INTO loan VALUES (3000, 'Micheal', 8000,To\_Date( '2023-08-10','yy-mm-dd'), 18)

INSERT INTO loan VALUES (4000, 'hari', 12000,To\_Date( '2023-09-05','yy-mm-dd'), 36)

INSERT INTO loan VALUES (5000, 'devi', 20000,To\_Date('2023-09-30','yy-mm-dd'), 60)



TRANSACTION INSERTION

DEPOSIT TABLE INSERTION 1

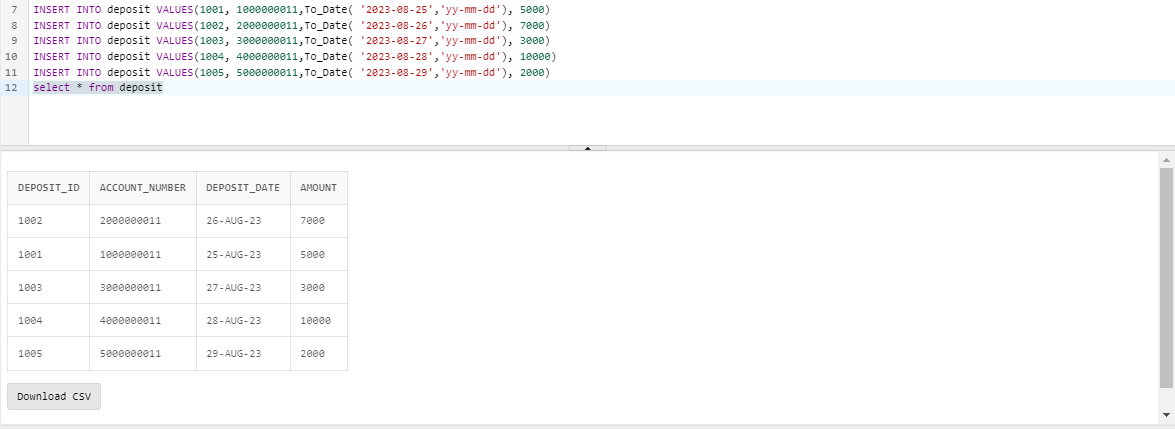
INSERT INTO deposit VALUES(1001, 1000000011,To\_Date( '2023-08-25','yy-mm-dd'), 5000)

INSERT INTO deposit VALUES(1002, 2000000011,To\_Date( '2023-08-26','yy-mm-dd'), 7000)

INSERT INTO deposit VALUES(1003, 3000000011,To\_Date( '2023-08-27','yy-mm-dd'), 3000)

INSERT INTO deposit VALUES(1004, 4000000011,To\_Date( '2023-08-28','yy-mm-dd'), 10000)

INSERT INTO deposit VALUES(1005, 5000000011,To\_Date( '2023-08-29','yy-mm-dd'), 2000)



LOAN PAYMENT TABLE INSERTION 2

CREATE TABLE loan\_paymentz(payment\_id INT PRIMARY KEY, loan\_id INT NOT NULL, payment\_amount int NOT NULL, payment\_date DATE NOT NULL,foreign key(loan\_id) references loan(loan\_id))

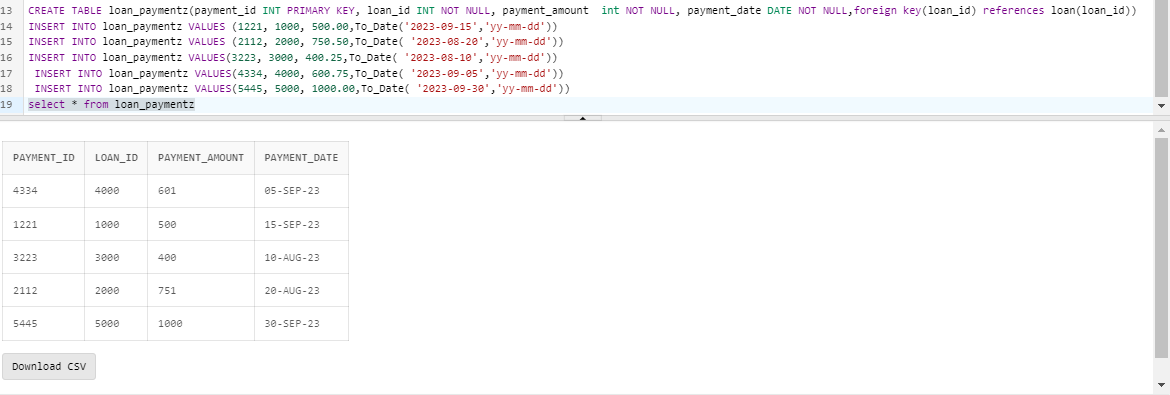
INSERT INTO loan\_paymentz VALUES (1221, 1000, 500.00,To\_Date('2023-09-15','yy-mm-dd'))

INSERT INTO loan\_paymentz VALUES (2112, 2000, 750.50,To\_Date( '2023-08-20','yy-mm-dd'))

INSERT INTO loan\_paymentz VALUES(3223, 3000, 400.25,To\_Date( '2023-08-10','yy-mm-dd'))

INSERT INTO loan\_paymentz VALUES(4334, 4000, 600.75,To\_Date( '2023-09-05','yy-mm-dd'))

INSERT INTO loan\_paymentz VALUES(5445, 5000, 1000.00,To\_Date( '2023-09-30','yy-mm-dd'))



PRODUCT BILL PAYMENT INSERTION 3

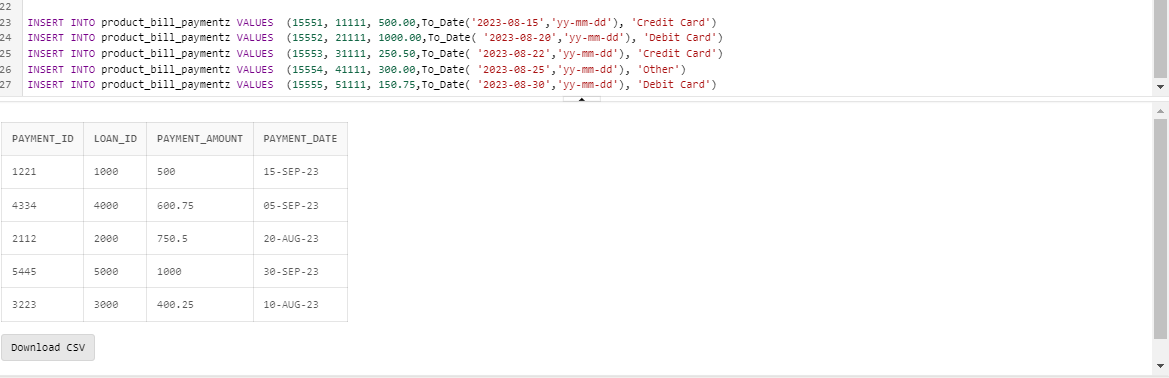
INSERT INTO product\_bill\_paymentz VALUES (15551, 11111, 500.00,To\_Date('2023-08-15','yy-mm-dd'), 'Credit Card')

INSERT INTO product\_bill\_paymentz VALUES (15552, 21111, 1000.00,To\_Date( '2023-08-20','yy-mm-dd'), 'Debit Card')

INSERT INTO product\_bill\_paymentz VALUES (15553, 31111, 250.50,To\_Date( '2023-08-22','yy-mm-dd'), 'Credit Card')

INSERT INTO product\_bill\_paymentz VALUES (15554, 41111, 300.00,To\_Date( '2023-08-25','yy-mm-dd'), 'Other')

INSERT INTO product\_bill\_paymentz VALUES (15555, 51111, 150.75,To\_Date( '2023-08-30','yy-mm-dd'), 'Debit Card')



WITHDRAWAL INSERTION 4

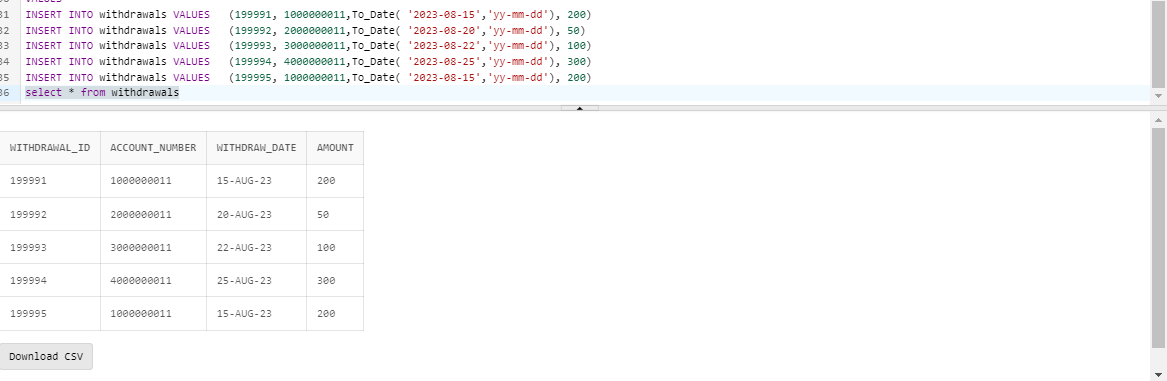
INSERT INTO withdrawals VALUES (199991, 1000000011,To\_Date( '2023-08-15','yy-mm-dd'), 200)

INSERT INTO withdrawals VALUES (199992, 2000000011,To\_Date( '2023-08-20','yy-mm-dd'), 50)

INSERT INTO withdrawals VALUES (199993, 3000000011,To\_Date( '2023-08-22','yy-mm-dd'), 100)

INSERT INTO withdrawals VALUES (199994, 4000000011,To\_Date( '2023-08-25','yy-mm-dd'), 300)

INSERT INTO withdrawals VALUES (199995, 1000000011,To\_Date( '2023-08-15','yy-mm-dd'), 200)



TRANSFER INSERTION 5

INSERT INTO transfers VALUES

(55551, 'Account A', 1, 'Account B', 500,To\_Date( '2023-08-15','yy-mm-dd'))

INSERT INTO transfers VALUES

(55552, 'Account C', 2, 'Account D', 750,To\_Date( '2023-08-20','yy-mm-dd'))

INSERT INTO transfers VALUES

(55553, 'Account E', 3, 'Account F', 1000,To\_Date( '2023-08-22','yy-mm-dd'))

INSERT INTO transfers VALUES

(55554, 'Account G', 4, 'Account H', 250,To\_Date( '2023-08-25','yy-mm-dd'))

INSERT INTO transfers VALUES

(55555, 'Account G', 4, 'Account H', 250,To\_Date( '2023-08-25','yy-mm-dd'))



TRANSACTION INSERTION

INSERT INTO Transactionn VALUES (1,1001,To\_date('2023-10-11',’yy-mm-dd’), 'Deposit', 1000.00, 'Deposit')

INSERT INTO Transactions VALUES (2,1002 ,To\_date('2023-10-12',’yy-mm-dd’), 'Withdrawal', 500.00, 'Withdrawal')

INSERT INTO Transactions VALUES (5,1003,To\_date( '2023-10-12',’yy-mm-dd’), 'Transfer to 2', 200.00, 'Transfer')

INSERT INTO Transactions VALUES (3,1004,To\_date( '2023-10-13',’yy-mm-dd’), 'Withdrawal', 300.00, 'Withdrawal')

INSERT INTO Transactions VALUES (4,1005To\_date(, '2023-10-14',’yy-mm-dd’), 'Deposit', 1500.00, 'Deposit')

**3.Delete**

* Delete from Customer Where Customer\_id = 111

**4.Update**

* Update product\_bill\_paymentz set payment\_type=credit card where billpayment\_id= “15553”

**…**

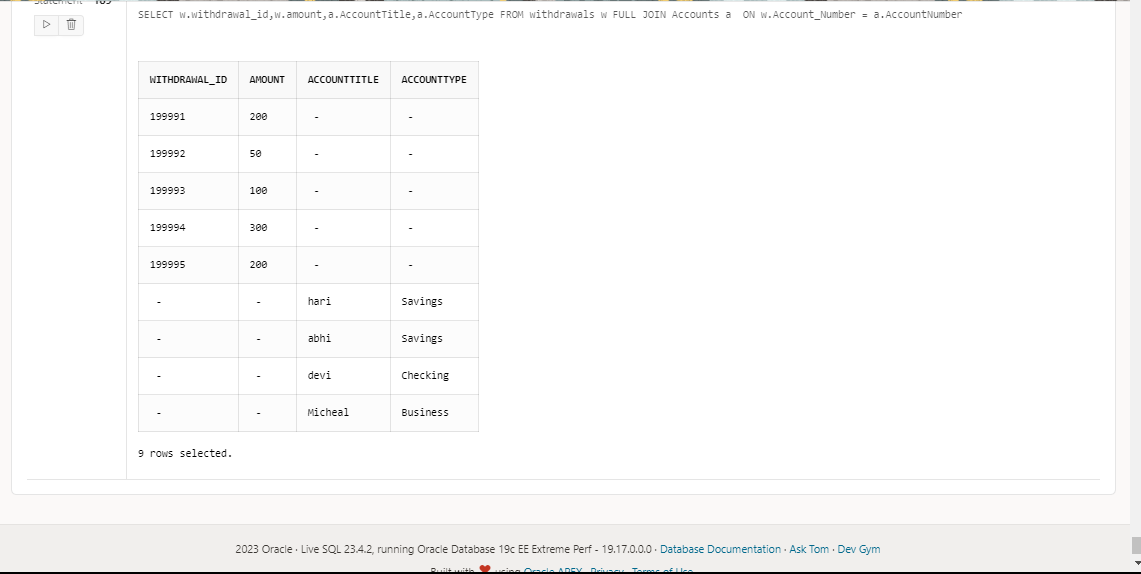
**-- Sql query file name<<File.sql>>**

**Screenshot:**

**<<Place outputs of execution>>**

**3.Section-3(Join)**

**Screenshot:**

****

**-- Sql query file name<<File.sql>>**

**Screenshot:**

**<<Place outputs of execution>>**

**Inference:**

**The loan and the loan payment are related by the loan id**

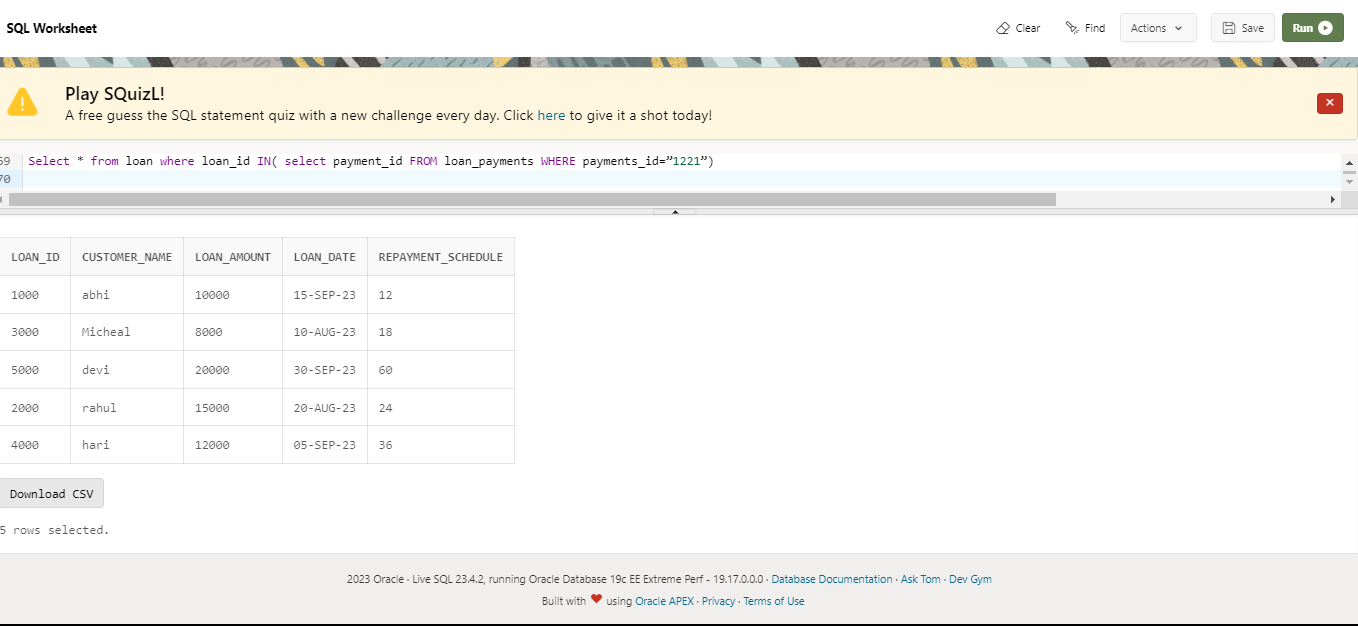
**This relation enables tracking between loan amount and the customer name.**

**4.Section-4(Nested Queries)**

**Screenshot:**

**Select \* from loan where loan\_id IN( select payment\_id FROM loan\_payments WHERE payments\_id=”1221”)**

**<<Place outputs of execution>>**

****

**Inference:**

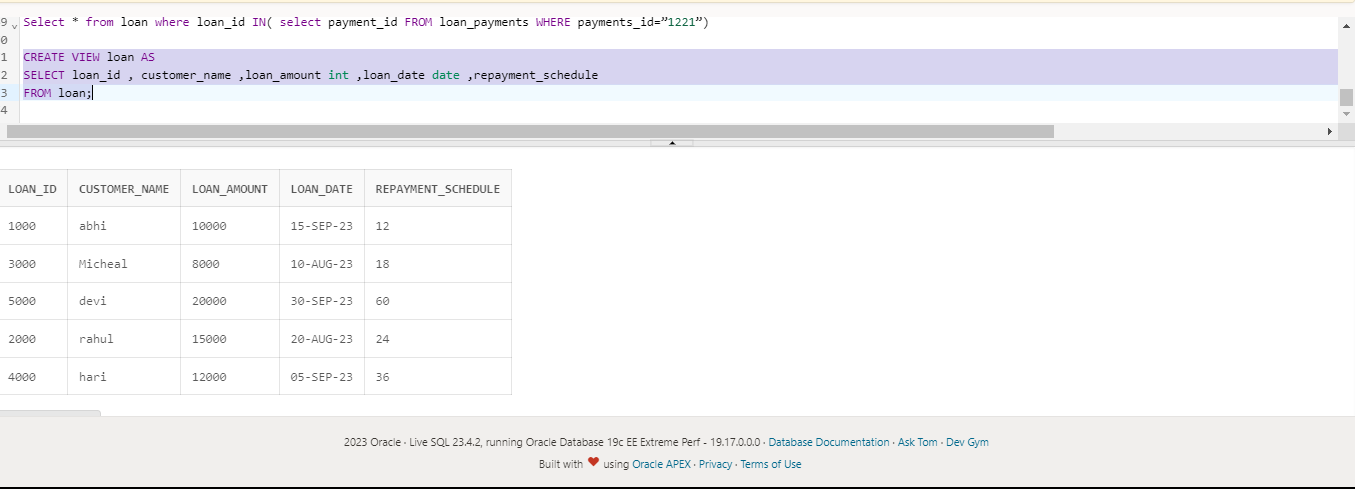
**For every loan there is a loan payment id and a method . using that a loan buyer can pay the loan amount**

**5.Section-5(view)**

**CREATE VIEW loan AS**

**SELECT loan\_id , customer\_name ,loan\_amount int ,loan\_date date ,repayment\_schedule**

**FROM loan;**

**Screenshot: **

**Inference: loanDetails will retrieve and display only these selected columns from the loan table, allowing users to view and work with this limited set of information without directly altering the original table.**

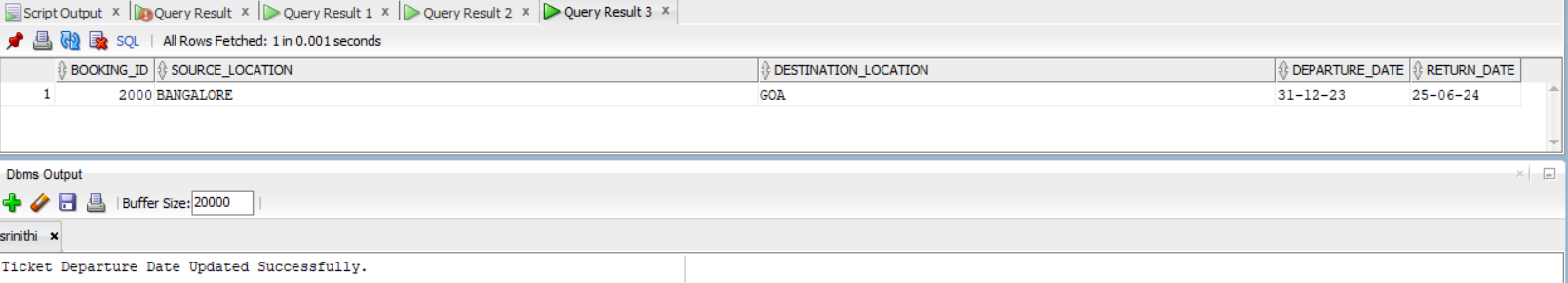
**6. Section-6(PLSQL file)**

**<<Can be more than one PL/SQL but should have function>>**

|  |  |
| --- | --- |
| **Function Name** | **Get total amount of a loan** |
| **Procedure Name** | **Update payment date** |
| **Expected Output** | **Function: the function calculates and returns the loan amount for a customer** |
| **Tables operated** | **Loan ,loan payment** |

**-- Sql query file name<<File.sql>>**

**Screenshot:**



**<<Place outputs of execution>>**

**Inference:**

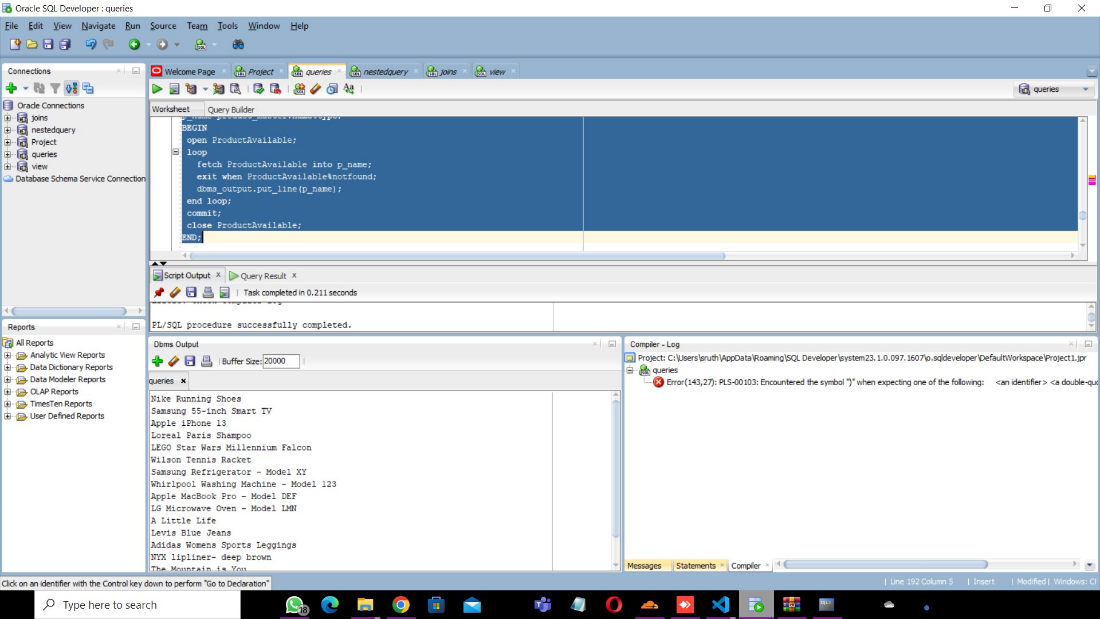
**<<Explain what you would like to explain about the output>>**

**7. Section-7(Cursor file)**

**<<Can be more than one cursor file>>**

|  |  |
| --- | --- |
| **Cursor Name** | **Explanation of the cursor** |
| **…** | **To retrive all the loan that are available in the bank by making the cursor point** |
| **Expected Output** | **Loan name** |

**-- Sql query file name<<File.sql>>**

**Screenshot:** 

**Inference:**

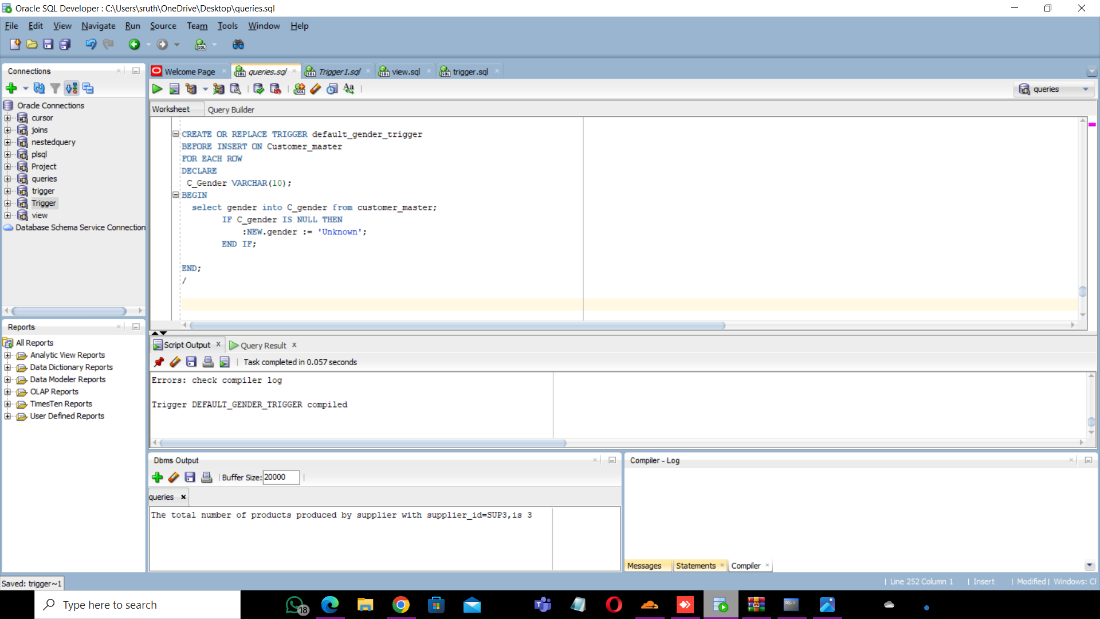
**<<Explain what you would like to explain about the output>>**

**8. Section-8(Trigger file)**

**<<Can be more than one trigger operation>>**

|  |  |
| --- | --- |
| **Cursor Name** | **1.Explanation of the trigger operation** |
| **Loans available** | **This trigger could be designed to respond to factors such as customer eligibility, credit score updates, or changes in the financial landscape. For instance, it may be set to activate when a customer's credit score reaches a predefined threshold, signaling that they now qualify for specific loan products. Alternatively, it might be triggered by the introduction of new loan products or changes in interest rates. The trigger could also be designed to consider factors such as a customer's financial behavior, such as consistent timely payments, which could make them eligible for additional loan options. By implementing this trigger, banks can automate the process of notifying eligible customers about available loans, providing a proactive and responsive approach to meeting customer financial needs.** |
| **Expected Output** | **The expected output of the trigger for available loans in a bank would be the initiation of a notification or alert system. When the trigger conditions are met, the system would generate notifications to eligible customers, informing them about the availability of specific loans. These notifications could be sent through various channels such as email, SMS, or mobile app notifications. The content of the notification would include details about the available loan products, terms, and application procedures. The trigger might also initiate updates in the bank's customer relationship management (CRM) system, marking the customer as eligible for specific loan offers. Overall, the expected output is a timely and personalized communication mechanism that enhances customer engagement and allows the bank to efficiently leverage opportunities to offer tailored financial solutions to eligible customers.** |

**-- Sql query file name<<File.sql>>**

**Screenshot:** 

**Inference:The trigger default \_gender\_trigger makes sure to check ,before insertion of each row that it**

**9.Section -9**

**<< Web Application>>**

* **5 Master Tables**
* **6 Transaction table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Form Name** | **Category(Master**  **/Transaction)** | **Table Name associated with the form** | **Form Name /css**  **For ex: Emp\_master.html**  **Emp\_master.css** | **File Name for reference in the shared drive** | **Type of operations**  **(Insert/Update/Delete**  **/Search/Display)** |
| **Create Customer form** | **Master form** | **Customer appplication** | **Customer.html** | **Customer.html** | **insert** |
| **Account form** | **Master form** | **Search account** | **Account.html** | **Account.html** | **search** |
| **Branch form** | **Master form** | **Branch details** | **Branch.html** | **Branch.html** | **display** |
| **Product form** | **Master form** | **Product details** | **Product.html** | **Product.html** | **display** |
| **Customer information** | **Master** | **Customer information** | **Customerinformation.html** | **Customer.html** | **display** |
| **Loan details** | **Master** | **Loan details** | **Loan.html** | **Loan.html** | **display** |
| **Check balance** | **Transaction** | **Check balance** | **Check.html** | **Check.html** | **display** |
| **Update existing account** | **Master** | **Update existing account** | **Update.html** | **Update.html** | **update** |
| **Loan payment** | **Transaction** | **Loan bill payment** | **Loan\_payment.html** | **Loan\_payment.html** | **Insert** |
| **Search customer** | **Master** | **Search customer** | **Search.html** | **Search.html** | **search** |
| **Delete account** | **master** | **Delete account** | **Delete.html** | **Delete.html** | **delete** |
| **Product bill payment** | **Transaction** | **Product bill payment** | **Bill\_payment.html** | **Bill\_payment.html** | **insert** |
| **Transfer** | **Transaction** | **Transfer** | **Transfer.html** | **Transfer.html** | **insert** |
| **Transaction** | **Transaction** | **Transactions** | **Transaction.html** | **Transaction.html** | **insert** |
| **Deposit** | **Transaction** | **Deposit** | **Deposit.html** | **Deposit.html** | **insert** |

**Operations :**

**<<Repeat the 1.insert for all the operations delete,update,search,display>>**

**1.Insert**

**<<keep the screenshot for the form operations for all the forms>>**

**Inference**

**--Write 1-2 sentence explaining the operation shown**

**-- record the video of the working output and keep in the drive**

**10.Section-10(No-SQL Application)**

**-- Can show for one Table alone**

**Technologies:**

|  |  |
| --- | --- |
| **Front End** | **HTML,CSS** |
| **Back End** | **MONGO DB** |
| **Editor** | **VISUAL STUDIO CODE** |
| **Language** | **JAVA SCRIPT** |
| **Framework** | **MONGO.JS** |
|  |  |
|  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Form Name** | **Category(Master**  **/Transaction)** | **Table Name associated with the form** | **Form Name /css**  **For ex: Emp\_master.html**  **Emp\_master.css** | **File Name for reference in the shared drive** | **Type of operations**  **(Insert/Update/Delete**  **/Search/Display)** |
| **LOAN FORM** | **MASTER** | **LOAN\_AMOUNT** | **LOAN.HTML** | **LOAN.HTML** | **INSERT/DISPLAY** |

**-- record the video of the working output and keep in the drive**

|  |  |  |
| --- | --- | --- |
| **Section No** | **Mark** | **Marks Awarded** |
| **1** | **5** |  |
| **2** | **10** |  |
| **3** | **5** |  |
| **4** | **5** |  |
| **5** | **5** |  |
| **6** | **10** |  |
| **7** | **10** |  |
| **8** | **10** |  |
| **9** | **30** |  |
| **10** | **10** |  |