REPORT

AIM:

Create a mini project to handle the operations of ATM using SQL queries and PL/ SQL.
(use the concept of Cursor, Trigger, Procedure/ function, Package and Exception Handling)

THEORY:

The theory is whenever a customer want's to Deposit or Withdraw the money from his/her bank , They must have and ATM card but in this project it is not possible , So we have made the Authentication page which will ask a customer for his/her ATM card number and PIN number for doing any task they want.

We created the Five modules.

1. Authentication
2. Deposit Money
3. Withdraw Money
4. Transfer Money
5. Logout

CODE/QUERY:

Prepared By

Marwadi Vikas-(18DCS043)

Adnan Mistry-(18DCS047)

Bharatiya Tirth-(18DCS008)

************************* Tables *****************************

CREATE TABLE Account_Info (account_no number(20),atm_number number(12),pin number(4),balance number(20));

CREATE TABLE Withdraw_Money (account_no number(20),withdraw_amount number(20),w_m_date TIMESTAMP);

CREATE TABLE Deposit_Money (account_no number(20),deposit_amount number(20),d_m_date TIMESTAMP);

CREATE TABLE Transfer_Money (s_account_no number(20),r_account_no number(20),transfer_amount number(20),t_m_date TIMESTAMP);

CREATE TABLE Before_Update_Balance (account_number number(20),balance number(20),date_time TIMESTAMP);

*********************** Insertion *************************

INSERT INTO Account_Info values (12345678,1234,1111,20000);

INSERT INTO Account_Info values (87654321,4321,4444,60000);

************************ Trigger *****************************

```
CREATE OR REPLACE TRIGGER t
BEFORE UPDATE OF balance ON Account_Info
FOR EACH ROW
BEGIN
   INSERT INTO Before_Update_Balance VALUES
(:OLD.account_no,:OLD.balance,SYSTIMESTAMP);
END t;
```

******************* Package Declaration ***************************

```
CREATE PACKAGE exception_package as
   invalid_service_type EXCEPTION;
   invalid_amount EXCEPTION;
   invalid_atm_number EXCEPTION;
   invalid_account_number EXCEPTION;
   invalid_pin EXCEPTION;
   maximum_amount EXCEPTION;
END;
```

************** Package Body Definition *****************

```
CREATE PACKAGE BODY exception_package
AS
   BEGIN
      dbms_output.put_line('This is From Exception Package');
   EXCEPTION
      when invalid_service_type then
         dbms_output.put_line('Please Select service type as in this three Deposit, Withdraw or Transfer');

      when invalid_amount then
         dbms_output.put_line('Please Enter Correct Amount of Money');

      when invalid_account_number then
         dbms_output.put_line('Please Enter Correct Account No of Receiver Account');

      when maximum_amount then
         dbms_output.put_line('Insufficient Balance');

      when invalid_atm_number then
         dbms_output.put_line('Please Enter Correct ATM Number');

      when invalid_pin then
         dbms_output.put_line('Please Enter Correct ATM PIN');
END exception_package;
```

***************** Procedures-Triggers *******************

```
CREATE OR REPLACE PROCEDURE deposit_amount (atm_no
Account_Info.atm_number%type,amount
Deposit_Money.deposit_amount%type) IS
  BEGIN
    DECLARE
      b_u_balance account_info.balance%type;
      final_balance number(20);
    BEGIN
      IF amount = null OR amount = 0
      THEN
         RAISE exception_package.invalid_amount;
      ELSE
         SELECT balance INTO b_u_balance FROM Account_Info WHERE
atm_number = atm_no;
         final_balance:= b_u_balance + amount;
         INSERT INTO Deposit_Money VALUES
(atm_no,amount,SYSTIMESTAMP);
         UPDATE Account_Info SET balance = final_balance WHERE
atm_number = atm_no;
         dbms_output.put_line('Amount ' || amount ||' Successfully Deposit In
your Account.');
      END IF;
    END;
  END deposit_amount;
```

************************** Continue… *************************

```
CREATE OR REPLACE PROCEDURE withdraw (atm_no Account_Info.atm_number%type,amount Withdraw_Money.withdraw_amount%type) IS
   BEGIN
      DECLARE
         b_u_balance account_info.balance%type;
         final_balance number(20);
      BEGIN
         SELECT balance INTO b_u_balance FROM Account_Info WHERE atm_number = atm_no;
         final_balance:= b_u_balance - amount;
         IF amount = null OR amount = 0
         THEN
             RAISE exception_package.invalid_amount;
         ELSIF b_u_balance < 0
         THEN
             RAISE exception_package.maximum_amount;
         ELSE
             INSERT INTO Withdraw_Money VALUES (atm_no,amount,SYSTIMESTAMP);
             UPDATE Account_Info SET balance = final_balance WHERE atm_number = atm_no;
             dbms_output.put_line('Money ' || amount ||' Successfully Withdraw from your Account.');
         END IF;
      END;
   END withdraw;
```

************************** Continue ***************************

```
CREATE OR REPLACE PROCEDURE transfer (atm_no
Account_Info.atm_number%type,r_ac_no
Account_Info.account_no%type,amount
Deposit_Money.deposit_amount%type) IS
   BEGIN
      DECLARE
         s_account_no account_info.balance%type;
         s_b_u_balance account_info.balance%type;
         r_b_u_balance account_info.balance%type;
         s_final_balance number(20);
         r_final_balance number(20);
      BEGIN
         SELECT account_no INTO s_account_no FROM Account_Info WHERE
atm_number = atm_no;
         SELECT balance INTO s_b_u_balance FROM Account_Info WHERE
atm_number = atm_no;
         s_final_balance:=s_b_u_balance - amount;
         SELECT balance INTO r_b_u_balance FROM Account_Info WHERE
account_no = r_ac_no;
         r_final_balance:=r_b_u_balance + amount;

         IF s_b_u_balance = 0 OR s_final_balance < 0
         THEN
             RAISE exception_package.maximum_amount;
         ELSE
             INSERT INTO Transfer_Money VALUES
(s_account_no,r_ac_no,amount,SYSTIMESTAMP);
             UPDATE Account_Info SET balance =
             CASE
                WHEN account_no = s_account_no then s_final_balance
                WHEN account_no = r_ac_no then r_final_balance
             END WHERE account_no in (s_account_no,r_ac_no);
             dbms_output.put_line('Money ' || amount ||' Successfully Transfer from
your Account to Receiver Account.');
         END IF;
      END;
   END transfer;
```

*********************** PLSQL Block ***********************

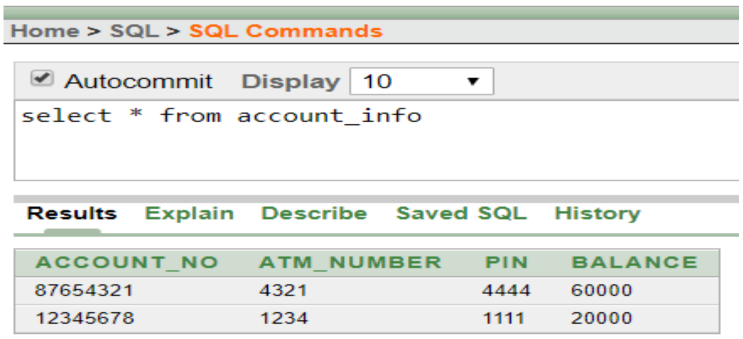
```
DECLARE
   Receiver_account_number account_info.account_no%type;
   atm_no account_info.atm_number%type;
   password account_info.pin%type;
   amount account_info.balance%type;
   a_no number(20);
   r_ac_no number(20);
   p_no number(20);
   Service_Type varchar2(20);
BEGIN
   atm_no:=:atm_no;
   password:=:password;
   Service_Type:=:Service_Type;
   amount:=:amount;
```

```
    Receiver_account_number:=:Receiver_account_number;
    select atm_number into a_no from Account_Info where atm_number = atm_no
and pin = password;
    if a_no = atm_no
    then
            if UPPER(Service_Type) = 'DEPOSIT'
            then
                deposit_amount(atm_no,amount);

            elsif UPPER(Service_Type) = 'WITHDRAW'
            then
                withdraw(atm_no,amount);

            elsif UPPER(Service_Type) = 'TRANSFER'
            then
                transfer(atm_no,Receiver_account_number,amount);
            else
                RAISE exception_package.invalid_service_type;
            end if;
    elsif password = null
    then
        RAISE exception_package.invalid_pin;
    else
        dbms_output.put_line('Invalid ATM Number');
    end if;
END;
```

----x----

RESULT/OUTPUT:

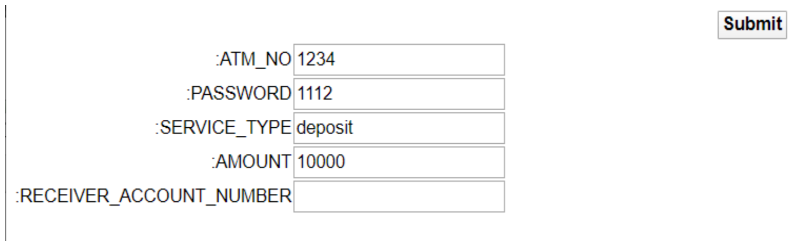
Before Transaction


Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

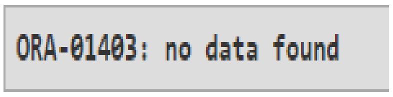
```
select * from account_info
```

ACCOUNT_NO	ATM_NUMBER	PIN	BALANCE
87654321	4321	4444	60000
12345678	1234	1111	20000

Authentication

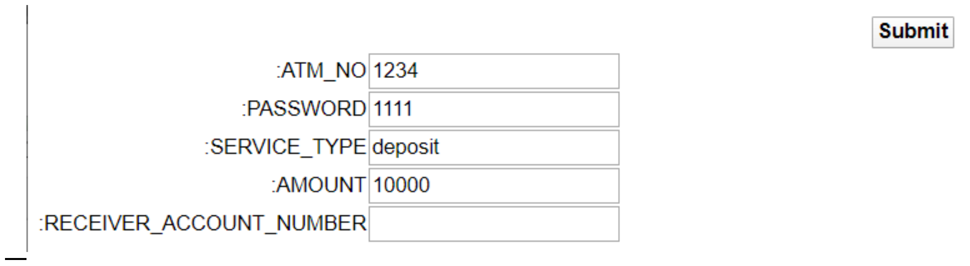


:ATM_NO	1234
:PASSWORD	1112
:SERVICE_TYPE	deposit
:AMOUNT	10000
:RECEIVER_ACCOUNT_NUMBER	

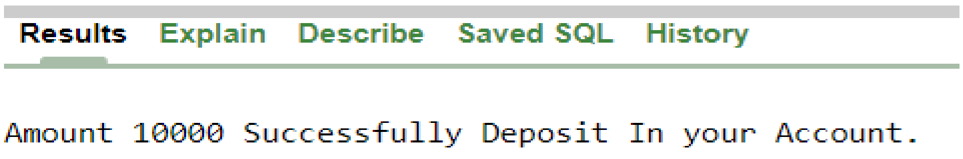


```
ORA-01403: no data found
```

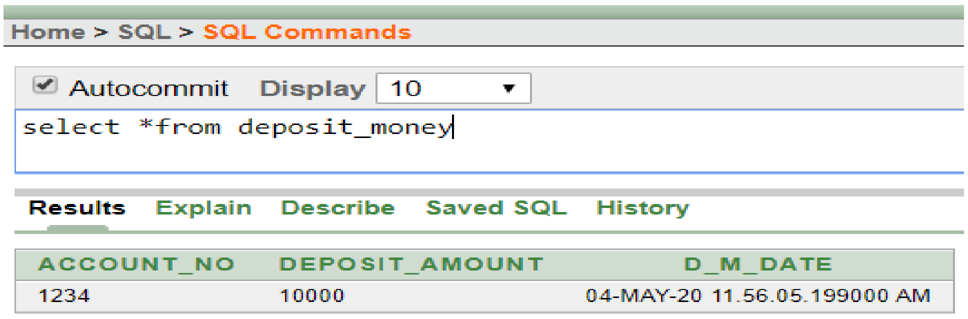
Deposit Money



:ATM_NO	1234
:PASSWORD	1111
:SERVICE_TYPE	deposit
:AMOUNT	10000
:RECEIVER_ACCOUNT_NUMBER	



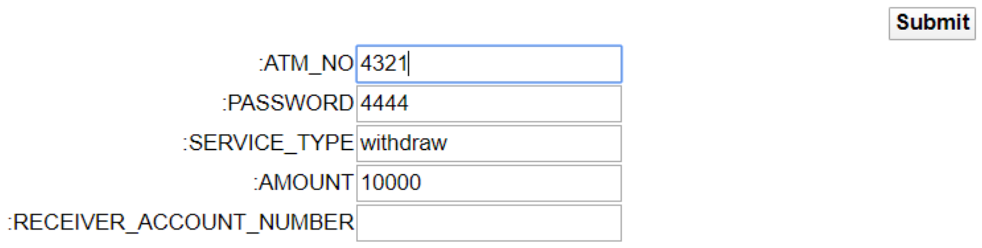
```
Amount 10000 Successfully Deposit In your Account.
```


Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

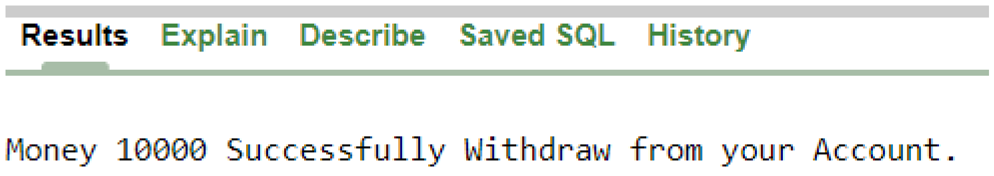
```
select *from deposit_money
```

ACCOUNT_NO	DEPOSIT_AMOUNT	D_M_DATE
1234	10000	04-MAY-20 11.56.05.199000 AM

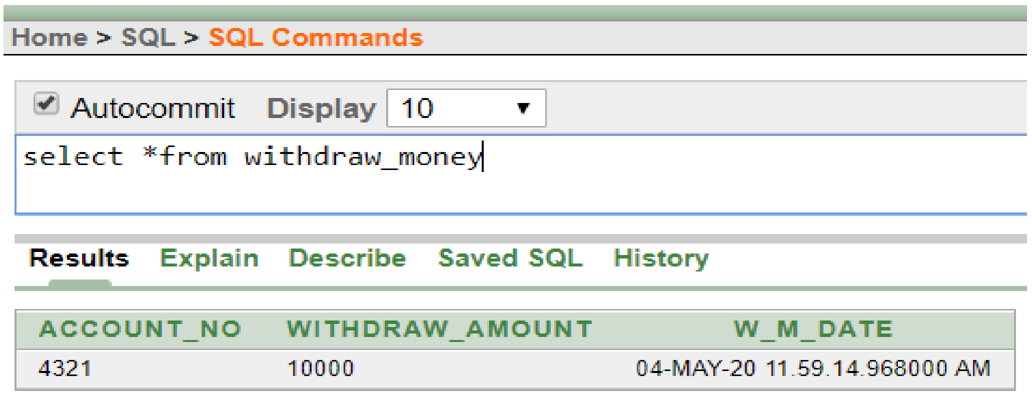
Withdraw Money



:ATM_NO	4321
:PASSWORD	4444
:SERVICE_TYPE	withdraw
:AMOUNT	10000
:RECEIVER_ACCOUNT_NUMBER	



```
Money 10000 Successfully Withdraw from your Account.
```

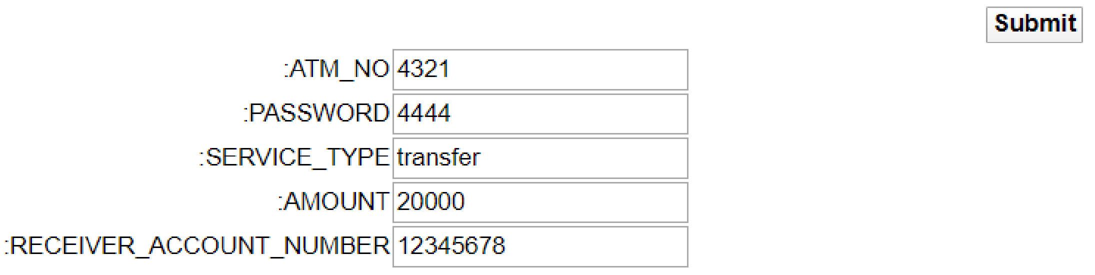


Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
select *from withdraw_money
```

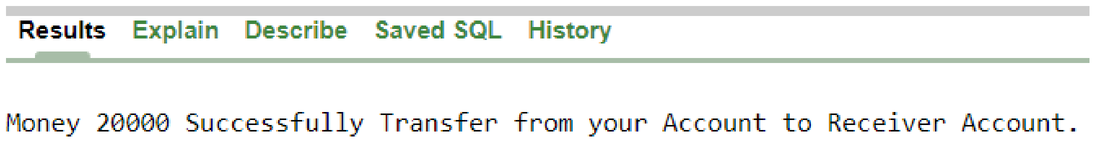
ACCOUNT_NO	WITHDRAW_AMOUNT	W_M_DATE
4321	10000	04-MAY-20 11.59.14.968000 AM

Transfer Money

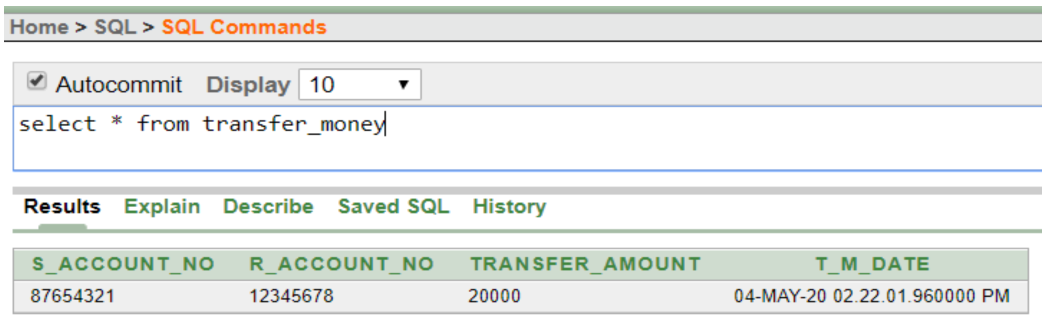


Submit

:ATM_NO	4321
:PASSWORD	4444
:SERVICE_TYPE	transfer
:AMOUNT	20000
:RECEIVER_ACCOUNT_NUMBER	12345678



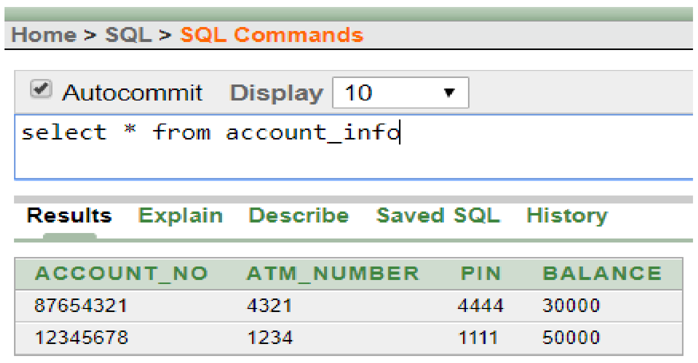
```
Money 20000 Successfully Transfer from your Account to Receiver Account.
```


Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
select * from transfer_money
```

S_ACCOUNT_NO	R_ACCOUNT_NO	TRANSFER_AMOUNT	T_M_DATE
87654321	12345678	20000	04-MAY-20 02.22.01.960000 PM

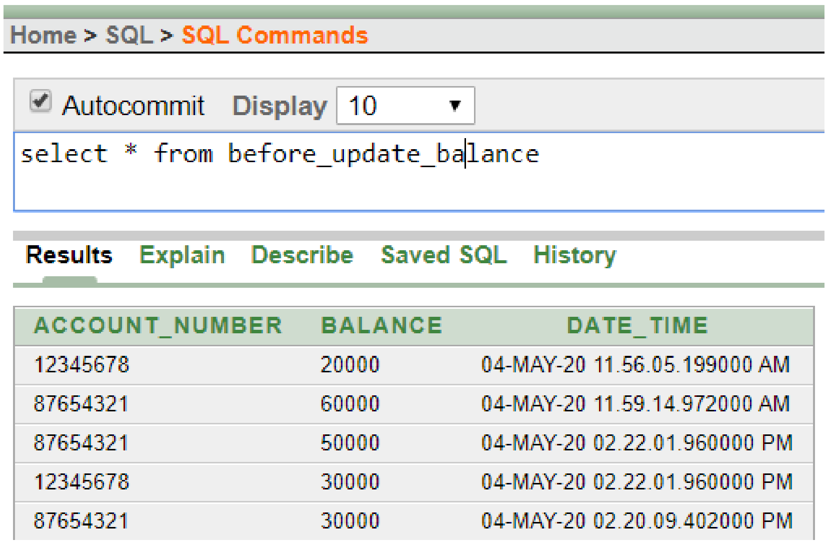
After Transaction


Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
select * from account_info
```

ACCOUNT_NO	ATM_NUMBER	PIN	BALANCE
87654321	4321	4444	30000
12345678	1234	1111	50000

Trigger also Store the balance before updation


Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History

```
select * from before_update_balance
```

ACCOUNT_NUMBER	BALANCE	DATE_TIME
12345678	20000	04-MAY-20 11.56.05.199000 AM
87654321	60000	04-MAY-20 11.59.14.972000 AM
87654321	50000	04-MAY-20 02.22.01.960000 PM
12345678	30000	04-MAY-20 02.22.01.960000 PM
87654321	30000	04-MAY-20 02.20.09.402000 PM

CONCLUSION:

After this project completion we learned how to work with Oracle Database, how to create trigger, procedures, packages and how to use that, also we faced many errors in between run the queries and how to resolved that queries, this all stuff we learned.

We also learned the concept of Exception Handling and different type datatype like timestamp.