B.TECH. (2020-24)

Artificial Intelligence

OPEN ENDED PROBLEM

on

DATABASE MANAGEMENT SYSTEMS
[CSE201]

Submitted To

Dr Gaurav Dubey

Submitted By

HITESH
A023119820027
4CSE11 (AI)

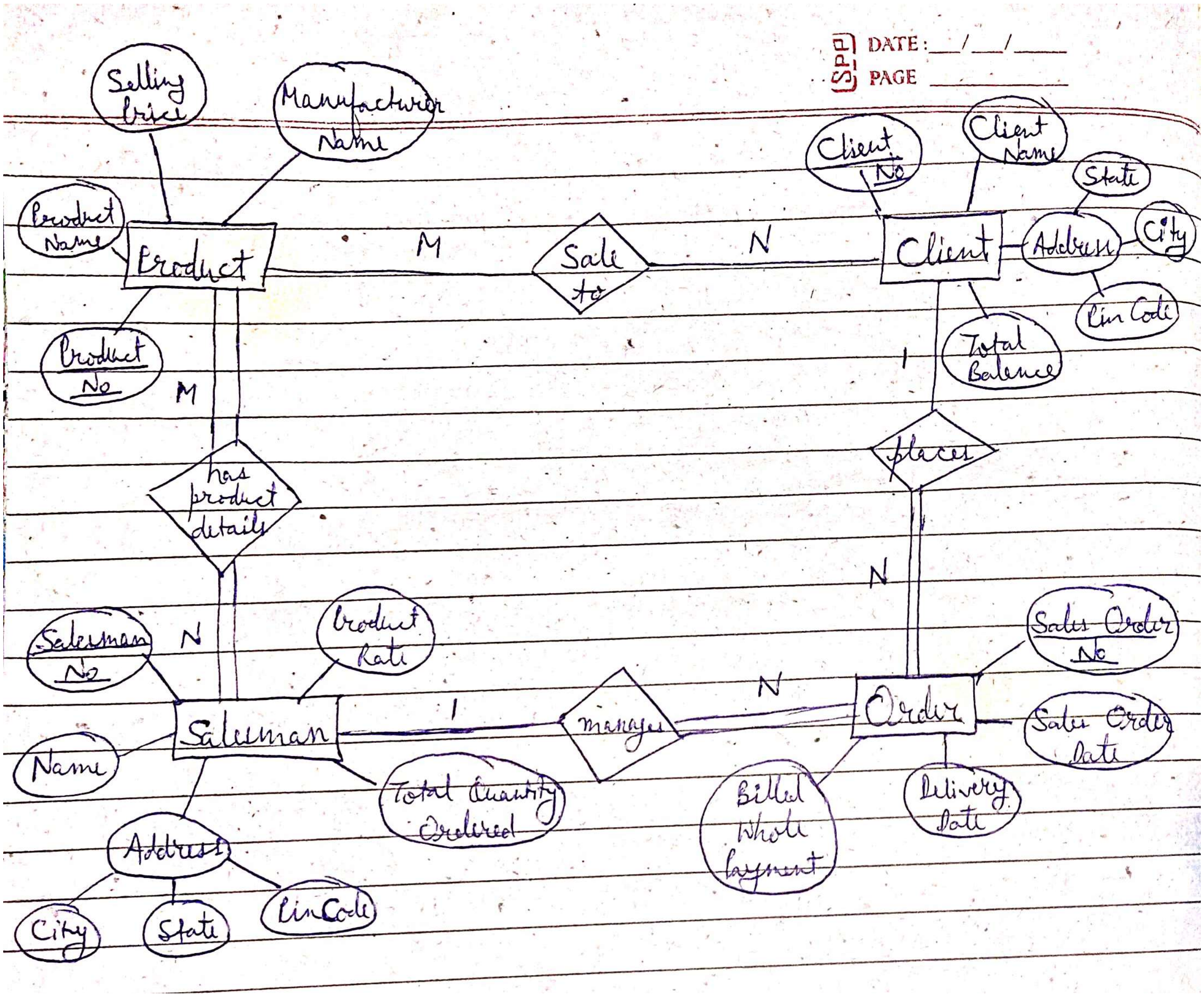
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY
AMITY UNIVERSITY UTTAR PRADESH
NOIDA (U.P)

Open Ended Problem

CASE STUDY - 1

Q.1. A database is being constructed for storing sales information system. A product can be described with a unique product number, product name, selling price, manufacturer name. The product can sale to a particular client and each client have its own unique client number, client name, client addresses, city, pin code, state and total balance to be required to paid. Each client order to buy product from the salesman. In the order, it has unique sales order number, sales order date, client number, salesman number (unique), billed whole payment by the party or not and its delivery date. The salesman has the name, addresses, city, pin code, state, salary of the salesman, delivery date, total quantity ordered, product rate.

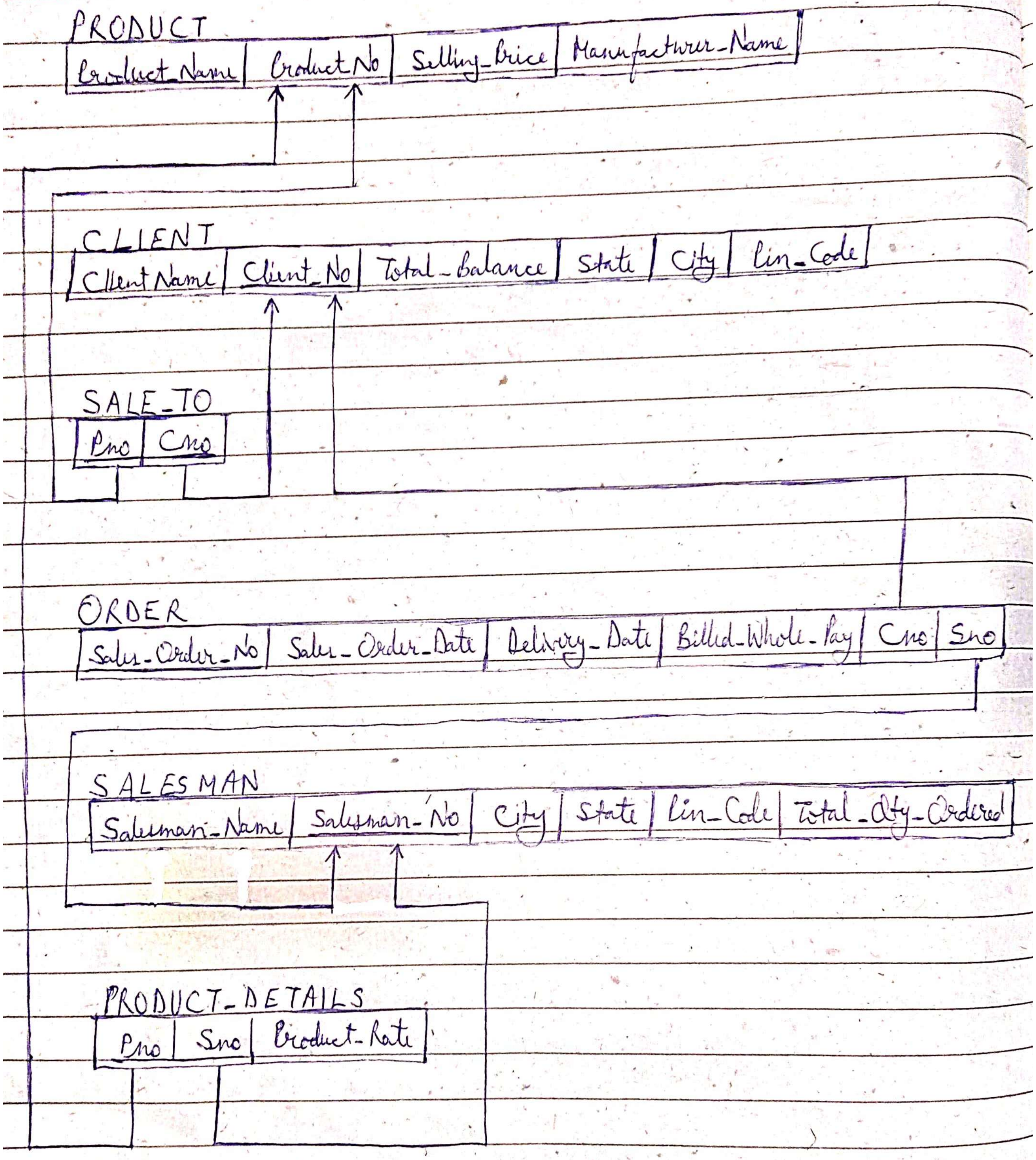
ER Diagram



❖ Assumptions taken for construction of ER Diagram

1. There can be some product which is not sold so there is partial participation from the side of entity ‘Product’ to entity ‘Client’.
2. Some clients may not order any product but are still in client list as they may be regular (or potential) clients who may order later so partial participation.
3. An order can only exist if it has been placed by a client to a salesman, thus total participate from the entity ‘Order’.
4. Some salesmen may not get any order to manage, so Partial Participation.
5. Each salesman would have product details for at least one product thus all salesmen would participate, and each product would have its details with some salesman because only then can they be sold, thus indicating total participation from both sides of the relation (has product details).
6. One product can be sale to many clients and one client can also buy many products thus the relation would have many-to-many cardinality.
7. A client may place more than one order while an order can only be placed by one client thus it would hold one-to-many cardinality from client to order.
8. A salesman may manage many orders, but one order can not be managed by more than one salesman which indicates one-to-many cardinality from salesman to order.
9. One salesman can have product details for one or more products and at the same time one single product can have its details with many salesmen. Thus, the relation between Product and Salesman would have many-to-many cardinality.

Relational Mapping



PRODUCT

Product_Name	Product_No	Selling_Price	Manufacturer_Name

CLIENT

Client Name	Client_No	Total_Balance	State	City	Pin_Code

SALE_TO

Pno	Cno

ORDER

Sales_Order_No	Sales_Order_Date	Delivery_Date	Billed_Whole_Pay	Cno	Sno

SALESMAN

Salesman_Name	Salesman_No	City	State	Pin_Code	Total_Qty_Ordered

PRODUCT_DETAILS

Pno	Sno	Product_Rate

Database Implementation in SQL

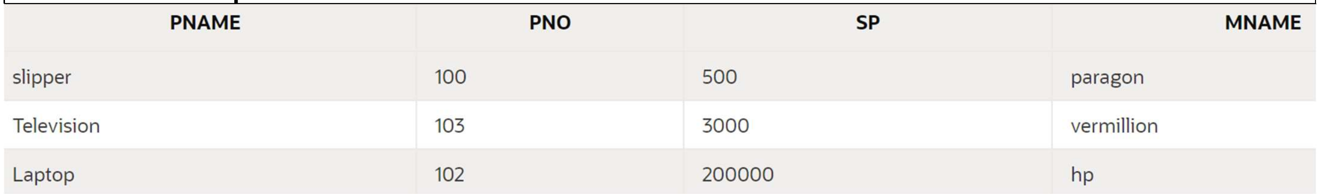
DDL Command Script to create Tables and Insert data to it

```
CREATE TABLE products
(
    pname varchar(20),
    pno number(10) PRIMARY KEY,
    sp real,
    mname varchar(20)
);

CREATE TABLE clients
(
    cno number(10) PRIMARY KEY,
    cname varchar(20),
    total_balance real,
    cstate varchar(20),
    city varchar(20),
    pincode number(10)
);

CREATE TABLE salesman
(
    sno number(10) PRIMARY KEY,
    sname varchar(20),
    city varchar(20),
    sstate varchar(20),
    pincode number(10),
    totalqty real
);

CREATE TABLE orders
(
    orderno number(10) PRIMARY KEY,
    sodate date,
    deldate date,
    billedpay real,
    cno number(10),
    sno number(10),

    CHECK (deldate > sodate),
    FOREIGN KEY (cno) REFERENCES clients(cno),
    FOREIGN KEY (sno) REFERENCES salesman(sno)
);

CREATE TABLE saleto
(
    pno number(10),
    cno number(10),
    FOREIGN KEY (pno) REFERENCES products (pno),
    FOREIGN KEY (cno) REFERENCES clients (cno),

    PRIMARY KEY (pno,cno)
);

CREATE TABLE productdetails
(
    pno number(10),
    sno number(10),
    productrate real,
    FOREIGN KEY (pno) REFERENCES products (pno),
    FOREIGN KEY (sno) REFERENCES salesman (sno),

    PRIMARY KEY (pno,sno)
);

CREATE TABLE orders2
(
    orderno number(10) PRIMARY KEY,
    sodate date,
    deldate date,
    billedpay real,
    cno number(10),
    sno number(10),

    CHECK (deldate > sodate),
    CHECK (sno = productdetails.sno),
    FOREIGN KEY (cno) REFERENCES clients(cno),
    FOREIGN KEY (sno) REFERENCES salesman(sno)
);

CREATE TABLE productdetails2
(
    pno number(10),
    sno number(10),
    productrate real,
    FOREIGN KEY (pno) REFERENCES products (pno),
    FOREIGN KEY (sno) REFERENCES salesman (sno),

    PRIMARY KEY (pno,sno)
);



INSERT INTO products VALUES('slipper',100,500,'paragon');
INSERT INTO products VALUES('Laptop',102,200000,'hp');
INSERT INTO products VALUES('Television',103,3000,'vermillion');

INSERT INTO clients VALUES(200,'jason',50,'kalos','lumious',110011);
INSERT INTO clients VALUES(201,'raph',500,'hoenn','mauville',120022);
INSERT INTO clients values(203,'nathon',60,'alola','hauoli',120011);
INSERT INTO salesman VALUES(300,'merchantA','coal','bituminous',130012,500);
INSERT INTO salesman VALUES(301,'merchantb','capricon','valley',140029,400);


INSERT INTO orders VALUES(401,'04/02/2022','04/20/2022',2000,203,300);
INSERT INTO orders VALUES(400,'04/01/2022','04/28/2022',2000,200,300);
INSERT INTO orders VALUES(402,'04/05/2022','05/02/2022',100,201,301);


INSERT INTO saleto VALUES(100,203);
INSERT INTO saleto VALUES(103,200);
INSERT INTO saleto VALUES(103,201);

INSERT INTO productdetails VALUES(100,300,3000);
INSERT INTO productdetails VALUES(102,300,3420);
INSERT INTO productdetails VALUES(103,301,3400);
INSERT INTO productdetails VALUES(103,300,400);
INSERT INTO productdetails VALUES(102,301,7777);
```

DML Commands

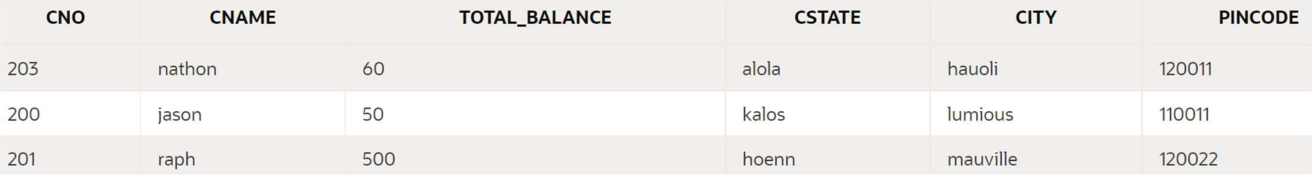
Viewing the Created Tables

```
select * from products
```



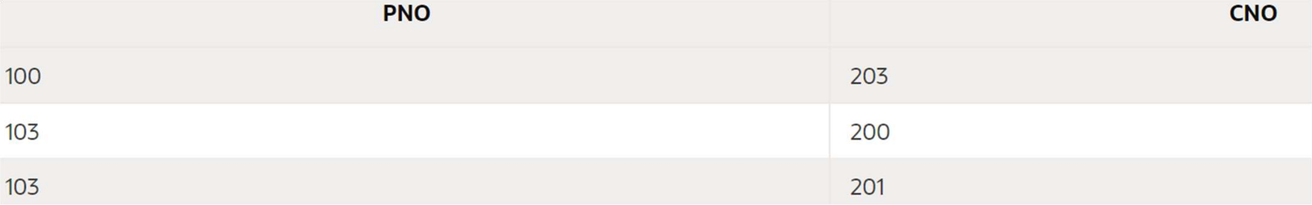
PNAME	PNO	SP	MNAME
slipper	100	500	paragon
Television	103	3000	vermillion
Laptop	102	200000	hp

```
select * from clients
```



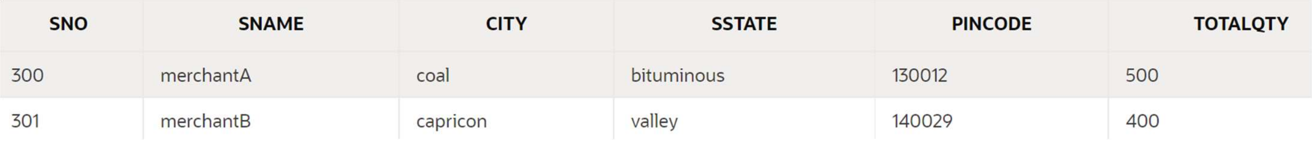
CNO	CNAME	TOTAL_BALANCE	CSTATE	CITY	PINCODE
203	nathon	60	alola	hauoli	120011
200	jason	50	kalos	lumious	110011
201	raph	500	hoenn	mauville	120022

```
select * from saleto
```



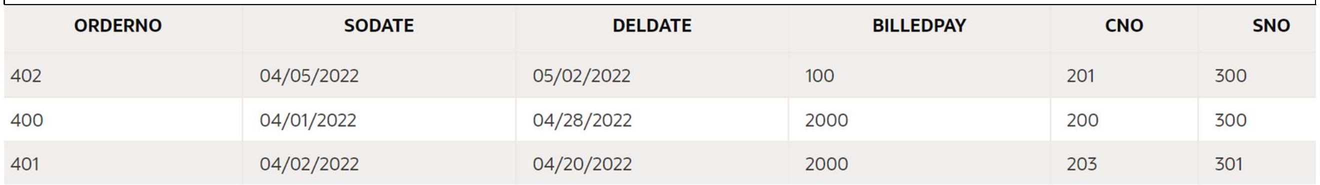
PNO	CNO
100	203
103	200
103	201

```
select * from salesman
```



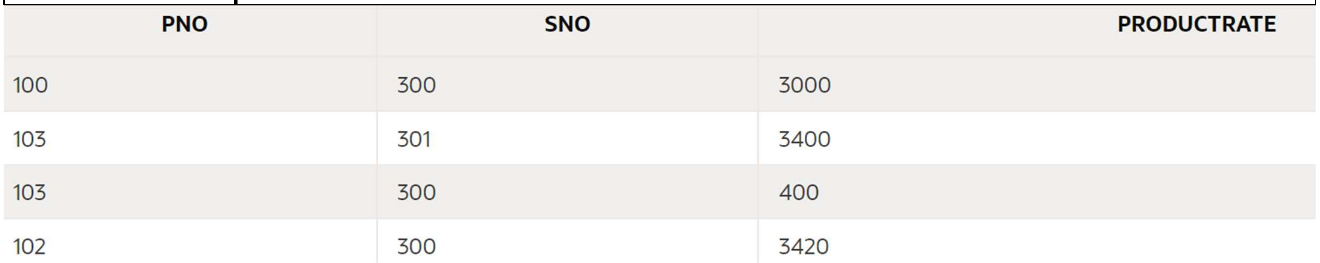
SNO	SNAME	CITY	SSTATE	PINCODE	TOTALQTY
300	merchantA	coal	bituminous	130012	500
301	merchantB	capricon	valley	140029	400

select * from orders



ORDERNO	SODATE	DELDATE	BILLEDPAY	CNO	SNO
402	04/05/2022	05/02/2022	100	201	300
400	04/01/2022	04/28/2022	2000	200	300
401	04/02/2022	04/20/2022	2000	203	301

select * from productdetails



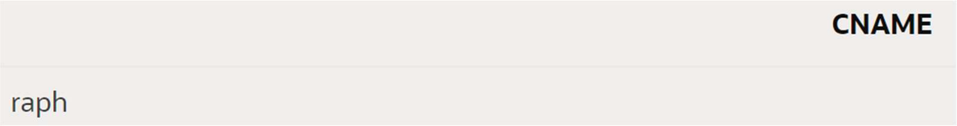
PNO	SNO	PRODUCTRATE
100	300	3000
103	301	3400
103	300	400
102	300	3420

SQL Queries for Validation

1) Display the names of clients who have an ‘p’ as third letter in their name.

```
Select cname
from clients
where cname LIKE '__p%'
```

Output

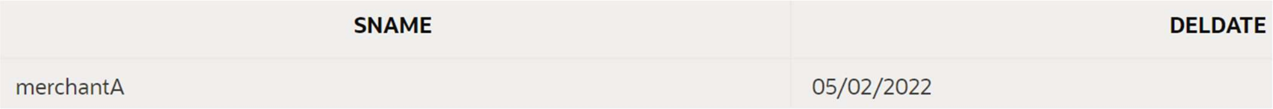


CNAME
raph

2) Display the salesman names and delivery date for those salesmen who got orders with delivery date for month of May.

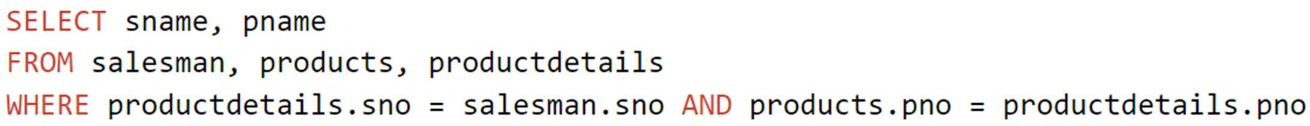
```
SELECT sname,deldate
from salesman,orders
where TO_CHAR(deldate, 'Mon') = 'May' AND salesman.sno = orders.sno
```

Output



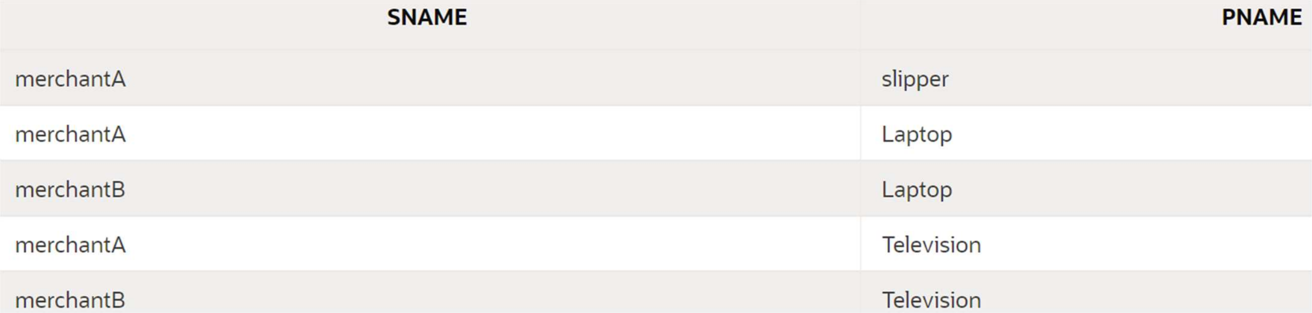
SNAME	DELDATE
merchantA	05/02/2022

3) Display the name of all salesmen along with the product details of the products which they can sell to clients.



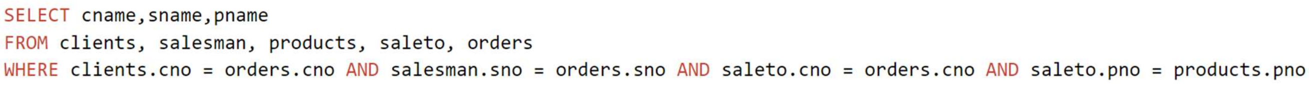
```
SELECT sname, pname
FROM salesman, products, productdetails
WHERE productdetails.sno = salesman.sno AND products.pno = productdetails.pno
```

Output



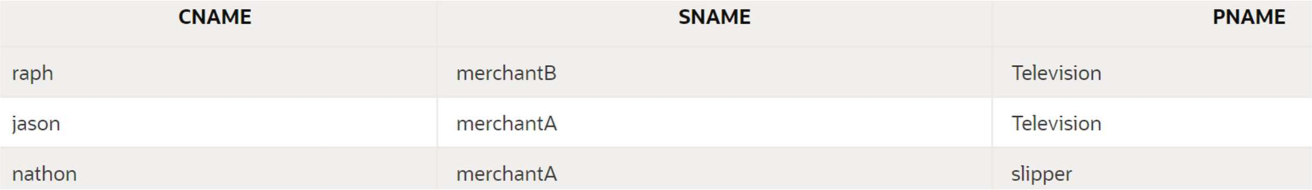
SNAME	PNAME
merchantA	slipper
merchantA	Laptop
merchantB	Laptop
merchantA	Television
merchantB	Television

4) Show the details for the Orders placed with client name, salesman, product name.



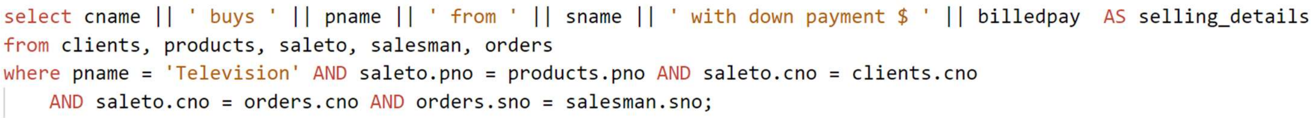
```
SELECT cname,sname,pname
FROM clients, salesman, products, saleto, orders
WHERE clients.cno = orders.cno AND salesman.sno = orders.sno AND saleto.cno = orders.cno AND saleto.pno = products.pno
```

Output



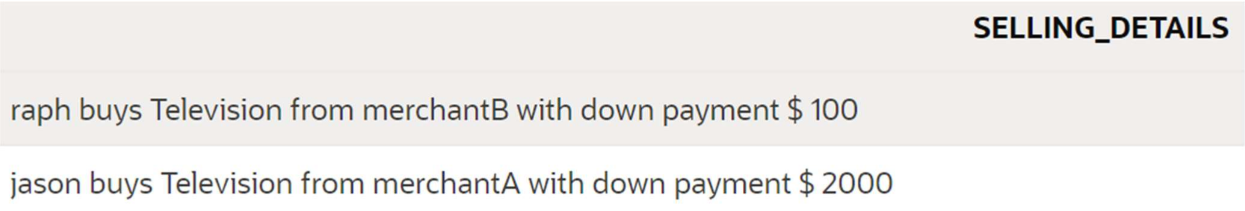
CNAME	SNAME	PNAME
raph	merchantB	Television
jason	merchantA	Television
nathon	merchantA	slipper

5) Display the names of clients who bought 'Television' in the format "jason buys Television from merchantA with down payment $ 2000"



```
select cname || ' buys ' || pname || ' from ' || sname || ' with down payment $ ' || billedpay  AS selling_details
from clients, products, saleto, salesman, orders
where pname = 'Television' AND saleto.pno = products.pno AND saleto.cno = clients.cno
    AND saleto.cno = orders.cno AND orders.sno = salesman.sno;
```

Output



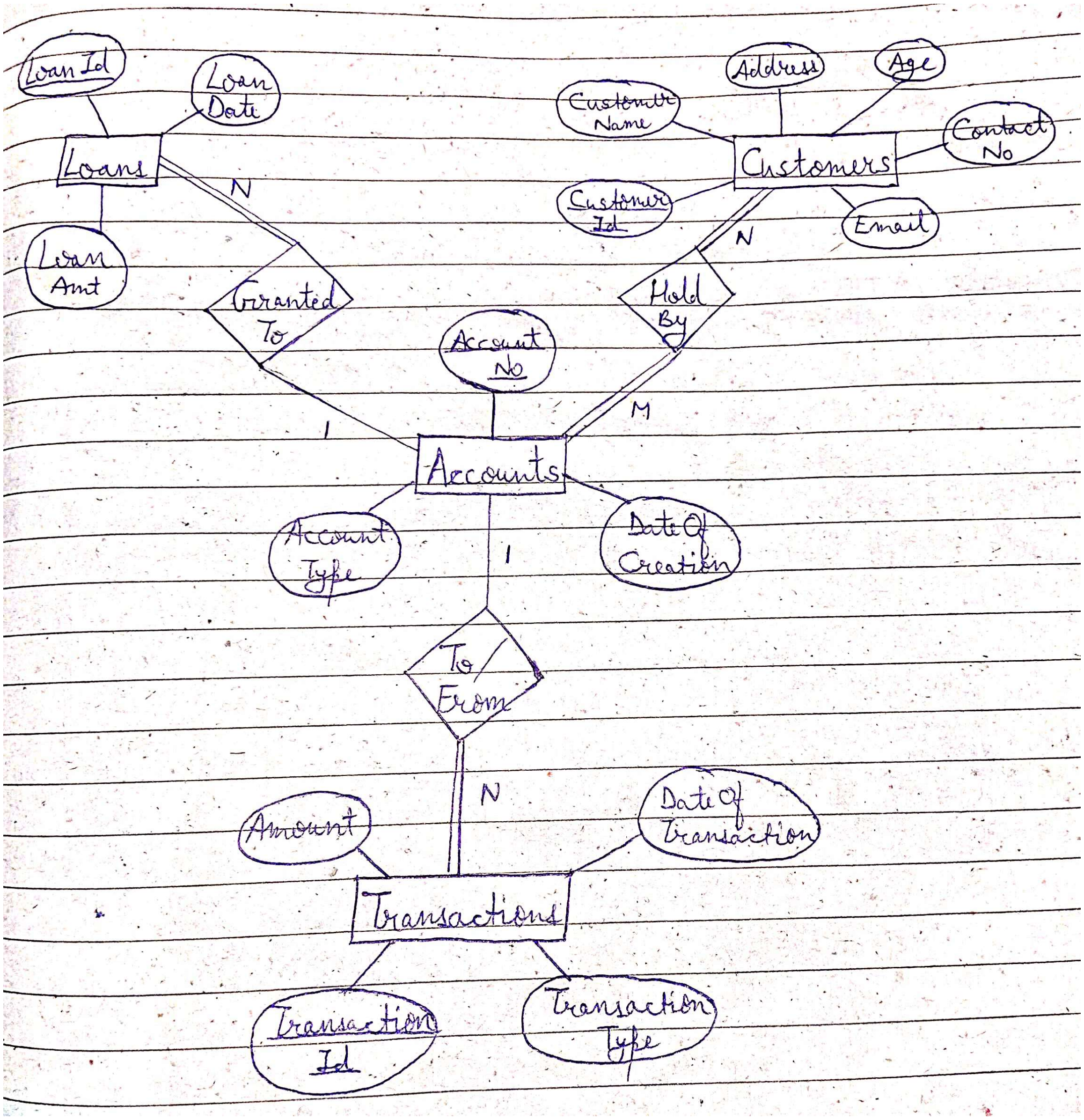
SELLING_DETAILS
raph buys Television from merchantB with down payment $ 100
jason buys Television from merchantA with down payment $ 2000

Open Ended Problem

CASE STUDY - 2

Q.2. A bank database keeps record of the details of customers, accounts, loans and transactions such as deposits or withdraws. Customer record should include customer id, customer name, address, age, contact number, email id etc., accounts details involve account number, account type (fixed account, savings account, monthly account etc), date of creation of the account. Transaction detail keeps information about amount deposited or withdrawn to/from a particular account and the date of transaction. The database should also store record of loans which include loan amount, loan date and the account number to which the loan is granted.

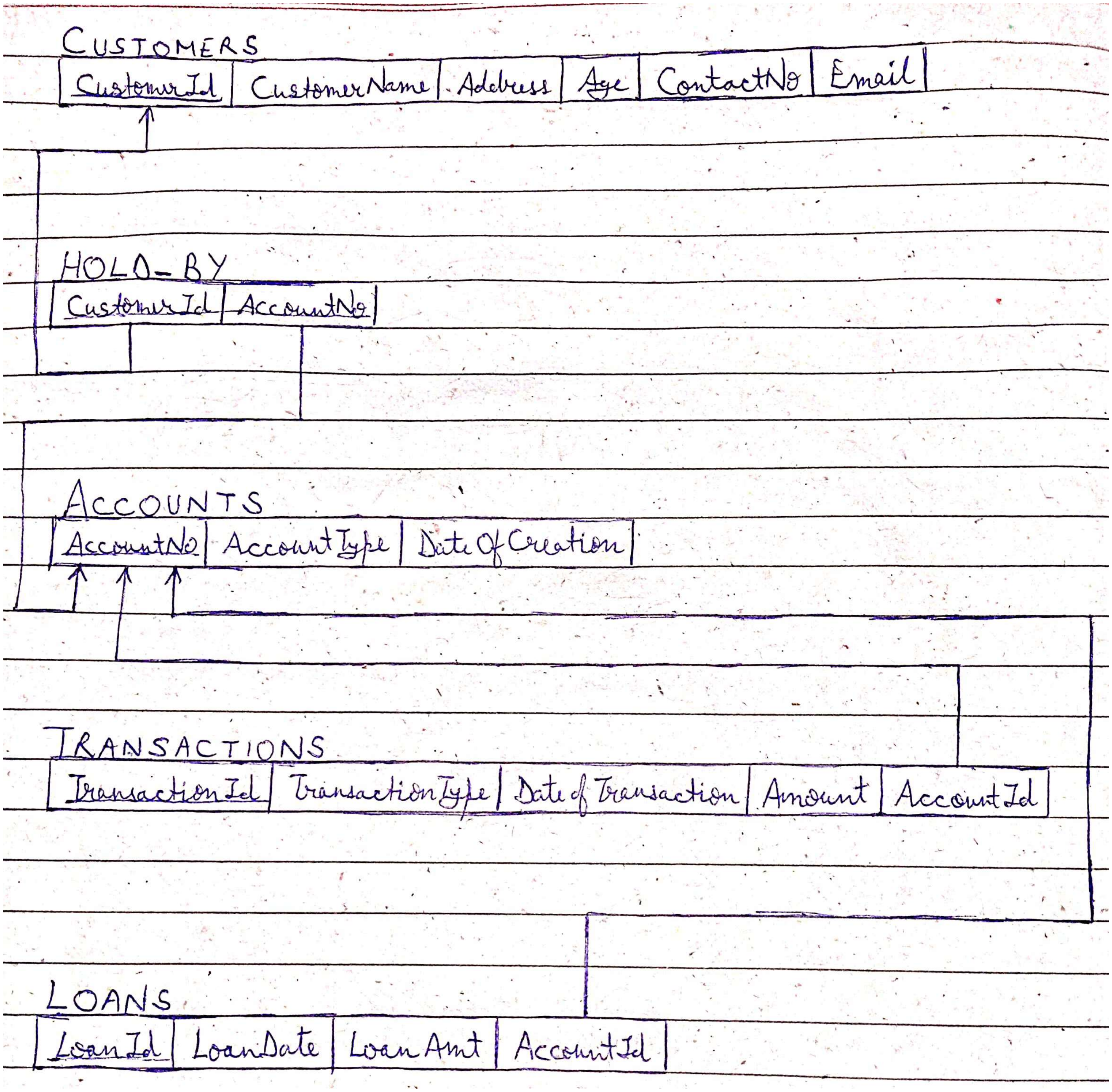
ER Diagram



- ❖ Assumptions taken for construction of ER Diagram

1. The entity “Loans” has a unique attribute ‘LoanId’ which is the Primary key attribute for the ER diagram.
2. The entity “Transactions” is assumed to have unique attribute ‘TransactionId’ as Primary Key for the ER diagram.
3. For the relation “Granted to”:
    a. An account holder may have multiple loans granted to him/her but each loan will have only one borrower which designates the cardinality one to many where 1 is at Accounts side and N at Loans.
    b. There will be no loan which is recorded but not have granted to an account but there can be accounts without any loans, thus, there will be total participation from entity ‘Loans’ but partial participation from entity ‘Accounts’.
4. For the relation “Hold by”:
    a. There will be many-to-many cardinality assigned to this relation because there can be more than one customer holding a single account in case of a Joint Account, as well as it is possible that one customer may have more than 1 account.
    b. All accounts will have some user assigned to it since otherwise the account can’t exist, meanwhile, each and every customer will be holding at least one account to be considered a customer. Thus, this shows that there is total participation at both ends of the relationship to the entities.
5. For relationship “To/From”:
    a. Cardinality is one-to-many with one ‘Accounts’ side while many at ‘Transactions’ side because a single transaction will either be of depositing to one account (or withdrawing from one account) but there is possibility that multiple transactions can be made for a single account.
    b. Participation is total on ‘Transactions’ side since there can be no transactions which is not mapped with an account for success of deposition or withdrawal.
    c. On the side of entity ‘Accounts’, there is partial participation because there can an account without any transactions being made *considering that depositing of security and minimum balance at the time of account creation is not considered as a transaction.*

Relational Mapping



Database Implementation in SQL

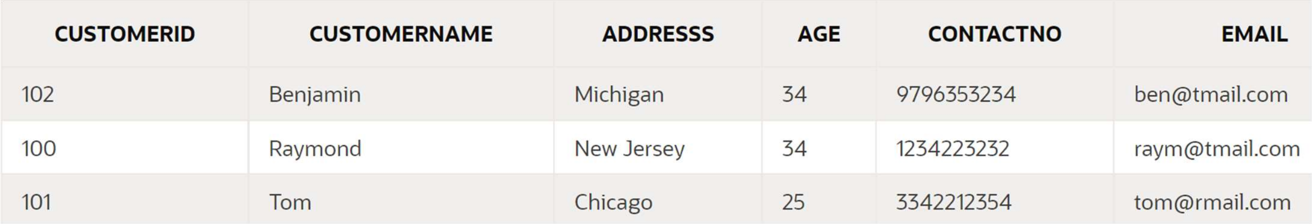
DDL Command Script to create Tables and Insert data to them

```
CREATE TABLE Customers
(
  CustomerId int,
  CustomerName varchar(20),
  Addresss varchar(25),
  Age int,
  ContactNo number(10),
  Email varchar(20),
  PRIMARY KEY (CustomerId)
);

CREATE TABLE Accounts
(
  AccountNo int,
  AccountType varchar(10),
  DateOfCreation date,

  PRIMARY KEY (AccountNo)
);

CREATE TABLE Hold_By
(
  AccountNo int,
  CustomerId int,

  PRIMARY KEY (AccountNo, CustomerId),
  FOREIGN KEY (AccountNo) REFERENCES Accounts(AccountNo),
  FOREIGN KEY (CustomerId) REFERENCES Customers(CustomerId)
);

CREATE TABLE Transactions
(
  TransactionNo int,
  TransactionType varchar(10),
  Amount float,
  DateOfTransaction date,
  AccountNo int,

  PRIMARY KEY (TransactionNo),
  FOREIGN KEY (AccountNo) REFERENCES Accounts(AccountNo)
);

CREATE TABLE Loans
(
  LoanId int,
  LoanAmt float,
  LoanDate date,
  AccountNo int,

  PRIMARY KEY (LoanId),
  FOREIGN KEY (AccountNo) REFERENCES Accounts(AccountNo)
);


INSERT INTO customers VALUES(100,'Raymond','New Jersey',34,1234223232,'raym@tmail.com')
INSERT INTO customers VALUES(101,'Tom','Chicago',25,3342212354,'tom@rmail.com')
INSERT INTO customers VALUES(102,'Benjamin','Michigan',34,9796353234,'ben@tmail.com')

INSERT INTO accounts VALUES(200444,'Fixed','02/22/2022')
INSERT INTO accounts VALUES(200555,'Savings','03/10/2022')
INSERT INTO accounts VALUES(200999,'Monthly','04/01/2022')

INSERT INTO hold_By VALUES(200444,100)
INSERT INTO hold_By VALUES(200555,101)
INSERT INTO hold_By VALUES(200999,102)

INSERT INTO transactions VALUES(300010,'Deposit',9000,'02/23/2022',200444)
INSERT INTO transactions VALUES(301010,'Deposit',4000,'03/25/2022',200555)
INSERT INTO transactions VALUES(302020,'Withdrawal',200,'04/05/2022',200999)

INSERT INTO loans VALUES(444100,35000,'04/02/2022',200555)
```

DML Commands

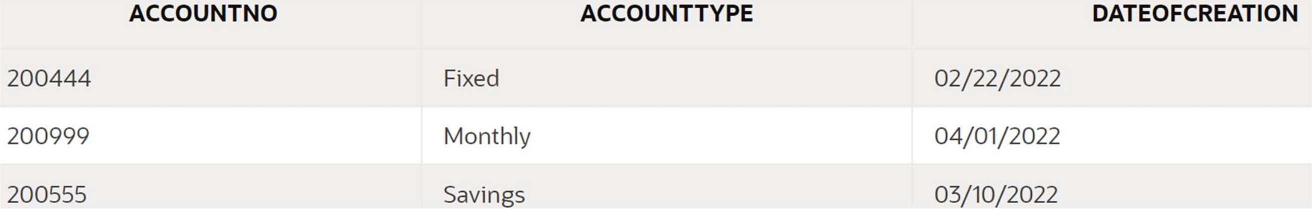
Viewing the Created Tables

```
SELECT * FROM customers;
```



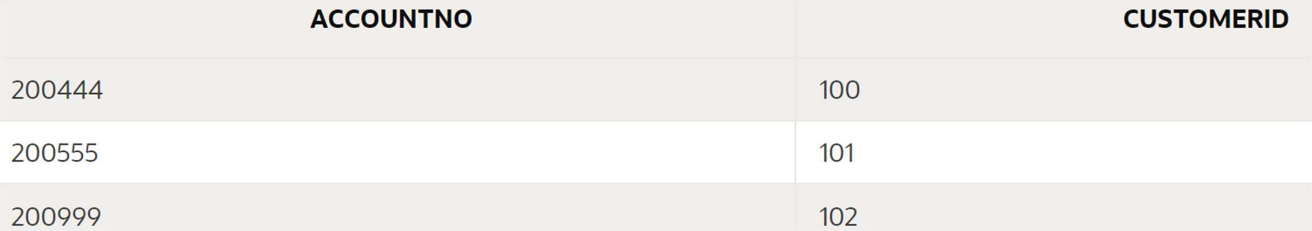
CUSTOMERID	CUSTOMERNAME	ADDRESSS	AGE	CONTACTNO	EMAIL
102	Benjamin	Michigan	34	9796353234	ben@tmail.com
100	Raymond	New Jersey	34	1234223232	raym@tmail.com
101	Tom	Chicago	25	3342212354	tom@rmail.com

```
SELECT * FROM accounts;
```



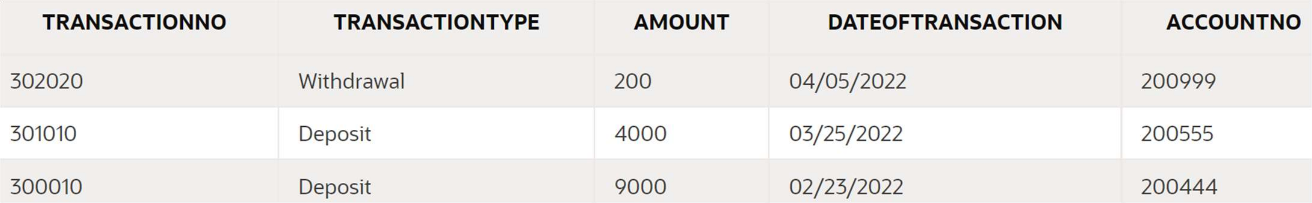
ACCOUNTNO	ACCOUNTTYPE	DATEOFCREATION
200444	Fixed	02/22/2022
200999	Monthly	04/01/2022
200555	Savings	03/10/2022

```
SELECT * FROM hold_by;
```



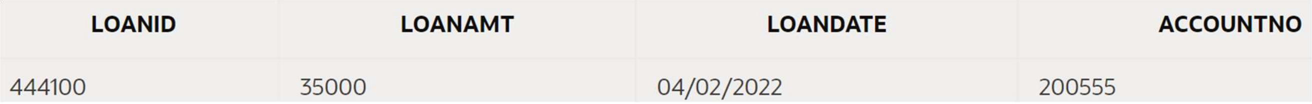
ACCOUNTNO	CUSTOMERID
200444	100
200555	101
200999	102

```
SELECT * FROM transactions;
```



TRANSACTIONNO	TRANSACTIONTYPE	AMOUNT	DATEOFTRANSACTION	ACCOUNTNO
302020	Withdrawal	200	04/05/2022	200999
301010	Deposit	4000	03/25/2022	200555
300010	Deposit	9000	02/23/2022	200444

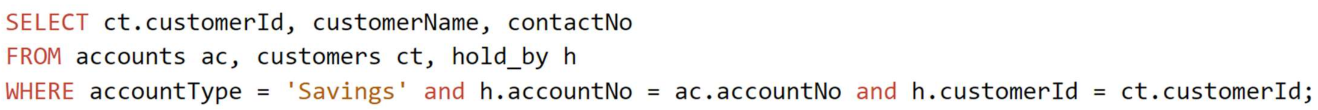
```
SELECT * FROM loans;
```



LOANID	LOANAMT	LOANDATE	ACCOUNTNO
444100	35000	04/02/2022	200555

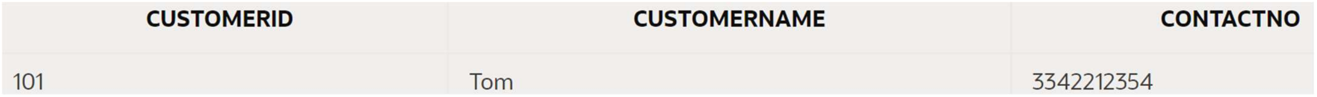
SQL Queries for Validation

1) Display the customer ID, customer names and their contact numbers who have ‘Savings’ Account.



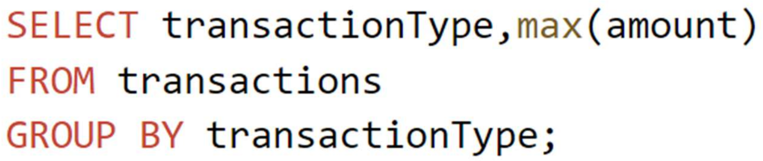
```
SELECT ct.customerId, customerName, contactNo
FROM accounts ac, customers ct, hold_by h
WHERE accountType = 'Savings' and h.accountNo = ac.accountNo and h.customerId = ct.customerId;
```

Output



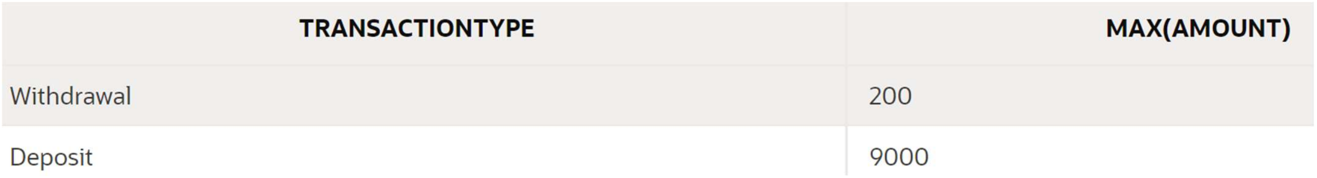
CUSTOMERID	CUSTOMERNAME	CONTACTNO
101	Tom	3342212354

2) Display the maximum amount transacted (for both deposited and withdrawal).



```
SELECT transactionType,max(amount)
FROM transactions
GROUP BY transactionType;
```

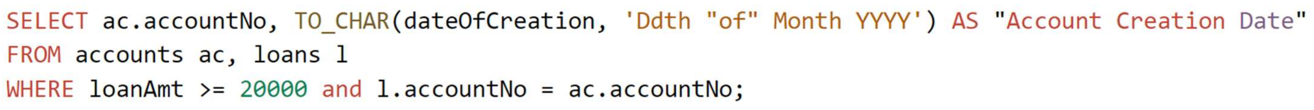
Output



TRANSACTIONTYPE	MAX(AMOUNT)
Withdrawal	200
Deposit	9000

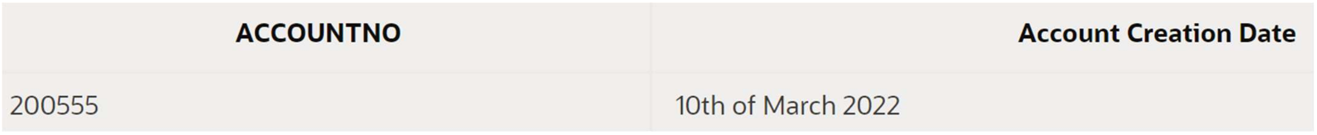
3) Show the accountNo and account creation date for the accounts granted with a loan amount of 20000 or more.

(Date of Creation should be in format – ‘10th of March 2022’)



```
SELECT ac.accountNo, TO_CHAR(dateOfCreation, 'Ddth "of" Month YYYY') AS "Account Creation Date"
FROM accounts ac, loans l
WHERE loanAmt >= 20000 and l.accountNo = ac.accountNo;
```

Output



ACCOUNTNO	Account Creation Date
200555	10th of March 2022