Part 1 - Algebra Queries:

Write relational algebra expressions that will produce a relation containing:

• Q1: Loan number with value over $1000.

- $\Pi_{\text{Loan\_number}}(\sigma \text{Loan\_amount} > 1000(\text{loan}))$

• Q2: Customers' name and email with the amount of their loan (the amount of loan should be NULL if a customer does not have any loan)

- $\Pi_{\text{Name, Email, Loan\_amount}}(\text{Customer} \bowtie_{\text{Customer\_id = Loan.Customer\_id}} \text{loan})$

• Q3: Retrieve the number of transactions per each account.

- $_{\text{Account\_number}}\ \text{COUNT}_{\text{Transaction\_id}}\ (\text{account} \bowtie_{\text{Account\_number = depositor.Account\_number}} \text{depositor})$

• Q4: Retrieve all the customers having their account in "active" state.

- $\Pi_{\text{Name,Customer\_id,Gender,Birth\_date,City,Address,Postal\_code,Home\_phone,Mobile\_phone,email}}$ $(\sigma \text{Status='Active'}(\text{customer} \bowtie_{\text{Customer\_id = Account.Customer\_id}} \text{account})$

Part 2 - SQL Queries:

Write a SQL command for the following:

• Q1: Retrieve the customers who are living in "Trondheim" (Returns 5 records)

-
  ```
  SELECT * FROM `customer` WHERE `City` IN ('Trondheim');
  ```

• Q2: Retrieve the customers who have their email address under the commercial internet domain (.com) (Returns 5 records)

-
  ```
  SELECT * FROM `customer` WHERE `Email` LIKE '%.com%';
  ```

• Q3: Retrieve the information of loans given to the customers in each branch between 2019-06-01 and 2020-06-01. (Returns 4 records)

```
SELECT * FROM `loan` WHERE `Starting_Date` BETWEEN '2019-06-01' AND '2020-
06-01';
```

• Q4: Retrieve the youngest customer who has taken a loan. (Returns 1 record)

```
SELECT customer.* FROM `customer` INNER JOIN loan ON customer.Customer_id =
 loan.Customer_id ORDER BY `Birth_date`DESC LIMIT 1;
```

• Q5: Write a SQL query that retrieves customers without any loans. (Returns 4 records)

```
SELECT customer.* FROM `customer` LEFT JOIN loan ON customer.Customer_id =
loan.Customer_id WHERE loan.Loan_number IS NULL;
```

• Q6: Retrieve the number of transactions for each account during the year 2019 (Returns 8 records)

```
SELECT account.Account_number, COUNT(DISTINCT depositor.Transaction_id)AS T
ransactions FROM `account` INNER JOIN depositor ON account.Account_number=d
epositor.Account_number WHERE depositor.Date < '2020-01-
01' GROUP BY account.Account_number;
```

• Q7: Add a new customer with information below then open an inactive account in the given branch:

o Name: Ryan Ishus o Address o City : Trondheim o Street: Bakkegata o No: 15 o Postal_code: 7049 o Home_Phone : 75432103 o Mobile_phone: 45464783 o Email : ryan00@realmail.no o Customer_id: 10016 o Gender: Male o Birth_date: 1991-01-10 o Branch: b2 o Account_number=ac1001 o Balance=$1000 o Opening_date= 2021-01-18 o Status= Inactive

```
INSERT INTO customer(`Name`,`Customer_id`,`Gender`,`Birth_date`,`City`,`Address`,`Postal_code`,`Home_Phone`,`Mobile_phone`,`Email`) VALUES ('Ryan Ishus', '10016', 'M', '1991-01-10', 'Trondheim', 'Bakkegata 15', '7049', '75432103', '45464783', 'ryan00@realmail.no');
```

```
INSERT INTO account(`Account_number`, `Customer_id`, `Branch_code`, `Balance`, `Opening_date`, `Status`) VALUES('ac1001','10016','b2','1000','2021-01-18','Inactive');
```

• Q8: Update the “Status” of account of customer Ryan Ishus to “Active”.

```
UPDATE account INNER JOIN customer ON account.Customer_id = customer.Customer_id SET account.Status = 'Active' WHERE customer.Name = 'Ryan Ishus';
```

• Q9: Delete the loans which their loan period is NULL.

```
DELETE FROM loan WHERE `Loan_period`IS NULL;
```