# Customer Managment

Team names

- *Osama Hatem*
- *Mohammed Taha*
- *Abdelrahman Magdi*
- *Abdelrahman Hmada*
- *Moaz Radwan*
- *Alaa El-said*

> week 1 work

- Database Design: Design a SQL database schema to manage customer data, including tables for customer information, transactions, and interactions.
- Implementation: Create and populate the SQL database using Microsoft SQL Server.
- SQL Queries: Write SQL queries to extract, update, and analyze customer data.

> > SQL SERVER Database Creation

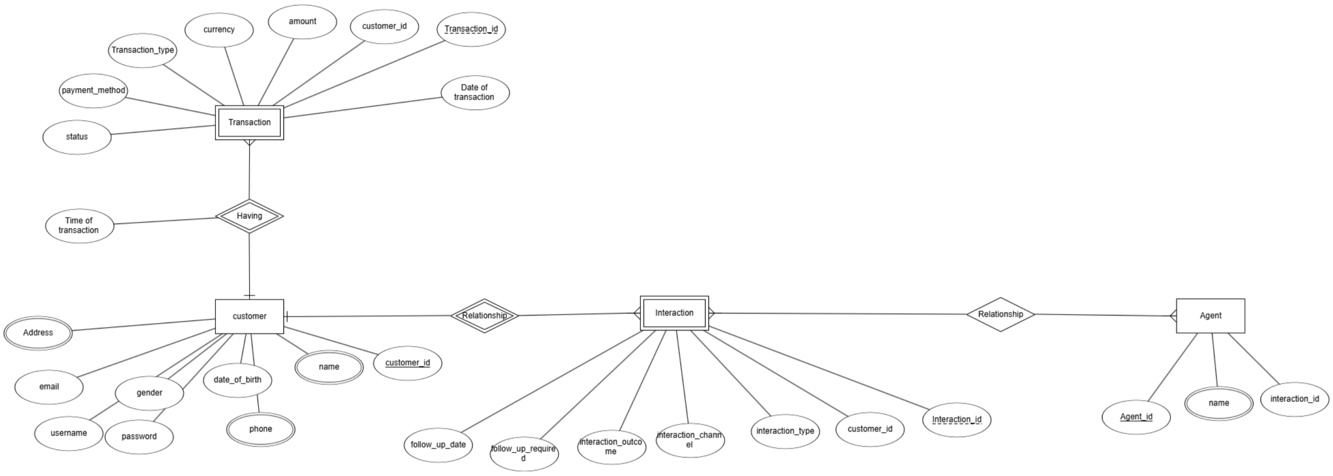
```
create database Customer_Management_sys
--Table creation stage
CREATE TABLE customer
(
  customer_id INT NOT NULL,
  date_of_birth DATE NOT NULL,
  gender CHAR(1) NOT NULL,
  email VARCHAR(30) NOT NULL,
  username VARCHAR(20) NOT NULL,
  Account_password VARCHAR(30) NOT NULL,
  FName VARCHAR(20) NOT NULL,
  LName VARCHAR(20) NOT NULL,
  PRIMARY KEY (customer_id)
);

CREATE TABLE TheTransaction
(
  Transaction_id INT NOT NULL,
  amount INT NOT NULL,
  currency VARCHAR(3) NOT NULL,
  Transaction_type VARCHAR(20) NOT NULL,
  payment_method VARCHAR(20) NOT NULL,
  payment_status VARCHAR(20) NOT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (Transaction_id),
  FOREIGN KEY (customer_id) REFERENCES customer(customer_id)
);
```

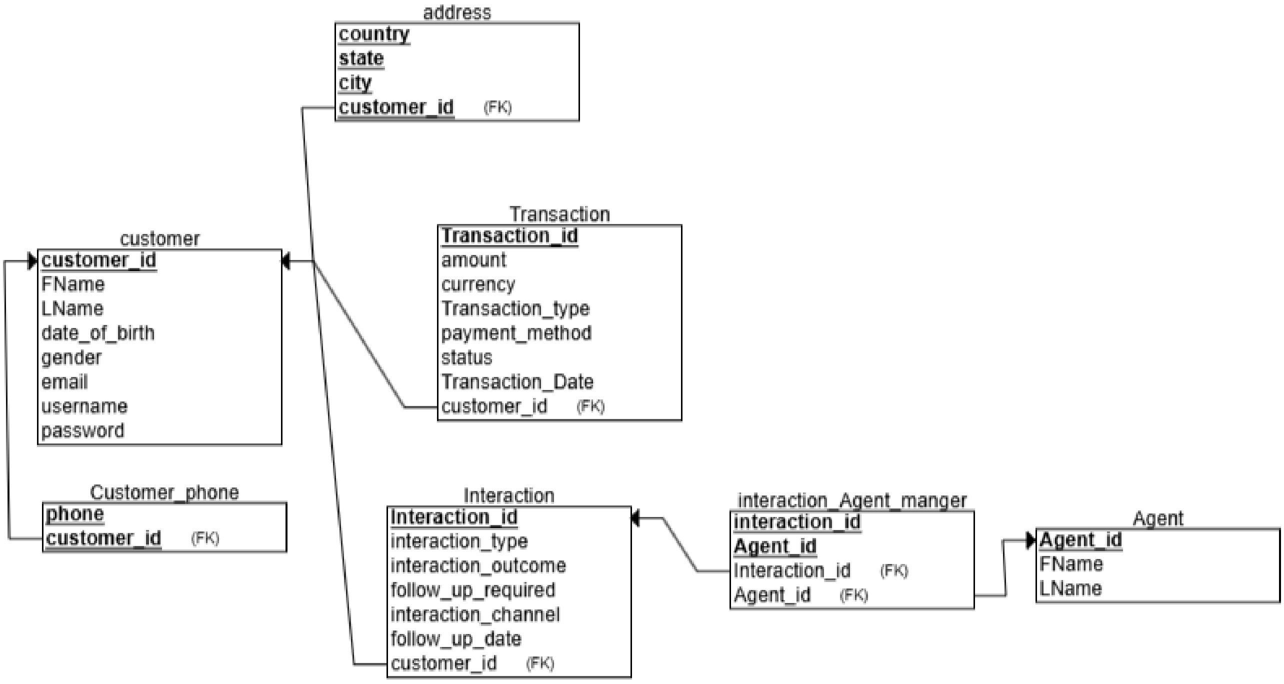
```
CREATE TABLE Interaction
(
  Interaction_id INT NOT NULL,
  interaction_type VARCHAR(20) NOT NULL,
  interaction_outcome VARCHAR(20) NOT NULL,
  follow_up_required CHAR(3) NOT NULL,
  interaction_channel VARCHAR(20) NOT NULL,
  follow_up_date DATE NOT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (Interaction_id),
  FOREIGN KEY (customer_id) REFERENCES customer(customer_id)
);

CREATE TABLE Customer_phone
(
  phone INT NOT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (phone, customer_id),
  FOREIGN KEY (customer_id) REFERENCES customer(customer_id)
);

CREATE TABLE Agent
(
  Agent_id INT NOT NULL,
  FName VARCHAR(20) NOT NULL,
  LName VARCHAR(20) NOT NULL,
  PRIMARY KEY (Agent_id)
);

CREATE TABLE The_address
(
  country VARCHAR(20) NOT NULL,
  government VARCHAR(20) NOT NULL,
  city VARCHAR(20) NOT NULL,
  customer_id INT NOT NULL,
  PRIMARY KEY (country, government, city, customer_id),
  FOREIGN KEY (customer_id) REFERENCES customer(customer_id)
);

CREATE TABLE interaction_Agent_manger
(
  interaction_id INT NOT NULL,
  Agent_id INT NOT NULL,
  PRIMARY KEY (interaction_id, Agent_id),
  FOREIGN KEY (Interaction_id) REFERENCES Interaction(Interaction_id),
  FOREIGN KEY (Agent_id) REFERENCES Agent(Agent_id)
);
```

> > Python script for the generative datasets

```python
from faker import Faker
import random

# Initialize Faker
fake = Faker()

# Function to get a country name without commas
def get_country_without_comma():
    country = fake.country()
    while ',' in country:
        country = fake.country()
    return country

# Function to escape single quotes in SQL strings
def escape_single_quotes(value):
    return value.replace("'", "''")

# File to write SQL insert statements
file_path = 'sql_insert_data.txt'

# Start writing to the file
with open(file_path, 'w') as file:

    # Customer table insert statements
    file.write("-- Customer table insert statements\n")
    for i in range(1, 11001):
        dob = fake.date_of_birth(minimum_age=18, maximum_age=80)
        gender = random.choice(['M', 'F'])
        email = escape_single_quotes(fake.email()[:30])  # Limit to 30 characters, escape single quotes
        username = escape_single_quotes(fake.user_name()[:20])  # Limit to 20 characters, escape single quotes
        password = escape_single_quotes(fake.password(length=10)[:30])  # Limit to 30 characters, escape single quotes
        fname = escape_single_quotes(fake.first_name()[:20])  # Limit to 20 characters, escape single quotes
        lname = escape_single_quotes(fake.last_name()[:20])  # Limit to 20 characters, escape single quotes
        file.write(f"INSERT INTO customer VALUES ({i}, '{dob}', '{gender}', '{email}', '{username}', '{password}', '{fname}', '{lname}');\n")

    # TheTransaction table insert statements
    file.write("\n-- TheTransaction table insert statements\n")
    for i in range(1, 11001):
        amount = round(random.uniform(10, 5000), 2)
        currency = random.choice(['USD', 'EUR', 'GBP'])
        transaction_type = random.choice(['Purchase', 'Refund'])
        payment_method = random.choice(['Credit Card', 'Debit Card', 'PayPal'])
        payment_status = random.choice(['Completed', 'Refunded', 'Pending'])
        customer_id = random.randint(1, 11000)
        file.write(f"INSERT INTO TheTransaction VALUES ({i}, {amount}, '{currency}', '{transaction_type}', '{payment_method}', '{payment_status}', {customer_id});\n")
    # Interaction table insert statements
    file.write("\n-- Interaction table insert statements\n")
    for i in range(1, 11001):
        interaction_type = random.choice(['Complaint', 'Feedback', 'Support Request'])
        interaction_outcome = random.choice(['Resolved', 'Implemented', 'Unresolved', 'Pending'])
        follow_up_required = random.choice(['Yes', 'No'])
        interaction_channel = random.choice(['Email', 'Phone', 'Chat'])
        follow_up_date = fake.date_between(start_date='today', end_date='+30d')
        customer_id = random.randint(1, 11000)
        file.write(f"INSERT INTO Interaction VALUES ({i}, '{interaction_type}', '{interaction_outcome}', '{follow_up_required}', '{interaction_channel}', '{follow_up_date}', {customer_id});\n")

    # Customer_phone table insert statements
    file.write("\n-- Customer_phone table insert statements\n")
    for i in range(1, 11001):
        phone = fake.random_number(digits=10)  # Random 10-digit phone number
        customer_id = random.randint(1, 11000)
        file.write(f"INSERT INTO Customer_phone VALUES ({phone}, {customer_id});\n")

    # Agent table insert statements
    file.write("\n-- Agent table insert statements\n")
    for i in range(1, 101):  # Assuming 100 unique agents
        fname = escape_single_quotes(fake.first_name()[:20])  # Escape single quotes
        lname = escape_single_quotes(fake.last_name()[:20])  # Escape single quotes
        file.write(f"INSERT INTO Agent VALUES ({i}, '{fname}', '{lname}');\n")

    # The_address table insert statements (updated table name)
    file.write("\n-- The_address table insert statements\n")
    for i in range(1, 11001):
        country = escape_single_quotes(get_country_without_comma()[:20])  # Escape single quotes, ensure no commas
        government = escape_single_quotes(fake.state()[:20])  # Escape single quotes
        city = escape_single_quotes(fake.city()[:20])  # Escape single quotes
        customer_id = random.randint(1, 11000)
        file.write(f"INSERT INTO The_address VALUES ('{country}', '{government}', '{city}', {customer_id});\n")

    # Interaction_Agent_Manager table insert statements
    file.write("\n-- Interaction_Agent_Manager table insert statements\n")
    for i in range(1, 11001):
        interaction_id = random.randint(1, 11000)
        agent_id = random.randint(1, 100)  # Referencing from the agents
        file.write(f"INSERT INTO interaction_Agent_manger VALUES ({interaction_id}, {agent_id});\n")

print(f"SQL insert statements generated in {file_path}")
```

ERD and Mapping Design




address
country
state
city
customer_id (FK)

Transaction
Transaction_id
amount
currency
Transaction_type
payment_method
status
Transaction_Date
customer_id (FK)

customer
customer_id
FName
LName
date_of_birth
gender
email
username
password

Customer_phone
phone
customer_id (FK)

Interaction
Interaction_id
interaction_type
interaction_outcome
follow_up_required
interaction_channel
follow_up_date
customer_id (FK)

interaction_Agent_manger
interaction_id
Agent_id
Interaction_id (FK)
Agent_id (FK)

Agent
Agent_id
FName
LName

SQL Analysis

Customer Analysis

```
--proc to see how maney customers in each country
create procedure CustomersInCountries
AS
select country , count(customer_id) as customers_in_countris
from The_address
Group by country
order by country


exec CustomersInCountries
```

```
--proc to see who of the customers have multiaddress and by how much
create procedure CustomersHaveMultiAddress
As
select c.customer_id ,c.FName +' '+ c.LName as fullName , count(TA.city) as
NUMofPLACESinCity
from customer as c join The_address as TA ON c.customer_id = TA.customer_id
group by c.customer_id,c.FName,c.LName
order by NUMofPLACESinCity  desc

exec CustomersHaveMultiAddress


-- proc to see who maney customers have multiphones and by how maney
create procedure CustomersHaveMultiPhone
AS
select c.customer_id ,c.FName +' '+ c.LName as fullName , count(p.phone) as
NUMofPhones
from customer as c join Customer_phone as p ON c.customer_id = p.customer_id
group by c.customer_id,c.FName,c.LName
order by NUMofPhones desc

exec CustomersHaveMultiPhone


--number of customers in each year
create procedure CUSTOMERSBIRTHDAY
AS
select Year(c.date_of_birth) as year_of_birth ,count(c.customer_id) as
CustomersInYear
from customer as c
group by year(c.date_of_birth)
order by CustomersInYear desc

exec CUSTOMERSBIRTHDAY

--how maney male and female
create procedure GENDERcounter
AS
select c.gender , count(c.customer_id) as NUMof_each_Gender
from customer as c
group by c.gender
order by NUMof_each_Gender desc

exec GENDERcounter
```