Task 1

Please analyse the conceptual data model of "Services" (Fig. 1.), which is incomplete, but the classes and the relationships between them may represent a part of the reality under consideration. ‘Services’ are understood directly

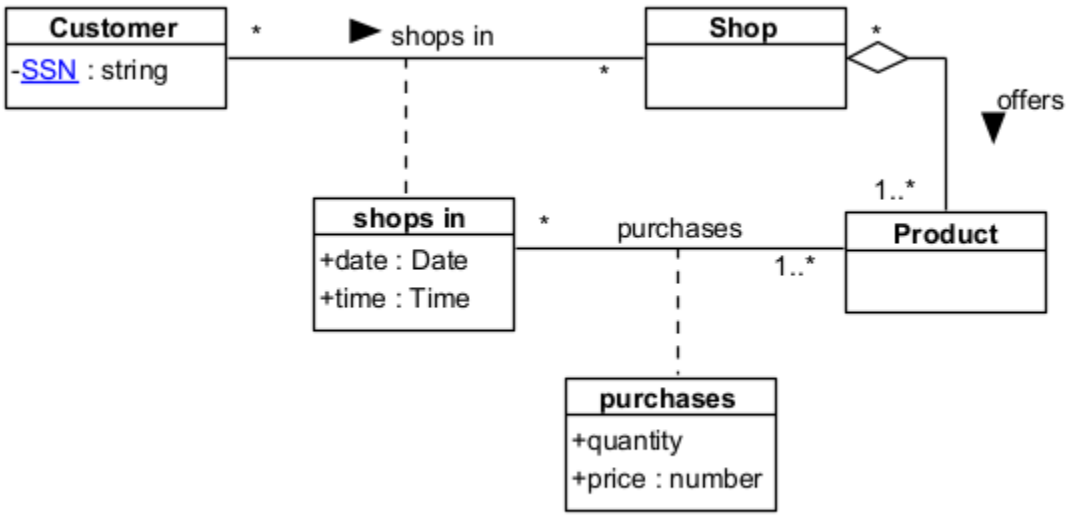


Fig. 1. Conceptual data model for “Services”

Then, complete the following tasks:

1. Verify the data model in the context of a given set of rules and domain constraints. Further, modify the set of rules and constraints (by supplementing their definition) or correct the conceptual data model (justify!) to obtain a valid model.

   Domain rules and constraints – preliminary list:

   R01 – Any customer can shop in many stores
   R02 – A customer can shop repeatedly in the same store
   R03 - Multiple customers can shop at the store
   R04 – Each purchase is made by the customer in the store on a specific day and time
   R05 – A store must offer at least one product
   R06 – The same product (type) can be offered by multiple stores

R07 – Each store can individually propose the price and quantity of the offered product

2. Provide a revised and complete version of the data model (complete UML class diagram)
3. Create a logical/physical data model as a DDL SQL script (including domain rules and constraints) while trying to comply with the SQL standard (omitting, if possible, native SQL implementation constructs).
4. Create a database in MS SQL 2017 or 2019. The implemented database should be a physical data model of the modeled part of the reality.
5. Test the created database. Enter several records into each table, checking the correctness of the implementation (remember to check both correct data and inconsistent data with the applicable rules – please comment and explain the obtained messages from the DBMS system).

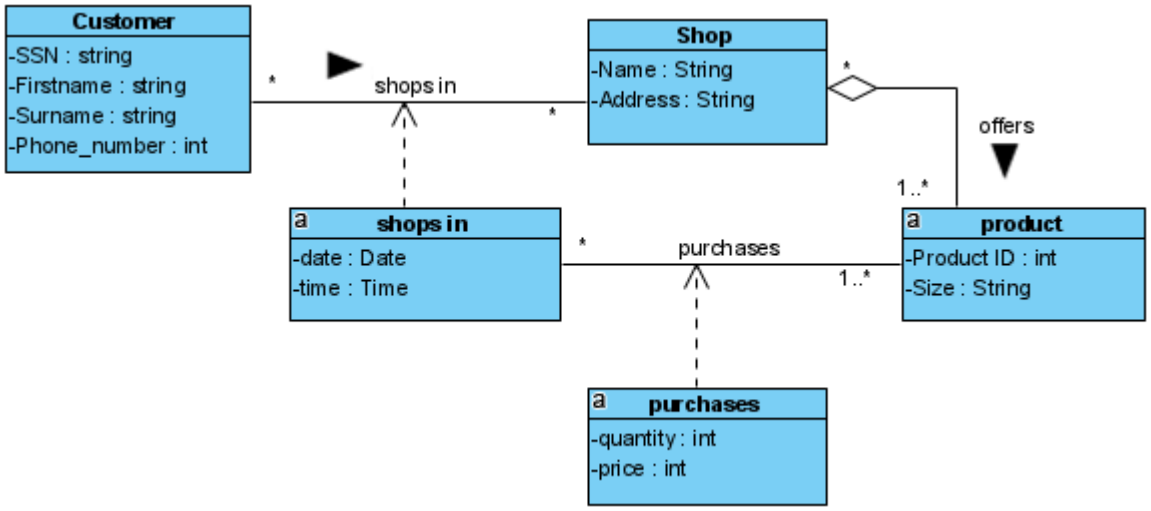
SOLUTIONS:

R01 – Any customer can shop in many stores (True)
R02 – A customer can shop repeatedly in the same store
(True)
R03 - Multiple customers can shop at the store (True)
R04 – Each purchase is made by the customer in the store on a specific day and time (True)
R05 – A store must offer at least one product (True)
R06 – The same product (type) can be offered by multiple stores (True)
R07 – Each store can individually propose the price and quantity of the offered product (True)
R08 – Shop must have 1 or more product

Constraints:

C01 – Price > 0
C02 – product > 0
C03 – date <=now
C04—time <= now
C05—CustomerID >= 0
C06—QuantityID >= 0
C07—price > 0

2. Provide a revised and complete version of the data model (complete UML class diagram).

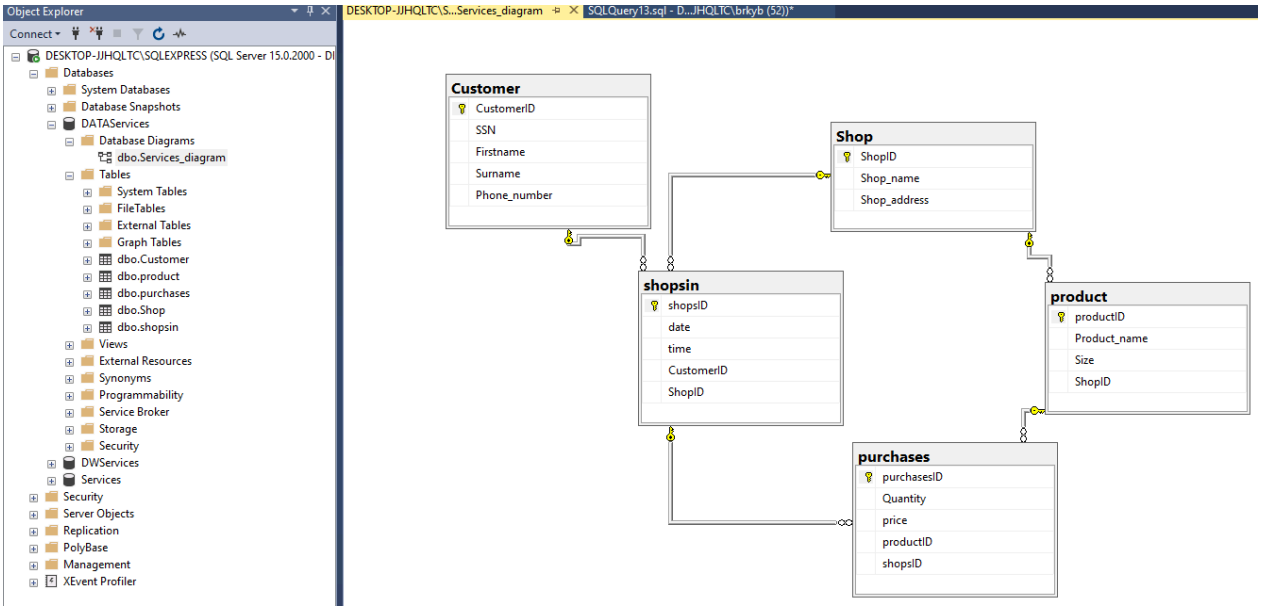


3- Create a logical/physical data model as a DDL SQL script (including domain rules and constraints) while trying to comply with the SQL standard (omitting, if possible, native SQL implementation constructs).

```
USE [DATAServices];

CREATE TABLE Customer (
CustomerID BIGINT NOT NULL PRIMARY KEY,
SSN varchar(11) NOT NULL,
Firstname varchar(255) NOT NULL,
Surname varchar(255) NOT NULL,
Phone_number BIGINT NOT NULL,
);

CREATE TABLE Shop(
ShopID int NOT NULL PRIMARY KEY,
Shop_name varchar(255) NOT NULL,
Shop_address varchar(255) NOT NULL,
);

CREATE TABLE shopsin(
shopsID int NOT NULL PRIMARY KEY,
date DATE NOT NULL,
time TIME NOT NULL,
CustomerID BIGINT NOT NULL FOREIGN KEY REFERENCES Customer(CustomerID),
ShopID int NOT NULL FOREIGN KEY REFERENCES Shop(ShopID),
);

CREATE TABLE product(
productID int NOT NULL PRIMARY KEY,
Product_name varchar(255) NOT NULL,
Size varchar(255) NOT NULL,
ShopID int NOT NULL FOREIGN KEY REFERENCES Shop(ShopID),
);
```

```
CREATE TABLE purchases(
purchasesID INT NOT NULL PRIMARY KEY,
Quantity INT NOT NULL,
price float NOT NULL,
productID INT NOT NULL FOREIGN KEY REFERENCES product(productID),
shopsID INT NOT NULL FOREIGN KEY REFERENCES shopsin(ShopsID),
);
```

4 Create a database in MS SQL 2017 or 2019. The implemented database should be a physical data model of the modeled part of the reality.



5 Test the created database. Enter several records into each table, checking the correctness of the implementation (remember to check both correct data and inconsistent data with the applicable rules – please comment and explain the obtained messages from the DBMS system).

```
USE [DATAServices];

--Customer
INSERT INTO Customer(CustomerID, SSN, Firstname, Surname, Phone_number)
VALUES (1, 1234, 'Berkay', 'Berber', 48739627591);
INSERT INTO Customer(CustomerID, SSN, Firstname, Surname, Phone_number)
VALUES (2, 2356, 'Furkan', 'Yılmaz', 48564545644);
INSERT INTO Customer(CustomerID, SSN, Firstname, Surname, Phone_number)
VALUES (3, 1526, 'Dilek', 'Yıldız', 48764654454);
INSERT INTO Customer(CustomerID, SSN, Firstname, Surname, Phone_number)
VALUES (4, 8795, 'Arvin', 'Timur', 48717893245);
INSERT INTO Customer(CustomerID, SSN, Firstname, Surname, Phone_number)
VALUES (5, 4526, 'Selin', 'Kaya', 48789486548);
```

```
--Shop
INSERT INTO Shop(ShopID, Shop_name, Shop_address)
VALUES (1, 'Dark Grace Shop', 'Miarki street');
INSERT INTO Shop(ShopID, Shop_name, Shop_address)
VALUES (2, 'Dash of Delish', 'Karola street');
INSERT INTO Shop(ShopID, Shop_name, Shop_address)
VALUES (3, 'The Saving Store', 'atatürk street');
INSERT INTO Shop(ShopID, Shop_name, Shop_address)
VALUES (4, 'Delicacy shop', 'Inonu street');
INSERT INTO Shop(ShopID, Shop_name, Shop_address)
VALUES (5, 'Desco', 'Aga street');

--Product
INSERT INTO product(productID, Product_name, Size, ShopID)
VALUES (1, 'Phone', 'S', 1);
INSERT INTO product(productID, Product_name, Size, ShopID)
VALUES (2, 'Laptop', 'M', 2);
INSERT INTO product(productID, Product_name, Size, ShopID)
VALUES (3, 'Earphone', 'L', 3);
INSERT INTO product(productID, Product_name, Size, ShopID)
VALUES (4, 'Keyboard', 'XL', 4);
INSERT INTO product(productID, Product_name, Size, ShopID)
VALUES (5, 'Mouse', 'XXL', 5);

--purchases
INSERT INTO purchases(purchasesID, Quantity, price, productID, shopsID)
VALUES (1, 1, 150, 1, 1);
INSERT INTO purchases(purchasesID, Quantity, price, productID, shopsID)
VALUES (2, 5, 750, 2, 2);
INSERT INTO purchases(purchasesID, Quantity, price, productID, shopsID)
VALUES (3, 8, 1500, 3, 3);
INSERT INTO purchases(purchasesID, Quantity, price, productID, shopsID)
VALUES (4, 3, 1150, 4, 4);
INSERT INTO purchases(purchasesID, Quantity, price, productID, shopsID)
VALUES (5, 2, 600, 5, 5);
--shopsin
INSERT INTO shopsin(shopsID, date, time, CustomerID, ShopID)
VALUES (1, '2021-07-10', '02-16-09', 1, 1);
INSERT INTO shopsin(shopsID, date, time, CustomerID, ShopID)
VALUES (2, '2020-06-09', '03-17-14', 2, 2);
INSERT INTO shopsin(shopsID, date, time, CustomerID, ShopID)
VALUES (3, '2019-08-12', '04-18-15', 3, 3);
INSERT INTO shopsin(shopsID, date, time, CustomerID, ShopID)
VALUES (4, '2018-03-14', '05-19-07', 4, 4);
INSERT INTO shopsin(shopsID, date, time, CustomerID, ShopID)
VALUES (5, '2021-09-18', '06-20-08', 5, 5);
```

There is above the code of how to insert data into created table one by one.
First of all, Using INSERT INTO ‘table name’ command ı declared the created attributes of the specific tables and than using VALUES command, ı had a chance to put my data accordingly to specified types of them when creating the tables.

There are representation of the tables after the inserted data in it.

Tables are represented with the command of:

```
USE [DATAServices];

SELECT * FROM product;
SELECT * FROM Customer;
SELECT * FROM purchases;
SELECT * FROM shopsin;
SELECT * FROM Shop;
```

	CustomerID	SSN	Firstname	Surname	Phone_number
▶	1	1234	Berkay	Berber	48739627591
	2	2356	Furkan	Yilmaz	48564545644
	3	1526	Dilek	Yildiz	48764654454
	4	8795	Arvin	Timur	48717893245
	5	4526	Selin	Kaya	48789486548
*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

	productID	Product_name	Size	ShopID
▶	1	Phone	S	1
	2	Laptop	M	2
	3	Earphone	L	3
	4	Keyboard	XL	4
	5	Mouse	XXL	5
*	*NULL*	*NULL*	*NULL*	*NULL*

	ShopID	Shop_name	Shop_address
▶	1	Dark Grace Shop	Miarki street
	2	Dash of Delish	Karola street
	3	The Saving Store	atatürk street
	4	Delicacy shop	Inonu street
	5	Desco	Aga street
*	*NULL*	*NULL*	*NULL*

	shopsID	date	time	CustomerID	ShopID
▶		2021-07-10	02:16:09	1	1
	2	2020-06-09	03:17:14	2	2
	3	2019-08-12	04:18:15	3	3
	4	2018-03-14	05:19:07	4	4
	5	2021-09-18	06:20:08	5	5
*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

	purchasesID	Quantity	price	productID	shopsID
▶	1	1	150	1	1
	2	5	750	2	2
	3	8	1500	3	3
	4	3	1150	4	4
	5	2	600	5	5
*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

I am sharing below my testing part and all the defined colums at left-side.