CSE 414 Databases

Final Project

Akif Kartal

19 June 2022

Database Tables and Relations

In this project we have **25 table**. Tables are as follows.

- Customer
- Address
- AddressType
- CustomerAdress
- TelephoneNumber
- Email
- Orders
- FavouriteRestaurant
- Basket
- BasketFood
- OrderFood
- DigitalWallet
- Restaurant
- PaymentType
- PaymentRestaurant
- WorkingHours
- Food
- Category
- Review
- Campaign
- Coupon
- CouponCustomer
- FoodLog
- WalletLog
- Offer

Relations

Between these tables we have following relations;

- 4 one to one relation
- 20 one to many relation
- 6 many to many relation

Total we have **30 relations.**

Creating Tables and Inserting Data

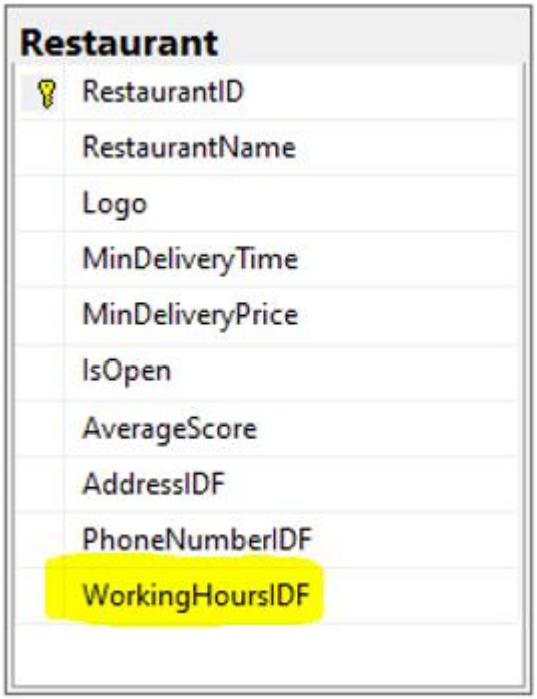
In this database everything made with SQL language. You can see in following images.

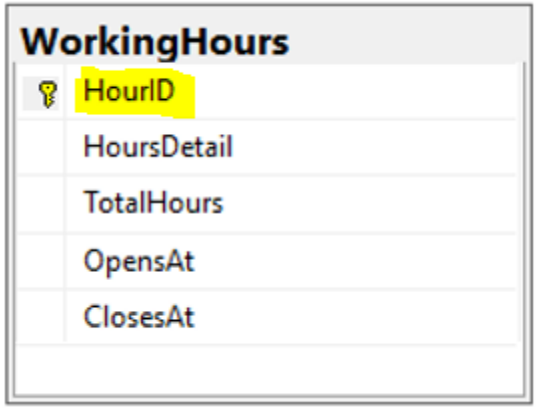
```
Create Table WalletLog (
    LogID int identity (1, 1) NOT NULL,
    LogTime datetime NULL,
    OperationDetails nvarchar(MAX) NULL ,
    OldBalance money NULL,
    NewBalance money NULL,
    WalletIDF int NULL,

    CONSTRAINT PK_WalletLog PRIMARY KEY CLUSTERED
    (
        LogID
    ),
    CONSTRAINT FK_Wallet_WalletLog FOREIGN KEY
    (
        WalletIDF
    ) REFERENCES dbo.DigitalWallet (
        WalletID
    )
)
```

Normalization

In this project, I have **3NF** normalization. For example, we have WorkingHours table for restaurant. This is example of 3NF and BCNF normalization.





Functional Dependency

HourID ⟶ HoursDetail, TotalHour, OpensAt, ClosesAt

Triggers

In this project, I have **5 triggers**. They are as follows.

1. trg_LogWallet
2. trg_ReviewAverage
3. trg_BasketPriceUpdate
4. trg_LogFood
5. trg_OrderFood

Views

In this project, I have **5 views**. They are as follows.

1. vm_Customer
2. vm_Order
3. vm_Max40Food
4. vm_Restaurant
5. vm_Comment

Join Queries

Since Customer and Email tables are relational tables, we can join them by using following queries.

```
➢ select * from Customer c Left Outer Join Email e on c.EmailIDF
= e.MailID

➢ select * from Customer c Right Outer Join Email e on
c.EmailIDF = e.MailID

➢ select * from Customer c Full Outer Join Email e on c.EmailIDF
= e.MailID
```

I also used inner join in views

```
from Customer c

inner join Email e on c.EmailIDF = e.MailID

inner join Basket b on b.BasketID = c.BasketIDF

inner join DigitalWallet d on d.WalletID = c.WalletIDF
```

Extra Details

In this project, I have additional features.

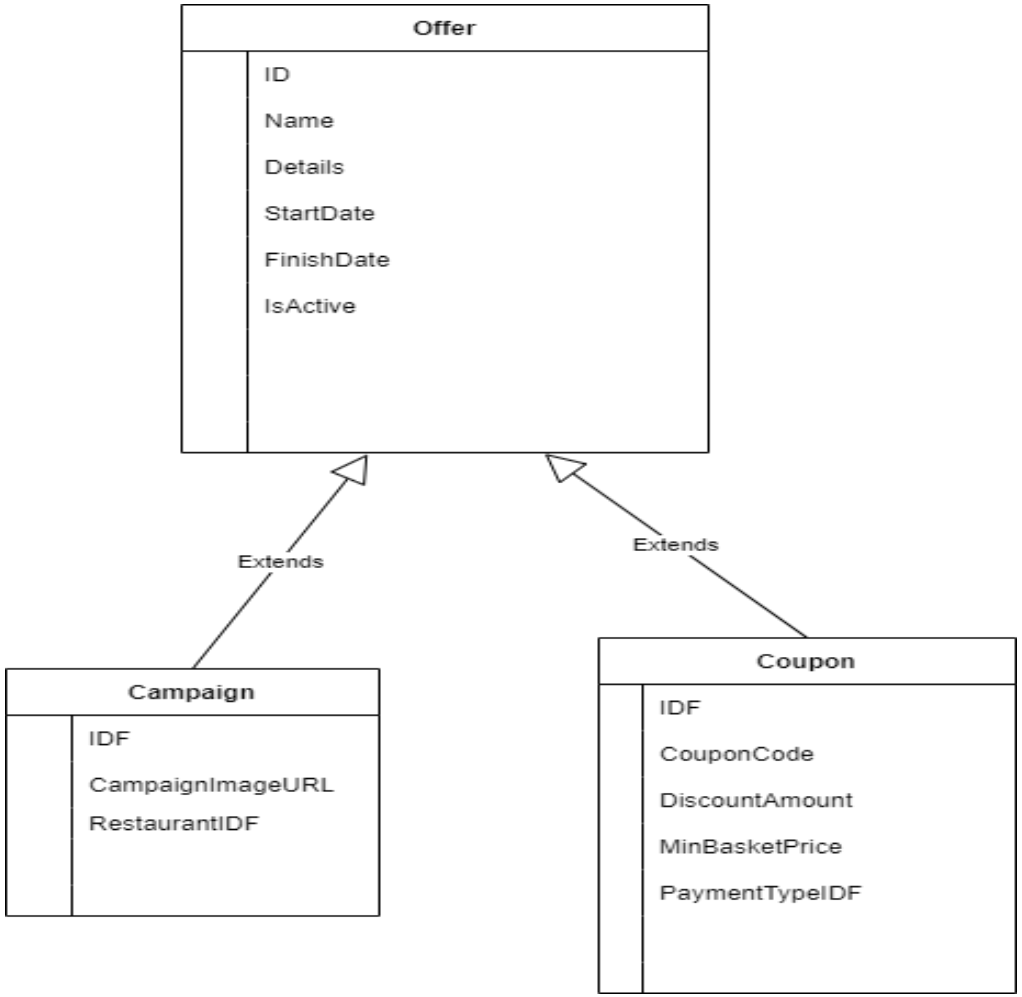
Transactions

In this project, I have **3 transaction**. They are as follows.

1. OrderPayment
2. insertCustomer
3. insertReview

Inheritance (Specialization)

In this project, I have implemented **inheritance**. Between Campaign and Coupon tables there is an inheritance.



Using Store Procedure

In this project, I have used store procedures.

```
create procedure sp_insertCustomerTransaction
(
    @FirstName nvarchar (50),
    @LastName nvarchar (50),
    @BirthDate date NULL ,
    @Password nvarchar (MAX) NULL ,
    @EmailIDF int NULL ,
    @BasketIDF int NULL ,
    @WalletIDF int NULL
)
as
Begin Transaction insertCustomer
declare @tempWalletId int = 0 , @tempEmailId int = 0, @tempBasketId int = 0
Select @tempWalletId = WalletIDF from Customer where WalletIDF = @WalletIDF
```

```
exec sp_insertCustomerTransaction 'Ahmet','Yılmaz','1998-07-27','test1',3,3,7
```

Using Check Constraint

In this project, I have used check constraint in customer create table query.

```
    CONSTRAINT FK_Customers_Baskets FOREIGN KEY
    (
        BasketIDF
    ) REFERENCES dbo.Basket (
        BasketID
    ),
    CONSTRAINT CK_Birthdate CHECK (BirthDate < getdate())
)
```


DEMO