Veritabanı Tasarımı:

HASAN YILDIZ

Proje Adı: " Veritabanı Tasarımı ve Uygulaması"
Proje Amaçları:
Proje Gereksinimleri:

1. Veritabanı Tasarımı:

```
CREATE DATABASE VeritabaniTasarimiVeUygulamasi
```

1. Ürünler Tablosu;

```
CREATE TABLE Ürünler (
    ÜrünID INT PRIMARY KEY IDENTITY(1,1),
    KategoriID INT,
    ÜrünAdı NVARCHAR(255) NOT NULL,
    Açıklama NVARCHAR(MAX),
    Fiyat DECIMAL(10,2) NOT NULL,
    Stok INT NOT NULL,
);
```

2. Kategoriler Tablosu;

```
CREATE TABLE Kategoriler (
    KategoriID INT PRIMARY KEY IDENTITY(1,1),
    KategoriAdı NVARCHAR(100) NOT NULL,
    Açıklama NVARCHAR(MAX)
);
```

3. Müşteriler Tablosu;

```
CREATE TABLE Müşteriler (
    MüşteriID INT PRIMARY KEY IDENTITY(1,1),
    Ad NVARCHAR(50) NOT NULL,
    Soyad NVARCHAR(50) NOT NULL,
    Cinsiyet CHAR(1),
    DoğumTarihi DATE,
    Telefon NVARCHAR(20),
    Email NVARCHAR(100),
    Adres NVARCHAR(MAX)
);
```

4. Siparişler Tablosu;

```
CREATE TABLE Siparişler (
    SiparişID INT PRIMARY KEY IDENTITY(1,1),
    MüşteriID INT,
    SiparişTarihi DATETIME NOT NULL,
    ToplamTutar DECIMAL(10,2) NOT NULL,
    KargoAdresi NVARCHAR(MAX),
    SiparişDurumu NVARCHAR(50)
);
```

Veritabanı Tasarımı:

5. Sipariş Detayları Tablosu;

```
CREATE TABLE SiparişDetayları (
    SiparişID INT,
    ÜrünID INT,
    Adet INT NOT NULL,
    BirimFiyat DECIMAL(10,2) NOT NULL,
    İndirim DECIMAL(10,2),
    PRIMARY KEY (SiparişID, ÜrünID)
);
```

2. Veri Girişi:

While döngüsü ile Müşteri oluşturma;

```
DECLARE @i INT = 1;
WHILE @i <= 100000
BEGIN
    INSERT INTO Müşteriler (Ad, Soyad, Cinsiyet, DoğumTarihi, Telefon, Email,
Adres)
    VALUES (
        'Ad' + CAST(@i AS NVARCHAR(10)),
        'Soyad' + CAST(@i AS NVARCHAR(10)),
        CASE WHEN RAND() < 0.5 THEN 'E' ELSE 'K' END,
        DATEADD(DAY, -CAST(RAND() * 365 * 50 AS INT), GETDATE()),
        '5' + RIGHT('0000000000' + CAST(CAST(RAND() * 1000000000 AS INT) AS
VARCHAR(10)), 10),
        'email' + CAST(@i AS NVARCHAR(10)) + '@email.com',
        'Adres' + CAST(@i AS NVARCHAR(10))
    );
    SET @i = @i + 1;
END;
```

While döngüsü ile Sipariş oluşturma;

```
DECLARE @i INT = 1;
DECLARE @MusteriID INT;
DECLARE @SiparisTarihi DATETIME;
DECLARE @ToplamTutar DECIMAL(10,2);
DECLARE @KargoAdresi NVARCHAR(MAX);
DECLARE @SiparisDurumu NVARCHAR(50);

WHILE @i <= 100000
BEGIN

    SELECT TOP 1 @MusteriID = MüşteriID FROM Müşteriler ORDER BY NEWID();

    SET @SiparisTarihi = DATEADD(DAY, -CAST(RAND() * 365 AS INT), GETDATE());

    SET @ToplamTutar = RAND() * 9900 + 100;

    SET @KargoAdresi = 'Adres' + CAST(@i AS NVARCHAR(10));
```

Veritabanı Tasarımı:

```
    SET @SiparisDurumu = (
        SELECT TOP 1 SiparişDurumu
        FROM (VALUES ('Hazırlanıyor'), ('Kargolandı'), ('Teslim Edildi')) AS
Durumlar(SiparişDurumu)
        ORDER BY NEWID()
    );

    INSERT INTO Siparişler (MüşteriID, SiparişTarihi, ToplamTutar,
KargoAdresi, SiparişDurumu)
    VALUES (@MusteriID, @SiparisTarihi, @ToplamTutar, @KargoAdresi,
@SiparisDurumu);

    SET @i = @i + 1;
END;
```

While döngüsü ile Kategori oluşturma;

```
DECLARE @i INT = 1;
WHILE @i <= 100000
BEGIN
    INSERT INTO Kategoriler (KategoriAdı, Açıklama)
    VALUES (
        'Kategori' + CAST(@i AS NVARCHAR(10)),
        'Bu ' + CAST(@i AS NVARCHAR(10)) + '. kategoridir.'
    );
    SET @i = @i + 1;
END;
```

While döngüsü ile Ürün oluşturma;

```
DECLARE @i INT = 1;
DECLARE @KategoriID INT;
DECLARE @UrunAdi NVARCHAR(255);
DECLARE @Aciklama NVARCHAR(MAX);
DECLARE @Fiyat DECIMAL(10,2);
DECLARE @Stok INT;

WHILE @i <= 100000
BEGIN
    SET @KategoriID = CAST(RAND() * 100000 + 1 AS INT);
    SET @UrunAdi = 'Ürün' + CAST(@i AS NVARCHAR(10));
    SET @Aciklama = 'Bu ' + CAST(@i AS NVARCHAR(10)) + '. üründür.';
    SET @Fiyat = RAND() * 9990 + 10;
    SET @Stok = CAST(RAND() * 100 + 1 AS INT);

    INSERT INTO Ürünler (KategoriID, ÜrünAdı, Açıklama, Fiyat, Stok)
    VALUES (@KategoriID, @UrunAdi, @Aciklama, @Fiyat, @Stok);

    SET @i = @i + 1;
END;
```

Veritabanı Tasarımı:

While döngüsü ile SiparişDetayları oluşturma;

```
DECLARE @i INT = 1;
DECLARE @SiparisID INT;
DECLARE @UrunID INT;
DECLARE @Adet INT;
DECLARE @BirimFiyat DECIMAL(10,2);
DECLARE @Indirim DECIMAL(10,2);

WHILE @i <= 100000
BEGIN
    SELECT TOP 1 @SiparisID = SiparişID FROM Siparişler ORDER BY NEWID();
    SELECT TOP 1 @UrunID = ÜrünID FROM Ürünler ORDER BY NEWID();
    SET @Adet = CAST(RAND() * 4 + 1 AS INT);
    SELECT @BirimFiyat = Fiyat FROM Ürünler WHERE ÜrünID = @UrunID;
    SET @Indirim = RAND() * 0.20 * @BirimFiyat;

    INSERT INTO SiparişDetayları (SiparişID, ÜrünID, Adet, BirimFiyat,
İndirim)
    VALUES (@SiparisID, @UrunID, @Adet, @BirimFiyat, @Indirim);

    SET @i = @i + 1;
END;
```

3. İlişkiler : FOREIGN KEY kısıtlamaları

```
ALTER TABLE Ürünler ADD FOREIGN KEY (KategoriID) REFERENCES
Kategoriler(KategoriID);
ALTER TABLE Siparişler ADD FOREIGN KEY (MüşteriID) REFERENCES
Müşteriler(MüşteriID);
ALTER TABLE SiparişDetayları ADD FOREIGN KEY (SiparişID) REFERENCES
Siparişler(SiparişID);
ALTER TABLE SiparişDetayları ADD FOREIGN KEY (ÜrünID) REFERENCES
Ürünler(ÜrünID);
```

4. Stored Procedure:

```
CREATE PROCEDURE sp_SiparisVer
@MusteriID INT,
@UrunID INT,
@Adet INT
AS
BEGIN
SET NOCOUNT ON;
DECLARE @ToplamTutar DECIMAL(10,2);
DECLARE @OncekiSiparisSayisi INT;
```

Veritabanı Tasarımı:

1. Ürün fiyatını ve stok durumunu kontrol etme;

```
DECLARE @UrunFiyati DECIMAL(10,2);
DECLARE @UrunStok INT;
SELECT @UrunFiyati = Fiyat, @UrunStok = Stok
FROM Ürünler WHERE ÜrünID = @UrunID;
IF @UrunFiyati IS NULL
THROW 50001, 'Geçersiz ürün ID', 1;
IF @UrunStok < @Adet
THROW 50002, 'Yetersiz stok', 1;
```

2. Toplam tutarı hesaplama;

```
SET @ToplamTutar = @UrunFiyati * @Adet;
```

3. Siparişi oluşturma;

```
DECLARE @SiparisID INT;
INSERT INTO Siparişler (MüşteriID, SiparişTarihi, ToplamTutar, SiparişDurumu)
VALUES (@MusteriID, GETDATE(), @ToplamTutar, 'Hazırlanıyor');
SET @SiparisID = SCOPE_IDENTITY();
```

4. Sipariş detayını ekleme;

```
INSERT INTO SiparişDetayları (SiparişID, ÜrünID, Adet, BirimFiyat)
VALUES (@SiparisID, @UrunID, @Adet, @UrunFiyati);
```

5. Ürün stok miktarını güncelleme;

```
    UPDATE Ürünler SET Stok = Stok - @Adet WHERE ÜrünID = @UrunID;
    - Müşterinin önceki sipariş sayısını alma
    SELECT @OncekiSiparisSayisi = COUNT(*)
    FROM Siparişler WHERE MüşteriID = @MusteriID;
```

6. Sonuçları döndürme;

```
    SELECT @SiparisID AS SiparisID, @ToplamTutar AS ToplamTutar,
@OncekiSiparisSayisi AS OncekiSiparisSayisi;
END;
```

5. Trigger:

```
CREATE TRIGGER tr_StokGuncelle ON SiparişDetayları
AFTER INSERT
AS
BEGIN
    SET NOCOUNT ON;
    UPDATE Ürünler
    SET Stok = Stok - (SELECT Adet FROM inserted WHERE ÜrünID =
Ürünler.ÜrünID)
    WHERE ÜrünID IN (SELECT ÜrünID FROM inserted);
END;
```

Veritabanı Tasarımı:

6. View:

```
CREATE VIEW vw_KategoriSatislari AS
SELECT K.KategoriAdı, SUM(SD.Adet * SD.BirimFiyat) AS ToplamSatis
FROM Kategoriler K
JOIN Ürünler U ON K.KategoriID = U.KategoriID
JOIN SiparişDetayları SD ON U.ÜrünID = SD.ÜrünID
GROUP BY K.KategoriAdı;
```

7. SQL Sorguları:

1. En çok satan 5 ürün bulma;

```
SELECT TOP 5 U.ÜrünAdı, SUM(SD.Adet) AS ToplamSatisAdedi
FROM Ürünler U
JOIN SiparişDetayları SD ON U.ÜrünID = SD.ÜrünID
GROUP BY U.ÜrünAdı
ORDER BY ToplamSatisAdedi DESC;
```

2. Belirli bir tarih aralığındaki toplam satışları bulma;

```
SELECT SUM(ToplamTutar) AS ToplamSatis
FROM Siparişler
WHERE SiparişTarihi BETWEEN '2024-05-01' AND '2024-05-31';
```

3. Müşteri bazında sipariş geçmişini bulma;

```
SELECT M.Ad, M.Soyad, S.SiparişID, S.SiparişTarihi, S.ToplamTutar
FROM Müşteriler M
JOIN Siparişler S ON M.MüşteriID = S.MüşteriID
ORDER BY M.MüşteriID, S.SiparişTarihi;
```

4. Toplam satışları hesaplama;

```
SELECT SUM(ToplamTutar) AS ToplamSatislar
FROM Siparişler;
```

5. MüşteriID'si 1 olan müşterinin siparişleri

```
SELECT S.SiparişID, S.SiparişTarihi, S.ToplamTutar, S.SiparişDurumu
FROM Siparişler S
JOIN Müşteriler M ON S.MüşteriID = M.MüşteriID
WHERE M.MüşteriID = 1;
```

6. --En çok satan ürünleri görme;

```
SELECT TOP 10 U.ÜrünAdı, SUM(SD.Adet) AS ToplamSatılanAdet
FROM Ürünler U
JOIN SiparişDetayları SD ON U.ÜrünID = SD.ÜrünID
GROUP BY U.ÜrünAdı
ORDER BY ToplamSatılanAdet DESC;
```

Veritabanı Tasarımı:

7. Elektronik kategorisindeki ürünleri görme;

```
SELECT U.ÜrünAdı, U.Fiyat, U.Stok
FROM Ürünler U
JOIN Kategoriler K ON U.KategoriID = K.KategoriID
WHERE K.KategoriAdı = 'Elektronik';
```

8. Stoğu 100'den az olan ürünler;

```
SELECT ÜrünAdı, Stok
FROM Ürünler
WHERE Stok < 100;
```

9. SiparişID'si 1 olan siparişin detayları;

```
SELECT S.SiparişID, U.ÜrünAdı, SD.Adet, SD.BirimFiyat, (SD.Adet *
SD.BirimFiyat) AS Tutar
FROM Siparişler S
JOIN SiparişDetayları SD ON S.SiparişID = SD.SiparişID
JOIN Ürünler U ON SD.ÜrünID = U.ÜrünID
WHERE S.SiparişID = 1;
```

10. Bir Ürünün Fiyatını Güncelleme:

```
UPDATE Ürünler
SET Fiyat = 12000.00
WHERE ÜrünID = 1;
```

11. Bir Müşterinin Adresini Değiştirme:

```
UPDATE Müşteriler
SET Adres = 'Mersin'
WHERE MüşteriID = 1;
```

12. Bir Siparişi İptal Etme (Silme):

```
DELETE FROM Siparişler
WHERE SiparişID = 1;
```