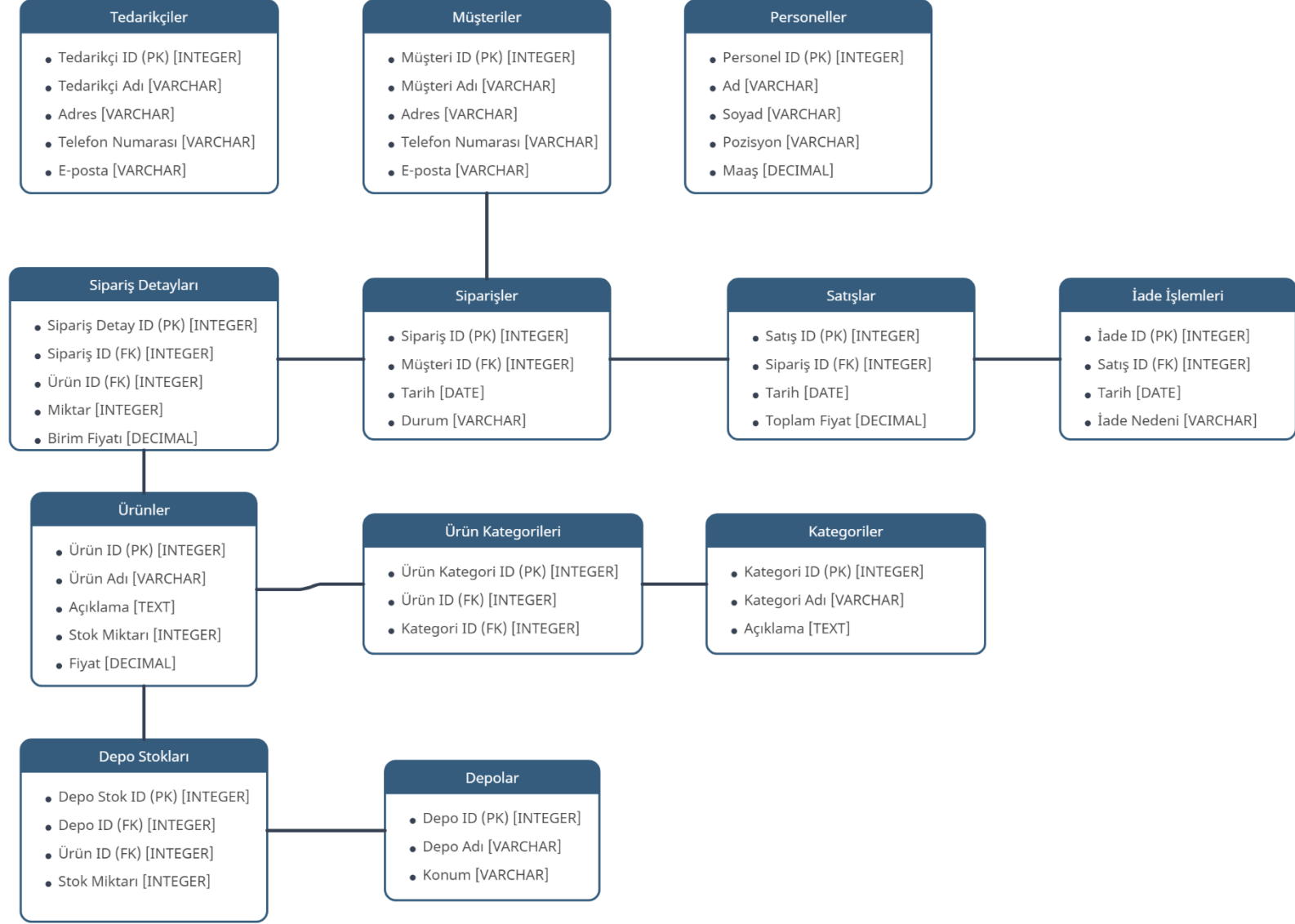


Muhammed Emin Ay

20120205038

Soru1)



## Soru2)

```
CREATE TABLE Urunler (  
    UrunID NUMBER PRIMARY KEY,  
    UrunAdi VARCHAR2(100),  
    Aciklama VARCHAR2(255),  
    StokMiktari NUMBER,  
    Fiyat NUMBER  
);
```

```
CREATE TABLE Tedarikciler (  
    TedarikciID NUMBER PRIMARY KEY,  
    TedarikciAdi VARCHAR2(100),  
    Adres VARCHAR2(255),  
    TelefonNumarasi VARCHAR2(20),  
    Eposta VARCHAR2(100)  
);
```

```
CREATE TABLE Depolar (  
    DepoID NUMBER PRIMARY KEY,  
    DepoAdi VARCHAR2(100),  
    Konum VARCHAR2(255)  
);
```

```
CREATE TABLE DepoStoklari (  
    DepoStokID NUMBER PRIMARY KEY,  
    DepoID NUMBER,  
    UrunID NUMBER,  
    StokMiktari NUMBER,  
    FOREIGN KEY (DepoID) REFERENCES Depolar(DepoID),  
    FOREIGN KEY (UrunID) REFERENCES Urunler(UrunID)  
);
```

```
CREATE TABLE Musteriler (  
    MusteriID NUMBER PRIMARY KEY,  
    MusteriAdi VARCHAR2(100),  
    Adres VARCHAR2(255),  
    TelefonNumarasi VARCHAR2(20),  
    Eposta VARCHAR2(100)  
);
```

```
CREATE TABLE Siparisler (  
    SiparisID NUMBER PRIMARY KEY,  
    MusteriID NUMBER,  
    Tarih DATE,  
    Durum VARCHAR2(50),  
    FOREIGN KEY (MusteriID) REFERENCES Musteriler(MusteriID)  
);
```

```
CREATE TABLE SiparisDetaylari (  
    SiparisDetayID NUMBER PRIMARY KEY,  
    SiparisID NUMBER,  
    UrunID NUMBER,  
    Miktar NUMBER,  
    BirimFiyati NUMBER,  
    FOREIGN KEY (SiparisID) REFERENCES Siparisler(SiparisID),  
    FOREIGN KEY (UrunID) REFERENCES Urunler(UrunID)  
);
```

```
CREATE TABLE Personel (  
    PersonelID NUMBER PRIMARY KEY,  
    Ad VARCHAR2(100),  
    Soyad VARCHAR2(100),
```

```
Pozisyon VARCHAR2(100),  
Maas NUMBER  
);
```

```
CREATE TABLE Satislar (  
    SatisID NUMBER PRIMARY KEY,  
    SiparisID NUMBER,  
    Tarih DATE,  
    ToplamFiyat NUMBER,  
    FOREIGN KEY (SiparisID) REFERENCES Siparisler(SiparisID)  
);
```

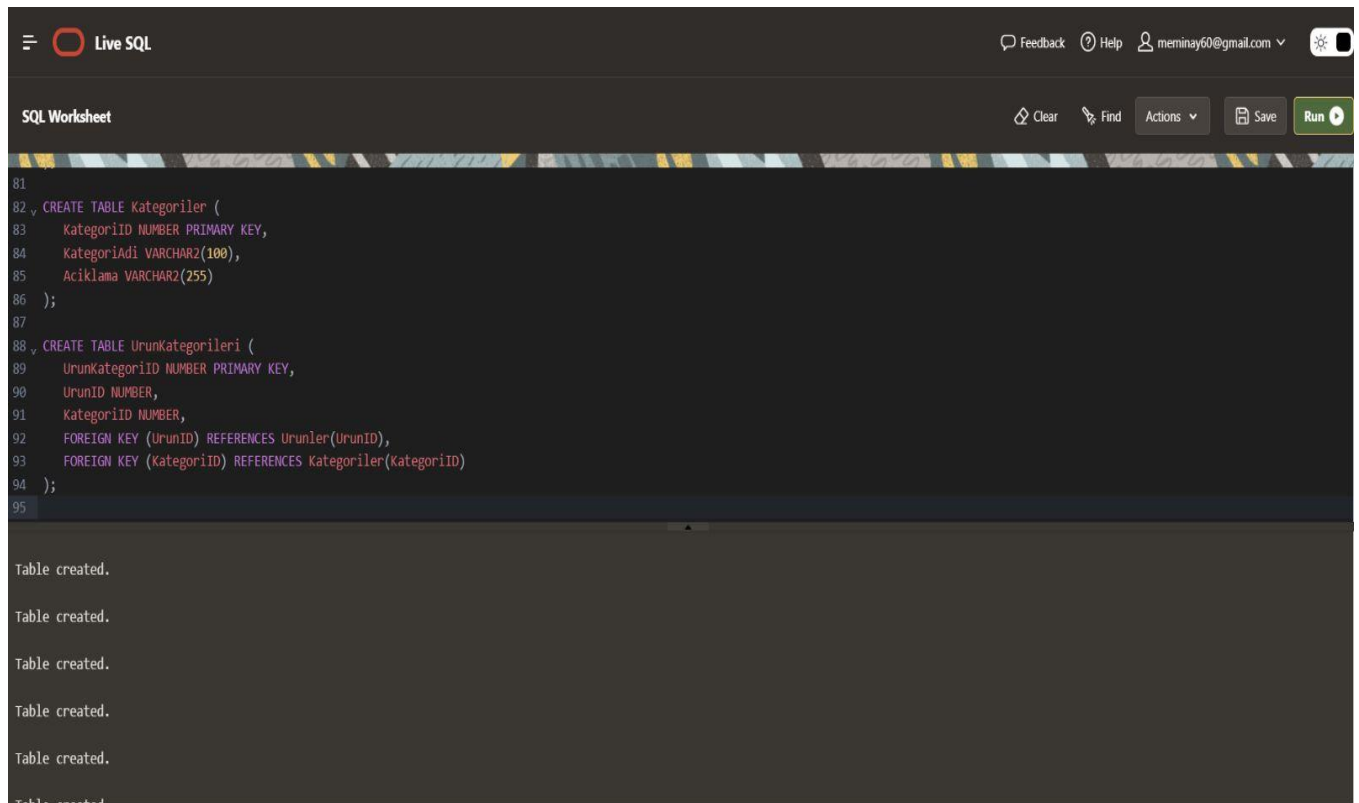
```
CREATE TABLE ladeIslemleri (  
    ladeID NUMBER PRIMARY KEY,  
    SatisID NUMBER,  
    Tarih DATE,  
    ladeNedeni VARCHAR2(255),  
    FOREIGN KEY (SatisID) REFERENCES Satislar(SatisID)  
);
```

```
CREATE TABLE Kategoriler (  
    KategoriID NUMBER PRIMARY KEY,  
    KategoriAdi VARCHAR2(100),  
    Aciklama VARCHAR2(255)  
);
```

```
CREATE TABLE UrunKategorileri (  
    UrunKategoriID NUMBER PRIMARY KEY,  
    UrunID NUMBER,  
    KategoriID NUMBER,  
    FOREIGN KEY (UrunID) REFERENCES Urunler(UrunID),
```

FOREIGN KEY (KategoriID) REFERENCES Kategoriler(KategoriID)

);



```
81
82 v CREATE TABLE Kategoriler (
83     KategoriID NUMBER PRIMARY KEY,
84     KategoriAdi VARCHAR2(100),
85     Aciklama VARCHAR2(255)
86 );
87
88 v CREATE TABLE Urunkategorileri (
89     UrunkategoriID NUMBER PRIMARY KEY,
90     UrunID NUMBER,
91     KategoriID NUMBER,
92     FOREIGN KEY (UrunID) REFERENCES Urunler(UrunID),
93     FOREIGN KEY (KategoriID) REFERENCES Kategoriler(KategoriID)
94 );
95
```

Table created.

Table created.

Table created.

Table created.

Table created.

Table created.

### Soru3)

CREATE INDEX FK\_DepoStoklari\_DepoID ON DepoStoklari (DepoID);

CREATE INDEX FK\_DepoStoklari\_UrunID ON DepoStoklari (UrunID);

CREATE INDEX IX\_DepoStoklari\_StokMiktari ON DepoStoklari (StokMiktari);

CREATE INDEX FK\_SiparisDetaylari\_SiparisID ON SiparisDetaylari (SiparisID);

CREATE INDEX FK\_SiparisDetaylari\_UrunID ON SiparisDetaylari (UrunID);

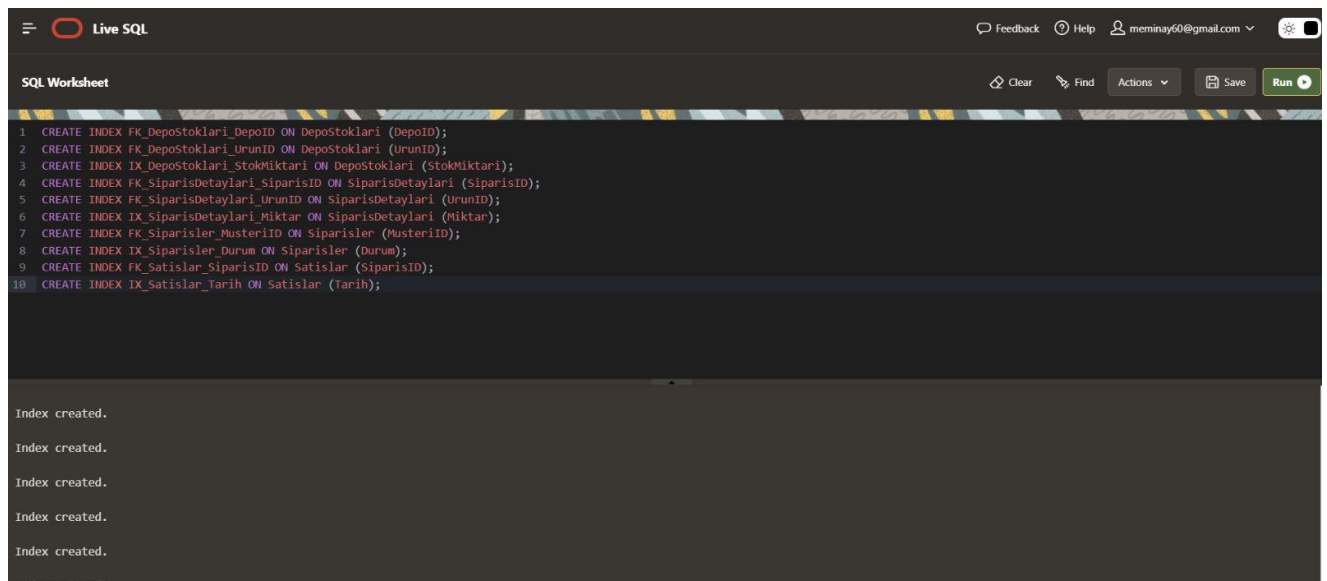
CREATE INDEX IX\_SiparisDetaylari\_Miktar ON SiparisDetaylari (Miktar);

CREATE INDEX FK\_Siparisler\_MusteriID ON Siparisler (MusteriID);

CREATE INDEX IX\_Siparisler\_Durum ON Siparisler (Durum);

CREATE INDEX FK\_Satislar\_SiparisID ON Satislar (SiparisID);

CREATE INDEX IX\_Satislar\_Tarih ON Satislar (Tarih);



The screenshot shows a web-based SQL editor titled "Live SQL". The interface includes a top navigation bar with a menu icon, the "Live SQL" logo, and links for "Feedback", "Help", and a user profile "meminay60@gmail.com". Below the navigation bar is a toolbar with "Clear", "Find", "Actions", "Save", and a "Run" button. The main area contains a list of 10 SQL statements for creating indexes. The output pane at the bottom shows the results of these statements, all of which were successfully executed, resulting in "Index created." messages.

```
1 CREATE INDEX FK_DepoStoklari_DepoID ON DepoStoklari (DepoID);
2 CREATE INDEX FK_DepoStoklari_UrunID ON DepoStoklari (UrunID);
3 CREATE INDEX IX_DepoStoklari_StokMiktari ON DepoStoklari (StokMiktari);
4 CREATE INDEX FK_SiparisDetaylari_SiparisID ON SiparisDetaylari (SiparisID);
5 CREATE INDEX FK_SiparisDetaylari_UrunID ON SiparisDetaylari (UrunID);
6 CREATE INDEX IX_SiparisDetaylari_Miktar ON SiparisDetaylari (Miktar);
7 CREATE INDEX FK_Siparisler_MusteriID ON Siparisler (MusteriID);
8 CREATE INDEX IX_Siparisler_Durum ON Siparisler (Durum);
9 CREATE INDEX FK_Satislar_SiparisID ON Satislar (SiparisID);
10 CREATE INDEX IX_Satislar_Tarih ON Satislar (Tarih);
```

Index created.  
Index created.  
Index created.  
Index created.  
Index created.  
Index created.

## Soru4)

CREATE OR REPLACE PACKAGE InventoryManagement AS

    PROCEDURE AddProduct(

        p\_ProductID IN NUMBER,

        p\_ProductName IN VARCHAR2,

        p\_Description IN VARCHAR2,

        p\_StockQuantity IN NUMBER,

        p\_Price IN NUMBER

    );

END InventoryManagement;

/

CREATE OR REPLACE PACKAGE BODY InventoryManagement AS

    PROCEDURE AddProduct(

        p\_ProductID IN NUMBER,

        p\_ProductName IN VARCHAR2,

        p\_Description IN VARCHAR2,

        p\_StockQuantity IN NUMBER,

        p\_Price IN NUMBER

    ) IS

BEGIN

```

INSERT INTO Urunler (UrunID, UrunAdi, Aciklama, StokMiktari, Fiyat)

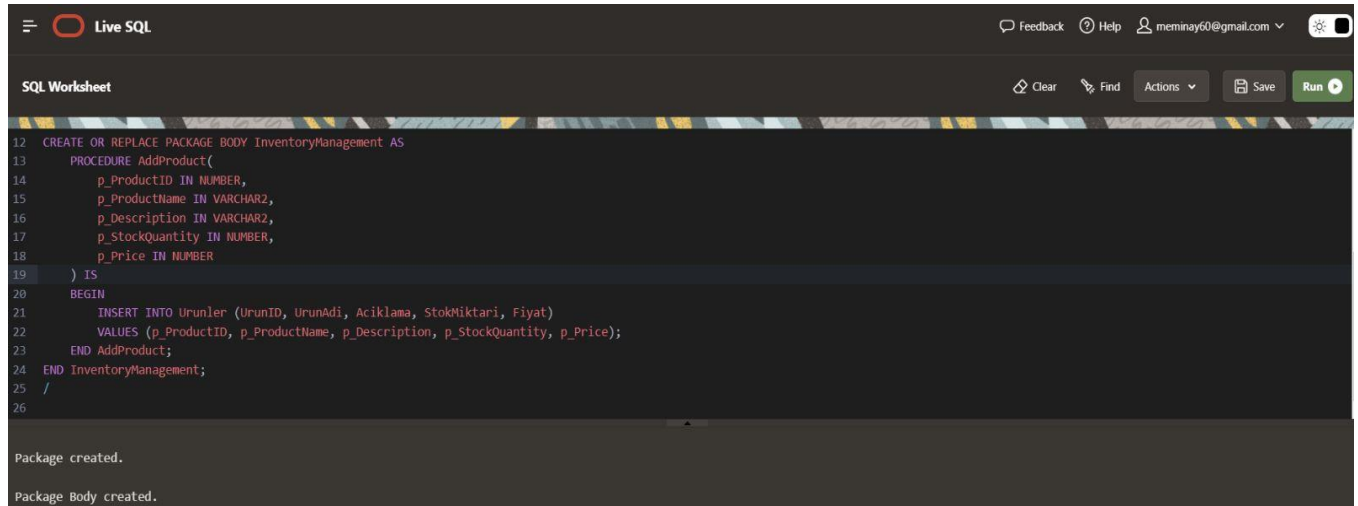
VALUES (p_ProductID, p_ProductName, p_Description, p_StockQuantity, p_Price);

END AddProduct;

END InventoryManagement;

/

```



```

Live SQL
Feedback Help meminay60@gmail.com
SQL Worksheet Clear Find Actions Save Run
12 CREATE OR REPLACE PACKAGE BODY InventoryManagement AS
13     PROCEDURE AddProduct(
14         p_ProductID IN NUMBER,
15         p_ProductName IN VARCHAR2,
16         p_Description IN VARCHAR2,
17         p_StockQuantity IN NUMBER,
18         p_Price IN NUMBER
19     ) IS
20     BEGIN
21         INSERT INTO Urunler (UrunID, UrunAdi, Aciklama, StokMiktari, Fiyat)
22         VALUES (p_ProductID, p_ProductName, p_Description, p_StockQuantity, p_Price);
23     END AddProduct;
24 END InventoryManagement;
25 /
26
Package created.
Package Body created.

```

```

BEGIN

InventoryManagement.AddProduct(1, 'Ürün 1', 'Bu bir ürün açıklamasıdır.', 10, 9.99);

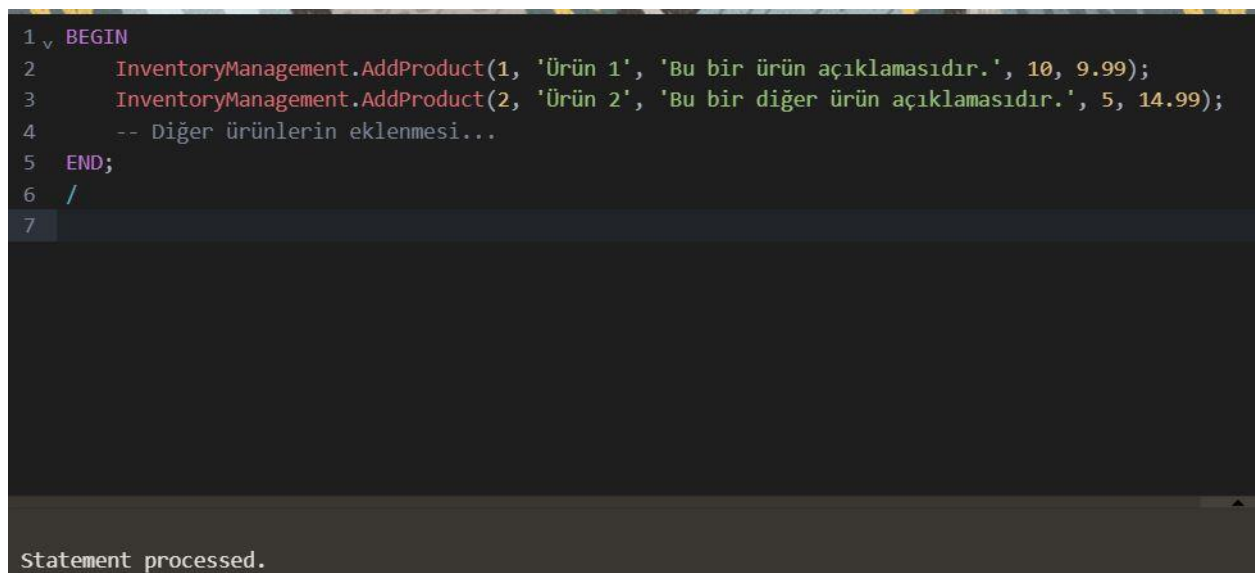
InventoryManagement.AddProduct(2, 'Ürün 2', 'Bu bir diğer ürün açıklamasıdır.', 5, 14.99);

-- Diğer ürünlerin eklenmesi...

END;

/

```



```

1 v BEGIN
2     InventoryManagement.AddProduct(1, 'Ürün 1', 'Bu bir ürün açıklamasıdır.', 10, 9.99);
3     InventoryManagement.AddProduct(2, 'Ürün 2', 'Bu bir diğer ürün açıklamasıdır.', 5, 14.99);
4     -- Diğer ürünlerin eklenmesi...
5 END;
6 /
7
Statement processed.

```

## Soru5)

```
CREATE OR REPLACE PACKAGE InventoryManagement AS
```

```
    PROCEDURE UpdateProduct(  
        p_ProductID IN NUMBER,  
        p_NewProductName IN VARCHAR2,  
        p_NewDescription IN VARCHAR2,  
        p_NewStockQuantity IN NUMBER,  
        p_NewPrice IN NUMBER  
    );
```

```
END InventoryManagement;
```

```
/
```

```
CREATE OR REPLACE PACKAGE BODY InventoryManagement AS
```

```
    PROCEDURE UpdateProduct(  
        p_ProductID IN NUMBER,  
        p_NewProductName IN VARCHAR2,  
        p_NewDescription IN VARCHAR2,  
        p_NewStockQuantity IN NUMBER,  
        p_NewPrice IN NUMBER  
    ) IS
```

```
    BEGIN
```

```
        UPDATE Urunler
```

```
        SET UrunAdi = p_NewProductName,
```

```
            Aciklama = p_NewDescription,
```

```
            StokMiktari = p_NewStockQuantity,
```

```
            Fiyat = p_NewPrice
```

```
        WHERE UrunID = p_ProductID;
```

```
    END UpdateProduct;
```

```
END InventoryManagement;
```



/

BEGIN

InventoryManagement.UpdateProduct(1, 'Yeni Ürün Adı', 'Yeni ürün açıklaması', 15, 19.99);

-- Diğer ürünlerin güncellenmesi...

END;

/

```
SQL Worksheet

22      SET UrunAdi = p_NewProductName,
23      Aciklama = p_NewDescription,
24      StokMiktari = p_NewStockQuantity,
25      Fiyat = p_NewPrice
26      WHERE UrunID = p_ProductID;
27  END UpdateProduct;
28  END InventoryManagement;
29  /
30
31  BEGIN
32      InventoryManagement.UpdateProduct(1, 'Yeni Ürün Adı', 'Yeni ürün açıklaması', 15, 19.99);
33      -- Diğer ürünlerin güncellenmesi...
34  END;
35  /
36

Package created.

Package Body created.

Statement processed.
```

## Soru6)

CREATE OR REPLACE PACKAGE InventoryManagement AS

PROCEDURE DeleteProduct(p\_ProductID IN NUMBER);

END InventoryManagement;

/

CREATE OR REPLACE PACKAGE BODY InventoryManagement AS

PROCEDURE DeleteProduct(p\_ProductID IN NUMBER) IS

```

BEGIN

    DELETE FROM Urunler WHERE UrunID = p_ProductID;

END DeleteProduct;

END InventoryManagement;

/

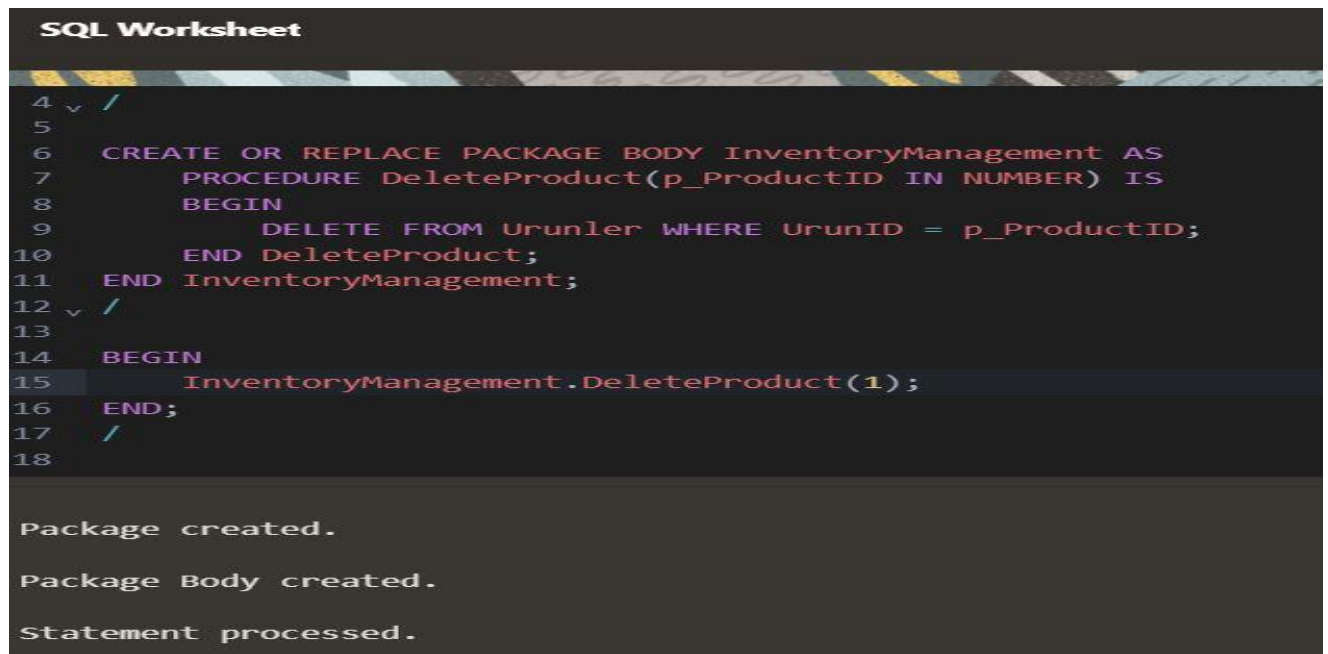
BEGIN

    InventoryManagement.DeleteProduct(1);

END;

/

```



```

SQL Worksheet

4 ✓ /
5
6 CREATE OR REPLACE PACKAGE BODY InventoryManagement AS
7     PROCEDURE DeleteProduct(p_ProductID IN NUMBER) IS
8     BEGIN
9         DELETE FROM Urunler WHERE UrunID = p_ProductID;
10    END DeleteProduct;
11 END InventoryManagement;
12 ✓ /
13
14 BEGIN
15     InventoryManagement.DeleteProduct(1);
16 END;
17 /
18

Package created.

Package Body created.

Statement processed.

```

## Soru7)

```

CREATE OR REPLACE TRIGGER urunler_insert_trigger

AFTER INSERT ON Urunler

FOR EACH ROW

DECLARE

    v_siparis_detay_id NUMBER;

BEGIN

    -- Yeni bir SiparisDetaylari kaydı oluşturuluyor

    INSERT INTO SiparisDetaylari (SiparisDetayID, SiparisID, UrunID, Miktar, BirimFiyati)

```

```
VALUES (SiparisDetaylari_Seq.NEXTVAL, NULL, :NEW.UrunID, 0, 0)
```

```
RETURNING SiparisDetayID INTO v_siparis_detay_id;
```

```
-- SiparisID değeri güncelleniyor
```

```
UPDATE SiparisDetaylari
```

```
SET SiparisID = v_siparis_detay_id
```

```
WHERE SiparisDetayID = v_siparis_detay_id;
```

```
COMMIT;
```

```
END;
```

```
/
```

**\*\*ORA-02289: sequence does not exist Hatası alınırsa ilk satıra ekleyin.**

```
CREATE SEQUENCE SiparisDetaylari_Seq START WITH 1 INCREMENT BY 1;
```

#### SQL Worksheet

```
1 v CREATE OR REPLACE TRIGGER urunler_insert_trigger
2   AFTER INSERT ON Urunler
3   FOR EACH ROW
4   DECLARE
5     v_siparis_detay_id NUMBER;
6 v BEGIN
7   -- Yeni bir SiparisDetaylari kaydı oluşturuluyor
8   INSERT INTO SiparisDetaylari (SiparisDetayID, SiparisID, UrunID, Miktar, BirimFiyati)
9   VALUES (SiparisDetaylari_Seq.NEXTVAL, NULL, :NEW.UrunID, 0, 0)
10  RETURNING SiparisDetayID INTO v_siparis_detay_id;
11
12  -- SiparisID değeri güncelleniyor
13 v UPDATE SiparisDetaylari
14  SET SiparisID = v_siparis_detay_id
15  WHERE SiparisDetayID = v_siparis_detay_id;
```

Trigger created.

## Soru8)

DECLARE

v\_depo\_id NUMBER := 1;-- Depo ID'si

BEGIN

FOR rec IN (SELECT \* FROM DepoStoklari ds

JOIN Depolar d ON ds.DepoID = d.DepoID

WHERE ds.DepoID = v\_depo\_id) LOOP

DBMS\_OUTPUT.PUT\_LINE('Depo Adı: ' || rec.DepoAdi);

DBMS\_OUTPUT.PUT\_LINE('Urun ID: ' || rec.UrunID);

DBMS\_OUTPUT.PUT\_LINE('Stok Miktarı: ' || rec.StokMiktari);

END LOOP;

END;

/

### SQL Worksheet

```
1 v DECLARE
2   v_depo_id NUMBER := 1; -- Depo ID'si
3 v BEGIN
4   FOR rec IN (SELECT * FROM DepoStoklari ds
5               JOIN Depolar d ON ds.DepoID = d.DepoID
6               WHERE ds.DepoID = v_depo_id) LOOP
7
8       DBMS_OUTPUT.PUT_LINE('Depo Adı: ' || rec.DepoAdi);
9       DBMS_OUTPUT.PUT_LINE('Urun ID: ' || rec.UrunID);
10      DBMS_OUTPUT.PUT_LINE('Stok Miktarı: ' || rec.StokMiktari);
11  END LOOP;
12 END;
13 /
14
```

Statement processed.

## Soru9)

Veritabanımızın 3. normal formda olduğunu göstermek için aşağıdaki durumları sağladığımızı varsayabiliriz:

1. Tablolar tek bir anahtar alanı (primary key) kullanarak birincil anahtarlarını belirlemiş durumda.
2. Her tablo, her bir alanın tek bir değeri temsil ettiği şekilde atomik alanlardan oluşuyor.
3. Verilerin tekrarlı kaydedilmesi veya bağımlılıkları önlemek için gereksiz alanlar yok.
4. Tablolar arasında ilişkiler, uygun şekilde foreign key kısıtlamalarıyla tanımlanmış durumda.

Bu varsayımlar altında, veritabanımızın 3. normal formda olduğunu söyleyebiliriz.

## Soru10)

CREATE OR REPLACE PROCEDURE SilTekrarEdenKayitlar(p\_tablo\_ad VARCHAR2, p\_alan\_ad VARCHAR2) IS

v\_sql VARCHAR2(1000);

v\_rowid ROWID;

v\_cursor SYS\_REFCURSOR;

BEGIN

v\_sql := 'SELECT MIN(rowid) AS min\_rowid FROM ' || p\_tablo\_ad || ' GROUP BY ' ||  
p\_alan\_ad || ' HAVING COUNT(\*) > 1';

OPEN v\_cursor FOR v\_sql;

LOOP

FETCH v\_cursor INTO v\_rowid;

EXIT WHEN v\_cursor%NOTFOUND;

v\_sql := 'DELETE FROM ' || p\_tablo\_ad || ' WHERE rowid NOT IN (SELECT :1 FROM DUAL)';

EXECUTE IMMEDIATE v\_sql USING v\_rowid;

END LOOP;

CLOSE v\_cursor;

COMMIT;

```

        DBMS_OUTPUT.PUT_LINE('Tekrar eden kayıtlar silindi.');
```

EXCEPTION

```

        WHEN OTHERS THEN

            DBMS_OUTPUT.PUT_LINE('Hata: ' || SQLERRM);
```

END SilTekrarEdenKayitlar;

/

-- Örnek Kullanım için

BEGIN

```

    SilTekrarEdenKayitlar('Ürünler', 'Ürün Adı');
```

-- Ürünler tablosunda Ürün Adı alanına göre tekrar eden kayıtları siler

-- Diğer tablolar için aynı şekilde kullanım yapabilirsiniz

END;

/

#### SQL Worksheet

```

1 CREATE OR REPLACE PROCEDURE SilTekrarEdenKayitlar(p_tablo_ad VARCHAR2, p_alan_ad VARCHAR2) IS
2     v_sql VARCHAR2(1000);
3     v_rowid ROWID;
4     v_cursor SYS_REFCURSOR;
5 BEGIN
6     v_sql := 'SELECT MIN(rowid) AS min_rowid FROM ' || p_tablo_ad || ' GROUP BY ' || p_alan_ad || ' HAVING COUNT(*) > 1';
7     OPEN v_cursor FOR v_sql;
8 LOOP
9     FETCH v_cursor INTO v_rowid;
10    EXIT WHEN v_cursor%NOTFOUND;
11    v_sql := 'DELETE FROM ' || p_tablo_ad || ' WHERE rowid NOT IN (SELECT :1 FROM DUAL)';
12    EXECUTE IMMEDIATE v_sql USING v_rowid;
13 END LOOP;
14 CLOSE v_cursor;
15
```

Procedure created.

Statement processed.