Veritabnı oluşturan öğrenciler :

215260613

225260611

Enes Attayr

Wael El Ahmed

Veritabnı Sistemi başlığı : Okul Otomasyonu

Veritabnı Sistemi Özeti :

Okul yönetimi için geliştirilen bu veritabanı, okulun temel bileşenlerini (öğrenciler, öğretmenler,

veliler, sınıflar, şubeler ...) ve bunlar arasındaki ilişkileri dijital ortamda tutmak ve yönetmek için

tasarlanmıştır. Bu veritabanı sayesinde okulda kayıtlı tüm öğrenciler, verilen dersler, sınavlar,

sınav notları, öğretmenler ve diğer yönetim bilgileri merkezi bir yapı altında toplanır. Bu yapı,

okul yönetimi ve akademik süreçlerin daha etkin bir şekilde yürütülmesini sağlar.

Veritabnı Sistemi Sağladığı Hizmetler : - Öğrenci Bilgilerinin Yönetimi : Öğrenci kayıtlarının yapılması, bilgilerinin güncellenmesi, sınav

notlarının saklanması ve sınıf atamalarının yapılması. - öğretmen ve Ders Yönetimi: Öğretmenlerin bilgilerinin tutulması, branşlarının ve verdikleri

derslerin kaydedilmesi. - not ve Sınav Takibi: Sınavların ve bu sınavlara ait notların kaydedilmesi, öğrencilerin sınav

performanslarının izlenmesi. - sınıf ve Şube Yönetimi: Sınıfların ve şubelerin oluşturulması, öğrencilerin ve öğretmenlerin

uygun sınıf ve şubelere atanması. - veli Bilgilerinin Saklanması: Velilerin bilgileri saklanarak gerektiğinde okul ile iletişim

kurabilmeleri sağlanır. - yönetici ve Şube Bilgileri: Yöneticilerin ve şubelerin detaylı bilgilerinin saklanması ve

düzenlenmesi. - -Raporlama: Farklı kriterlere göre raporlar oluşturabilme.

Veritabnı Sistemi Varlıkları :

öğrenci: Öğrenci kimlik bilgilerini ve öğrenciye ait veli, sınıf ve şube bilgilerini içerir.

öğretmen: Öğretmenin kimlik bilgilerini, branşını ve ders verdiği sınıf bilgilerini içerir.

yönetici: Okul yöneticisi kimlik bilgilerini ve sorumlu olduğu şube bilgilerini içerir.

ders: Her dersin adını, kredi bilgisini ve dersi veren öğretmeni içerir.

sınav: Her sınavın ders bilgisi ve sınav tarihi gibi bilgilerini saklar.

not: Öğrenci sınavlarından aldığı puan bilgilerini saklar.

sınıf: Sınıfların adı ve bağlı olduğu şube bilgisini içerir.

veli: Öğrencinin velisinin kimlik ve iletişim bilgilerini içerir.

şube: Şubelerin adı ve okul yeri gibi bilgilerini içerir.

Veri Yapıları :

öğrenci (Öğrenci id , Sınıf id , Veli id , Ad , Soyad , Doğum_Tarih , Adres , Telefon_No , Cinsiyet)

öğretmen (Öğretmen id , Ad , Soyad , Adres , Telefon_No , Branş)

yönetici (Yönetici id , Ad , Soyad , Adres , Telefon_No , Görev)

ders (Dres id , Öğretmen id , Ders_Adı , Kredi)

sınav (Sınav id , Ders id , Sınav_Türü)

not (Not id , Sınav id , Öğrenci id , Puan)

sınıf (Sınıf id , Şube id , Sınıf_Adı)

veli (Veli id , Ad , Soyad , Adres , Telefon_No)

şube (Şube id , Şube_Adı)

Öğrenci - Sınav (Öğrenci id , Sınav id , Tarih)

Sınıf - Ders (Sınıf id , Dres id)

Veritabnı Sistemi İlişkileri :

Öğrenci - Veli: " Velisi " bire-bir ilişki (Her öğrencinin bir velisi olabilir).

Öğrenci - Sınıf: " Kayıtlı Olduğu Sınıf " bire-çok ilişki (Bir sınıfta birden fazla öğrenci olabilir).

Öğrenci - Şube: " Bağlı Olduğu Şube " bire-çok ilişki (Her öğrenci bir şubeye bağlıdır).

Öğrenci - Sınav: " Katıldığı Sınavlar " çoktan-çoğa ilişki (Bir öğrenci birden fazla sınava girebilir, her sınavda birden fazla öğrenci olabilir).

Öğrenci - Not: "Aldığı Notlar " bire-çok ilişki (Bir öğrencinin birden fazla notu olabilir).

Öğretmen - Ders: " Vermekte Olduğu Dersler " bire-çok ilişki (Bir öğretmen birden fazla ders verebilir).

Öğretmen - Sınıf : "Ders Verdiği Sınıflar" bire-çok ilişki (Bir öğretmen birden fazla sınıfa ders verebilir).

Öğretmen - Şube: "Görev Yaptığı Şube" bire-çok ilişki (Bir öğretmen birden fazla şubede ders verebilir).

Yönetici - Şube: " Sorumlu Olduğu Şube " bire-çok ilişki (Her yönetici bir veya birden fazla şubeden sorumlu olabilir).

Ders - Sınav: " İlgili Sınavlar "bire-çok ilişki (Bir dersin birden fazla sınavı olabilir).

Ders - Sınıf: " Dersin Verildiği Sınıflar " çoktan-çoğa ilişki (Bir ders birden fazla sınıfta okutulabilir).

Sınav - Not: " Verilen Notlar " bire-çok ilişki (Her sınavın birden fazla not kaydı olabilir).

Sınıf - Şube: " Ait Olduğu Şube " bire-çok ilişki (Bir sınıf belirli bir şubeye aittir).

Tablolar ve Açıklamaları

veli tablosu :

```
CREATE TABLE veli (
veli_id INT PRIMARY KEY IDENTITY,
ad VARCHAR(50) NOT NULL,
    soyad VARCHAR(50) NOT NULL,
    adres VARCHAR(255),
    telefon_no VARCHAR(15)
```

);

açıklama : veli bilgilerini tutar. veli_id benzersiz veli kimliğidir. ad ve soyad zorunlu alanlardır,

adres ve telefon_no ise isteğe bağlıdır.

şube tablosu :

```
CREATE TABLE şube (
    şube_id INT PRIMARY KEY IDENTITY,
    şube_adı VARCHAR(50) NOT NULL
);
```

açıklama : okulun farklı şubelerini tutar. şube_id benzersiz şube kimliğidir. şube_adı zorunludur.

sınıf tablosu :

```
create table sınıf (
    sınıf_id INT PRIMARY KEY IDENTITY,
    şube_id INT NOT NULL,
    sınıf_adı VARCHAR(50),
    FOREIGN KEY (şube_id) REFERENCES şube(şube_id)
);
```

Açıklama : Sınıf bilgilerini tutar. sınıf_id benzersiz sınıf kimliğidir. şube_id şube tablosuna

referanstır ve zorunludur. sınıf_adı isteğe bağlıdır.

öğretmen tablosu :

```
CREATE TABLE öğretmen (
    öğretmen_id INT PRIMARY KEY IDENTITY,
    ad VARCHAR(50) NOT NULL,
    soyad VARCHAR(50) NOT NULL,
    adres VARCHAR(255),
```

```
    telefon_no VARCHAR(15),
    branş VARCHAR(50)
);
```

Açıklama : öğretmen bilgilerini tutar. öğretmen_id benzersiz öğretmen kimliğidir. ad ve soyad zorunludur. adres, telefon_no ve branş isteğe bağlıdır.

ders tablosu :

```
CREATE TABLE ders (
    ders_id INT PRIMARY KEY IDENTITY,
    öğretmen_id INT NOT NULL,
    ders_adı VARCHAR(100) NOT NULL,
    kredi INT NOT NULL,
    FOREIGN KEY (öğretmen_id) REFERENCES öğretmen(öğretmen_id)
);
```

Açıklama : ders bilgilerini tutar. ders_id benzersiz ders kimliğidir. öğretmen_id öğretmen tablosuna referanstır ve zorunludur. ders_adı ve kredi zorunludur.

yönetici tablosu :

```
CREATE TABLE yönetici (
    yönetici_id INT PRIMARY KEY IDENTITY,
    ad VARCHAR(50) NOT NULL,
    soyad VARCHAR(50) NOT NULL,
    adres VARCHAR(255),
    telefon_no VARCHAR(15),
    görev VARCHAR(50)
```

);

Açıklama : yönetici bilgilerini tutar. yönetici_id benzersiz yönetici kimliğidir. ad ve soyad zorunludur. adres, telefon_no ve görev isteğe bağlıdır.

öğrenci tablosu :

```
CREATE TABLE öğrenci (
    öğrenci_id INT PRIMARY KEY IDENTITY,
    sınıf_id INT NOT NULL,
    veli_id INT NOT NULL,
    ad VARCHAR(50) NOT NULL,
    soyad VARCHAR(50) NOT NULL,
    doğum_tarih DATE,
    adres VARCHAR(255),
    telefon_no VARCHAR(15),
  cinsiyet VARCHAR(10),
    FOREIGN KEY (sınıf_id) REFERENCES sınıf(sınıf_id),
    FOREIGN KEY (veli_id) REFERENCES veli(veli_id)
);
```

Açıklama : öğrenci bilgilerini tutar. öğrenci_id benzersiz öğrenci kimliğidir. sınıf_id ve veli_id zorunludur. ad, soyad, doğum_tarih, adres, telefon_no ve cinsiyet isteğe bağlıdır.

sınav tablosu :

```
CREATE TABLE sınav (
    sınav_id INT PRIMARY KEY IDENTITY,
    ders_id INT NOT NULL,
```

```
    sınav_türü VARCHAR(50),
    FOREIGN KEY (ders_id) REFERENCES ders(ders_id)
);
```

Açıklama : sınav bilgilerini tutar. sınav_id benzersiz sınav kimliğidir. ders_id ders tablosuna referanstır ve zorunludur. sınav_türü isteğe bağlıdır.

notlar tablosu :

```
CREATE TABLE notlar (
    not_id INT PRIMARY KEY IDENTITY,
    sınav_id INT NOT NULL,
    öğrenci_id INT NOT NULL,
    puan INT NOT NULL,
    FOREIGN KEY (sınav_id) REFERENCES sınav(sınav_id),
    FOREIGN KEY (öğrenci_id) REFERENCES öğrenci(öğrenci_id)
);
```

Açıklama: not bilgilerini tutar. not_id benzersiz not kimliğidir. sınav_id ve öğrenci_id zorunludur. puan zorunludur.

İlişkiler ve Açıklamaları

Öğrenci - Sınav İlişkisi Tablosu (Çoktan-Çoğa)

```
CREATE TABLE öğrenci_sınav (
    öğrenci_id INT NOT NULL,
    sınav_id INT NOT NULL,
    tarih DATE NOT NULL,
    PRIMARY KEY (öğrenci_id, sınav_id),
```

FOREIGN KEY (öğrenci_id) REFERENCES öğrenci(öğrenci_id),

FOREIGN KEY (sınav_id) REFERENCES sınav(sınav_id)

);

açıklama : öğrenci ve sınav ilişkilerini tutar. öğrenci_id ve sınav_id zorunludur. tarih sınav tarihini tutar ve zorunludur.

Sınıf - Ders İlişkisi Tablosu (Çoktan-Çoğa)

CREATE TABLE sınıf_ders (

sınıf_id INT NOT NULL,

ders_id INT NOT NULL,

PRIMARY KEY (sınıf_id, ders_id),

FOREIGN KEY (sınıf_id) REFERENCES sınıf(sınıf_id),

FOREIGN KEY (ders_id) REFERENCES ders(ders_id)

);

açıklama : sınıf ve ders ilişkilerini tutar. sınıf_id ve ders_id zorunludur.

Örnek Veriler ve Açıklamaları

Veli Tablosu (20 Örnek Veri)

INSERT INTO veli (ad, soyad, adres, telefon_no) VALUES

('Ali', 'Kaya', 'Ankara', '05011111111'),

('Ayşe', 'Yılmaz', 'Istanbul', '05022222222'),

('Fatma', 'Demir', 'Izmir', '05033333333'),

('Mehmet', 'Çelik', 'Bursa', '05044444444'),

('Ahmet', 'Arslan', 'Antalya', '05055555555'),

('Hasan', 'Güneş', 'Mersin', '05066666666'),

('Zeynep', 'Şahin', 'Adana', '05077777777'),

('Elif', 'Koç', 'Konya', '05088888888'),

('Hüseyin', 'Bulut', 'Gaziantep', '05099999999'),

('Cem', 'Doğan', 'Diyarbakır', '05100000000'),

('Selin', 'Aksoy', 'Eskişehir', '05111111111'),

('Kemal', 'Polat', 'Kayseri', '05122222222'),

('Leyla', 'Özkan', 'Trabzon', '05133333333'),

('Burak', 'Çakır', 'Samsun', '05144444444'),

('Derya', 'Kılıç', 'Manisa', '05155555555'),

('Ebru', 'Şimşek', 'Erzurum', '05166666666'),

('Ferhat', 'Yıldırım', 'Van', '05177777777'),

('Buse', 'Erdoğan', 'Kocaeli', '05188888888'),

('Efe', 'Aydın', 'Aydın', '05199999999'),

('Naz', 'Kurt', 'Malatya', '05200000000');

Açıklama: Veli tablosuna 20 örnek veri eklenmiştir. Her bir veri, bir velinin ad, soyad, adres ve telefon_no bilgilerini içerir.

Şube Tablosu (5 Örnek Veri)

INSERT INTO şube (şube_adı) VALUES

('A Şubesi'),

('B Şubesi'),

('C Şubesi'),

('D Şubesi'),

('E Şubesi');

Açıklama: Şube tablosuna 5 örnek veri eklenmiştir. Her bir veri, bir şubenin şube_adı bilgisini içerir.

sınıf tablosu (10 Örnek Veri)

INSERT INTO sınıf (şube_id, sınıf_adı) VALUES

(1, '1-A'),

(1, '2-A'),

(2, '1-B'),

(2, '2-B'),

(3, '1-C'),

(3, '2-C'),

(4, '1-D'),

(4, '2-D'),

(5, '1-E'),

(5, '2-E');

Açıklama: Sınıf tablosuna 10 örnek veri eklenmiştir. Her bir veri, bir sınıfın şube_id ve sınıf_adı bilgilerini içerir.

öğretmen tablosu (10 Örnek Veri)

INSERT INTO öğretmen (ad, soyad, adres, telefon_no, branş) VALUES

('Ahmet', 'Yıldız', 'Ankara', '05311111111', 'Matematik'),

('Ayşe', 'Demir', 'Istanbul', '05322222222', 'Fizik'),

('Fatma', 'Kaya', 'Izmir', '05333333333', 'Kimya'),

('Mehmet', 'Güneş', 'Bursa', '05344444444', 'Biyoloji'),

('Hasan', 'Çelik', 'Antalya', '05355555555', 'Tarih'),

('Zeynep', 'Koç', 'Mersin', '05366666666', 'Coğrafya'),

('Hüseyin', 'Bulut', 'Adana', '05377777777', 'Türkçe'),

('Elif', 'Doğan', 'Konya', '05388888888', 'İngilizce'),

('Cem', 'Arslan', 'Gaziantep', '05399999999', 'Felsefe'),

('Selin', 'Polat', 'Diyarbakır', '05400000000', 'Edebiyat');

Açıklama : Öğretmen tablosuna 10 örnek veri eklenmiştir. Her bir veri, bir öğretmenin ad,

soyad, adres, telefon_no ve branş bilgilerini içerir.

ders tablosu (10 Örnek Veri)

INSERT INTO ders (öğretmen_id, ders_adı, kredi) VALUES

(1, 'Matematik 101', 3),

(2, 'Fizik 101', 4),

(3, 'Kimya 101', 3),

(4, 'Biyoloji 101', 2),

(5, 'Tarih 101', 3),

(6, 'Coğrafya 101', 2),

(7, 'Türkçe 101', 3),

(8, 'İngilizce 101', 3),

(9, 'Felsefe 101', 2),

(10, 'Edebiyat 101', 3);

Açıklama: Ders tablosuna 10 örnek veri eklenmiştir. Her bir veri, bir dersin öğretmen_id,

ders_adı ve kredi bilgilerini içerir.

yönetici tablosu (5 Örnek Veri)

INSERT INTO yönetici (ad, soyad, adres, telefon_no, görev) VALUES

('Ali', 'Yılmaz', 'Ankara', '05411111111', 'Müdür'),

('Ayşe', 'Kaya', 'Istanbul', '05422222222', 'Müdür Yardımcısı'),

('Fatma', 'Demir', 'Izmir', '05433333333', 'Sekreter'),

('Mehmet', 'Çelik', 'Bursa', '05444444444', 'Muhasebeci'),

('Ahmet', 'Arslan', 'Antalya', '05455555555', 'Teknik Destek');

Açıklama: Yönetici tablosuna 5 örnek veri eklenmiştir. Her bir veri, bir yöneticinin ad, soyad,

adres, telefon_no ve görev bilgilerini içerir.

öğrenci tablosu (20 Örnek Veri)

INSERT INTO öğrenci (sınıf_id, veli_id, ad, soyad, doğum_tarih, adres,

telefon_no, cinsiyet) VALUES

(1, 1, 'Veli', 'Kaya', '2010-05-12', 'Ankara', '05511111111', 'Erkek'),

(2, 2, 'Can', 'Yılmaz', '2011-07-23', 'Istanbul', '05522222222', 'Erkek'),

(3, 3, 'Melis', 'Demir', '2012-08-15', 'Izmir', '05533333333', 'Kız'),

(4, 4, 'Eren', 'Çelik', '2013-10-09', 'Bursa', '05544444444', 'Erkek'),

(5, 5, 'Bora', 'Arslan', '2010-11-17', 'Antalya', '05555555555', 'Erkek'),

(6, 6, 'Cenk', 'Güneş', '2011-03-21', 'Mersin', '05566666666', 'Erkek'),

(7, 7, 'Leyla', 'Şahin', '2010-12-12', 'Adana', '05577777777', 'Kız'),

(8, 8, 'Deniz', 'Koç', '2012-09-14', 'Konya', '05588888888', 'Erkek'),

(9, 9, 'Mert', 'Bulut', '2011-01-10', 'Gaziantep', '05599999999', 'Erkek'),

(10, 10, 'Zehra', 'Doğan', '2010-04-19', 'Diyarbakır', '05600000000', 'Kız'),

(1, 11, 'Ezgi', 'Aksoy', '2011-06-30', 'Eskişehir', '05611111111', 'Kız'),

(2, 12, 'Kerem', 'Polat', '2013-02-15', 'Kayseri', '05622222222', 'Erkek'),

(3, 13, 'Aylin', 'Özkan', '2012-08-25', 'Trabzon', '05633333333', 'Kız'),

(4, 14, 'Barış', 'Çakır', '2011-11-01', 'Samsun', '05644444444', 'Erkek'),

(5, 15, 'Dilan', 'Kılıç', '2010-09-09', 'Manisa', '05655555555', 'Kız'),

(6, 16, 'Esra', 'Şimşek', '2013-03-10', 'Erzurum', '05666666666', 'Kız'),

(7, 17, 'Musa', 'Yıldırım', '2012-07-14', 'Van', '05677777777', 'Erkek'),

(8, 18, 'Irem', 'Erdoğan', '2011-05-20', 'Kocaeli', '05688888888', 'Kız'),

(9, 19, 'Alp', 'Aydın', '2010-12-08', 'Aydın', '05699999999', 'Erkek'),

(10, 20, 'Selin', 'Kurt', '2013-10-29', 'Malatya', '05700000000', 'Kız');

açıklama : öğrenci tablosuna 20 örnek veri eklenmiştir. Her bir veri, bir öğrencinin sınıf_id,

veli_id, ad, soyad, doğum_tarih, adres, telefon_no ve cinsiyet bilgilerini içerir.

sınav tablosu (10 Örnek Veri)

INSERT INTO sınav (ders_id, sınav_türü) VALUES

(1, 'Yazılı'),

(2, 'Sözlü'),

(3, 'Quiz'),

(4, 'Yazılı'),

(5, 'Sözlü'),

(6, 'Quiz'),

(7, 'Yazılı'),

(8, 'Sözlü'),

(9, 'Quiz'),

(10, 'Yazılı');

Açıklama: Sınav tablosuna 10 örnek veri eklenmiştir. Her bir veri, bir sınavın ders_id ve sınav_türü bilgilerini içerir.

notlar tablosu (20 Örnek Veri)

INSERT INTO notlar (sınav_id, öğrenci_id, puan) VALUES

(1, 1, 85),

(2, 2, 90),

(3, 3, 78),

(4, 4, 88),

(5, 5, 92),

(6, 6, 76),

(7, 7, 84),

(8, 8, 89),

(9, 9, 91),

(10, 10, 87),

(1, 11, 80),

(2, 12, 95),

(3, 13, 82),

(4, 14, 77),

(5, 15, 88),

(6, 16, 93),

(7, 17, 79),

(8, 18, 85),

(9, 19, 94),

(10, 20, 86);

Açıklama: Notlar tablosuna 20 örnek veri eklenmiştir. Her bir veri, bir öğrencinin sınav_id,

öğrenci_id ve puan bilgilerini içerir.

öğrenci - sınav ilişkisi tabosu (20 Örnek Veri)

INSERT INTO öğrenci_sınav (öğrenci_id, sınav_id, tarih) VALUES

(1, 1, '2024-01-01'),

(2, 2, '2024-01-02'),

(3, 3, '2024-01-03'),

(4, 4, '2024-01-04'),

(5, 5, '2024-01-05'),

(6, 6, '2024-01-06'),

(7, 7, '2024-01-07'),

(8, 8, '2024-01-08'),

(9, 9, '2024-01-09'),

(10, 10, '2024-01-10'),

(11, 1, '2024-01-11'),

(12, 2, '2024-01-12'),

(13, 3, '2024-01-13'),

(14, 4, '2024-01-14'),

(15, 5, '2024-01-15'),

(16, 6, '2024-01-16'),

(17, 7, '2024-01-17'),

(18, 8, '2024-01-18'),

(19, 9, '2024-01-19'),

(20, 10, '2024-01-20');

Açıklama: Öğrenci - sınav ilişkisi tablosuna 20 örnek veri eklenmiştir. Her bir veri, bir öğrenci_id, sınav_id ve tarih bilgilerini içerir.

Sınıf - Ders İlişkisi Tablosu (10 Örnek Veri)

sınıf- ders ilişkisi tabosu (10 Örnek Veri)

INSERT INTO sınıf_ders (sınıf_id, ders_id) VALUES

(1, 1),

(2, 2),

(3, 3),

(4, 4),

(5, 5),

(6, 6),

(7, 7),

(8, 8),

(9, 9),

(10, 10);

Açıklama: Sınıf - ders ilişkisi tablosuna 10 örnek veri eklenmiştir. Her bir veri, bir sınıf_id ve

ders_id bilgilerini içerir.

Ek Kodlar ve Açıklamaları

saklı yordam oluştutrma

Stored Procedure: öğrenci Notu Hesaplama

```
CREATE PROCEDURE Hesapla_Ogrenci_Notu
    @DersID INT
AS
BEGIN
    -- Belirli bir dersteki tüm öğrencilerin final notunu hesaplar
    DECLARE @NotToplam INT = 0;
    DECLARE @OgrenciSayisi INT = 0;

    -- Belirtilen dersteki öğrencilerin puanlarını toplar
    SELECT @NotToplam = SUM(puan), @OgrenciSayisi = COUNT(*)
    FROM notlar
    INNER JOIN sınav ON notlar.sınav_id = sınav.sınav_id
    WHERE sınav.ders_id = @DersID;

    -- Eğer derste öğrenci varsa, ortalama hesaplanır
    IF @OgrenciSayisi > 0
    BEGIN
        PRINT 'Dersin ortalaması: ' + CAST(@NotToplam / @OgrenciSayisi AS
VARCHAR(10));
    END
```

```
    ELSE
    BEGIN
        PRINT 'Bu derste hiç öğrenci yok!';
    END
END;
```

Açıklama: Bu stored procedure, belirli bir ders için öğrencilerin not ortalamasını hesaplar ve çıktısını verir. @DersID parametresi, hangi dersin notlarının hesaplanacağını belirtir.

trigger oluştutrma

Trigger: Not 50'nin Altındaysa Güncelleme

```
CREATE TRIGGER NotAltiElliTrigger
ON notlar
AFTER INSERT
AS
BEGIN --    öğrencinin notu 50'den düşükse kaydını güncelle
    DECLARE @OgrenciID INT;
    DECLARE @Puan INT;

    SELECT @OgrenciID = öğrenci_id, @Puan = puan FROM inserted;

    IF @Puan < 50
    BEGIN
        UPDATE öğrenci
        SET adres = 'Bu öğrenci başarısız'
        WHERE öğrenci_id = @OgrenciID;
```

```
    END
END;
```

Açıklama: Bu trigger, notlar tablosuna yeni bir not eklendiğinde tetiklenir. Eğer eklenen not 50'nin altındaysa, ilgili öğrencinin adres alanı 'Bu öğrenci başarısız' olarak güncellenir.

Transaction: Öğrenci ve Not Eklemek

```
BEGIN TRANSACTION;

BEGIN TRY
    -- Öğrenci tablosuna yeni öğrenci ekleniyor
    INSERT INTO öğrenci (sınıf_id, veli_id, ad, soyad, doğum_tarih, adres,
telefon_no, cinsiyet)
    VALUES (1, 1, 'Mehmet', 'Öztürk', '2014-05-15', 'Istanbul', '05712345678',
'Erkek');

    -- Yeni öğrenci için notlar tablosuna puan ekleniyor
    DECLARE @OgrenciID INT = SCOPE_IDENTITY();     -- Yeni öğrencinin ID'si alınıyor
    INSERT INTO notlar (sınav_id, öğrenci_id, puan)
    VALUES (1, @OgrenciID, 85);

    -- Eğer burada hata olmazsa, işlem onaylanır
    COMMIT;
END TRY
BEGIN CATCH -- Bir hata oluşursa, yapılan değişiklikler geri alınır
ROLLBACK; -- Hata mesajı gösterilir
```

```
PRINT 'Bir hata oluştu: ' + ERROR_MESSAGE();
END CATCH;
```

Açıklama: Bu transaction, bir öğrenci ve onun notunu eklemeyi sağlar. Eğer işlem sırasında bir hata oluşursa, transaction geri alınır (rollback edilir).

Rapor Özeti :

Okul Otomasyonu projesi, eğitim kurumlarının öğrenci, veli, öğretmen ve diğer idari bilgilerini etkin bir şekilde yönetmelerine olanak tanır. Veritabanı yapısı ve ilişkileri, kullanıcıların verileri hızlı ve kolay bir şekilde yönetmesini sağlar. Ek olarak, stored procedures ve triggers gibi özellikler, veritabanı işlemlerini otomatikleştirir ve daha güvenilir hale getirir. Bu sistem, eğitim yönetiminde büyük kolaylıklar sağlayacaktır.