T.C
FIRAT ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Müzik Sosyal Medya Uygulaması Veri Tabanı Sistemi

Final Raporu

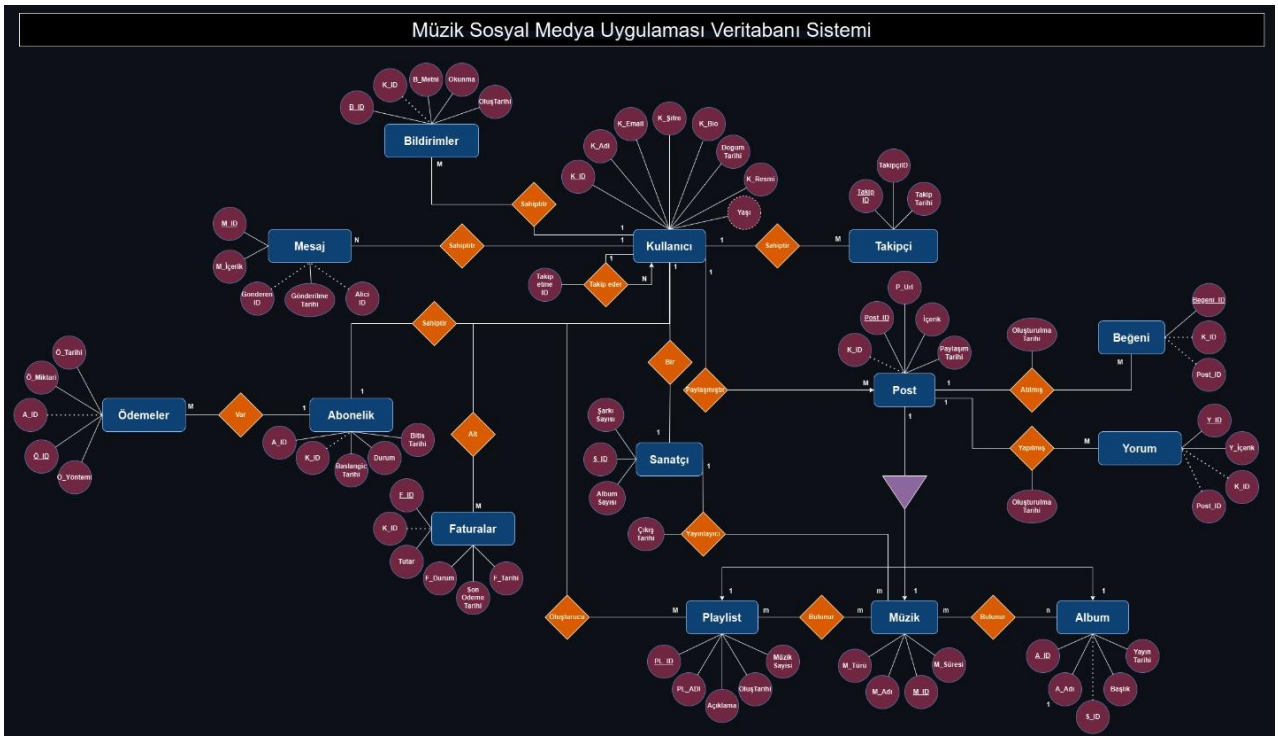
AKİF KARACA 220260608
SEYİT İLHAM FIRTINA 220260140
HALİL İBRAHİM TURAN 220260044

1. Proje Konusu

Bu projede, "**müzik odaklı bir sosyal medya platformunun"** veri tabanı yönetim sistemi geliştirilecektir. Sistem, kullanıcıların şarkı ve albüm paylaşmalarına, müzik listeleri oluşturmalarına ve diğer kullanıcılarla etkileşimde bulunmalarına olanak tanırken, sanatçıların eserlerini tanıtmalarını ve kullanıcılarla bağ kurmalarını sağlar. Ayrıca, kullanıcıların beğeni, yorum, takip gibi sosyal medya aktivitelerini, aboneliklerini ve ödemelerini yönetmesine imkân tanır. Veri tabanı tasarımı, verilerin güvenilir ve etkili bir şekilde saklanmasını, güncellenmesini ve sorgulanmasını sağlayacak şekilde optimize edilmiştir.

2. E-R Diyagramı

Projenin E-R diyagramı:



3. İlişki şeması oluşturma aşaması:

- **Kullanıcı** (K_ID, K_AD, K_Email, K_Şifre, K_Bio, DogumTarihi, Yasi, K_Resmi)
- **PlayList** (PL_ID, Post_ID , PL_ADI, Aciklama, OlusTarihi, MuzikSayisi, OlusturanKullanici)
- **Paylaşım** (Post_ID, P_Url, PaylasimTarihi, K_ID, Icerik)
- **Faturalar** (F_ID, K_ID, Tutar, F_Durum, SonÖdemeTarihi, FaturaTarihi)
- **Mesaj** (M_ID, M_İçerik, GonderenID, AliciID, GönderilmeTarihi)
- **Abonelik** (Ab_ID, K_ID, BaşlangıçTarihi, Durum, BitişTarihi)
- **Ödemeler** (Ö_ID, Ö_Tarihi, Ö_Miktarı, A_ID, Ö_Yöntemi)
- **Bildirimler** (B_ID, K_ID, B_Metni, Okunma, OluşTarihi)

- **Takipçi** (TakipID, TakipciID, TakipEdilenID, TakipTarihi)
- **Yorum** (Y_ID, K_ID, Y_Icerik, YorumTarihi, Post_ID)
- **Müzik** (M_ID, Post_ID ,CikisTarihi, M_Turu, M_Suresi, S_ID)
- **Beğeni** (Begeni_ID, K_ID, Post_ID, BegeniTarihi)
- **Albüm** (A_ID, Post_ID , A_Adi, S_ID, Baslik, YayinTarihi)
- **Sanatçı** (S_ID, AlbumSayısı, ŞarkıSayısı, K_ID)
- **Playlist_Muzik** (PL_ID, M_ID)
- **Album_Muzik** (A_ID, M_ID)

4. Tablo Güncelleme ve Normalizayon işlemi:

1.Kullanıcı (User) Normalizasyonu süreci:
Analiz:

- **"Yasi"** transitif olarak "**DogumTarihi"** üzerinden "**K_ID'ye"** bağımlıdır.
- Bu nedenle tablo "**1 NF'TİR**. **2 NF"** değildir.

Çözüm:

- **Transitif** bağımlılığı ortadan kaldırmak için "**Yasi"** bağımsız bir tabloya taşınır.
  Kullanıcı (K_ID, K_AD, K_Email, K_Şifre, K_Bio, DogumTarihi, K_Resmi)
  Kullanıcı_Yaş (K_ID, Yasi)
- Tablo **BCNF** olmuştur.

Diğer Tablolar BCNF Formundadır.

4.1. Normalizasyon işlemi sonrası İlişki şeması oluşturma aşaması:

- **Kullanıcı** (K_ID, K_AD, K_Email, K_Şifre, K_Bio, DogumTarihi, K_Resmi)
- **Kullanıcı_Yaş** (K_ID, Yasi)
- **Bildirimler** (B_ID, K_ID, B_Metni, Okunma, OluşTarihi)
- **Mesaj** (M_ID, M_İçerik, GonderenID, AliciID, GönderilmeTarihi)
- **Sanatçı** (S_ID, AlbumSayısı, ŞarkıSayısı, K_ID)
- **Paylaşım** (Post_ID, P_Url, PaylasimTarihi, K_ID, Icerik)
- **Beğeni** (Begeni_ID, K_ID, Post_ID, BegeniTarihi)
- **Yorum** (Y_ID, K_ID, Y_Icerik, YorumTarihi, Post_ID)
- **PlayList** (PL_ID, Post_ID, PL_ADI, Aciklama, OlusTarihi, OlusturanKullanici)
- **Playlist_Music** (PL_ID, M_ID)
- **Albüm** (A_ID, Post_ID, A_Adi, S_ID, Baslik, YayinTarihi)
- **Album_Muzik** (A_ID, M_ID)
- **Müzik** (M_ID, Post_ID, CikisTarihi, M_Turu, M_Suresi, S_ID)
- **Abonelik** (AB_ID, K_ID, BaşlangıçTarihi, Durum, BitişTarihi)
- **Ödemeler** (Ö_ID, Ö_Tarihi, Ö_Miktarı, AB_ID, Ö_Yöntemi)

- **Faturalar** (F_ID, K_ID, Tutar, F_Durum, SonÖdemeTarihi, FaturaTarihi)
- **Takipçi** (TakipID, TakipciID, TakipEdilenID, TakipTarihi)

5. Veri tabanı Tabloları Oluşturma Aşaması:

"muzikuygulamasi" adlı veri tabanının tasarım adımlarını ve her bir tablonun yapısını detaylandırmaktadır.

1.Veritabanı Oluşturma:

```
CREATE DATABASE muzikuygulamasi;
```

2.Tabloları Oluşturma:

Kullanıcı Tablosu:

```
CREATE TABLE Kullanıcı (
    K_ID INT PRIMARY KEY IDENTITY(1,1),
    K_AD NVARCHAR(100) NOT NULL,
    K_Email NVARCHAR(255) UNIQUE NOT NULL,
    K_Şifre NVARCHAR(255) NOT NULL,
    K_Bio NVARCHAR(MAX),
    DogumTarihi DATE NOT NULL,
    K_Resmi NVARCHAR(MAX)
);
```

Kullanıcı Yaş Tablosu:

```
CREATE TABLE Kullanıcı_Yaş (
    K_ID INT PRIMARY KEY,
    Yasi INT NOT NULL,
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID)
);
```

Bildirimler Tablosu:

```
CREATE TABLE Bildirimler (
    B_ID INT PRIMARY KEY,
    K_ID INT NOT NULL,
    B_Metni NVARCHAR(MAX),
    Okunma BIT NOT NULL,
    OluşTarihi DATETIME NOT NULL,
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID)
);
```

Mesajlar Tablosu:

```
CREATE TABLE Mesaj (
    M_ID INT PRIMARY KEY,
    M_İçerik NVARCHAR(MAX),
    GonderenID INT NOT NULL,
    AliciID INT NOT NULL,
    GönderilmeTarihi DATETIME NOT NULL,
    FOREIGN KEY (GonderenID) REFERENCES Kullanıcı(K_ID),
    FOREIGN KEY (AliciID) REFERENCES Kullanıcı(K_ID)
);
```

Sanatçı Tablosu:

```
CREATE TABLE Sanatçı (
    S_ID INT PRIMARY KEY,
    AlbumSayısı INT NOT NULL,
    ŞarkıSayısı INT NOT NULL,
    K_ID INT NOT NULL,
```

```
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID)
);
```

Paylaşım Tablosu:

```
CREATE TABLE Paylaşım (
    Post_ID INT PRIMARY KEY,
    P_Url NVARCHAR(MAX),
    PaylasimTarihi DATETIME NOT NULL,
    K_ID INT NOT NULL,
    Icerik NVARCHAR(MAX),
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID)
);
```

Beğeni Tablosu:

```
CREATE TABLE Beğeni (
    Begeni_ID INT PRIMARY KEY,
    K_ID INT NOT NULL,
    Post_ID INT NOT NULL,
    BegeniTarihi DATETIME NOT NULL,
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID),
    FOREIGN KEY (Post_ID) REFERENCES Paylaşım(Post_ID)
);
```

Yorum Tablosu:

```
CREATE TABLE Yorum (
    Y_ID INT PRIMARY KEY,
    K_ID INT NOT NULL,
    Y_Icerik NVARCHAR(MAX),
    YorumTarihi DATETIME NOT NULL,
    Post_ID INT NOT NULL,
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID),
    FOREIGN KEY (Post_ID) REFERENCES Paylaşım(Post_ID)
);
```

Playlist Tablosu:

```
CREATE TABLE Çalma_Listesi (
    PL_ID INT PRIMARY KEY,
    Post_ID INT NOT NULL,
    PL_ADI NVARCHAR(100) NOT NULL,
    Aciklama NVARCHAR(MAX),
    OlusTarihi DATETIME NOT NULL,
    OlusturanKullanici INT NOT NULL,
    FOREIGN KEY (OlusturanKullanici) REFERENCES Kullanıcı(K_ID),
    FOREIGN KEY (Post_ID) REFERENCES Paylaşım(Post_ID)
);
```

Playlist_Music Tablosu:

```
CREATE TABLE Playlist_Music (
    PL_ID INT NOT NULL,
    M_ID INT NOT NULL,
    PRIMARY KEY (PL_ID, M_ID),
    FOREIGN KEY (PL_ID) REFERENCES Çalma_Listesi(PL_ID),
    FOREIGN KEY (M_ID) REFERENCES Müzik(M_ID)
);
```

Albüm Tablosu:

```
CREATE TABLE Albüm (
    A_ID INT PRIMARY KEY,
    Post_ID INT NOT NULL,
    A_Adi NVARCHAR(100) NOT NULL,
    S_ID INT NOT NULL,
    Baslik NVARCHAR(MAX),
    YayinTarihi DATE NOT NULL,
    FOREIGN KEY (S_ID) REFERENCES Sanatçı(S_ID),
    FOREIGN KEY (Post_ID) REFERENCES Paylaşım(Post_ID)
```

```
);
```

Albüm_Music Tablosu:

```
CREATE TABLE Album_Muzik (
    A_ID INT NOT NULL,
    M_ID INT NOT NULL,
    PRIMARY KEY (A_ID, M_ID),
    FOREIGN KEY (A_ID) REFERENCES Albüm(A_ID),
    FOREIGN KEY (M_ID) REFERENCES Müzik(M_ID)
);
```

Abonelik Tablosu:

```
CREATE TABLE Abonelik (
    AB_ID INT PRIMARY KEY,
    K_ID INT NOT NULL,
    BaşlangıçTarihi DATE NOT NULL,
    Durum NVARCHAR(50) NOT NULL,
    BitişTarihi DATE,
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID)
);
```

Ödemeler Tablosu:

```
CREATE TABLE Ödemeler (
    Ö_ID INT PRIMARY KEY,
    Ö_Tarihi DATETIME NOT NULL,
    Ö_Miktarı DECIMAL(10, 2) NOT NULL,
    AB_ID INT NOT NULL,
    Ö_Yöntemi NVARCHAR(50) NOT NULL,
    FOREIGN KEY (AB_ID) REFERENCES Abonelik(AB_ID)
);
```

Faturalar Tablosu:

```
CREATE TABLE Faturalar (
    F_ID INT PRIMARY KEY,
    K_ID INT NOT NULL,
    Tutar DECIMAL(10, 2) NOT NULL,
    F_Durum NVARCHAR(50) NOT NULL,
    SonÖdemeTarihi DATE NOT NULL,
    FaturaTarihi DATE NOT NULL,
    FOREIGN KEY (K_ID) REFERENCES Kullanıcı(K_ID)
);
```

Takipçi Tablosu:

```
CREATE TABLE Takipçi (
    TakipID INT PRIMARY KEY,
    TakipciID INT NOT NULL,
    TakipEdilenID INT NOT NULL,
    TakipTarihi DATETIME NOT NULL,
    FOREIGN KEY (TakipciID) REFERENCES Kullanıcı(K_ID),
    FOREIGN KEY (TakipEdilenID) REFERENCES Kullanıcı(K_ID)
);
```

Müzik Tablosu:

```
CREATE TABLE Müzik (
    M_ID INT PRIMARY KEY,
    Post_ID INT NOT NULL,
    CikisTarihi DATE NOT NULL,
    M_Turu NVARCHAR(50),
    M_Suresi INT NOT NULL,
    S_ID INT NOT NULL,
    FOREIGN KEY (S_ID) REFERENCES Sanatçı(S_ID),
    FOREIGN KEY (Post_ID) REFERENCES Paylaşım(Post_ID),
);
```

6. Veri tabanındaki tabloların içeriklerini doldurma sürecini detayları:

Kullanıcı Tablosu Doldurma :

```
INSERT INTO Kullanıcı (K_AD, K_Email, K_Şifre, K_Bio, DogumTarihi, K_Resmi)
VALUES
('Halil İbrahim Turan', 'halilturan2023@gmail.com', 'Halil123', 'Müzik sever.', '2003-09-03', NULL),
('Seyit İlham Fırtına', 'seyit@gmail.com', 'Seyit12345', 'Sanatçı biyografisi yazarı.', '1995-05-15', NULL),
('Akif Karaca', 'sorest@gmail.com', 'Akif2021', 'Pop müzik aşığı.', '2003-01-24', NULL),
...
('Leyla Akın', 'leyla@gmail.com', 'Leyla789!', 'Müzik platformu tasarımcısı.', '1994-04-04', NULL);
```

	K_ID	K_AD	K_Email	K_Şifre	K_Bio	DogumTarihi	K_Resmi
1	1	Halil İbrahim Turan	halilturan2023@gmail.com	Halil123	Müzik sever.	2003-09-03	NULL
2	2	Seyit İlham Fırtına	seyit1953@gmail.com	Seyit12345	Sanatçı biyografisi yazarı.	2002-08-13	NULL
3	3	Akif Karaca	akifkaraca1989@gmail.com	Akif2021	Pop müzik aşığı.	2003-01-23	NULL
4	4	Mehmet Kaya	mehmet@gmail.com	Mehmet2021	Rock müzik aşığı.	2001-03-22	NULL
5	5	Kerem Can	kerem@gmail.com	Kerem1234!	DJ ve prodüktör.	1994-11-11	NULL
6	6	Fatma Er	fatma@gmail.com	Fatma2022!	Müzik bloggeri.	2000-06-18	NULL
7	7	Hakan Uslu	hakan@gmail.com	Hakan!321	Jazz hayranı.	1998-02-14	NULL
8	8	Elif Çelik	elif@gmail.com	Elif456!	Albüm koleksiyoncusu.	1989-09-30	NULL
9	9	Yusuf Şahin	yusuf@gmail.com	Yusuf1987	Metal müzik hayranı.	1987-12-25	NULL
10	10	Leyla Akın	leyla@gmail.com	Leyla789!	Müzik platformu tasarımcısı.	1996-04-04	NULL

Bildirimler Tablosu Doldurma :

```
INSERT INTO Bildirimler (B_ID, K_ID, B_Metni, Okunma, OluşTarihi)
VALUES
(1, 1, 'Yeni takipçiniz var!', 0, '2025-01-01'),
(2, 2, 'Paylaşımınız beğenildi.', 1, '2025-01-02'),
(3, 3, 'Yeni bir mesaj aldınız.', 0, '2025-01-03'),
...
(10, 10, 'Yeni bir takipçiniz var!', 1, '2025-01-10');
```

Sorgu:

```
Select* From Bildirimler;
```

	B_ID	K_ID	B_Metni	Okunma	OluşTarihi
1	1	1	Yeni takipçiniz var!	0	2025-01-01
2	2	2	Paylaşımınız beğenildi.	1	2025-01-02
3	3	3	Yeni bir mesaj aldınız.	0	2025-01-03
4	4	4	Albümünüz beğenildi.	1	2025-01-04
5	5	5	Yeni bir yorum yapıldı.	0	2025-01-05
6	6	6	Çalma listenize müzik eklendi.	1	2025-01-06
7	7	7	Sanatçı takibiniz güncellendi.	0	2025-01-07
8	8	8	Takipçiniz bir paylaşım yaptı.	1	2025-01-08
9	9	9	Albümünüz popüler oluyor!	0	2025-01-09
10	10	10	Yeni bir takipçiniz var!	1	2025-01-10

Mesaj Tablosu Doldurma :

```
INSERT INTO Mesaj (M_ID, M_İçerik, GonderenID, AliciID, GönderilmeTarihi)
VALUES
(1, 'Haydaa bu albüm gerçekten iyiymiş.', 2, 1, '2025-01-01 10:00:00'),
(2, 'Teşekkürler!', 1, 2, '2025-01-01 10:05:00'),
(3, 'Yeni şarkılarınızı bekliyoruz.', 3, 5, '2025-01-02 12:00:00'),
```

```
...
(10, 'Elbette, hemen paylaşıyorum.', 4, 10, '2025-01-08 12:00:00');
```

	M_ID	M_İçerik	GonderenID	AliciID	GönderilmeTarihi
1	1	Haydaa bu albüm gerçekten iyiymiş.	2	1	2025-01-01 10:00:00.000
2	2	Teşekkürler!	1	2	2025-01-01 10:05:00.000
3	3	Yeni şarkılarınızı bekliyoruz.	3	5	2025-01-02 12:00:00.000
4	4	Pop müzik koleksiyonunuz 10 numara.	4	8	2025-01-03 14:30:00.000
5	5	Rap listesi önerisi var mı?	7	6	2025-01-04 16:45:00.000
6	6	Bu albüm gerçekten özel!	5	3	2025-01-05 18:00:00.000
7	7	Takip için teşekkürler.	8	7	2025-01-06 20:15:00.000
8	8	Yorumlarınızı bekliyoruz.	9	10	2025-01-07 09:30:00.000
9	9	Çalma listesi paylaşabilir misiniz?	10	4	2025-01-08 11:45:00.000
10	10	Elbette, hemen paylaşıyorum.	4	10	2025-01-08 12:00:00.000

Takipçi Tablosu Doldurma :

```
INSERT INTO Takipçi (TakipID, TakipciID, TakipEdilenID, TakipTarihi)
VALUES
(1, 1, 2, '2025-01-01'),
(2, 3, 1, '2025-01-02'),
(3, 4, 3, '2025-01-03'),
...
(10, 2, 10, '2025-01-10');
```

Bu sorgu, **"Takipçi"** tablosunda yer alan tüm sütunları seçer ve sadece **"TakipTarihi"** sütunundaki değeri **'2025-01-01'** ile **'2025-01-05'** (dahil) arasında olan kayıtları getirir.

```
Sorgu:
Select* From Takipçi Where TakipTarihi BETWEEN '2025-01-01' AND '2025-01-05';
```

	TakipID	TakipciID	TakipEdilenID	TakipTarihi
1	1	1	2	2025-01-01
2	2	3	1	2025-01-02
3	3	4	3	2025-01-03
4	4	5	4	2025-01-04
5	5	6	5	2025-01-05

Paylaşım Tablosu Doldurma :

```
INSERT INTO Paylaşım (Post_ID, P_Url, PaylasimTarihi, K_ID, Icerik)
VALUES
(1, 'https://musisocial.com/post1', '2025-01-01 09:00:00', 1, 'En sevdiğim şarkı.'),
(2, 'https://musisocial.com/post2', '2025-01-02 10:00:00', 2, 'Yeni albümüm
yayında!'),
(3, 'https://musisocial.com/post3', '2025-01-03 11:00:00', 3, 'Çalma listem: Rap.'),
...
(10, 'https://musisocial.com/post10', '2025-01-10 18:00:00', 10, 'Müzik platformu
önerisi.');
```

```
Sorgu:
Select* From Paylaşım;
```

	Post_ID	P_Url	PaylasimTarihi	K_ID	Icerik
1	1	https://musisocial.com/post1	2025-01-01 09:00:00.000	1	En sevdiğim şarkı.
2	2	https://musisocial.com/post2	2025-01-02 10:00:00.000	2	Yeni albümüm yayında!
3	3	https://musisocial.com/post3	2025-01-03 11:00:00.000	3	Çalma listem: Rap.
4	4	https://musisocial.com/post4	2025-01-04 12:00:00.000	4	Pop müzik severlere önerim.
5	5	https://musisocial.com/post5	2025-01-05 13:00:00.000	5	Yeni remiximi dinleyin!
6	6	https://musisocial.com/post6	2025-01-06 14:00:00.000	6	Müzik bloggerından öneriler.
7	7	https://musisocial.com/post7	2025-01-07 15:00:00.000	7	Rap gecesi.
8	8	https://musisocial.com/post8	2025-01-08 16:00:00.000	8	Albüm koleksiyonum.
9	9	https://musisocial.com/post9	2025-01-09 17:00:00.000	9	Metal müzikle bir gün.
10	10	https://musisocial.com/post10	2025-01-10 18:00:00.000	10	Müzik platformu önerisi.

Müzik Tablosu Doldurma :

```
INSERT INTO Müzik (Post_ID, CikisTarihi, M_Turu, M_Suresi, S_ID)
VALUES
(1, '2020-01-01', 'Pop', 210, 1),
(2, '2021-06-15', 'Rock', 180, 2),
(3, '2022-09-20', 'Rap', 240, 3),
...
(10, '2023-01-15', 'Pop', 220, 10);
Sorgu:
Select* From Müzik WHERE M_Turu='Pop';
```

Bu sorgu, **"Müzik"** tablosunda yer alan tüm satırları seçer ve sadece **"M_Turu"** sütununda değeri **'Pop'** olan kayıtları

	M_ID	Post_ID	CikisTarihi	M_Turu	M_Suresi	S_ID
1	1	1	2020-01-01	Pop	210	1
2	4	4	2019-11-11	Pop	200	4
3	10	10	2023-01-15	Pop	220	10

Sanatçı Tablosu Doldurma :

```
INSERT INTO Sanatçı (S_ID, AlbumSayısı, ŞarkıSayısı, K_ID)
VALUES
(1, 5, 50, 1),  -- Ali Yılmaz
(2, 4, 40, 2),  -- Ayşe Demir
(3, 6, 60, 3),  -- Mehmet Kaya
...
(10, 2, 25, 10); -- Leyla Akın

Sorgu:
SELECT* FROM Sanatçı WHERE S_ID IN(1,2,3);
```

Bu sorgu, **"Sanatçı"** tablosunda yer alan tüm sütunları seçer ve sadece **"S_ID"** sütunundaki değeri **1**, **2**, veya **3** olan

	S_ID	AlbumSayısı	ŞarkıSayısı	K_ID
1	1	5	50	1
2	2	4	40	2
3	3	6	60	3

Abonelik Tablosu Doldurma :

```
INSERT INTO Abonelik (AB_ID, K_ID, BaşlangıçTarihi, Durum, BitişTarihi)
VALUES
(1, 1, '2024-01-01', 'Aktif', '2025-01-01'),
(2, 2, '2023-06-15', 'Pasif', '2024-06-15'),
(3, 3, '2023-09-20', 'Aktif', '2024-09-20'),
...
(10, 10, '2023-01-15', 'Aktif', '2024-01-15');

Sorgu:
Select* From Abonelik;
```

	AB_ID	K_ID	BaşlangıçTarihi	Durum	BitişTarihi
1	1	1	2024-01-01	Aktif	2025-01-01
2	2	2	2023-06-15	Pasif	2024-06-15
3	3	3	2023-09-20	Aktif	2024-09-20
4	4	4	2022-11-11	Aktif	2023-11-11
5	5	5	2023-03-05	Pasif	2024-03-05
6	6	6	2023-12-31	Aktif	2024-12-31
7	7	7	2022-08-22	Aktif	2023-08-22
8	8	8	2021-10-01	Pasif	2022-10-01
9	9	9	2022-05-10	Aktif	2023-05-10
10	10	10	2023-01-15	Aktif	2024-01-15

…

7. SQL Stored Procedure:

1--"YorumEkleme" adında bir stored procedure. Bu stored procedure, bir kullanıcı tarafından yapılan yorumu Yorum tablosuna eklemek için kullanılır.

```
CREATE PROCEDURE YorumEkleme
    @Y_ID INT,
    @K_ID INT,
    @Post_ID INT,
    @Y_Icerik NVARCHAR(MAX),
    @YorumTarihi DATETIME
AS
BEGIN
    INSERT INTO Yorum (Y_ID,K_ID, Post_ID, Y_Icerik, YorumTarihi)
    VALUES (@Y_ID,@K_ID, @Post_ID, @Y_Icerik, @YorumTarihi);
END;
```

--Kullanım : Yoruma ait ID, yorumu yapan kullanıcı ID'si, yorum yapılan paylaşımın ID'si, içerik ve tarih bilgisi alınarak tabloya kaydedilir.

```
EXEC YorumEkleme
    @Y_ID = 6,
    @K_ID = 2, -- Yorumu yapan kullanıcının ID'si
    @Post_ID = 1, -- Yorum yapılan paylaşımın ID'si
    @Y_Icerik = 'Harika bir paylaşım!',
    @YorumTarihi = '2025-01-11';

Select*From Yorum Where Y_ID=6;
```

	Y_ID	K_ID	Y_Icerik	YorumTarihi	Post_ID
1	6	2	Harika bir paylaşım!	2025-01-11 00:00:00.000	1

2-- Aşağıdaki Stored Procedure bir paylaşım silindiğinde, ilişkili veriler (yorumlar, beğeniler ve müzikler) otomatik olarak silinir. Böylece veri bütünlüğü korunur.

```
CREATE PROCEDURE SilPaylaşım
    @Post_ID INT
AS
BEGIN
    BEGIN TRANSACTION;
```

```
    BEGIN TRY
        -- Yorumları Sil
        DELETE FROM Yorum
        WHERE Post_ID = @Post_ID;

        -- Beğenileri Sil
        DELETE FROM Beğeni
        WHERE Post_ID = @Post_ID;

        -- Müzikleri Sil
        DELETE FROM Müzik
        WHERE Post_ID = @Post_ID;

        -- Paylaşımı Sil
        DELETE FROM Paylaşım
        WHERE Post_ID = @Post_ID;

        -- İşlemi Onayla
        COMMIT TRANSACTION;
        PRINT 'Paylaşım ve ilişkili veriler başarıyla silindi.';
    END TRY
    BEGIN CATCH
        -- Hata Durumunda İşlemi Geri Al
        ROLLBACK TRANSACTION;
        PRINT 'Hata oluştu. İşlem geri alındı.';
        THROW;
    END CATCH;
END;
```

-- Procedure Kullanımı: PostID = 1 olan paylaşımı ve ilişkileri siler ve aşağıdaki çıktıya sahip olur.

```
EXEC SilPaylaşım @Post_ID = 1;
```

```
(1 row affected)
Paylaşım ve ilişkili veriler başarıyla silindi.
```

-- Paylaşım Tablosunun Procedure kullanımı sonraki durumu.

	Post_ID	P_Url	PaylasimTarihi	K_ID	Icerik
1	2	https://musisocial.com/post2	2025-01-02 10:00:00.000	2	Yeni albümüm yayında!
2	3	https://musisocial.com/post3	2025-01-03 11:00:00.000	3	Çalma listem: Rap.
3	4	https://musisocial.com/post4	2025-01-04 12:00:00.000	4	Pop müzik severlere önerim.
4	5	https://musisocial.com/post5	2025-01-05 13:00:00.000	5	Yeni remiximi dinleyin!
5	6	https://musisocial.com/post6	2025-01-06 14:00:00.000	6	Müzik bloggerından öneriler.
6	7	https://musisocial.com/post7	2025-01-07 15:00:00.000	7	Rap gecesi.
7	8	https://musisocial.com/post8	2025-01-08 16:00:00.000	8	Albüm koleksiyonum.
8	9	https://musisocial.com/post9	2025-01-09 17:00:00.000	9	Metal müzikle bir gün.
9	10	https://musisocial.com/post10	2025-01-10 18:00:00.000	10	Müzik platformu önerisi.

8. SQL Trigger Örnekleri:

“Trigger”, bir tabloda gerçekleşen “**INSERT”**, “**UPDATE”** veya “**DELETE”** gibi veri değişikliklerini otomatik olarak tetikleyen ve belirtilen işlemleri gerçekleştiren bir veritabanı nesnesidir.

1-- Bu Trigger, "Abonelik" tablosunda bir güncelleme yapıldığında, BitişTarihi geçmiş bir tarih olan aboneliklerin Durum sütununu otomatik olarak 'Pasif' olarak günceller.

```
CREATE TRIGGER AbonelikDurumGüncelle
ON Abonelik
AFTER UPDATE
AS
BEGIN
    IF EXISTS (SELECT * FROM INSERTED WHERE BitişTarihi < GETDATE())
    BEGIN
        UPDATE Abonelik
        SET Durum = 'Pasif'
        WHERE AB_ID IN (SELECT AB_ID FROM INSERTED WHERE BitişTarihi < GETDATE());
    END;
END;
```

-- Aşağıdaki komut, "Abonelik" tablosunda AB_ID = 3 olan aboneliğin BitişTarihi sütununu '2025-01-01' olarak günceller. Ardından, güncellenen aboneliğin bilgilerini sorgulayarak detaylarını görüntüler. (durumun 'Pasif' olarak güncellenmesi).

```
UPDATE Abonelik
SET BitişTarihi = '2025-01-01'
WHERE AB_ID = 3; -- Güncellenen abonelik ID'si
Select* From Abonelik WHERE AB_ID=3;
```

-- Trigger İşleminden önce Abonelik tablosu K_ID=3 olan abonenin abonelik durumu (AKTİF)

3	3	3	2023-09-20	Aktif	2024-09-20

-- Trigger İşleminden sonra Abonelik tablosu K_ID=3 olan abonenin abonelik durumu (PASİF)

	AB_ID	K_ID	BaşlangıçTarihi	Durum	BitişTarihi
1	3	3	2023-09-20	Pasif	2025-01-01

2-- Paylaşım tablosunda bir güncelleme işlemi yapıldığında çalışan ve paylaşım sahibine, paylaşımının güncellendiğine dair bir bildirim ekleyen bir trigger oluşturacağız.

```
CREATE TRIGGER GüncellemeBildirimGönder
ON Paylaşım
AFTER UPDATE
AS
BEGIN
    SET NOCOUNT ON;

    -- Bildirim Ekle
    INSERT INTO Bildirimler (K_ID, B_Metni, Okunma, OluşTarihi)
    SELECT
        K_ID,
        'Paylaşımınız güncellenmiştir: ' + Icerik,
        0,
        GETDATE()
    FROM
        INSERTED;

    PRINT 'Güncelleme bildirimi başarıyla gönderildi.';
END;
```

-- Fakat bu Trigger'da aşağıdaki hata Bildirimler tablosundaki B_ID sütununun NULL değer almasına izin verilmediğini, ancak trigger'daki INSERT işleminin B_ID için değer sağlamadığını gösterir. Bu yüzden triggerda B_ID için değer sağlayacağız.

```
Msg 515, Level 16, State 2, Procedure GüncellemeBildirimGönder, Line 9 [Batch Start Line 440]
Cannot insert the value NULL into column 'B_ID', table 'master.dbo.Bildirimler'; column does not allow nulls. INSERT fails
The statement has been terminated.
```

-- Çözüm olarak öncelikle

```
DROP trigger GüncellemeBildirimGönder;
```

komutu ile var olan triggerı veri tabanımızdan siliyoruz ve sonrasından yeni Trigger'da her INSERT(Ekleme) işleminde "B_ID" için bir değer oluşturmamız gerekiyor, Mevcut en büyük B_ID değerine bir ekleyerek yeni bir değer sağlar.

```
CREATE TRIGGER GüncellemeBildirimGönder
ON Paylaşım
AFTER UPDATE
AS
BEGIN
    SET NOCOUNT ON;

    -- En büyük B_ID'yi bul ve yeni bir değer oluştur
    DECLARE @NextB_ID INT;
    SELECT @NextB_ID = ISNULL(MAX(B_ID), 0) + 1 FROM Bildirimler;

    -- Bildirim Ekle
    INSERT INTO Bildirimler (B_ID, K_ID, B_Metni, Okunma, OluşTarihi)
    SELECT
        @NextB_ID + ROW_NUMBER() OVER (ORDER BY (SELECT NULL)), -- B_ID'yi artır
        K_ID,
        'Paylaşımınız güncellenmiştir: ' + Icerik,
        0,
        GETDATE()
    FROM
        INSERTED;

    PRINT 'Güncelleme bildirimi başarıyla gönderildi.';
END;
```

-- Bu komut ile Post_ID = 10 için "Bildirim" ve "Paylaşım_içeriği" aşağıdaki gibi güncellenir.

```
UPDATE Paylaşım
SET Icerik = 'Platform önerisi olan var mı?', PaylasimTarihi = GETDATE()
WHERE Post_ID = 10;
Select*From Paylaşım Where Post_ID=10;
Select*From Bildirimler Where K_ID=10;
```

-- Trigger Kullanımı Öncesi:

	Post_ID	P_Url	PaylasimTarihi	K_ID	Icerik
1	10	https://musisocial.com/post10	2025-01-10 18:00:00.000	10	Müzik platformu önerisi.

-- Trigger Kullanımı Sonrası :

	Post_ID	P_Url	PaylasimTarihi	K_ID	Icerik
1	10	https://musisocial.com/post10	2025-01-05 21:52:08.350	10	Platform önerisi olan var mı?

	B_ID	K_ID	B_Metni	Okunma	OluşTarihi
1	10	10	Yeni bir takipçiniz var!	1	2025-01-10 00:00:00.000
2	12	10	Paylaşımınız güncellenmiştir: Platform önerisi o...	0	2025-01-05 21:52:08.357

9. SQL Views Örnekleri:

-- Aşağıdaki "View", kullanıcıların yaptığı yorumları, ilgili "paylaşım" ve "paylaşım sahibinin" detaylarıyla birlikte ilişkilendirerek daha kapsamlı bir veri sunar.

```
CREATE VIEW KullanıcıYorumDetayları AS
SELECT
    Yorum.Y_ID AS YorumID,
    Kullanıcı.K_ID AS KullanıcıID,
    Kullanıcı.K_AD AS KullanıcıAdı,
    Yorum.Y_Icerik AS YorumİçeriĞi,
    Paylaşım.Post_ID AS PaylaşımID,
    Paylaşım.Icerik AS PaylaşımİçeriĞi,
    Paylaşım.K_ID AS PaylaşımSahibiID,
    Sahibi.K_AD AS PaylaşımSahibiAdı,
    Yorum.YorumTarihi AS YorumTarihi
FROM
    Yorum
INNER JOIN
    Kullanıcı ON Yorum.K_ID = Kullanıcı.K_ID
INNER JOIN
    Paylaşım ON Yorum.Post_ID = Paylaşım.Post_ID
INNER JOIN
    Kullanıcı AS Sahibi ON Paylaşım.K_ID = Sahibi.K_ID;
```

Sorgu:

```
SELECT * FROM KullanıcıYorumDetayları;
```

	YorumID	KullanıcıID	KullanıcıAdı	YorumİçeriĞi	PaylaşımID	PaylaşımİçeriĞi	PaylaşımSahibiID	PaylaşımSahibiAdı	YorumTarihi
1	6	2	Seyit İlham Fırtına	Harika bir paylaşım!	1	En sevdiğim şarkı.	1	Halil İbrahim Turan	2025-01-11 00:00:00.000

10. SQL Transaction Örnekleri:

-- Aşağıdaki "TRANSACTION" bir kullanıcı ekleme işlemini başlatır, hata durumunda işlemi geri alır ve başarılı olduğunda değişiklikleri onaylar.

```
BEGIN TRANSACTION;
BEGIN TRY
    -- Kullanıcıyı ekle
    INSERT INTO Kullanıcı (K_AD, K_Email, K_Şifre, K_Bio, DogumTarihi, K_Resmi)
```

```
    VALUES ('Yusuf Soysal','istanbulbeyefendisi@gmail.com', 'şeyhmus123', 'Müzik
sever', '1990-01-01', NULL);

    -- İşlemi onayla
    COMMIT TRANSACTION;
    PRINT 'Kullanıcı başarıyla eklendi.';
END TRY
BEGIN CATCH
    -- Hata durumunda işlemi geri al
    ROLLBACK TRANSACTION;
    PRINT 'Bir hata oluştu. İşlem geri alındı.';
    THROW;
END CATCH;

Select*From Kullanıcı Where K_AD='Yusuf Soysal';
```

	K_ID	K_AD	K_Email	K_Şifre	K_Bio	DogumTarihi	K_Resmi
1	18	Yusuf Soysal	istanbulbeyefendisi@gmail.com	şeyhmus123	Müzik sever	1990-01-01	NULL

-- Aşağıdaki transactiondaki K_AD yerine sayısal bir değer insert yaptığımızda transaction ROLLBACK TRANSACTION yaparak işlemi geri alacaktır.

```
BEGIN TRANSACTION;
BEGIN TRY
    -- Kullanıcıyı ekle
    INSERT INTO Kullanıcı (K_AD, K_Email, K_Şifre, K_Bio, DogumTarihi, K_Resmi)
    VALUES (11,'istanbulbeyefendisi@gmail.com', 'şeyhmus123', 'Müzik sever', '1990-01-
01', NULL);

    -- İşlemi onayla
    COMMIT TRANSACTION;
    PRINT 'Kullanıcı başarıyla eklendi.';
END TRY
BEGIN CATCH
    -- Hata durumunda işlemi geri al
    ROLLBACK TRANSACTION;

    -- Hata mesajını göster
    PRINT 'Bir hata oluştu. İşlem geri alındı.';
    THROW;
END CATCH;
```

```
(0 rows affected)
Bir hata oluştu. İşlem geri alındı.
Msg 2627, Level 14, State 1, Line 340
Violation of UNIQUE KEY constraint 'UQ__Kullanıc__F1E61AC8C5E3BA21'. Cannot insert duplicate key in object 'dbo.Kullanıcı'. The duplicate key value is (istanbulbey
```