

**ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



BLM4522 PROJE RAPORU

SQL Server Tabanlı Veritabanı Yönetim Uygulamaları
<https://github.com/JineenRihawi>

**Umut Akylbek kyzы 21291003
Cenin Rihavi 21291007**

Öğretim Görevlisi Enver Bağcı

23.05.2025

[İçindekiler](#)

- 1. Giriş**
- 2. Veritabanı Performans Optimizasyonu ve İzleme**
- 3. Veritabanı Yedekleme ve Felaketten Kurtarma Plan**
- 4. Veritabanı Güvenliği ve Erişim Kontrolü Veri Temizleme ve ETL Süreçleri Tasarımı**
- 5. Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları**
- 6. Veri Temizleme ve ETL Süreçleri Tasarımı**
- 7. Veritabanı Yükseltme ve Sürüm Yönetimi**
- 8. Veritabanı Yedekleme ve Otomasyon Çalışması**
- 9. Sonuç**
- 10. Kaynakça**

1. GİRİŞ

Bu rapor, büyük ölçekli bir veritabanı sisteminde gerçekleştirilen kapsamlı yönetim, bakım, optimizasyon ve güvenlik çalışmalarını içermektedir. Modern veri tabanı yönetimi; performans iyileştirmeleri, veri güvenliği, felaket senaryolarına hazırlık, yedekleme otomasyonu, veri kalitesi süreçleri ve sistem sürekliliği gibi birçok kritik alanı kapsamaktadır. Bu bağlamda, SQL Server ortamında yürütülen çeşitli çalışmalarla; sistemin verimli, güvenli ve sürdürülebilir bir şekilde çalışması hedeflenmiştir.

Raporda yer alan her bir başlık, sistem yönetiminin belirli bir yönünü ele almakta ve uygulanan teknik detaylarla birlikte incelenmektedir. Performans izleme ve optimizasyonundan başlayarak, yedekleme stratejileri, felaket kurtarma planları, erişim denetimi, dağıtık veritabanı yönetimi, ETL süreçleri, sürüm yükseltme ve otomasyon gibi geniş bir yelpazede yapılan işlemler detaylı bir şekilde açıklanacaktır.

Amaç, sadece mevcut sistemi yönetmek değil, aynı zamanda potansiyel riskleri önceden tespit edip proaktif çözümlerle sistemin dayanıklılığını artırmaktır.

2. Veritabanı Performans Optimizasyonu ve İzleme

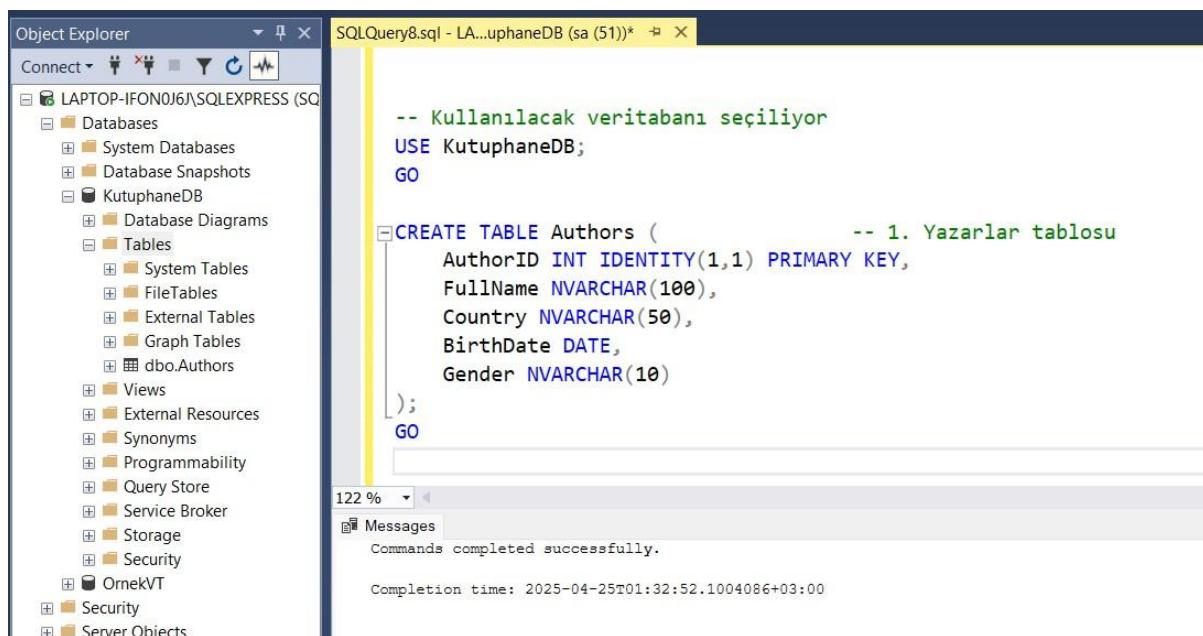
Amaç, veritabanı üzerinde çalışan sorguların kaynak tüketimini analiz ederek, sistem performansını düşüren unsurları tespit etmek ve bu doğrultuda iyileştirme adımları önermektir.

2.1 Veri Tabanı Oluşturma Komutu

```
-- Veritabanı oluşturuluyor
CREATE DATABASE KutuphaneDB;
GO
```

Şekil 1 Veri Tabanı Oluşturma

2.2 Yazarlar, Kitaplar, Okuyucular ve Ödünç Alma Kayıtları Tabloları Oluşturma Komutları



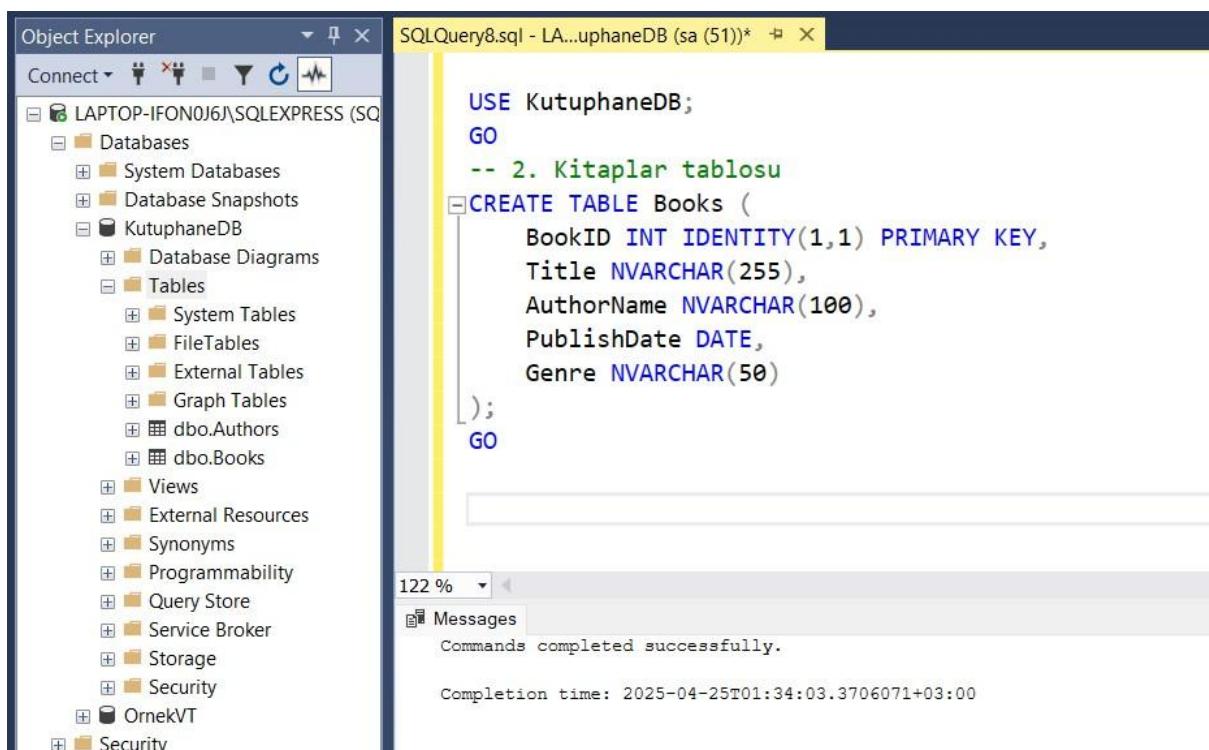
The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the 'KutuphaneDB' database, which contains the 'Authors' table. The SQL Query window displays the T-SQL code for creating the 'Authors' table:

```
-- Kullanılacak veritabanı seçiliyor
USE KutuphaneDB;
GO

CREATE TABLE Authors (
    AuthorID INT IDENTITY(1,1) PRIMARY KEY,
    FullName NVARCHAR(100),
    Country NVARCHAR(50),
    BirthDate DATE,
    Gender NVARCHAR(10)
);
GO
```

The 'Messages' pane at the bottom indicates that the command completed successfully.

Sekil 2 Yazarlar Tablosu



The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the 'KutuphaneDB' database, which contains the 'Books' table. The SQL Query window displays the T-SQL code for creating the 'Books' table:

```
USE KutuphaneDB;
GO

-- 2. Kitaplar tablosu
CREATE TABLE Books (
    BookID INT IDENTITY(1,1) PRIMARY KEY,
    Title NVARCHAR(255),
    AuthorName NVARCHAR(100),
    PublishDate DATE,
    Genre NVARCHAR(50)
);
GO
```

The 'Messages' pane at the bottom indicates that the command completed successfully.

Sekil 3 Kitaplar Tablosu

The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the 'KutuphaneDB' database under the 'Tables' node. The SQL Query window contains the following code:

```
USE KutuphaneDB;
GO
CREATE TABLE Readers (
    ReaderID INT IDENTITY(1,1) PRIMARY KEY,
    FullName NVARCHAR(100),
    Address NVARCHAR(255),
    Email NVARCHAR(100),
    Phone NVARCHAR(50)
);
GO
```

The status bar at the bottom indicates a completion time of 2025-04-25T01:35:16.4363592+03:00.

Şekil 4 Okuyucular Tablosu

The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the 'KutuphaneDB' database under the 'Tables' node. The SQL Query window contains the following code:

```
USE KutuphaneDB;
GO
CREATE TABLE Borrowings (
    BorrowingID INT IDENTITY(1,1) PRIMARY KEY,
    BookID INT FOREIGN KEY REFERENCES Books(BookID), -- Ödünç alınan kitap ID'si
    ReaderID INT FOREIGN KEY REFERENCES Readers(ReaderID), -- Ödünç alan okuyucu ID'si
    BorrowDate DATE,
    ReturnDate DATE
);
GO
```

The status bar at the bottom indicates a completion time of 2025-04-25T01:37:02.7938882+03:00.

Şekil 5 Ödünç Alma Kayıtları Tablosu

2.3 Tablolara Veri Ekleme

```

Object Explorer
SQLQuery6.sql - LAP...ninkullanici1 (51)* SQLQuery4.sql - LA...uphaneDB (sa (52))* ×

SELECT * FROM Authors
VALUES
('Nicholas McGuire', 'USA', '1975-03-12', 'Male'),
('James Salas', 'UK', '1980-07-25', 'Male'),
('Michael Olson', 'Canada', '1968-11-09', 'Male'),
('Raven Carpenter', 'USA', '1990-02-18', 'Female'),
('David Fowler', 'Germany', '1973-10-01', 'Male'),
('Lisa Richards', 'France', '1988-12-04', 'Female'),
('Amanda Fox', 'Italy', '1983-06-30', 'Female'),
('Stephen Sanders', 'Australia', '1979-04-22', 'Male'),
('Megan Terry', 'Spain', '1992-08-11', 'Female'),
('Gary Hicks', 'Mexico', '1985-01-15', 'Male'),
('Emily Sullivan', 'USA', '1991-05-05', 'Female'),
('Daniel Vargas', 'Portugal', '1976-09-17', 'Male'),
('Isabelle Leonard', 'France', '1984-03-22', 'Female'),
('Henry Morrison', 'Ireland', '1977-06-29', 'Male'),
('Patricia Bishop', 'Canada', '1982-11-13', 'Female'),
('Caleb Yates', 'Germany', '1989-07-02', 'Male'),
('Olivia Becker', 'Austria', '1993-10-25', 'Female'),
('John Lee', 'USA', '1986-01-09', 'Male'),
('Angela Barnes', 'UK', '1990-09-03', 'Female'),
('Ryan Cox', 'USA', '1981-02-14', 'Male');

```

Messages
Commands completed successfully.
Completion time: 2025-04-25T07:26:21.6187943+03:00

Sekil 6 Yazarlar Tablosuna veri ekleme

```

Object Explorer
SQLQuery6.sql - LAP...ninkullanici1 (51)* SQLQuery4.sql - LA...uphaneDB (sa (52))* ×

SELECT * FROM Books
VALUES
('Focused bandwidth-monitored standardization', 'Nicholas McGuire', '2022-01-20', 'Fiction'),
('Public-key fresh-thinking project', 'James Salas', '2015-05-31', 'Fiction'),
('Synergized neutral emulation', 'Michael Olson', '2022-10-24', 'Fantasy'),
('Expanded modular toolset', 'Raven Carpenter', '2021-08-26', 'Science'),
('Synergistic clear-thinking pricing structure', 'David Fowler', '2021-12-29', 'History'),
('Exclusive client-sensitive paradigm', 'Lisa Richards', '2023-04-10', 'Philosophy'),
('Ergonomic mobile migration', 'Amanda Fox', '2020-11-03', 'Romance'),
('Sharable static application', 'Stephen Sanders', '2019-06-14', 'Adventure'),
('Implemented grid-enabled installation', 'Megan Terry', '2022-07-19', 'Biography'),
('Intuitive tangible core', 'Gary Hicks', '2017-03-08', 'Fantasy'),
('Extended background internet solution', 'Emily Sullivan', '2021-05-11', 'Poetry'),
('Compatible national instruction set', 'Daniel Vargas', '2020-09-23', 'Science'),
('Integrated mission-critical open system', 'Isabelle Leonard', '2022-02-17', 'Drama'),
('Organized system-worthy infrastructure', 'Henry Morrison', '2023-03-30', 'Fiction'),
('Distributed dynamic projection', 'Patricia Bishop', '2019-01-05', 'Thriller'),
('Streamlined well-modulated neural-net', 'Caleb Yates', '2021-10-10', 'Children'),
('Multi-layered neutral neural-net', 'Olivia Becker', '2018-12-12', 'Technology'),
('Configurable executive interface', 'John Lee', '2016-08-04', 'Poetry'),
('Switchable 6th generation initiative', 'Angela Barnes', '2020-02-22', 'Self-Help'),
('Automated cohesive orchestration', 'Ryan Cox', '2023-06-07', 'History');

```

Messages
Commands completed successfully.
Completion time: 2025-04-25T07:26:21.6187943+03:00

Sekil 7 Kitaplar Tablosuna veri ekleme

```

INSERT INTO Readers (FullName, Email, Phone, Address)
VALUES
('Aylin Demir', 'aylin.demir@example.com', '+905321112233', 'Kadıköy, İstanbul'),
('Mehmet Yıldız', 'mehmet.yildiz@example.com', '+905324445566', 'Beşiktaş, İstanbul'),
('Zeynep Kaya', 'zeynep.kaya@example.com', '+905326778899', 'Çankaya, Ankara'),
('Ahmet Kurt', 'ahmet.kurt@example.com', '+905329990011', 'Beylikdüzü, İstanbul'),
('Elif Öz', 'elif.oz@example.com', '+905321223344', 'Ataşehir, İstanbul'),
('Cem Karaca', 'cem.karaca@example.com', '+905325556677', 'Kızılay, Ankara'),
('Buse Tan', 'buse.tan@example.com', '+905328889000', 'Üsküdar, İstanbul'),
('Deniz Arslan', 'deniz.arslan@example.com', '+905321234567', 'Şişli, İstanbul'),
('Selin Doğan', 'selin.dogan@example.com', '+905322345678', 'Fatih, İstanbul'),
('Emir Koç', 'emir.koc@example.com', '+905323456789', 'Kadıköy, İstanbul'),
('Nazlı Şahin', 'nazli.sahin@example.com', '+905324567890', 'Çankaya, Ankara'),
('Berk Yılmaz', 'berk.yilmaz@example.com', '+905325678901', 'Pendik, İstanbul'),
('Derya Acar', 'derya.acar@example.com', '+905326789012', 'Kartal, İstanbul'),
('Okan Güneş', 'okan.gunes@example.com', '+905327890123', 'Keçiören, Ankara'),
('İrem Polat', 'irem.polat@example.com', '+905328901234', 'Ümraniye, İstanbul'),
('Ali Ural', 'ali.ural@example.com', '+905329012345', 'Göztepe, İstanbul'),
('Mert Eren', 'mert.eren@example.com', '+905321098765', 'Mecidiyeköy, İstanbul'),
('Pelin Ceylan', 'pelin.ceylan@example.com', '+905322198765', 'Sarıyer, İstanbul'),
('Barış Aksoy', 'baris.aksoy@example.com', '+905323298765', 'Beyoğlu, İstanbul'),
('Sevgi Kaplan', 'sevgi.kaplan@example.com', '+905324398765', 'Eskişehir Yolu, Ankara');

```

Sekil 8 Okuyucular Tablosuna veri ekleme

```

INSERT INTO Borrowings (ReaderID, BookID, BorrowDate, ReturnDate)
VALUES
(1, 3, '2023-03-01', '2023-03-15'),
(2, 5, '2023-04-10', '2023-04-25'),
(3, 7, '2022-11-12', '2022-11-26'),
(4, 1, '2021-09-05', '2021-09-20'),
(5, 8, '2023-06-18', NULL),
(6, 4, '2020-02-10', '2020-02-24'),
(7, 10, '2022-01-14', '2022-01-28'),
(8, 2, '2023-07-19', NULL),
(9, 6, '2021-05-03', '2021-05-17'),
(10, 9, '2020-12-25', '2021-01-08'),
(11, 12, '2023-01-02', '2023-01-16'),
(12, 13, '2022-04-09', '2022-04-23'),
(13, 14, '2023-02-11', NULL),
(14, 15, '2022-09-07', '2022-09-21'),
(15, 11, '2021-07-15', '2021-07-29'),
(16, 16, '2020-03-20', '2020-04-03'),
(17, 18, '2021-10-12', '2021-10-26'),
(18, 17, '2022-05-23', NULL),
(19, 19, '2023-03-30', NULL),
(20, 20, '2023-04-14', NULL);

```

Sekil 9 Ödünç Alanlar Tablosuna veri ekleme

2.4 Tüm Tabloları Listeleme:

Veritabanında yer alan tüm kullanıcı (base) tablolarını listeler. Performans veya yapı analizine başlamadan önce tablo listesini görmek önemlidir.

The screenshot shows the SSMS interface. On the left, the Object Explorer displays the database structure of 'LAPTOP-IFON0J6\SQLEXPRESS (SQ...)' with nodes like Databases, Tables, Views, and Security. In the center, a query window titled 'SQLQuery8.sql - LA...uphaneDB (sa (51))' contains the following T-SQL code:

```
SELECT TABLE_SCHEMA, TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE'
ORDER BY TABLE_SCHEMA, TABLE_NAME;
```

A comment in the code reads: '-- Kütüphane veritabanındaki tüm -- temel (base) tabloları listeler'. Below the code, the 'Results' tab shows a table with four rows:

	TABLE_SCHEMA	TABLE_NAME
1	dbo	Authors
2	dbo	Books
3	dbo	Borrowings
4	dbo	Readers

Şekil 10 Tüm Tabloları Listeleme

2.5 Alternatif Soru: En Çok CPU Kullanan Sorgular

Bu soru, SQL Server'da en fazla CPU kullanan sorguları ortalama CPU süresine göre listeler. sys.dm_exec_query_stats görünümünden alınan verilerle, her sorgunun kaç kez çalıştığı, toplam ve ortalama CPU süresi, ilk ve son çalıştırılma zamanı gibi bilgiler elde edilir. CROSS APPLY ile sorgunun metni de görüntülenir.

Amaç, sistem kaynaklarını en çok tüketen sorguları tespit edip performans sorunlarını analiz etmektir. Bu sayede iyileştirme gerektiren sorgular belirlenip, indeksleme ya da sorgu optimizasyonu gibi adımlarla sistem verimliliği artırılabilir.

```

-- En çok CPU kullanan 10 sorguyu listeler (ortalama CPU zamanı baz alınarak)
SELECT TOP 10
    total_worker_time / execution_count AS Avg_CPU_Time, -- Ortalama CPU süresi
    execution_count, -- Sorgunun kaç kez çalıştığı
    total_worker_time, -- Toplam CPU zamanı
    SUBSTRING(qt.text,
        (qs.statement_start_offset / 2) + 1,
        ((CASE qs.statement_end_offset
            WHEN -1 THEN DATALENGTH(qt.text)
            ELSE qs.statement_end_offset
        END - qs.statement_start_offset) / 2) + 1) AS QueryText -- Sorgu metni
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) qt
ORDER BY Avg_CPU_Time DESC;

```

	Avg_CPU_Time	execution_count	total_worker_time	QueryText
1	97516	1	97516	SELECT target_data FROM sys.dm_xe_sess...

Sekil 11 En Çok CPU Kullanan Sorgular

2.6 Sistemde Kullanılmayan veya Az Kullanılan İndekslerin Tespiti

Bu sorgu, veritabanındaki kullanıcı tabloları üzerinde tanımlı indekslerin ne kadar aktif kullanıldığını analiz eder. `sys.dm_db_index_usage_stats` görünümünden alınan arama (seek), tarama (scan) ve lookup istatistikleri ile hangi indekslerin nadiren kullanıldığı belirlenebilir.

Amaç, performansa katkı sağlamayan indeksleri tespit edip, sistem kaynaklarını boş harcayan yapıları temizlemektir. Böylece hem yazma işlemlerinde (update/insert) performans artar hem de bakım maliyeti azalır.

```

-- En az kullanılan indeksleri tespit eder (arama, tarama, lookup sayısına göre)
SELECT
    OBJECT_NAME(i.object_id) AS TableName,
    i.name AS IndexName,
    i.index_id,
    s.user_seeks, s.user_scans, -- Arama ve tarama sayısı
    s.user_lookups, s.user_updates
FROM sys.indexes AS i
LEFT JOIN sys.dm_db_index_usage_stats AS s
    ON i.object_id = s.object_id AND i.index_id = s.index_id
WHERE OBJECTPROPERTY(i.object_id, 'IsUserTable') = 1 -- Sadece kullanıcı tabloları
    AND i.name IS NOT NULL
ORDER BY (ISNULL(s.user_seeks, 0) + ISNULL(s.user_scans, 0) + ISNULL(s.user_lookups, 0)) ASC;

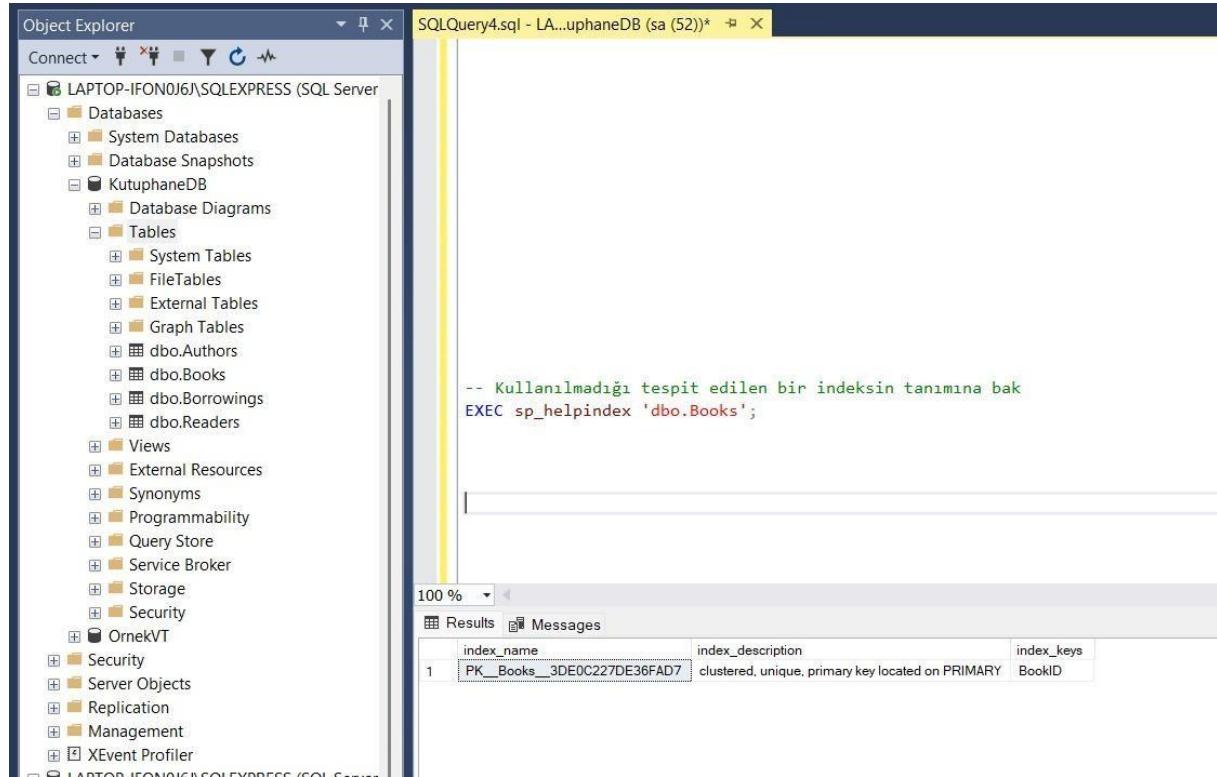
```

TableName	IndexName	index_id	user_seeks	user_scans	user_lookups	user_updates
1 Authors	PK__Authors__70DAFC143ACD9C7A	1	0	0	0	1
2 Borrowings	PK__Borrowings__6C093F730650673	1	0	0	0	1
3 Books	PK__Books__3DE0C227246EB47F	1	1	0	0	1
4 Readers	PK__Readers__8E67A58108A56118	1	1	0	0	1

Sekil 12 Az Kullanılan İndekslerin Tespiti

Bu adımda, kullanılmadığı belirlenen bir indeksin yapısını ve hangi sütunlar üzerinde tanımlandığını görmek için `sp_helpindex` prosedürü kullanılır. Bu, indeksin gerçekten gereksiz olup olmadığını anlamak adına önemlidir.

İnceleme sonrası gereksiz olduğu kesinleşen indeksler `DROP INDEX` komutu ile kaldırılarak veritabanı daha optimize hâle getirilir. Böylece veri yazma işlemleri hızlanır, disk kullanımı azalır ve bakım süreçleri sadeleşir.



```
-- Kullanılmadığı tespit edilen bir indeksin tanımına bak
EXEC sp_helpindex 'dbo.Books';
```

index_name	index_description	index_keys
PK_Books_3DE0C227DE36FAD7	clustered, unique, primary key located on PRIMARY	BookID

Şekil 13 Az Kullanılan İndekslerin Tespitinin sonucu

2.7 Borrowings Tablosu için Yeni İndeks Oluşturma

Şekil 14 , "Borrowings" tablosunda performans iyileştirmesi amacıyla oluşturulan yeni bir indeksi göstermektedir. "ReaderId" ve "BookId" sütunları üzerinde tanımlanan bu indeks, bu sütunlarda yapılan aramaların hızlanmasılığını sağlamaktadır. İndeksin "NONCLUSTERED" olarak belirlenmesi, veri erişim hızını artırırken, tablonun fiziksel düzenini korumaktadır.

Bu optimizasyon, özellikle sık kullanılan sorguların yanıt sürelerini önemli ölçüde azaltarak sistem genelinde daha verimli bir performans sunmayı hedeflemektedir.

```
-- OkuyucuId ve KitapId Üzerinde aramaları hızlandırmak için yeni bir indeks
CREATE NONCLUSTERED INDEX IX_Borrowings_ReaderId_BookId
ON Borrowings (ReaderId, BookId); -- Borrowings tablosunda
```

101 %

Messages
Commands completed successfully.
Completion time: 2026-04-26T02:55:53.2370462+03:00

Şekil 14 Borrowings Tablosu için Yeni İndeks Oluşturma

2.8 En Yavaş Sorguların Tespiti

Şekil 15, KutuphaneDB veritabanında çalıştırılan sorguların performans analizini göstermektedir. sys.dm_exec_query_stats dinamik yönetim görünümü kullanılarak, ortalama çalışma süresi en yüksek olan ilk 5 sorgu tespit edilmiştir.

63763 ms ortalama süreyle en yavaş sorgu, veritabanı adını ve boyutunu getiren bir SELECT işlemidir. 10761 ms ortalama süreye sahip ikinci sorgu, bu analizin kendisini gerçekleştiren sorgudur.

Diğer sorgular, sistem tablolarından bilgi çeken işlemlerdir.

```
-- Ortalama çalışma süresi en yüksek olan sorguları getir
SELECT TOP 5
    total_elapsed_time / execution_count AS AvgTime, -- Ortalama süre
    execution_count, -- Çalışma sayısı
    total_elapsed_time, -- Toplam süre
    SUBSTRING(qt.text,
        (qs.statement_start_offset / 2) + 1,
        ((CASE qs.statement_end_offset
            WHEN -1 THEN DATALENGTH(qt.text)
            ELSE qs.statement_end_offset
        END - qs.statement_start_offset) / 2) + 1) AS QueryText -- Sorgu metni
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) qt
ORDER BY AvgTime DESC;
```

100 %

Results

	AvgTime	execution_count	total_elapsed_time	QueryText
1	63763	1	63763	SELECT db.name AS [Name], CAST(0 AS bit) AS [IsFa...
2	10761	4	43046	SELECT TOP 5 total_elapsed_time / execution_cou...
3	3603	1	3603	SELECT clmn.name AS [Name], clmn.column_id AS...
4	381	1	381	SELECT db.collation_name AS [Collation], db.name A...

Şekil 15 En Yavaş Sorguların Tespiti

2.9 Tablo Boyutları ve Disk Kullanımı

The screenshot shows the Object Explorer on the left with the database 'KutuphaneDB' selected. The central pane displays a T-SQL query named 'SQLQuery8.sql' which retrieves information about table sizes and disk usage. The results are shown in a table at the bottom.

```
-- Tablolardan disk kullanım bilgilerini ve satır sayılarını getirir
SELECT
    t.name AS TableName,
    SUM(p.rows) AS RowCounts,
    SUM(a.total_pages) * 8 AS TotalSpaceKB,
    SUM(a.used_pages) * 8 AS UsedSpaceKB
FROM sys.tables t
INNER JOIN sys.indexes i ON t.object_id = i.object_id
INNER JOIN sys.partitions p ON i.object_id = p.object_id AND i.index_id = p.index_id
INNER JOIN sys.allocation_units a ON p.partition_id = a.container_id
GROUP BY t.name
ORDER BY TotalSpaceKB DESC; -- En fazla yer kaplayanlar önce gelir
```

TableName	RowCounts	TotalSpaceKB	UsedSpaceKB
Borrowings	40	144	32
Readers	20	72	16
Authors	20	72	16
Books	20	72	16

Şekil 16 Tablo Boyutları ve Disk kullanımı

Kısıtlı Erişimli Kullanıcı Hesabı Oluşturma

Şekil 17, Kütüphane veritabanı sisteminde güvenliği sağlamak amacıyla oluşturulan kısıtlı erişimli bir kullanıcı hesabını göstermektedir.

Yapılan İşlemler:

1. **Login Oluşturma:**
 - o "Genikkullanıcı" adında yeni bir SQL Server login hesabı oluşturulmuştur.
 - o Güvenli bir parola belirlenmiştir (Password23).
2. **Veritabanı Kullanıcısı Tanımlama:**
 - o Oluşturulan login, Kütüphane veritabanı için bir kullanıcı hesabına bağlanmıştır.
3. **Yetkilendirme:**
 - o Kullanıcıya sadece db_datareader rolü atanarak yalnızca veri okuma (SELECT) yetkisi verilmiştir.
 - o Bu, kullanıcının verileri görüntüleyebilmesini ancak değiştirememesini sağlar.

```

-- Yeni kullanıcı için login oluştur
CREATE LOGIN Ceninkullanici WITH PASSWORD = 'Password23'; -- Güvenli şifre

-- Kütüphane veritabanı için kullanıcı oluştur
CREATE USER Ceninkullanici FOR LOGIN Ceninkullanici;

-- Sadece okuma yetkisi ver (db_datareader rolüne ekleme)
EXEC sp_addrolemember 'db_datareader', 'Ceninkullanici';

```

Commands completed successfully.

Completion time: 2025-04-25T06:56:46.1845501+03:00

Şekil 17 Hesap oluşturma

Readers Tablosundan Kullanıcı Verilerinin Çekilmesi

```

USE KutuphaneDB
GO
SELECT * FROM Readers

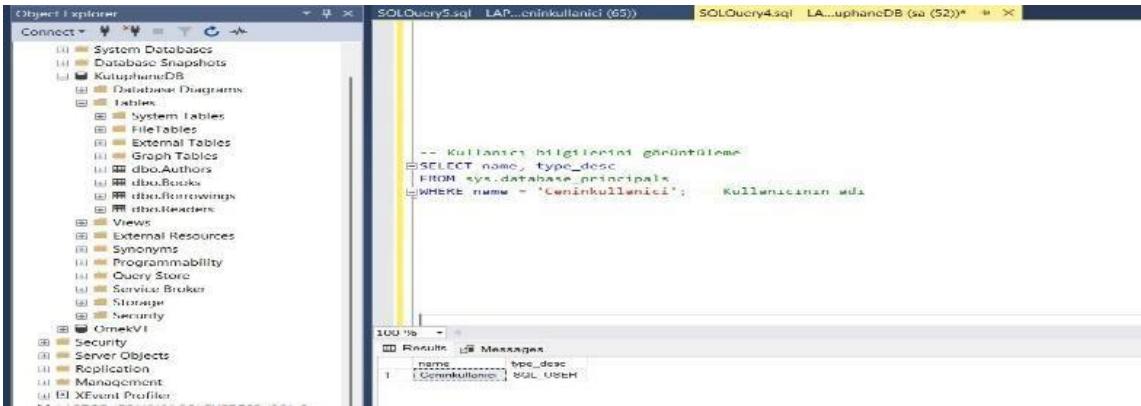
```

ReaderID	FullName	Address	Email	Phone
1	Aylin Demir	Kadıköy, İstanbul	aylin.demir@example.com	+905321112233
2	Mehmet Yıldız	Besiktas, İstanbul	mehmet.yildiz@example.com	+905324445566
3	Zeynep Kaya	Çankaya, Ankara	zeynep.kaya@example.com	+905326778899
4	Ahmet Kurt	Beylikdüzü, İstanbul	ahmet.kurt@example.com	+905329990011
5	Elif Öz	Ataşehir, İstanbul	elif.oz@example.com	+905321223344
6	Cem Karaca	Kızılay, Ankara	cem.karaca@example.com	+905325556677
7	Buse Tan	Üsküdar, İstanbul	buse.tan@example.com	+905328899900
8	Deniz Arslan	Sisli, İstanbul	deniz.arslan@example.com	+905321234567
9	Selin Dogan	Fatih, İstanbul	selin.dogan@example.com	+905322345678
10	Emir Koç	Kadıköy, İstanbul	emir.koc@example.com	+905323456789
11	Nazlı Sahin	Çankaya, Ankara	nazli.sahin@example.com	+905324567890
12	Berk Yılmaz	Pendik, İstanbul	berk.yilmaz@example.com	+905325678901
13	Derya Acar	Kartal, İstanbul	derya.acar@example.com	+905326789012
14	Okan Güneş	Keciören, Ankara	okan.gunes@example.com	+905327890123
15	Irem Polat	Ümraniye, İstanbul	irem.polat@example.com	+905328901234
16	Ali Vural	Göztepe, İstanbul	ali.vural@example.com	+905329012345
17	Murat Ercan	Marmaritöre, İstanbul	murat.ercan@example.com	+905321008768

Şekil 18 Kullanıcı Verilerinin Çekilmesi

Kullanıcı Yetki Kontrolü

Bu görsel, KutuphaneDB veritabanında "Ceninkullanici" adlı kullanıcının varlığını ve temel özelliklerini doğrulamak için yapılan bir sistem sorgusunu göstermektedir.



The screenshot shows the Object Explorer on the left with the 'KutuphaneDB' database selected. On the right, two queries are running in separate panes:

```
-- Kullanıcı bilgilerini görüntüleme
SELECT name, type_desc
FROM sys.database_principals
WHERE name = 'Ceninkullanici';
```

The results pane shows the following output:

name	type_desc
Ceninkullanici	SYSUSER

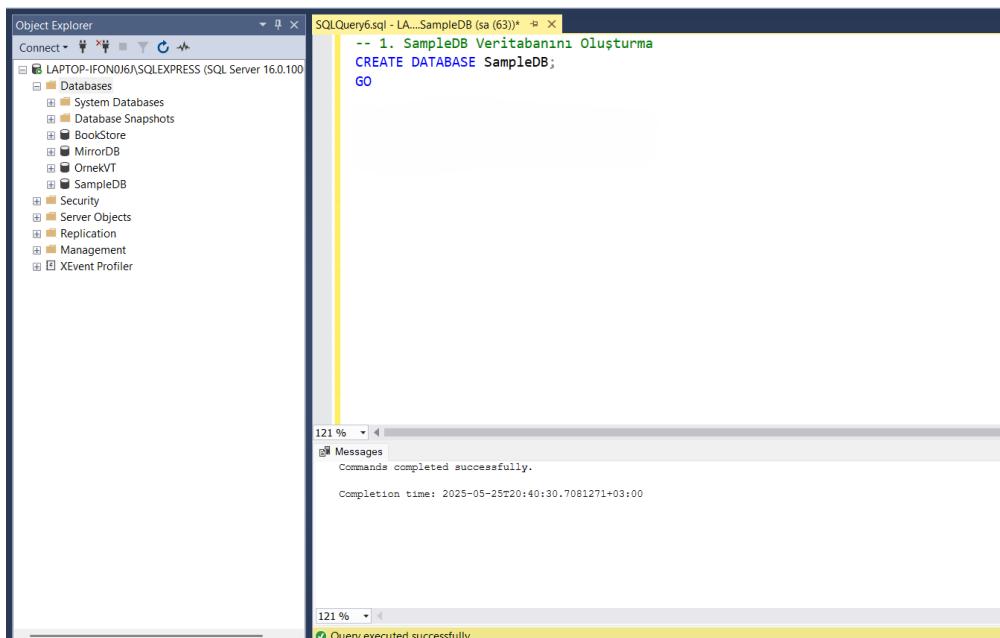
Şekil 19 Kullanıcı Yetki Verme

3 Veritabanı Yedekleme ve Felaketten Kurtarma Planı

SQL Server üzerinde yürütülen bu çalışmada, tam, fark ve artık (transaction log) yedekleme teknikleri kullanılarak veri güvenliğini sağlayan stratejiler geliştirilmiştir. Ayrıca, point-in-time restore yöntemiyle belirli bir zamana geri dönmeye ve database mirroring ile yedek sistem senaryoları örneklerle uygulanmıştır.

3.1 SampleDB ve MirrorDB

- **SampleDB:** Ana veritabanı (ürtim sistemi gibi).



The screenshot shows the Object Explorer on the left with the 'LAPTOP-IFON0J6\SQLEXPRESS' server selected. On the right, a query window titled 'SQLQuery6.sql - LA...SampleDB (sa (63))' contains the following command:

```
-- 1. SampleDB Veritabanını Oluşturma
CREATE DATABASE SampleDB;
GO
```

The results pane shows the message: 'Commands completed successfully.' and 'Completion time: 2025-05-25 20:40:30.7081271+03:00'. A status bar at the bottom indicates 'Query executed successfully.'

Örnek Tablo ve Veri

SampleDB veritabanına Employees adlı bir tablo oluşturulmuş ve çeşitli departmanlardan çalışan verileri eklenmiştir.

The screenshot shows the Object Explorer on the left and a query window on the right. The query window contains the following T-SQL code:

```
-- 3. SampleDB Veritabanına Employees Tablosu Ekleme
USE SampleDB;
GO
CREATE TABLE Employees (
    EmployeeID INT PRIMARY KEY IDENTITY(1,1),
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    Department NVARCHAR(50),
    HireDate DATE,
    Salary DECIMAL(10, 2)
);
GO

-- 4. Employees Tablosuna Örnek Veriler Ekleme
INSERT INTO Employees (FirstName, LastName, Department, HireDate, Salary)
VALUES
    ('Ali', 'Yılmaz', 'IT', '2023-01-15', 7500.00),
    ('Ayşe', 'Demir', 'HR', '2022-06-20', 6500.00),
    ('Mehmet', 'Kara', 'Finance', '2021-03-10', 8000.00),
    ('Fatma', 'Çelik', 'Marketing', '2024-02-01', 7000.00),
    ('Ahmet', 'Şahin', 'IT', '2023-11-12', 7200.00);
GO
```

The status bar at the bottom indicates "Query executed successfully."

3.2 Yedekleme Stratejileri

1.1. Kurtarma Modeli:

Veritabanı kurtarma modeli FULL olarak ayarlanmıştır. Bu sayede transaction log yedekleri alınabilir.

ALTER DATABASE SampleDB SET RECOVERY FULL;

1.2. Yedekleme Türleri

1.2.1. Tam (Full) Yedek

The screenshot shows the Object Explorer on the left and a query window on the right. The query window contains the following T-SQL command:

```
-- 5. Orijinal Yedekleme ve Veri Yedekleme Komutları
-- 5.1 SampleDB için Tam Yedekleme
BACKUP DATABASE [SampleDB]
TO DISK = 'C:\Backup\SampleDB_FullBackup.bak'
WITH NOLOG, FILE = 'N:SampleDB_Full', Database Backup,
    SKIP, NORECOVERY, STATS = 10;
GO
```

The status bar at the bottom indicates "Query executed successfully."

1.2.2. Fark (Differential) Yedek

The screenshot shows the Object Explorer on the left and a SQL Query window on the right. The query window contains the following T-SQL code:

```
-- 5.2 SampleDB için Fark Yedekleme
-- SampleDB veritabanında tam yedekten sonra değişen verilerin fark yedeğini alır.
BACKUP DATABASE SampleDB
TO DISK = N'C:\Backup\SampleDB_DiffBackup.bak'
WITH DIFFERENTIAL,
NOFORMAT, NOINIT,
NAME = N'SampleDB-Differential Database Backup',
SKIP, NOREWIND, NOUNLOAD, STATS = 10;
GO
```

The Messages pane at the bottom shows the backup progress and completion details:

```
121 % 121 %
12 percent processed.
21 percent processed.
33 percent processed.
42 percent processed.
50 percent processed.
63 percent processed.
71 percent processed.
80 percent processed.
92 percent processed.
100 percent processed.
Processed 200 pages for database 'SampleDB', file 'SampleDB' on file 1.
Processed 1 pages for database 'SampleDB', file 'SampleDB_log' on file 1.
BACKUP DATABASE WITH DIFFERENTIAL successfully processed 201 pages in 0.022 seconds (71.089 MB/sec).

Completion time: 2025-05-25T20:56:47.5825232+03:00
121 % 121 %
Query executed successfully.
```

1.2.3. Artık (Transaction Log) Yedek

The screenshot shows the Object Explorer on the left and a SQL Query window on the right. The query window contains the following T-SQL code:

```
-- 5.5 SampleDB için Artık (Transaction Log) Yedekleme
-- SampleDB veritabanının transaction log'larını yedekler, veri değişikliklerinin kaydını tutar.
BACKUP LOG SampleDB
TO DISK = N'C:\Backup\SampleDB_LogBackup.trn'
WITH NOFORMAT, NOINIT,
NAME = N'SampleDB-Transaction Log Backup',
SKIP, NOREWIND, NOUNLOAD, STATS = 10;
GO
```

The Messages pane at the bottom shows the backup completion details:

```
121 % 121 %
100 percent processed.
Processed 1 pages for database 'SampleDB', file 'SampleDB_log' on file 1.
BACKUP LOG successfully processed 1 pages in 0.006 seconds (0.976 MB/sec).

Completion time: 2025-05-25T21:10:36.4017104+03:00
121 % 121 %

```

3.3 Felaket Senaryosu: Point-in-Time Recovery

Senaryo:

25 Mayıs 2025, saat 22:30'da veritabanına zarar verildi. En son yedekten bu zamana kadar olan tüm işlemler geri yüklenmelidir.

Aşamalar:

1. Veritabanını Tek Kullanıcı Moduna Alma:

The screenshot shows the SSMS interface. The Object Explorer on the left lists databases including BookStore, MirrorDB, OrnekVT, and SampleDB. The right pane contains a query window titled 'SQLQuery6.sql - LAP...SampleDB (sa (63))'. The query is:

```
-- 5.6 SampleDB için Point-in-Time Restore Senaryosu
-- 5.6.1 Tek Kullanıcı Moduna Al
-- Veritabanını tek kullanıci moduna alır, tüm aktif bağlantıları keser.
ALTER DATABASE SampleDB SET SINGLE_USER WITH ROLLBACK IMMEDIATE;
GO
```

The 'Messages' pane at the bottom shows the command completed successfully.

2. Kuyruk Log Yedeği (Tail Log) Alma:

The screenshot shows the SSMS interface. The Object Explorer on the left lists databases including BookStore, MirrorDB, OrnekVT, and SampleDB. The right pane contains a query window titled 'SQLQuery6.sql - LAP...SS.master (sa (63))'. The query is:

```
-- 5.6.2 Transaction Log Yedeği Alma (Log kuyruğunu yedekler)
-- Veritabanının log kuyruğunu yedekler, mevcut logları korur.
BACKUP LOG SampleDB
TO DISK = N'C:\Backup\SampleDB_TailLogBackup.trn'
WITH NORECOVERY;
GO
```

The 'Messages' pane at the bottom shows the backup process completed successfully.

3. Tam Yedeği Geri Yükle (NORECOVERY):

The screenshot shows the Object Explorer on the left and a query window on the right. The query window contains the following T-SQL code:

```
-- 5.6.2 Full Backup Geri Yükle (NORECOVERY)
-- Tam yedeği geri yükler, ancak işlemi tamamlamaz, sonraki yedeklerin uygulanması için bekler.
RESTORE DATABASE SampleDB
FROM DISK = N'C:\Backup\SampleDB_FullBackup.bak'
WITH NORECOVERY;
GO
```

The Messages pane at the bottom shows the execution results:

```
Processed 552 pages for database 'SampleDB', file 'SampleDB' on file 1.
Processed 1 pages for database 'SampleDB', file 'SampleDB_log' on file 1.
RESTORE DATABASE successfully processed 553 pages in 0.024 seconds (179.748 MB/sec).

Completion time: 2025-05-25T21:26:53.5908199+03:00
```

A green status bar at the bottom indicates "Query executed successfully."

4. Fark Yedeği Geri Yükle (Varsa):

The screenshot shows the Object Explorer on the left and a query window on the right. The query window contains the following T-SQL code:

```
-- 5.6.4 Fark Yedeği Geri Yükle (Varsa, NORECOVERY)
-- Fark yedeğini geri yükler, yine işlemi tamamlamaz, log yedeği için bekler.
RESTORE DATABASE SampleDB
FROM DISK = N'C:\Backup\SampleDB_DiffBackup.bak'
WITH NORECOVERY;
GO
```

The Messages pane at the bottom shows the execution results:

```
Processed 200 pages for database 'SampleDB', file 'SampleDB' on file 1.
Processed 1 pages for database 'SampleDB', file 'SampleDB_log' on file 1.
RESTORE DATABASE successfully processed 201 pages in 0.010 seconds (156.396 MB/sec).

Completion time: 2025-05-25T21:27:33.6972755+03:00
```

A green status bar at the bottom indicates "Query executed successfully."

5. Log Yedeğini Geri Yükle (Belirli Zamana Kadar):

```
RESTORE LOG SampleDB
FROM DISK = 'C:\Backup\SampleDB_LogBackup.trn'
WITH STOPAT = 'TIME',
RECOVERY;
```

```

Object Explorer
Connect ▾
LAPTOP-IFON0J6\SQLEXPRESS
  Databases
  Security
  Server Objects
  Replication
  Management
  XEvent Profiler

SQLQuery6.sql - LAP...SS.master (sa (65))*
RESTORE DATABASE SampleDB
FROM DISK = N'C:\Backup\SampleDB_FullBackup.bak'
WITH NORECOVERY;
GO

-- Log Yedeği Geri Yükle (Belirli Zamana Kadar, RECOVERY)
-- Transaction log yedegini belirlenen zamana kadar geri yükler ve işlemi tamamlar (veritabanı açılır).
RESTORE LOG SampleDB
FROM DISK = N'C:\Backup\SampleDB_LogBackup.trn'
WITH STOPAT = '2025-05-26T19:48:29',
      RECOVERY;
GO

--Veritabanını Çok Kullanıcı Moduna Geri Alma
ALTER DATABASE SampleDB SET MULTI_USER;
GO

--Geri yüklenen verileri kontrol et: Employees tablosundaki verileri görüntüle
SELECT * FROM Employees;

--Veritabanındaki tüm kullanıcı tablolarını liste
SELECT name
FROM sys.tables;

--Tam yedek dosyasının içeriği dosyaları ve detaylarını liste
RESTORE FILELISTONLY
FROM DISK = 'C:\Backup\SampleDB_FullBackup.bak';
100 %
  Messages
Msg 4312, Level 16, State 1, Line 101
This log cannot be restored because a gap in the log chain was created. Use more recent data backups to bridge the gap.
Msg 3018, Level 16, State 1, Line 101
RESTORE LOG is terminating abnormally.

Completion time: 2025-05-26T21:54:15.8722659+03:00

```

6. Çok Kullanıcılı Moda Alma:

ALTER DATABASE SampleDB SET MULTI_USER;

```

Object Explorer
Connect ▾
LAPTOP-IFON0J6\SQLEXPRESS
  Databases
  Security
  Server Objects
  Replication
  Management
  XEvent Profiler

SQLQuery6.sql - LAP...SS.master (sa (65))*
-- Transaction log yedegini belirlenen zamana kadar geri yükler ve işlemi tamamlar (veritabanı açılır).
RESTORE LOG SampleDB
FROM DISK = N'C:\Backup\SampleDB_LogBackup.trn'
WITH STOPAT = '2025-05-26T19:48:29',
      RECOVERY;
GO

--Veritabanını Çok Kullanıcı Moduna Geri Alma
ALTER DATABASE SampleDB SET MULTI_USER;
GO

--Geri yüklenen verileri kontrol et: Employees tablosundaki verileri görüntüle
SELECT * FROM Employees;

--Veritabanındaki tüm kullanıcı tablolarını liste
SELECT name
FROM sys.tables;

--Tam yedek dosyasının içeriği dosyaları ve detaylarını liste
RESTORE FILELISTONLY
FROM DISK = 'C:\Backup\SampleDB_FullBackup.bak';

--Log yedek dosyasının başlık bilgilerini kontrol et
RESTORE HEADERONLY
FROM DISK = 'C:\Backup\SampleDB_LogBackup.trn';
100 %
  Messages
Msg 5052, Level 16, State 1, Line 108
ALTER DATABASE is not permitted while a database is in the Restoring state.
Msg 5069, Level 16, State 1, Line 108
ALTER DATABASE statement failed.

Completion time: 2025-05-26T21:54:33.5737394+03:00

```

3.4 Test ve Doğrulama

- SELECT * FROM Employees sorgusu ile geri yüklenen veriler kontrol edildi.
- sys.tables üzerinden tablolar listelendi.

- Yedek dosyalarının içeriği ve boyutları karşılaştırıldı.
- Geri yükleme işlemlerinde **STATS = 10** opsyonu ile ilerleme takibi yapıldı.

The screenshot shows the Object Explorer on the left with a connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. A database named 'SampleDB' is selected under 'Databases' and is shown as 'Restoring...'. The central pane displays a query window titled 'SQLQuery6.sql - LAP...SS.master (sa (65))'. The query contains several comments and T-SQL commands for database restoration:

```
-- Veritabanını Çok Kullanıcı Moduna Geri Alma
ALTER DATABASE SampleDB SET MULTI_USER;
GO

-- Geri yüklenen verileri kontrol et: Employees tablosundaki verileri görüntüle
SELECT * FROM Employees;

-- Veritabanındaki tüm kullanıcı tablolarını listele
SELECT name
FROM sys.tables;

-- Tam yedek dosyasının içeriği dosyaları ve detaylarını listele
RESTORE FILELISTONLY
FROM DISK = 'C:\Backup\SampleDB_FullBackup.bak';

-- Log yedek dosyasının başlık bilgilerini kontrol et
RESTORE HEADERONLY
FROM DISK = 'C:\Backup\SampleDB_LogBackup.trn';

-- Geri yükleme işlemi sırasında %10'luk ilerleme göstergesiyle takip et
RESTORE DATABASE SampleDB
FROM DISK = 'C:\Backup\SampleDB_FullBackup.bak'
WITH NORECOVERY, STATS = 10;
;
```

The 'Results' tab at the bottom shows the output of the 'SELECT name...' query:

	name
1	spt_fallback_db
2	spt_fallback_dev
3	spt_fallback_usg
4	LogKayitları
5	LogKullanicilar

4 Veritabanı Güvenliği ve Erişim Kontrolü

Güvenli veritabanı oluşturma, erişim yetkilendirmesi, veri şifreleme, SQL Injection'a karşı koruma ve kullanıcı aktivitelerinin izlenmesi gibi birçok güvenlik başlığı altında işlemler gerçekleştirilmiştir.

4.1 Veri tabanı Oluşturma Komutları

The screenshot shows the Object Explorer on the left with a connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. A database named 'OrnekVT' is selected under 'Databases'. The central pane displays a query window titled 'SQLQuery1.sql - LA...S.OrnekVT (sa (68))'. The query contains T-SQL commands for creating a database and a table:

```
CREATE DATABASE OrnekVT;
GO
USE OrnekVT;
GO

CREATE TABLE Musteriler (
    MusteriID INT PRIMARY KEY IDENTITY(1,1),
    AdSoyad NVARCHAR(100),
    Email NVARCHAR(100)
);
GO
```

The 'Messages' tab at the bottom shows the completion message:

Commands completed successfully.
Completion time: 2025-04-23T19:54:50.8728140+03:00

Şekil 1 Veri Tabanı Oluşturma Komutları

4.2 Örnek Tablo Yapıları

The screenshot shows the Object Explorer on the left with a connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. Under 'Databases', 'OrnekVT' is selected. The center pane displays a query window titled 'SQLQuery1.sql - LA...S.OrnekVT (sa (68))'. The query creates a database 'OrnekVT' and a table 'Musteriler' with columns MusteriID, AdSoyad, and Email. The 'Messages' pane at the bottom shows the command completed successfully.

```
CREATE DATABASE OrnekVT;
GO
USE OrnekVT;
GO
CREATE TABLE Musteriler (
    MusteriID INT PRIMARY KEY IDENTITY(1,1),
    AdSoyad NVARCHAR(100),
    Email NVARCHAR(100)
);
GO
```

102 %

Messages
Commands completed successfully.
Completion time: 2025-04-23T19:54:50.8728140+03:00

Şekil 2 Örnek Tablo Yapıları

4.3 Kimlik Doğrulama ve Yetki Yönetimi

4.3.1 SQL Server ve Windows Authentication

- SQL Server Authentication ve Windows Authentication yöntemleriyle kullanıcı oturum açma sistemlerinin karşılaştırılması ve kullanılması.
- Yeni kullanıcı hesapları oluşturulacak (CREATE LOGIN, CREATE USER) ve kullanıcı yetkilendirmesi yapılacak (GRANT, DENY, REVOKE komutları).
- Role-based access control (RBAC) uygulanacak; kullanıcılar özel rollerle sınıflandırılacak.

4.3.2 SQL Server Authentication Kullanıcısı Oluşturma

SQL Server Authentication kullanabilmek için sunucunun mixed mode (SQL Server and Windows Authentication mode) olarak yapılandırılmış olması gereklidir. Bu ayar SQL Server Management Studio (SSMS) üzerinden "Server Properties > Security" kısmından yapılır.

Kullanıcı oluşturmak için aşağıdaki komut kullanılır:

The screenshot shows the Object Explorer on the left with a connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. Under 'Tables', 'dbo.Musteriler' is selected. The center pane displays a query window titled 'SQLQuery1.sql - LA...S.OrnekVT (sa (51))'. The query creates a login 'Umutkullanici' with password 'Guv3nliP@rola!', a database user 'Umutkullanici' for the 'master' database, and grants public role to the user. The 'Results' pane at the bottom shows the created user information.

```
-- Giriş (login) oluştur
CREATE LOGIN Umutkullanici WITH PASSWORD = 'Guv3nliP@rola!';
-- Veritabanı kullanıcısı oluştur
USE OrnekVT;
GO
CREATE USER Umutkullanici FOR LOGIN Umutkullanici;
GO
```

UserName	RoleName	LoginName	DefDBName	DefSchemaName	UserID	SID
Umutkullanici	public	Umutkullanici	master	dbo	7	0x071D720F2C6EFE43A9211A50F083B39

Şekil 3 Veri Tabanı Kullanıcısı Oluşturma (SQL Server Authentication)

4.3.3 Windows Authentication Kullanıcısı Oluşturma

Windows Authentication yöntemi, kullanıcıların mevcut Windows oturum bilgileriyle SQL Server'a bağlanmasına olanak tanır. Bu yöntem, merkezi kullanıcı yönetimi ve daha güvenli kimlik doğrulama sağladığı için kurumsal ortamlarda yaygın olarak tercih edilir.

The screenshot shows the Object Explorer on the left with the connection set to 'LAPTOP-IFON0J6\SQLEXPRESS'. In the center, a query window titled 'SQLQuery1.sql' contains the following T-SQL code:

```
-- Veritabanı kullanıcıyı olarak ekle  
USE OrnekVT;  
GO  
CREATE USER [laptop-ifon0j6\asus] FOR LOGIN [laptop-ifon0j6\asus];  
GO
```

Below the query window, the 'Messages' pane displays the output:

```
Commands completed successfully.  
Completion time: 2025-04-23T21:08:34.5432610+03:00
```

Şekil 4 Veri Tabanı Kullanıcısı Oluşturma (Windows Authentication)

4.4 Yetki Verme (GRANT, DENY, REVOKE)

Projede, kullanıcıların sadece yetkili oldukları işlemleri yapabilmesi için **GRANT**, **DENY** ve **REVOKE** komutlarıyla yetki yönetimi sağlanmış; ayrıca kullanıcılar, rollere atanarak toplu yetkilendirme yapılmıştır.

4.4.1 GRANT Komutu:

Belirli bir kullanıcıya veya role bir izin (örneğin SELECT, INSERT) vermek için kullanılır.

The screenshot shows the Object Explorer on the left with the connection set to 'LAPTOP-IFON0J6\SQLEXPRESS'. In the center, a query window titled 'LAPTOP-IFON0J6\SQLEXPRESS - dbo.Musteri' contains the following T-SQL code:

```
GRANT SELECT ON Musteri TO Umutkullanici; -- Sadece veri okuma yetkisi ver  
SELECT * FROM Musteri;
```

Below the query window, the 'Results' pane displays the output:

MusteriID	AdSoyad	Email
1	Ali Yılmaz	ali.yilmaz@example.com
2	Ayse Demir	ayse.demir@example.com
3	Mehmet Kaya	mehmet.kaya@example.com
4	Cenin Rihavi	tanrikhaw45@gmail.com
5	Umut Akyubek kyz	umutaky22@gmail.com

Şekil 5 GRANT Komutu

The screenshot shows the Object Explorer on the left and three query panes on the right. The top pane contains the following SQL code:

```
USE OrnekVT;
INSERT INTO Musteri (AdSoyad, Email) VALUES ('Test Deneme', 'test@ornek.com');
SELECT * FROM Musteri;
```

The bottom pane shows the results of the SELECT statement:

MusteriID	AdSoyad	Email
1	Ali Yilmaz	ali.yilmaz@example.com
2	Ayse Demir	ayse.demir@example.com
3	Mehmet Kaya	mehmet.kaya@example.com
4	Cenin Rihavi	tarikrihawi45@gmail.com
5	Umut Akyibek kyz	umutaky22@gmail.com
6	Test Deneme	test@ornek.com

Şekil 6 GRANT Komutunu Test Etme

4.4.2 DENY Komutu:

Belirli bir kullanıcıya veya role bir izni açıkça reddetmek için kullanılır. GRANT'tan üstündür.

The screenshot shows the Object Explorer on the left and three query panes on the right. The middle pane contains the following SQL code:

```
DENY INSERT ON Musteri TO Umutkullanici; -- Veri ekleme yetkisini engelle (DENY INSERT)
```

The bottom pane shows the message:

Commands completed successfully.
Completion time: 2025-04-24T16:10:24.3613787+03:00

Şekil 7 DENY Komutu

The screenshot shows the Object Explorer on the left and three query panes on the right. The middle pane contains the following SQL code:

```
USE OrnekVT;
INSERT INTO Musteri (AdSoyad, Email) VALUES ('Test Deneme', 'test@ornek.com');
```

The bottom pane shows the error message:

Msg 229, Level 14, State 5, Line 2
The INSERT permission was denied on the object 'Musteri', database 'OrnekVT', schema 'dbo'.
Completion time: 2025-04-24T16:10:45.6333323+03:00

Şekil 8 DENY Komutunu Test Etme

4.4.3 REVOKE Komutu:

Daha önce verilmiş olan bir izni veya reddedilmiş bir izni **geri almak** için kullanılır.

The screenshot shows the SSMS interface. On the left, the Object Explorer displays two servers: 'LAPTOP-IFON0J6\SQLEXPRESS' and 'LAPTOP-IFON0J6\SQLEXPRESS'. Under the first server, 'Security' is expanded, showing 'Databases', 'Server Objects', 'Replication', 'Management', and 'XEvent Profiler'. Under 'Databases', 'OrnekVT' is selected. On the right, a query window titled 'SQLQuery2.sql - LAPTOP-IFON0J6\Umutkullanici (75)*' contains the following SQL command:

```
-- INSERT yetkisini kaldır
REVOKE INSERT ON Musteri FROM Umutkullanici;
```

The status bar at the bottom indicates 'Commands completed successfully.' and the completion time: 'Completion time: 2025-04-24T16:14:50.9091315+03:00'.

Şekil 9 REVOKE Komutu

4.5 Role-Based Access Control (RBAC)

SQL Server'daki yerleşik rollerden bazıları:

- db_datareader: tüm tabloları okuma yetkisi
- db_datawriter: tüm tablolara yazma yetkisi
- db_owner: tüm işlemleri yapma yetkisi

```

EXEC sp_addrolemember 'db_datareader', 'Umutkullanici'; -- sql_kullanici kullanıcısını sadece okuma rolüne ekle
EXEC sp_addrolemember 'db_owner', 'laptop-ifon0j6j\asus'; -- Windows kullanıcısına tam yetki ver

```

Şekil 10 Role-Based Access Control (RBAC)

UserName	RoleName	LoginName	DefDBName	DefSchemaName	UserID	SID
Umutkullanici	db_datareader	Umutkullanici	master	dbo	7	0x1598D080FB3BB54D83AEE53178B3E3F4

Şekil 11 SQL Server Authentication Kullanıcı Rol Adı

UserName	RoleName	LoginName	DefDBName	DefSchemaName	UserID	SID
laptop-ifon0j6j\asus	db_owner	NULL	NULL	dbo	6	0x0105000000000005150000001836088BD7725CC0578...

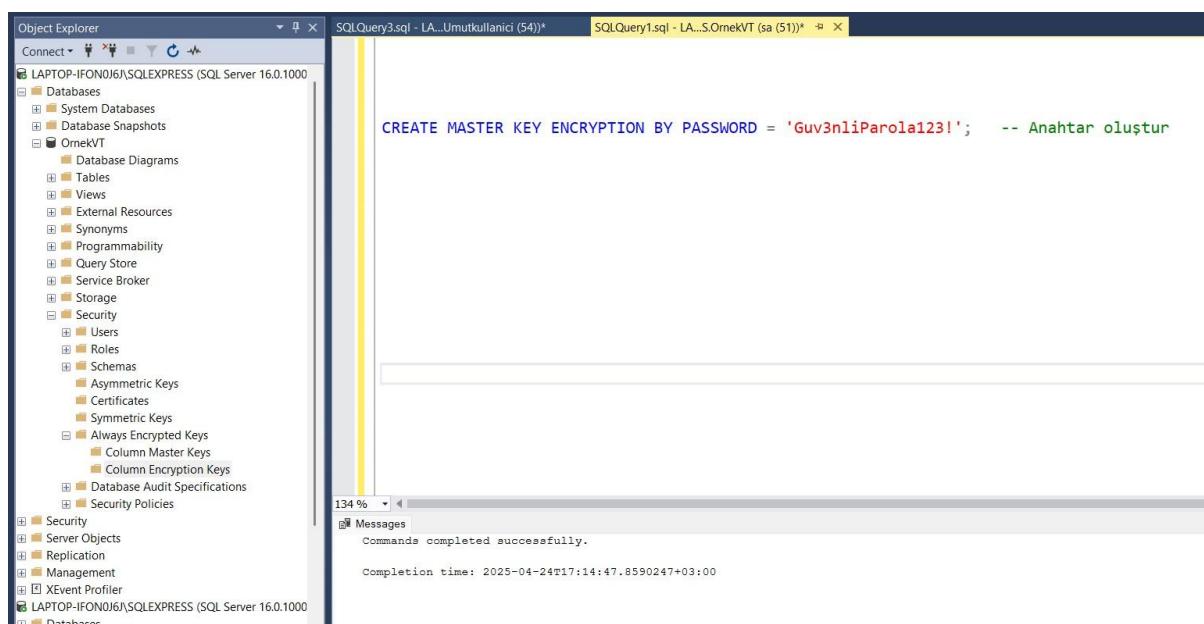
Şekil 12 Windows Authentication Kullanıcı Rol Adı

4.6 VERİ ŞİFRELEME

4.6.1 Şifreleme Yöntemlerinin Karşılaştırılması (TDE vs. Kolon Bazlı)

Projede veri şifreleme için öncelikle **Transparent Data Encryption (TDE)** yöntemi planlanmış olsa da kullanılan SQL Server sürümünün **Standard Edition** olması nedeniyle bu özellik desteklenmemektedir. TDE yalnızca Enterprise, Developer veya Azure gibi gelişmiş sürümlerde aktif olarak kullanılabilir. Bu nedenle, proje kapsamında alternatif bir yöntem olarak **kolon bazlı veri şifreleme** tekniği tercih edilmiştir.

Kolon bazlı şifrelemede, hassas veriler (örneğin müşteri isimleri veya e-posta adresleri) veritabanına kaydedilmeden önce özel bir sertifika ile şifrelenir. Bu amaçla önce bir **Master Key** ve ardından bir **sertifika** oluşturulur. Veriler EncryptByCert fonksiyonu ile şifrelenecek saklanır ve DecryptByCert ile çözümlenebilir. Bu yöntem sayesinde, özellikle TDE desteklenmeyen ortamlarda veri güvenliği sağlanmış olur.

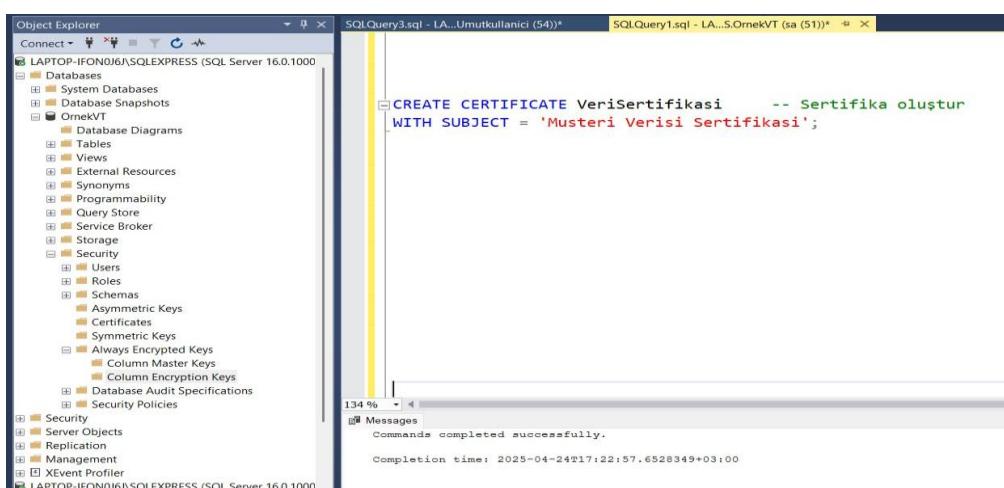


The screenshot shows the Object Explorer on the left and two query panes on the right. The Object Explorer lists databases, security, and server objects. The top query pane contains the command:

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Guv3nliParola123!'; -- Anahtar oluştur
```

The bottom query pane shows the message "Commands completed successfully." and the completion time: 2025-04-24T17:14:47.8590247+03:00.

Şekil 13 Anahtar Oluşturma



The screenshot shows the Object Explorer on the left and two query panes on the right. The top query pane contains the command:

```
CREATE CERTIFICATE VeriSertifikasi -- Sertifika oluştur  
WITH SUBJECT = 'Müşteri Verisi Sertifikası';
```

The bottom query pane shows the message "Commands completed successfully." and the completion time: 2025-04-24T17:22:57.6528349+03:00.

Şekil 14 Sertifika Oluşturma

The screenshot shows the Object Explorer on the left and the SQL Query Editor on the right. In the Object Explorer, the database 'OrnekVT' is selected. In the SQL Query Editor, the following SQL code is run:

```
CREATE TABLE SifreliMusteriler (      -- Şifreli tablo oluştur
    ID INT IDENTITY,
    AdSoyad VARBINARY(MAX)
);
```

The execution completed successfully with the message "Commands completed successfully." and a completion time of "2025-04-24T17:23:32.7068532+03:00".

Sekil 15 Şifreli Tablo Oluşturma

The screenshot shows the Object Explorer on the left and the SQL Query Editor on the right. In the Object Explorer, the database 'OrnekVT' is selected. In the SQL Query Editor, the following SQL code is run:

```
INSERT INTO SifreliMusteriler (AdSoyad)      -- Veri ekleme (şifreli)
VALUES (EncryptByCert(Cert_ID('VeriSertifikasi')), 'Ahmet Yılmaz');
```

The execution completed successfully with the message "(1 row affected)" and a completion time of "2025-04-24T17:24:58.6894592+03:00".

Sekil 16 Şifreli Tabloya Veri Ekleme

The screenshot shows the Object Explorer on the left with 'LAPTOP-IFON0J6\SQLEXPRESS' selected. In the center, there are two tabs: 'SQLQuery3.sql - LA...Umurtkullanici (54)*' and 'SQLQuery1.sql - LA...S.OrnekVT (sa (51))*'. The 'SQLQuery1.sql' tab is active, displaying the following T-SQL code:

```

SELECT CONVERT(NVARCHAR, DecryptByCert(Cert_ID('VeriSertifikasi'), AdSoyad)) -- Veri okuma (decryption)
FROM SifreliMusteriler;

```

The results pane at the bottom shows one row with the value '布魯斯·威爾斯'.

Şekil 17 Şifreli Tablodan Veri Okuma

4.7 SQL INJECTION GÜVENLİĞİ

SQL Injection Nedir?

SQL Injection, kötü niyetli kullanıcıların, uygulama aracılığıyla veritabanına gönderilen SQL sorgularına zararlı komutlar ekleyerek sisteme izinsiz erişim sağlamaya çalıştığı bir saldırı türüdür. Bu saldırı tekniği, kullanıcı girdilerinin doğru şekildefiltrelenmemesi durumunda veritabanında veri sızdırma, silme veya sistem ele geçirme gibi ciddi güvenlik açıklarına yol açabilir.

4.7.1 Güvensiz Sorgu Örneği ve Analizi

Güvensiz sorgular, kullanıcı girdisinin doğrudan SQL komutlarına eklenmesiyle oluşturulan sorgulardır. Bu yöntem SQL Injection saldırılara açiktır. Örneğin, kullanıcıdan alınan bir e-posta adresinin doğrudan WHERE koşuluna eklenmesi saldırganların zararlı SQL komutları çalıştırmasına sebep olabilir. Bu yüzden bu yöntem kesinlikle kullanılmamalıdır.

The screenshot shows the Object Explorer on the left with 'LAPTOP-IFON0J6\SQLEXPRESS' selected. In the center, there is one tab: 'SQLQuery1.sql - LA...S.OrnekVT (sa (68))*'. The query window contains the following malicious code:

```

-- Bu sorgu SQL injection'a açıktır
DECLARE @kullaniciAdi NVARCHAR(100) = 'admin'; DROP TABLE Musteriler;--';
EXEC('SELECT * FROM Musteriler WHERE AdSoyad = ''' + @kullaniciAdi + '''');

```

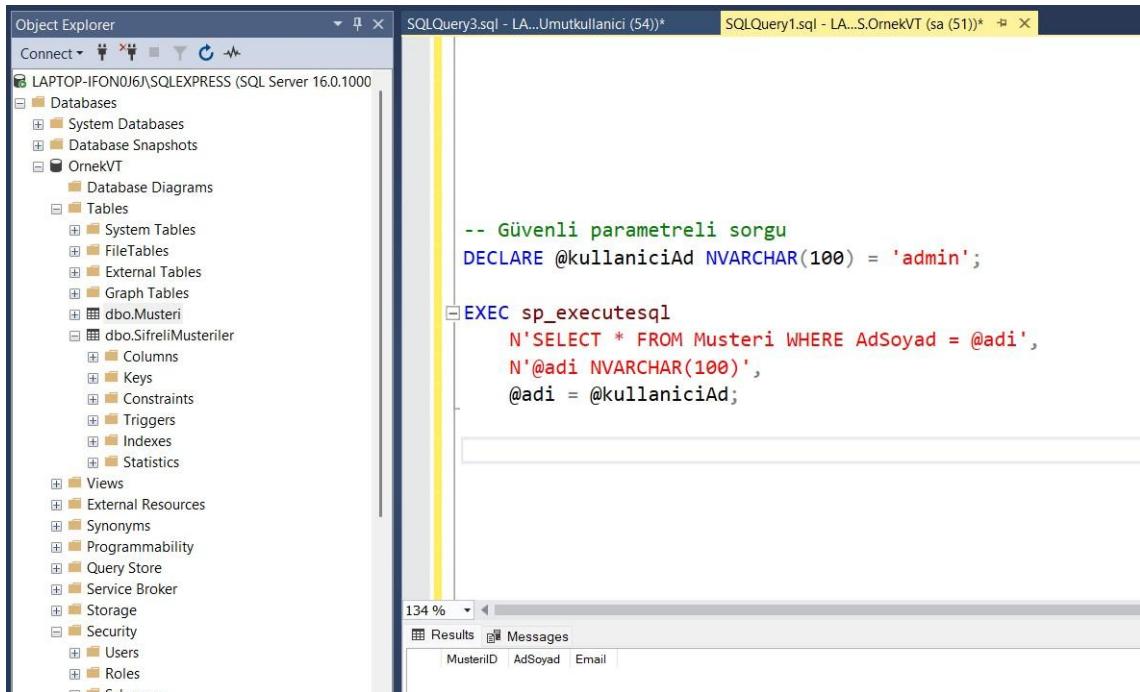
The results pane at the bottom shows the following table data:

MusteriID	AdSoyad	Email
1	admin	admin@sql.com

Şekil 18 Güvensiz Sorgu Örneği ve Analizi

4.8 Parametreli Sorgularla Güvenli Kodlama

Güvenli sorgular, kullanıcı girdisinin doğrudan sorguya gömülmesi yerine, parametreler aracılığıyla sorguya ilettilmesiyle oluşturulur. Bu yöntem, SQL Injection saldırısını engeller çünkü veriler otomatik olarak doğrulanır ve komut olarak değil, veri olarak değerlendirilir. SQL Server'da bu amaçla sp_executesql gibi yöntemler veya uygulama tarafında SqlCommand.Parameters kullanılır.



The screenshot shows the Object Explorer on the left and two query panes on the right. The left pane shows a database named 'OrnekVT' with various objects like Tables, Views, and Security. The right pane has tabs for 'SQLQuery3.sql - L...Umutkullanici (54)*' and 'SQLQuery1.sql - L...S.OrnekVT (sa (51))*. The 'SQLQuery1.sql' tab is active, displaying the following code:

```
-- Güvenli parametreli sorgu
DECLARE @kullaniciAd NVARCHAR(100) = 'admin';

EXEC sp_executesql
    N'SELECT * FROM Musteri WHERE AdSoyad = @adi',
    N'@adi NVARCHAR(100)',
    @adi = @kullaniciAd;
```

Şekil 19 Güvenli Parametreli Sorgu

4.9 KULLANICI AKTİVİTELERİNİ İZLEME

Trigger ile Loglama Yöntemi

Bu trigger, Musteri tablosu üzerinde yapılan **INSERT**, **DELETE** ve **UPDATE** işlemlerini otomatik olarak LogKayitlar tablosuna kaydeder. Her işlem gerçekleştiğinde, işlem türü (@Islem), işlemi yapan kullanıcı (SYSTEM_USER) ve açıklama bilgisi ile loglanır. Bu sayede, veritabanı üzerinde yapılan değişiklikler izlenebilir hale gelir ve kullanıcı aktiviteleri güvenli bir şekilde takip edilebilir. Audit özelliği kullanılmadığında etkili bir alternatif olarak kullanılabilir.

```
-- Eğer daha önce var olan bir trigger varsa, onu sil
DROP TRIGGER IF EXISTS trg_Musteriler_Log;
GO
```

Messages
Commands completed successfully.
Completion time: 2025-04-24T17:54:33.8468069+03:00

Şekil 20 Trigger Silme (Eğer Önceden Var İse)

```
CREATE TRIGGER trg_Musteriler_Log -- Yeni trigger oluşturuluyor
ON Musteri -- Trigger, Musteri tablosu üzerinde çalışacak
AFTER INSERT, DELETE, UPDATE
AS
BEGIN
    -- İşlem türünü tutacak değişken
    DECLARE @Islem NVARCHAR(50);

    -- Eğer hem INSERT hem de DELETE işlemleri yapılmışsa, bu bir UPDATE işlemidir
    IF EXISTS (SELECT * FROM inserted) AND EXISTS (SELECT * FROM deleted)
        SET @Islem = 'UPDATE';
    ELSE IF EXISTS (SELECT * FROM inserted)
        SET @Islem = 'INSERT';
    ELSE IF EXISTS (SELECT * FROM deleted)
        SET @Islem = 'DELETE';

    -- LogKayitlar tablosuna işlem türünü, kullanıcı bilgisini ve açıklamayı ekle
    INSERT INTO LogKayitlar (IslemTuru, KullanicisiSistemi, Aciklama)
    VALUES (
        @Islem, -- İşlem türü (INSERT, UPDATE, DELETE)
        SYSTEM_USER, -- Sistemi kullanan kullanıcı adı
        'Musteriler tablosunda ' + @Islem + ' işlemi yapıldı.' -- İşlemenin açıklaması
    );
END
GO
```

Messages
Commands completed successfully.
Completion time: 2025-04-24T17:56:05.4227644+03:00

Şekil 21 Trigger Oluşturma

```

CREATE TABLE LogKayitlar (
    LogID INT IDENTITY(1,1) PRIMARY KEY,
    IslemTuru NVARCHAR(50),
    Tarih DATETIME DEFAULT GETDATE(),
    KullaniciSistemi NVARCHAR(100),
    Aciklama NVARCHAR(MAX)
);

SELECT * FROM LogKayitlar;

```

Şekil 22 Log Kayıtları Tablosu Oluşturma

Log kayıtları tablosunun çalışmasını test etmek amacıyla, INSERT, UPDATE ve DELETE komutları kullanılmıştır.

```

USE OrnekVT
INSERT INTO Musteri (AdSoyad)
VALUES ('Ahmet Yılmaz');

select * from Musteri

```

MusteriID	AdSoyad	Email
1	Ali Yılmaz	ali.yilmaz@example.com
2	Ayse Demir	ayse.demir@example.com
3	Mehmet Kaya	mehmet.kaya@example.com
4	Cenin Rihavi	tankirihawi45@gmail.com
5	Umut Akyıldız	umutAky22@gmail.com
6	Ahmet Yılmaz	NULL

Şekil 23 Müşteri Tablosuna Veri Ekleme

The screenshot shows the Object Explorer on the left with the database 'OrnekVT' selected. The 'Tables' node is expanded, showing 'dbo.Musteri'. The 'SQLQuery3.sql' tab in the center contains the following SQL code:

```

UPDATE Musteri
SET AdSoyad = 'Mehmet Yilmaz'
WHERE AdSoyad = 'Ahmet Yilmaz';

select * from Musteri

```

The 'Results' tab on the right displays the updated data in the 'Musteri' table:

MusteriID	AdSoyad	Email
1	Ali Yilmaz	ali.yilmaz@example.com
2	Ayse Demir	ayse.demir@example.com
3	Mehmet Kaya	mehmet.kaya@example.com
4	Cenin Rihavi	tarikrihawi45@gmail.com
5	Umut Akybek kzy	umutAky22@gmail.com
6	Mehmet Yilmaz	NULL

A green status bar at the bottom indicates: **Query executed successfully.**

Şekil 24 Müşteri Tablosundaki Veriyi Update Etme

The screenshot shows the Object Explorer on the left with the database 'OrnekVT' selected. The 'Tables' node is expanded, showing 'dbo.Musteri'. The 'SQLQuery3.sql' tab in the center contains the following SQL code:

```

DELETE FROM Musteri
WHERE AdSoyad = 'Mehmet Yilmaz';

SELECT * FROM Musteri

```

The 'Results' tab on the right displays the remaining data in the 'Musteri' table:

MusteriID	AdSoyad	Email
1	Ali Yilmaz	ali.yilmaz@example.com
2	Ayse Demir	ayse.demir@example.com
3	Mehmet Kaya	mehmet.kaya@example.com
4	Cenin Rihavi	tarikrihawi45@gmail.com
5	Umut Akybek kzy	umutAky22@gmail.com

Şekil 25 Müşteri Tablosundan Veri Silme

The screenshot shows the SSMS interface. The Object Explorer on the left lists various database objects under the 'OrnekVT' database, including Database Snapshots, Tables (System Tables, FileTables, External Tables, Graph Tables), Views, External Resources, Synonyms, Programmability, Query Store, Service Broker, Storage, Security (Users, Roles, Schemas, Asymmetric Keys, Certificates, VeriSertifikasi, Symmetric Keys, Always Encrypted Keys, Database Audit Specifications, Security Policies), and Server Objects. The main window displays the results of the query 'select * from LogKayitlar'. The results table has columns: LogID, IslemTuru, Tarih, KullaniciSistemi, and Aciklama. The data is as follows:

LogID	IslemTuru	Tarih	KullaniciSistemi	Aciklama
1	INSERT	2025-04-24 18:08:21.517	sa	Musteriler tablosunda INSERT işlemi yapıldı.
2	UPDATE	2025-04-24 18:09:12.190	sa	Musteriler tablosunda UPDATE işlemi yapıldı.
3	DELETE	2025-04-24 18:09:40.810	sa	Musteriler tablosunda DELETE işlemi yapıldı.
4	INSERT	2025-04-24 18:10:00.100	sa	Musteriler tablosunda INSERT işlemi yapıldı.
5	UPDATE	2025-04-24 18:10:39.240	sa	Musteriler tablosunda UPDATE işlemi yapıldı.
6	DELETE	2025-04-24 18:11:05.113	sa	Musteriler tablosunda DELETE işlemi yapıldı.

Şekil 26 Log Kayıtları Tablosu

Notlar:

- SSMS üzerinden **Security> Logins** sekmesinden kullanıcıları görüp, şifreleri değiştirebilirsin.
- Yetkilerde hata yaparsan `sp_droprolememer` ve `DROP USER, DROP LOGIN`

5 Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları

Bu projenin temel amacı; dağıtık veritabanı sistemlerinde yaygın olarak kullanılan veri çoğaltma (replikasyon), yük dengeleme ve failover (başarısızlık durumunda otomatik geçiş) gibi kritik kavramları, SQL Server ortamında örnek bir senaryo ile simüle etmektir. Gerçek dünyada, yüksek erişilebilirlik ve kesintisiz hizmet gerektiren sistemlerde bu tekniklerin uygulanması büyük önem taşımaktadır. Bu çalışma, söz konusu yapıların temel mantığını kavramaya yönelik basit ve anlaşılır bir yaklaşım sunar.

5.1 Teknik ve Kavramsal Temeller

1. Veritabanı Replikasyonu

Veritabanı replikasyonu, bir veritabanındaki verilerin başka bir veritabanına kopyalanarak çoğaltılması ve bu verilerin güncel hâlde tutulması işlemidir. Böylece veriler yedeklenmiş olur ve sistemler arasında senkronizasyon sağlanır. SQL Server, replikasyon işlemleri için Snapshot, Transactional ve Merge gibi çeşitli modeller sunmaktadır. Bu projede, manuel olarak gerçekleştirilen temel bir Transactional Replikasyon senaryosu örneği uygulanmıştır.

1.2. Yük Dengeleme

Yük dengeleme, kullanıcı isteklerinin birden fazla veritabanı sunucusuna eşit şekilde dağıtılması işlemidir. Bu yöntem, sunucuların aşırı yük altında kalmasını öner ve genel sistem performansını iyileştirir.

Projede, basit bir yük dengeleme senaryosu simüle edilmiştir. Gelen isteklerin hangi sunucuya yönlendirildiği bir log tablosu üzerinden kaydedilmiş ve analiz edilmiştir.

1.3.Failover Senaryoları

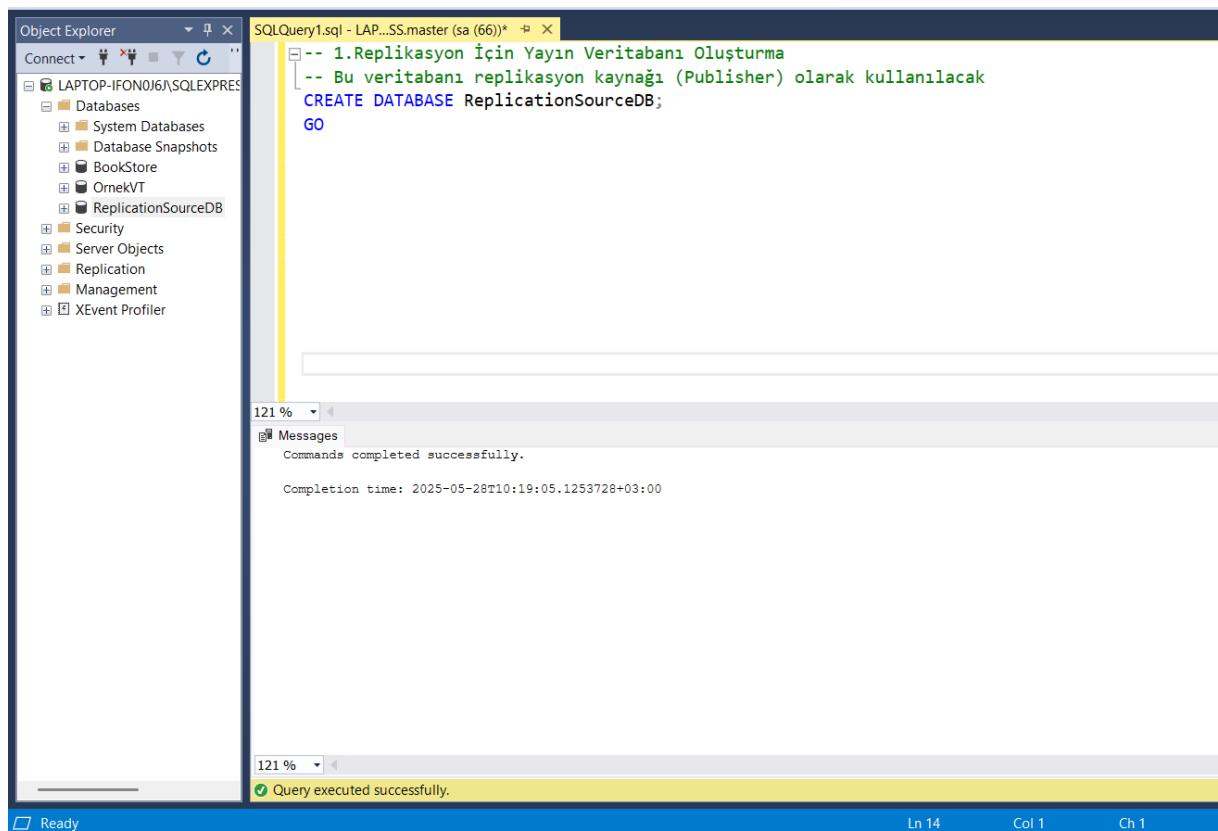
Failover, bir sistem bileşeni çalışmaz hâle geldiğinde hizmetin kesintisiz devam etmesini sağlamak amacıyla otomatik olarak yedek sisteme geçiş yapılmasıdır. Proje kapsamında bu durum, kaynak veritabanı kullanılamaz hâle geldiğinde hedef veritabanı üzerinden veri okuma işleminin gerçekleştirilemesiyle simüle edilmiştir. Bu sayede sistemin sürekliliği sağlanmış olur.

5.2 Uygulama

5.2.1 Veritabanlarının Oluşturulması

İki ayrı veritabanı oluşturulmuştur:

- **ReplicationSourceDB:** Veri girişlerinin yapıldığı kaynak veritabanı



The screenshot shows the Object Explorer on the left and a SQL Query window on the right. The Object Explorer lists databases like BookStore, OrnekVT, and ReplicationSourceDB. The query window contains the following SQL code:

```
-- 1.Replikasyon İçin Yayın Veritabanı Oluşturma
-- Bu veritabanı replikasyon kaynağı (Publisher) olarak kullanılacak
CREATE DATABASE ReplicationSourceDB;
GO
```

The Messages pane at the bottom shows the command completed successfully with the completion time: 2025-05-28T10:19:05.1253728+03:00. A green status bar at the bottom right indicates "Query executed successfully."

- **ReplicationTargetDB:** Yedek verileri tutan hedef veritabanı

The screenshot shows the Object Explorer on the left with a connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. In the center, a query window titled 'SQLQuery1.sql - LAP...SS.master (sa (66))*' contains the following T-SQL code:

```
-- Bu veritabanı replikasyon hedefi (Subscriber) olarak kullanılacak
CREATE DATABASE ReplicationTargetDB;
GO
```

The 'Messages' pane at the bottom displays the execution results:

Commands completed successfully.
Completion time: 2025-05-28T10:21:12.6290605+03:00

A yellow bar at the bottom of the messages pane indicates: **Query executed successfully.**

5.2.3 Tabloların Oluşturulması

Her iki veritabanında da **ReplicatedTable** adında, aynı yapıya sahip bir tablo oluşturulmuştur. Bu tablolar, kaynak ve hedef veritabanları arasında veri replikasyonunun sağlıklı şekilde yapılabilmesi için temel yapı taşıını oluşturmaktadır.

Kaynak veritabanındaki tabloda, veri eklendiğinde **LastUpdated** sütunu otomatik olarak güncel tarih ve saatü üretir. Bu sayede veri girişi sırasında zaman bilgisi elle girilmeden otomatik olarak kayıt altına alınır.

Hedef veritabanındaki tabloda ise **LastUpdated** sütunu manuel olarak doldurulmaktadır. Bu yapı, replikasyon senaryosunun kontrollü şekilde gerçekleştirilmesini sağlar. Kaynak tablo, veri girişlerinin yapıldığı yer olurken; hedef tablo, bu verilerin yansıtıldığı yapıdır.

-Kaynak Tablo:

The screenshot shows the Object Explorer on the left with the connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. The 'ReplicationSourceDB' database is selected. In the center, a query window titled 'SQLQuery1.sql - LA...nSourceDB (sa (66))*' contains the following T-SQL code:

```
-- Kaynak veritabanında "ReplicatedTable" adlı tabloyu oluştur
USE ReplicationSourceDB;
GO
CREATE TABLE ReplicatedTable (
    ID INT PRIMARY KEY,
    DataValue NVARCHAR(100),
    LastUpdated DATETIME DEFAULT GETDATE()
);
GO
```

The status bar at the bottom indicates 'Query executed successfully.'

-- Hedef tablo:

The screenshot shows the Object Explorer on the left with the connection to 'LAPTOP-IFON0J6\SQLEXPRESS'. The 'ReplicationTargetDB' database is selected. In the center, a query window titled 'SQLQuery1.sql - LA...nTargetDB (sa (66))*' contains the following T-SQL code:

```
-- Hedef veritabanında aynı yapıda "ReplicatedTable" oluştur
USE ReplicationTargetDB;
GO
CREATE TABLE ReplicatedTable (
    ID INT PRIMARY KEY,
    DataValue NVARCHAR(100),
    LastUpdated DATETIME
);
GO
```

The status bar at the bottom indicates 'Query executed successfully.'

5.2.4 Örnek Veri Girişi

Bu adımda, sistemin test edilebilmesi ve replikasyon işleminin simül edilebilmesi için kaynak veritabanına örnek veriler girilmiştir.

Girilen veriler, daha sonra hedef veritabanına senkronize edilerek veri aktarımının başarılı bir şekilde gerçekleşip gerçekleşmediği gözlemlenecektir.

Bu sayede, manuel replikasyon senaryosunun işleyişi ve tablolar arası veri eşleşmesinin doğruluğu test edilmiş olacaktır.

```
-- Kaynak tabloya üç satır örnek veri eklenir
USE ReplicationSourceDB;
GO
INSERT INTO ReplicatedTable (ID, DataValue)
VALUES
(1, 'Deneme Veri 1'),
(2, 'Deneme Veri 2'),
(3, 'Deneme Veri 3');
GO
```

(3 rows affected)

Completion time: 2025-05-28T12:12:41.6446989+03:00

5.3 Manuel Replikasyon

Bu bölümde, replikasyon süreci manuel olarak gerçekleştirilmiştir. Kaynak veritabanındaki veriler, hedef veritabanına elle yazılan SQL sorguları ile aktarılmıştır. Veri aktarımı sırasında sadece hedef veritabanında bulunmayan yeni kayıtlar kopyalanmıştır. Bu işlemde, kayıtlar ID sütunu baz alınarak karşılaştırılmış ve yalnızca hedefte olmayan veriler eklenmiştir. Bu sayede temel düzeyde bir transactional replikasyon simülasyonu yapılmıştır.

5.4 Failover Senaryosu

Failover, sistem bileşenlerinden birinin arızalanması durumunda, hizmetin kesintisiz devam etmesi için otomatik ya da manuel olarak alternatif sisteme geçiş yapılmasıdır. Bu projede, basit bir failover senaryosu manuel replikasyon sürecinin parçası olarak ele alınmıştır. Kaynak veritabanının erişilemez olduğu durumlarda, veri okuma işlemleri hedef veritabanından gerçekleştirilmiştir. Böylece, sistemin yüksek erişilebilirlik ve süreklilik sağlanması amaçlanmıştır. Bu senaryo, gerçek dünyadaki donanım veya yazılım hatalarında yedek sistemin devreye girmesine yönelik temel bir örnek olarak simüle edilmiştir.

```
-- Hedef tabloya sadece kaynaka olup hedefte olmayan kayıtlar eklenir
USE ReplicationTargetDB;
GO
INSERT INTO ReplicatedTable (ID, DataValue, LastUpdated)
SELECT src.ID, src.DataValue, src.LastUpdated
FROM ReplicationSourceDB.dbo.ReplicatedTable src
LEFT JOIN ReplicationTargetDB.dbo.ReplicatedTable tgt
    ON src.ID = tgt.ID
WHERE tgt.ID IS NULL;
GO

SELECT * FROM ReplicationTargetDB.dbo.ReplicatedTable;
```

ID	DataValue	LastUpdated
1	Deneme Veri 1	2025-05-28 12:12:41.637
2	Deneme Veri 2	2025-05-28 12:12:41.637
3	Deneme Veri 3	2025-05-28 12:12:41.637

Query executed successfully.

5.5 Yük Dengeleme Simülasyonu

Bu bölümde, sistemde gerçekleşen isteklerin hangi sunucuya yönlendirildiği takip edilmiştir. Yük dengeleme işlemi, istemcilerden gelen veri taleplerinin eşit ve dengeyi şekilde birden fazla sunucuya dağıtılması esasına dayanır. Projede, bu süreç basit bir şekilde iki sunucu (Server1 ve Server2) üzerinde simüle edilmiştir.

İsteklerin hangi sunucuya gittiği bilgisi, LoadBalancerLog adlı özel bir log tablosunda kayıt altına alınmıştır. Bu tablo, her gelen isteğin hedef sunucusunu, isteğin zamanını ve diğer ilgili bilgileri tutarak sistemdeki yük dağılımının izlenmesini sağlamaktadır. Böylece, yük dengeleme mekanizmasının nasıl çalıştığı ve isteklerin nasıl dağıldığı gözlemlenebilir hale gelmiştir.

```
-- Log tablosu oluşturulur; her istek burada kayıt altına alınacaktır
USE ReplicationSourceDB;
GO
CREATE TABLE LoadBalancerLog (
    ID INT IDENTITY(1,1) PRIMARY KEY,
    AccessTime DATETIME DEFAULT GETDATE(),
    ServerName NVARCHAR(50)
);
GO
```

Commands completed successfully.

Completion time: 2025-05-28t12:18:08.2449230+03:00

Daha sonra gelen istekler Server1 ve Server2'ye yönlendirilmiş gibi simüle edilmiştir:

The screenshot shows two parallel sessions in SQL Server Management Studio (SSMS) Object Explorer. Both sessions are connected to the same database, 'nSourceDB', using the 'sa' user.

Session 1 (Top):

- Query: `-- Server1'e gelen istek örneği
INSERT INTO LoadBalancerLog (ServerName) VALUES ('Server1');`
- Result: `(1 row affected)`
- Completion time: `2025-05-28T12:19:13.9381270+03:00`

Session 2 (Bottom):

- Query: `-- Server2'ye gelen istek örneği
INSERT INTO LoadBalancerLog (ServerName) VALUES ('Server2');`
- Result: `(1 row affected)`
- Completion time: `2025-05-28T12:19:59.4861286+03:00`

İstatistiksel analiz yapılmıştır:

The screenshot shows the SSMS interface. On the left, the Object Explorer pane displays the database structure of 'LAPTOP-IFON0J6\SQLEXPRESS'. In the center, a query window titled 'SQLQuery1.sql' contains the following T-SQL code:

```
-- Log tablosu üzerinden yük dağılımı analiz edilir
SELECT ServerName, COUNT(*) AS RequestCount
FROM LoadBalancerLog
GROUP BY ServerName;
GO
```

Below the code, the 'Results' tab shows the output:

ServerName	RequestCount	
1	Server1	1
2	Server2	1

6 Veri Temizleme ve ETL Süreçleri Tasarımı

Projenin amacı; hatalı, eksik ya da tutarsız verilerin tespit edilerek düzeltilmesi, farklı biçimlerde gelen verilerin standart hale getirilmesi ve ardından bu verilerin temiz bir yapıda yeni tablolara aktarılmasıdır. Ayrıca veri kalitesi hakkında raporlamalar yapılarak yapılan işlemlerin izlenebilirliği sağlanmıştır.

6.1 Veri Tabanı Oluşturma

Öncelikle SQL Server'da bir veritabanı oluşturalım. Adını örnek olsun diye ETL_ExampleDB koyuyorum.

The screenshot shows the SSMS interface. On the left, the Object Explorer pane displays the database structure of 'KELEBEK'. In the center, a query window titled 'SQLQuery1.sql' contains the following T-SQL code:

```
-- Veritabanı oluşturma
CREATE DATABASE ETL_ExampleDB;
GO

-- Oluşturduğumuz veritabanına geçiş yapalım
USE ETL_ExampleDB;
GO
```

Below the code, the 'Messages' tab shows the output:

Command(s) completed successfully.

At the bottom, the status bar indicates: KELEBEK (11.0 SP3) | KELEBEK\Kelebek (54) | ETL_ExampleDB | 00:00:00 | 0 rows

Şekil 1 Veri Tabanı Oluşturma

6.2 Tabloları Oluşturma Komutları:

Şimdi yeni oluşturduğumuz ETL_ExampleDB veritabanı içinde örnek tabloyu oluşturup veri ekleyelim.

The screenshot shows the Object Explorer on the left with the connection to KELEBEK (SQL Server 11.0.6020 - KELEBEK). In the center, a query window titled 'SQLQuery1.sql - KE...LEBEK\Kelebek (54)*' contains the following SQL code:

```
-- Örnek table oluşturma
CREATE TABLE dbo.Orders (
    OrderID INT PRIMARY KEY,
    CustomerName NVARCHAR(100),
    OrderDate NVARCHAR(20), -- Bilerek yanlış formatta (string olacak)
    Amount DECIMAL(10,2)
);
GO
```

The 'Messages' pane at the bottom shows the message: 'Command(s) completed successfully.'

Sekil 2 Orders Tablosu

The screenshot shows the Object Explorer on the left with the connection to KELEBEK (SQL Server 11.0.6020 - KELEBEK). In the center, a query window titled 'SQLQuery1.sql - KE...LEBEK\Kelebek (54)*' contains the following SQL code:

```
-- Örnek veri ekleme (bazi hatalarla)
INSERT INTO dbo.Orders (OrderID, CustomerName, OrderDate, Amount) VALUES
(1, 'Ahmet Yılmaz', '2023-01-15', 250.00),
(2, 'Ayşe Demir', '15/02/2023', 180.50), -- Tarih formatı farklı
(3, 'Mehmet Çelik', NULL, 300.00), -- Eksik tarih
(4, NULL, '2023-03-01', 150.00), -- Eksik müşteri adı
(5, 'Fatma Kaya', '2023-03-05', NULL); -- Eksik tutar
GO
```

The 'Messages' pane at the bottom shows the message: '(5 row(s) affected)'.

Sekil 3 Orders Tablosuna veriler ekleme

Ne yapıyoruz?

- ✓ Orders tablosunu oluşturduk.
- ✓ OrderDate kolonu tarih olması gerekirken yanlışlıkla metin (string) olarak tanımlandı.
- ✓ 5 kayıt ekledik; içinde eksik ve hatalı veriler var.

The screenshot shows the Object Explorer on the left with the connection to KELEBEK (SQL Server 11.0.6020 - KELEBEK). In the center, a query window titled 'SQLQuery1.sql - KE...LEBEK\Kelebek (54)*' contains the following SQL code:

```
SELECT * FROM dbo.Orders;
```

The 'Results' pane at the bottom displays the following table:

	OrderID	CustomerName	OrderDate	Amount
1	1	Ahmet Yılmaz	2023-01-15	250.00
2	2	Ayşe Demir	15/02/2023	180.50
3	3	Mehmet Çelik	NULL	300.00
4	4	NULL	2023-03-01	150.00
5	5	Fatma Kaya	2023-03-05	NULL

The 'Messages' pane at the bottom shows the message: 'Query executed successfully.'

Sekil 4 Orders Tablosuna verileri görüntüle

6.3 Veri Temizleme - Eksik ve Hatalı Verilerin Tespiti

Şimdi veri temizlemeye başlayalım. İlk olarak, verideki eksik (NULL) veya tutarsız olan kayıtları tespit edeceğiz.

The screenshot shows the SSMS interface with the Object Explorer on the left and a query window titled 'SQLQuery1.sql'. The query is:

```
-- Eksik veya hatalı verileri bulma
SELECT
    OrderID,
    CustomerName,
    OrderDate,
    Amount,
    CASE
        WHEN CustomerName IS NULL THEN 'Eksik Müşteri Adı'
        WHEN OrderDate IS NULL THEN 'Eksik Tarih'
        WHEN Amount IS NULL THEN 'Eksik Tutar'
        WHEN TRY_CONVERT(DATE, OrderDate, 23) IS NULL THEN 'Geçersiz Tarih Formatı'
        ELSE 'Geçerli Kayıt'
    END AS Durum
FROM dbo.Orders;
```

The results grid shows five rows of data with their corresponding 'Durum' (Status) column values:

OrderID	CustomerName	OrderDate	Amount	Durum
1	Ahmet Yılmaz	2023-01-15	250.00	Geçerli Kayıt
2	Ayşe Demir	15/02/2023	180.50	Geçersiz Tarih Formatı
3	Mehmet Çelik	NULL	300.00	Eksik Tarih
4	NULL	2023-03-01	150.00	Eksik Müşteri Adı
5	Fatma Kaya	2023-03-05	NULL	Eksik Tutar

Message bar at the bottom: 'Query executed successfully.'

Şekil 5 Hatalı verileri bulma

Ne yapıyoruz?

- ✓ CustomerName, OrderDate ve Amount kolonlarında NULL olanları tespit ediyoruz.
- ✓ OrderDate tarih formatı geçerli değilse (TRY_CONVERT fonksiyonu kullanılarak) onu da belirliyoruz.
- ✓ Her satır için bir "Durum" etiketi yazıyoruz.

6.4 Veri Temizleme - Hatalı Tarih Formatını Düzenleme

Şimdi sıra hatalı tarih formatını düzeltmeyektedir.

Elimizdeki tarih 15/02/2023 gibi gün/ay/yıl formatında, ama SQL Server bunu standart YYYY-MM-DD formatı gibi algılamıyor.

Bunu düzeltmek için:

- ✓ Tarihi DD/MM/YYYY formatından YYYY-MM-DD formatına çevireceğiz.
- ✓ SQL Server'da bu tip dönüşüm için biraz string işlemi yapacağız.

The screenshot shows the SSMS interface with the Object Explorer on the left and a query window titled 'SQLQuery1.sql'. The query is:

```
-- Hatalı tarih formatını düzeltmek için yeni bir kolon oluşturup güncelleme yapacağız
-- 1. Yeni kolon ekle (temilenmiş tarih)
ALTER TABLE dbo.Orders
ADD cleanOrderDate DATE;
GO
```

Message bar at the bottom: 'Command(s) completed successfully.'

Şekil 6 yeni kolon ekleme

The screenshot shows the Object Explorer on the left with the database 'KELEBEK' selected. The 'dbo.Orders' table is expanded. The 'Script' tab in the top ribbon is active, displaying a T-SQL script. The script updates the 'OrderDate' column in the 'dbo.Orders' table to a new column 'CleanOrderDate'. It uses TRY_CONVERT to handle NULL values and concatenates substrings to format the date as DD/MM/YYYY. A 'GO' statement is at the end. The 'Messages' pane below shows '(1 row(s) affected)' and a green checkmark indicating success.

```
-- 3. DD/MM/YYYY formatındaki tarihleri dönüştürelim
UPDATE dbo.Orders
SET CleanOrderDate = TRY_CONVERT(DATE,
        SUBSTRING(OrderDate, 7, 4) + '-' + SUBSTRING(OrderDate, 4, 2) + '-' + SUBSTRING(OrderDate, 1, 2), 23)
WHERE TRY_CONVERT(DATE, OrderDate, 23) IS NULL
AND OrderDate IS NOT NULL;
GO
```

Query executed successfully. | KELEBEK (11.0 SP3) | KELEBEK\Kelebek (54) | ETL_ExampleDB | 00:00:00 | 0 rows

Şekil 7 DD/MM/YYYY formatındaki tarihleri dönüştürme

The screenshot shows the Object Explorer on the left with the database 'KELEBEK' selected. The 'dbo.Orders' table is selected. The 'SQL Query1.sql' tab in the top ribbon is active, containing a simple SELECT statement that retrieves OrderID, OrderDate, and CleanOrderDate columns from the 'dbo.Orders' table. The 'Results' pane below shows the execution results. The table has five rows with data: OrderID 1, OrderDate 2023-01-15, CleanOrderDate 2023-01-15; OrderID 2, OrderDate 15/02/2023, CleanOrderDate 2023-02-15; OrderID 3, OrderDate NULL, CleanOrderDate NULL; OrderID 4, OrderDate 2023-03-01, CleanOrderDate 2023-03-01; OrderID 5, OrderDate 2023-03-05, CleanOrderDate 2023-03-05.

```
-- Sonuçları kontrol et
SELECT OrderID, OrderDate, CleanOrderDate FROM dbo.Orders;
```

OrderID	OrderDate	CleanOrderDate
1	2023-01-15	2023-01-15
2	15/02/2023	2023-02-15
3	NULL	NULL
4	2023-03-01	2023-03-01
5	2023-03-05	2023-03-05

Query executed successfully. | KELEBEK (11.0 SP3) | KELEBEK\Kelebek (54) | ETL_ExampleDB | 00:00:00 | 5 rows

Şekil 8 Sonuçları kontrol etme

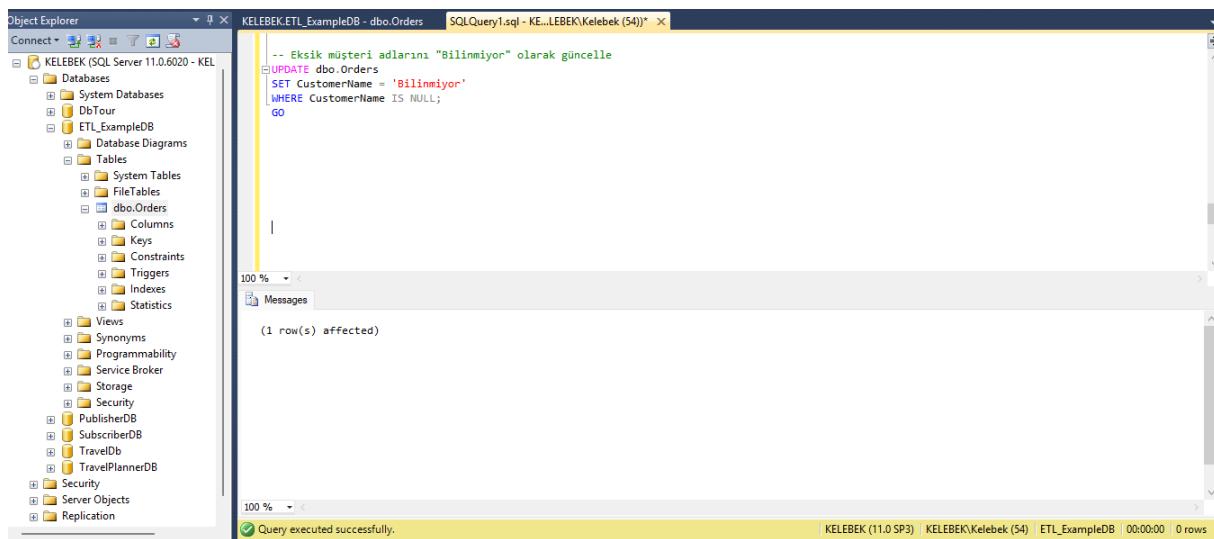
Ne yapıyoruz?

- ✓ CleanOrderDate adında yeni bir tarih (DATE) türünde kolon açıyoruz.
- ✓ İlk olarak OrderDate doğru formatta olanları dönüştürüp CleanOrderDate'a yazıyoruz.
- ✓ Sonra gün/ay/yıl formatında olanları string parçalama ile doğru formata çevirip aynı kolona yazıyoruz.
- ✓ Son olarak yeni tarihi kontrol ediyoruz.

6.5 Eksik Verilerin Düzeltilmesi

Şimdi diğer eksik verileri ele alalım:

- ✓ CustomerName NULL olanları belirli bir değerle (örneğin, 'Bilinmiyor') dolduralım.
- ✓ Amount NULL olanları da 0 (veya başka uygun bir değer) yapalım.

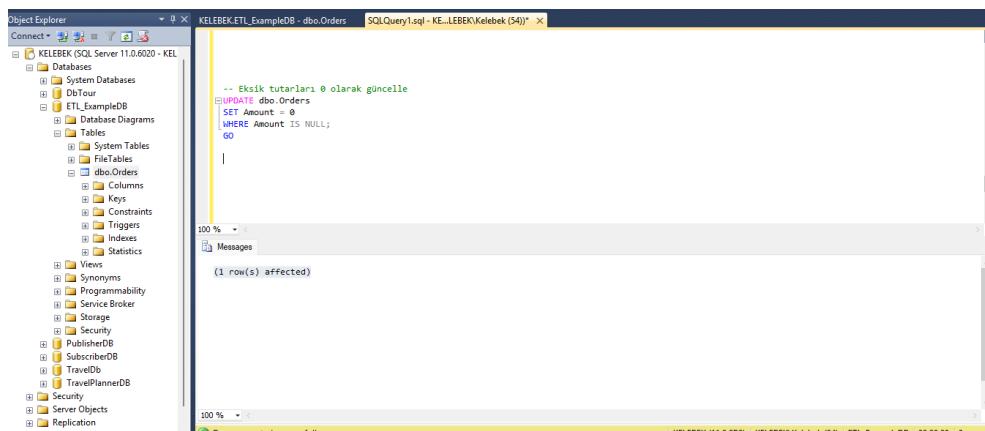


```
-- Eksik müşteri adlarını "Bilinmiyor" olarak güncelle
UPDATE dbo.Orders
SET CustomerName = 'Bilinmiyor'
WHERE CustomerName IS NULL;
GO
```

(1 row(s) affected)

Query executed successfully.

Sekil 9 Eksik müşteri adlarını "Bilinmiyor" olarak güncelleme

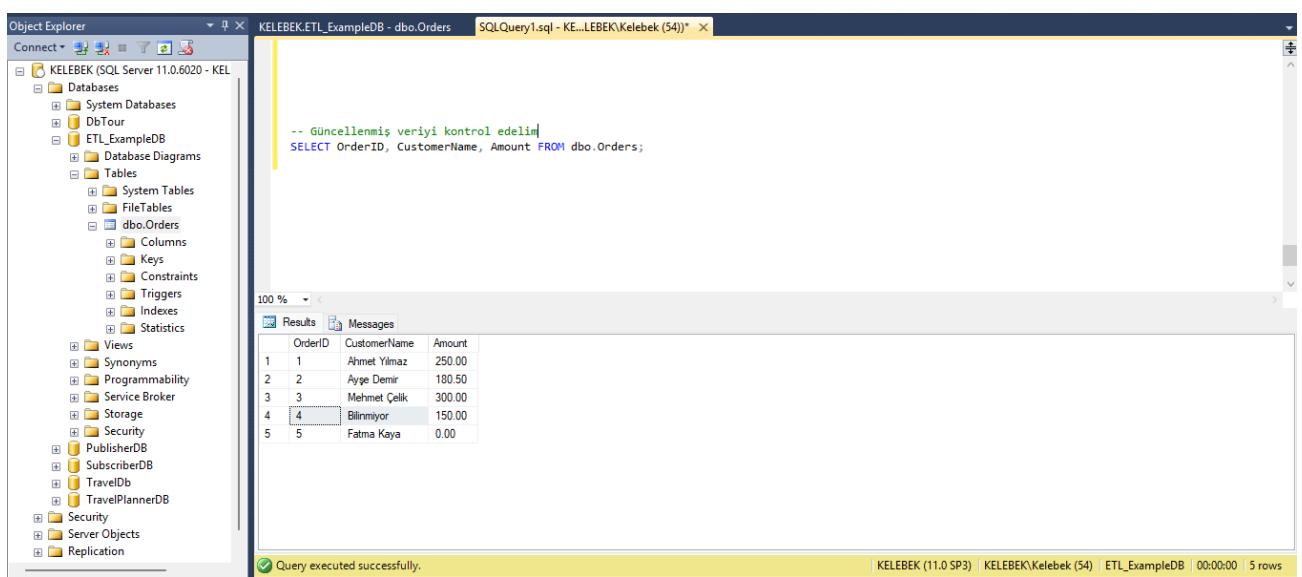


```
-- Eksik tutarları 0 olarak güncelle
UPDATE dbo.Orders
SET Amount = 0
WHERE Amount IS NULL;
GO
```

(1 row(s) affected)

Query executed successfully.

Sekil 10 Eksik tutarları 0 olarak güncelle



```
-- Güncellenmiş veriyi kontrol edelim
SELECT OrderID, CustomerName, Amount FROM dbo.Orders;
```

	OrderID	CustomerName	Amount
1	1	Ahmet Yılmaz	250.00
2	2	Ayşe Demir	180.50
3	3	Mehmet Çelik	300.00
4	4	Bilinmiyor	150.00
5	5	Fatma Kaya	0.00

Query executed successfully.

Sekil 11 Verileri Kontrol Etme

Ne yapıyoruz?

- ✓ CustomerName boş olanları 'Bilinmiyor' olarak değiştirdik.
- ✓ Amount boş olanları 0 yaptık.

6.6 Temizlenmiş Veriyi Yüklemek

Şimdi elimizde temizlenmiş ve dönüştürülmüş veriler var.

Temizlenmiş verileri dbo.CleanOrders adında yeni bir tabloya yükleyelim.

The screenshot shows the Object Explorer on the left and a SQL Query window on the right. The query window contains the following SQL code:

```
-- Temizlenmiş veri için yeni tablo oluştur
CREATE TABLE dbo.CleanOrders (
    OrderID INT PRIMARY KEY,
    CustomerName NVARCHAR(100),
    OrderDate DATE,
    Amount DECIMAL(10, 2)
);
GO
```

The status bar at the bottom of the window indicates "Query executed successfully." and "0 rows".

Şekil 11 Temizlenmiş veri için yeni tablo oluşturma

The screenshot shows the Object Explorer on the left and a SQL Query window on the right. The query window contains the following SQL code:

```
-- Temizlenmiş veriyi yeni tabloya aktar
INSERT INTO dbo.CleanOrders (OrderID, CustomerName, OrderDate, Amount)
SELECT OrderID, CustomerName, CleanOrderDate, Amount
FROM dbo.Orders;
GO
```

The status bar at the bottom of the window indicates "(5 row(s) affected)" and "Query executed successfully." and "0 rows".

Şekil 12 Temizlenmiş veriyi yeni tabloya aktar

```
-- Yeni tabloyu kontrol et
SELECT * FROM dbo.CleanOrders;
```

OrderID	CustomerName	OrderDate	Amount
1	Ahmet Yılmaz	2023-01-15	250.00
2	Ayşe Demir	2023-02-15	180.50
3	Mehmet Çelik	NULL	300.00
4	Bilinmiyor	2023-03-01	150.00
5	Fatma Kaya	2023-03-05	0.00

Query executed successfully.

Şekil 13 Yeni tabloyu kontrol etme

Ne yapıyoruz?

- ✓ Temizlenmiş veriyi tutacak yeni tablo oluşturduk.
- ✓ CleanOrderDate ve güncellenmiş müşteri adı ve tutarlarla yeni tabloyu doldurduk.

6.7 Veri Kalitesi Raporu Oluşturma

Şimdi veri temizleme ve dönüşüm sürecinde neler yapıldığına dair basit bir rapor hazırlayalım.

Raporumuzda:

- ✓ Kaç tane tarih eksik veya hatalıydı?
- ✓ Kaç müşteri adı eksikti?
- ✓ Kaç tutar eksikti? gibi bilgileri göreceğiz.

```
SELECT
    COUNT(*) AS ToplamKayıt,
    SUM(CASE WHEN OrderDate IS NULL THEN 1 ELSE 0 END) AS EksikTarihSayısı,
    SUM(CASE WHEN CustomerName = 'Bilinmiyor' THEN 1 ELSE 0 END) AS EksikMusteriSayısı,
    SUM(CASE WHEN Amount = 0 THEN 1 ELSE 0 END) AS EksikTutarSayısı
FROM dbo.CleanOrders;
```

ToplamKayıt	EksikTarihSayısı	EksikMusteriSayısı	EksikTutarSayısı
5	1	1	1

Query executed successfully.

Şekil 14 Rapor Oluşturma

Yani:

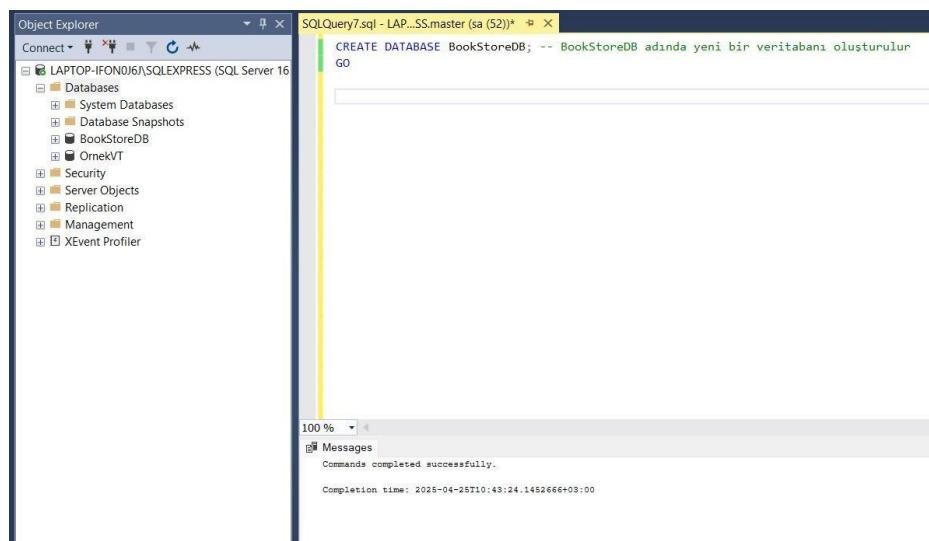
- ✓ Toplam 5 kayıt var,
- ✓ 1 kayıt tarih eksik (OrderID 3),
- ✓ 1 kayıt müşteri adı eksik ve "Bilinmiyor" olarak doldurulmuş,
- ✓ 1 kayıt tutar eksik ve 0 olarak düzeltilmiş.

Bu rapor sayesinde veri kalitesini takip edebiliyoruz.

7 Veri Tabanı Yükseltme ve Sürüm Yönetimi

Veri tabanı sistemlerinin daha yeni sürümlere yükseltilmesi sürecini, bu süreçte izlenecek stratejileri, risklere karşı alınacak önlemleri ve yükselme sonrası test ile geri dönüş planlarını ele almaktadır. Sürüm kontrolü ve yapı değişikliklerinin izlenmesiyle sistem sürekliliği ve veri bütünlüğünün korunması amaçlanmaktadır.

7.1 Veri Tabanı Oluşturma Komutu



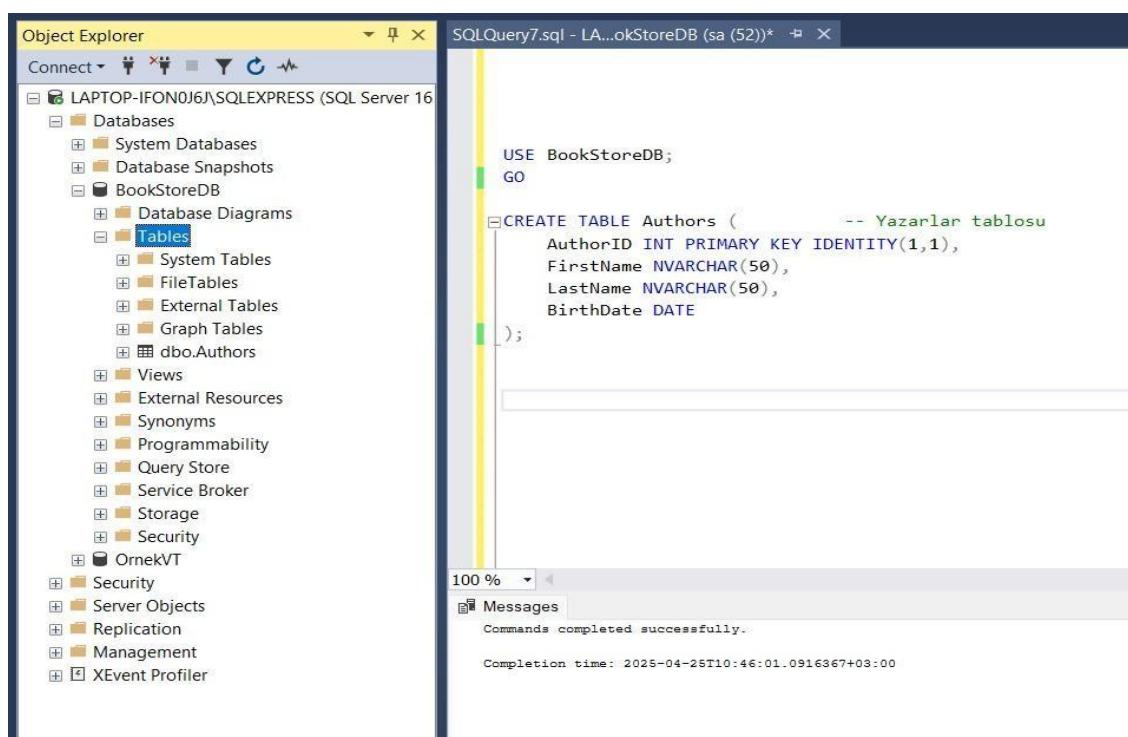
The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer displays the connection to 'LAPTOP-IFON0J6\SQLEXPRESS (SQL Server 16)' and its databases, including 'System Databases', 'Database Snapshots', 'BookStoreDB', and 'OrnekVt'. The SQL Query window contains the command:

```
CREATE DATABASE BookStoreDB; -- BookStoreDB adında yeni bir veritabanı oluşturulur
GO
```

The status bar at the bottom indicates 'Commands completed successfully.' and 'Completion time: 2025-04-25T10:48:24.1452666+03:00'.

Şekil 1 Veri Tabanı Oluşturma

7.2 Tabloları Oluşturma Komutları:



The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer displays the connection to 'LAPTOP-IFON0J6\SQLEXPRESS (SQL Server 16)' and its databases, including 'System Databases', 'Database Snapshots', 'BookStoreDB', and 'OrnekVt'. The SQL Query window contains the command:

```
USE BookStoreDB;
GO

CREATE TABLE Authors ( -- Yazarlar tablosu
    AuthorID INT PRIMARY KEY IDENTITY(1,1),
    FirstName NVARCHAR(50),
    LastName NVARCHAR(50),
    BirthDate DATE
);
```

The status bar at the bottom indicates 'Commands completed successfully.' and 'Completion time: 2025-04-25T10:46:01.0916367+03:00'.

Şekil 2 Yazarlar Tablosu

The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the 'BookStoreDB' database under 'LAPTOP-IFON0J6\SQLEXPRESS (SQL Server 16)'. The SQL Query window contains the following SQL code:

```
USE BookStoreDB;
GO

CREATE TABLE Books (
    BookID INT PRIMARY KEY IDENTITY(1,1),
    Title NVARCHAR(100),
    Genre NVARCHAR(50),
    Price DECIMAL(10,2),
    AuthorID INT FOREIGN KEY REFERENCES Authors(AuthorID)
);
```

The 'Messages' pane at the bottom shows the command completed successfully.

Şekil 3 Kitaplar Tablosu

The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the 'BookStoreDB' database under 'LAPTOP-IFON0J6\SQLEXPRESS (SQL Server 16)'. The SQL Query window contains the following SQL code:

```
USE BookStoreDB;
GO

CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY IDENTITY(1,1),
    FullName NVARCHAR(100),
    Email NVARCHAR(100),
    RegisteredDate DATE
);
```

The 'Messages' pane at the bottom shows the command completed successfully.

Şekil 4 Müşteriler Tablosu

The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the BookStoreDB database with its tables (Authors, Books, Customers, Orders). The SQL Query window contains the following code:

```
USE BookStoreDB;
GO

CREATE TABLE Orders (
    -- Siparişler tablosu
    OrderID INT PRIMARY KEY IDENTITY(1,1),
    CustomerID INT FOREIGN KEY REFERENCES Customers(CustomerID),
    OrderDate DATETIME,
    TotalAmount DECIMAL(10,2)
);
```

The Messages pane at the bottom shows "Commands completed successfully." and the completion time.

Şekil 5 Siparişler Tablosu

The screenshot shows the Object Explorer on the left and the SQL Query window on the right. The Object Explorer lists the BookStoreDB database with its tables (Authors, Books, Customers, OrderDetails). The SQL Query window contains the following code:

```
USE BookStoreDB;
GO

CREATE TABLE OrderDetails (
    -- Sipariş detayları tablosu
    OrderDetailID INT PRIMARY KEY IDENTITY(1,1),
    OrderID INT FOREIGN KEY REFERENCES Orders(OrderID),
    BookID INT FOREIGN KEY REFERENCES Books(BookID),
    Quantity INT,
    UnitPrice DECIMAL(10,2)
);
```

The Messages pane at the bottom shows "Commands completed successfully." and the completion time.

Şekil 5 Sipariş Detay Tablosu

7.3 Tablolara Veri Ekleme

The screenshot shows the SSMS interface. The Object Explorer on the left lists the database structure for 'LAPTOP-IFON0J6\SQLEXPRESS (SQL Server)'. Under the 'BookStore' database, the 'Tables' node is expanded, showing 'dbo.Authors', 'dbo.Books', 'dbo.Customers', 'dbo.OrderDetails', and 'dbo.Orders'. The 'Messages' pane at the bottom right shows '(3 rows affected)' and a completion time of '2025-04-25T12:42:29.0864663+03:00'.

```
INSERT INTO Authors (FirstName, LastName, BirthDate)
VALUES
('George', 'Orwell', '1903-06-25'),
('J.K.', 'Rowling', '1965-07-31'),
('Jane', 'Austen', '1775-12-16');
```

Sekil 6 Yazarlar Tablosuna Veri Ekleme

The screenshot shows the SSMS interface. The Object Explorer on the left lists the database structure for 'LAPTOP-IFON0J6\SQLEXPRESS (SQL Server)'. Under the 'BookStoreDB' database, the 'Tables' node is expanded, showing 'dbo.Authors', 'dbo.Books', 'dbo.Customers', 'dbo.OrderDetails', and 'dbo.Orders'. The 'Messages' pane at the bottom right shows '(3 rows affected)' and a completion time of '2026-04-26T10:57:30.0912480+03:00'.

```
INSERT INTO Books (Title, Genre, Price, AuthorID)
VALUES
('1984', 'Dystopian', 45.50, 1),
('Harry Potter and the Sorcerer''s Stone', 'Fantasy', 60.00, 2),
('Pride and Prejudice', 'Classic', 39.90, 3);
```

Sekil 7 Kitaplar Tablosuna veri ekleme

The screenshot shows the Object Explorer on the left and the SQL Query Editor on the right. The Object Explorer lists the BookStoreDB database and its objects. The SQL Query Editor contains the following SQL code:

```
INSERT INTO Customers (FullName, Email, RegisteredDate)
VALUES
('Ayşe Yılmaz', 'ayse@example.com', '2023-02-10'),
('Mehmet Demir', 'mehmet@example.com', '2024-05-15');

select * from Customers
```

The Results tab shows the output of the query:

	CustomerID	FullName	Email	RegisteredDate
1	1	Ayşe Yılmaz	ayse@example.com	2023-02-10
2	2	Mehmet Demir	mehmet@example.com	2024-05-15

Şekil 8 Müşteriler Tablosuna veri ekleme

The screenshot shows the Object Explorer on the left and the SQL Query Editor on the right. The Object Explorer lists the BookStoreDB database and its objects. The SQL Query Editor contains the following SQL code:

```
INSERT INTO OrderDetails (OrderID, BookID, Quantity, UnitPrice)
VALUES
(1, 1, 1, 45.50),
(1, 3, 1, 39.90),
(2, 2, 1, 60.00);
```

The Messages tab shows the execution results:

(3 rows affected)
Completion time: 2025-04-25T11:00:05.3159523+03:00

The Status bar at the bottom indicates: Query executed successfully.

Şekil 9 Sipariş Detay Tablosuna veri ekleme

```

Object Explorer
SQLQuery7.sql - LA...okStoreDB (sa (52))*
100 %
Messages
(2 rows affected)
Completion time: 2025-04-25T10:59:22.7802867+03:00

```

```

INSERT INTO Orders (CustomerID, OrderDate, TotalAmount)
VALUES
(1, '2024-04-01', 85.40),
(2, '2024-04-10', 60.00);

```

Şekil 10 Siparişler Tablosuna veri ekleme

7.4 Okuma Yetkisine Sahip Login Oluşturulması

Bu SQL komutları dizisi, bir veritabanında yalnızca okuma yetkisi bulunan bir kullanıcı oluşturmak için kullanılır. Bu işlem genellikle sadece raporlama veya veri görüntüleme amacıyla kullanılan kullanıcılar için uygulanır.

```

Object Explorer
SQLQuery7.sql - LA...okStoreDB (sa (52))*
100 %
Messages
Commands completed successfully.
Completion time: 2025-04-25T10:54:03.2812167+03:00

```

```

CREATE LOGIN readbooklogin WITH PASSWORD = 'StrongPassword123!';
CREATE USER readbookuser FOR LOGIN readbooklogin;
EXEC sp_addrolemember 'db_datareader', 'readbookuser';

```

Şekil 11 Login Oluşturma

7.5 Temel Sorgular

- ✓ OrderDate >= '2024-01-01': 1 Ocak 2024 ve sonrasında siparişleri dahil eder.
- ✓ OrderDate <'2025-01-01': 1 Ocak 2025 tarihinden önceki siparişleri dahil eder.
- ✓ Bu iki koşul bir araya geldiğinde, sorgu **2024 yılına ait (1 Ocak- 31 Aralık) tüm siparişleri** getirir.
- ✓ Bu yöntem, saat bilgisi içeren datetime sütunlarında da güvenilir sonuç verir; çünkü '2025-01-01' tarihinden küçük olan tüm kayıtlar alınır, yani 2024 yılının son saniyesine kadar olanlar dahil edilir.

```
SELECT *
FROM Orders
WHERE OrderDate >= '2024-01-01' AND OrderDate < '2025-01-01'; -- 2024 yılı içindeki siparişleri getirir
```

OrderID	CustomerID	OrderDate	TotalAmount
1	1	2024-04-01 00:00:00.000	85.40
2	2	2024-04-10 00:00:00.000	60.00

Şekil 12 2024 Yılın İçindeki Siparşler

```
-- Müşteri, Kitap ve sipariş bilgilerini
-- birlestiren detaylı sorgu

SELECT
    c.FullName,
    b.Title,
    o.OrderDate,
    od.Quantity,
    od.UnitPrice
FROM OrderDetails od
JOIN Orders o ON od.OrderID = o.OrderID
JOIN Customers c ON o.CustomerID = c.CustomerID
JOIN Books b ON od.BookID = b.BookID;
```

FullName	Title	OrderDate	Quantity	UnitPrice
Ayse Yılmaz	1984	2024-04-01 00:00:00.000	1	45.50
Ayse Yılmaz	Pride and Prejudice	2024-04-01 00:00:00.000	1	39.90
Mehmet Demir	Harry Potter and the Sorcerer's Stone	2024-04-10 00:00:00.000	1	60.00

Şekil 13 Müşteri, Kitap ve Sipariş Bilgilerini Birleştirilen Sorgu

7.6 Performans ve İndeks Analizleri

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer on the left, the BookStoreDB database is selected, displaying its schema. The central pane shows a query window titled 'SQLQuery7.sql - LA...okStoreDB (sa (52))'. The query retrieves the top 5 most CPU-intensive queries from the sys.dm_exec_query_stats system view, including their execution count and text. The results grid below shows one row with an average CPU time of 5465 ms.

```

SELECT TOP 5
    qs.total_worker_time / qs.execution_count AS Avg_CPU_Time,
    qs.execution_count,
    SUBSTRING(qt.text, qs.execution_start_offset / 2 + 1,
              (CASE WHEN qs.execution_end_offset = -1
                     THEN LEN(qt.text) * 2
                     ELSE qs.execution_end_offset END - qs.execution_start_offset) / 2
            ) AS QueryText
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) qt
ORDER BY Avg_CPU_Time DESC;

```

	Avg_CPU_Time	execution_count	QueryText
1	5465	6	SELECT TOP 5

Şekil 14 En Çok CPU Kullanan İlk 5 Sorgu Listeleme

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer on the left, the BookStoreDB database is selected. The central pane shows a query window titled 'SQLQuery7.sql - LA...okStoreDB (sa (52))'. The query analyzes index usage statistics for tables in the database. The results grid below lists five tables and their corresponding indexes, along with usage metrics like user seeks, scans, and lookups.

```

SELECT
    OBJECT_NAME(i.object_id) AS TableName,
    i.name AS IndexName,
    i.index_id,
    dm.user_seeks,
    dm.user_scans,
    dm.user_lookups,
    dm.user_updates
FROM sys.indexes i
INNER JOIN sys.dm_db_index_usage_stats dm
    ON i.object_id = dm.object_id AND i.index_id = dm.index_id
WHERE OBJECTPROPERTY(i.object_id, 'IsUserTable') = 1; -- Sadece kullanıcı tabloları

```

TableName	IndexName	index_id	user_seeks	user_scans	user_lookups	user_updates
1 Authors	PK__Authors__70DAFC147330EE9A	1	1	0	0	1
2 Books	PK__Books__3DE0C22755451425	1	2	0	0	1
3 Customers	PK__Customer__A4AE64B866BB85AE	1	2	1	0	1
4 Orders	PK__Orders__C3905BAF6A1C8350	1	2	2	0	1
5 OrderDetails	PK__OrderDet__D3B9D30CCA4CA2F0	1	0	2	0	1

Şekil 15 İndex Kullanım Durumunu Analiz Etme

7.7 Şema Değişikliklerini İzleme

The screenshot shows the Object Explorer on the left and a SQL Query window on the right.

Object Explorer:

- LAPTOP-IFON0J6\SQLEXPRESS (SQL Server)
 - Databases
 - System Databases
 - Database Snapshots
 - BookStoreDB
 - Database Diagrams
 - Tables
 - System Tables
 - FileTables
 - External Tables
 - Graph Tables
 - dbo.Authors
 - dbo.Books
 - dbo.Customers
 - dbo.OrderDetails
 - dbo.Orders
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Query Store
 - Service Broker
 - Storage
 - Security
 - OrnekVTK
 - Security
 - Server Objects
 - Replication
 - Management
 - XEvent Profiler
 - LAPTOP-IFON0J6\SQLEXPRESS (SQL Server)
 - Databases
 - Security
 - Server Objects

SQL Query Window:

```

CREATE TABLE SchemaChangeLog ( -- Şema değişikliklerini kaydedecek tablo oluşturuluyor
    LogID INT PRIMARY KEY IDENTITY(1,1),
    EventData XML, -- Değişiklik olay verisi
    ChangeDate DATETIME DEFAULT GETDATE()
);
GO

```

Results Grid:

LogID	EventData	ChangeDate
1	<EVENT_INSTANCE><EventType>CREATE_TABLE</EventType>	2025-04-25 11:07:21.480

Current Database:

CurrentDatabase
BookStoreDB

Extended Properties Grid:

LogID	ChangeDate	EventType	ObjectName	ExecutedCommand
1	2025-04-25 11:07:21.480	CREATE_TABLE	SchemaChangeLog	CREATE TABLE SchemaChangeLog (-- Şema de...)

Şekil 16 Şema Değişikliklerini Kaydedecek Tablo Oluşturma

The screenshot shows the Object Explorer on the left and a SQL Query window on the right.

Object Explorer:

- LAPTOP-IFON0J6\SQLEXPRESS (SQL Server)
 - System Databases
 - Database Snapshots
 - BookStoreDB
 - Database Diagrams
 - Tables
 - Views
 - External Resources
 - Synonyms
 - Programmability
 - Stored Procedures
 - Functions
 - Database Triggers
 - Assemblies
 - Types
 - Rules
 - Defaults
 - Sequences
 - Query Store
 - Service Broker
 - Storage
 - Security
 - OrnekVTK
 - Security
 - Server Objects
 - Replication
 - Management
 - XEvent Profiler
 - LAPTOP-IFON0J6\SQLEXPRESS (SQL Server)
 - Databases
 - Security
 - Server Objects

SQL Query Window:

```

-- DDL trigger -> Veritabanı düzeyinde yapılan tüm şema değişikliklerini loglar
CREATE TRIGGER trgSchemaChange
ON DATABASE
FOR DDL_DATABASE_LEVEL_EVENTS
AS
BEGIN
    INSERT INTO SchemaChangeLog (EventData)
    VALUES (EVENTDATA()); -- Olay verisini tabloya ekler
END;

SELECT * FROM sys.triggers WHERE name = 'trgSchemaChange';

```

Results Grid:

name	object_id	parent_class	parent_class_desc	parent_id	type	type_desc	create_date	modify_date	is_ms_shipped	is_disabled	is_not_for_replication	is_instead_of_trigger
trgSchemaChange												

Şekil 17 DDL Trigger

7.9 Örnek Şema Değişikliği

The screenshot shows the Object Explorer on the left and a query window on the right. The query window contains the following T-SQL code:

```
-- Books tablosuna yeni bir sütun ekleniyor
ALTER TABLE Books ADD Publisher1 NVARCHAR(100);

EXEC sp_help 'Books';
```

The results pane shows the structure of the Books table, including the newly added Publisher1 column.

Name	Owner	Type	Created_datetime							
Books	dbo	user table	2025-04-25 10:47:17.747							
Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation	
BookID	int	no	4	10	0	yes	(n/a)	(n/a)	NULL	
Title	nvarchar	no	200			yes	(n/a)	(n/a)	SQL_Latin1_General_CI_AS	
Genre	nvarchar	no	100			yes	(n/a)	(n/a)	SQL_Latin1_General_CI_AS	
Price	decimal	no	9	10	2	yes	(n/a)	(n/a)	NULL	
AuthorID	int	no	4	10	0	yes	(n/a)	(n/a)	NULL	
Publisher	nvarchar	no	200			yes	(n/a)	(n/a)	SQL_Latin1_General_CI_AS	
Identity	Seed	Increment	Not For Replication							
	BookID	1	0							
RowGuidCol										
No rowguidcol column defined.										
Data_located_on_filegroup										
PRIMARY										
index_name	index_description	index_keys								
PK_Books_3DE0C22755451425	clustered, unique primary key located on PRIMARY	BookID								
constraint_type	constraint_name	delete_action	update_action	status_enabled	status_for_replication	constraint_keys				
FOREIGN KEY	FK__Books__AuthorID__4BAC3F29	No Action	No Action	Enabled	Is_For_Replication	AuthorID				
PRIMARY KEY	PK_Books_3DE0C22755451425	(n/a)	(n/a)	(n/a)	(n/a)	REFERENCES BookStoreDB.dbo.Authors (AuthorID)				
Table is referenced by foreign key										
BookStoreDB.dbo.OrderDetails FK__OrderData__Boo...										

Şekil 18 Kitap Tablosuna Yeni Sütun Ekleme

Log ve Versiyon Kontrolleri

The screenshot shows the Object Explorer on the left and a query window on the right. The query window contains the following T-SQL code:

```
-- Şema değişiklik kayıtlarını listeleme
SELECT * FROM dbo.SchemaChangeLog; -- Ham kayıtları getirir
```

The results pane shows the SchemaChangeLog table, which contains a single entry for the creation of the Books table.

LogID	EventData	ChangeDate
1	<EVENT_INSTANCE><EventType>CREATE_TABLE</EventTy...	2025-04-25 11:07:21.480

Şekil 19 Şema Değişiklik Kayıtlarını Listeleme

The screenshot shows the SSMS interface. On the left, the Object Explorer tree view is expanded to show the 'BookStoreDB' database, including its tables, views, and other objects. In the center, a query window titled 'SQLQuery7.sql - LA...okStoreDB (sa (52))' contains the following SQL code:

```
-- Aktif veritabanı adı
SELECT DB_NAME() AS CurrentDatabase;
```

The results pane at the bottom shows a single row of data:

CurrentDatabase
BookStoreDB

Şekil 20 Aktif veri Tabanı

The screenshot shows the SSMS interface. On the left, the Object Explorer tree view is expanded to show the 'BookStoreDB' database, including its tables, views, and other objects. In the center, a query window titled 'SQLQuery7.sql - LA...okStoreDB (sa (52))' contains the following SQL code:

```
-- Şema değişiklik loglarını detaylı gösterir
SELECT
    LogID,
    ChangeDate,
    EventData.value('/EVENT_INSTANCE/EventType[1]', 'NVARCHAR(100)') AS EventType,
    EventData.value('/EVENT_INSTANCE/ObjectName[1]', 'NVARCHAR(100)') AS ObjectName,
    EventData.value('/EVENT_INSTANCE/TSQLCommand/CommandText[1]', 'NVARCHAR(MAX)') AS ExecutedCommand
FROM dbo.SchemaChangeLog
ORDER BY ChangeDate DESC; -- En yeni değişiklik en üstte
```

The results pane at the bottom shows a single row of data:

LogID	ChangeDate	EventType	ObjectName	ExecutedCommand
1	2025-04-25 11:07:21.480	CREATE_TABLE	SchemaChangeLog	CREATE TABLE SchemaChangeLog (-- Şema de...)

Şekil 21 Şema Değişiklik Logları

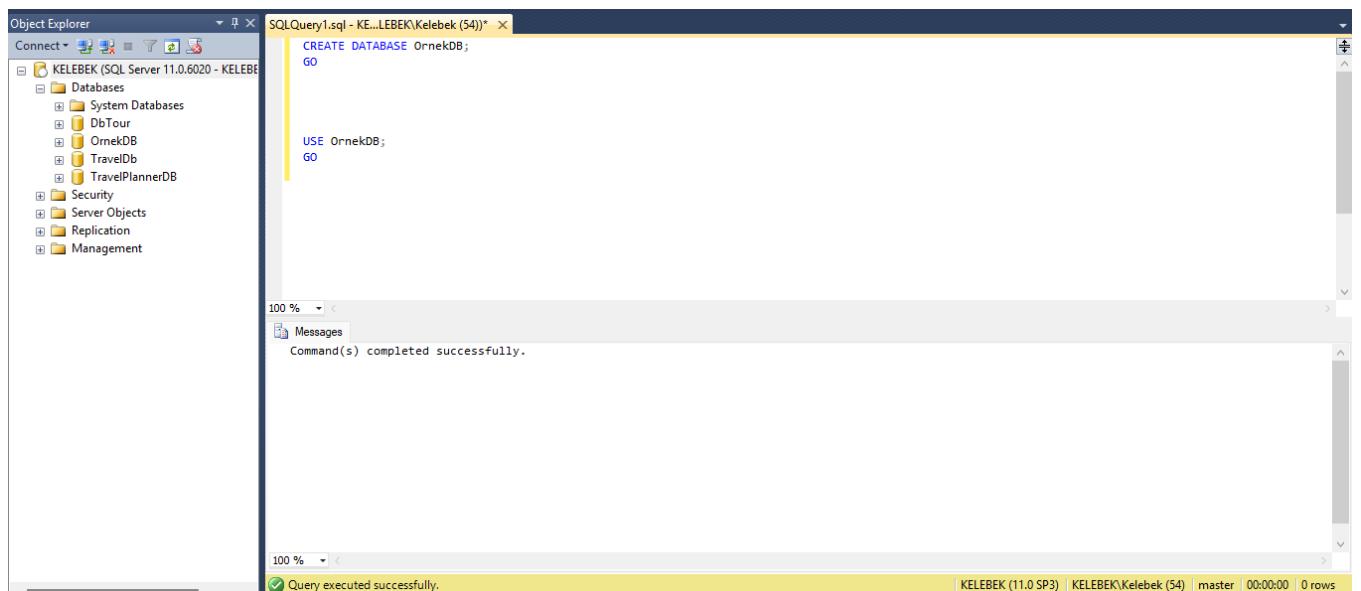
8 Veritabanı Yedekleme ve Otomasyon Çalışması

Bu proje, SQL Server üzerinde bulunan bir veritabanının **düzenli olarak yedeklenmesini otomatikleştirmek, yedekleme sonuçlarını loglamak ve hatalı durumlarda kullanıcıyı bilgilendirmek** amacıyla hazırlanmıştır. Uygulama, eğitim amaçlıdır ve temel seviye SQL, batch ve PowerShell komutlarıyla gerçekleştirilmiştir.

8.1 Veri Tabanı Oluşturma

İlk olarak, SQL Server Management Studio (SSMS) üzerinden veya SQL komutlarıyla OrnekDB adında bir veritabanı oluşturuldu. Bu veritabanı içine, örnek kullanıcı verilerini tutmak için bir tablo eklendi.

- Oluşturulan tablo: Kullanıcılar
- Örnek veriler: İsim, soyisim, e-posta bilgileri



The screenshot shows the SSMS interface with the Object Explorer on the left and a query window on the right. The query window contains the following SQL code:

```
CREATE DATABASE OrnekDB;
GO

USE OrnekDB;
GO
```

The status bar at the bottom indicates "Query executed successfully."

Şekil 1 Veri Tabanı Oluşturma

8.2 Tabloları Oluşturma Komutları:



The screenshot shows the SSMS interface with the Object Explorer on the left and a query window on the right. The query window contains the following SQL code:

```
CREATE TABLE Kullanıcılar (
    KullanıcıID INT PRIMARY KEY IDENTITY(1,1),
    Ad VARCHAR(50),
    Soyad NVARCHAR(50),
    Email NVARCHAR(100)
);
GO
```

The status bar at the bottom indicates "Query executed successfully."

Şekil 2 Kullanıcılar Tablosu

```

Object Explorer
KELEBEK (SQL Server 11.0.6020 - KELEBEK)
Connect Databases
KELEBEK
System Databases
DbTour
OrnekDB
Database Diagrams
Tables
System Tables
FileTables
dbo.Kullaniciilar
Views
Synonyms
Programmability
Service Broker
Storage
Security
TravellDb
TravelPlannerDB
Security
Server Objects
Replication
Management

KELEBEK.OrnekDB - dbo.Kullaniciilar SQLQuery1.sql - KE...LEBEK\Kelebek (54)*
INSERT INTO Kullaniciilar (Ad, Soyad, Email)
VALUES
('Ahmet', 'Yilmaz', 'ahmet.yilmaz@example.com'),
('Ayse', 'Demir', 'ayse.demir@example.com'),
('Mehmet', 'Kaya', 'mehmet.kaya@example.com');
GO

100 %
Messages
(3 row(s) affected)

100 %
Query executed successfully.

```

Şekil 3 Kullanıcılar Tablosuna veriler ekleme

KullaniciID	Ad	Soyad	Email
1	Ahmet	Yilmaz	ahmet.yilmaz@...
2	Ayse	Demir	ayse.demir@...
3	Mehmet	Kaya	mehmet.kaya@...
NULL	NULL	NULL	NULL

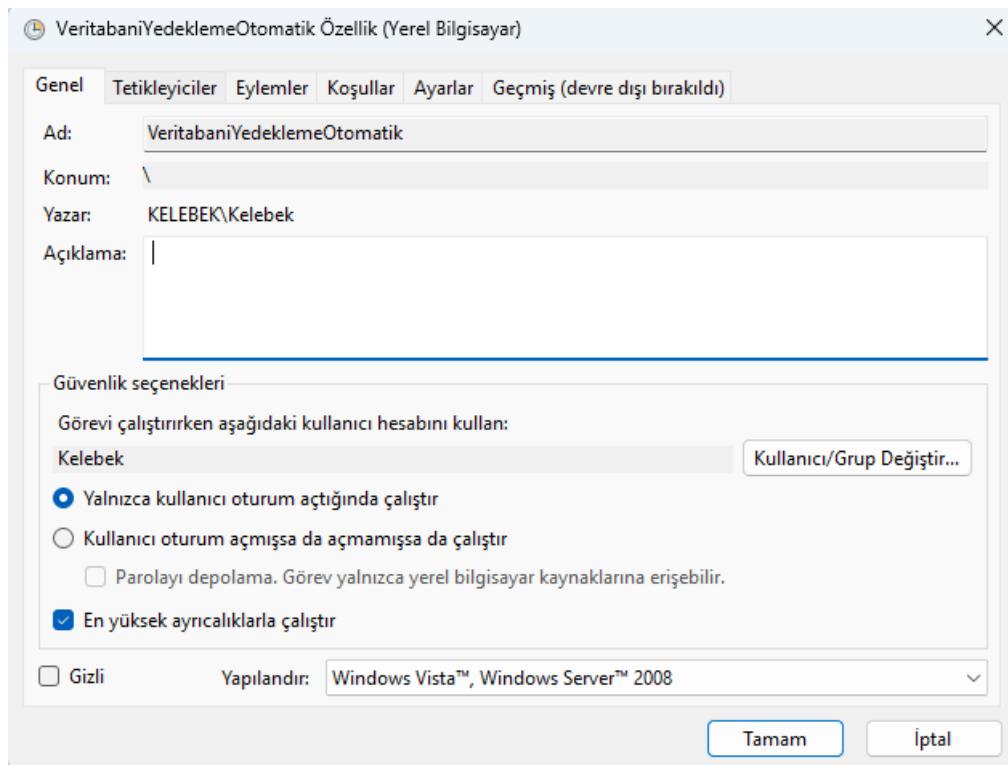
Şekil 4 Kullanıcılar Tablosundaki Veriler

8.3 Otomatikleştirme: Görev Zamanlayıcı ile Yedekleme

Veritabanı yedekleme işleminin her gün belirli bir saatte otomatik olarak çalıştırılması için Windows Görev Zamanlayıcı kullanılmıştır.

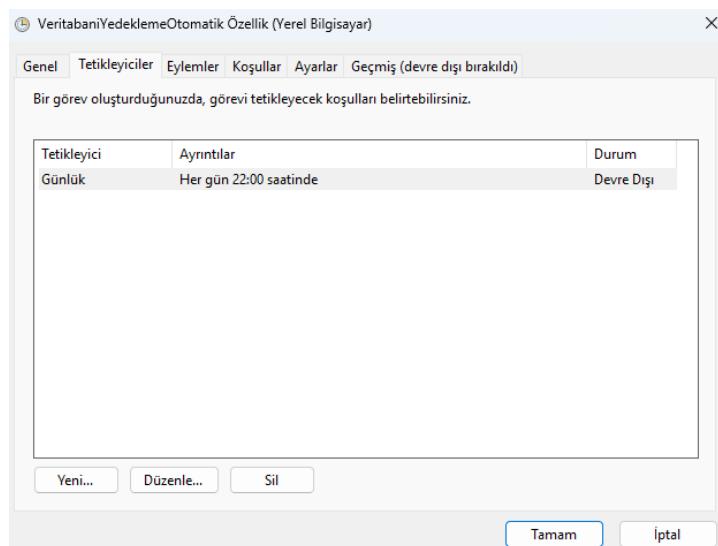
Tanımlanan Görev Ayarları:

- **Görev Adı:** VeritabaniYedeklemeOtomatik
- **Kullanıcı:** KELEBEK\Kelebek
- **Yetki:** En yüksek ayrıcalıklarla çalıştır
- **Durum:** (Geçici olarak) devre dışı



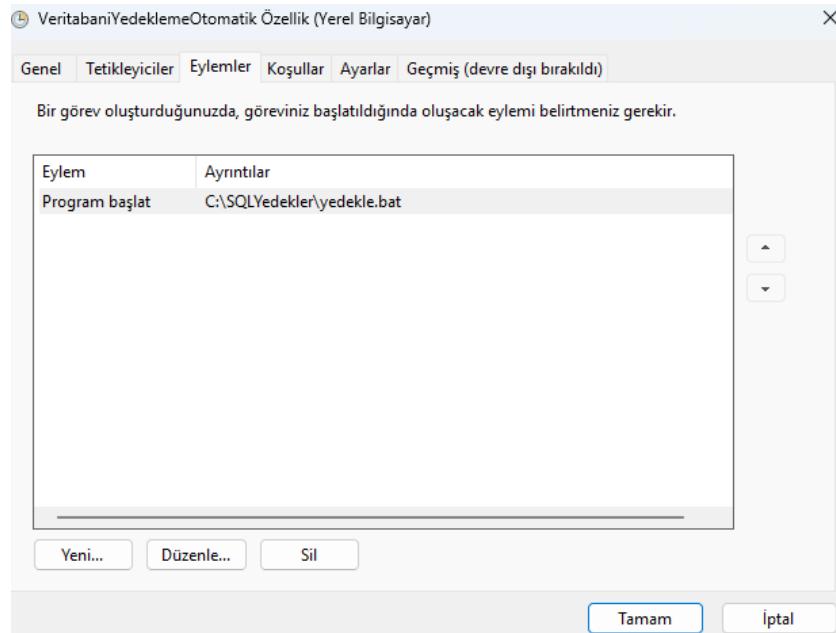
Tetikleyici:

- Her gün saat **22:00**'de çalışacak şekilde ayarlandı.



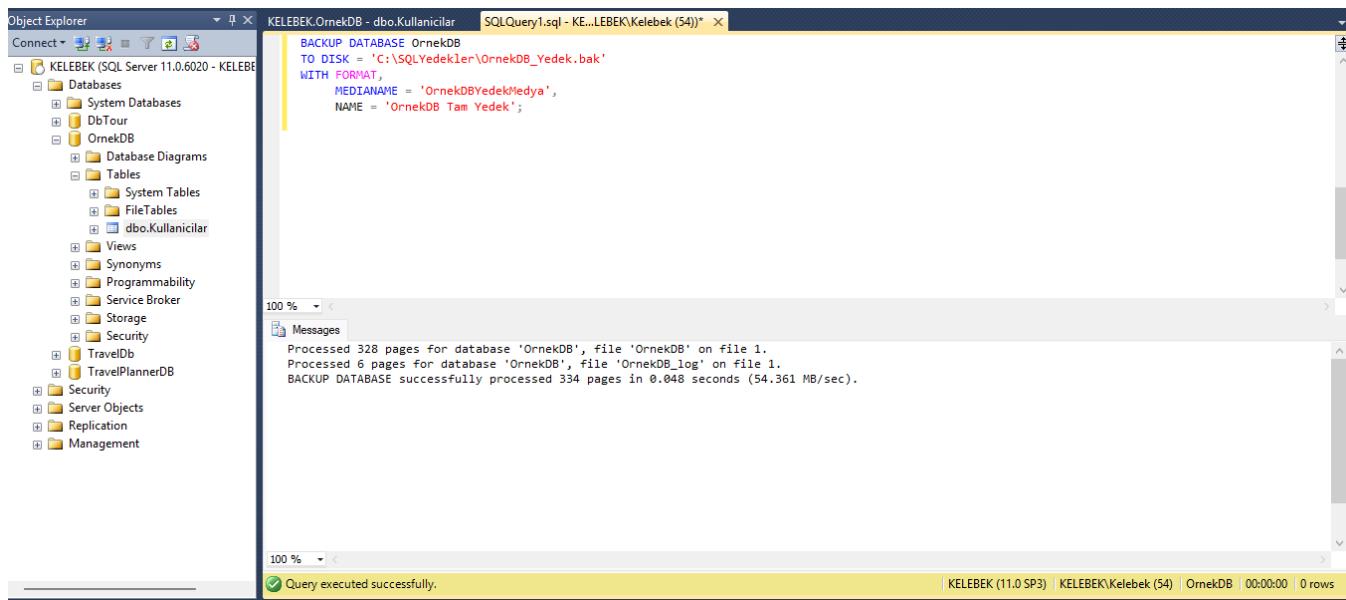
Eylem:

- C:\SQLYedekler\yedekle.bat dosyasını çalıştırarak yedekleme ve loglama işlemini başlatır.



8.4 Veritabanı Yedeği Alma Komutunun Hazırlanması

SQL Server'da bulunan BACKUP DATABASE komutu kullanılarak OrnekDB veritabanının .bak uzantılı bir yedeği alındı. Yedekleme işlemi elle yapılabileceği gibi, bu projede otomasyon için .bat dosyasıyla komut hazırlandı.

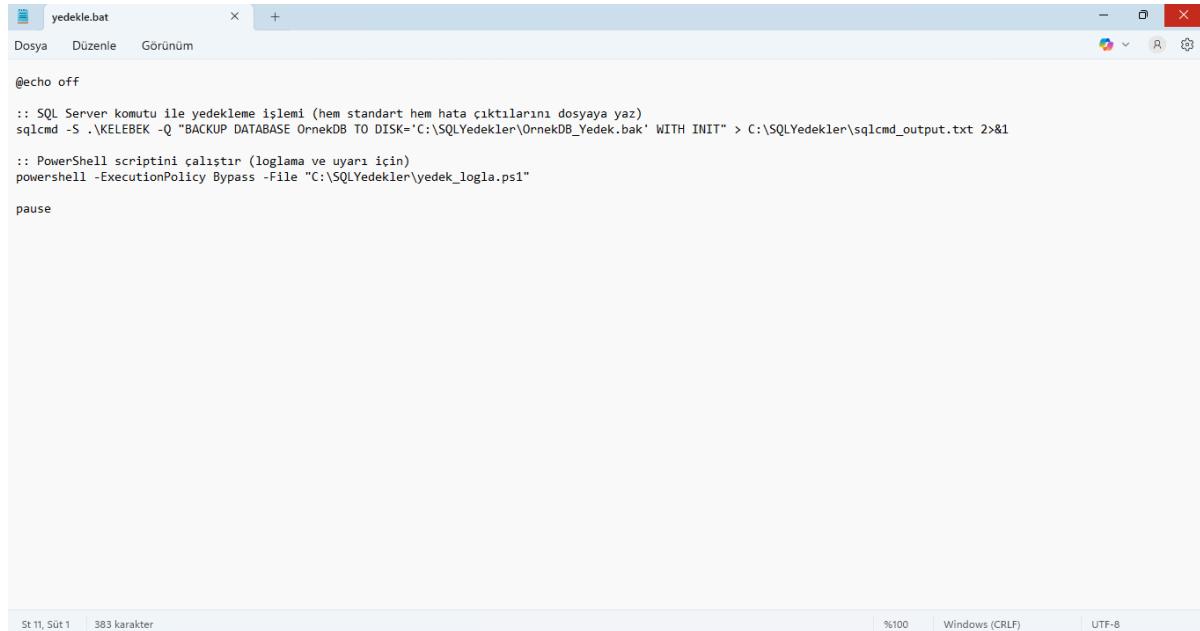


Şekil 5 Yedek Alma

8.5 Batch Dosyasının Oluşturulması (yedekle.bat)

Veritabanı yedeğini komut satırından otomatik almak için bir batch dosyası oluşturuldu. Bu dosya:

- SQL yedek komutunu çalıştırır.
- Çıktıyı .txt dosyasına kaydeder.
- Yedekleme sonrası loglama için PowerShell scriptini tetikler.



```

yedekle.bat
Dosya Düzenle Görünüm
@echo off
:: SQL Server komutu ile yedekleme işlemi (hem standart hem hata çıktılarını dosyaya yaz)
sqlcmd -S .\KELEBEK -Q "BACKUP DATABASE OrnekDB TO DISK='C:\SQLYedekler\OrnekDB_Yedek.bak' WITH INIT" > C:\SQLYedekler\sqlcmd_output.txt 2>&1
:: PowerShell scriptini çalıştır (loglama ve uyarı için)
powershell -ExecutionPolicy Bypass -File "C:\SQLYedekler\yedek_logla.ps1"
pause

```

St 11, Süt 1 383 karakter %100 Windows (CRLF) UTF-8

Şekil 6 Batch Dosyasının Oluşturulması

Açıklama:

- ✓ **sqlcmd:** SQL Server'a bağlanıp yedek komutunu çalıştırır.
- ✓ **... 2>&1:** Hem başarılı hem hatalı çıktıları sqlcmd_output.txt dosyasına yazar.
- ✓ **powershell:** Yedekleme sonucunu kontrol edip loglayan PowerShell dosyasını çalıştırır.
- ✓ **pause:** Dosya çalıştırıldıktan sonra komut ekranının kapanmasını önler, kullanıcıya sonucu gösterir.

8.6 PowerShell Scripti ile Loglama

Batch dosyası çalıştırıldıktan sonra PowerShell scripti devreye girer:

- Yedeklemenin başarılı mı yoksa hatalı mı olduğunu SQLCMD çıktısından kontrol eder.
- Sonucu tarih-saat bilgisiyle birlikte bir log dosyasına (yedek_log.txt) yazar.

```
# yedek_logla.ps1
$logDosyası = "C:\SQLYedekler\yedek_log.txt"
$tarih = Get-Date -Format "yyyy-MM-dd HH:mm:ss"

Add-Content -Path $logDosyası -Value "$tarih - PowerShell script çalıştı."

# sqlcmd çıktı dosyasını kontrol et
$sqlCmdOutput = Get-Content "C:\SQLYedekler\sqlcmd_output.txt" -Raw

if ($sqlCmdOutput -like "*BACKUP DATABASE successfully*") {
    Add-Content -Path $logDosyası -Value "$tarih - Yedekleme başarılı."
} else {
    Add-Content -Path $logDosyası -Value "$tarih - HATA: Yedekleme başarısız!"
}

|
```

St 16, Süt 3 520 karakter %100 Windows (CRLF) UTF-8

Şekil 7 PowerShell Scripti ile Loglama

Notlar:

- ✓ Eğer yedekleme başarılıysa yedek_log.txt dosyasına “Yedekleme başarılı” satırı eklenir.
- ✓ Eğer hata varsa, “Yedekleme başarısız!” olarak kayıt düşülür.

8.7 Raporlama ve Test Sonuçları

Bu bölümde, projenin test edilip sonuçların gözlemlendiği kısım yer alır. Burada amaç, otomatik yedekleme sisteminin başarıyla çalışıp çalışmadığını göstermektir.

Başarılı Yedekleme Testi:

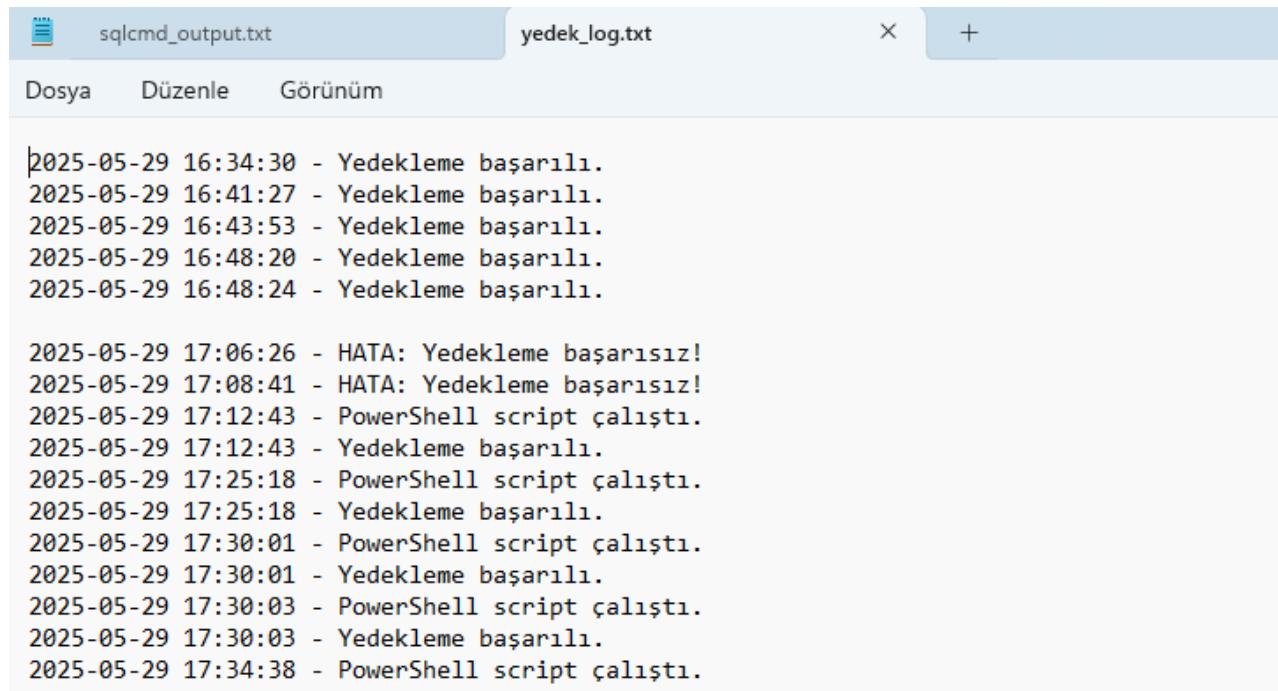
- ✓ yedekle.bat dosyası çalıştırıldı.
- ✓ OrnekDB veritabanı başarıyla .bak dosyasına yedeklendi.
- ✓ sqlcmd_output.txt dosyasında şu ifade görüldü:

```
Msg 911, Level 16, State 11, Server KELEBEK\KELEBEK, Line 1
Database 'OrnekDB' does not exist. Make sure that the name is entered correctly.
Msg 3013, Level 16, State 1, Server KELEBEK\KELEBEK, Line 1
BACKUP DATABASE is terminating abnormally.
```

St 1, Süt 1 244 karakter %100 Windows (CRLF) UTF-8

Şekil 8 test1

✓ **yedek_log.txt** dosyasına şu kayıt eklendi:



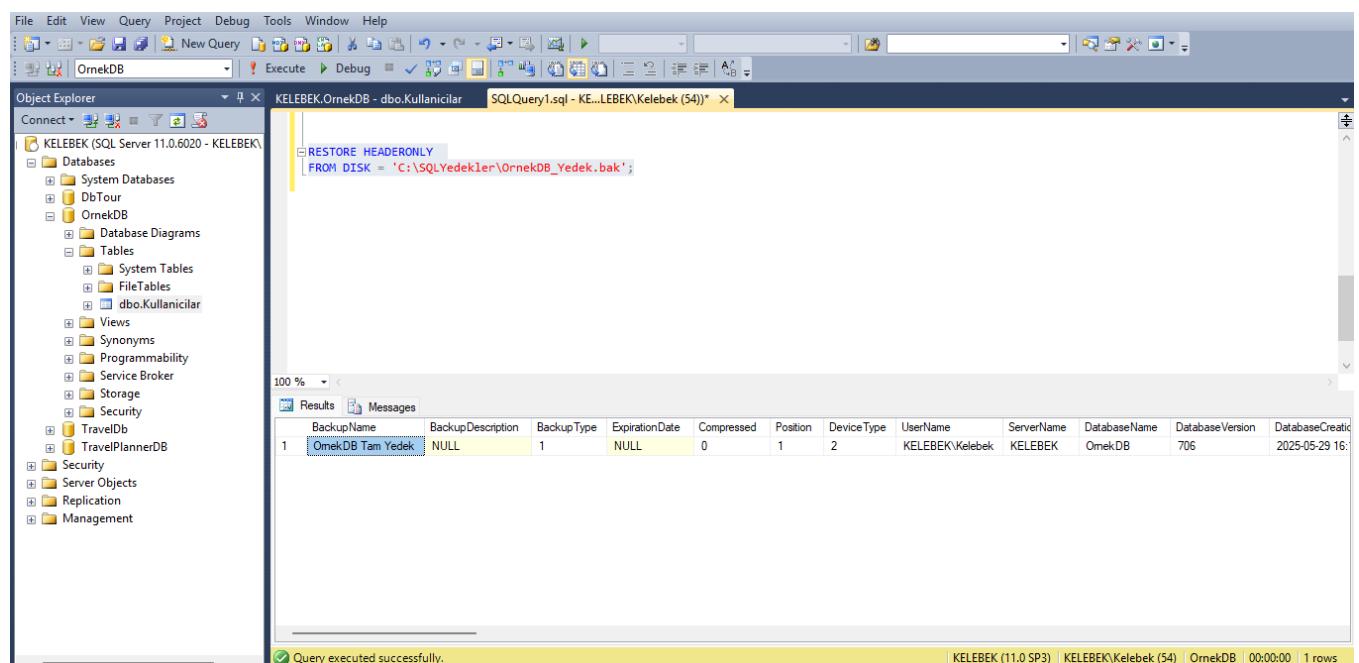
The screenshot shows a Windows Notepad window with two tabs: 'sqlcmd_output.txt' and 'yedek_log.txt'. The 'yedek_log.txt' tab is active and contains the following log entries:

```
2025-05-29 16:34:30 - Yedekleme başarılı.
2025-05-29 16:41:27 - Yedekleme başarılı.
2025-05-29 16:43:53 - Yedekleme başarılı.
2025-05-29 16:48:20 - Yedekleme başarılı.
2025-05-29 16:48:24 - Yedekleme başarılı.

2025-05-29 17:06:26 - HATA: Yedekleme başarısız!
2025-05-29 17:08:41 - HATA: Yedekleme başarısız!
2025-05-29 17:12:43 - PowerShell script çalıştı.
2025-05-29 17:12:43 - Yedekleme başarılı.
2025-05-29 17:25:18 - PowerShell script çalıştı.
2025-05-29 17:25:18 - Yedekleme başarılı.
2025-05-29 17:30:01 - PowerShell script çalıştı.
2025-05-29 17:30:01 - Yedekleme başarılı.
2025-05-29 17:30:03 - PowerShell script çalıştı.
2025-05-29 17:30:03 - Yedekleme başarılı.
2025-05-29 17:34:38 - PowerShell script çalıştı.
```

Şekil 9 test2

RESTORE HEADERONLY komutu, bir SQL Server yedek dosyasının (örneğin .bak uzantılı) içeriği hakkında bilgi almak için kullanılır. Bu komut, yedeğin içindeki meta verileri gösterir fakat yedeği geri yüklemez.

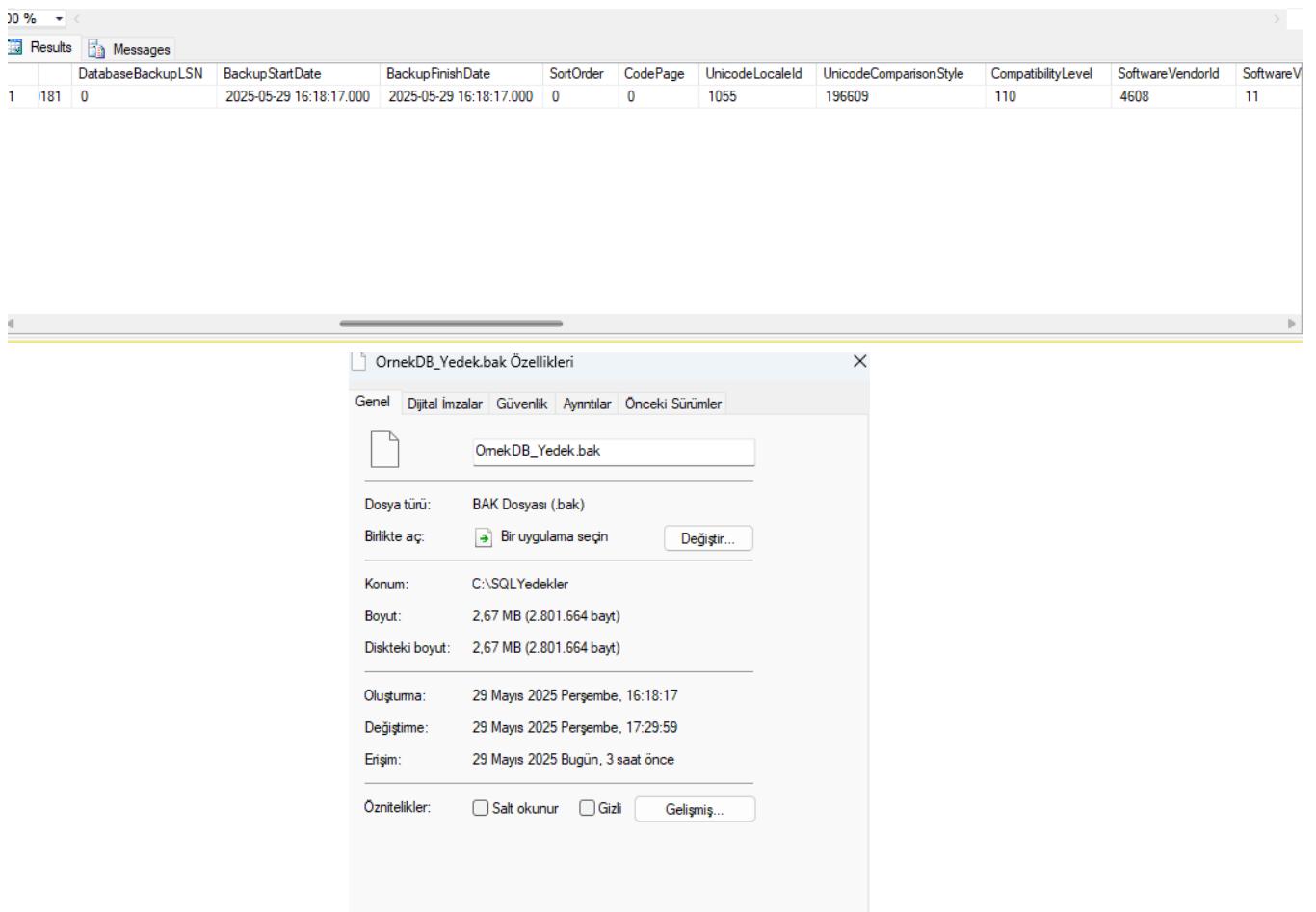


The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the database structure for 'KELEBEK' (SQL Server 11.0.6020 - KELEBEK). It includes 'Databases' (OrnekDB), 'Tables' (dbo.Kullanicilar), and other system objects.
- Query Editor:** Contains the T-SQL command:

```
RESTORE HEADERONLY  
FROM DISK = 'C:\SQLYedekler\OrnekDB_Yedek.bak';
```
- Results Grid:** Displays the results of the restore operation. A single row is shown in the 'Results' pane:

BackupName	BackupDescription	BackupType	ExpirationDate	Compressed	Position	DeviceType	UserName	ServerName	DatabaseName	DatabaseVersion	DatabaseCreate
1	OrnekDB Tam Yedek	NULL	1	NULL	0	1	2	KELEBEK\Kelebek	KELEBEK	OrnekDB	706
- Status Bar:** Shows the message "Query executed successfully." and the session details: KELEBEK (11.0 SP3) | KELEBEK\Kelebek (54) | OrnekDB | 00:00:00 | 1 rows.



Şekil 10 test3

8.8 Hatalı Durumlar için Uyarı (Opsiyonel Geliştirme)

Bu adımda, yedekleme işlemi başarısız olduğunda kullanıcıyı bilgilendirmek için ek bir kontrol ve uyarı mekanizması geliştirildi. Amaç, yedekleme işleminin güvenilirliğini artırmak ve yöneticilerin hatalardan zamanında haberdar olmasına sağlamaktır.

PowerShell scripti, SQL yedekleme işleminin çıktı dosyasını (**sqlcmd_output.txt**) analiz eder. Eğer "**BACKUP DATABASE successfully**" ifadesi bulunmazsa, log dosyasına hata mesajı eklenir ve görsel bir uyarı (popup) gösterilir.

```

# yedek_logla.ps1

$logDosyasi = "C:\SQLYedekler\yedek_log.txt"
$tarih = Get-Date -Format "yyyy-MM-dd HH:mm:ss"

Add-Content -Path $logDosyasi -Value "$tarih - PowerShell script çalıştı."

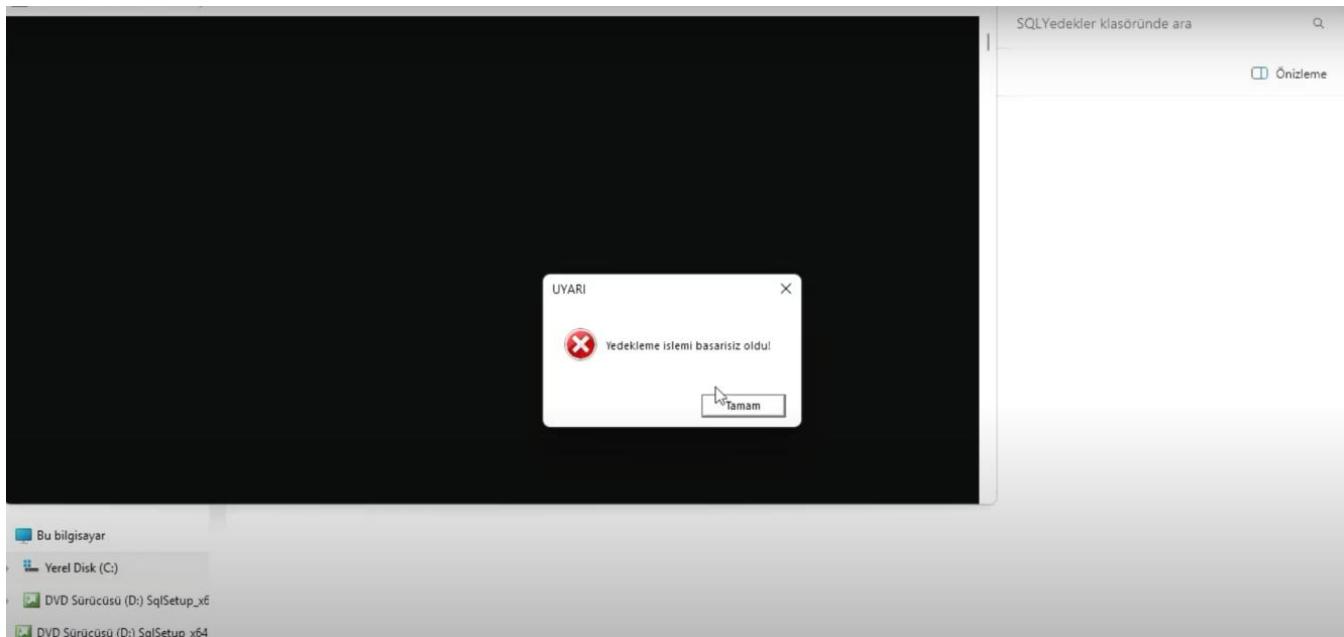
# sqlcmd çıktı dosyasını kontrol et
$sqlCmdOutput = Get-Content "C:\SQLYedekler\sqlcmd_output.txt" -Raw

if ($sqlCmdOutput -like "*BACKUP DATABASE successfully*") {
    Add-Content -Path $logDosyasi -Value "$tarih - Yedekleme başarılı."
} else {
    Add-Content -Path $logDosyasi -Value "$tarih - HATA: Yedekleme başarısız!"

    # Hata popup bildirimi
    Add-Type -AssemblyName System.Windows.Forms
    [System.Windows.Forms.MessageBox]::Show("Yedekleme işlemi başarısız oldu!", "UYARI", "OK", "Error")
}

```

Şekil 11 Hatalı Durumlar için Uyarı



Şekil 12 Uyarı mesajı

9. Sonuç

Bu rapor, SQL Server tabanlı veri tabanı yönetimi süreçlerinde gerçekleştirilen performans optimizasyonu, yedekleme stratejileri, güvenlik önlemleri, yük dengeleme, ETL süreçleri ve sürüm yönetimi gibi kritik konuları kapsamaktadır.

Veri tabanı sistemlerinin verimli, güvenli ve sürdürülebilir bir şekilde çalışmasını sağlamak amacıyla sorgu iyileştirme, indeks yönetimi, veri temizleme, erişim kontrolü ve otomatik yedekleme gibi çeşitli teknik uygulamalar hayatı geçirilmiştir. Ayrıca felaket senaryolarına hazırlık, replikasyon ve yükseltme planları da detaylı bir şekilde ele alınmıştır.

Amaç, kurumsal veri bütünlüğünü koruyarak sistem sürekliliğini sağlamak ve olası risklere karşı önceden önlem almaktır. Bu bağlamda yapılan çalışmalar, güçlü bir veri tabanı altyapısının oluşturulmasına katkı sağlamaktadır.

10.Kaynakça

1.Veritabanı Yedekleme ve Felaketten Kurtarma Planı

https://youtu.be/Z_tDQQ7cH4o

2. Veritabanı Performans Optimizasyonu ve İzleme

https://youtu.be/_7IRUNPf8KI

3. Veritabanı Güvenliği ve Erişim Kontrolü

https://youtu.be/Z_tDQQ7cH4o

4. Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları

https://youtu.be/lG_MhRHH5vA

5. Veri Temizleme ve ETL Süreçleri Tasarımı

https://youtu.be/HFIgI-v_aFo

6. Veritabanı Yükseltme ve Sürüm Yönetimi

https://youtu.be/4jqJKQoWM_s

7. Veritabanı Yedekleme ve Otomasyon Çalışması

<https://youtu.be/VzA9bT6KCwM>

8. Github

<https://github.com/JineenRihawi>

