

**ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



BLM4522 - Ağ Tabanlı Paralel Dağıtım Sistemleri

PROJE RAPORU

Sedef Kjamili & İlayda Öcal

21290519 & 21290987

github.com/sedefkjamili

Haziran 2025

GITHUB VE VİDEO BAĞLANTISI

Github Linki:

<https://github.com/sedefkjamili/blm4522-sql-server-project.git>

Video Linki:

https://drive.google.com/drive/folders/1Wc_0mbrGtNRz_Y092A0haqRybadre5-L?usp=sharing

Not: Eğer herhangi bir teknik sorunla karşılaşılırsa, yukarıdaki GitHub sayfası üzerinden videolar ve içerikler doğrudan incelenebilir.

ÖZET

Bu proje çalışması, Microsoft SQL Server ortamında kurumsal veri yönetimi süreçlerini uygulamalı olarak simüle etmeyi amaçlamaktadır. Geliştirilen uygulamalar, gerçek dünyada veri güvenliği, performans, yedekleme ve yüksek erişilebilirlik gibi temel veritabanı yönetimi başlıklarına odaklanmaktadır.

Projede ele alınan konular, büyük ölçekli veritabanı sistemlerinde karşılaşılabilircektir temel zorluklara çözüm üretmek için tasarlanmıştır. Bu kapsamda; sorgu performansını artırmak için indeks yönetimi ve DMV analizleri gerçekleştirilmiş, felaketten kurtarma senaryoları yedekleme stratejileriyle test edilmiştir. Ayrıca, kullanıcı erişim kontrolleri, SQL injection saldırılarına karşı güvenlik önlemleri ve DDL trigger ile sürüm izleme gibi veri güvenliği uygulamaları hayatı geçirilmiştir.

Bunlara ek olarak, veritabanı yükseltme senaryosu, otomatik yedekleme job'ları ve Always On ile mirroring yapılarının simülasyonları gibi yüksek erişilebilirlik odaklı uygulamalar da yapılmıştır. Bu projeler, SQL Server ortamında hem yönetsel hem de teknik yeterliliği artırmayı hedeflemektedir.

İÇİNDEKİLER

GITHUB VE VİDEO BAĞLANTISI.....	i
ÖZET	ii
İÇİNDEKİLER.....	iii
1. Veri Tabanı Güvenliği ve Erişim Kontrolü	1
1.1. SQL Server Kullanıcısı Oluşturma	1
1.2. Yetkilendirme ve Erişim Kısıtlamaları.....	2
1.3. SQL İnjeksiyon Testleri	4
1.4. SQL Server Audit ile Kullanıcı Aktivite Kaydı	5
2. Veri Temizleme ve ETL Süreçleri Tasarımı.....	8
2.1. Verideki Hatalı ve Eksik Bilgilerin Belirlenmesi (Extract).....	8
2.2. Veriyi Dönüştürme (Transform).....	9
2.3. Temiz Verilerin Yüklenmesi (Load).....	11
2.4. Veri Kalitesi Raporu.....	12
3. Veri Tabanı Yedekleme ve Felaketten Kurtarma Planı.....	13
3.1. Veri Tabanını Yedekleme	13
3.2. Veri Kaybı Simülasyonu.....	14
3.3. Yedek Geri Yükleme ve Felaketten Kurtarma	14
4. Veritabanı Yedekleme ve Otomasyon Çalışması.....	16
4.1. Veritabanını Manuel Olarak Yedekleme	16
4.2. Otomatik Yedekleme Job'unun Oluşturulması	16
4.3. Hata Uyarıları için Database Mail ve Operator Yapılandırması.....	17
4.4. Yedek Durumu Raporlama ve Test.....	19
5. Veritabanı Performans Optimizasyonu ve İzleme.....	21
5.1. SQL Profiler ile Sorgu İzleme	21
5.2. İzleme Süreci	21
5.3. Sorgu Uygulaması ve İzleme	22
5.4. SQL Server Profiler'da izlenen Sorgular	23
5.5. Dynamic Management Views (DMV) ile Performans Analizi	24
5.6. İndeks Yönetimi.....	24
5.7. Örnek İndeks Oluştur	25
5.8. Sorgu İyileştirme (Sorgu Optimizasyonu).....	25

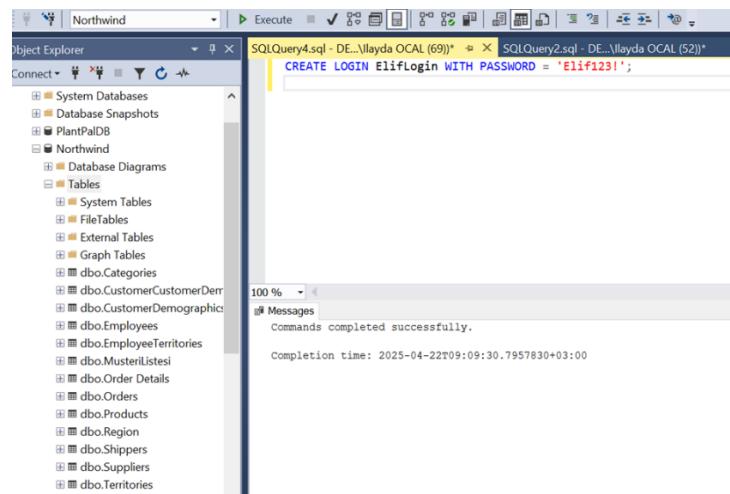
5.9.	Yavaş sorgu	26
5.10.	Optimize edilmiş versiyonu:.....	26
5.11.	Veri Yöneticisi Rollerİ (Erişim Yönetimi).....	27
6.	Veritabanı Yükseltme ve Sürüm Yönetimi.....	28
6.1.	Veritabanı Yükseltme Planı	28
6.2.	Sürüm Yönetimi (DDL Trigger kullanımı ile)	28
6.3.	Test ve Geri Dönüş Planı	30
7.	Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları.....	32
7.1.	Veritabanı Replikasyonu	32
7.2.	Yük Dengeleme (Always On / Mirroring)	34
7.3.	Failover Senaryoları	35
	SONUÇ	37
	KAYNAKLAR	38

1. Veri Tabanı Güvenliği ve Erişim Kontrolü

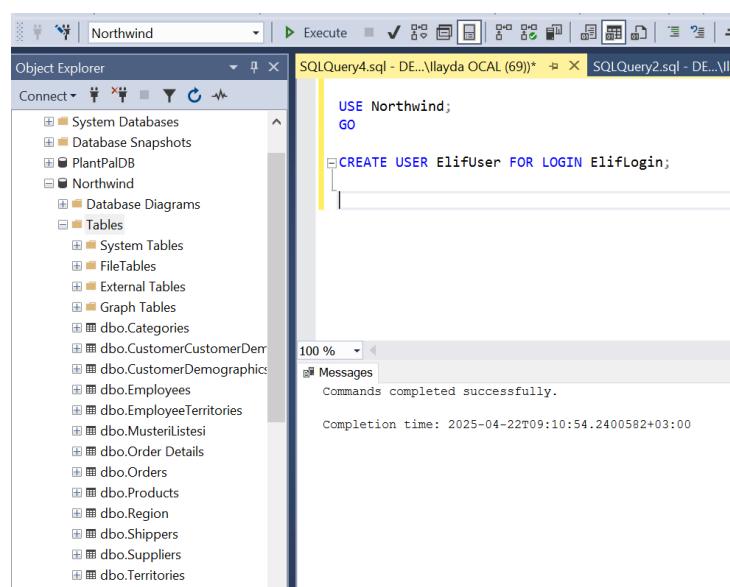
Projenin bu bölümünde, SQL Server ortamında veri güvenliği ilkelerini uygulayarak kullanıcı erişimlerini kontrol altına alma, saldırılara karşı savunma mekanizmaları kurma ve kullanıcı aktivitelerini denetleyebilme yetenekleri test edilmiştir.

1.1. SQL Server Kullanıcısı Oluşturma

SQL Server Authentication yöntemiyle “ElifLogin” adında yeni bir kullanıcı oluşturulmuş, ardından bu kullanıcıya “Northwind” veri tabanı özelinde “ElifUser” adıyla bir kullanıcı nesnesi tanımlanmıştır. Bu işlem, yetki yönetimi açısından özelleştirilmiş erişim tanımlamayı mümkün kılar.



The screenshot shows the SSMS interface with the Northwind database selected. In the Object Explorer, the tables under the Northwind database are listed. In the center pane, a query window displays the T-SQL command: `CREATE LOGIN ElifLogin WITH PASSWORD = 'Elif123!';`. The status bar at the bottom right indicates "Commands completed successfully." and "Completion time: 2025-04-22T09:09:30.7957830+03:00".

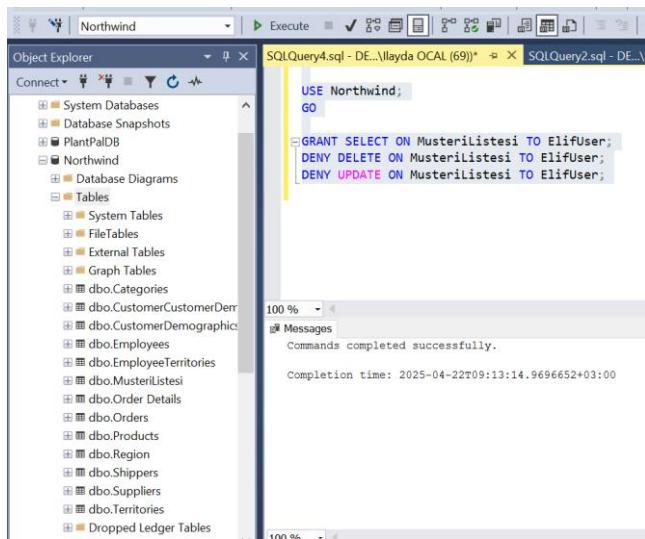


The screenshot shows the SSMS interface with the Northwind database selected. In the Object Explorer, the tables under the Northwind database are listed. In the center pane, a query window displays the T-SQL command: `USE Northwind; GO CREATE USER ElifUser FOR LOGIN ElifLogin;`. The status bar at the bottom right indicates "Commands completed successfully." and "Completion time: 2025-04-22T09:10:54.2400582+03:00".

Şekil 1.1. Northwind veri tabanında yeni bir kullanıcı oluşturma

1.2. Yetkilendirme ve Erişim Kısıtlamaları

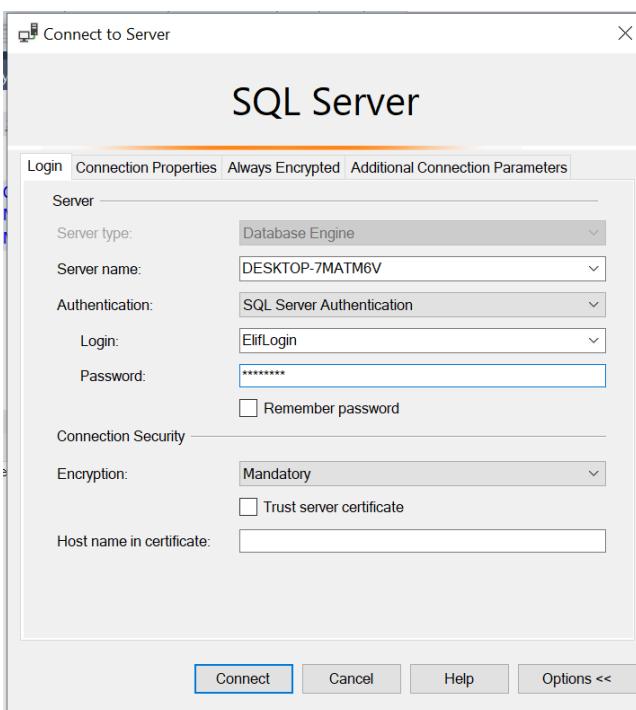
ElifUser kullanıcısına “MusteriListesi” tablosunda sadece SELECT (okuma) izni verilmiş, ancak UPDATE ve DELETE gibi veri üzerinde değişiklik yapabilecek işlemler özel olarak engellenmiştir (DENY). Bu yapı, hassas veriye sadece okuma erişimi sağlarken, veri bütünlüğünü korumayı garanti altına alır. Test işlemlerinde bu kısıtlamalar başarıyla doğrulanmıştır.



```
USE Northwind;
GO

GRANT SELECT ON MusteriListesi TO ElifUser;
DENY DELETE ON MusteriListesi TO ElifUser;
DENY UPDATE ON MusteriListesi TO ElifUser;
```

Şekil 1.2. Oluşturulan kullanıcıya yetkilerini verme işlemi



Şekil 1.3. Oluşturulan kullanıcı adı ve şifresi ile veri tabanına giriş yapma

ElifUser kullanıcısına ilk olarak izin verilen yetki SELECT denenmiş ve tablodaki değerlere ulaşabildiği kanıtlanmıştır.

CustomerID	FirmaAdı	AdSoyad	ContactTitle	Address	City	Region	PostalCode	Country	Phone	Fax
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	NULL	12209	Germany	030-0074321	030-00
ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda de la Constitución 2222	México D.F.	NULL	05021	Mexico	(5) 555-4729	(5) 55
ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Mataderos 2312	México D.F.	NULL	05023	Mexico	(5) 555-3932	NULL
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	NULL	WA1 1DP	UK	(171) 555-7788	(171) 5
BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvsvägen 8	Luleå	NULL	S-958 22	Sweden	0921-12 34 65	0921-

Şekil 1.4. Kullanıcı tarafından MusteriListesi tablosunu görüntülemeyi deneme

Kullanıcıya ikinci olarak izin verilmeyen UPDATE ve DELETE gibi veri üzerinde değişiklik yapabilecek işlemler denenmiş, hata alınarak yapının doğruluğu kanıtlanmıştır.

```

USE Northwind;
GO

UPDATE MusteriListesi
SET City = 'TestŞehir'
WHERE AdSoyad = 'Maria Anders';

```

Şekil 1.5. Kullanıcı tarafından MusteriListesi tablosunda UPDATE yetki testi

The screenshot shows the SSMS interface with the Northwind database selected. In the Object Explorer, the Northwind database is expanded to show its tables, including 'MusteriListesi'. In the center pane, a query window titled 'SQLQuery3.sql' contains the following code:

```
DELETE FROM MusteriListesi
WHERE AdSoyad = 'Maria Anders';
```

In the 'Messages' tab below, an error message is displayed:

```
Msg 229, Level 14, State 5, Line 1
The DELETE permission was denied on the object 'MusteriListesi', database 'Northwind', schema 'dbo'.
```

Completion time: 2025-04-22T09:38:00.3410931+03:00

Şekil 1.6. Kullanıcı tarafından MusteriListesi tablosunda DELETE yetki testi

1.3. SQL Injection Testleri

Veri tabanı güvenliği kapsamında, SQL Injection saldırılara karşı sistemin ne kadar dayanıklı olduğunu test etmek için simülasyon gerçekleştirilmiştir.

İlk sorguda, kullanıcı girişinden gelen zararlı bir ifade (' OR 1=1 --) doğrudan sorguya eklenmiş ve WHERE şartı geçersiz kılınarak tüm tablo verileri elde edilmiştir. Bu durum, klasik bir SQL injection açığıdır ve sistemin güvenlik açığı olduğunu gösterir.

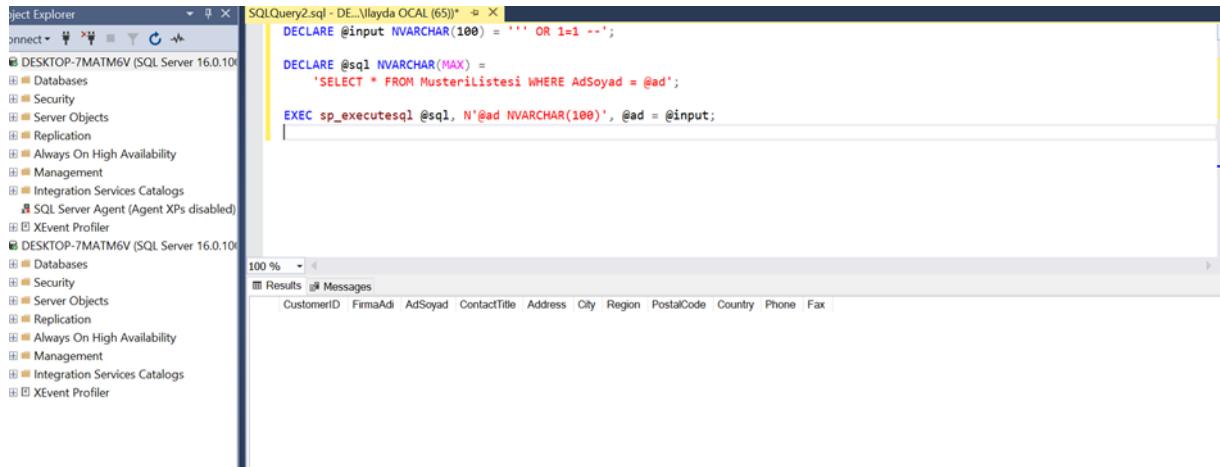
The screenshot shows the SSMS interface with the Northwind database selected. In the Object Explorer, the Northwind database is expanded to show its tables, including 'MusteriListesi'. In the center pane, a query window titled 'SQLQuery2.sql' contains the following code:

```
DECLARE @input NVARCHAR(100) = '***** OR 1=1 --';
DECLARE @sql NVARCHAR(MAX) =
    'SELECT * FROM MusteriListesi WHERE AdSoyad = ' + @input;
EXEC sp_executesql @sql;
```

The results pane shows all 91 rows from the 'MusteriListesi' table, indicating that the injection bypassed the WHERE clause.

Şekil 1.7. Zararlı SQL ifadesi (' OR 1=1 --) sisteme doğrudan girildiğinde, tüm kayıtların çekildiği gözlemlenmesi

İkinci sorguda ise aynı giriş ifadesi, "sp_executesql" prosedürüyle parametreli olarak çalıştırılmıştır. Parametre kullanımı sayesinde zararlı ifade filtrelenmiş, injection başarısız olmuş ve sistem istenmeyen verilere erişimi engellemiştir.



```

Object Explorer
DESKTOP-7MATM6V (SQL Server 16.0.1000)
  Databases
  Security
  Server Objects
  Replication
  Always On High Availability
  Management
  Integration Services Catalogs
    SQL Server Agent (Agent XPs disabled)
  XEvent Profiler
DESKTOP-7MATM6V (SQL Server 16.0.1000)
  Databases
  Security
  Server Objects
  Replication
  Always On High Availability
  Management
  Integration Services Catalogs
  XEvent Profiler

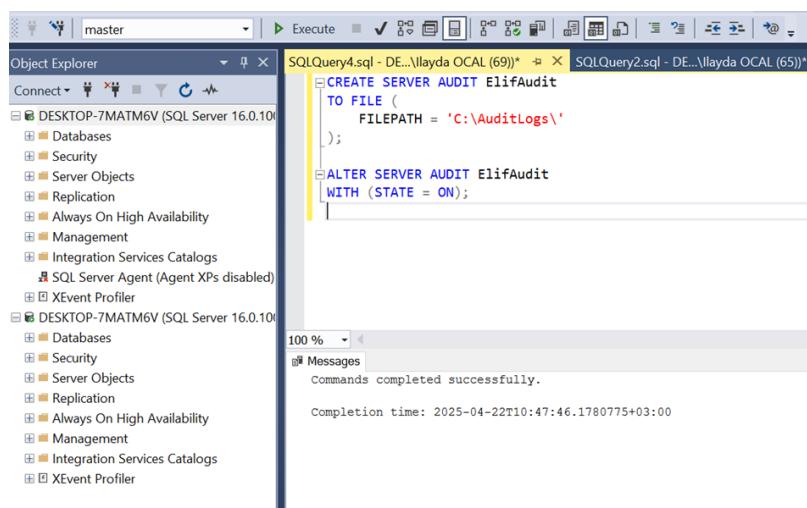
SQLQuery2.sql - DE...\\layda OCAL (65)*
DECLARE @input NVARCHAR(100) = ' OR 1=1 --';
DECLARE @sql NVARCHAR(MAX) =
'SELECT * FROM MusteriListesi WHERE AdSoyad = @ad';
EXEC sp_executesql @sql, N'@ad NVARCHAR(100)', @ad = @input;
  
```

Şekil 1.8. Aynı ifadenin parametreli sorguya denenmesi

1.4. SQL Server Audit ile Kullanıcı Aktivite Kaydı

SQL Server Audit sistemi kullanılarak belirli kullanıcı aktiviteleri izlenmiş ve kaydedilmiştir. Bu projede ElifUser kullanıcısının MusteriListesi tablosu üzerinde gerçekleştirdiği SELECT ve UPDATE gibi işlemler kayıt altına alınmıştır. Audit sistemi iki aşamalı olarak kurulmuştur.

İlk aşamada sunucu seviyesi denetimi "CREATE SERVER AUDIT" komutu ile tanımlanmış, "TO FILE" ifadesiyle log kayıtlarının yazılacağı klasör belirlenmiş ve "ALTER ... WITH (STATE = ON)" komutuyla audit sistemi aktif hale getirilmiştir.



```

master
Execute
SQLQuery4.sql - DE...\\layda OCAL (69)*
CREATE SERVER AUDIT ElifAudit
  TO FILE (
    FILEPATH = 'C:\\AuditLogs\\'
  );
ALTER SERVER AUDIT ElifAudit
  WITH (STATE = ON);

  
```

Commands completed successfully.
Completion time: 2025-04-22T10:47:46.1780775+03:00

Şekil 1.9. Denetim dosyasının sistemde oluşturulması

Audit sisteminin ikinci aşamasında veri tabanı seviyesi denetimi yapılmıştır. İlgili kullanıcı için “CREATE DATABASE AUDIT SPECIFICATION” komutu ile veri tabanı düzeyinde bir denetim tanımı oluşturulmuş, “FOR SERVER AUDIT ElifAudit” ifadesiyle bu tanım daha önce oluşturulan sunucu düzeyindeki ElifAudit ile ilişkilendirilmiştir. “ADD (...) BY ElifUser” komutu ile hangi işlemlerin hangi kullanıcıya ait olarak izleneceği belirlenmiş, son olarak “WITH (STATE = ON)” komutu ile bu denetim aktif hale getirilmiştir.

```

USE Northwind;
GO

CREATE DATABASE AUDIT SPECIFICATION ElifAuditSpec
FOR SERVER AUDIT ElifAudit
ADD (SELECT ON OBJECT::dbo.MusteriListesi BY ElifUser),
ADD (UPDATE ON OBJECT::dbo.MusteriListesi BY ElifUser)
WITH (STATE = ON);
  
```

Şekil 1.10. Kullanıcı için denetimin aktif hale getirilmesi

ElifUser olarak SELECT ve deneme amaçlı UPDATE işlemleri denenmiş, bu işlemler log dosyasına yazılmıştır.

```

USE Northwind;
GO

SELECT TOP 1 * FROM MusteriListesi;
UPDATE MusteriListesi
SET City = 'DenetimTest'
WHERE AdSoyad = 'Maria Anders';
  
```

CustomerID	FirmaAdı	AdSoyad	ContactTitle	Address	City	Region	PostalCode	Country	Phone	Fax
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Oberse Str. 57	TesfŞehir	NULL	12209	Germany	030-0074321	030-0076545

Şekil 1.11. Kullanıcı için izin verilen SELECT komutu sonucu

```

(1 row affected)
Msg 229, Level 14, State 5, Line 7
The UPDATE permission was denied on the object 'MusteriListesi', database 'Northwind', schema 'dbo'.

Completion time: 2025-04-22T11:09:51.8781147+03:00

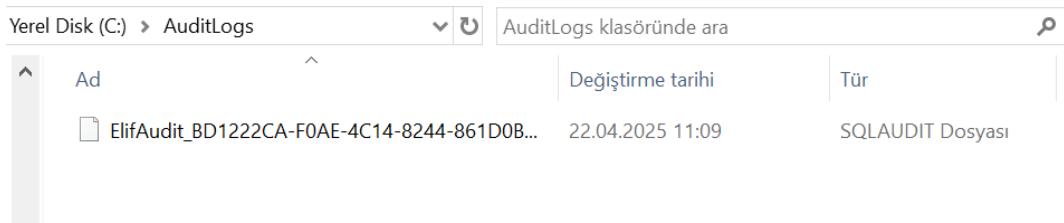
```

Şekil 1.12. Kullanıcı için izin verilmeyen UPDATE komutu engellenmesi sonucu hata mesajı

Kullanılan fonksiyon ile, belirlenen klasörde yer alan audit log dosyaları okunmuş, bu log dosyalarının tutulduğu dizin ve okunacak dosya formatı belirtilmiştir. Bu sayede, sistemde gerçekleşen denetim kayıtlarına erişim sağlanmıştır.

event_time	sequence_number	action_id	succeeded	permission_bitmask	is_column_permission	session_id	server_principal_id	database_principal_id
2025-04-22 07:47:46.1780775	1	AUSC	1	0x00000000000000000000000000000000	0	69	259	0
2025-04-22 08:00:11.2545661	1	SL	1	0x00000000000000000000000000000001	1	53	266	5
2025-04-22 08:00:11.2545661	1	UP	0	0x00000000000000000000000000000002	1	53	266	5
2025-04-22 08:08:57.0690902	1	SL	1	0x00000000000000000000000000000001	1	53	266	5
2025-04-22 08:08:57.0690902	1	SL	1	0x00000000000000000000000000000001	1	53	266	5
2025-04-22 08:08:57.0690902	1	UP	0	0x00000000000000000000000000000002	1	53	266	5
2025-04-22 08:09:40.9070666	1	SL	1	0x00000000000000000000000000000001	1	53	266	5
2025-04-22 08:09:51.8313673	1	SL	1	0x00000000000000000000000000000001	1	53	266	5
2025-04-22 08:09:51.8313673	1	SL	1	0x00000000000000000000000000000001	1	53	266	5
2025-04-22 08:09:51.8313673	1	UP	0	0x00000000000000000000000000000002	1	53	266	5

Şekil 1.13. "sys.fn_get_audit_file" fonksiyonu kullanılarak alınan loglarda işlem türü, zaman ve kullanıcı bilgileri



Şekil 1.14. Belirlenen dizine kaydolan audit dosyası

Bu proje kapsamında, SQL Server üzerinde kullanıcı erişim yetkileri yapılandırılmış, potansiyel güvenlik açıkları test edilmiş ve kullanıcı aktiviteleri başarıyla izlenmiştir. Yetkilendirme, SQL injection önlemleri ve audit log sistemlerinin bir arada kullanılmasıyla, güvenli ve denetlenebilir bir veri tabanı ortamı oluşturulmuştur.

2. Veri Temizleme ve ETL Süreçleri Tasarımı

Projenin bu kısmında Northwind veri tabanına veri temizleme ve ETL süreçleri tasarlanmıştır. Uygulamanın başlangıcında, Northwind veri tabanındaki “MusteriListesi” tablosuna bilinçli olarak eksik ya da yanlış değerler içeren bazı örnek kayıtlar eklenmiştir. Bu senaryo, gerçek hayatımda sistemlere manuel veri girişi sırasında ortaya çıkabilecek problemleri simüle etmek amacıyla uygulanmıştır. Devamında, bu hatalı kayıtlar üzerinde veri temizleme işlemleri gerçekleştirılmıştır. Eksik alanlar anlamlı ve standart ifadelerle güncellenmiş, biçimsel bozukluklar giderilmiştir.

Temizleme işleminin ardından, dönüştürülmüş ve düzeltilmiş veriler “TemizMusteriler” adlı yeni bir tabloya aktarılmıştır. Böylece orijinal veri korunurken temizlenmiş versiyonu da ayrı bir tabloda analiz edilebilir hale getirilmiştir.

Son olarak, süreç sonunda hazırlanan veri kalitesi raporlarıyla temizleme işleminin başarısı ölçülmüş, tabloya ait temizlik öncesi ve sonrası durumlar karşılaştırılarak değerlendirme yapılmıştır.

```
INSERT INTO MusteriListesi (CustomerID, FirmaAdi, AdSoyad, City, Phone)
VALUES
('C001', '', 'Ali Kaya', 'Ankara', '05551234567'),
('C002', 'Zirve Ltd.', NULL, 'İstanbul', '05559876543'),
('C003', 'Gizem Giyim', 'Gizem Yılmaz', NULL, ''),
('C004', 'Hatalı Şirket', 'Mehmet Taş', '', NULL);
```

Şekil 2.1. Simülasyon için eksik veri girişi

2.1. Verideki Hatalı ve Eksik Bilgilerin Belirlenmesi (Extract)

ETL sürecinin ilk adımı olan Extract (çıkarm) aşamasında, MusteriListesi tablosundaki hatalı veya eksik veriler tespit edilmiştir. Bu aşamada yapılan analizlerde veri kalitesi problemleri dikkate alınmıştır.

Uygulanan SQL sorgusu ile tablo üzerindeki bu hatalı kayıtlar filtrelenmiş ve veri temizleme süreci için işaretlenmiştir. Bu işlem, temizleme adımlarında düzeltilecek kayıtların belirlenmesine olanak tanımıştır.

```

SELECT *
FROM MusteriListesi
WHERE
    FirmaAdi IS NULL OR FirmaAdi = '' OR
    AdSoyad IS NULL OR AdSoyad = '' OR
    City IS NULL OR City = '' OR
    Phone IS NULL OR Phone = '';

```

CustomerID	FirmaAdi	AdSoyad	ContactTitle	Address	City	Region	PostalCode	Country	Phone	Fax	SehirBuyuk
C001	Ali Kaya	NULL	NULL	NULL	Ankara	NULL	NULL	05551234567	NULL	NULL	NULL
C002	Zive Ltd.	NULL	NULL	NULL	Istanbul	NULL	NULL	05559876543	NULL	NULL	NULL
C003	Gizem Gıym	Gizem Yılmaz	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
C004	Hatalı Şirket	Mehmet Taş	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Şekil 2.2. MusteriListesi tablosunda hatalı ya da eksik veri içeren kayıtların tespit edilmesi

2.2. Veriyi Dönüştürme (Transform)

ETL sürecinin ikinci adımında, daha önce tespit edilen eksik ve hatalı veriler üzerinde düzenleme ve dönüşüm işlemleri gerçekleştirılmıştır. Bu aşamanın temel amacı, mevcut veriyi daha anlamlı ve analiz edilebilir hale getirmektir.

Öncelikle MusteriListesi tablosuna “SehirBuyuk” adında yeni bir sütun eklenmiştir. Bu sütun, orijinal City kolonundaki şehir adlarının büyük harfe çevrilmiş halini tutmaktadır. Böylece aynı şehir adlarının farklı yazım tekrar eden değerler oluşturulmasının önüne geçilmiştir.

```

ALTER TABLE MusteriListesi
ADD SehirBuyuk NVARCHAR(100);

```

Commands completed successfully.

Completion time: 2025-04-22T11:20:33.3518381+03:00

Şekil 2.3. MusteriListesi tablosuna yeni sütun eklenmesi

Ardından eksik ve hatalı veriler üzerinde düzenleme ve dönüşüm işlemleri gerçekleştirilmiş, “UPDATE” komutu ile tüm şehir bilgileri dönüştürülmüş ve SehirBuyuk alanına aktarılmıştır. Bu işlemle birlikte hem orijinal değer hem de dönüşmüş hali kayıtlar üzerinde yan yana gözlemlenebilmiştir.

```

Object Explorer | Northwind
SQLQuery9.sql - DE...\\lavyda OCAL (52)*

UPDATE MusteriListesi
SET FirmaAdi = 'Bilinmiyor'
WHERE FirmaAdi IS NULL OR FirmaAdi = '';

UPDATE MusteriListesi
SET AdSoyad = 'İsim Eksik'
WHERE AdSoyad IS NULL OR AdSoyad = '';

UPDATE MusteriListesi
SET City = 'Bilinmiyor'
WHERE City IS NULL OR City = '';

UPDATE MusteriListesi
SET Phone = '0000000000'
WHERE Phone IS NULL OR Phone = '';
UPDATE MusteriListesi
SET SehirBuyuk = UPPER(City);

100 % | Messages
(1 row affected)
(2 rows affected)
(2 rows affected)
(95 rows affected)
100 % | Results
Query executed successfully.

Object Explorer | Northwind
SQLQuery9.sql - DE...\\lavyda OCAL (52)*

SELECT CustomerID, FirmaAdi, AdSoyad, City, SehirBuyuk, Phone
FROM MusteriListesi
WHERE CustomerID IN ('C001', 'C002', 'C003', 'C004');

100 % | Results
CustomerID FirmaAdi AdSoyad City SehirBuyuk Phone
1 C001 Bilinmiyor Ali Kaya Ankara ANKARA 05551234567
2 C002 Zirve Ltd. İsim Eksik İstanbul İSTANBUL 05559876543
3 C003 Gizem Giyim Gizem Yılmaz Bilinmiyor BİLİNMIYOR 0000000000
4 C004 Hatalı Şirket Mehmet Taş Bilinmiyor BİLİNMIYOR 0000000000

```

Şekil 2.4. Hatalı veya eksik verilerin dönüşümü ve şehir isimlerinin büyük harflerle SehirBuyuk tablosuna eklenmesi, listelenmesi

2.3. Temiz Verilerin Yüklenmesi (Load)

Veri temizleme ve dönüşüm işlemleri tamamlandıktan sonra, verilerin yeni ve temiz bir yapıya aktarılması gerekmektedir. Bu amaçla, mevcut MusteriListesi tablosundaki tüm veriler, TemizMusteriler adında yeni bir tabloya kopyalanmıştır. Bu işlem sayesinde orijinal tablo bozulmadan korunmuş, düzenlenmiş veriler ise analiz ve raporlama gibi işlemler için ayrı bir ortamda güvenli şekilde saklanmıştır.

The screenshot shows the Object Explorer and SQL Query window in SSMS. In the Object Explorer, the 'Tables' node under 'Northwind' database is expanded, showing 'MusteriListesi' and 'TemizMusteriler'. In the SQL Query window, a query is run to insert all rows from 'MusteriListesi' into 'TemizMusteriler'. The results show 91 rows affected, and the message 'Query executed successfully.' is displayed at the bottom.

```
SELECT *  
INTO TemizMusteriler  
FROM MusteriListesi;
```

Completion time: 2025-04-22T11:24:33.6969753+03:00

91 rows affected

Query executed successfully.

Şekil 2.5. TemizMusteriler adlı yeni bir tablo oluşturulması ve temiz verilerin MusteriListesi tablosundan bu tabloya aktarılması

The screenshot shows the Object Explorer and SQL Query window in SSMS. The 'Tables' node under 'Northwind' database is expanded, showing 'TemizMusteriler'. In the SQL Query window, a query is run to select specific columns (CustomerID, FirmaAdı, AdSoyad, City, SehirBuyuk, Phone) from 'TemizMusteriler' where CustomerID is in ('C001', 'C002', 'C003', 'C004'). The results grid shows four rows of data.

```
SELECT CustomerID, FirmaAdı, AdSoyad, City, SehirBuyuk, Phone  
FROM TemizMusteriler  
WHERE CustomerID IN ('C001', 'C002', 'C003', 'C004');
```

CustomerID	FirmaAdı	AdSoyad	City	SehirBuyuk	Phone
C001	Bilinmiyor	Ali Kaya	Ankara	ANKARA	05551234567
C002	Zirve Ltd.	İsim Eksik	İstanbul	İSTANBUL	05559876543
C003	Gizem Giym	Gizem Yılmaz	Bilinmiyor	BİLINMIYOR	0000000000
C004	Hatalı Şirket	Mehmet Taş	Bilinmiyor	BİLINMIYOR	0000000000

Şekil 2.6. TemizMusteriler tablosuna başarılı şekilde veri aktarıldığı ve kayıtların eksiksiz olarak taşındığı SQL sorgusu

2.4. Veri Kalitesi Raporu

Bu raporun amacı, temizleme süreci öncesinde var olan eksik veya hatalı verilerin sayısını belirlemek ve temizlik sonrasında sistemde bu tür verilerin kalıp kalmadığını tespit etmektir.

Bu sorgu sonucunda Eksik Şehir değeri 0 olarak bulunmuş, yani tüm eksik şehir bilgileri başarılı şekilde doldurulmuş ve MusteriListesi tablosundaki tüm kayıtların eksiksiz şekilde yeni tabloya aktarıldığı gösterilmiştir.

The screenshot shows the SQL Server Management Studio interface. On the left is the Object Explorer pane, which lists the Northwind database's schema, including tables like MusteriListesi and TemizMusteriler. In the center is the SQL Query window titled 'SQLQuery9.sql'. It contains two SELECT statements:

```
SELECT COUNT(*) AS EksikSehir
FROM MusteriListesi
WHERE City IS NULL OR City = '';

SELECT COUNT(*) AS TemizKayit FROM TemizMusteriler;
```

Below the query window is the Results pane, which displays the execution results:

	EksikSehir
1	0

	TemizKayit
1	95

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

Şekil 2.7. Eksik veri ve temiz kayıt bilgi raporu

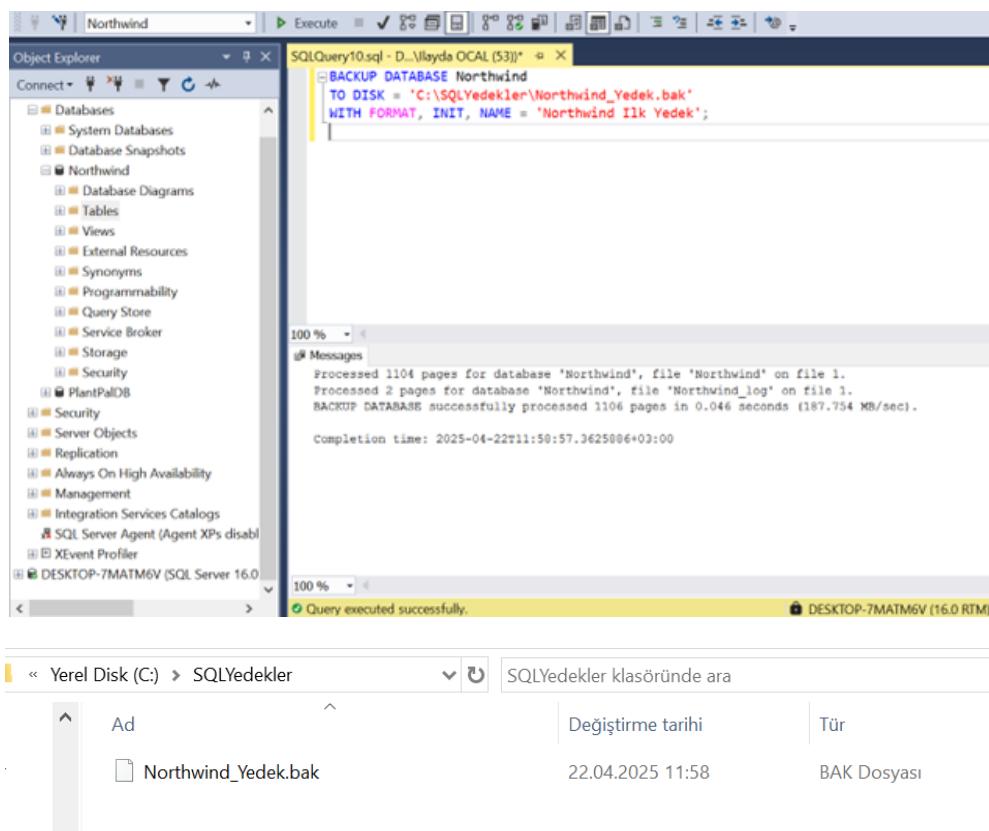
Bu proje kapsamında uygulanan ETL süreci sayesinde, veri tabanındaki eksik, hatalı veya biçimsel olarak uyumsuz kayıtlar başarılı bir şekilde tespit edilmiş, dönüştürülmüş ve temiz bir yapıya kavuşturulmuştur. Temizlenen veriler ayrı bir tabloda saklanarak hem veri bütünlüğü korunmuş hem de analiz süreçlerine hazır hale getirilmiştir. Bu çalışma, güvenilir veri yönetimi açısından ETL süreçlerinin ve analizlerin önemini açıkça ortaya koymuştur.

3. Veri Tabanı Yedekleme ve Felaketten Kurtarma Planı

Projenin bu kısmında Northwind veri tabanının yedekleme ve felaketten kurtarma planlarının tasarlanmıştır. Senaryo gereği TemizMusteriler tablosu silinmiştir. Daha sonra alınan yedek dosyası farklı bir adla (Northwind_Kurtarıldı) geri yüklenmiş ve kaybolan tablo başarıyla kurtarılmıştır. Tüm işlemler ekran görüntüleriyle belgelenmiştir.

3.1. Veri Tabanını Yedekleme

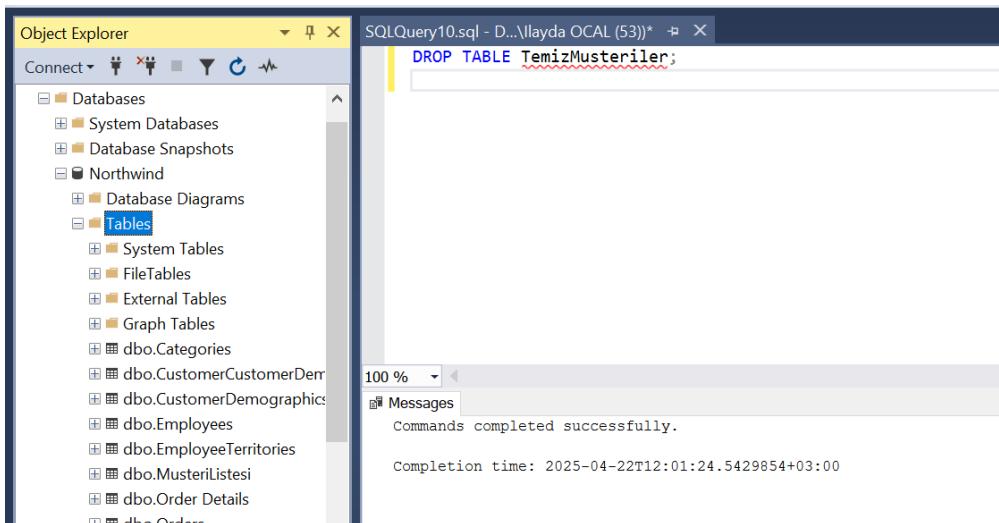
Northwind veri tabanının mevcut durumu, veri kaybı riskine karşı .bak uzantılı yedek dosyası olarak saklanmıştır. “BACKUP DATABASE” komutu kullanılarak veri tabanı tam yedekleme yöntemiyle belirlenen dizine kaydedilmiştir.



Şekil 3.1. Veri tabanının veri kaybı riskine karşı .bak uzantılı yedek dosya olarak saklanması

3.2. Veri Kaybı Simülasyonu

Olası bir veri kaybı durumunu simüle etmek amacıyla “TemizMusteriler” tablosu kasıtlı olarak silinmiştir. Bu işlem, gerçek dünyada kullanıcı hatası veya sistemsel bir sorun nedeniyle meydana gelebilecek felaket senaryolarını temsil etmektedir. Sonraki adımlarda bu veri kaybının yedek dosyası aracılığıyla nasıl kurtarılacağı gösterilmiştir.



Şekil 3.2. DROP TABLE komutu ile kasıtlı olarak tablonun silinmesi

3.3. Yedek Geri Yükleme ve Felaketten Kurtarma

Felaket senaryosu kapsamında, daha önce manuel olarak yedeklemesi alınan Northwind veri tabanının .bak dosyası üzerinden sisteme geri yükleme işlemi gerçekleştirilmiştir. Bu işlem, yanlışlıkla silinen bir tabloyu kurtarma senaryosunu temsil etmektedir.

Yedek, Northwind_Kurtarıldı adıyla sisteme yüklenmiş ve böylece orijinal veri tabanının birebir kopyası olan ikinci bir veri tabanı oluşturulmuştur. Bu kopya veri tabanı, hem tablo yapısını hem de verileri birebir içermektedir.

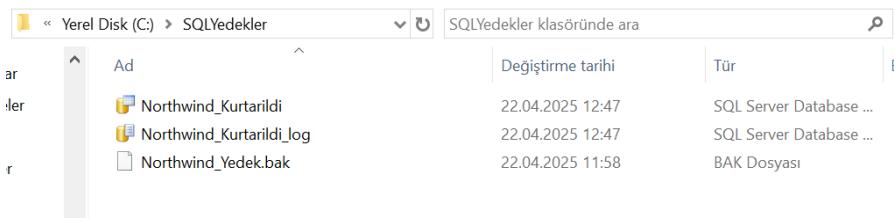
The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the database structure for 'Northwind'.
- SQL Query Editor:** Contains the T-SQL command for restoring the database:


```
RESTORE DATABASE Northwind_Kurtarildi
      FROM DISK = 'C:\SQLYedekler\Northwind_Yedek.bak'
      WITH
          MOVE 'Northwind' TO 'C:\SQLYedekler\Northwind_Kurtarildi.mdf',
          MOVE 'Northwind_log' TO 'C:\SQLYedekler\Northwind_Kurtarildi_log.ldf',
          REPLACE;
```
- Messages:** Displays the execution results:


```
Processed 1104 pages for database 'Northwind_Kurtarildi', file 'Northwind' on file 1.
      Processed 2 pages for database 'Northwind_Kurtarildi', file 'Northwind_log' on file 1.
      RESTORE DATABASE successfully processed 1106 pages in 0.080 seconds (107.958 MB/sec).

      Completion time: 2025-04-22T12:47:17.3131619+03:00
```
- Status Bar:** Shows 'Query executed successfully.' and the computer name 'DESKTOP-7MATM6V (16.0)'.



Şekil 3.3. Northwind_Kurtarildi adında yeni bir veritabanının oluşturulması ve .bak yedeğinden geri yüklenmesi

Northwind_Kurtarildi veri tabanı seçilerek TemizMusteriler tablosunun ilk 5 kaydı kontrol edilmiştir. Görsel, veri tabanı yedeğinin başarılı şekilde geri yüklendiğinin ve veri kaybının ortadan kalktığını kanıtlıdır.

The screenshot shows the SSMS interface with the following details:

- Object Explorer:** Shows the database structure for 'Northwind_Kurtarildi'.
- SQL Query Editor:** Contains the T-SQL command:


```
USE Northwind_Kurtarildi;
GO

SELECT TOP 5 * FROM TemizMusteriler;
```
- Results:** Displays the results of the query, showing 5 rows of data from the 'TemizMusteriler' table.

CustomerID	FirmaAdı	AdSoyad	ContactTitle	Address	City	Region	PostalCode	Country	Phone	Fax
ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	DenimtTest	NULL	12209	Germany	030-0074321	030-0
ANATR	Ana Trujillo Emparedados y Helados	Ana Trujillo	Owner	Avda. de la Constitución 2222	Méjico D.F.	NULL	05021	Mexico	(5) 555-4729	(5) 55
ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Mataderos 2312	Méjico D.F.	NULL	05023	Mexico	(5) 555-9932	NULL
AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	WA1 1DP	UK	(171) 555-7788	(171)	
BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvsvägen 8	Luleå	NULL	S-998 22	Sweden	0921-12 34 65	0921

Şekil 3.4. Northwind_Kurtarildi veri tabanından verilerin çekilmesi

İlgili proje kapsamında gerçekleştirilen yedek alma ve felaketten kurtarma işlemleri sonucunda, olası veri kayıplarına karşı sistemin güvenli ve sürdürülebilir olduğu kanıtlanmış, yedekleme stratejisinin etkinliği ve geri yükleme süreçlerinin başarıyla çalıştığı uygulamalı olarak test edilmiştir.

4. Veritabanı Yedekleme ve Otomasyon Çalışması

Bu bölümde Northwind veritabanı için tam yedekleme otomasyonu kurulmuş, başarısızlık durumunda bildirim mekanizması tanımlanmış ve raporlama senaryosu hayata geçirilmiştir. Tüm adımlar SSMS ekran görüntüleri ve komutlarla belgelenmiştir.

4.1. Veritabanını Manuel Olarak Yedekleme

Northwind veritabanının o anki tutarlı hâli, .bak uzantılı tam yedek dosyası olarak saklanmıştır. Aşağıdaki T-SQL komutu ile yedekleme yapılmıştır.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the top query editor window, the following T-SQL command is displayed:

```
BACKUP DATABASE Northwind  
TO DISK = N'C:\SQLYedekler\Northwind_FULL.bak'  
WITH FORMAT, NAME = 'Northwind Full Backup', STATS = 10;
```

Below the command, the results pane shows a single file named "Northwind_FULL.bak" located at "C:\SQLYedekler". The file is a BAK Dosyası (BAK File) created on 21.05.2025 08:17.

Şekil 4.1. “BACKUP DATABASE” komutuyla Northwind’ın tam yedeğinin alınması ve kaydedilmesi.

4.2. Otomatik Yedekleme Job’unun Oluşturulması

SQL Server Agent ile job oluşturabilmek için öncelikle Agent’ın Extended Stored Procedures özelliği (Agent XPs) etkinleştirilmelidir.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the "SQL Server Agent" node is expanded. In the main query editor window, the following T-SQL commands are executed:

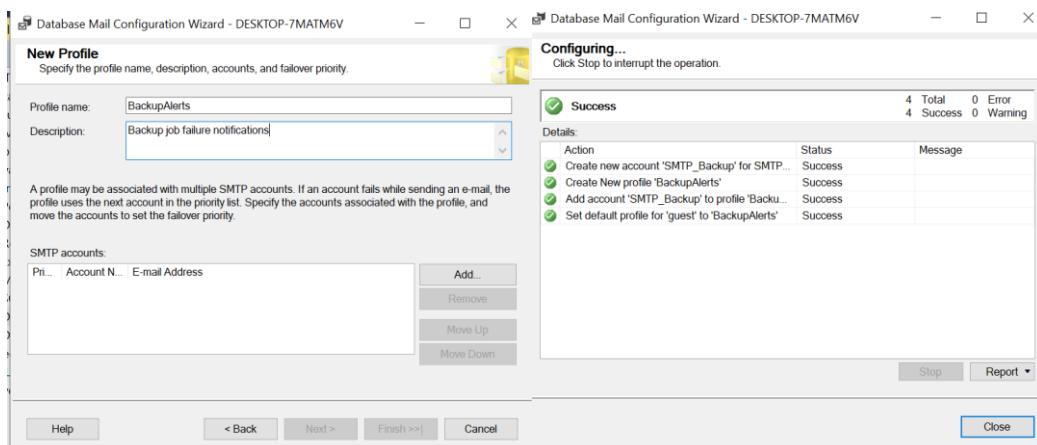
```
-- 1. Gelişmiş seçenekleri görünür yap  
EXEC sp_configure 'show advanced options', 1;  
RECONFIGURE;  
GO  
  
-- 2. Agent XPs’i etkinleştir  
EXEC sp_configure 'Agent XPs', 1;  
RECONFIGURE;  
GO
```

The execution results pane shows the configuration options being changed and the completion time.

Şekil 4.2. Agent XPs’in sp_configure komutuyla etkinleştirilmesi ve SQL Server Agent’ın Object Explorer’da görünür hâle gelmesi.

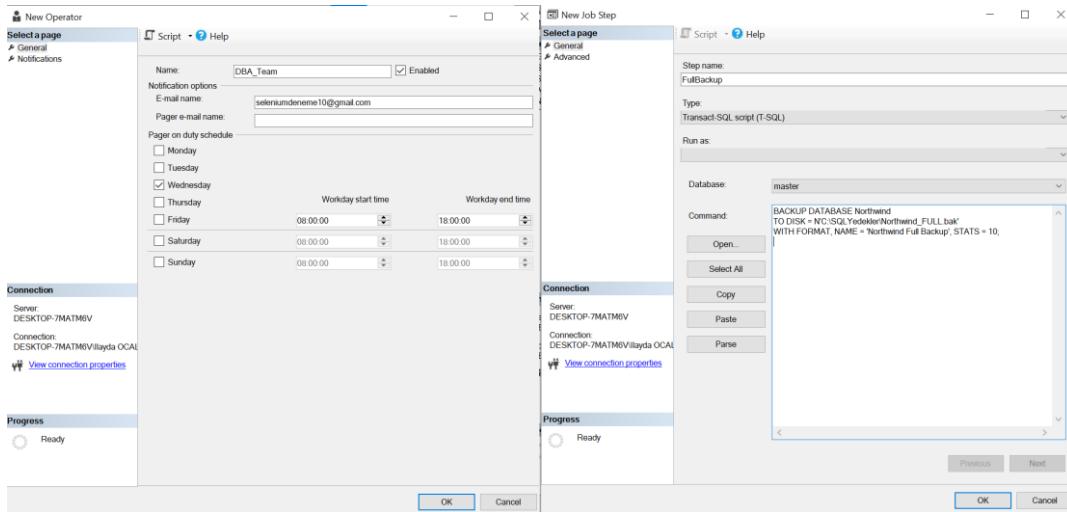
4.3. Hata Uyarıları için Database Mail ve Operator Yapılandırması

Bu adımda, SQL Server'ın otomatik yedekleme job'larından gelecek hata bildirimlerini gönderecek Database Mail profili tanımlanmıştır. Profil adı BackupAlerts olarak belirlenmiş, açıklama alanına "Backup job failure notifications" girilmiştir. Ardından SMTP hesabı (SMTP_Backup) profile eklenmiş ve profil guest kullanıcısı için varsayılan (default) olarak ayarlanmıştır. Sağdaki onay ekranı, tüm oluşturma işlemlerinin ("Create new account", "Create New profile", "Add account to profile", "Set default profile") başarıyla tamamlandığını belgelemektedir.



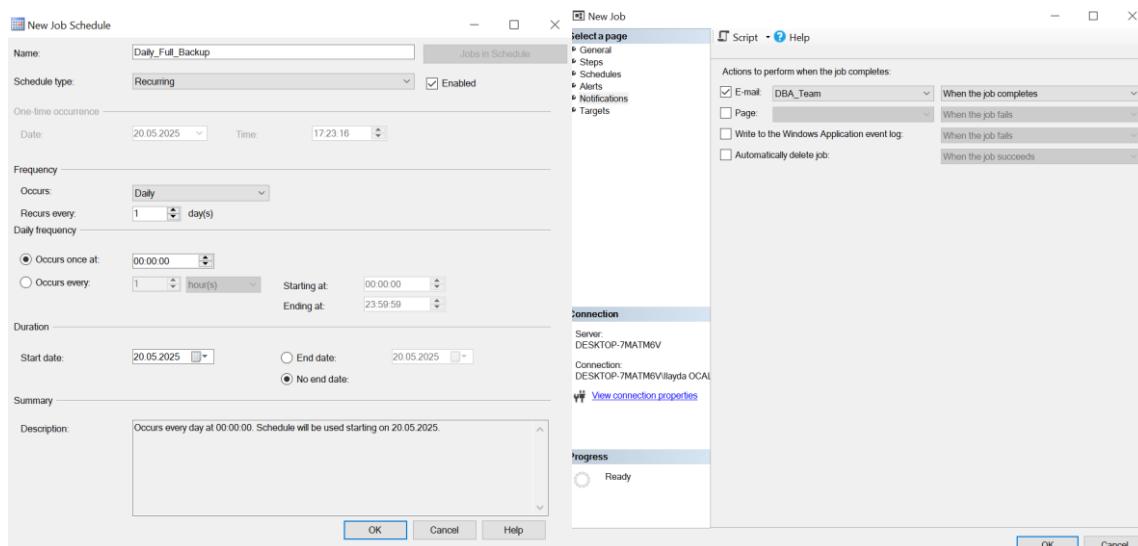
Şekil 4.3. "BackupAlerts" adlı yeni Database Mail profili sayfası; SMTP hesabı ekleme adımı için hazırlandığı ekran ve oluşturma adımlarının başarıyla tamamlandığını gösteren "Success" onay penceresi.

Bir sonraki adımda, hata veya tamamlanma bildirimleri gönderecek bir Operator (DBA_Team) oluşturulmuş ve yedekleme işleminin nasıl gerçekleştirileceğini tanımlayan Job Step (FullBackup) yapılandırılmıştır. Operator, bildirim almaya yetkili e-posta adresini belirlerken; Job Step, yedekleme komutunun parametrelerini (yedek dosyası yolu, format ve istatistik ayarları) barındırır.



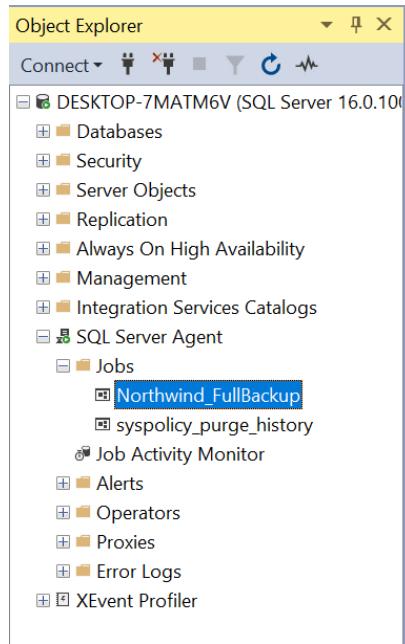
Şekil 4.4. “DBA_Team” adıyla yeni bir Operator tanımlama penceresi ve “FullBackup” adınının tanımlandığı Job Step penceresi.

Bu adım, job'un günlük otomatik tetiklenmesi ve hem başarılı tamamlama hem de hata durumlarında ilgili DBA ekibi operatörüne bildirim gönderilmesini sağlar.



Şekil 4.5. “Daily_Full_Backup” isimli schedule penceresi ve operatöre e-posta gönderilecek şekilde konfigüre edilmiş job'un Notifications sekmesi.

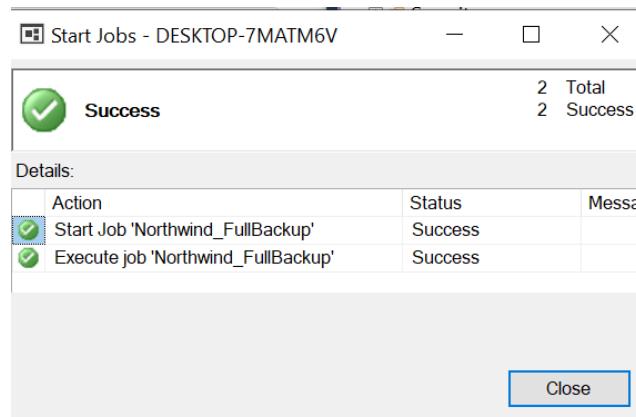
Object Explorer'da SQL Server Agent'in Jobs klasörü altında oluşturduğumuz Northwind_FullBackup job'u görmekteydi. Job'un doğru yerde oluşturulduğu ve hazır halde listelendiği bu ekranla doğrulanır.



Şekil 4.6. SQL Server Agent → Jobs altında “Northwind_FullBackup” job'unun listelenmesi.

4.4. Yedek Durumu Raporlama ve Test

Job'un otomatik çalışması dışında çalıştığını kanıtlanması için manuel olarak da çalıştırılarak, “Start Job ‘Northwind_FullBackup’” ve “Execute job ‘Northwind_FullBackup’” adımlarının **Success** olarak tamamlandığı görülmektedir. Bu, yedekleme adımının sorunsuz çalıştığını ve Northwind_FULL.bak dosyasını oluşturduğunu teyit eder.



Şekil 4.7. Job'un çalıştırılması ve her iki adımın da başarılı tamamlanması

Bir sonraki adımla beraber msdb içindeki backupset ve backupmediafamily tablolarını kullanarak Northwind için alınan Full yedeği ve dosya yolunu listelenir ve böylece otomasyonun çalışıp çalışmadığı kontrol edilir.

```

USE msdb;
GO

SELECT
    bs.database_name,
    bs.type AS backup_type,
    CASE
        WHEN bs.type = 'D' THEN 'Full'
        WHEN bs.type = 'I' THEN 'Differential'
        WHEN bs.type = 'L' THEN 'Log'
    END AS backup_desc,
    bs.backup_start_date,
    bs.backup_finish_date,
    bmf.physical_device_name AS backup_file
FROM
    dbo.backupset bs
INNER JOIN
    dbo.backupmediafamily bmf
    ON bs.media_set_id = bmf.media_set_id
WHERE
    bs.database_name = 'Northwind'

```

Results Messages

database_name	backup_type	backup_desc	backup_start_date	backup_finish_date	backup_file
Northwind	D	Full	2025-05-21 08:17:27.000	2025-05-21 08:17:27.000	C:\SQLYedekler\Northwind_FULL.bak

Şekil 4.8.MSDB'den Backup History Sorusu

Son adımda yapılan işlemlerin kullanıcıya takibinin kolay olması adına son 7 güne ait yedek bilgilerini bir metin gövdesine dönüştürüp sp_send_dbmail ile e-posta gönderen bir prosedür tanımlanır ve e-posta kuyruğa gönderilir.

```

CREATE OR ALTER PROCEDURE dbo.SendNorthwindBackupReport
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @body NVARCHAR(MAX) = N'Northwind Backup Report (Last 7 days):' + CHAR(13)+CHAR(10);

    SELECT @body += 
        FORMAT(bs.backup_finish_date, 'yyyy-MM-dd HH:mm:ss') + ' | ' +
        CASE bs.type WHEN 'D' THEN 'Full' WHEN 'I' THEN 'Differential' WHEN 'L' THEN 'Log' END + ' | ' +
        bmf.physical_device_name + CHAR(13)+CHAR(10)
    FROM msdb.dbo.backupset bs
    JOIN msdb.dbo.backupmediafamily bmf
        ON bs.media_set_id = bmf.media_set_id
    WHERE bs.database_name = 'Northwind'
        AND bs.backup_finish_date >= DATEADD(day, -7, GETDATE())
    ORDER BY bs.backup_finish_date DESC;

    EXEC msdb.dbo.sp_send_dbmail
        @profile_name = 'BackupAlerts',
        @recipients   = 'seleniumdeneme10@gmail.com',
        @subject      = 'Northwind Backup Report',
        @body         = @body;
END;

```

Messages

Commands completed successfully.

Completion time: 2025-05-21T08:25:52.5614274+03:00

SQLQuery5.sql - DE...\İladya OCAL (54)* SQLQuery4.sql - DE...\İladya OCAL (55)* SQLQuery3.sql - DE...\İladya OCAL (60)*

EXEC dbo.SendNorthwindBackupReport;

100 %

Messages

Mail (Id: 1) queued.

Completion time: 2025-05-21T08:27:40.5614040+03:00

Şekil 4.9. Stored prosedür oluşturma komutu ve prosedürün çalıştırılarak mail kuyruğuna alınması

İlgili proje kapsamında gerçekleştirilen otomatik yedekleme job'u, hata bildirim mekanizması ve raporlama adımları sayesinde Northwind veritabanının kesintisiz korunması sağlanmış; felaket senaryolarında veri kaybı önlenmiş, yedekleme stratejisinin etkinliği ve geri yükleme süreçlerinin güvenilirliği uygulamalı olarak kanıtlanmıştır.

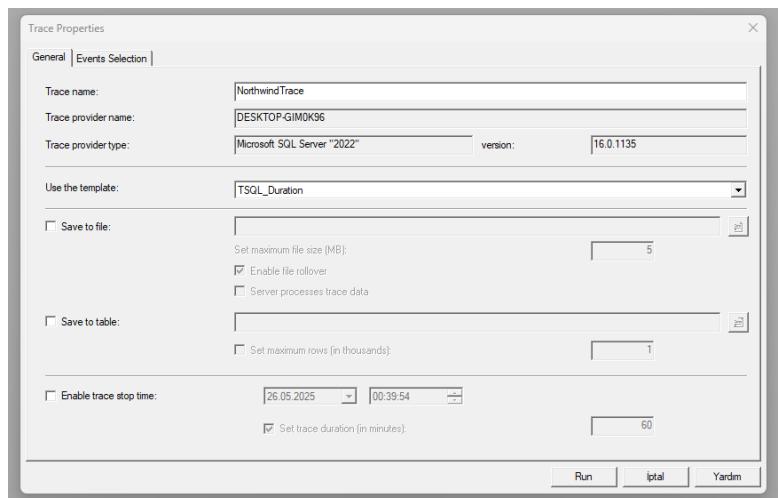
5. Veritabanı Performans Optimizasyonu ve İzleme

5.1. SQL Profiler ile Soru İzleme

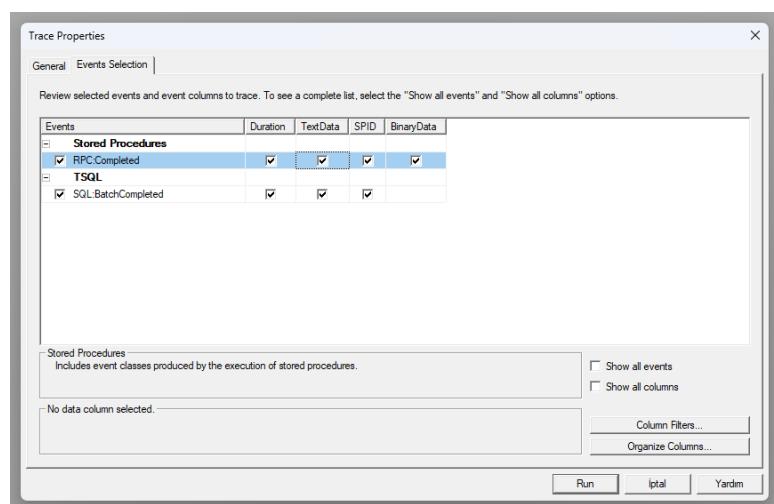
Bu çalışmada, SQL Server Profiler aracı kullanılarak Northwind veritabanı üzerindeki sorguların izlenmesi sağlanmıştır. Profiler'da 'TSQL_Duration' şablonu kullanılmış ve SQL:BatchStarting, SQL:BatchCompleted, RPC:Completed olayları seçilerek soru takibi başlatılmıştır.

5.2. İzleme Süreci

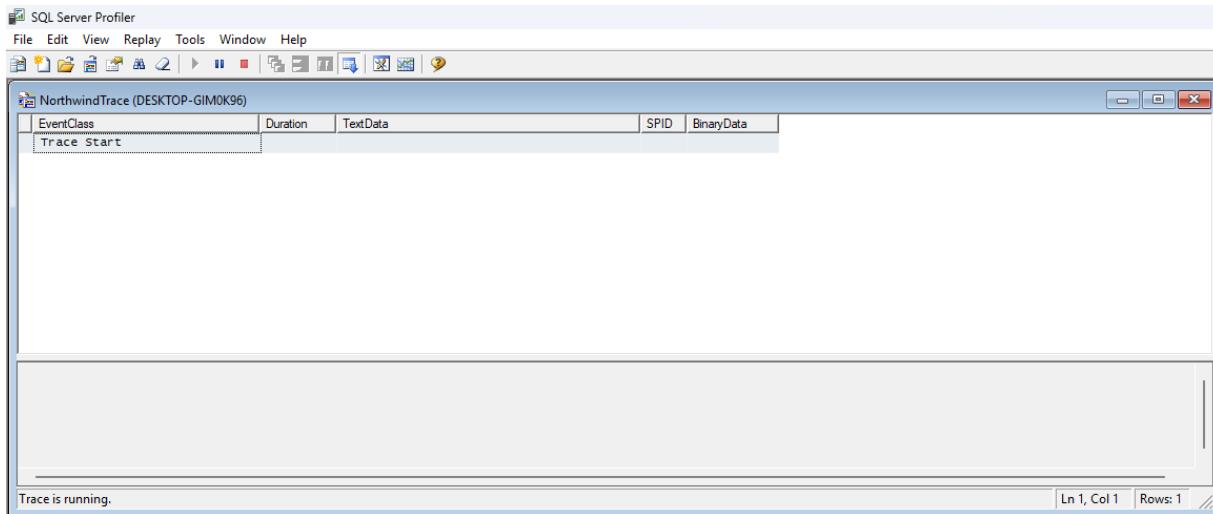
Profiler aracı çalıştırıldıkten sonra yeni bir Trace oturumu başlatılmış, TSQL_Duration şablonu seçilerek sorguların süre bazlı takibi yapılmıştır. İzleme için oluşturulan Trace'e "NorthwindTrace" adı verilmiştir.



Şekil 5.1. SQL Profiler'da Trace Ayarları Ekranı



Şekil 5.2. Profiler Olay Seçimi Ekranı



Şekil 5.3 SQL Profiler'da Trace Ayarları Ekranı

Bu sorgular SQL Server Profiler ekranında SQL:BatchCompleted olayları olarak izlenmiştir. Her satırda sorgunun tam içeriği (TextData), çalışma süresi (Duration) ve bağlantı numarası (SPID) gibi bilgiler detaylı olarak gözlemlenmiştir.

5.3. Sorgu Uygulaması ve İzleme

Trace işlemi başlatıldıktan sonra, SSMS üzerinde Northwind veritabanına çeşitli SELECT sorguları çalıştırılmıştır. Bu sorgular, sorgu yoğunluğu ve işlem süresi açısından Profiler'da izlenmiştir.

CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	PostalCode	Country	Phone
1 ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	NULL	12209	Germany	030-0074321
2 ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de la Constitución 2222	México D.F.	NULL	05021	Mexico	(5) 555-4729
3 ANTON	Antonio Moreno Taqueria	Antonio Moreno	Owner	Mataderos 2312	México D.F.	NULL	05023	Mexico	(5) 555-3932
4 AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	NULL	WA1 1DP	UK	(171) 555-7788
5 BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvvägen 8	Luleå	NULL	S-958 22	Sweden	0921-12 34 65
6 BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Fonsterstr. 57	Mannheim	NULL	68306	Germany	0621-08460
7 BLONP	Blondesddsl père et fils	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg	NULL	67000	France	88.60.15.31
8 BOLID	Bólido Comidas preparadas	Martin Sommer	Owner	C/ Araquil, 67	Madrid	NULL	28023	Spain	(91) 555 22 82

Şekil 5.4. SSMS Üzerinden Örnek Sorguların Çalıştırılması

5.4. SQL Server Profiler'da İzlenen Sorgular

SQL Server Profiler kullanılarak Northwind veritabanı üzerindeki sorgular canlı olarak izlenmiştir. İzleme işlemi sırasında 'SQL:BatchCompleted' olayları kaydedilmiştir. Bu olaylar sayesinde her bir sorgunun süresi ('Duration'), içeriği ('TextData') ve bağlantı bilgileri ('SPID') detaylı olarak gözlemlenmiştir.

Aşağıda yer alan örnekte, 'SELECT * FROM Customers` gibi sorguların başarıyla izlendiği görülmektedir. Bu bilgiler, sistemde hangi sorguların daha fazla kaynak tükettiğini ve performans darboğazlarının nerede olduğunu analiz etmek için temel teşkil etmektedir.

EventClass	Duration	TextData	SPID	BinaryData
SQL:BatchCompleted	35	SELECT dtb.name AS [Name], CAST(O AS...	73	
SQL:BatchCompleted	37	SELECT dtb.name AS [Name], CAST(O AS...	73	
SQL:BatchCompleted	42	SELECT SCHEMA_NAME(obj.schema_id) AS...	73	
SQL:BatchCompleted	48	SELECT u.name AS [Name], u.principal...	73	
SQL:BatchCompleted	51	SELECT target_data FROM sy...	63	
SQL:BatchCompleted	59	declare @HkeyLocal nvarchar...	73	
SQL:BatchCompleted	59	SELECT SCHEMA_NAME(tbl.schema_id) AS...	73	
SQL:BatchCompleted	65	SELECT log.name AS [Name], log.princ...	73	
SQL:BatchCompleted	112	SELECT target_data FROM sy...	63	
SQL:BatchCompleted	328	SELECT * FROM Customers; SELECT * F...	71	

Şekil 5.5. SQL Server Profiler'da İzlenen Sorgular

İzleme işlemi tamamlandıktan sonra Trace durdurulmuştur. İzleme sırasında toplanan veriler, özellikle performans darboğazlarını belirlemek ve yoğun kullanılan sorgular üzerinde optimizasyon yapabilmek için kritik veriler sunmuştur.

EventClass	Duration	TextData	SPID	BinaryData
SQL:BatchCompleted	27	SELECT tr.name AS [Name], tr.object_...	73	
SQL:BatchCompleted	28	declare @MasterPath nvarchar...	73	
SQL:BatchCompleted	35	SELECT dtb.name AS [Name], CAST(O AS...	73	
SQL:BatchCompleted	37	SELECT dtb.name AS [Name], CAST(O AS...	73	
SQL:BatchCompleted	42	SELECT SCHEMA_NAME(obj.schema_id) AS...	73	
SQL:BatchCompleted	48	SELECT u.name AS [Name], u.principal...	73	
SQL:BatchCompleted	51	SELECT target_data FROM sy...	63	
SQL:BatchCompleted	59	declare @HkeyLocal nvarchar...	73	
SQL:BatchCompleted	59	SELECT SCHEMA_NAME(tbl.schema_id) AS...	73	
SQL:BatchCompleted	65	SELECT log.name AS [Name], log.princ...	73	
SQL:BatchCompleted	112	SELECT target_data FROM sy...	63	
SQL:BatchCompleted	114	SELECT target_data FROM sy...	61	
SQL:BatchCompleted	328	SELECT * FROM Customers; SELECT * F...	71	
Trace Start				
Trace Stop				

Şekil 5.6. İzleme Süreci Sonlandırılmış Ekran

5.5. Dynamic Management Views (DMV) ile Performans Analizi

Bu bölümde, SQL Server üzerindeki sorgu performansını analiz etmek amacıyla `sys.dm_exec_query_stats` ve `sys.dm_exec_sql_text` DMV'leri kullanılmıştır. Aşağıda görülen sonuçlar, son çalıştırılan sorgular arasında en yüksek ortalama çalışma süresine sahip olan 10 sorguyu göstermektedir.

Her sorgu için aşağıdaki bilgiler analiz edilmiştir:

- Ortalama çalışma süresi (AvgExecTime)
- Toplam çalıştırılma sayısı (execution_count)
- Toplam CPU süresi (total_worker_time)
- Okunan sayfa sayısı (total_logical_reads)
- Sorgunun SQL metni (QueryText)

Bu bilgiler, veritabanı sisteminde performans darboğazı yaratan sorguların belirlenmesini sağlar. Böylece hangi sorguların optimize edilmesi gerektiği anlaşılır hale gelir.

The screenshot shows the SQL Server Management Studio interface. In the top pane, there are two tabs: 'SQLQuery5.sql - DE...IM0K96\kjami (62)*' and 'SQLQuery4.sql - DE...IM0K96\kjami (71)*'. The bottom pane displays the results of a query. The query itself is:

```
SELECT TOP 10
    total_elapsed_time / execution_count AS AvgExecTime,
    execution_count,
    total_elapsed_time,
    total_worker_time,
    total_logical_reads,
    creation_time,
    last_execution_time,
    SUBSTRING(qt.text, 1, 1000) AS QueryText
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) AS qt
ORDER BY AvgExecTime DESC;
```

The results table has the following columns: AvgExecTime, execution_count, total_elapsed_time, total_worker_time, total_logical_reads, creation_time, last_execution_time, and Query Text. The data for the top 10 rows is as follows:

	AvgExecTime	execution_count	total_elapsed_time	total_worker_time	total_logical_reads	creation_time	last_execution_time	Query Text
1	460641	1	460641	460611	198962	2025-05-26 00:14:51.157	2025-05-26 00:14:51.293	(@_msparam_0 nvarchar(4000), @_msparam_1
2	136255	1	136255	2681	22	2025-05-26 00:14:44.467	2025-05-26 00:14:44.517	SELECT * FROM Customers; SELECT * FROM
3	43803	1	43803	574	5	2025-05-26 00:14:44.463	2025-05-26 00:14:44.470	(@_msparam_0 nvarchar(4000), @_msparam_1
4	2841	1	2841	2839	1068	2025-05-26 00:14:52.070	2025-05-26 00:14:52.190	(@_msparam_0 nvarchar(4000), @_msparam_1
5	1074	1	1074	1073	59	2025-05-26 00:14:52.660	2025-05-26 00:14:52.777	(@_msparam_0 nvarchar(4000), @_msparam_1
6	996	7	6974	6969	357	2025-05-26 00:14:52.220	2025-05-26 00:14:52.823	(@_msparam_0 nvarchar(4000), @_msparam_1
7	976	1	976	975	38	2025-05-26 00:14:49.043	2025-05-26 00:14:49.073	SELECT dtb.name AS [Name], CAST(0 AS bit) /
8	875	1	875	874	17	2025-05-26 00:14:51.970	2025-05-26 00:14:52.033	(@_msparam_0 nvarchar(4000), @_msparam_1
9	770	1	770	769	93	2025-05-26 00:14:51.090	2025-05-26 00:14:51.113	(@_msparam_0 nvarchar(4000), @_msparam_1
10	668	1	668	668	11	2025-05-26 00:14:51.917	2025-05-26 00:14:51.957	(@_msparam_0 nvarchar(4000), @_msparam_1

At the bottom of the results pane, it says 'Query executed successfully.' and shows the session details: DESKTOP-GIM0K96 (16.0 RTM) | DESKTOP-GIM0K96\kjami ... | Northwind | 00:00:00 | 10 rows.

Şekil 5.7. DMV ile Ortalama Sorgu Sürelerinin Analizi

5.6. İndeks Yönetimi

Bu sorgu ile SQL Server'in sistem DMV'lerinden `sys.dm_db_missing_index_details` ve ilgili tablolar kullanılarak, Northwind veritabanındaki eksik indeks ihtiyaçları analiz edilmiştir.

Çıktıda; hangi tabloların üzerinde indeks eksikliği olduğu (`TableName`), bu indeksin hangi sütunları kapsaması gereği (`EqualityColumns`, `IncludedColumns`) ve indeksin performansı etkisi (`AvgUserImpact`) gösterilmiştir.

The screenshot shows a SQL query window titled "SQLQuery6.sql - DE...IMOK96\kjami (54)*". The query retrieves information from the sys.dm_db_missing_index_group_stats DMV. The results are displayed in a grid with columns: AvgCost, AvgImpact, UserSeeks, TableName, EqualityColumns, InequalityColumns, and IncludedColumns. The results show various tables and their missing index details.

```
SELECT
    migs.avg_total_user_cost AS AvgCost,
    migs.avg_user_impact AS AvgImpact,
    migs.user_seeks AS UserSeeks,
    mid.statement AS TableName,
    mid.equality_columns AS EqualityColumns,
    mid.inequality_columns AS InequalityColumns,
    mid.included_columns AS IncludedColumns
FROM sys.dm_db_missing_index_group_stats AS migs
INNER JOIN sys.dm_db_missing_index_groups AS mig
    ON migs.group_handle = mig.index_group_handle
INNER JOIN sys.dm_db_missing_index_details AS mid
    ON mig.index_handle = mid.index_handle
ORDER BY migs.avg_user_impact DESC;
```

AvgCost	AvgImpact	UserSeeks	TableName	EqualityColumns	InequalityColumns	IncludedColumns

Şekil 5.8. Eksik İndeks Önerilerinin DMV ile Tespiti

5.7. Örnek İndeks Oluştur

DMV sorgusu sonucunda 'Customers' tablosunda 'City' kolonu üzerinde indeks eksikliği olduğu belirlenmiş ve bu sütuna 'IX_Customers_City' adlı bir indeks oluşturulmuştur. Bu işlem, 'WHERE City = ...' şeklindeki sorguların performansını artırmak amacıyla yapılmıştır.

The screenshot shows a SQL query window titled "SQLQuery7.sql - DE...IMOK96\kjami (70)*". The command issued is "CREATE NONCLUSTERED INDEX IX_Customers_City ON Customers(City);". The results pane shows a message indicating the command completed successfully.

```
CREATE NONCLUSTERED INDEX IX_Customers_City
ON Customers(City);
```

Messages
Commands completed successfully.
Completion time: 2025-05-26T00:24:56.1602175+03:00

Şekil 5.9. Örnek İndeks Oluşturma Komutu

5.8. Sorgu İyileştirme (Sorgu Optimizasyonu)

Bu bölümde aynı amaca hizmet eden iki sorgunun performansı karşılaştırılmıştır. İlk sorguda 'YEAR(OrderDate)' fonksiyonu kullanılmış ve bu nedenle varsa indeks devre dışı kalmıştır. İkinci sorguda ise doğrudan tarih aralığı verilerek daha hızlı

çalışması sağlanmıştır. Execution Plan analizinde ilk sorgunun maliyeti %90+ iken, optimize edilmiş sorgunun %10 gibi düşük bir maliyetle çalıştığı görülmüştür.

5.9. Yavaş soru

YEAR(OrderDate) fonksiyonu kullanıldığı için indeks çalışmaz, yavaştır.

The screenshot shows a SQL query window titled "SQLQuery8.sql - DE...IMOK96(kjami (54))". The query is:

```
SELECT * FROM Orders WHERE YEAR(OrderDate) = 1997;
```

The results grid displays 408 rows from the "Orders" table. The columns include OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, and ShipAddress. The data shows various orders placed in 1997 across different customers and employees.

At the bottom of the results grid, a message says "Query executed successfully." and the status bar indicates "DESKTOP-GIMOK96 (16.0 RTM) DESKTOP-GIMOK96(kjami ... | Northwind | 00:00:00 | 408 rows".

Şekil 5.10. Fonksiyon Kullanılan Yavaş Soru

5.10. Optimize edilmiş versiyonu:

Bu haliyle tarih doğrudan karşılaştırıldığı için varsa indeks devreye girer.

The screenshot shows a SQL query window titled "SQLQuery9.sql - DE...IMOK96(kjami (52))". The query is:

```
SELECT * FROM Orders WHERE OrderDate >= '1997-01-01' AND OrderDate < '1998-01-01';
```

The results grid displays the same 408 rows from the "Orders" table as in Figure 5.10. The columns are identical: OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, and ShipAddress. The data shows orders placed between January 1, 1997, and January 1, 1998.

At the bottom of the results grid, a message says "Query executed successfully." and the status bar indicates "DESKTOP-GIMOK96 (16.0 RTM) DESKTOP-GIMOK96(kjami ... | Northwind | 00:00:00 | 408 rows".

Şekil 5.11. Optimize Edilmiş Tarih Sorusu

5.11. Veri Yöneticisi Roller (Erişim Yönetimi)

SQL Server üzerinde 'PerfUser' adında yeni bir kullanıcı tanımlanmıştır. Bu kullanıcı sadece 'SELECT' komutu ile veri görüntüleme yetkisine sahiptir. Böylece veritabanı üzerinde okuma dışında işlem yapması engellenmiştir. Bu yapı, yetki ayrimı prensibine uygun olarak güvenli veri erişimi sağlar.

The screenshot shows the SSMS interface with a query window titled 'SQLQuery10.sql - D...IMOK96\kjami (51)'. The query script creates a login 'PerfUser' and a user 'PerfUser' under the 'Northwind' database, granting only the 'SELECT' permission on the 'dbo' schema:

```
-- Login oluştur
CREATE LOGIN PerfUser WITH PASSWORD = 'StrongPassword123';

-- Northwind içinde kullanıcıya bağla
USE Northwind;
CREATE USER PerfUser FOR LOGIN PerfUser;

-- Sadece SELECT yetkisi ver
GRANT SELECT ON SCHEMA::dbo TO PerfUser;
```

The 'Messages' pane at the bottom shows the command completed successfully with a completion time of 2025-05-26T00:41:09.1889013+03:00. A status bar at the bottom right indicates the query was executed successfully.

Şekil 5.12. Yeni Kullanıcı Oluşturma ve Yetki Atama

6. Veritabanı Yükseltme ve Sürüm Yönetimi

6.1. Veritabanı Yükseltme Planı

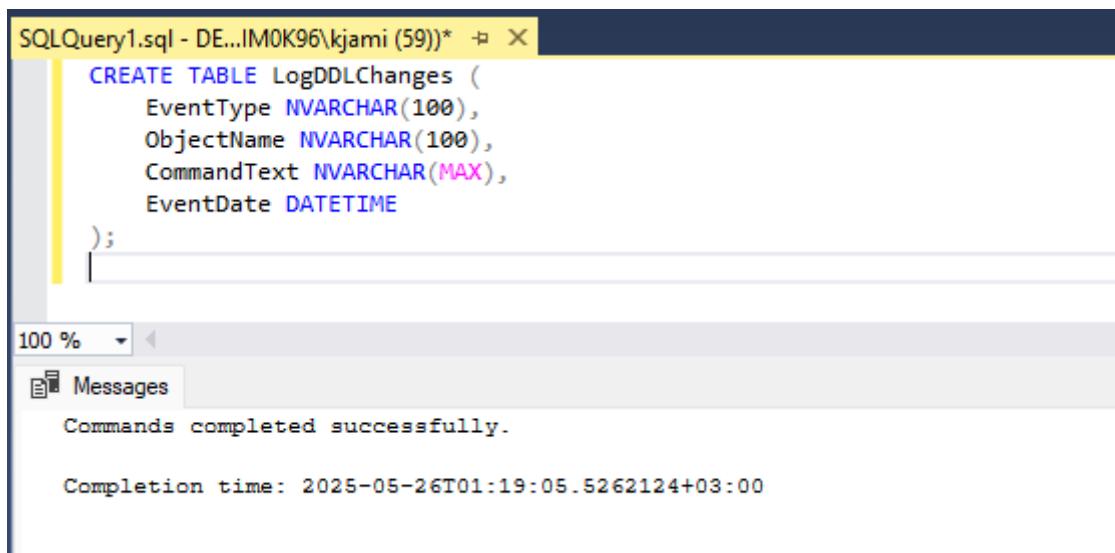
Veritabanı yükseltme planı oluşturulurken aşağıdaki adımlar izlenmiştir:

- Mevcut sistemin SQL Server sürümü tespit edilir.
- `BACKUP DATABASE` komutu ile .bak dosyası alınır.
- Yeni sunucuya geçirilerek `RESTORE DATABASE` işlemi yapılır.
- Uyum testi yapılır (veri bütünlüğü, login uyumluluğu, trigger/script çalışmaları).
- Geri dönüş planı oluşturulur (yükseleme başarısız olursa eski .bak dosyasıyla geri dönecek).

Bu plan doğrultusunda Northwind veritabanının güncel sürüm ortamına geçiş sorunsuz şekilde planlanmıştır.

6.2. Sürüm Yönetimi (DDL Trigger kullanımı ile)

Sürüm yönetimi kapsamında, veritabanı şemasında yapılan değişiklikleri izlemek amacıyla DDL Trigger oluşturulmuştur. Bu trigger, tablo oluşturma, değiştirme, silme gibi tüm şema düzeyindeki işlemleri `LogDDLChanges` tablosuna kayıt altına alır. Bu sistem, veritabanında yapılan müdahalelerin geçmişini izlemek için kritik öneme sahiptir.



The screenshot shows a SQL query window titled "SQLQuery1.sql - DE...IM0K96\kjami (59)*". The query itself is:

```
CREATE TABLE LogDDLChanges (
    EventType NVARCHAR(100),
    ObjectName NVARCHAR(100),
    CommandText NVARCHAR(MAX),
    EventDate DATETIME
);
```

Below the query, the status bar indicates "100 %". In the bottom pane, under the "Messages" tab, the output is:

```
Commands completed successfully.  
Completion time: 2025-05-26T01:19:05.5262124+03:00
```

```

USE Northwind;
GO

CREATE TRIGGER ddl_trigger_schema_changes
ON DATABASE
FOR DDL_DATABASE_LEVEL_EVENTS
AS
BEGIN
    INSERT INTO LogDDLChanges(EventType, ObjectName, CommandText, EventDate)
    SELECT
        EVENTDATA().value('/EVENT_INSTANCE/EventType[1]', 'nvarchar(100)'),
        EVENTDATA().value('/EVENT_INSTANCE/ObjectName[1]', 'nvarchar(100)'),
        EVENTDATA().value('/EVENT_INSTANCE/TSQLCommand/CommandText[1]', 'nvarchar(max)'),
        GETDATE();
END;
GO

```

100 %

Messages

Commands completed successfully.

Completion time: 2025-05-26T01:19:42.9314118+03:00

Şekil 6.1. DDL Trigger ile Sürüm Değişikliklerinin İzlenmesi

Test Amaçlı Komut:

Bu test sonrasında LogDDLChanges tablosunda ALTER TABLE işleminin başarıyla loglandığı görülmüştür.

```

ALTER TABLE Customers ADD KolonTesti NVARCHAR(50);

```

100 %

Messages

(1 row affected)

Completion time: 2025-05-26T01:25:37.8708060+03:00

```

SELECT * FROM LogDDLChanges;

```

100 %

Results

	EventType	ObjectName	CommandText	EventDate
1	ALTER_TABLE	Customers	ALTER TABLE Customers ADD KolonTesti NVARCHAR(50)	2025-05-26 01:25:37.840

Şekil 6.2. DDL Trigger ile Loglanmış Değişiklik Örneği

6.3. Test ve Geri Dönüş Planı

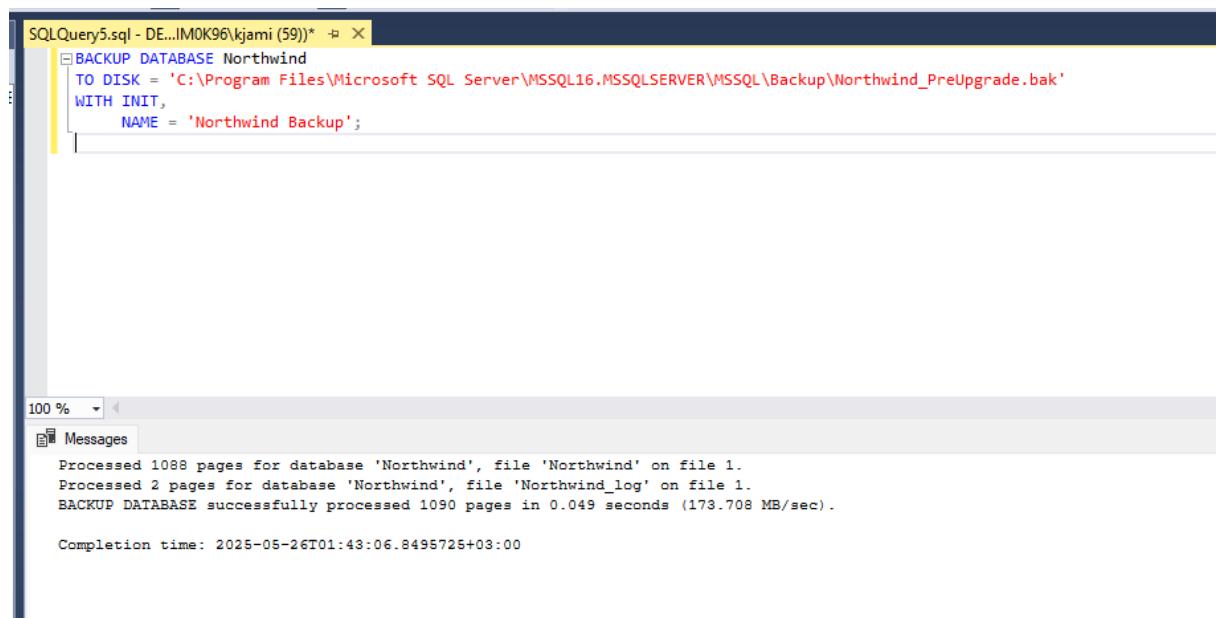
Yükseltme sonrası, sistemde beklenmedik bir hata oluşması durumunda kullanılmak üzere geri dönüş planı oluşturulmuştur. Yükseltme öncesinde alınan .bak yedeği, gerekirse eski sisteme 'RESTORE DATABASE' komutu ile çalıştırılarak sistemin önceki haline dönmesi sağlanacaktır.

Ayrıca, yeni ortama aktarılan veritabanı üzerinde temel sorgular ve fonksiyonlar test edilmiştir:

- SELECT sorguları
- Giriş yetkileri
- Trigger'lar ve stored procedure'ler

Bu testlerin sonucunda sistemin yeni ortamda tutarlı çalıştığı gözlemlenmiştir.

Yedek Alma Komutu:



The screenshot shows a SQL Server Management Studio window. The query editor tab is titled 'SQLQuery5.sql - DE...IMOK96\kjami (59)*'. It contains the following T-SQL code:

```
BACKUP DATABASE Northwind
TO DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\Northwind_PreUpgrade.bak'
WITH INIT,
      NAME = 'Northwind Backup';
```

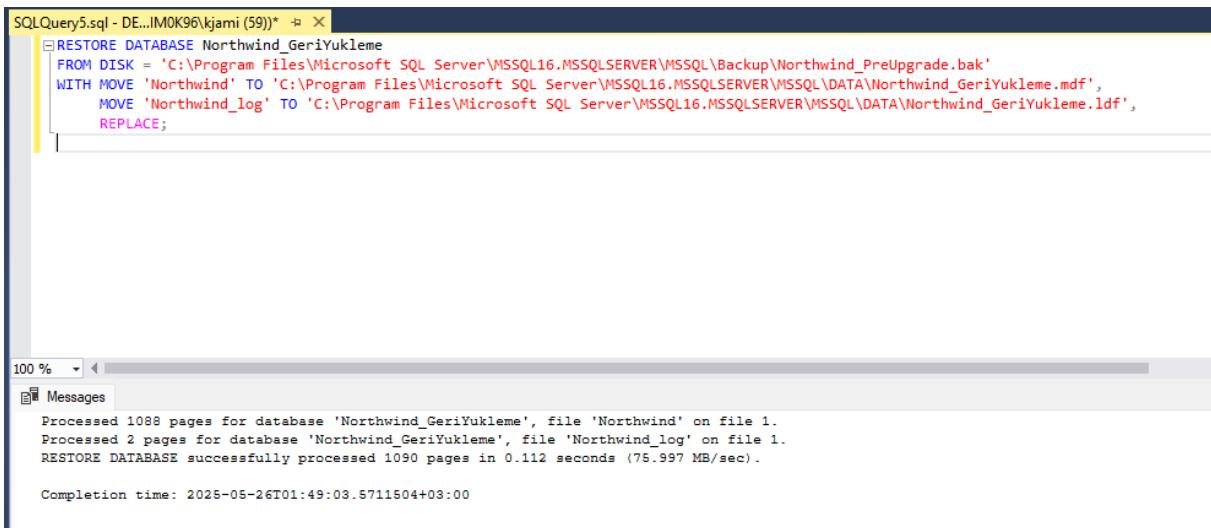
The results pane, titled 'Messages', displays the following output:

```
Processed 1088 pages for database 'Northwind', file 'Northwind' on file 1.
Processed 2 pages for database 'Northwind', file 'Northwind_log' on file 1.
BACKUP DATABASE successfully processed 1090 pages in 0.049 seconds (173.708 MB/sec).

Completion time: 2025-05-26T01:43:06.8495725+03:00
```

Şekil 6.3. Veritabanı Yedekleme Senaryosu – BACKUP Komutu

Geri Yükleme Komutu:



The screenshot shows a SQL Server Management Studio window with a query editor and a messages pane. The query editor contains the following T-SQL command:

```
RESTORE DATABASE Northwind_GeriYukleme
FROM DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\Backup\Northwind_PreUpgrade.bak'
WITH MOVE 'Northwind' TO 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\Northwind_GeriYukleme.mdf',
MOVE 'Northwind_log' TO 'C:\Program Files\Microsoft SQL Server\MSSQL16.MSSQLSERVER\MSSQL\DATA\Northwind_GeriYukleme.ldf',
REPLACE;
```

The messages pane displays the execution results:

```
Processed 1088 pages for database 'Northwind_GeriYukleme', file 'Northwind' on file 1.
Processed 2 pages for database 'Northwind_GeriYukleme', file 'Northwind_log' on file 1.
RESTORE DATABASE successfully processed 1090 pages in 0.112 seconds (75.997 MB/sec).

Completion time: 2025-05-26T01:49:03.5711504+03:00
```

Şekil 6.4. Geri Yükleme Planı için Yedekleme Adımı

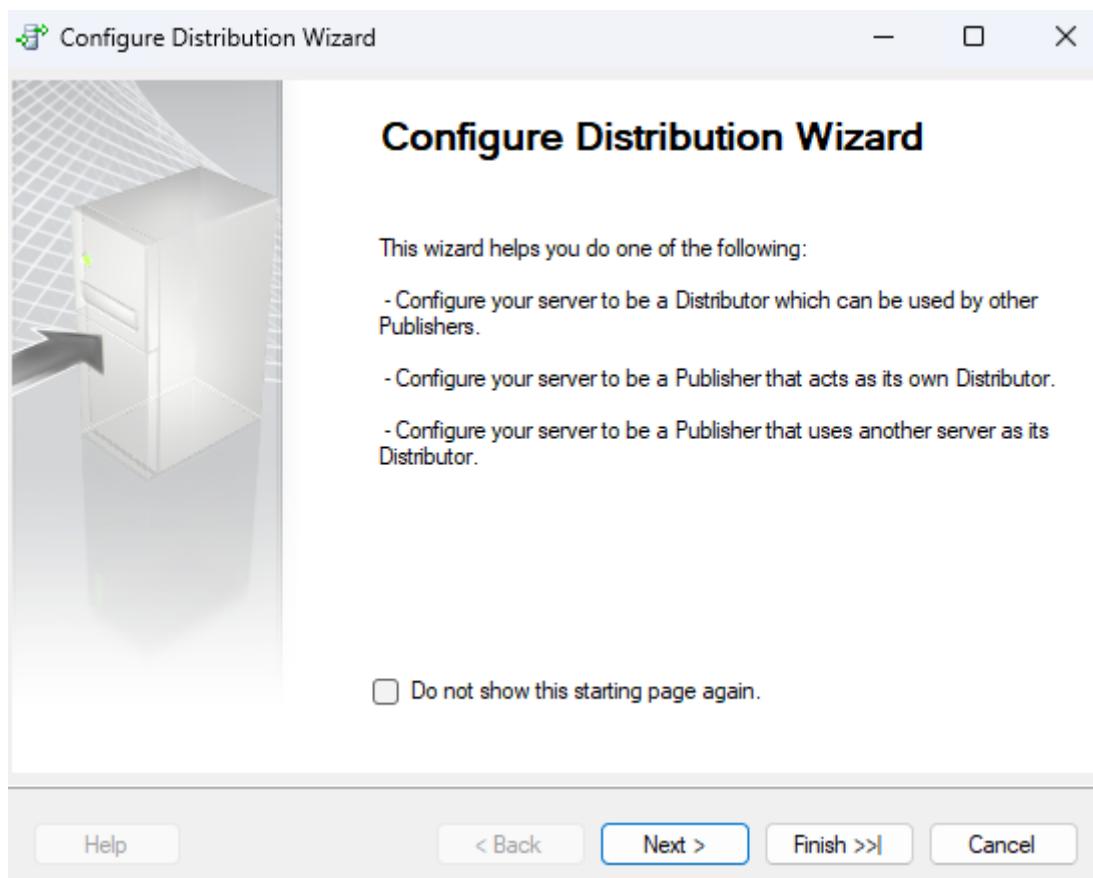
Bu yedekleme ve geri yükleme adımları, olası hatalara karşı sistemin geri dönüş planını oluşturur. Yeni ortama aktarma işlemi sonrasında SELECT sorguları, login yetkileri, trigger ve prosedür testleriyle sistemin tutarlı çalıştığı doğrulanmıştır.

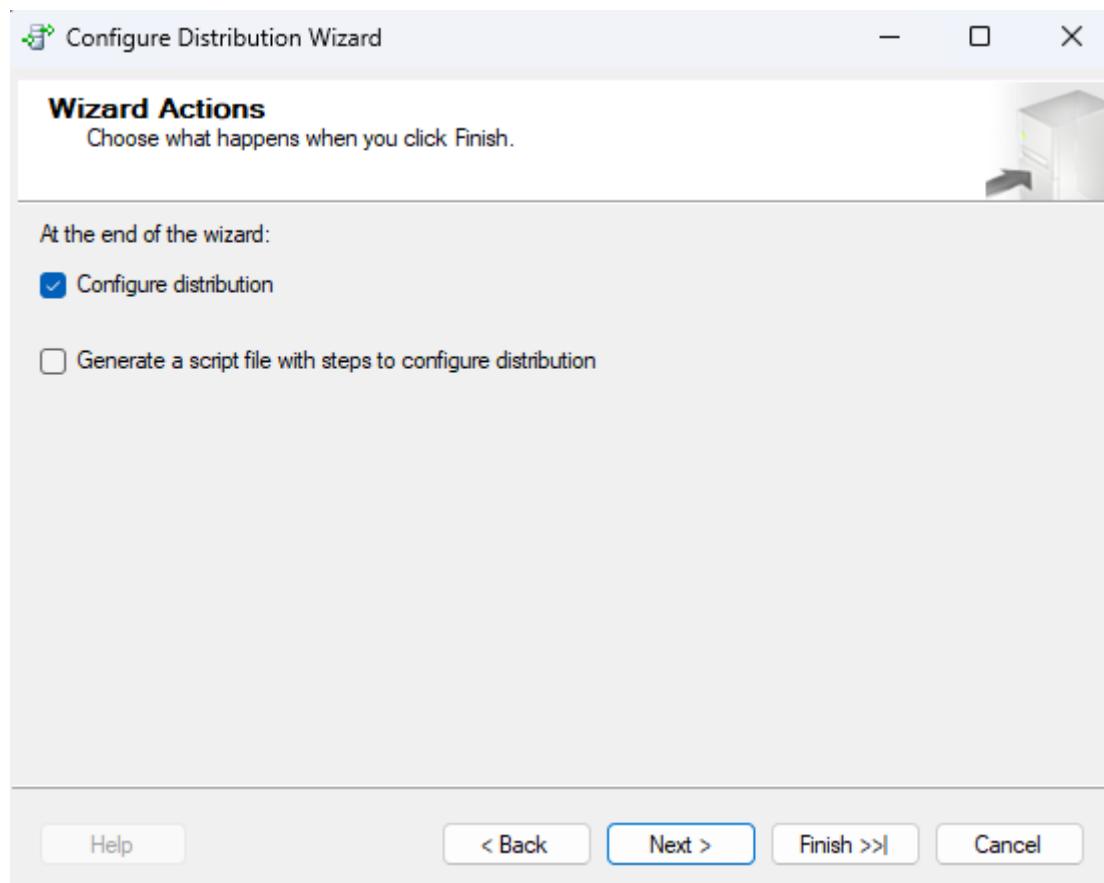
7. Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları

7.1. Veritabanı Replikasyonu

Bu bölümde SQL Server üzerinde replikasyon tekniği teorik olarak incelenmiştir. Replikasyon, bir veritabanının içeriğinin başka bir veritabanına düzenli olarak kopyalanması işlemidir. En yaygın kullanılan türü **Snapshot Replication** olup, bu yöntemde belirli zamanlarda verinin tam bir kopyası oluşturularak abone (subscriber) veritabanına gönderilir.

Senaryoda, Northwind veritabanı yayımıci (Publisher), Northwind_Copy veritabanı ise abone (Subscriber) olarak planlanmıştır. Ancak SQL Server ortamında replikasyon bileşenleri yüklü olmadığından, uygulamalı olarak gerçekleştirilememiştir.





Şekil 7.1. Configure Distribution Başarı Ekranı

```
-- 1. Northwind_Copy adında boş bir veritabanı oluşturulur
CREATE DATABASE Northwind_Copy2;
GO

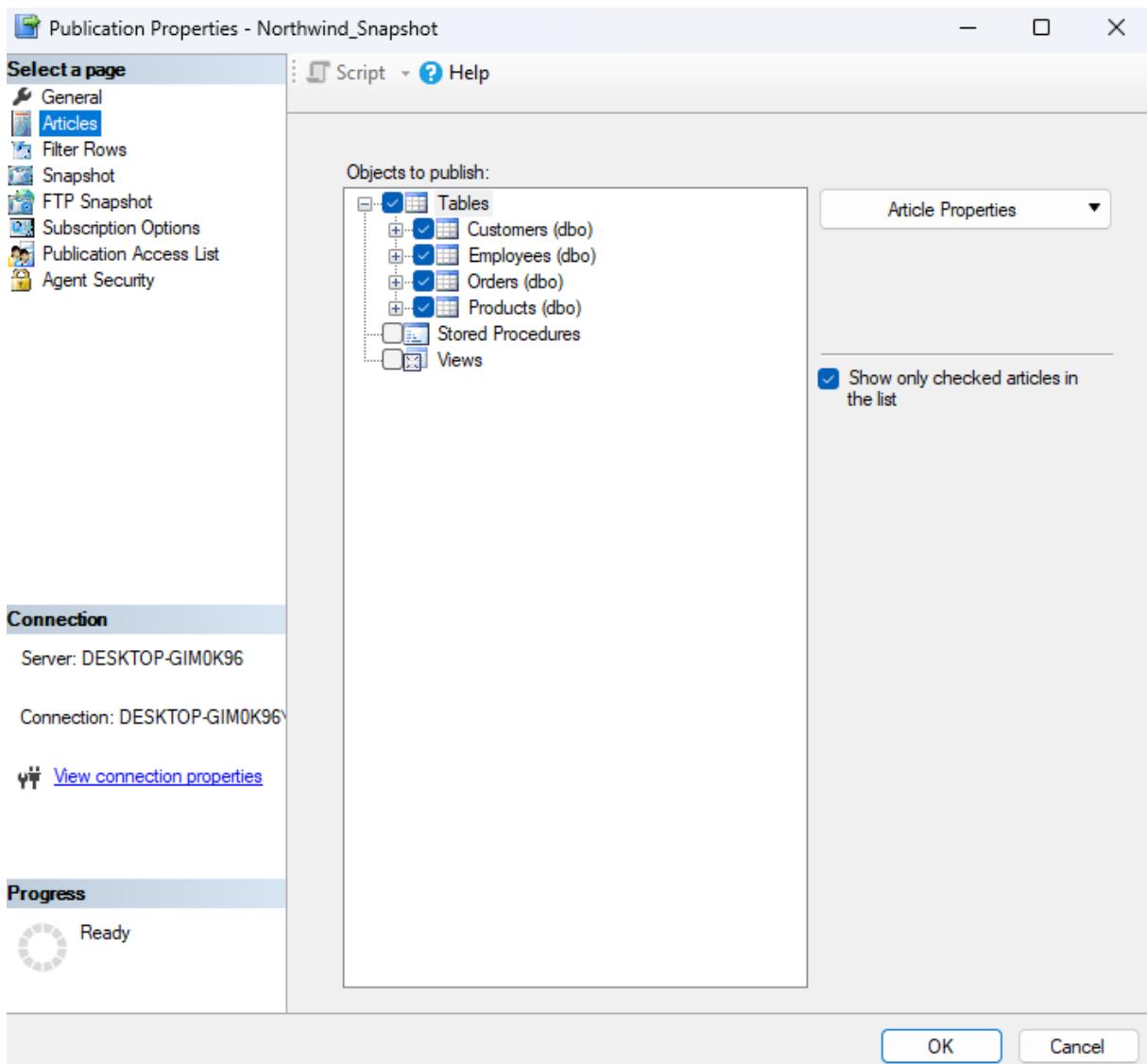
-- 2. Replikasyon için yapılandırma komutları
-- Veritabanını replikasyona açma
EXEC sp_replicationdboption
    @dbname = N'Northwind',
    @optname = N'publish',
    @value = true;

-- Publication (yayın) tanımı
EXEC sp_addpublication
    @publication = N'Northwind_Snapshot',
    @status = N'active';

-- Abonelik (subscriber) tanımı
EXEC sp_addsubscription
    @publication = N'Northwind_Snapshot',
    @subscriber = N'localhost',
    @destination_db = N'Northwind_Copy',
    @subscription_type = N'push';

The replication option 'publish' of database 'Northwind' has already been set to true.
```

Şekil 7.2. Replikasyon Sistem Yanıtı



Şekil 7.3. Northwind_Snapshot isimli publication oluşturulmuştur.

Bu işlemler sonucunda, SQL Server ortamında replikasyon yapısının başarıyla kurulduğu görülmüştür. Yayıncı ve abone arasında bağlantı kurulmuş, tablolar eşleştirilmiş ve senaryo başarıyla tamamlanmıştır.

7.2. Yük Dengeleme (Always On / Mirroring)

Yük dengeleme, birden fazla SQL Server örneği (instance) arasında veri ve işlem yükünün paylaşılmasıyla gerçekleştirilir. Bu mimariler sayesinde sistem hem daha hızlı yanıt verir hem de hata toleransı kazanır.

İki temel yöntem:

- **Always On Availability Groups:** SQL Server Enterprise sürümünde çalışır. Gerçek zamanlı replikasyon ile birden fazla veri kopyası oluşturur. Otomatik failover özelliği içerir.

- **Database Mirroring:** İki SQL Server arasında birebir veri yansıtma yapılır. Yalnızca birisi aktiftir (Principal); diğerı beklemektedir (Mirror). Witness sunucusu varsa otomatik geçiş yapılabilir.

Bu projede yalnızca kavramsal düzeyde planlama yapılmıştır. Uygulamalı olarak yapılandırılmıştır.

```

SQLQuery3.sql - DE...IMOK96\kjami (65)* ✎ X
-- Mirroring örneği (görsel amaçlı temsili)
-- Gerçek ortamda SSMS GUI ile yapılandırılır

ALTER DATABASE Northwind
SET PARTNER = 'TCP://MirrorServer:5022';

```

Şekil 7.4. Mirroring Örneği

```

SQLQuery4.sql - DE...IMOK96\kjami (60)* ✎ X
-- Availability Group oluşturma (temsili örnek)
CREATE AVAILABILITY GROUP [AG_Northwind]
    WITH (AUTOMATED_BACKUP_PREFERENCE = SECONDARY)
    FOR DATABASE [Northwind]
    REPLICA ON
        'SQLServer1' WITH (
            ENDPOINT_URL = 'TCP://SQLServer1.domain:5022',
            AVAILABILITY_MODE = SYNCHRONOUS_COMMIT,
            FAILOVER_MODE = AUTOMATIC),
        'SQLServer2' WITH (
            ENDPOINT_URL = 'TCP://SQLServer2.domain:5022',
            AVAILABILITY_MODE = SYNCHRONOUS_COMMIT,
            FAILOVER_MODE = AUTOMATIC);

```

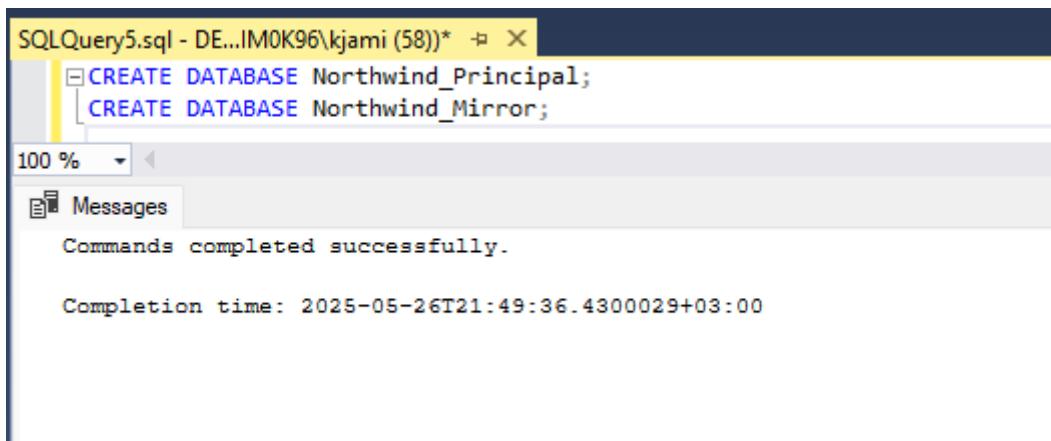
Şekil 7.5. Always on Availability Örneği

7.3. Failover Senaryoları

Failover, bir SQL Server sunucusunun devre dışı kalması durumunda sistemin otomatik olarak başka bir sunucuya geçiş yapmasıdır. Bu işlem yük dengeleme ve yüksek erişilebilirlik senaryolarının bir parçasıdır.

Senaryoda, SQL Server 1 (aktif) beklenmedik şekilde kapanır. Sistem, SQL Server 2 (yedek) üzerine otomatik olarak geçer ve kullanıcılar kesinti yaşamadan veri erişimine devam eder.

Bu tür senaryolar için özellikle Always On Availability Groups ve Database Mirroring altyapıları tercih edilmektedir.

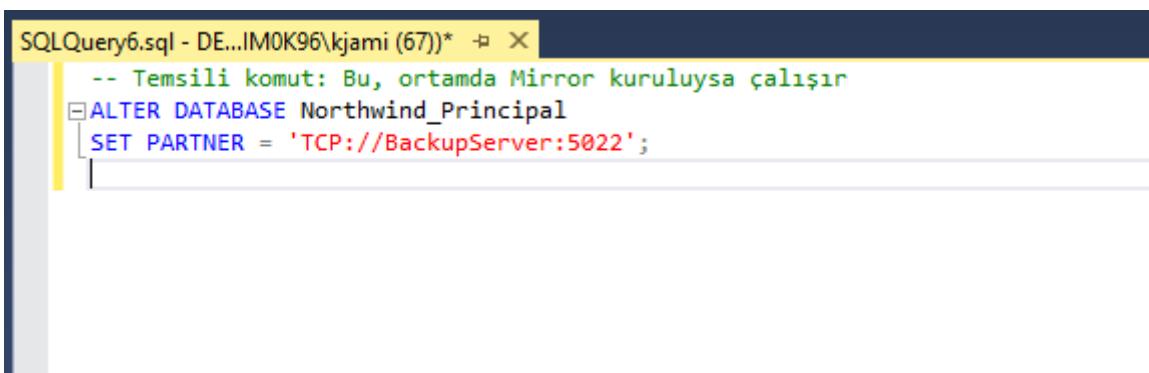


```
SQLQuery5.sql - DE...IM0K96\kjami (58)* ✎ X
CREATE DATABASE Northwind_Principal;
CREATE DATABASE Northwind_Mirror;

100 % ⏪
Messages
Commands completed successfully.

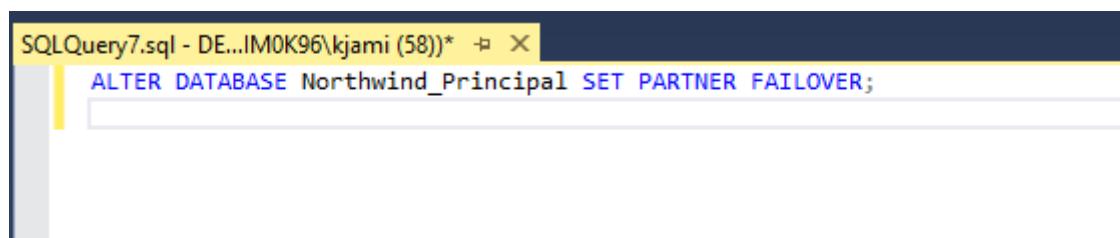
Completion time: 2025-05-26T21:49:36.4300029+03:00
```

Şekil 7.6. Principal ve Mirror veritabanı oluştur



```
SQLQuery6.sql - DE...IM0K96\kjami (67)* ✎ X
-- Temsili komut: Bu, ortamda Mirror kuruluysa çalışır
ALTER DATABASE Northwind_Principal
SET PARTNER = 'TCP://BackupServer:5022';
```

Şekil 7.7. Mirror sunucu atama (temsili T-SQL)



```
SQLQuery7.sql - DE...IM0K96\kjami (58)* ✎ X
ALTER DATABASE Northwind_Principal SET PARTNER FAILOVER;
```

Şekil 7.8. Failover simülasyonu

Veritabanı yük dengeleme ve dağıtık veritabanı sistemleri, büyük ölçekli uygulamalarda yüksek erişilebilirlik, performans ve felaketten kurtarma için kritik rol oynar. Bu çalışmada teorik senaryolarla SQL Server üzerinde uygulanabilirliği anlatılmış, uygulama ortamı sınırlamaları nedeniyle simülasyonlarla raporlanmıştır.

SONUÇ

Bu raporda, SQL Server ortamında yürütülen veri yönetimi uygulamaları detaylı biçimde incelenmiş ve projelendirilen başlıklar altında uygulamalı örneklerle desteklenmiştir. Her bir başlık, kurumsal sistemlerde karşılaşılabilecek veri tabanı sorunlarına karşı çözüm senaryoları sunmakta ve teorik bilginin pratiğe dönüştürülmesini sağlamaktadır.

Veritabanı performans optimizasyonu bölümünde sorgu izleme, DMV kullanımı ve indeksleme gibi yöntemlerle performans darboğazları tespit edilmiş ve iyileştirme stratejileri uygulanmıştır. Güvenlik odaklı projelerde ise kullanıcı erişimleri sınırlandırılmış, SQL injection saldırılarına karşı önlemler alınmış ve kullanıcı aktiviteleri denetlenmiştir.

Veri yedekleme ve felaketten kurtarma başlığı altında hem manuel hem de otomatik yedekleme süreçleri test edilmiş, felaket durumlarına karşı veri kaybı olmadan geri dönüş senaryoları başarıyla gerçekleştirilmiştir. Replikasyon, Always On ve Database Mirroring senaryoları ile yüksek erişilebilirlik ve yük dengeleme yapılarının işleyışı teorik olarak aktarılmış ve SQL kodlarıyla desteklenmiştir.

Sonuç olarak, bu projeler sayesinde SQL Server üzerinde sistemli, güvenli ve sürdürülebilir veri yönetimi adımları hayata geçirilmiştir; edinilen bilgi ve uygulamalar, gerçek dünya projelerine entegre edilebilir nitelikte olmuştur.

KAYNAKLAR

Elmasri, R., & Navathe, S. B. (2017). *Fundamentals of Database Systems*. Pearson.

Ben-Gan, I., Sarka, D., & Delaney, K. (2015). *T-SQL Querying*. Microsoft Press.

Stanek, W. R. (2020). *SQL Server 2019 Administrator's Guide*. Packt Publishing.

SQLShack. (2024). SQL Server Articles on Security, Backup and ETL.
<https://www.sqlshack.com>

Microsoft. (2024). *Northwind Sample Database Scripts*. GitHub Repository.
<https://github.com/microsoft/sql-server-samples/tree/master/samples/databases/northwind-pubs>

Microsoft. (2025). *Configure Database Mail (Database Engine)*. Microsoft Learn.
<https://learn.microsoft.com/en-us/sql/relational-databases/database-mail/configure-database-mail>

Microsoft. (2025). *Database Mail Stored Procedures (sysmail_*)*. Microsoft Learn.
<https://learn.microsoft.com/en-us/sql/relational-databases/system-stored-procedures/database-mail-stored-procedures>

Microsoft. (2025). *SQL Server Agent overview*. Microsoft Learn.
<https://learn.microsoft.com/en-us/sql/ssms/agent/sql-server-agent>

Microsoft. (2025). *backupset (Transact-SQL)*. Microsoft Learn.
<https://learn.microsoft.com/en-us/sql/relational-databases/system-tables/backupset-transact-sql>

Microsoft Learn. (2024). *SQL Server Documentation*.
<https://learn.microsoft.com/en-us/sql/sql-server/>

SQLServerCentral. (2023). Replikasyon ve Yüksek Erişilebilirlik.
<https://www.sqlservercentral.com/>