

ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



BLM4522 – Ağ Tabanlı Paralel Dağıtım Sistemleri Final Proje Raporu

Ali Furkan Şahin - 20290289

Özge Çelik - 21290514

<https://github.com/FurkanSahinnn/BLM4522-Ag-tabanli-paralel-dagitim-sistemleri.git>

<https://drive.google.com/file/d/1fVMPAiNJvxPm0nFl8wZB9juh6z6L-lpx/view>

İçindekiler:

1. Veritabanı Performans Optimizasyonu ve İzleme

- a. Veritabanı İzleme
- b. İndeks Yönetimi
- c. Sorgu İyileştirme
- d. Veri Yöneticisi Rolleri

2. Veritabanı Yedekleme ve Felaketten Kurtarma Planı

- a. Tam, Artık, Fark Yedeklemeleri
- b. Zamanlayıcılarla Yedekleme
- c. Felaketten Kurtarma Senaryoları
- d. Test Yedekleme Senaryoları

3. Veritabanı Güvenliği ve Erişim Kontrolü

- a. Erişim Yönetimi
- b. SQL Server Authentication ve Windows Authentication Kullanımı
- c. Veri Şifreleme
- d. Audit Logları

4. Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları

- a. Veritabanı Replikasyonu
- b. Yük Dengeleme
- c. Failover Senaryoları

5. Veri Temizleme ve ETL Süreçleri Tasarımı

- a. Veri Temizleme
- b. Veri Dönüştürme
- c. Veri Yükleme
- d. Veri Kalitesi Raporları

6. Veritabanı Yükseltme ve Sürüm Yönetimi

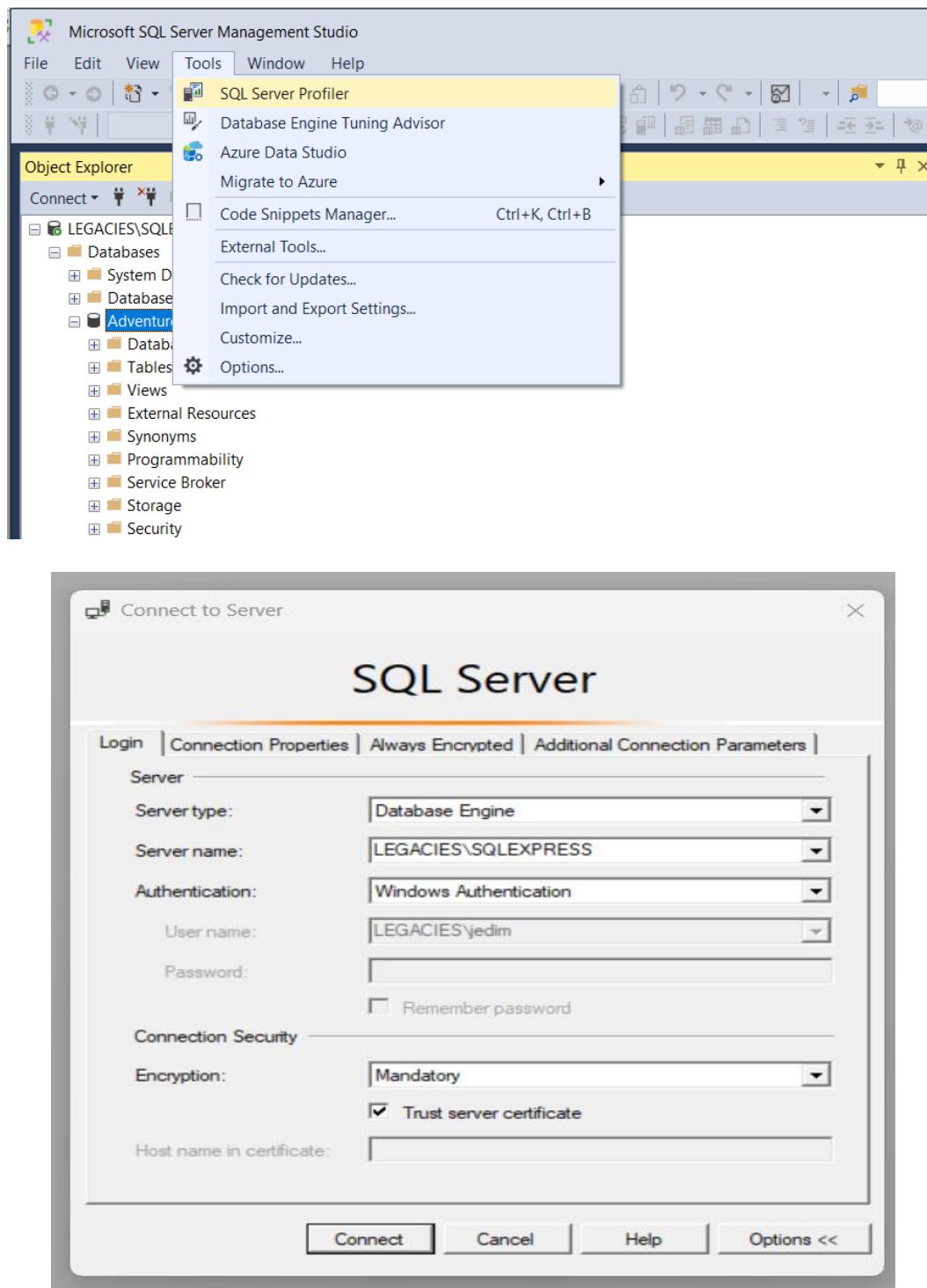
- a. Veritabanı Yükseltme Planı
- b. Sürüm Yönetimi
- c. Test ve Geri Dönüş Planı

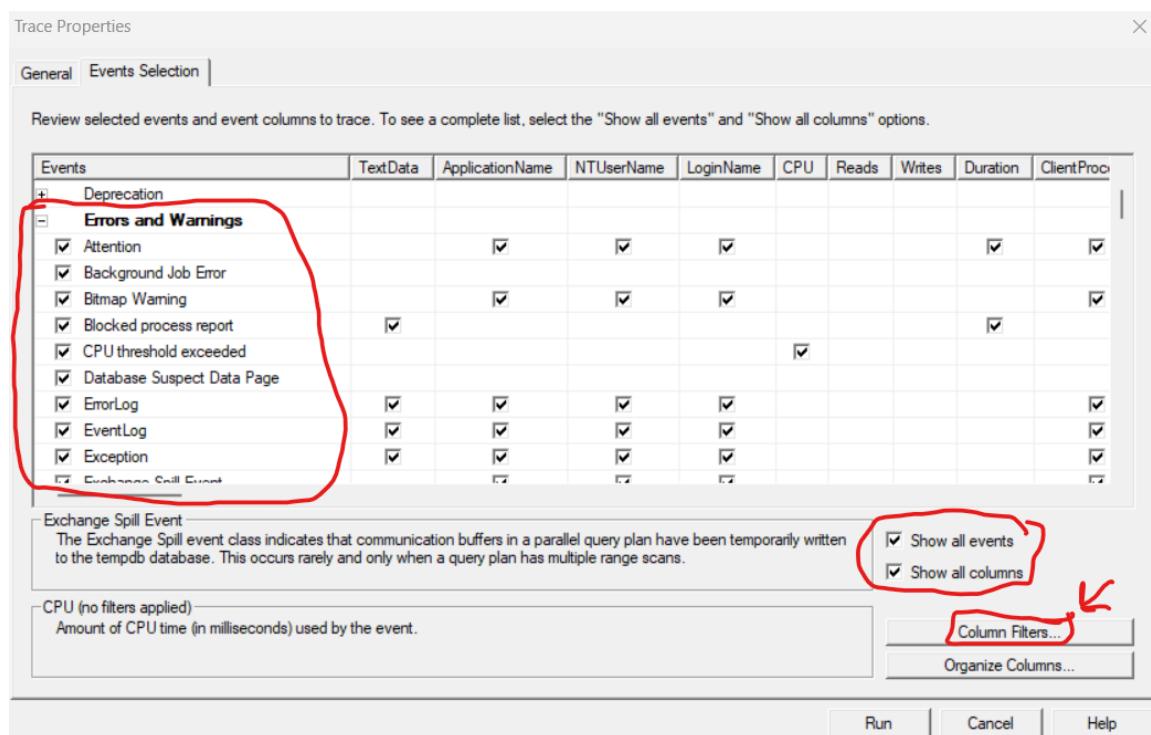
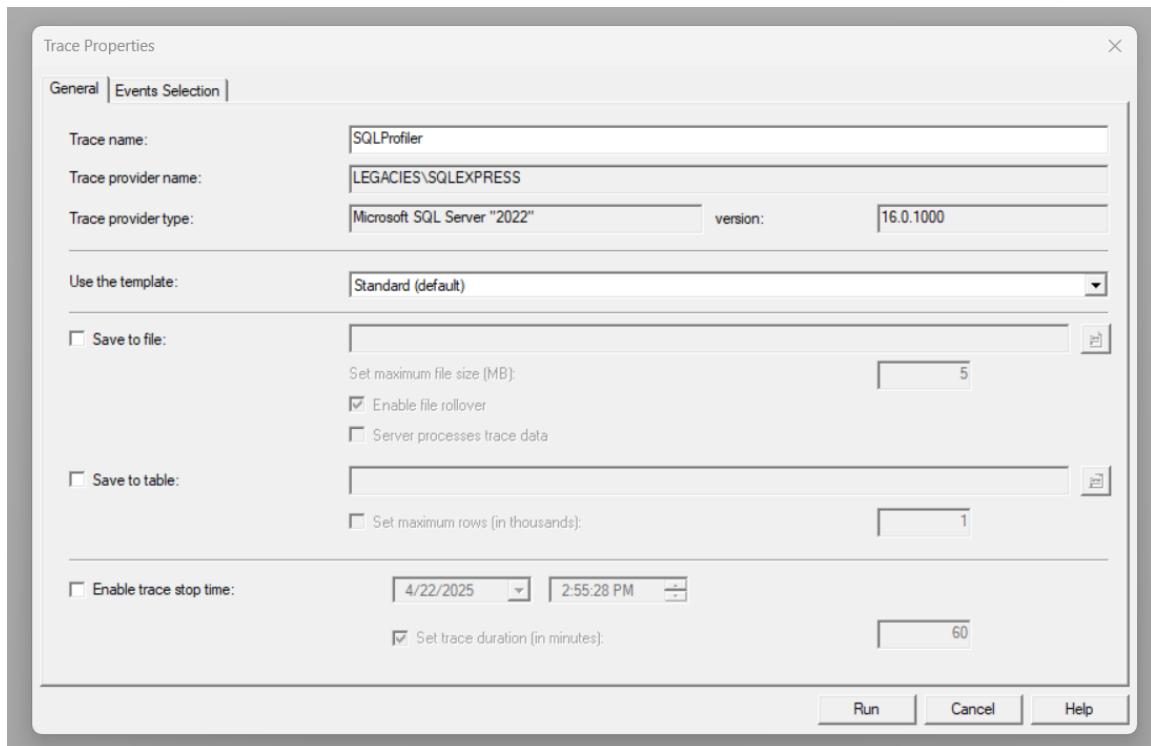
7. Veritabanı Yedekleme ve Otomasyon Çalışması

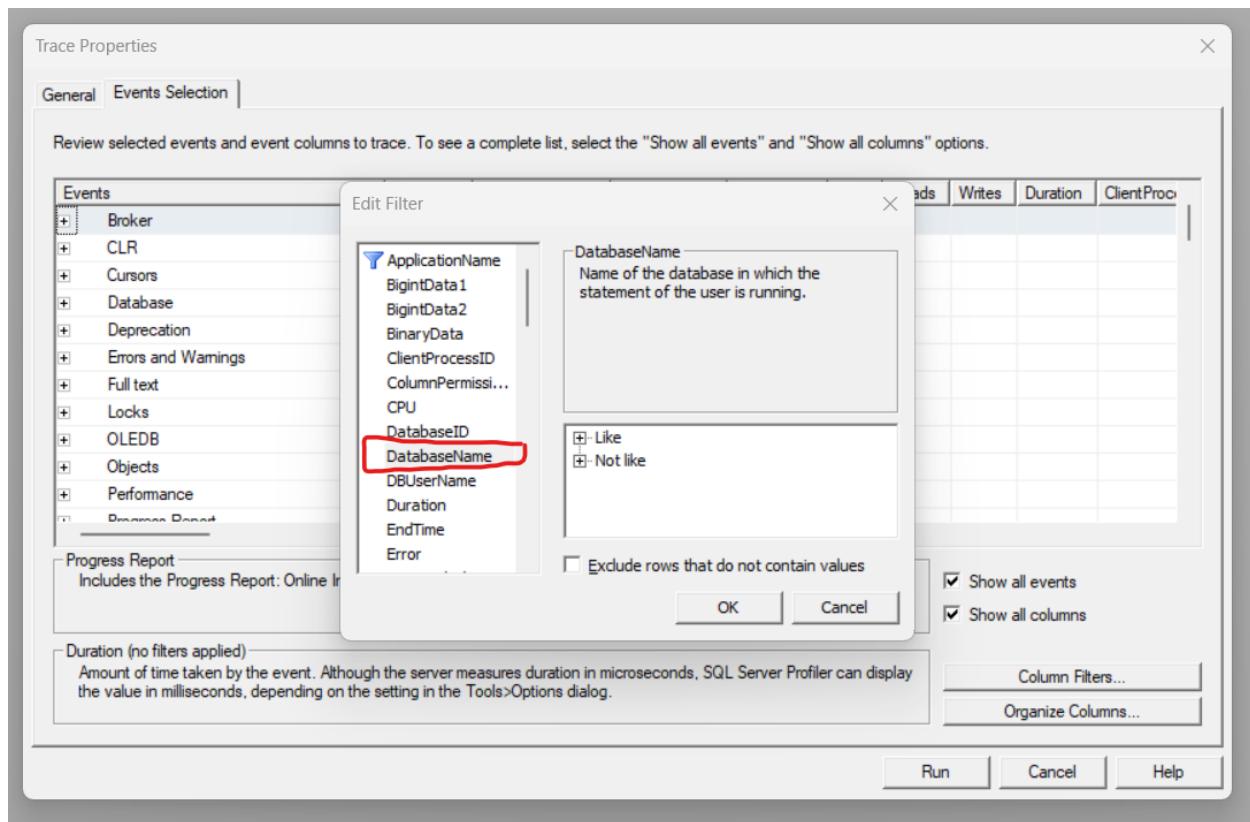
- a. Yedekleme Süreçlerini Otomatikleştirme
- b. Yedekleme Raporları Oluşturma
- c. Otomatik Yedekleme Uyarıları

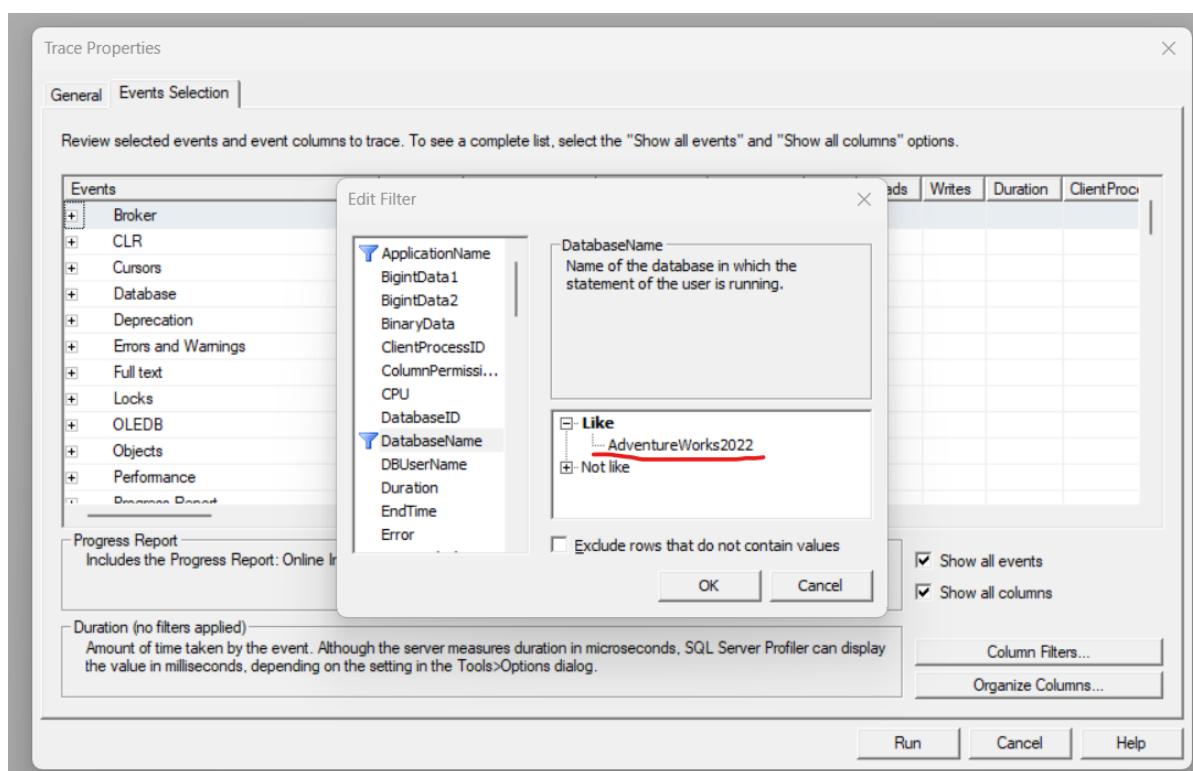
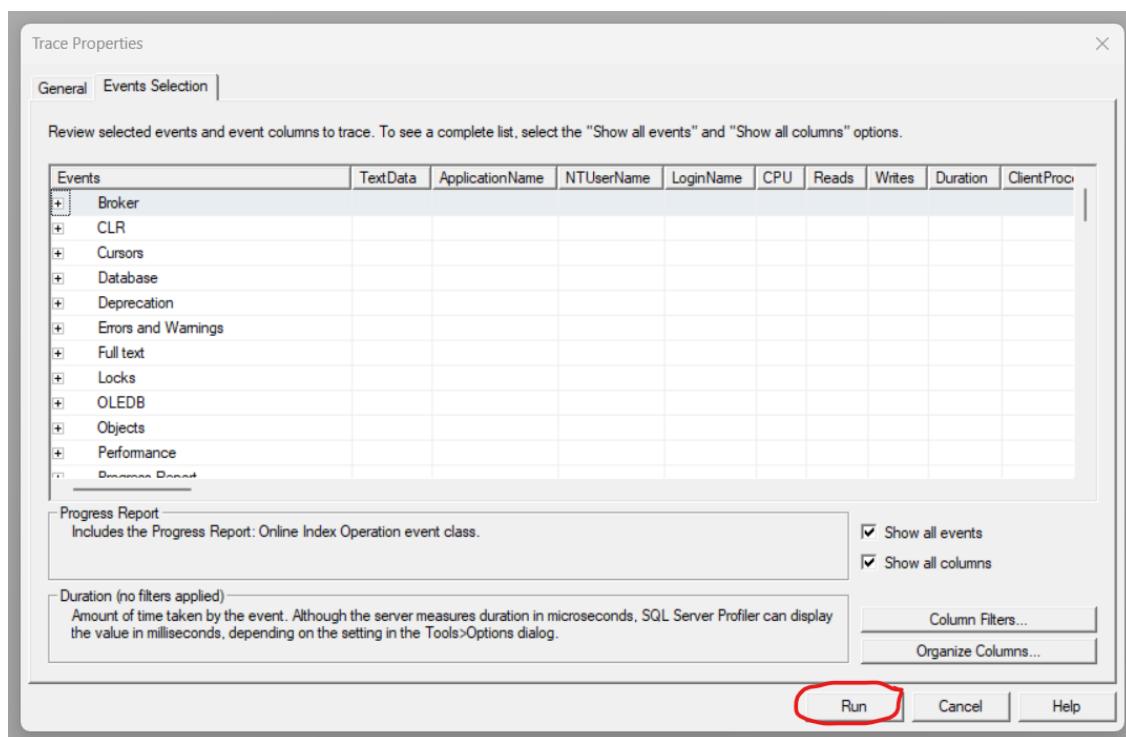
1. Veritabanı Performans Optimizasyonu ve İzleme

a. Veritabanı İzleme:









SQLQuery1.sql - LE...GACIES\jedim (73)*

```
SELECT * FROM HumanResources.Department
```

100 %

Results Messages

	DepartmentID	Name	GroupName	ModifiedDate
1	1	Engineering	Research and Development	2008-04-30 00:00:00.000
2	2	Tool Design	Research and Development	2008-04-30 00:00:00.000
3	3	Sales	Sales and Marketing	2008-04-30 00:00:00.000
4	4	Marketing	Sales and Marketing	2008-04-30 00:00:00.000
5	5	Purchasing	Inventory Management	2008-04-30 00:00:00.000
6	6	Research and Development	Research and Development	2008-04-30 00:00:00.000
7	7	Production	Manufacturing	2008-04-30 00:00:00.000
8	8	Production Control	Manufacturing	2008-04-30 00:00:00.000
9	9	Human Resources	Executive General and Administration	2008-04-30 00:00:00.000
10	10	Finance	Executive General and Administration	2008-04-30 00:00:00.000
11	11	Information Services	Executive General and Administration	2008-04-30 00:00:00.000
12	12	Document Control	Quality Assurance	2008-04-30 00:00:00.000
13	13	Quality Assurance	Quality Assurance	2008-04-30 00:00:00.000
14	14	Facilities and Maintenance	Executive General and Administration	2008-04-30 00:00:00.000
15	15	Shipping and Receiving	Inventory Management	2008-04-30 00:00:00.000
16	16	Executive	Executive General and Administration	2008-04-30 00:00:00.000

SQL:BatchStarting SELECT * FROM HumanResources.Department Microsoft SQL Server jedim Legaci...
SQL:BatchCompleted SELECT * FROM HumanResources.Department Microsoft SQL Server jedim Legaci... 0 67 0 6 18480 73 2025-04-23 13:32:03...
SQL:BatchStarting SELECT * FROM HumanResources.Department Microsoft SQL Server jedim Legaci...
18480 73 2025-04-23 13:32:43...

SELECT * FROM HumanResources.Department

SQLQuery1.sql - LE...GACIES\jedim (73)*

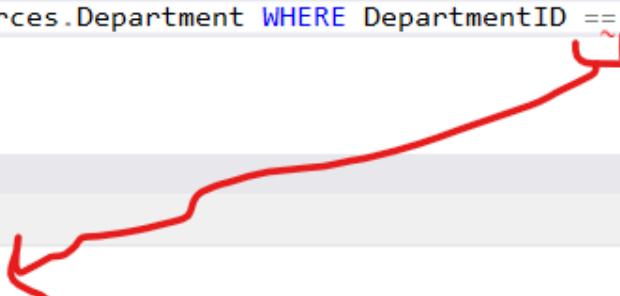
```
SELECT * FROM HumanResources.Department WHERE DepartmentID == 300
```

100 %

Messages

Msg 102, Level 15, State 1, Line 1
Incorrect syntax near '='.

Completion time: 2025-04-23T13:32:51.7250204+03:00



Exception	Incorrect syntax near '='.	Microsoft SQL Server	jedim	Legacy...		18480	73	2025-04-23 13:32:51...
User Error Message	Incorrect syntax near '='.	Microsoft SQL Server	jedim	Legacy...		18480	73	2025-04-23 13:32:51...
SQL:BatchStarting	SELECT * FROM HumanResources.Depart...	Microsoft SQL Server	jedim	Legacy...		18480	73	2025-04-23 13:32:51...

Incorrect syntax near '='.

b. İndeks Yönetimi:

```

USE AdventureWorks2022
SELECT
    OBJECT_NAME(i.object_id) AS TableName,
    i.name AS IndexName,
    i.type_desc,
    ISNULL(s.user_seeks, 0) AS user_seeks,
    ISNULL(s.user_scans, 0) AS user_scans,
    ISNULL(s.user_lookups, 0) AS user_lookups,
    ISNULL(s.user_updates, 0) AS user_updates,
    s.last_user_seek,
    s.last_user_scan,
    s.last_user_lookup
FROM sys.indexes i
LEFT JOIN sys.dm_db_index_usage_stats s
    ON i.object_id = s.object_id AND i.index_id = s.index_id
WHERE OBJECTPROPERTY(i.object_id, 'IsUserTable') = 1
    AND i.name IS NOT NULL
ORDER BY (ISNULL(s.user_seeks, 0) + ISNULL(s.user_scans, 0) + ISNULL(s.user_lookups, 0)) ASC;

```

100 %

Messages

SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2025-04-24T02:52:18.9895584+03:00

Sistemde kullanılmayan ve gereksiz olan indeksleri bulmayı sağlayan sql kodudur.

The screenshot shows a SQL Server Management Studio window with three tabs: SQLQuery7.sql, SQLQuery9.sql, and SQLQuery8.sql. The SQLQuery9.sql tab is active and contains the following T-SQL code:

```
USE AdventureWorks2022
SELECT
    OBJECT_NAME(i.object_id) AS TableName,
    i.name AS IndexName,
    i.type_desc,
    ISNULL(s.user_seeks, 0) AS user_seeks,
    ISNULL(s.user_scans, 0) AS user_scans,
    ISNULL(s.user_lookups, 0) AS user_lookups,
    ISNULL(s.user_updates, 0) AS user_updates,
    s.last_user_seek,
    s.last_user_scan,
    s.last_user_lookup
FROM sys.indexes i
LEFT JOIN sys.dm_db_index_usage_stats s
    ON i.object_id = s.object_id AND i.index_id = s.index_id
WHERE OBJECTPROPERTY(i.object_id, 'IsUserTable') = 1
    AND i.name IS NOT NULL
ORDER BY (ISNULL(s.user_seeks, 0) + ISNULL(s.user_scans, 0) + ISNULL(s.user_lookups, 0)) ASC;
```

The results grid shows a list of tables and their corresponding indexes, along with usage statistics. The table 'Address' has two indexes highlighted in blue: IX_Address_AddressLine1_AddressLine2_City_StateProvinceID and IX_Address_StateProvinceID. Both are listed as NONCLUSTERED indexes with zero user activity.

İndeksleri kaldırırken PK ve AK olmayanları seçmemeliyiz çünkü PK ve AK database'lerde constraint olarak tutulmaktadır. Bu tarz indeksleri silerken permission hatası alırsın. Bu yüzden ismi IX olanların kaldırılması mümkün değildir. Aşağıdaki iki resimde, kaldırmak için IX olarak seçilenler ve indeks kaldırma kodları yer almaktadır.

```
DROP INDEX IX_ShoppingCartItem_ShoppingCartID_ProductID ON Sales.ShoppingCartItem;
```

```
DROP INDEX IX_Address_AddressLine1_AddressLine2_City_StateProvinceID_PostalCode ON Person.Address;
```

Bu iki tane indeks kaldırma komutunu çalıştırmadan önce iki tane base query belirleyip. Bu iki base query'i indeks kaldırma öncesi ve sonrası olmak üzere her biri için 2 defa çalıştırıp Sql profiler kışmından duration süresindeki düşüşü (hız artışı) gözlemleyeceğiz.

SQL:BatchStarting	USE Adventureworks2022	SELECT * F...	Microsoft SQ...	jedim	Legaci...					
SQL:BatchCompleted	USE Adventureworks2022	SELECT * F...	Microsoft SQ...	jedim	Legaci...	0	210	0	3	
SQL:BatchStarting	DROP INDEX IX_ShoppingCartItem_Shopp...	Microsoft SQ...	jedim	Legaci...						
SQL:BatchCompleted	DROP INDEX IX_ShoppingCartItem_Shopp...	Microsoft SQ...	jedim	Legaci...	0	114	15	4		
SQL:BatchStarting	SELECT * FROM Sales.ShoppingCartItem	Microsoft SQ...	jedim	Legaci...						
SQL:BatchCompleted	SELECT * FROM Sales.ShoppingCartItem	Microsoft SQ...	jedim	Legaci...	0	24	0	2		


```
ELECT * FROM Sales.ShoppingCartItem
```

Sorgunun 3 tane sonucu olduğu için bu kadar az sample’ın bulunduğu bir query yerine AddressLine için query yazıp fazlaca büyük sample sayısına sahip query’lerdeki performans artışını gözlemleyeceğiz.

Yaklaşık 2 kat kadar hız artışı vardır. (273 → 146)

c. Sorgu İyileştirme:

Sql’de bir query’de kullanılabilir her keyword’ün database’de oluşturduğu sistem yükleri vardır.

SQL:BatchCompleted	SELECT * FROM Person.Address	Microsoft SQ...	jedim	Legaci...	31	666	0	273
SQL:BatchStarting	DROP INDEX IX_Address_AddressLine1...	Microsoft SQ...	jedim	Legaci...				
SQL:BatchCompleted	DROP INDEX IX_Address_AddressLine1...	Microsoft SQ...	jedim	Legaci...	0	253	10	4
SQL:BatchStarting	SELECT * FROM Person.Address	Microsoft SQ...	jedim	Legaci...				
SQL:BatchCompleted	SELECT * FROM Person.Address	Microsoft SQ...	jedim	Legaci...	16	364	0	146

SELECT * FROM Person.Address		
Keyword 1	Keyword 2	Hangisi daha hızlı?
Where	Having	Where
Select *	Select col1, col2 ...	Select col1, col2, ...
In	Join	Join

Aşağıdaki tüm query’lerin sonuçları birebir aynıdır!

1. Soru (in kullanımı)

SQLQuery2.sql - LE...GACIES\jedim (76) SQLQuery1.sql - LE...GACIES\jedim (55)*

```
SELECT *
FROM Sales.SalesOrderDetail
WHERE ProductID IN (
    SELECT ProductID
    FROM Production.Product
    WHERE Name LIKE '%Road%'
);
```

100 %

Results Messages Execution plan

Query 1: Query cost (relative to the batch) : 100%

```
SELECT * FROM Sales.SalesOrderDetail WHERE ProductID IN ( SELECT ProductID FROM Production.Proc
Missing Index (Impact 99.0792): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] (
```

```
graph TD
    A[SELECT Cost: 0 %] --> B[Hash Match (Inner Join) Cost: 36 %]
    B --> C[Index Scan (NonClustered) Cost: 0 %]
    C --> D[Compute Scalar Cost: 1 %]
    C --> E[Compute Scalar Cost: 1 %]
    D --> F[Clustered Index Scan (Clustered) Cost: 62 %]
```

SQL:BatchStarting	SELECT * FROM Sales.SalesOrderDeta...	Microsoft SQ...	jedim	Legaci...				
SQL:BatchCompleted	SELECT * FROM Sales.SalesOrderDeta...	Microsoft SQ...	jedim	Legaci...	78	1769	0	245
SQL:BatchStarting	SET STATISTICS XML OFF	Microsoft SQ...	jedim	Legaci...				

```
SELECT *
FROM Sales.SalesOrderDetail
WHERE ProductID IN (
    SELECT ProductID
    FROM Production.Product
    WHERE Name LIKE '%Road%'
);
```

2. Soru (Join kullanımı)

SQLQuery2.sql - LE...GACIES\jedim (76) SQLQuery1.sql - LE...GACIES\jedim (55)*

```

SELECT sod.*  

    FROM Sales.SalesOrderDetail sod  

JOIN Production.Product p ON sod.ProductID = p.ProductID  

WHERE p.Name LIKE '%Road%';

```

100 %

Results Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT sod.* FROM Sales.SalesOrderDetail sod JOIN Production.Product p ON sod.ProductID = p.ProductID

Missing Index (Impact 99.0792): CREATE NONCLUSTERED INDEX [<Name of Missing Index, sysname,>] ON [Product].[AK_Product_Name]...

```

graph TD
    A[SELECT sod.*  
FROM Sales.SalesOrderDetail sod  
JOIN Production.Product p ON sod.ProductID = p.ProductID  
WHERE p.Name LIKE '%Road%';] --> B[Hash Match (Inner Join)  
Cost: 36 %  
0.032s  
34628 of  
45973 (75%)]
    B --> C[Index Scan (NonClustered)  
[Product].[AK_Product_Name]...  
Cost: 0 %  
0.001s  
103 of  
101 (101%)]
    C --> D[Compute Scalar  
Cost: 1 %]
    C --> E[Compute Scalar  
Cost: 1 %]
    D --> F[Clustered Index Scan (Clustered Index Scan)  
[SalesOrderDetail].[PK_SalesOrderDetail]  
Cost: 62 %  
0.019s  
121317 of  
121317 (100%)]

```

SQL:BatchStarting

SQL:BatchCompleted

SQL:BatchStarting

SQL:BatchCompleted

	Microsoft SQ...	jedim	Legaci...				
SELECT sod.* FROM Sales.SalesOrderD...	Microsoft SQ...	jedim	Legaci...	109	1937	0	233
SELECT sod.* FROM Sales.SalesOrderD...	Microsoft SQ...	jedim	Legaci...	0	0	0	0
SET STATISTICS XML OFF	Microsoft SQ...	jedim	Legaci...				
SET STATISTICS XML OFF	Microsoft SQ...	jedim	Legaci...				

```

SELECT sod.*  

FROM Sales.SalesOrderDetail sod  

JOIN Production.Product p ON sod.ProductID = p.ProductID  

WHERE p.Name LIKE '%Road%';

```

3. Sorgu (Join + Select col1, col2, ...)

SQLQuery2.sql - LE...GACIES\jedim (76) ✎ X SQLQuery1.sql - LE...GACIES\jedim (55)* ✎ X

```
SELECT sod.SalesOrderID, sod.SalesOrderDetailID, sod.CarrierTrackingNumber
FROM Sales.SalesOrderDetail sod
JOIN Production.Product p ON sod.ProductID = p.ProductID
WHERE p.Name LIKE '%Road%';
```

100 % ▾

Results Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

```
SELECT sod.SalesOrderID, sod.SalesOrderDetailID, sod.CarrierTrackingN
Missing Index (Impact 99.2437): CREATE NONCLUSTERED INDEX [<Name of M]
```

```
graph TD
    A[Hash Match  
(Inner Join)  
Cost: 37 %  
0.031s  
34628 of  
45973 (75%)] --> B[Index Scan (NonClustered)  
[Product].[AK_Product_Name]...  
Cost: 0 %  
0.000s  
103 of  
101 (101%)]
    A --> C[Clustered Index Scan (Cluste...  
[SalesOrderDetail].[PK_Sales...  
Cost: 63 %  
0.018s  
121317 of  
121317 (100%)]
    B --> C
```

d. Veri Yöneticisi Rollerı:

```
SQLQuery2.sql - LE...GACIES\jedim (76)          SQLQuery1.sql - LE...GACIES\jedim (55)*  ✎ X
CREATE LOGIN report_user_login WITH PASSWORD = 'Report123!';
CREATE LOGIN editor_user_login WITH PASSWORD = 'Edit123!';
CREATE LOGIN admin_user_login WITH PASSWORD = 'Admin123!';

100 %  ◀
Messages
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 2 ms.
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2025-04-22T15:32:58.0878185+03:00
```

Sql server authentication için login oluşturulması lazım.

```
USE AdventureWorks2022;
CREATE USER report_user FOR LOGIN report_user_login;
CREATE USER editor_user FOR LOGIN editor_user_login;
CREATE USER admin_user FOR LOGIN admin_user_login;

100 %  ◀
Messages
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 2 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.

SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2025-04-22T15:33:30.8323288+03:00
```

Sonrasında kullanıcılar oluşturulmalı ve login'e bağlanmalı.

```
CREATE ROLE report_role;
CREATE ROLE editor_role;
CREATE ROLE admin_role;
```

100 %

Messages

```
Commands completed successfully.
```

Veritabanına bu kullanıcılara yetkilendirme verebilmek için roller oluşturulmalı.

```
GRANT SELECT ON SCHEMA::Sales TO report_role;
GRANT SELECT, UPDATE ON SCHEMA::Sales TO editor_role;
GRANT SELECT, INSERT, UPDATE, DELETE ON SCHEMA::Sales TO admin_role;
```

100 %

Messages

```
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.
```

```
SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
```

```
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.
```

```
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.
```

```
SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
```

```
Completion time: 2025-04-22T15:37:05.9096362+03:00
```

Rollere yetkilendirme verilmeli.

```
EXEC sp_addrolemember 'report_role', 'report_user';
EXEC sp_addrolemember 'editor_role', 'editor_user';
EXEC sp_addrolemember 'admin_role', 'admin_user';
```

100 %

Messages

```
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 0 ms.
```

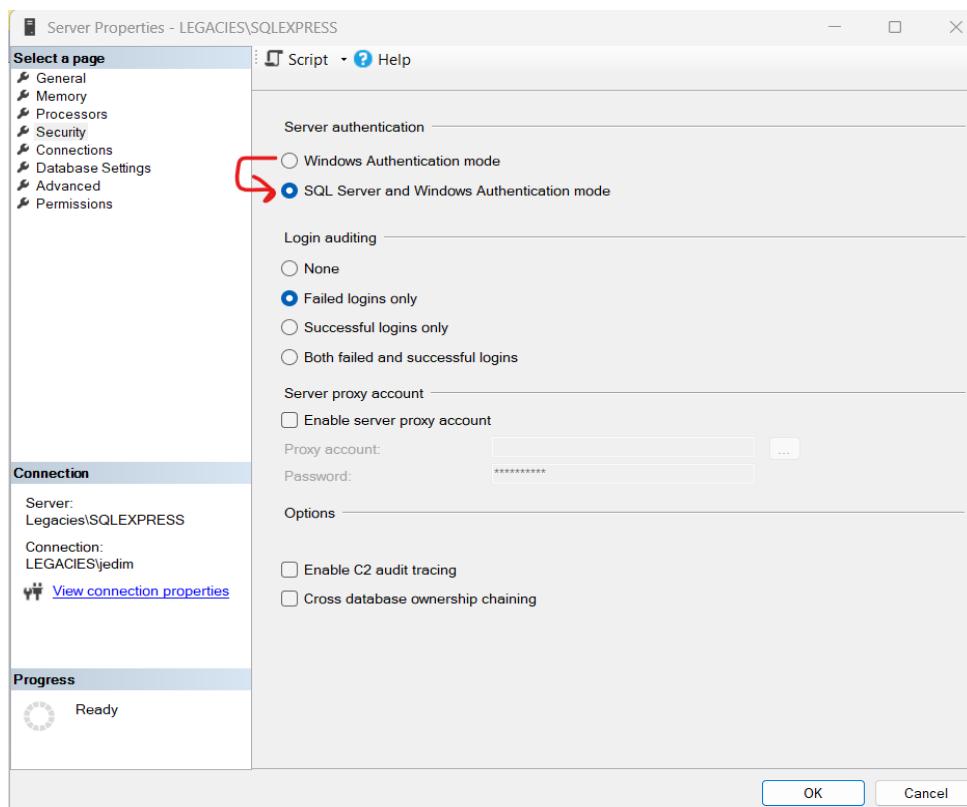
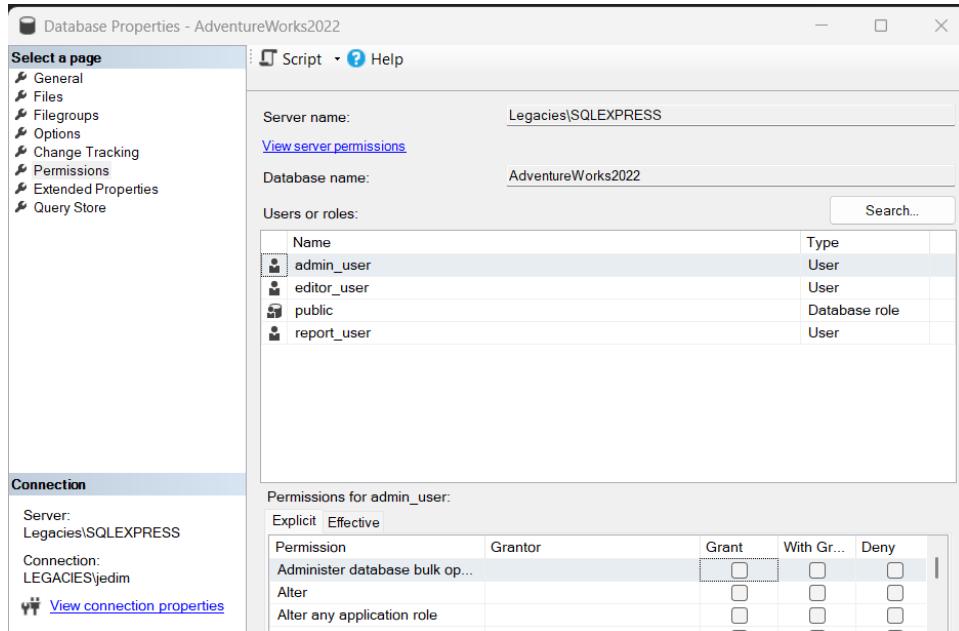
```
SQL Server Execution Times:
CPU time = 0 ms, elapsed time = 0 ms.
```

```
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 2 ms.
```

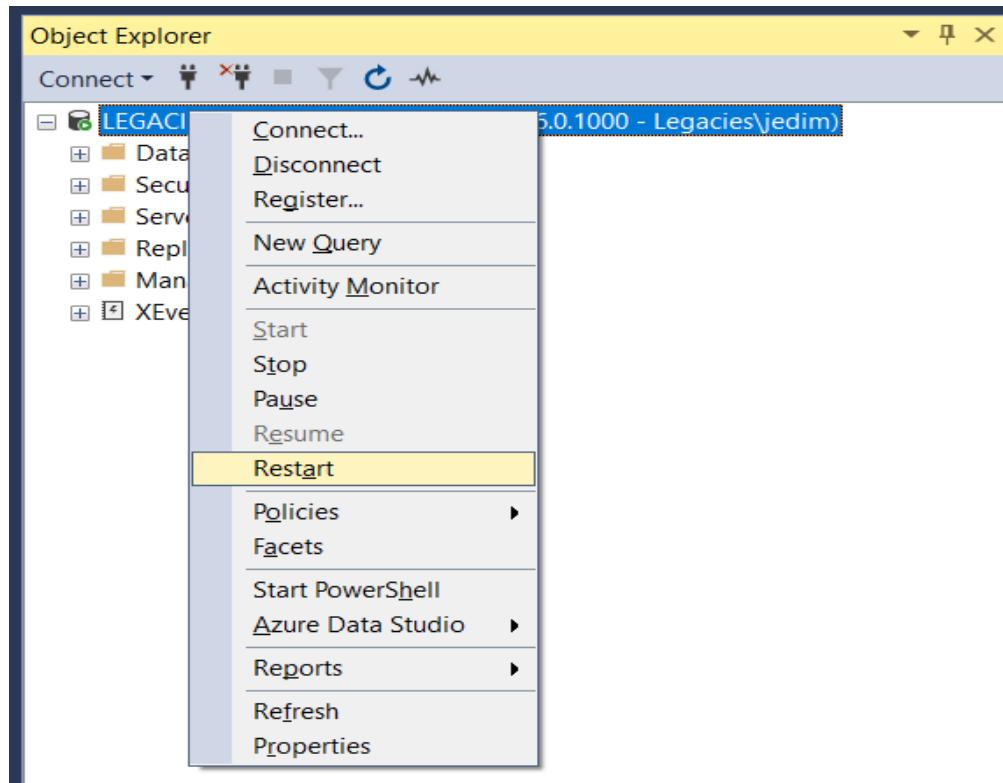
```
SQL Server parse and compile time:
CPU time = 0 ms, elapsed time = 2 ms.
```

Roller veritabanına sp_addrolemember komutu ile assign edilmeli.

Hata almamak için database ayarlarından sql server authentication özelliğinin aktif edilmesi lazım.

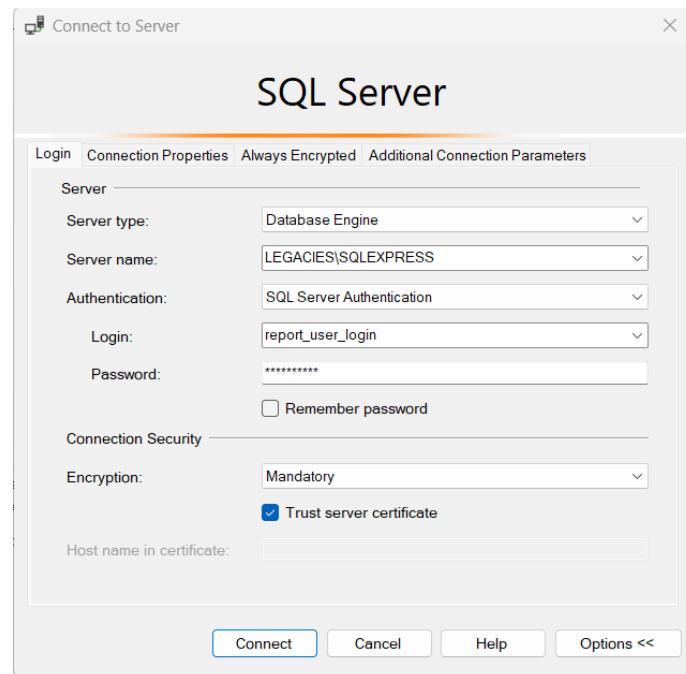


Aktif olduğundan emin olmak için server'ın aktif edilmesi lazım.



Eklenen kullanıcılarla giriş yapıp yetkilendirmelerinin doğru çalışıp çalışmadığını test edelim.

Report Kullanıcısı için:



SQLQuery2.sql - LEG...t_user_login (107)* # X SQLQuery1.sql - LE...GACIES\jedim (60)*

```
SELECT TOP 10 * FROM Sales.SalesOrderHeader; -- izin verilen
```

100 %

Results Messages

SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate
1	43659	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
2	43660	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
3	43661	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
4	43662	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
5	43663	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
6	43664	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
7	43665	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
8	43666	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
9	43667	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000
10	43668	2011-05-31 00:00:00.000	2011-06-12 00:00:00.000	2011-06-07 00:00:00.000

UPDATE Sales.SalesOrderHeader SET Comment = 'Test' WHERE SalesOrderID = 10; -- izin verilmeyen

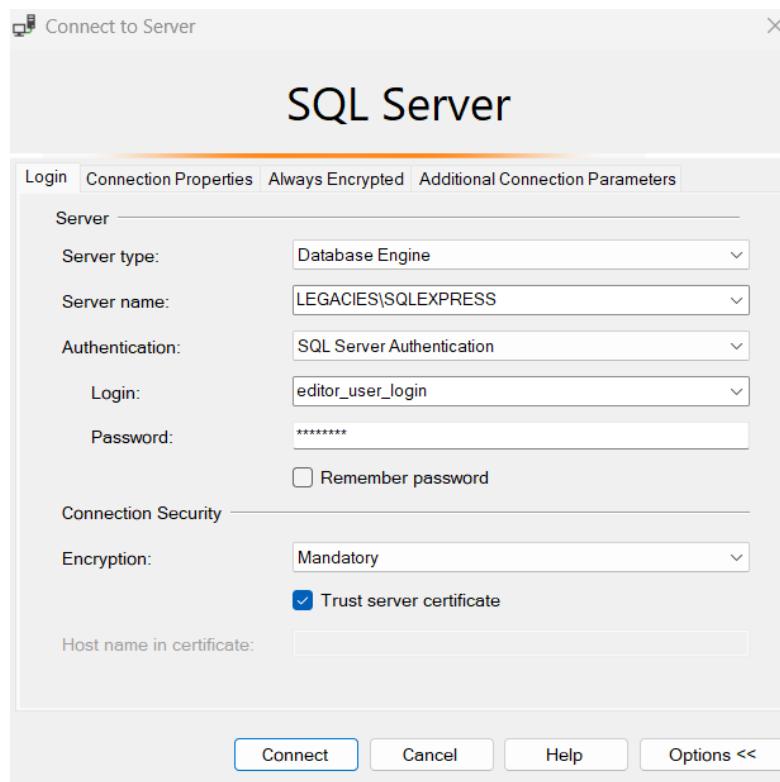
100 %

Messages

```
Msg 229, Level 14, State 5, Line 3
The UPDATE permission was denied on the object 'SalesOrderHeader', database 'AdventureWorks2022', schema 'Sales'.
```

Completion time: 2025-04-22T16:04:26.2790325+03:00

Editor Kullanıcısı için:



SQLQuery3.sql - LEG...or_user_login (67)*

```
SELECT * FROM Sales.SalesOrderHeader;
```

100 %

Results Messages

	SalesOrderID	RevisionNumber	OrderDate	D
1	43659	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
2	43660	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
3	43661	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
4	43662	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
5	43663	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
6	43664	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
7	43665	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
8	43666	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
9	43667	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
10	43668	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
11	43669	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000
12	43670	8	2011-05-31 00:00:00.000	2011-05-31 00:00:00.000

```
SELECT SalesOrderID, Comment FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43659;
UPDATE Sales.SalesOrderHeader SET Comment = 'Updated' WHERE SalesOrderID = 43659;
SELECT SalesOrderID, Comment FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43659;
```

100 %

Results Messages

SalesOrderID	Comment	
1	43659	NULL

SalesOrderID	Comment	
1	43659	Updated

```
DELETE FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43659;
```

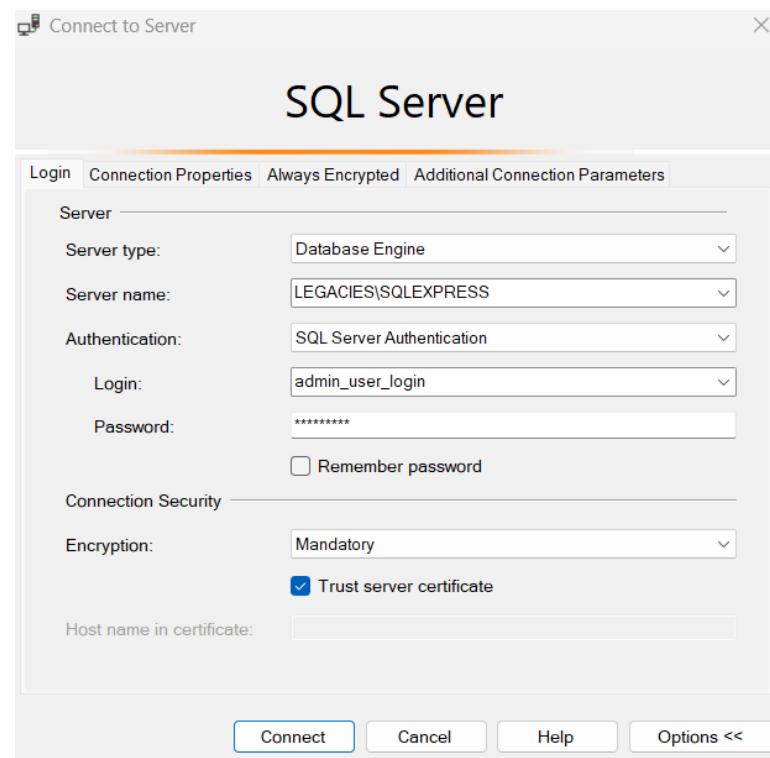
100 %

Messages

Msg 229, Level 14, State 5, Line 7
The DELETE permission was denied on the object 'SalesOrderHeader', database 'AdventureWorks2022', schema 'Sales'.

Completion time: 2025-04-22T16:12:23.6185134+03:00

Admin Kullanıcısı için:



The screenshot shows the SSMS interface with a query window titled 'SQLQuery4.sql - LEG...in_user_login (83)*'. The query executed is 'SELECT * FROM Sales.SalesOrderHeader;'. The results pane displays the following data:

	SalesOrderID	RevisionNumber	OrderDate	DueDate
1	43659	9	2011-05-31 00:00:00.000	2011
2	43660	8	2011-05-31 00:00:00.000	2011
3	43661	8	2011-05-31 00:00:00.000	2011
4	43662	8	2011-05-31 00:00:00.000	2011
5	43663	8	2011-05-31 00:00:00.000	2011
6	43664	8	2011-05-31 00:00:00.000	2011

```

SQLQuery4.sql - LEG...in_user_login (83)*  □ X
SELECT * FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43659;
DELETE FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43659;
SELECT * FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43659;
100 %
Results Messages
SalesOrderID RevisionNumber OrderDate DueDate ShipDate
1 43659 9 2011-05-31 00:00:00.000 2011-06-12 00:00:00.000 2011-06-07 00:00:00.000

```

```

SELECT SalesOrderID, Comment FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43660;
UPDATE Sales.SalesOrderHeader SET Comment = 'Admin OK' WHERE SalesOrderID = 43660;
SELECT SalesOrderID, Comment FROM Sales.SalesOrderHeader WHERE SalesOrderID = 43660;
100 %
Results Messages
SalesOrderID Comment
1 43660 NULL

```

SalesOrderID	Comment	
1	43660	Admin OK

```

SELECT * FROM Sales.SalesOrderHeader WHERE RevisionNumber = 1;
INSERT INTO [AdventureWorks2022].[Sales].[SalesOrderHeader] (RevisionNumber, OrderDate, DueDate, ShipDate, Status, OnlineOrderFlag,
PurchaseOrderNumber, AccountNumber, CustomerID, SalesPersonID, TerritoryID, BillToAddressID, ShipToAddressID,
ShipMethodID, CreditCardID, CreditCardApprovalCode, CurrencyRateID, SubTotal, TaxAmt, Freight, Comment, Rowguid,
ModifiedDate)
VALUES (
    1, GETDATE(), DATEADD(DAY, 5, GETDATE()), DATEADD(DAY, 7, GETDATE()), 5, 1, N'P099999', N'AC1234567', 11000, 279, 3,
    1, 1, 4, 10, N'APPROVED123', 1, 100.00, 10.00, 5.00, N'Test Order Inserted', NEWID(), GETDATE()
);
SELECT * FROM Sales.SalesOrderHeader WHERE RevisionNumber = 1;
100 %
Results Messages

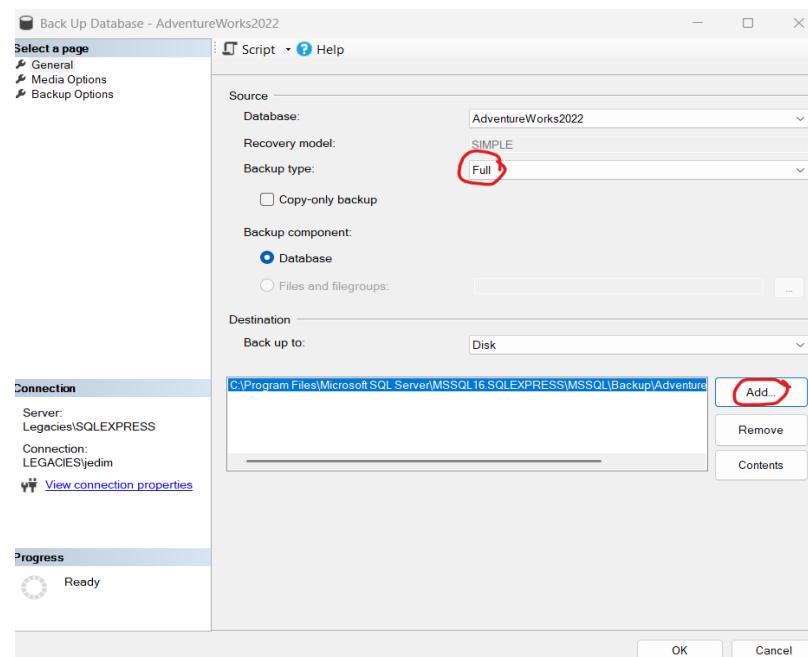
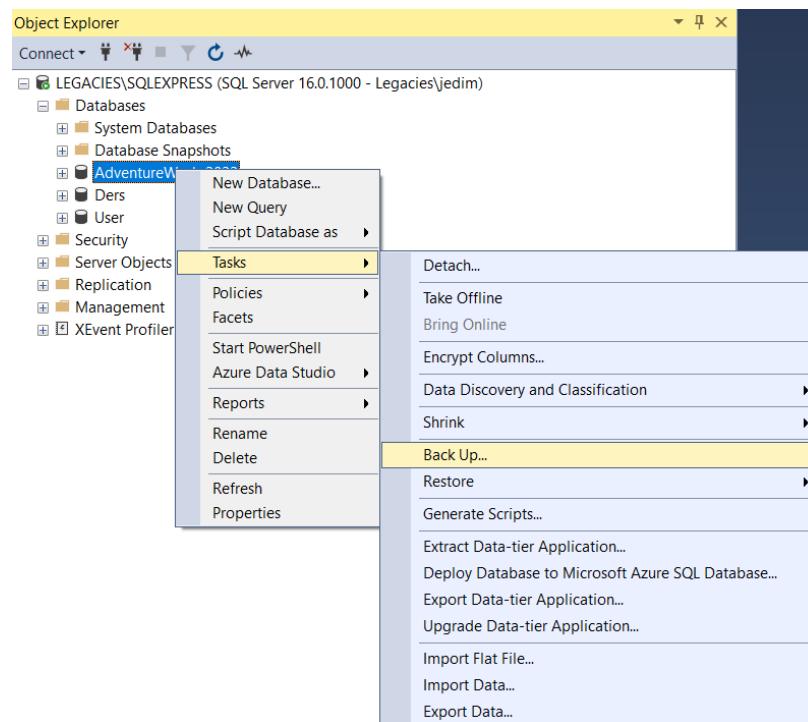
```

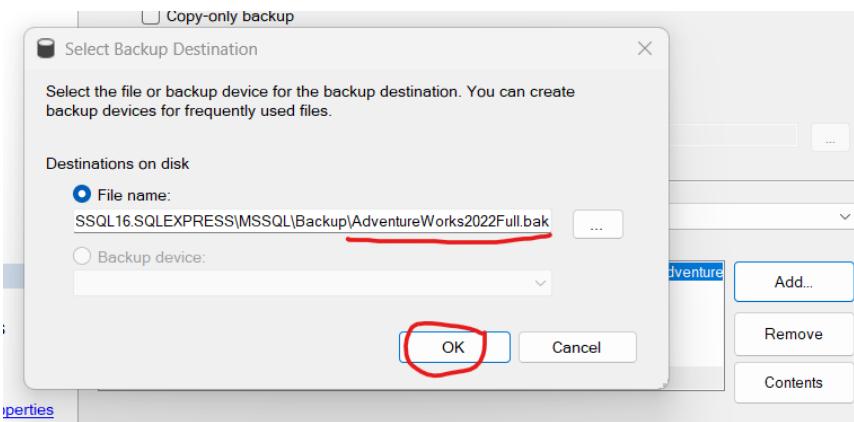
SalesOrderID	RevisionNumber	OrderDate	DueDate	ShipDate	Status	OnlineOrderFlag	SalesOrderNumber	PurchaseOrderNumber	AccountNumber	CustomerID	
1	75125	1	2025-04-22 16:28:21.360	2025-04-27 16:28:21.360	2025-04-29 16:28:21.360	5	1	S075125	P099999	AC1234567	11000

2. Veritabanı Yedekleme ve Felaketten Kurtarma Planı

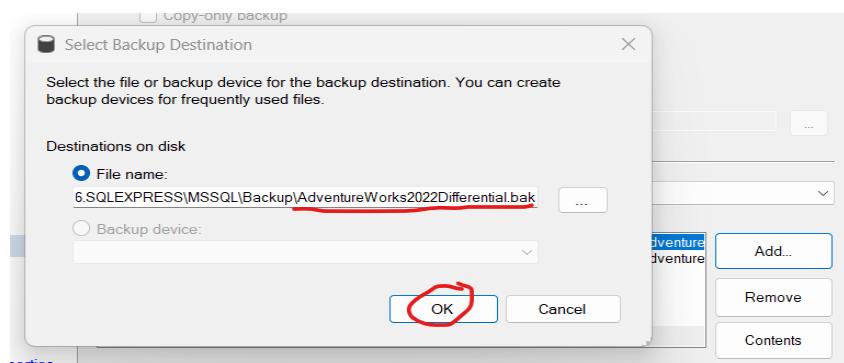
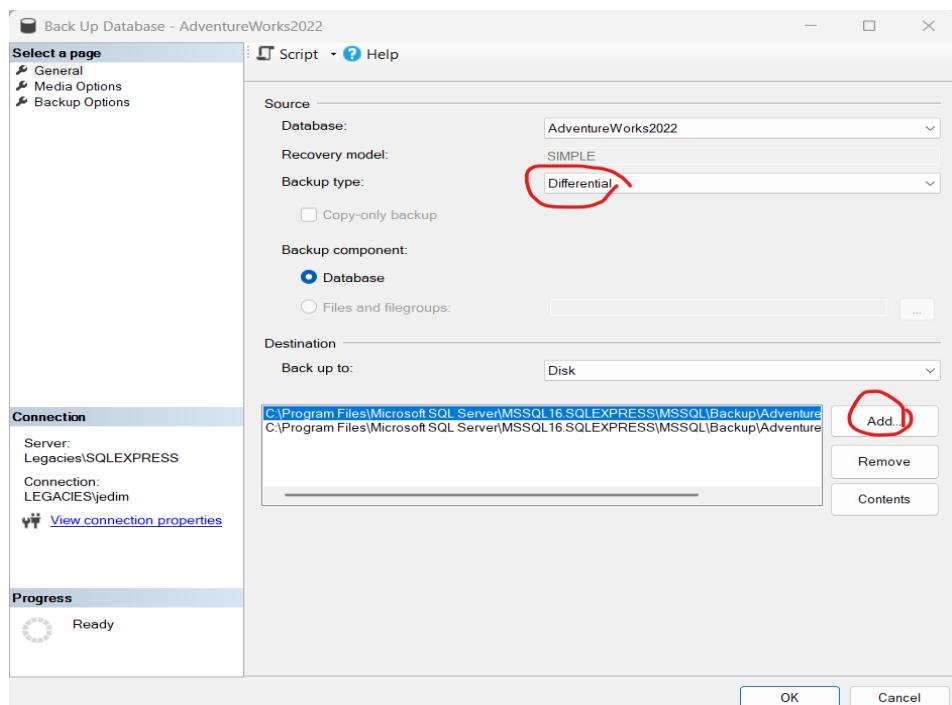
a. Tam, Artık ve Fark Yedeklemeleri:

Tam Yedekleme:

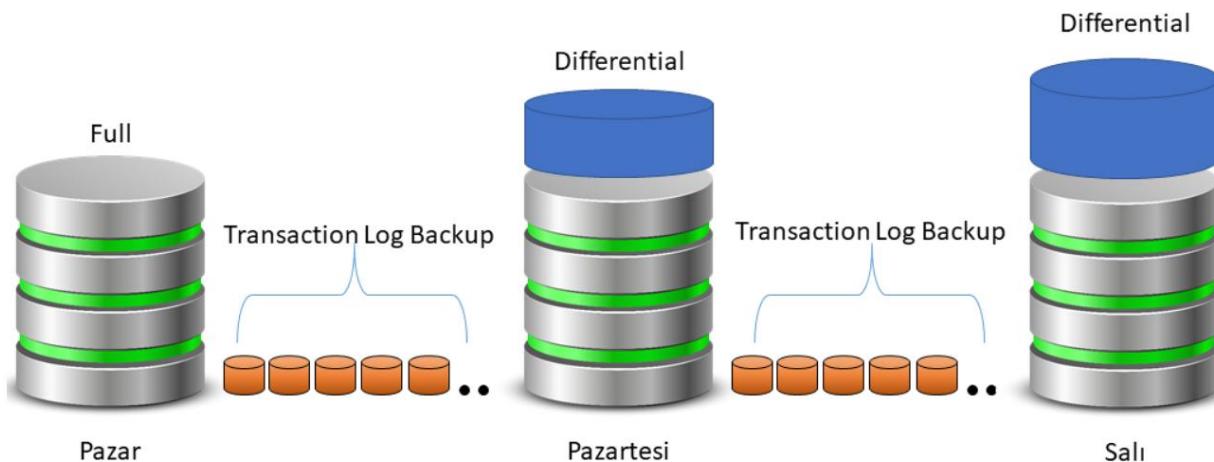




Fark Yedekleme:



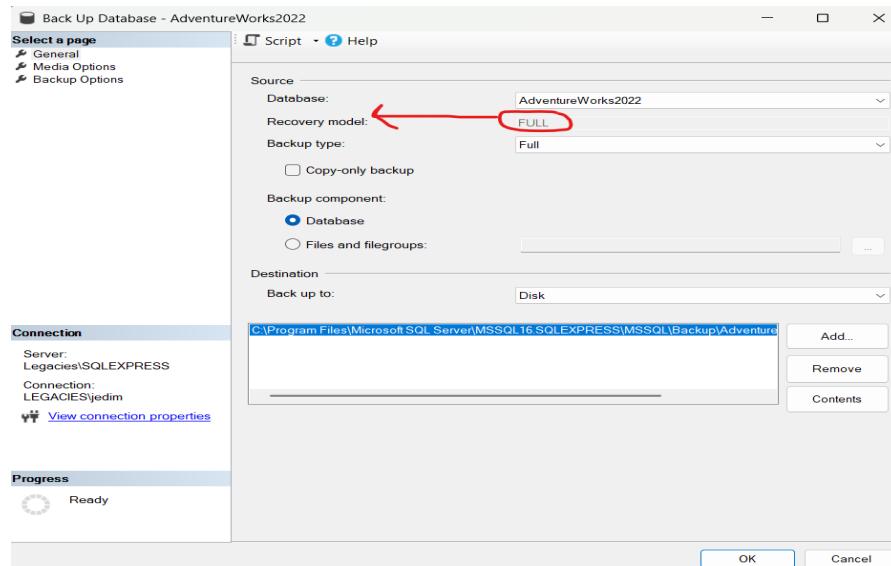
Artık Yedekleme:

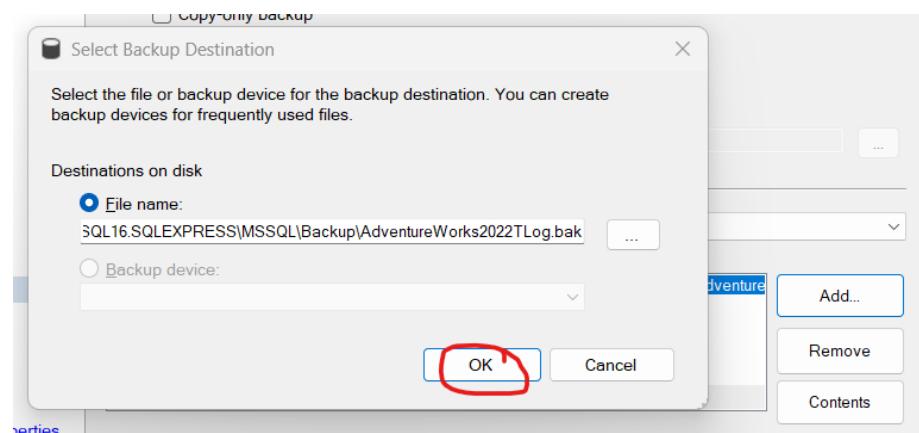
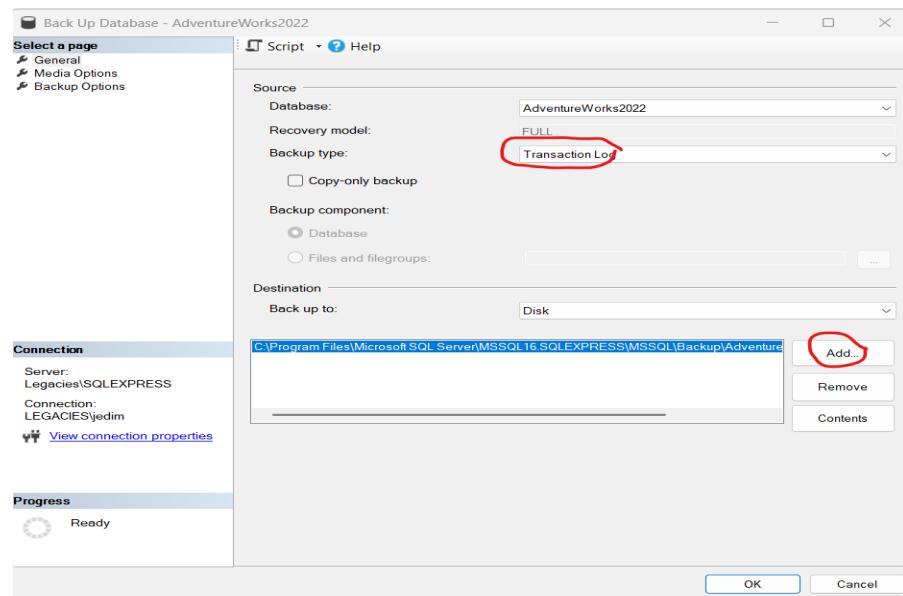


Artık yedekleme yapabilmek için öncesinde full yedeklemenin yapılmış olması lazım ve recovery modun FULL olması lazım çünkü artık yedeklemede

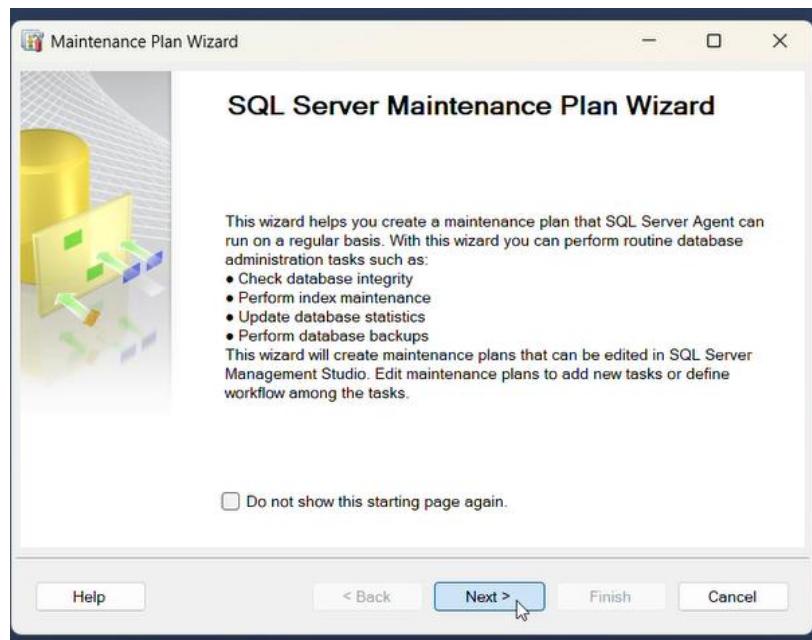
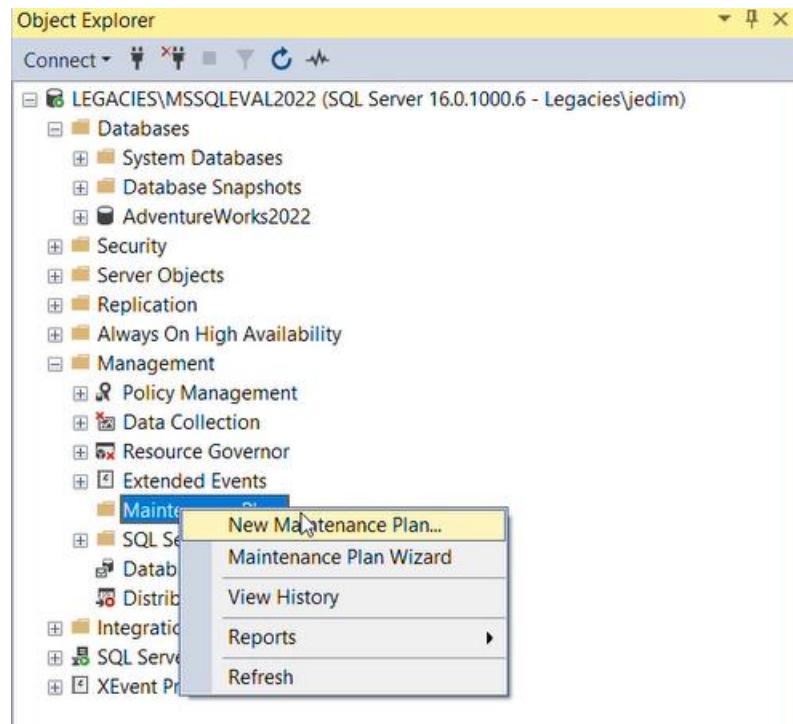
```
SQLQuery1.sql - LE...GACIES\jedim (53)* ↗ X
USE [master]
ALTER DATABASE AdventureWorks2022 SET RECOVERY FULL;
100 % ▾
Messages
Commands completed successfully.

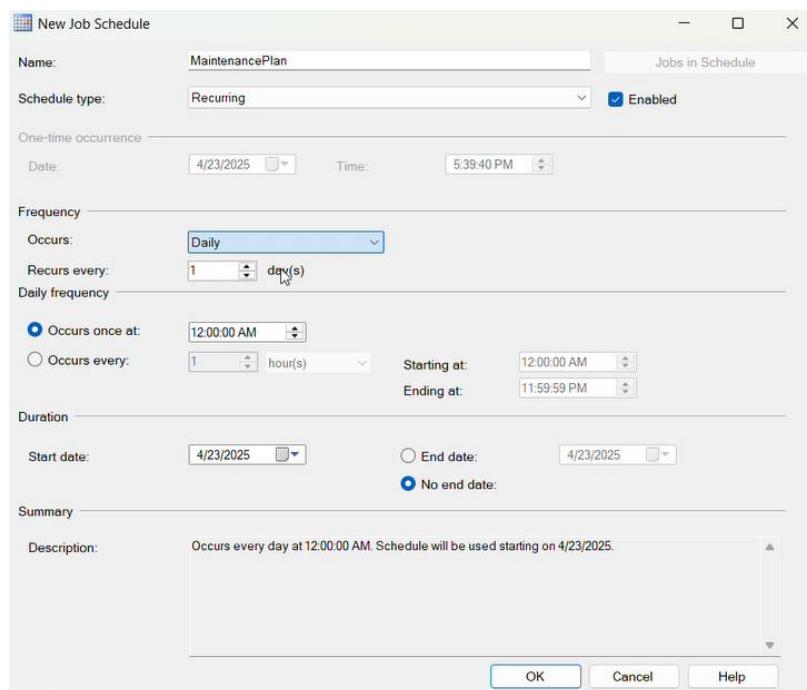
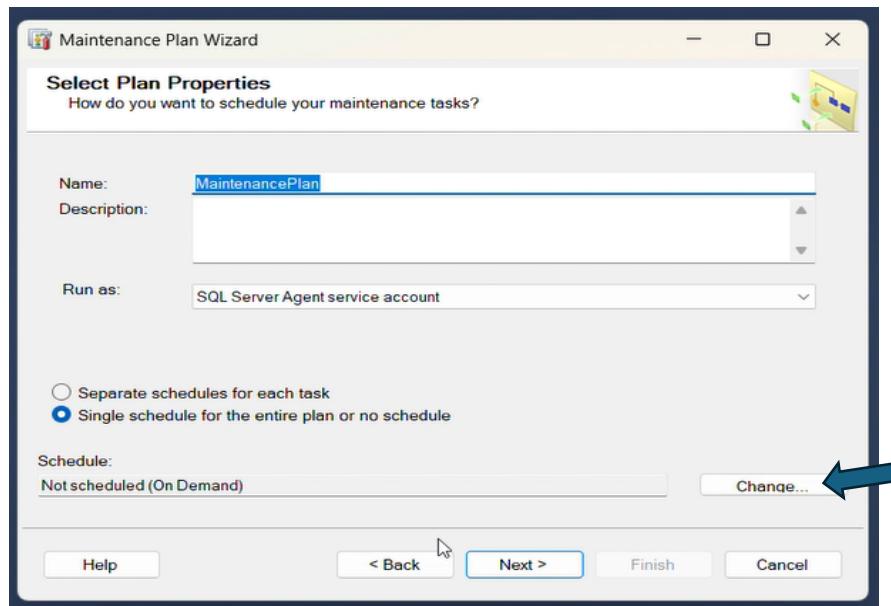
Completion time: 2025-04-22T21:59:45.8727157+03:00
```

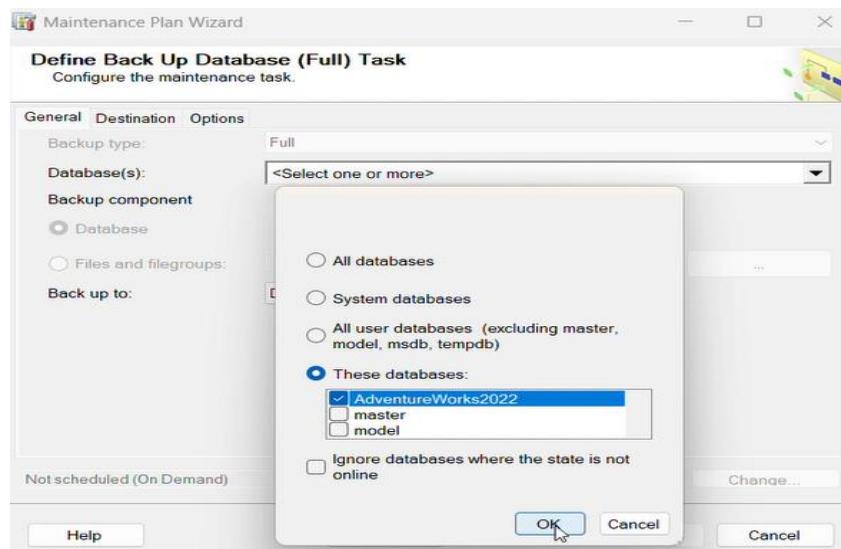
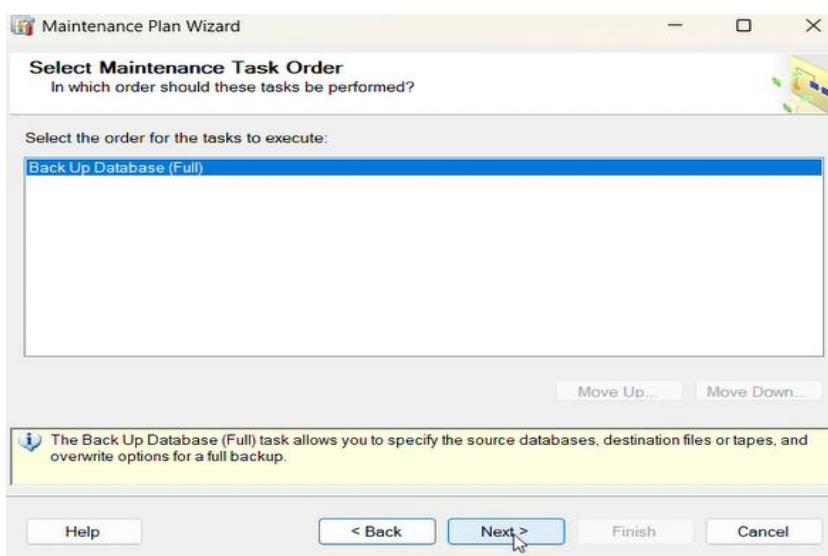
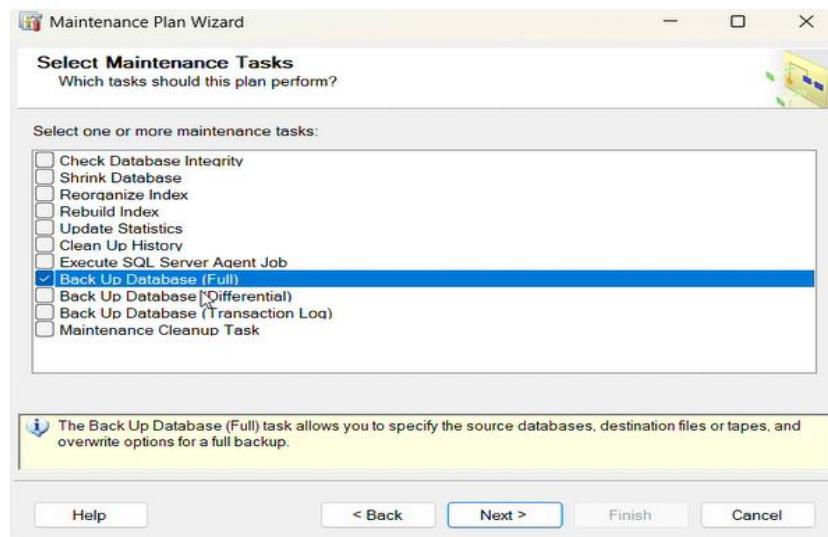


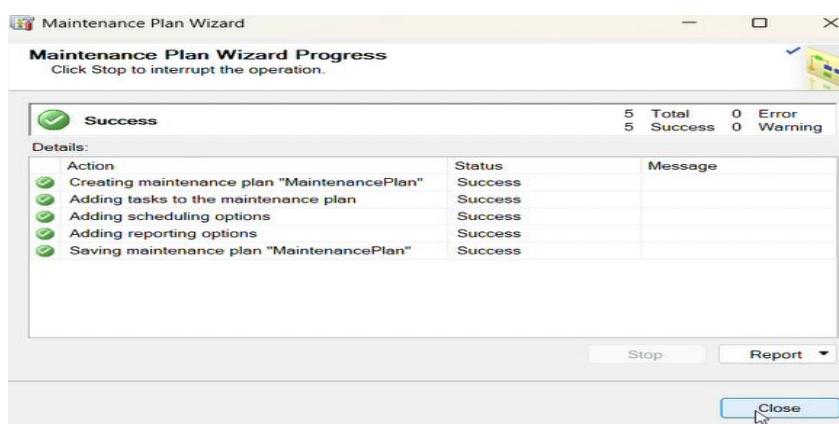
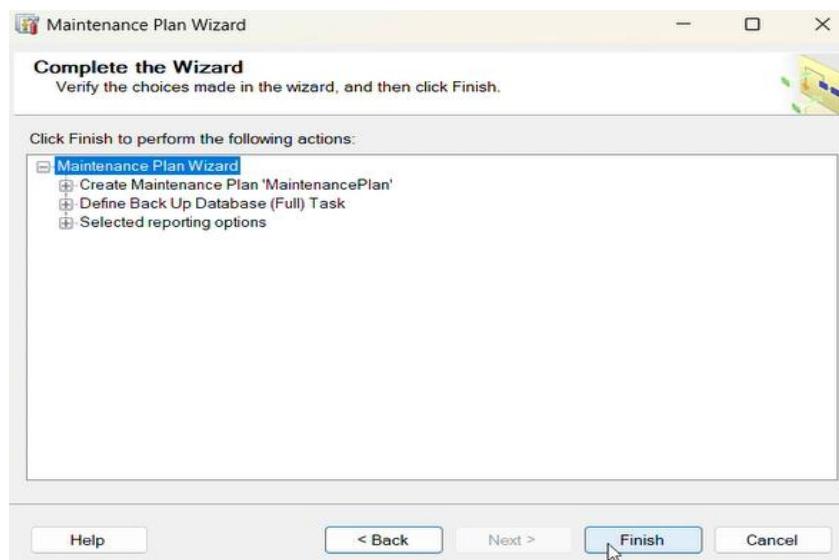
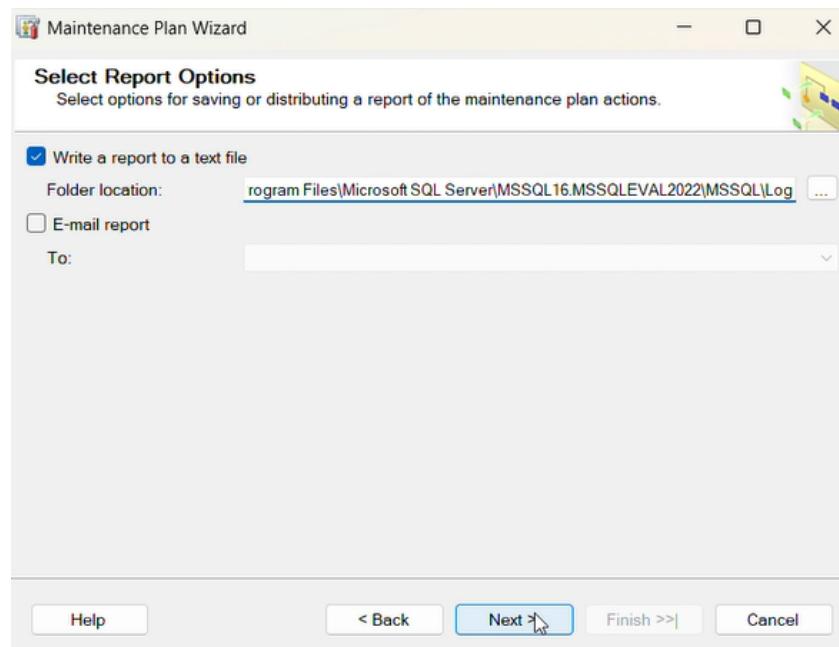


b. Zamanlayıcılarla Yedeklemeleri:



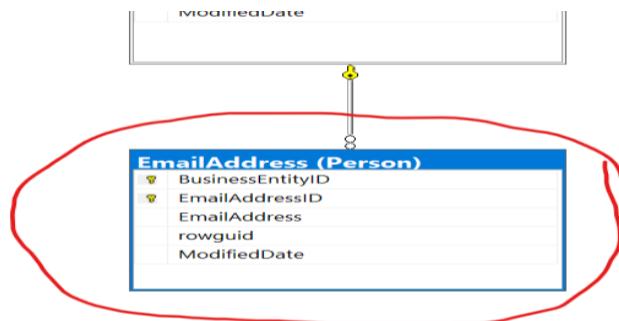






c. Felaketten Kurtarma Senaryoları:

Burada yanlışlıkla Person.EmailAddress table'ını sildiğimiz senaryoyu ele alıyoruz.
Normal şartlarda kendisinden pk alınan bir tabloyu silemeyez ama bu senaryoda herhangi bir fk bağımlılığı olmayan Person.EmailAddress tablosunu siliyoruz.



Tabloyu siliyoruz.

```
Legacies\SQLEXPRESS2022 - Diagram_0*
SQLQu
DROP TABLE Person.EmailAddress;

100 % < 
Messages
Commands completed successfully.

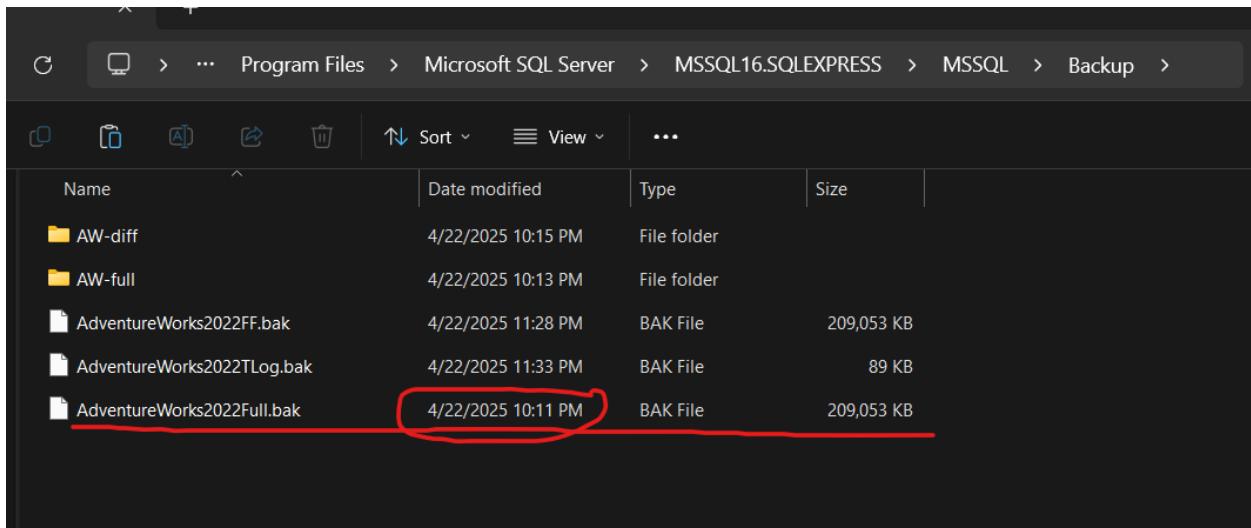
Completion time: 2025-04-23T00:25:47.2161830+03:00
```

```
SELECT * FROM Person.EmailAddress

100 % <
Messages
Msg 208, Level 16, State 1, Line 3
Invalid object name 'Person.EmailAddress'.

Completion time: 2025-04-23T00:26:57.6764682+03:00
```

Silinen tabloyu geri getirmek için backup yaptığımız dosyadan restore ediyoruz.



The screenshot shows the SSMS interface. In the Object Explorer, the 'Adventureworks' database is selected. A context menu is open over the database, with 'Restore' highlighted. The SQL Query Editor window contains the following code:

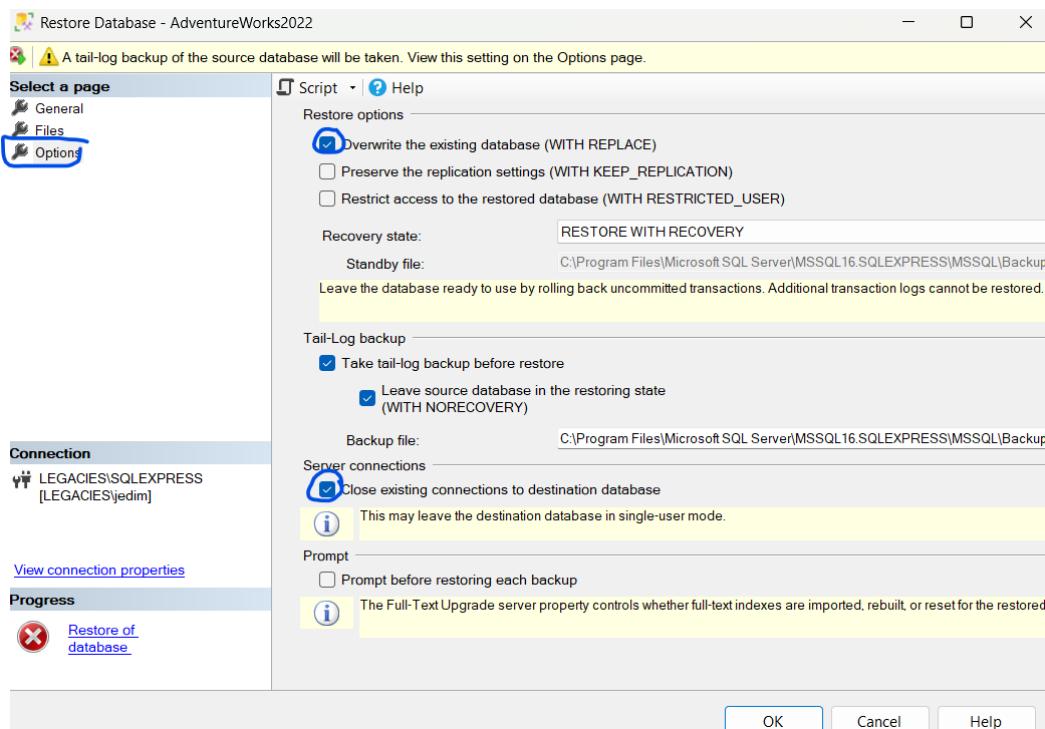
```
DROP TABLE Person.EmailAddress;
SELECT * FROM Person.EmailAddress;
```

The Messages pane shows an error:

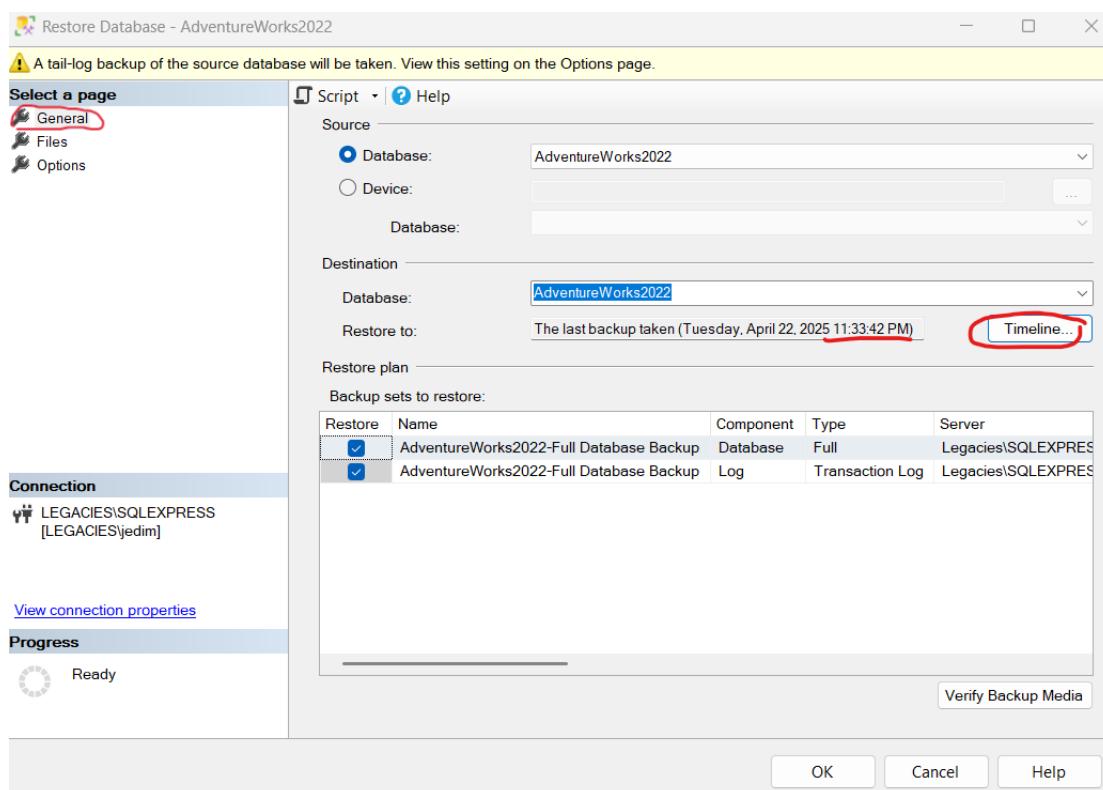
```
Msg 208, Level 16, State 1, Line 3
Invalid object name 'Person.EmailAddress'.
```

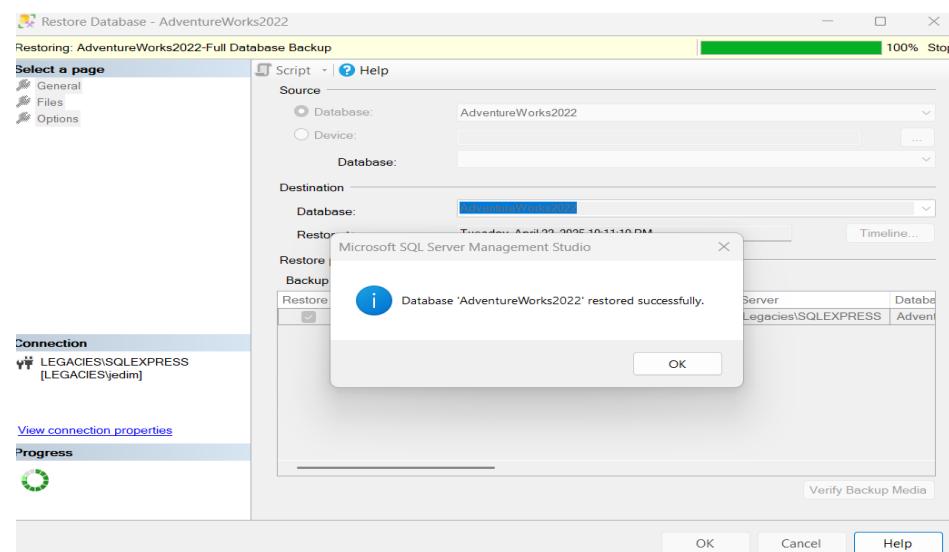
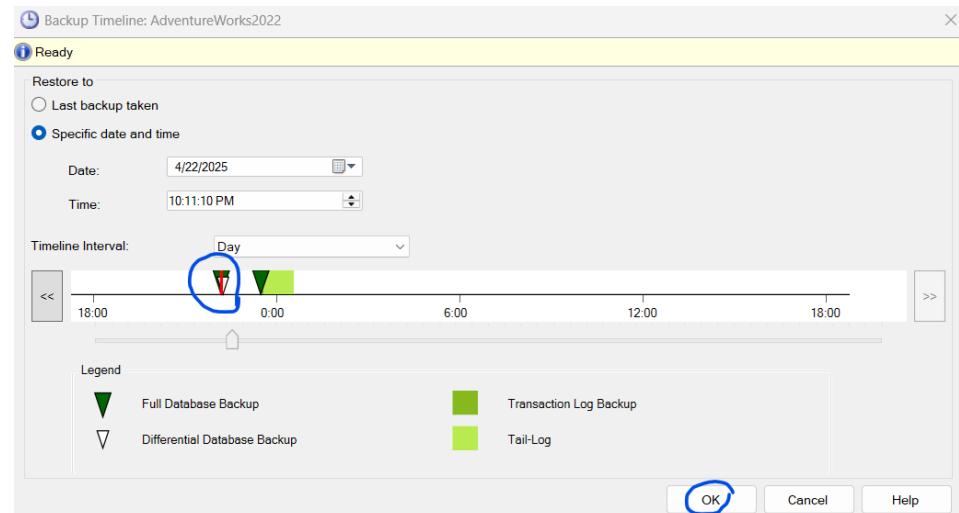
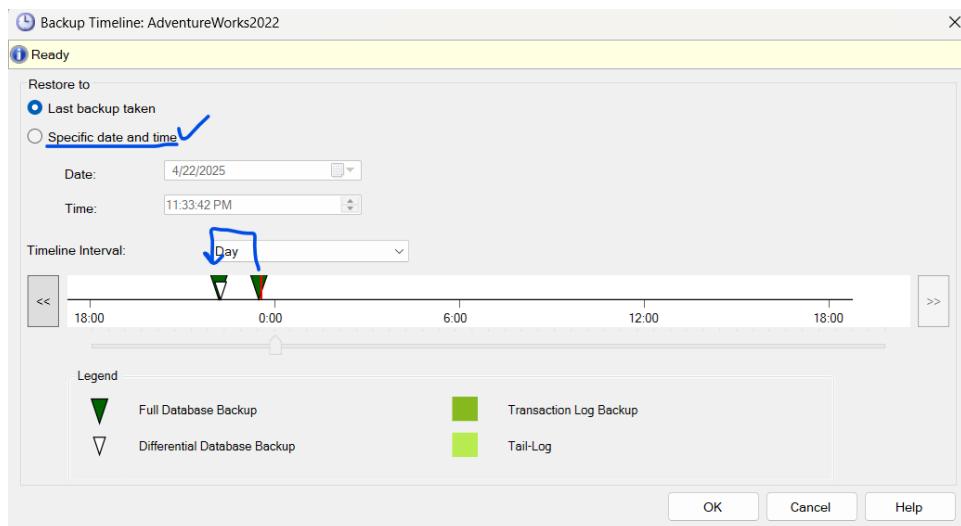
The completion time is 2025-04-23T00:26:57.6764682+03:00.

A tooltip for 'Restore' lists three options: Database..., Files and Filegroups..., and Transaction Log... .



Yukarıdaki seçeneklerden, destination database’ın bağlantısını kopartmazsa hata alırız.





The screenshot shows the SSMS interface. On the left, the Object Explorer pane displays a tree view of database objects under the 'Tables' category. A blue oval highlights the 'Person.EmailAddress' table. On the right, the 'Diagram_0*' window shows a query result grid for the 'Person.EmailAddress' table. The grid has columns: BusinessEntityID, EmailAddressID, and EmailAddress. The data consists of 19 rows of email addresses.

	BusinessEntityID	EmailAddressID	EmailAddress
1	1	1	ken0@Adventureworks.com
2	2	2	terri0@Adventureworks.com
3	3	3	roberto0@Adventureworks.com
4	4	4	rob0@Adventureworks.com
5	5	5	gail0@Adventureworks.com
6	6	6	jossef0@Adventureworks.com
7	7	7	dylan0@Adventureworks.com
8	8	8	diane1@Adventureworks.com
9	9	9	gigi0@Adventureworks.com
10	10	10	michael6@Adventureworks.com
11	11	11	ovidiu0@Adventureworks.com
12	12	12	thierry0@Adventureworks.com
13	13	13	janice0@Adventureworks.com
14	14	14	michael8@Adventureworks.com
15	15	15	sharon0@Adventureworks.com
16	16	16	david0@Adventureworks.com
17	17	17	kevin0@Adventureworks.com
18	18	18	john5@Adventureworks.com
19	19	19	louis7@Adventureworks.com

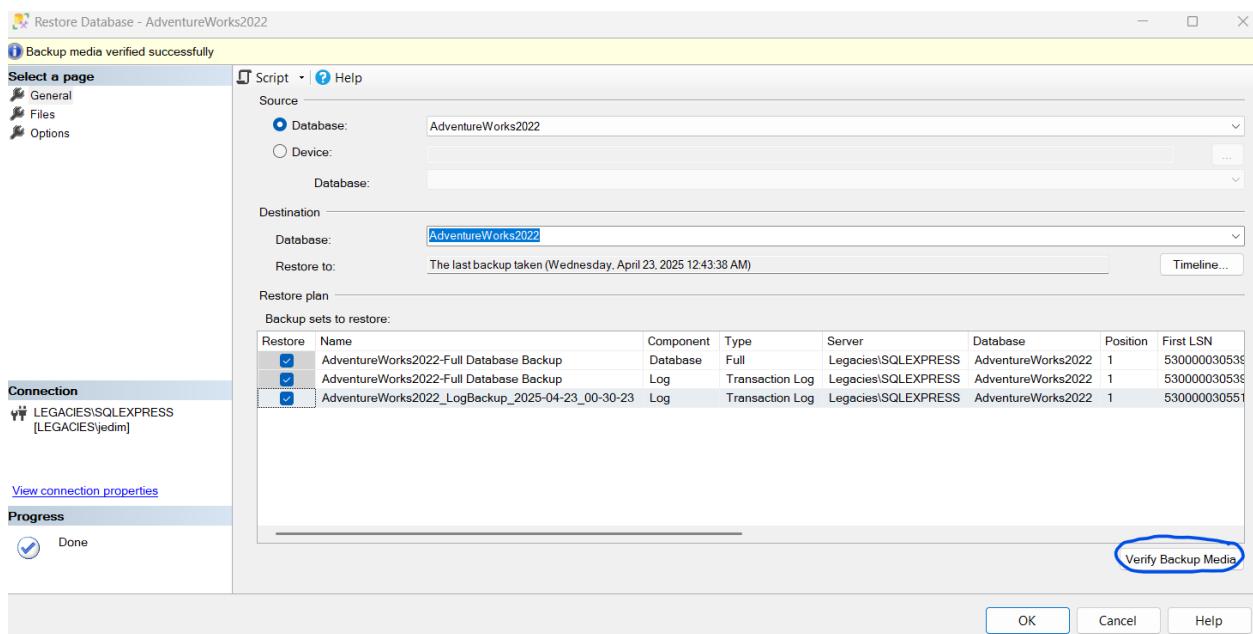
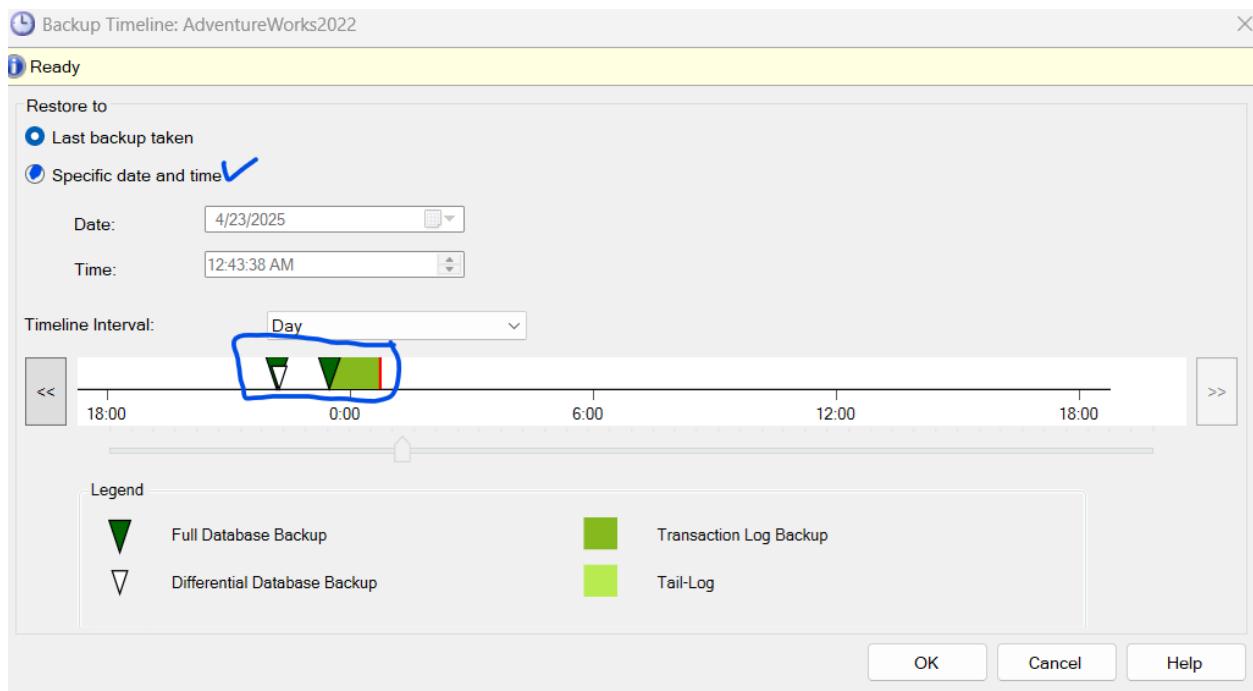
d. Test Yedekleme Senaryoları:

Bu kısımda hem sql kodu ile hem de ssms'in ui'si ile nasıl yapılacağını ele alacağız.

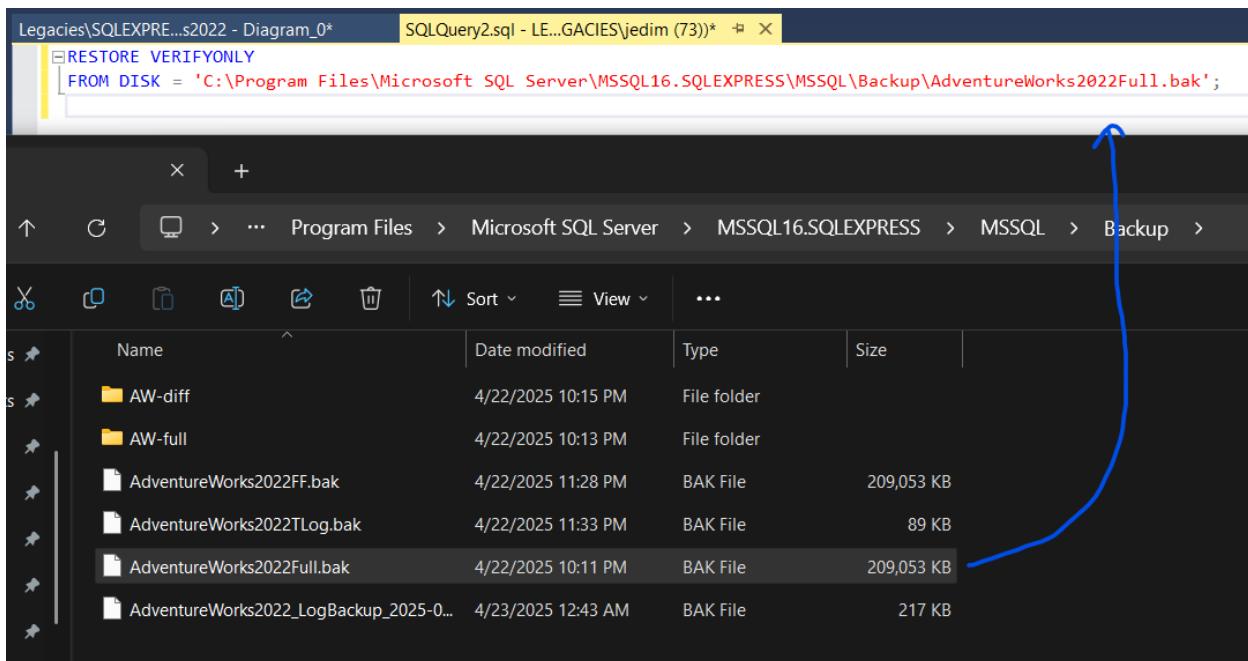
Database'e sağ tıklayıp backup kısmına tıklayacağız.

The screenshot shows the 'Restore Database' dialog box. In the 'Source' section, 'Database' is selected and 'AdventureWorks2022' is chosen. In the 'Destination' section, 'Database' is also set to 'AdventureWorks2022'. The 'Restore to:' field shows the restore date as 'Tuesday, April 22, 2025 11:28:47 PM'. A blue oval highlights the 'Timeline...' button. In the 'Restore plan' section, there is a table titled 'Backup sets to restore'. The first row in this table is highlighted with a blue oval. The table has columns: Restore, Name, Component, Type, Server, and Database. The 'Restore' column has a checked checkbox. The 'Name' column shows 'AdventureWorks2022-Full Database Backup'. The 'Component' column is 'Database', 'Type' is 'Full', 'Server' is 'Legacies\SQLEXPRESS', and 'Database' is 'AdventureWorks2022'. At the bottom right of the dialog are 'OK', 'Cancel', and 'Help' buttons.

Verify etmek istediğimiz database backup'ını ve üzerinde daha önce yapılmış backup geçmişini seçip verify edebiliriz.



Buraya kadar olan kısım ssms'in ui'ı ile öncesinde backup ettiğimiz database'in verify edilmesini içeriyyordu. Şimdi ise kod ile nasıl yapılacağını ele alacağız. Kod ile yapması çok daha hızlı ve basittir.



3. Veritabanı Güvenliği ve Erişim Kontrolü

a. Erişim Yönetimi:

Bu kısmı, [“Veritabanı Performans Optimizasyonu ve İzleme”](#) projesinin [“Veri Yöneticisi Rolleri”](#) kısmında ele aldığımız için burada tekrardan yazmaya gerek yoktur. Bu kısımla ilgili her şey, ilk projede detaylıca ele alınmıştır.

b. Veri Şifreleme:

```
/*
https://learn.microsoft.com/en-us/sql/relational-databases/security/
USE master;
GO

CREATE MASTER KEY ENCRYPTION BY PASSWORD = '<UseStrongPasswordHere>';
GO
```

Veritabanını şifrelemeden önce ilk aşamada, master veritabanında bir veritabanı anahtarı diğer bir deyişle Database Master Key (DMK) oluşturmalıyız. Bu master key, sertifikaları ve şifrelemek için kullanılacak anahtarı korumak için ön koruma sağlar.

```
CREATE CERTIFICATE MyServerCert
    WITH SUBJECT = 'My DEK Certificate';
GO
```

İkinci aşama olarak master veritabanı içinde bir sertifika oluşturulur. Bu sertifika, veriyi şifrelemek için kullanılacak anahtarı şifrelemeyi sağlayacak.

```
USE AdventureWorks2022;
GO

CREATE DATABASE ENCRYPTION KEY
    WITH ALGORITHM = AES_256
    ENCRYPTION BY SERVER CERTIFICATE MyServerCert;
GO
```

|

Şifreleme yapmak istediğimiz database için, bir veritabanı şifreleme anahtarı oluşturur. AES'in 128, 192, 256 bitlik şifreleme seçeneklerinden en güçlü olan 256 bitlik olanı seçiyoruz ve öncesinde oluşturduğumuz sertifika ile bu anahtarı birbirine bağlıyoruz. Böylece veritabanı şifreleme anahtarı (DEK) sertifika ile korunmuş oluyor.

```
ALTER DATABASE AdventureWorks2022
    SET ENCRYPTION ON;
GO
```

Oluşturduğumuz DEK ile veritabanını şifreliyoruz.

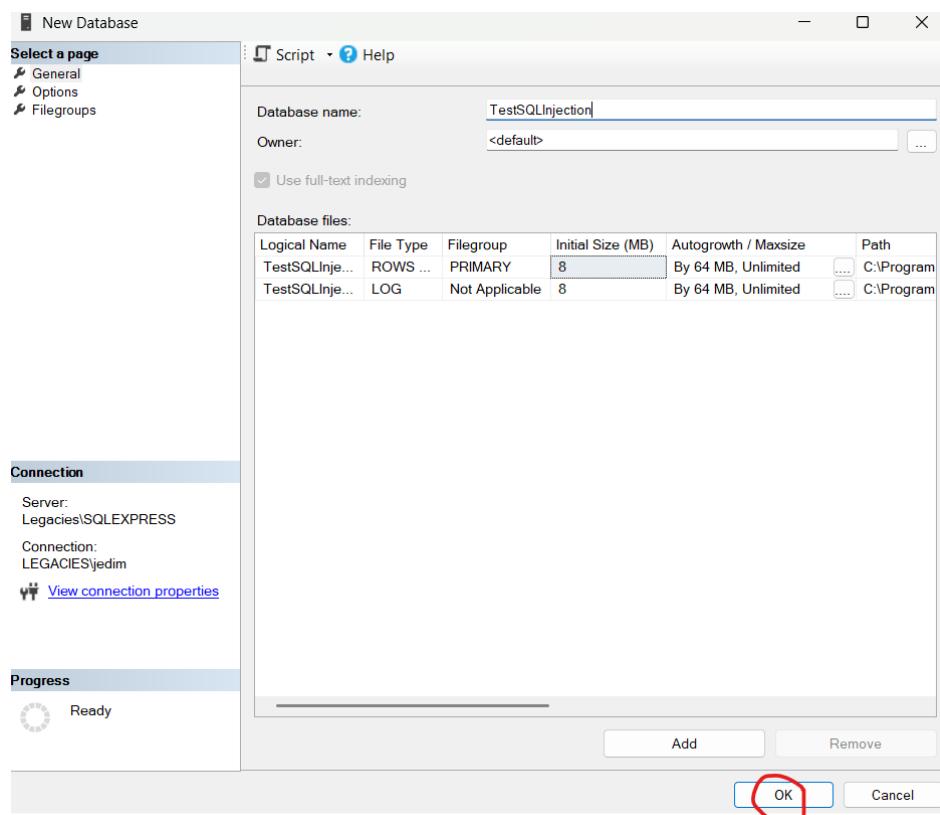
```
/* https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/enable-database-encryption
USE AdventureWorks2022;
GO

/* The value 3 represents an encrypted state
on the database and transaction logs. */
SELECT *
FROM sys.dm_database_encryption_keys
WHERE encryption_state = 3;|
```

Şifreleme durumunu görmek için microsoft'un sql server encryption web sitesindeki sql komutlarını kullanarak veritabanının doğru bir şekilde şifrelenip şifrelenmediğini gözlemliyoruz.

c. SQL Injection Testleri:

Normal şartlarda select komutu dışındaki diğer komutlar ile uygulanabilecek sql injection saldırısına karşı herhangi bir koruması yoktur. Select komutu dışındaki komutları ise constraint ekleyerek engelleyebilmektedir. Select kısmından gelecek bir saldırının MSSQL veritabanı düzeyinde, dışarıdan log kayıtlarını izleyip müdahale edilmediği sürece otomatik olarak engellenebilmesi pek mümkün değildir. Bu yüzden, sql injection saldırısına karşı client bazlı bir koruma yapılması gerekmektedir. Normal sorgular yerine parametreli sorgular kullanarak saldırının, query'lerin açıklarını kullanmasının önüne geçilir. SQL injection kısmı için bir test ortamı oluşturulmuştur. Yeni bir veritabanı oluşturulup içerisine mock user data'ları (yapay kullanıcı verileri) eklenmiştir. Flask üzerinden bir client hazırlanmış olup parametresiz ve parametreli sorgulardaki farklar ele alınmıştır.



TestSQLInjection isminde bir veritabanı oluşturuldu.

SQLQuery2.sql - LE...GACIES\jedim (78)* X SQLQuery1.sql - LE...GACIES\jedim (73)*

```
CREATE TABLE Users (
    UserID INT IDENTITY PRIMARY KEY,
    Username NVARCHAR(50) NOT NULL,
    Password NVARCHAR(50) NOT NULL,
    Email NVARCHAR(100),
    Role NVARCHAR(20),
);

INSERT INTO Users (Username, Password, Email, Role)
VALUES
('admin', 'admin123', 'admin@example.com', 'admin'),
('user1', 'userpass1', 'user1@example.com', 'user'),
('user2', 'userpass2', 'user2@example.com', 'user'),
('user3', 'userpass3', 'user3@example.com', 'user'),
('guest', 'guestpass', 'guest@example.com', 'user'),
('sysadmin', 'sys123', 'sys@example.com', 'admin');
```

100 %

Messages

(6 rows affected)

Completion time: 2025-04-23T14:34:04.4565588+03:00

Örnek bir user tablosu ve mock user verileri oluşturuldu. Password bilgileri, bilerek hashlenmemiş bir şekilde tutuldu. Best case'de kesinlikle password'ları veritabanlarında hashlenmiş bir şekilde tutulması gerekmektedir ama saldırgaların genellikle password yerine veritabanındaki hassas bilgilere erişmek istediginden ötürü bu şekilde test ortamı oluşturuldu.

SQLQuery4.sql - LE...GACIES\jedim (79)* X SQLQuery1.sql - LE

```
SELECT * FROM Users;
```

100 %

Results Messages

	UserID	Username	Password	Email	Role
1	1	admin	admin123	admin@example.com	admin
2	2	user1	userpass1	user1@example.com	user
3	3	user2	userpass2	user2@example.com	user
4	4	user3	userpass3	user3@example.com	user
5	5	guest	guestpass	guest@example.com	user
6	6	sysadmin	sys123	sys@example.com	admin

Tablonun başarılı bir şekilde oluşturulup oluşturulmadığını test etmek için basit bir sorgu yazıldı.

Parametresiz Query Senaryosu:

```
8     app = Flask(__name__)
9
10    # Veritabanı bağlantısı
11    connection_string = (
12        "Driver={ODBC Driver 17 for SQL Server};"
13        "Server=LEGACIES\SQLEXPRESS;"
14        "Database=TestSQLInjection;"
15        "Trusted_Connection=yes;"
16    )
17
18    @app.route('/', methods=['GET', 'POST'])
19    def login():
20        result = ''
21        if request.method == 'POST':
22            username = request.form['username']
23            password = request.form['password']
24            query = f'SELECT * FROM Users WHERE Username = \'{username}\' AND Password = \'{password}\''
25
26            try:
27                conn = pyodbc.connect(connection_string)
28                cursor = conn.cursor()
29                cursor.execute(query)
30                rows = cursor.fetchall()
31                if rows:
32                    result = f'Giriş başarılı! Kullanıcı: {rows}'
```

Problems Output Debug Console **Terminal** Ports Postman Console

* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000

Burada flask projesi yer almaktadır. Connection string değişkeni, ilgili veritabanına erişmek için yazıldı. Query değişkeninde ise parametresiz bir sorgu yazılmıştır. Client, 5000 portunda başlatıldı.

Login

Username:

Password:

Giriş başarılı! Kullanıcı: [(1, 'admin', 'admin123', 'admin@example.com', 'admin'), (2, 'user1', 'userpass1', 'user1@example.com', 'user'), (3, 'user2', 'userpass2', 'user2@example.com', 'user'), (4, 'user3', 'userpass3', 'user3@example.com', 'user'), (5, 'guest', 'guestpass', 'guest@example.com', 'user'), (6, 'sysadmin', 'sys123', 'sys@example.com', 'admin')]

Gördüğü üzere saldırgan username'e “- OR 1=1” yazıp password kısmını boş bıraktığı zaman, client'teki parametresiz query'de şöyle gözükmektedir:

“SELECT * FROM WHERE Username = ‘’ OR 1=1 -- ‘ AND Password”

Sql'de comment line – işaretü ile atıldığı için “ ‘ AND Password “ kısmı comment line içerisinde alınmış olur. Böylece aslında tüm kullanıcı bilgileri getirilmiş olunur.

Bunu önlemenin birkaç basit yolu var. Username ve password data class'ında validation annotation kullanarak bu durum rahatlıkla çözülür. Bunlara literatürde middleware denir. Bir endpoint'e atılan post isteğinden gelen veriler, veritabanında işlem yapmayı sağlamadan önce middleware katmanı ile validate edilir. Diğer bir yöntem ise parametreli yöntem kullanmaktadır. Bu sayede, bu ve bunun gibi komutlarla sql syntax'ını manipüle edilemez durumda olur.

Parametreli Query Senaryosu:

```
@app.route('/', methods=['GET', 'POST'])
def login():
    result = ''
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        #query = f"SELECT * FROM Users WHERE Username = '{username}' AND Password = '{password}'"
        query = "SELECT * FROM Users WHERE Username = ? AND Password = ?"
        try:
            conn = pyodbc.connect(connection_string)
            cursor = conn.cursor()
            cursor.execute(query, (username, password))
            rows = cursor.fetchall()
            if rows:
                result = f'Giriş başarılı! Kullanıcı: {rows}'
            else:
                result = 'Hatalı giriş!'
        except Exception as e:
            result = f'Hata oluştu: {e}'

    return render_template_string("""
        <h2>Login</h2>
    
```

Burada ise parametreli bir query ile sql injection'a karşı bir koruma sağlandı.

Login

Username:

Password:

Hatalı giriş!

Veritabanı düzeyinde sql injection komut türlerini Insert, Delete ve Update gibi işlemler için bloklamak için veritabanı için constraint açılabilir:

The screenshot shows the SSMS interface with two tabs: 'SQLQuery5.sql' and 'SQLQuery4.sql'. The code in 'SQLQuery5.sql' is as follows:

```
USE TestSQLInjection
ALTER TABLE Users
ADD CONSTRAINT CK_Username_NoInjection
CHECK (
    Username NOT LIKE '%--%'
    Username NOT LIKE '%''%
    Username NOT LIKE '%;%'
    Username NOT LIKE '% OR %'
);
```

The 'Messages' pane at the bottom shows the output:

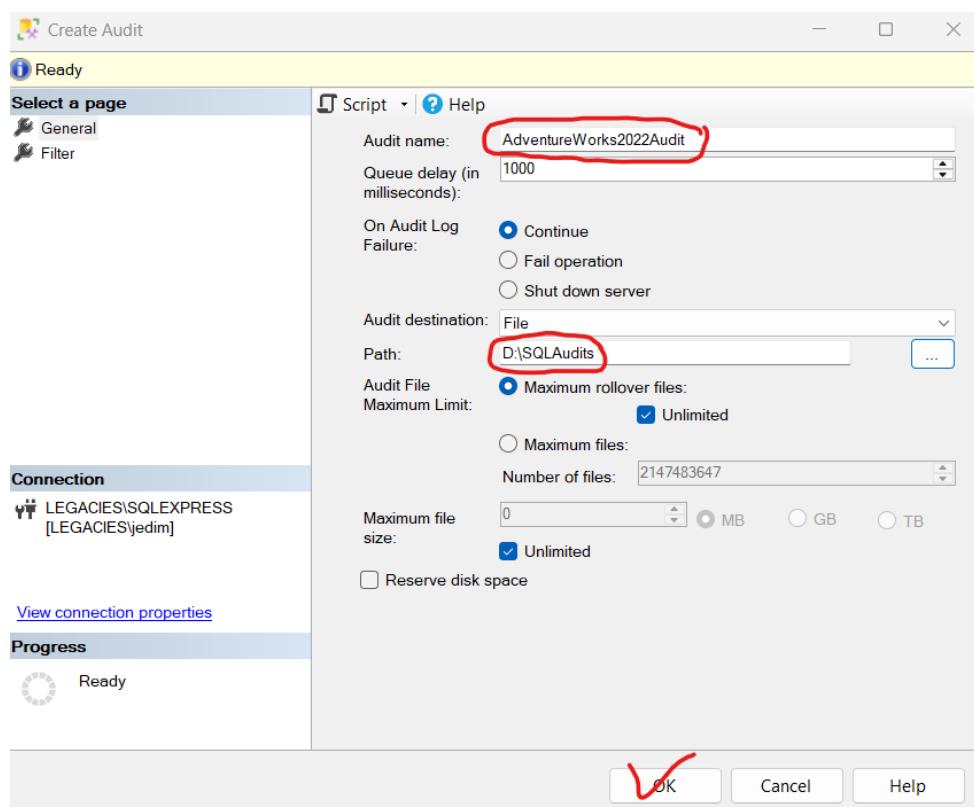
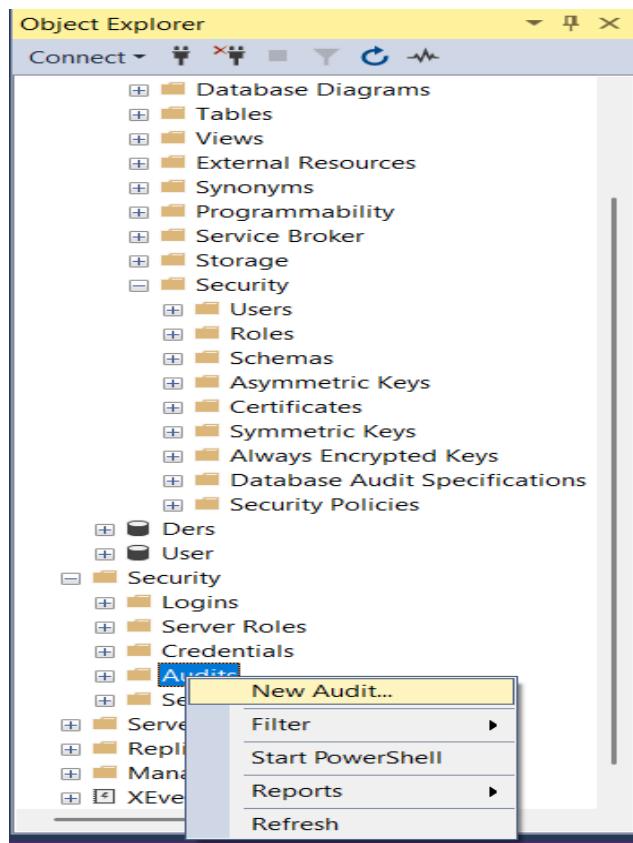
```
Commands completed successfully.
```

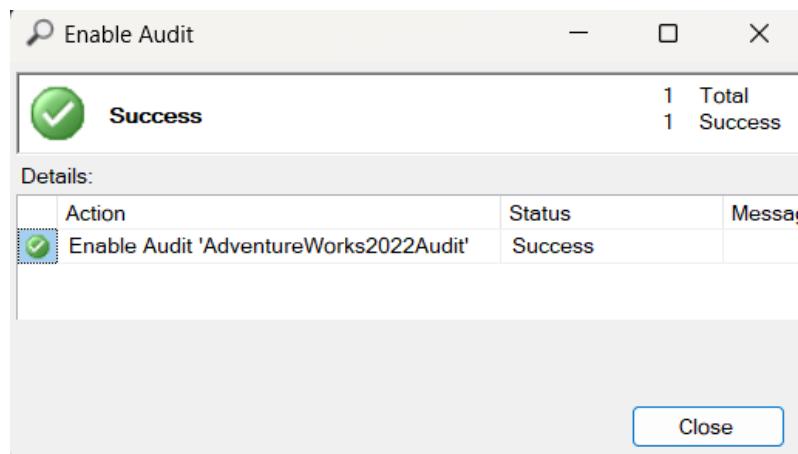
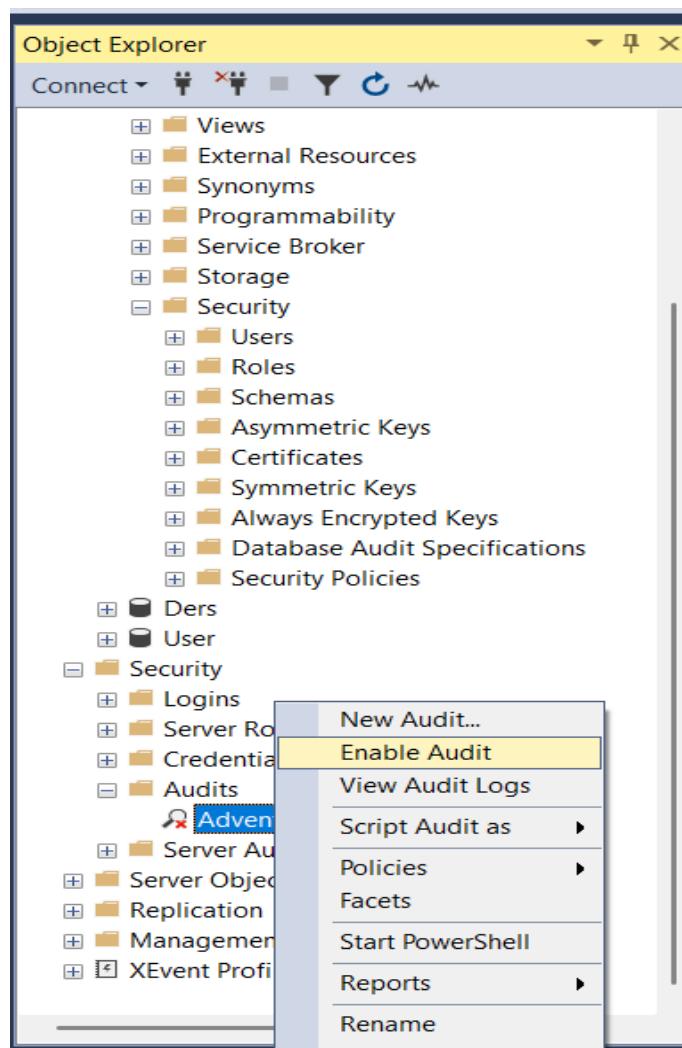
Completion time: 2025-04-23T15:54:15.5000453+03:00

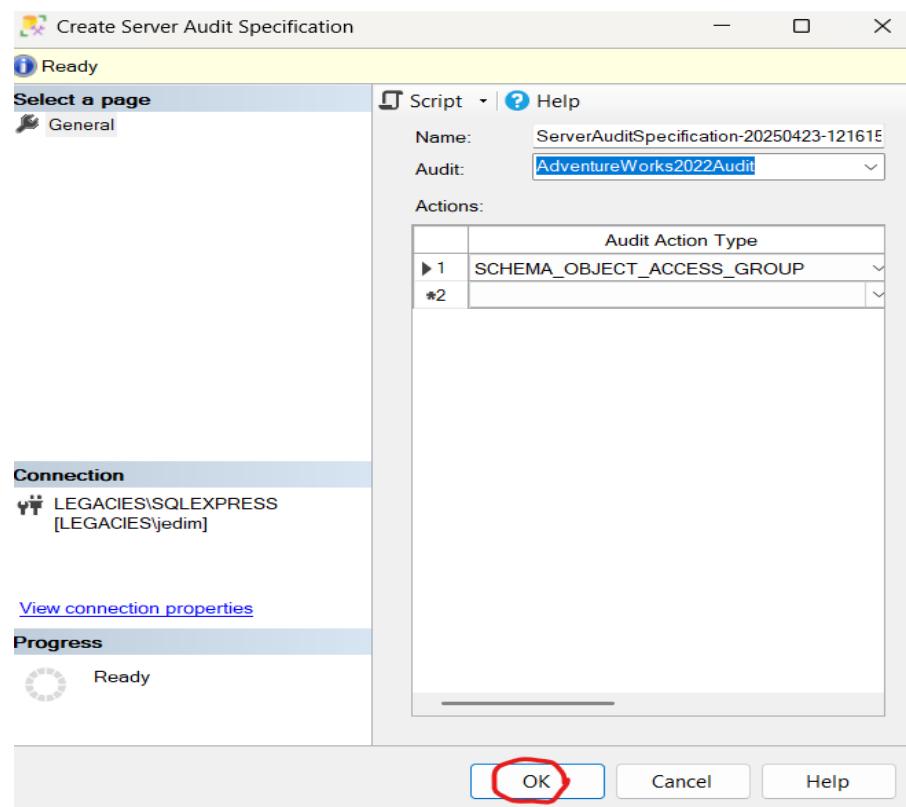
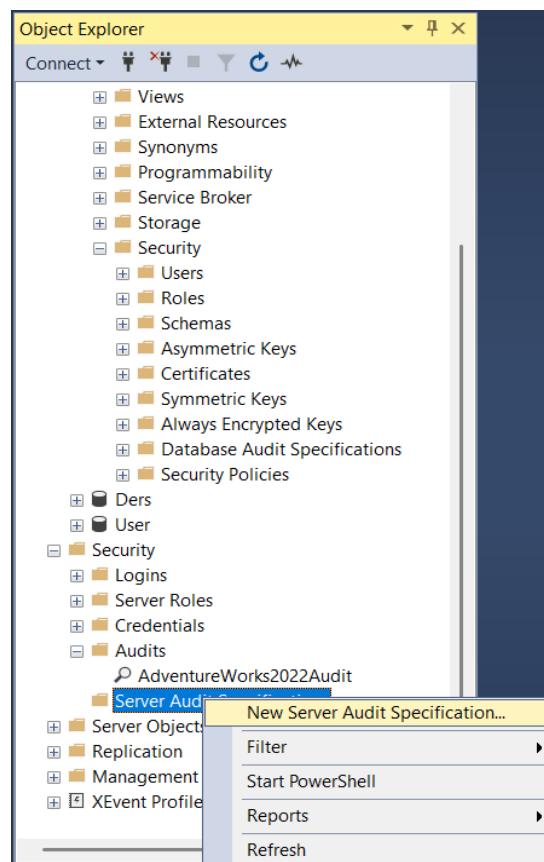
d. Audit Logları:

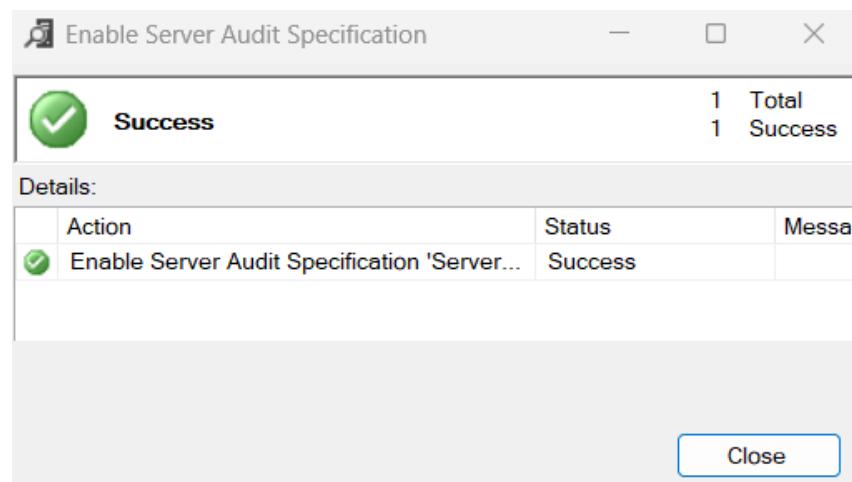
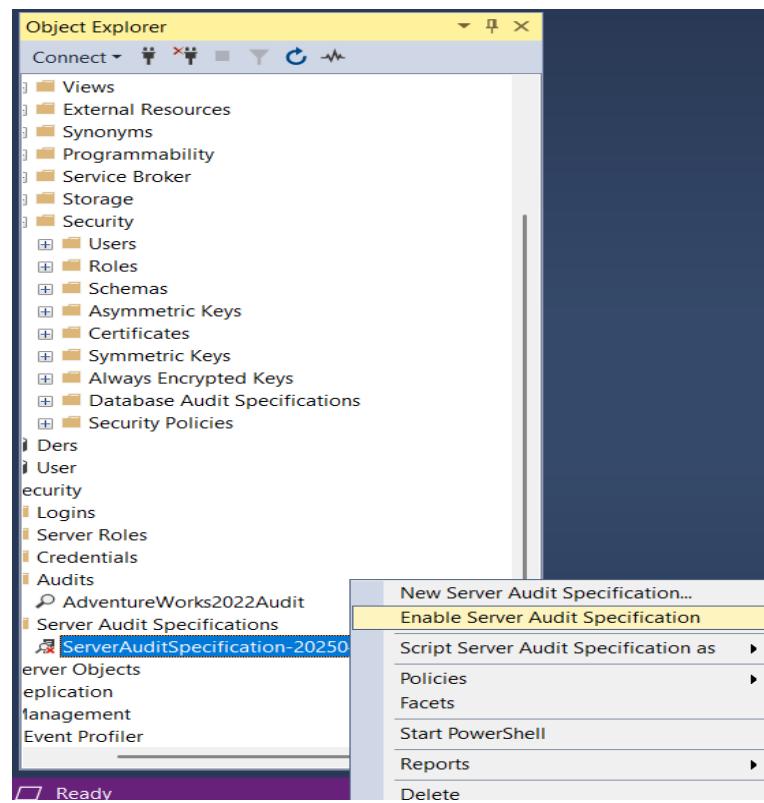
Audit logları ssms'teki kullanıcıların veritabanı/veritabanları üzerindeki aktivitesini gözlemlmek için kullanılır. Ssms'teki bazı audit log bilgileri,

Column Name	Description
Event time	Olayın gerçekleşme saati
Sequence number	Olay sırası
Permission bitmask	Yapılan işlemin izin tipi
Is column permission	Column seviyesinde mi işlem izni verildi?
Database principal id	Sorgu hangi yetki id'si ile veritabanında yapıldı?
Server principal id	Bağlantıyı başlatanın id'si









SQLQuery1.sql - LE..GACIES\jedim (73)*

```

SELECT *
FROM sys.fn_get_audit_file('D:\SQLAudits\AdventureWorks2022Audit_9290E345-40DC-491E-BAAE-E5D8D259E754_0_133898732858930000.sqlaudit',
    NULL,
    NULL);

```

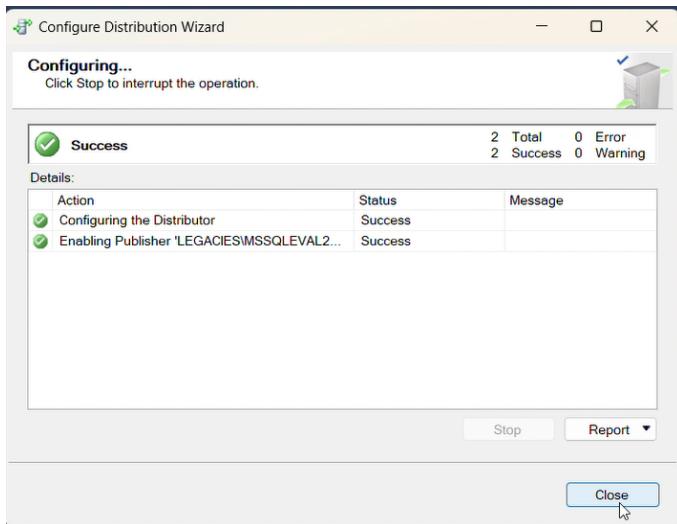
100 %

Results Messages

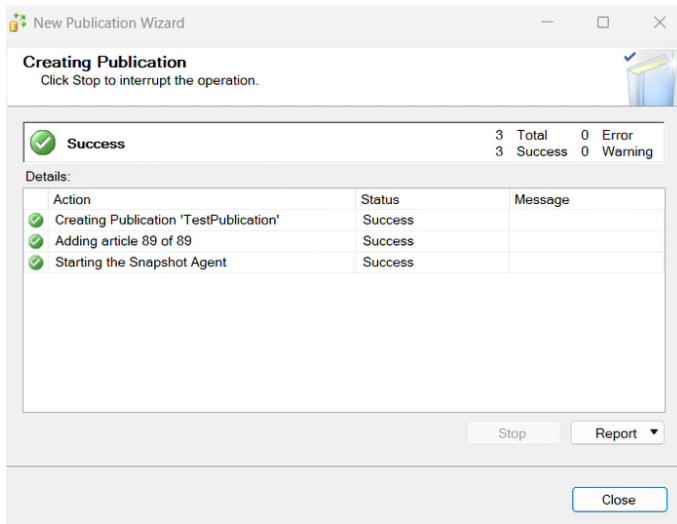
	event_time	sequence_number	action_id	succeeded	permission_bitmask	is_column_permission	session_id	server_principal_id	database_principal_id	target_server_principal_id
1	2025-04-23 09:14:45.9992023	1	AUSC	1	0x00	0	74	259	0	0
2	2025-04-23 09:19:02.1561275	1	SL	1	0x0001	1	78	259	1	0
3	2025-04-23 09:19:02.1561275	1	SL	1	0x0001	1	78	259	1	0
4	2025-04-23 09:19:02.1561275	1	SL	1	0x0001	1	78	259	1	0
5	2025-04-23 09:19:02.1561275	1	SL	1	0x0001	1	78	259	1	0
6	2025-04-23 09:19:02.1561275	1	SL	1	0x0001	1	78	259	1	0
7	2025-04-23 09:19:02.1561275	1	SL	1	0x0001	1	78	259	1	0
8	2025-04-23 09:21:50.3406771	1	SL	1	0x0001	1	59	259	1	0
9	2025-04-23 09:21:50.3497835	1	SL	1	0x0001	1	67	259	1	0
10	2025-04-23 09:22:46.3905517	1	SL	1	0x0001	1	71	259	1	0
11	2025-04-23 09:22:46.4000953	1	SL	1	0x0001	1	72	259	1	0
12	2025-04-23 09:22:48.5312250	1	SL	1	0x0001	1	73	259	1	0
13	2025-04-23 09:22:48.5475683	1	SL	1	0x0001	1	73	259	1	0
14	2025-04-23 09:22:48.5530863	1	EX	1	0x0020	0	73	259	1	0

4. Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları

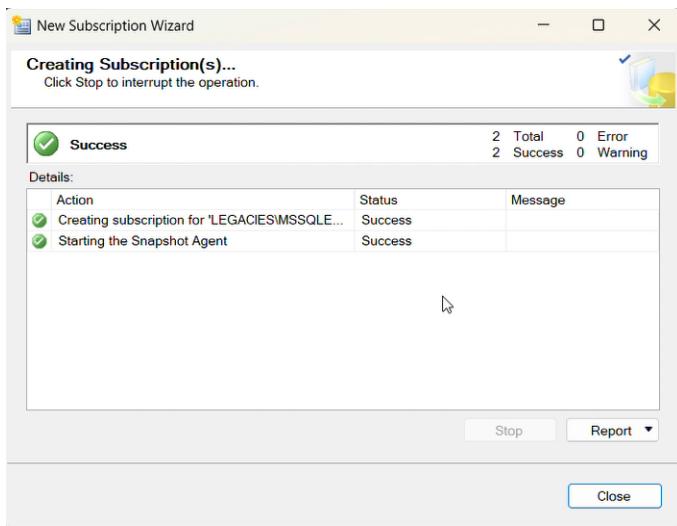
a. Veritabanı Replikasyonu:



b. Yük Dengeleme:



c. Failover Senaryoları:



5. Veri Temizleme ve ETL Süreçleri Tasarımı

a. Veri Temizleme:

Eksik, hatalı veya uygunsuz veriler tespit edip düzeltildi.

```
SELECT BusinessEntityID, FirstName, LastName
FROM Person.Person
WHERE FirstName IS NULL OR FirstName = ''
    OR LastName IS NULL OR LastName = '';

UPDATE Person.Person
SET FirstName = 'Unknown'
WHERE FirstName IS NULL OR FirstName = '';

UPDATE Person.Person
SET LastName = 'Unknown'
WHERE LastName IS NULL OR LastName = '';

SELECT BusinessEntityID, EmailPromotion
FROM Person.Person
WHERE EmailPromotion NOT IN (0, 1, 2);

UPDATE Person.Person
SET EmailPromotion = 0
WHERE EmailPromotion NOT IN (0, 1, 2);
```

b. Veri Dönüştürme:

Verileri daha okunabilir ve tutarlı hale getirildi.

```
UPDATE Person.Person
SET FirstName = UPPER(LEFT(FirstName, 1)) + LOWER(SUBSTRING(FirstName, 2, LEN(FirstName) - 1));
UPDATE Person.Person
SET LastName = UPPER(LEFT(LastName, 1)) + LOWER(SUBSTRING(LastName, 2, LEN(LastName) - 1));

SELECT BusinessEntityID, FirstName, LastName, EmailPromotion
FROM Person.Person
ORDER BY BusinessEntityID;
```

c. Veri Yükleme:

Hedef tablo oluşturuldu ve veriler yüklandı.

```
IF OBJECT_ID('Person.Person_Cleaned', 'U') IS NOT NULL
    DROP TABLE Person.Person_Cleaned;

CREATE TABLE Person.Person_Cleaned (
    BusinessEntityID INT PRIMARY KEY,
    FirstName NVARCHAR(50),
    LastName NVARCHAR(50),
    EmailPromotion INT
);

INSERT INTO Person.Person_Cleaned (BusinessEntityID, FirstName, LastName, EmailPromotion)
SELECT BusinessEntityID, FirstName, LastName, EmailPromotion
FROM Person.Person;
```

d. Veri Kalitesi Raporları:

```
SELECT
    SUM(CASE WHEN FirstName = 'Unknown' THEN 1 ELSE 0 END) AS UnknownFirstNames,
    SUM(CASE WHEN LastName = 'Unknown' THEN 1 ELSE 0 END) AS UnknownLastNames
FROM Person.Person;
```

```
SQLQuery17.sql - M...SI\Özge Çelik (76)*      SQLQuery16.sql - M...SI\Özge Çelik (75))*
SELECT EmailPromotion, COUNT(*) AS Count
FROM Person.Person
GROUP BY EmailPromotion
ORDER BY EmailPromotion;
```

100 %

Results Messages

	EmailPromotion	Count
1	0	11158
2	1	5044
3	2	3770

```
SQLQuery17.sql - M...SI\Özge Çelik (76)*      SQLQuery18.sql - M...SI\Özge Çelik (76))
SELECT COUNT(*) AS CleanedRecordCount
FROM Person.Person_Cleaned;
```

100 %

Results Messages

	CleanedRecordCount
1	19972

6. Veritabanı Yükseltme ve Sürüm Yönetimi

a. Veritabanı Yükseltme Planı:

```
SELECT name, compatibility_level
FROM sys.databases
WHERE name = 'AdventureWorks2019';
```

1 %

Results Messages

name	compatibility_level
AdventureWorks2019	150

Geçerli uyumluluk seviyesi kontrol edildi.

```
BACKUP DATABASE AdventureWorks2019  
TO DISK = 'C:\Users\Özge\Documents\AdventureWorks2019_PreUpgrade.bak'  
WITH INIT, COMPRESSION;
```

Yedek alındı.

```
ALTER DATABASE AdventureWorks2019  
SET COMPATIBILITY_LEVEL = 160;
```

Uyumluluk seviyesi güncellendi.

b. Sürüm Yönetimi:

```
CREATE TABLE dbo.SchemaChangeLog (  
    ID INT IDENTITY PRIMARY KEY,  
    EventType NVARCHAR(100),  
    EventDDL NVARCHAR(MAX),  
    EventTime DATETIME,  
    ChangedBy NVARCHAR(100)  
)
```

Log tablosu oluşturuldu.

```
CREATE TRIGGER trg_DDL_Change_Log  
ON DATABASE  
FOR DDL_DATABASE_LEVEL_EVENTS  
AS  
BEGIN  
    INSERT INTO dbo.SchemaChangeLog(EventType, EventDDL, EventTime, ChangedBy)  
    SELECT EVENTDATA().value('/EVENT_INSTANCE/EventType[1]', 'NVARCHAR(100)'),  
           EVENTDATA().value('/EVENT_INSTANCE/TSQLCommand/CommandText[1]', 'NVARCHAR(MAX)'),  
           GETDATE(),  
           SYSTEM_USER;  
END;
```

DDL Trigger oluşturuldu.

```
ALTER TABLE Person.Person ADD InstagramHandle NVARCHAR(100);
```

```
SELECT * FROM dbo.SchemaChangeLog ORDER BY EventTime DESC;
```

Results Messages					
ID	EventType	EventDDL	EventTime	ChangedBy	
1	ALTER_TABLE	ALTER TABLE Person.Person ADD InstagramHandle NVARCHAR(400);	2025-05-21 16:07:05.923	MSI\Özge Çelik	

Test edildi.

c. Test ve Geri Dönüş Planı:

```
EXEC dbo.uspGetEmployeeManagers @BusinessEntityID = 1;
```

Stored procedure, view ve function'lar test edildi.

```
SELECT COUNT(*) FROM HumanResources.Employee;
```

Results Messages	
(No column name)	
290	

Veri bütünlüğü control edildi.

```
SELECT * FROM sys.dm_db_missing_index_details;
```

Execution plan'ları izlendi. Sorguların yavaşladığı yerlerde indeks önerisi alındı.

```
RESTORE DATABASE AdventureWorks2019
FROM DISK = 'C:\Backup\AdventureWorks2019_PreUpgrade.bak'
WITH REPLACE;
```

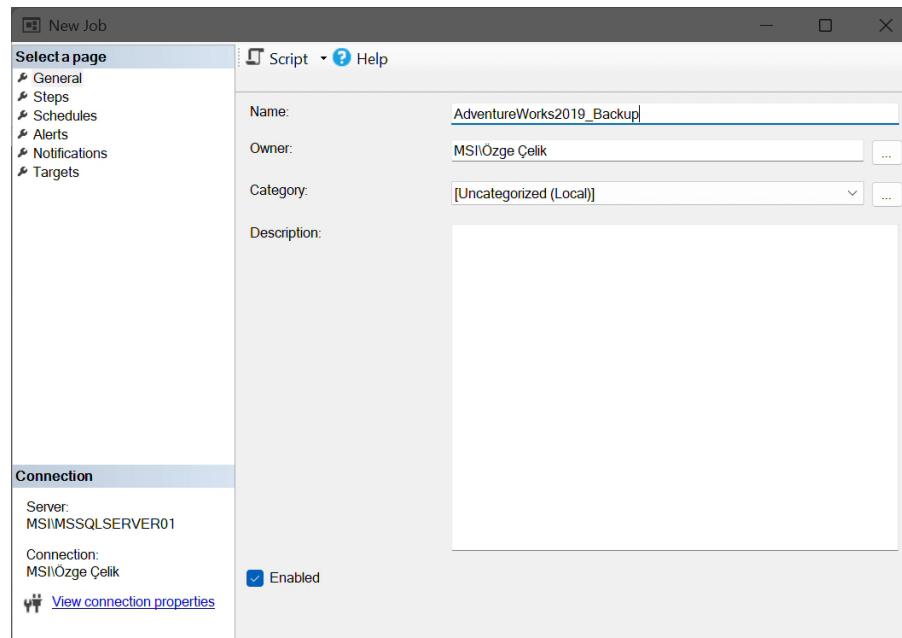
Önceden alınan yedek geri yüklendi.

```
DROP TRIGGER trg_DDL_Change_Log ON DATABASE;
```

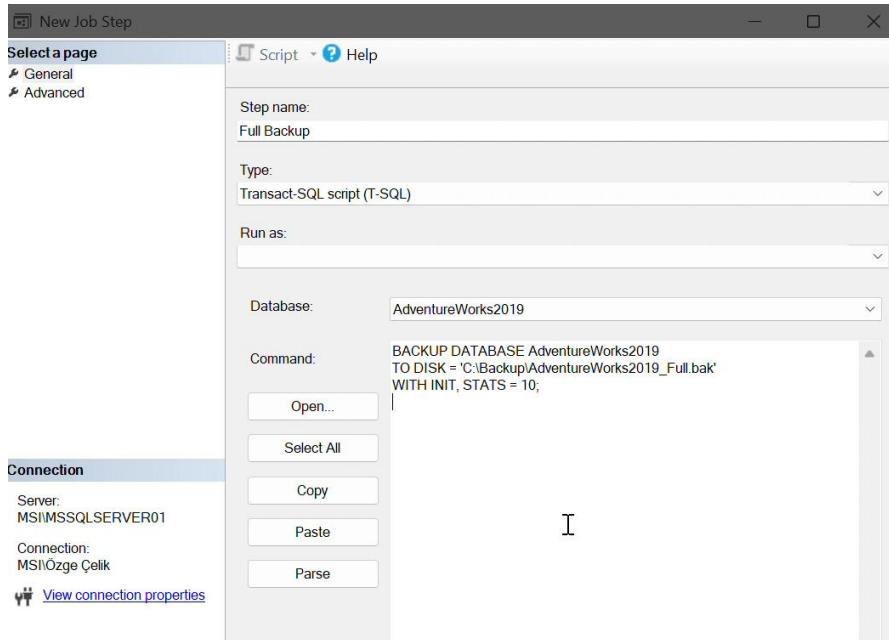
DDL trigger silindi.

7. Veritabanı Yedekleme ve Otomasyon Çalışması

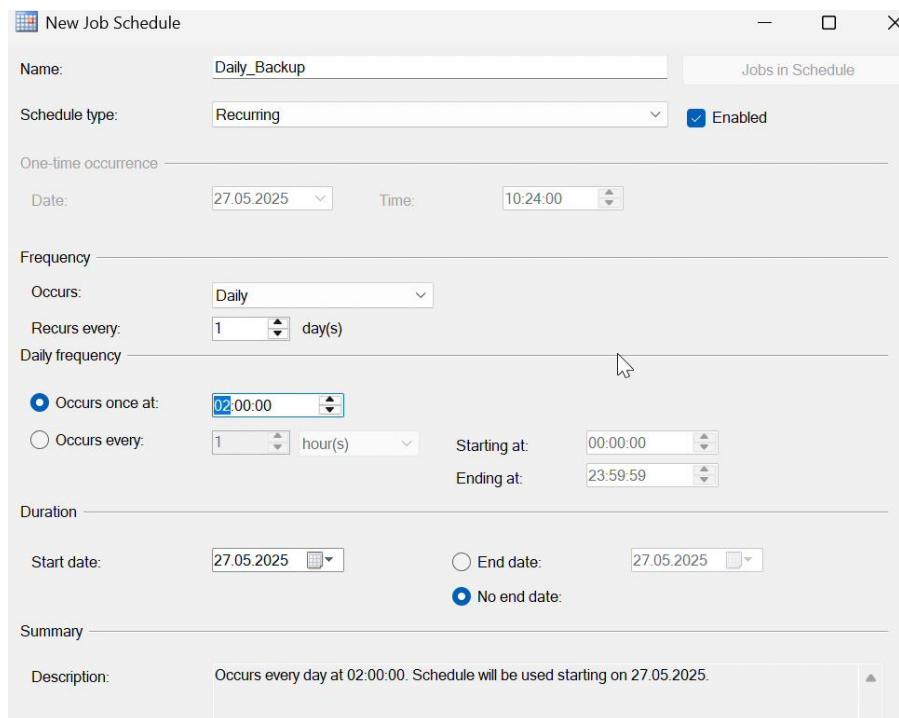
a. Yedekleme Süreçlerini Otomatikleştirme:



Yeni bir job oluşturuldu.

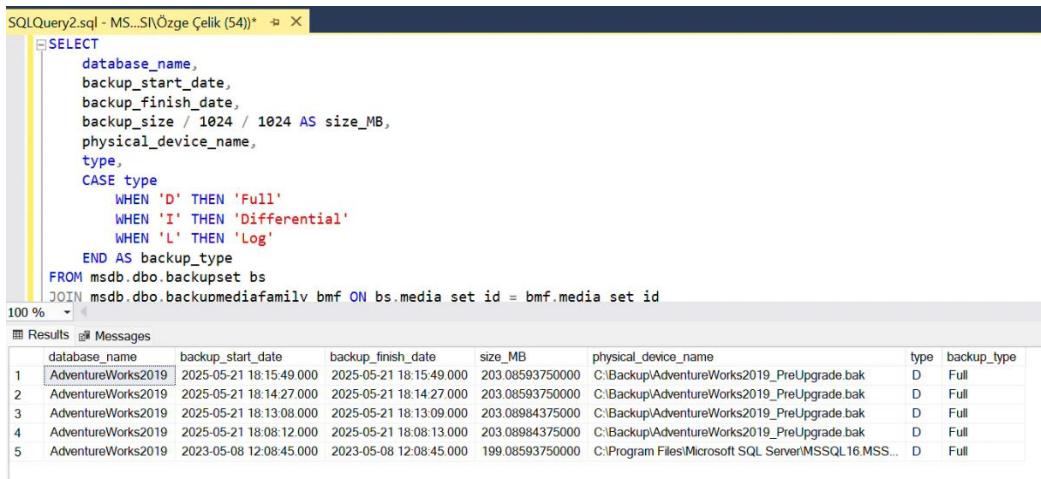


Job'a step eklendi.



Zamanlama oluşturuldu.

b. Yedekleme Raporları Oluşturma:



SQLQuery2.sql - MS...SI\Özge Çelik (54)*

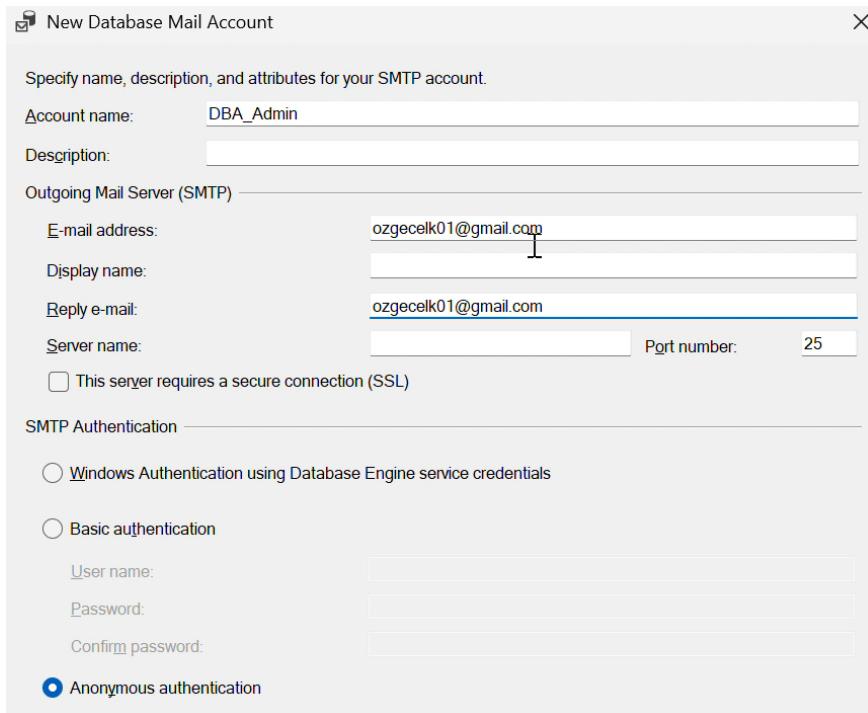
```
SELECT
    database_name,
    backup_start_date,
    backup_finish_date,
    backup_size / 1024 / 1024 AS size_MB,
    physical_device_name,
    type,
    CASE type
        WHEN 'D' THEN 'Full'
        WHEN 'I' THEN 'Differential'
        WHEN 'L' THEN 'Log'
    END AS backup_type
FROM msdb.dbo.backupset bs
JOIN msdb.dbo.backupmediafamily bmf ON bs.media set id = bmf.media set id
```

100 %

Results Messages

	database_name	backup_start_date	backup_finish_date	size_MB	physical_device_name	type	backup_type
1	AdventureWorks2019	2025-05-21 18:15:49.000	2025-05-21 18:15:49.000	203.08593750000	C:\Backup\AdventureWorks2019_PreUpgrade.bak	D	Full
2	AdventureWorks2019	2025-05-21 18:14:27.000	2025-05-21 18:14:27.000	203.08593750000	C:\Backup\AdventureWorks2019_PreUpgrade.bak	D	Full
3	AdventureWorks2019	2025-05-21 18:13:08.000	2025-05-21 18:13:09.000	203.08984375000	C:\Backup\AdventureWorks2019_PreUpgrade.bak	D	Full
4	AdventureWorks2019	2025-05-21 18:08:12.000	2025-05-21 18:08:13.000	203.08984375000	C:\Backup\AdventureWorks2019_PreUpgrade.bak	D	Full
5	AdventureWorks2019	2023-05-08 12:08:45.000	2023-05-08 12:08:45.000	199.08593750000	C:\Program Files\Microsoft SQL Server\MSSQL16.MSS...	D	Full

c. Otomatik Yedekleme Uyarıları:



Yedekleme başarısız olduğunda yönetici e-posta ile uyarılacak.