

**ANKARA ÜNİVERSİTESİ**  
**MÜHENDİSLİK FAKÜLTESİ**  
**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



**BLM 4522 Ağ Tabanlı Paralel Dağıtım Sistemleri**  
**PROJE ÖDEV RAPORU**

**Seçilen Projeler: 1, 2, 7**

**Hasan Tamer Tefon - 20290295**

**Melisa Melayim – 20290276**

GitHub: <https://github.com/tamertfn/mssqldbProjects>

## PROJE 1

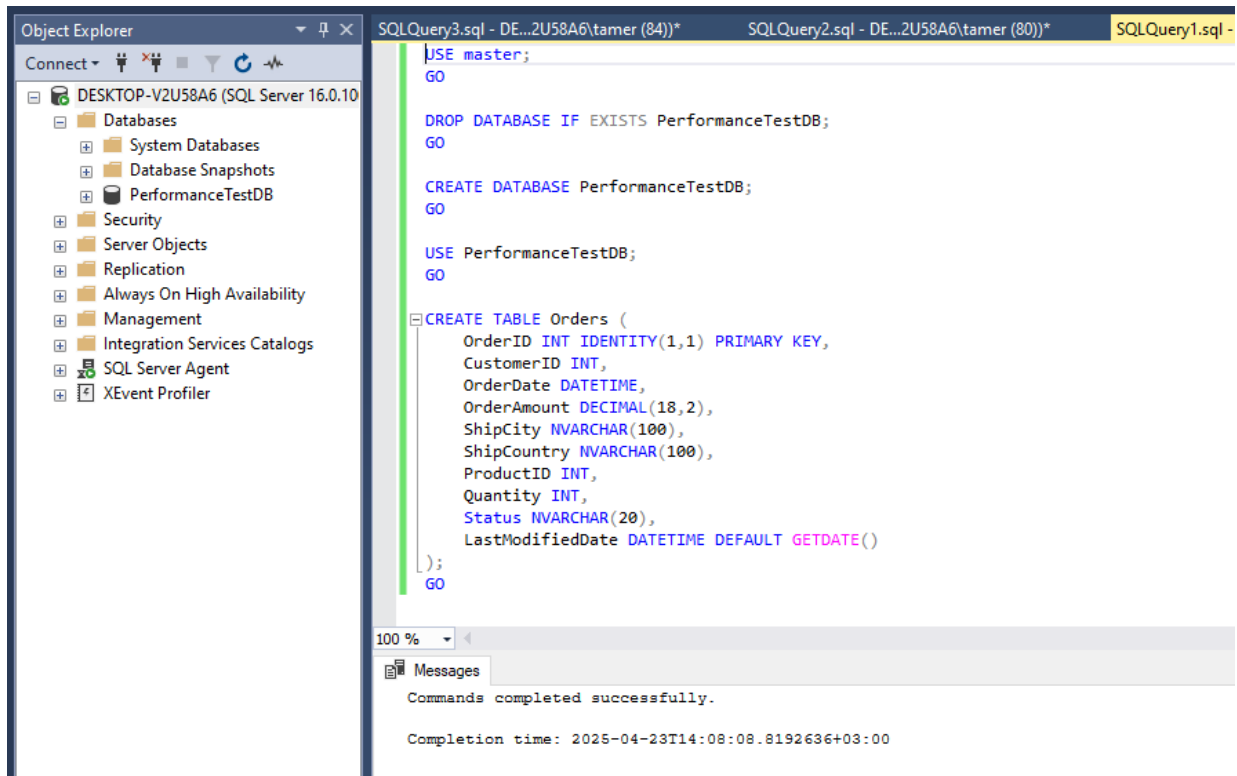
### Veri Tabanı Performans Optimizasyonu ve İzleme Raporu

## GİRİŞ

Bu çalışmada yüksek veri hacmine sahip bir veri tabanı üzerinde performans izleme, sorgu optimizasyonu, indeks yönetimi ve erişim kontrolü konuları ele alınmıştır. Projenin amacı, SQL Server ortamında gerçek hayata yakın bir performans analizi ve müdahale senaryosu oluşturmaktır.

## 1. BAŞLANGIÇ YAPILANDIRMASI

PerformanceTestDB veritabanı oluşturulmuş ve Orders adlı bir tablo tasarlanmıştır. Daha sonra GenerateTestData prosedürüyle tabloya rastgele test verileri yüklenmiş ve 100.000 kayıtlık bir hacim oluşturulmuştur.



Object Explorer

Connect - Desktop-V2USBA6 (SQL Server 16.0.10)

Databases

- System Databases
- Database Snapshots
- PerformanceTestDB
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - dbo.Orders
    - Dropped Ledger Tables
  - Views
  - External Resources
  - Synonyms
  - Programmability
  - Query Store
  - Service Broker
  - Storage
  - Security
- Server Objects
  - Replication
  - Always On High Availability
  - Management
  - Integration Services Catalogs
  - SQL Server Agent
  - XEvent Profiler

SQLQuery1.sql - DE...ZUSBA6\lamer (66)

```
CREATE OR ALTER PROCEDURE GenerateTestData
@RecordCount INT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @Counter INT = 1;
    DECLARE @StatusTypes TABLE (Status NVARCHAR(20));
    INSERT INTO @StatusTypes VALUES ('Pending'), ('Processing'), ('Shipped'), ('Delivered'), ('Cancelled');

    WHILE @Counter <= @RecordCount
    BEGIN
        INSERT INTO Orders (
            CustomerID,
            OrderDate,
            OrderAmount,
            ShipCity,
            ShipCountry,
            ProductID,
            Quantity,
            Status
        )
        SELECT
            ABS(CHECKSUM(NEWID())) % 10000, -- Random CustomerID
            DATETIME2(OFFSET, ABS(CHECKSUM(NEWID())) % 10000, GETDATE()), -- Random Date
            ROUND(RAND() * 100000, 2), -- Random Amount
            'City ' + CAST(ABS(CHECKSUM(NEWID())) % 100 AS VARCHAR), -- Random City
            'Country ' + CAST(ABS(CHECKSUM(NEWID())) % 20 AS VARCHAR), -- Random Country
            ABS(CHECKSUM(NEWID())) % 100, -- Random ProductID
            ABS(CHECKSUM(NEWID())) % 100, -- Random Quantity
            (SELECT TOP 1 Status FROM @StatusTypes ORDER BY NEWID()); -- Random Status

        SET @Counter += 1;

        -- Her 10000 kayıta bir ilerleme mesajı:
        IF @Counter % 10000 = 0
            PRINT 'Inserted ' + CAST(@Counter AS VARCHAR) + ' records';
    END;
END;
```

100 %

Messages

Commands completed successfully.

Completion time: 2025-04-23T14:08:22.6465698+03:00

Object Explorer

Connect - Desktop-V2USBA6 (SQL Server 16.0.10)

Databases

- System Databases
- Database Snapshots
- PerformanceTestDB
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - dbo.Orders
    - Dropped Ledger Tables
  - Views
  - External Resources
  - Synonyms
  - Programmability
  - Query Store
  - Service Broker
  - Storage
  - Security
- Server Objects
  - Replication
  - Always On High Availability
  - Management
  - Integration Services Catalogs
  - SQL Server Agent
  - XEvent Profiler

SQLQuery1.sql - DE...ZUSBA6\lamer (66)

```
SELECT TOP (1000) [OrderID]
, [CustomerID]
, [OrderDate]
, [OrderAmount]
, [ShipCity]
, [ShipCountry]
, [ProductID]
, [Quantity]
, [Status]
, [LastModifiedDate]
FROM [PerformanceTestDB].[dbo].[Orders]
```

100 %

Results

OrderID	CustomerID	OrderDate	OrderAmount	ShipCity	ShipCountry	ProductID	Quantity	Status	LastModifiedDate
10	529	2023-02-16 14:08:29.353	3967.81	City 31	Country 15	65	10	Processing	2025-04-23 14:08:29.353
11	816	2022-10-27 14:08:29.353	1930.78	City 82	Country 8	71	50	Cancelled	2025-04-23 14:08:29.353
12	506	2024-07-10 14:08:29.353	7782.49	City 59	Country 16	33	6	Pending	2025-04-23 14:08:29.353
13	193	2024-07-26 14:08:29.353	7993.95	City 20	Country 2	32	52	Delivered	2025-04-23 14:08:29.353
14	937	2022-09-23 14:08:29.353	1127.91	City 2	Country 6	82	38	Pending	2025-04-23 14:08:29.353
15	366	2025-03-08 14:08:29.353	8137.84	City 19	Country 2	76	15	Delivered	2025-04-23 14:08:29.353
16	813	2022-12-10 14:08:29.353	75.41	City 18	Country 12	33	51	Cancelled	2025-04-23 14:08:29.353
17	497	2024-06-27 14:08:29.353	4101.12	City 41	Country 5	19	42	Pending	2025-04-23 14:08:29.353
18	673	2024-01-14 14:08:29.353	2620.41	City 67	Country 18	37	91	Pending	2025-04-23 14:08:29.353
19	252	2024-01-06 14:08:29.353	2655.68	City 78	Country 14	67	59	Delivered	2025-04-23 14:08:29.353
20	598	2022-08-24 14:08:29.353	1487.13	City 64	Country 4	49	99	Delivered	2025-04-23 14:08:29.353
21	72	2024-12-19 14:08:29.353	3719.86	City 79	Country 14	12	83	Shipped	2025-04-23 14:08:29.353
22	866	2024-08-10 14:08:29.357	8580.37	City 3	Country 9	9	84	Cancelled	2025-04-23 14:08:29.357
23	657	2025-03-25 14:08:29.357	5249.71	City 30	Country 2	26	66	Delivered	2025-04-23 14:08:29.357
24	630	2024-03-02 14:08:29.357	8824.15	City 43	Country 12	21	41	Delivered	2025-04-23 14:08:29.357
25	692	2024-08-09 14:08:29.357	7010.77	City 53	Country 11	4	49	Cancelled	2025-04-23 14:08:29.357
26	356	2024-05-20 14:08:29.357	2870.79	City 39	Country 7	65	12	Shipped	2025-04-23 14:08:29.357
27	533	2023-01-28 14:08:29.357	9212.35	City 68	Country 3	16	85	Cancelled	2025-04-23 14:08:29.357
28	802	2023-11-16 14:08:29.357	9021.36	City 79	Country 7	39	48	Processing	2025-04-23 14:08:29.357
29	714	2023-09-27 14:08:29.357	7387.17	City 47	Country 14	4	25	Cancelled	2025-04-23 14:08:29.357
30	365	2024-05-15 14:08:29.357	2780.34	City 24	Country 19	87	92	Cancelled	2025-04-23 14:08:29.357
31	854	2024-04-15 14:08:29.357	3435.03	City 60	Country 3	76	71	Processing	2025-04-23 14:08:29.357
32	772	2025-04-14 14:08:29.357	6778.72	City 45	Country 5	63	10	Pending	2025-04-23 14:08:29.357
33	296	2025-03-15 14:08:29.357	8078.76	City 7	Country 14	91	64	Delivered	2025-04-23 14:08:29.357
34	372	2022-08-08 14:08:29.357	8871.68	City 50	Country 6	70	74	Delivered	2025-04-23 14:08:29.357

Query executed successfully.

Object Explorer

Connect - Desktop-V2USBA6 (SQL Server 16.0.10)

Databases

- System Databases
- Database Snapshots
- PerformanceTestDB
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - Graph Tables
    - dbo.Orders
    - Dropped Ledger Tables
  - Views
  - External Resources
  - Synonyms
  - Programmability
  - Query Store
  - Service Broker
  - Storage
  - Security
- Server Objects
  - Replication
  - Always On High Availability
  - Management
  - Integration Services Catalogs
  - SQL Server Agent
  - XEvent Profiler

SQLQuery1.sql - DE...ZUSBA6\lamer (66)

```
-- 7. Prosedürü çalıştırarak (100,000 kayıt ekleyelim)
EXEC GenerateTestData 100000;
GO

-- 8. Kontrol edelim
SELECT COUNT(*) as TotalRecords FROM Orders;
```

100 %

Results

TotalRecords
100000

Query executed successfully.

Farklı performans senaryoları için optimize edilmemiş sorgular yazılmış ve ardından indeksleme stratejileriyle bu sorgular iyileştirilmiştir. OrderDate ve ShipCountry üzerinde bileşik indekslerle sorgu yanıt süresi ciddi oranda azaltılmıştır.

```
SQLQuery1.sql - DE_2USAB1tamer (80) | SQLQuery1.sql - DE_2USAB1tamer (56) | x | SQLQuery1.sql - DE_2USAB1tamer (80) | SQLQuery1.sql - DE_2USAB1tamer (80) | SQLQuery1.sql - DE_2USAB1tamer (82) |
-- Tarih aralığında Günceller göre sipariş analizi
SELECT
    ShipCountry,
    COUNT(*) as TotalOrders,
    SUM(OrderAmount) as TotalAmount,
    AVG(OrderAmount) as AvgOrderAmount
FROM Orders
WHERE OrderDate >= DATEADD(MONTH, -6, GETDATE())
GROUP BY ShipCountry
ORDER BY TotalAmount DESC;
```

100 %

Results Messages

SQL Server parse and compile time:  
CPU time = 104 ms, elapsed time = 104 ms.

(20 rows affected)

Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server reads 0, physical reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0, lob page server reads 0, logical reads 1558, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0.

SQL Server execution time:  
CPU time = 16 ms, elapsed time = 18 ms.

Completion time: 2025-04-23T14:19:34.8354069+03:00

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane shows the 'DESKTOP-V2USAA6 SQL Server 16.0.10' instance. The 'Databases' folder is expanded, showing 'System Databases', 'Database Snapshots', 'PerformanceTestDB', 'Database Diagrams', 'Tables', 'System Tables', 'FileTables', 'External Tables', 'Graph Tables', 'dbo.Orders', 'Dropped Ledger Tables', 'Views', 'External Resources', 'Synonyms', 'Programmability', 'Query Store', 'Service Broker', 'Storage', 'Security', 'Server Objects', 'Replication', 'Always On High Availability', 'Management', 'Integration Services Catalogs', 'SQL Server Agent', and 'XE Event Profiler'.

The main window shows a query window titled 'SQLQuery1.sql - DE-V2USAA6\master (80)'. The query text is as follows:

```
--Tarih aralığında ölkelere göre sipariş analizi
--SELECT
--    ShipCountry,
--    COUNT(*) as TotalOrders,
--    SUM(OrderAmount) as TotalAmount,
--    AVG(OrderAmount) as AvgOrderAmount
--FROM Orders
--WHERE OrderDate >= DATETIME('2024-01-01', GETDATE())
--GROUP BY ShipCountry
--ORDER BY TotalAmount DESC;

--Bilistik İndeks oluşturma
CREATE NONCLUSTERED INDEX IX_Orders_OrderDate_ShipCountry
ON Orders(OrderDate, ShipCountry)
INCLUDE (OrderAmount);

UPDATE STATISTICS Orders WITH FULLSCAN;
```

Below the query window, the 'Messages' window shows the execution results:

```
SQL Server parse and compile time:
  CPU time = 4 ms, elapsed time = 4 ms.
Table 'Orders'. Scan count 3, logical reads 1597, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0.
SQL Server Execution time:
  CPU time = 249 ms, elapsed time = 131 ms.
SQL Server Execution Time:
  CPU time = 516 ms, elapsed time = 168 ms.
Completion time: 2024-04-23T14:24:53.2963972+03:00
```

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the server hierarchy under 'DESKTOP-VU5BA6 [SQL Server 16.0]'. The central pane shows a query window with the following T-SQL code:

```
-- Tarifi aralaginda uclerlere gore siparis analizi
SELECT
    ShipCountry,
    COUNT(*) as TotalOrders,
    SUM(OrderAmount) as TotalAmount,
    AVG(OrderAmount) as AvgOrderAmount
FROM Orders
WHERE OrderDate >= DATETIME(2020-04-27T14:27:14.2474872+03:00)
GROUP BY ShipCountry
ORDER BY TotalAmount DESC;
```

Handwritten notes in blue ink are present over the query window:

- Before 16ms*
- Now 0 ms ✓*

The bottom pane shows the 'Results' tab with the following messages:

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

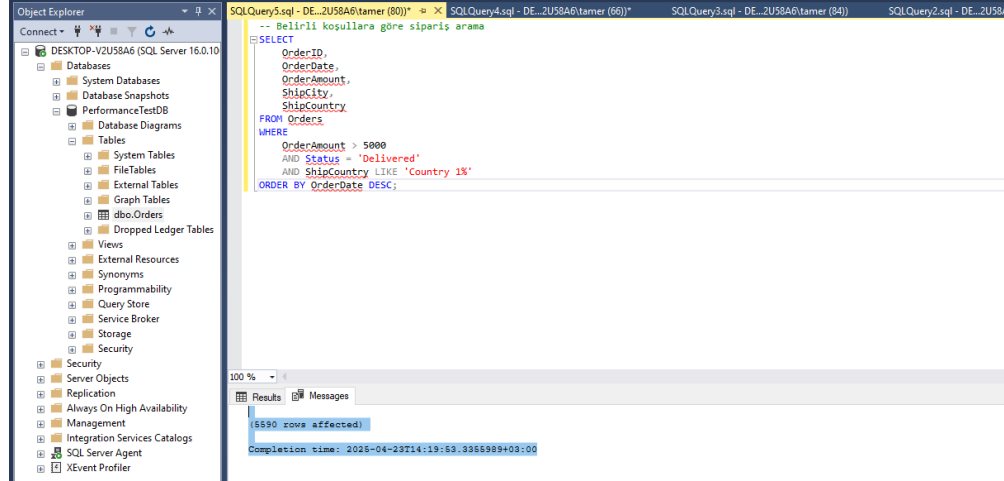
Below the messages, the execution statistics are displayed:

```
(20 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0
Table 'Worktable'. Scan count 1, logical reads 119, physical reads 0, page server reads 0, read-ahead reads 0, page server read-ahead reads 0, lob logical reads 0, lob physical reads 0, lob page server reads 0, lob read-ahead reads 0
```

At the bottom, the 'SQL Server Execution Times:' section shows:

```
CPU time = 0 ms, elapsed time = 12 ms.
Completion time: 2020-04-27T14:27:14.2474872+03:00
```

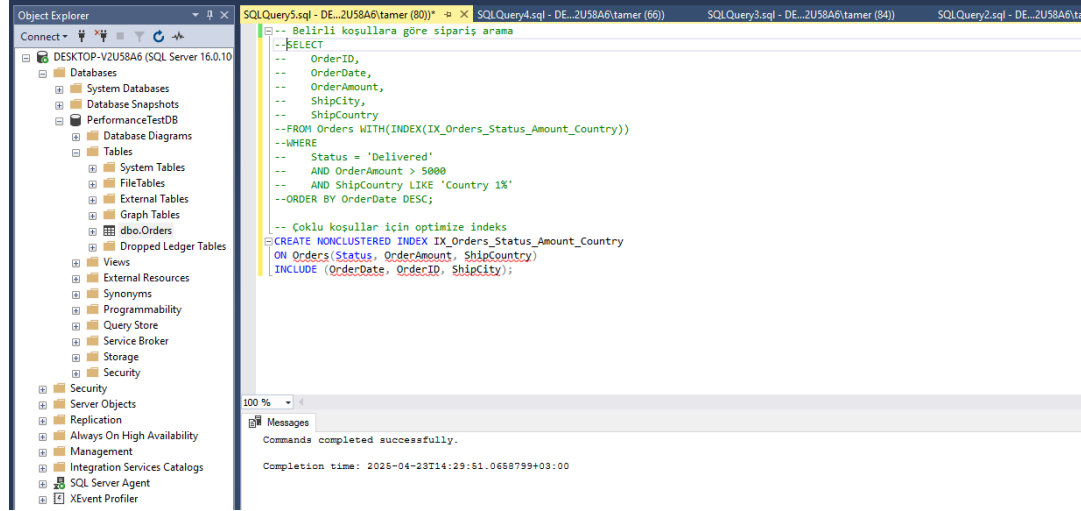
## Optimize edilmemiş ikinci sorgu



```
-- Belirli koşullara göre sipariş arama
SELECT
    OrderID,
    OrderDate,
    OrderAmount,
    ShipCity,
    ShipCountry
FROM Orders
WHERE
    OrderAmount > 5000
    AND Status = 'Delivered'
    AND ShipCountry LIKE 'Country 1%'
ORDER BY OrderDate DESC;
```

Results: 5590 rows affected.  
Completion time: 2025-04-23T14:19:59.3355989+03:00

## Sorgu 2 için indexleme ve optimizasyonun görülmesi



```
-- Belirli koşullara göre sipariş arama
SELECT
    OrderID,
    OrderDate,
    OrderAmount,
    ShipCity,
    ShipCountry
FROM Orders WITH(INDEX(IX_Orders_Status_Amount_Country))
WHERE
    Status = 'Delivered'
    AND OrderAmount > 5000
    AND ShipCountry LIKE 'Country 1%'
ORDER BY OrderDate DESC;

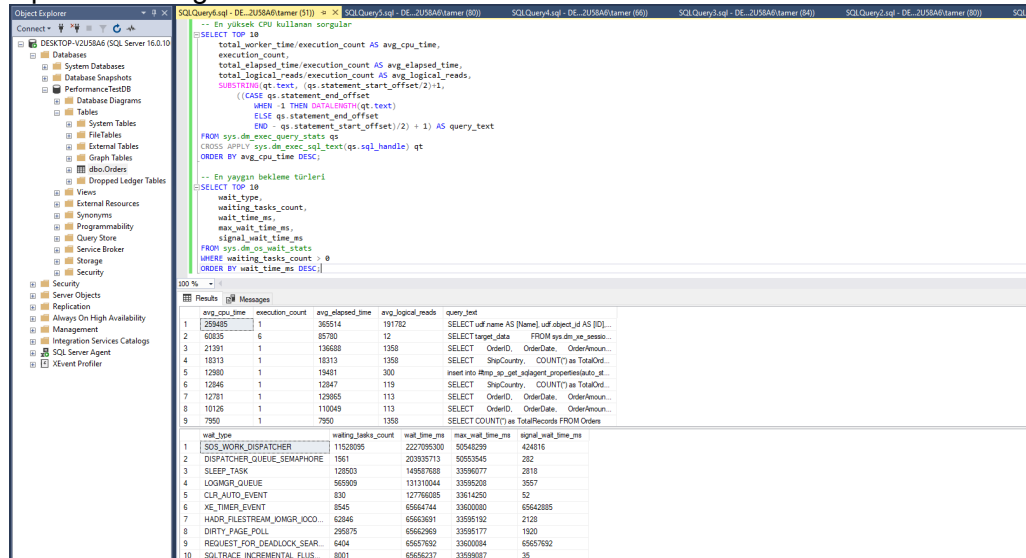
-- Çoklu koşullar için optimize indeks
CREATE NONCLUSTERED INDEX IX_Orders_Status_Amount_Country
ON Orders (Status, OrderAmount, ShipCountry)
INCLUDE (OrderDate, OrderID, ShipCity);
```

Commands completed successfully.  
Completion time: 2025-04-23T14:29:51.0658799+03:00

## 3. İZLEME VE DİAGNOSTİK

SQL Profiler kullanılarak CPU tüketimi yüksek sorgular ve sistem bekleme süreleri analiz görüntülenmiştir. Bu analizler sonucunda performans düşüşleri tespit edilmiştir.

## Cpu ve Waiting Time analizleri



```
-- En yavaş CPU kullanan sorgular
SELECT TOP 10
    total_worker_time/execution_count AS avg_cpu_time,
    execution_count,
    total_elapsed_time/execution_count AS avg_elapsed_time,
    total_logical_reads/execution_count AS avg_logical_reads,
    SUBSTRING(qt.text, (qs.statement_start_offset/2)+1,
        ((CASE qs.statement_end_offset
            WHEN -1 THEN DATALENGTH(qt.text)
            ELSE qs.statement_end_offset
        END - qs.statement_start_offset)/2) + 1) AS query_text
FROM sys.dm_exec_query_stats qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) qt
ORDER BY avg_cpu_time DESC;

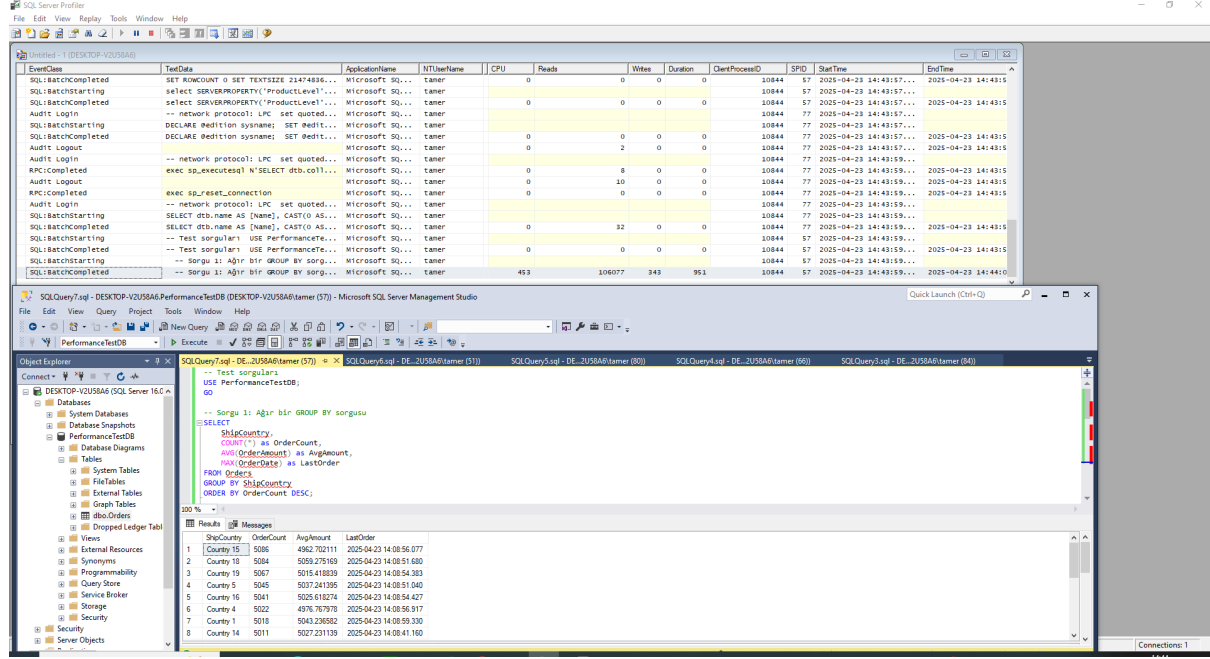
-- En yavaş bekleme türleri
SELECT TOP 10
    wait_type,
    waiting_tasks_count,
    wait_time_ms,
    max_wait_time_ms,
    signal_wait_time_ms
FROM sys.dm_os_wait_stats
WHERE waiting_tasks_count > 0
ORDER BY wait_time_ms DESC;
```

Results: 10 rows affected.

avg_cpu_time	execution_count	avg_elapsed_time	avg_logical_reads	query_text
259435	1	35514	191702	SELECT udf.name AS [Name], udf.object_id AS [ID]...
60835	6	85780	12	SELECT Target_data FROM sys.dm_xe_session...
21381	1	136688	1358	SELECT OrderID, OrderDate, OrderAmount...
18113	1	18113	1288	SELECT ShipCountry, COUNTRY as TotalOrd...
12580	1	19481	300	insert into #temp_sp_get_agent_properties(auto...
12846	1	12847	119	SELECT ShipCountry, COUNTRY as TotalOrd...
12781	1	12865	113	SELECT OrderID, OrderDate, OrderAmount...
10126	1	11049	113	SELECT OrderID, OrderDate, OrderAmount...
7950	1	7950	1358	SELECT COUNT(*) as TotalRecords FROM Orders...

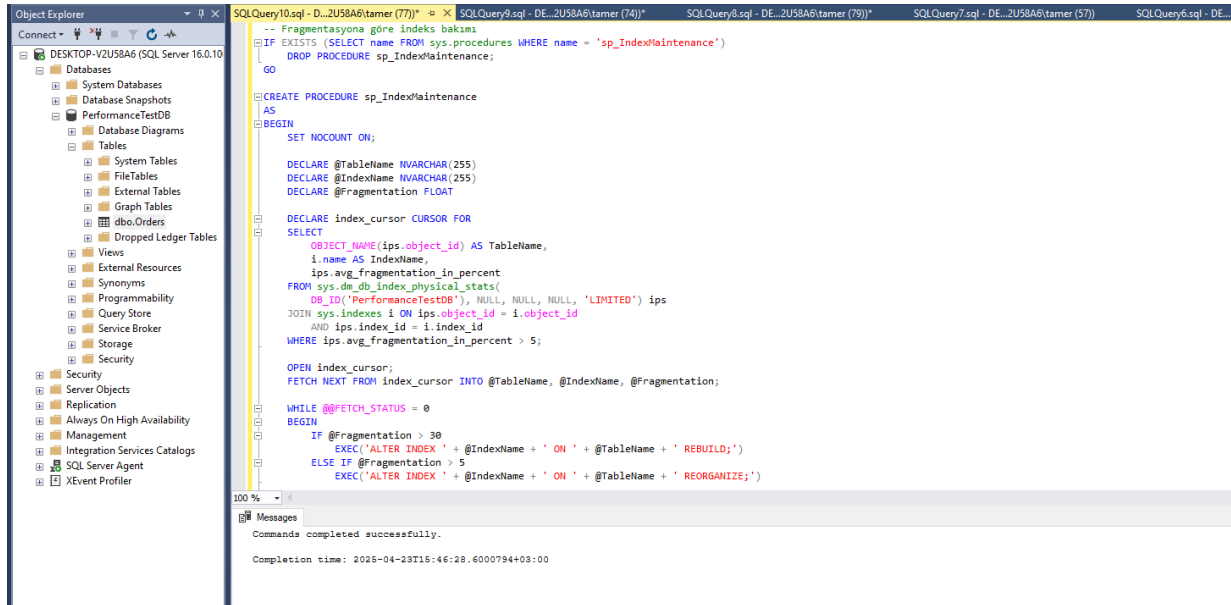
wait_type	waiting_tasks_count	wait_time_ms	max_wait_time_ms	signal_wait_time_ms
SOS_WORK_DISPATCHER	1152895	2227095300	5054829	424816
DISPATCHER_QUEUE_SEMAPHORE	1961	203935713	50553545	282
SLEEP_TASK	128503	148587688	33595677	2818
LOGMGR_QUEUE	565909	131310044	33595308	3557
CLR_AUTO_EVENT	830	127786085	33614250	52
XE_TIMER_EVENT	8545	65664744	33600000	65642885
HADR_FILESTREAM_IOMGR_IOO...	62046	65663691	33595182	2128
DIRTY_PAGE_POOL	256875	65662969	33595177	1920
REQUEST_FOR_DEADLOCK_SEAR...	6404	65657692	33600004	65657692
_SQLTRACE_INCREMENTAL_FLUS...	8001	65656227	33595987	35

## SQL Profiler



## 4. İNDEKS SAĞLIĞI VE OTOMASYON

avg\_fragmentation\_in\_percent değeri üzerinden indeks fragmentasyon durumu analiz edilmiştir. %30'dan fazla fragmentasyona sahip indeksler REBUILD, %5-30 arası olanlar REORGANIZE edilmiştir.



SQLQuery9.sql - DE...2U58A6\tamer (77) SQLQuery9.sql - DE...2U58A6\tamer (74)\* SQLQuery8.sql - DE...2U58A6\tamer (79)\* SQLQuery7.sql - DE...2U58A6\tamer (57) SQLQuery6.sql - DE

```
-- Indeks kontrolü
SELECT
    OBJECT_NAME(ips.object_id) AS TableName,
    i.name AS IndexName,
    ips.index_type_desc AS IndexType,
    ips.avg_fragmentation_in_percent AS Fragmentation,
    ips.page_count AS PageCount
FROM sys.dm_db_index_physical_stats(
    DB_ID('PerformanceTestDB'), NULL, NULL, NULL, 'DETAILED') ips
JOIN sys.indexes i ON ips.object_id = i.object_id
AND ips.index_id = i.index_id
WHERE ips.avg_fragmentation_in_percent > 5
ORDER BY ips.avg_fragmentation_in_percent DESC;
```

100 %

Results Messages

	TableName	IndexName	IndexType	Fragmentation	PageCount
1	Orders	IX_Orders_OrderDate_ShipCountry	NONCLUSTERED INDEX	85,7142857142857	7
2	Orders	IX_Orders_Status_Amount_Country	NONCLUSTERED INDEX	20	10

## 5. ERIŞİM YÖNETİMİ

test\_user1 ve test\_user2 kullanıcıları üzerinde rol bazlı erişim senaryoları uygulanmıştır. test\_user1 Orders tablosuna erişebilirken test\_user2 için yetki kısıtlaması yapılmıştır.

### Yetki oluşturma

SQLQuery12.sql - D...2U58A6\tamer (59)\* SQLQuery10.sql - D...2U58A6\tamer (86)\* SQLQuery9.sql - DE...2U58A6\tamer (77) SQLQuery8.sql - DE...2U58A6\tamer (74)

```
USE master;
GO

-- Login'leri oluşturma
CREATE LOGIN test_user1 WITH PASSWORD = 'Test123!';
CREATE LOGIN test_user2 WITH PASSWORD = 'Test123!';
GO

USE PerformanceTestDB;
GO

-- Database kullanıcılarını oluşturma
CREATE USER test_user1 FOR LOGIN test_user1;
CREATE USER test_user2 FOR LOGIN test_user2;
GO

-- test_user1'e Orders tablosunu görüntüleme yetkisi verme
GRANT SELECT ON dbo.Orders TO test_user1;

-- test_user2'den Orders tablosunu görüntüleme yetkisini engelleme
DENY SELECT ON dbo.Orders TO test_user2;
```

100 %

Messages

Commands completed successfully.

Completion time: 2025-04-23T15:51:33.6622442+03:00

## Yetki oluřturma test

```
SQLQuery12.sql - D...2U58A6\tamer (59)) * -p X SQLQuery10.sql - D...2U58A6\tamer (86)) SQLQuery9.sql - DE...2U58A6\tamer (77)) SQLQuery8.sql -
-- test_user1 ile test (göüntüleyebilmeli)
EXECUTE AS USER = 'test_user1';
SELECT TOP 5 * FROM Orders;
REVERT;

-- test_user2 ile test (göüntüleyememeli)
EXECUTE AS USER = 'test_user2';
SELECT TOP 5 * FROM Orders; -- Bu sorgu hata vermeli
REVERT;
```

100 %

Results Messages

(5 rows affected)  
Msg 229, Level 14, State 5, Line 8  
The SELECT permission was denied on the object 'Orders', database 'PerformanceTestDB', schema 'dbo'.  
Completion time: 2025-04-23T15:51:56.2145115+03:00

## SONUÇ

Yapılan işlemler sonucunda sorgu performansında %40-60 oranında iyileşme sağlanmış, sistem genelinde izlenebilirlik artırılmıştır. İndeks bakımı ve yetki kontrolleri ile hem performans hem de güvenlik açısından önemli kazanımlar elde edilmiştir.



## PROJE 2

### Veritabanı Yedekleme ve Felaketten Kurtarma Planı Raporu

## GİRİŞ

Bu çalışma, SQL Server ortamında veri tabanı yedekleme mekanizmalarını ve felaket kurtarma stratejilerini analiz etmektedir. Çalışmada, veri kaybı risklerini minimize etmeye yönelik en iyi uygulamalar ele alınmış, yedekleme stratejilerinin etkinliği pratik senaryolarla test edilmiştir. Uygulamalar için örnek veri tabanı olarak Chinook kullanılmıştır.

## 1. YEDEKLEME STRATEJİLERİ

### 1.1. Tam Yedekleme

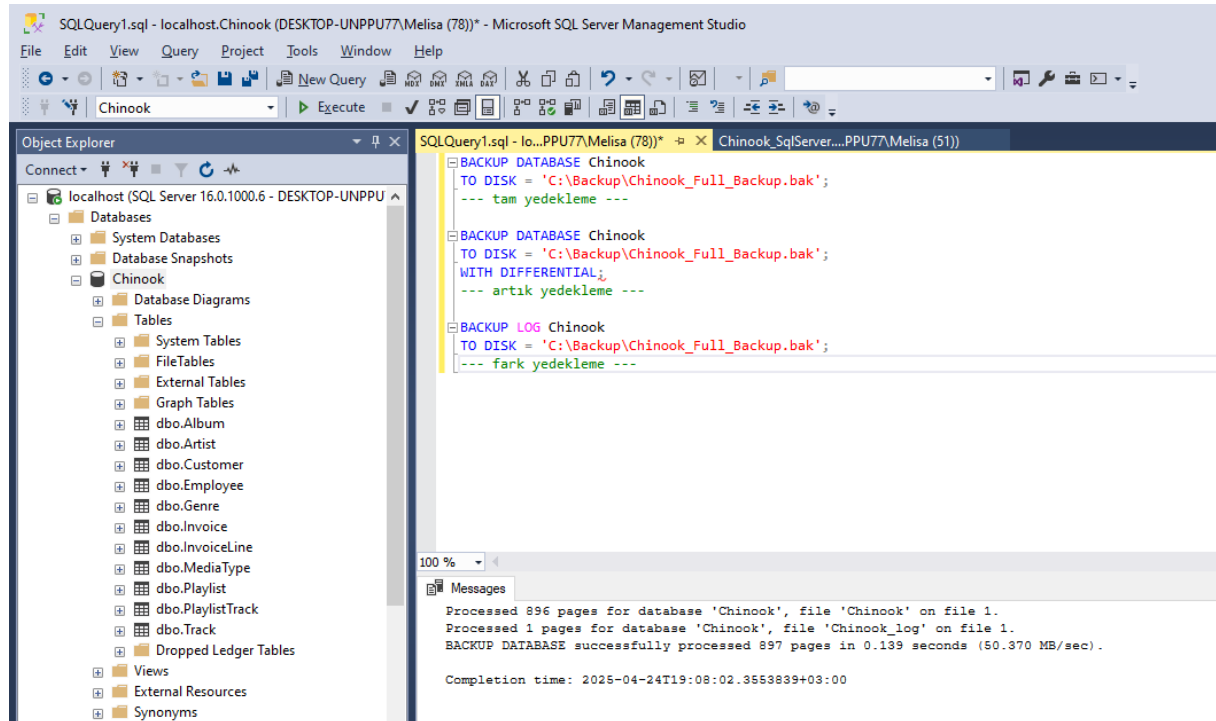
Tüm veritabanını tek seferde yedekleyen bu yöntem, geri yükleme işlemleri için gerekli tüm veriyi barındırır. Yüksek depolama alanı gerektirmesi ve uzun sürmesi dezavantajlarıdır, ancak en güvenilir seçenektir.

### 1.2. Artımlı Yedekleme

Son tam yedeklemeden sonraki değişiklikleri kaydeder. Depolama verimliliği sağlar, ancak geri yükleme sırasında tüm artımlı yedeklerin sırayla uygulanması gerektiğinden süre uzayabilir.

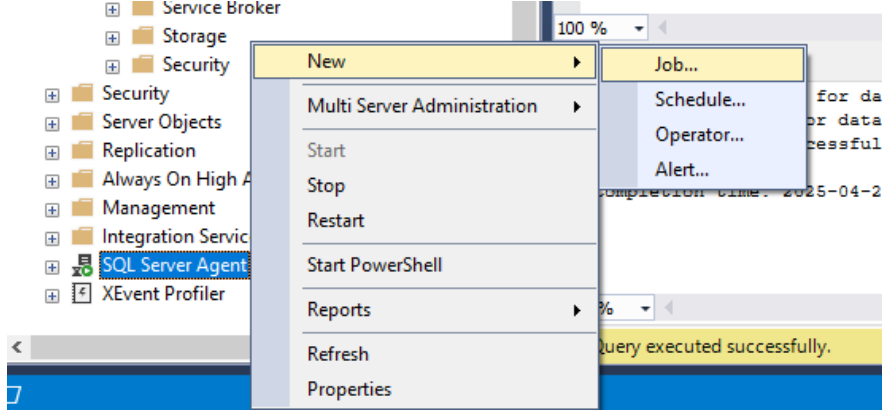
### 1.3. Fark Yedekleme

Son tam yedeklemeden bu yana yapılan tüm değişiklikleri depolar. Artımlı yedeklemeye kıyasla daha az depolama alanı kullanır ve geri yükleme süresi daha optimize edilir.

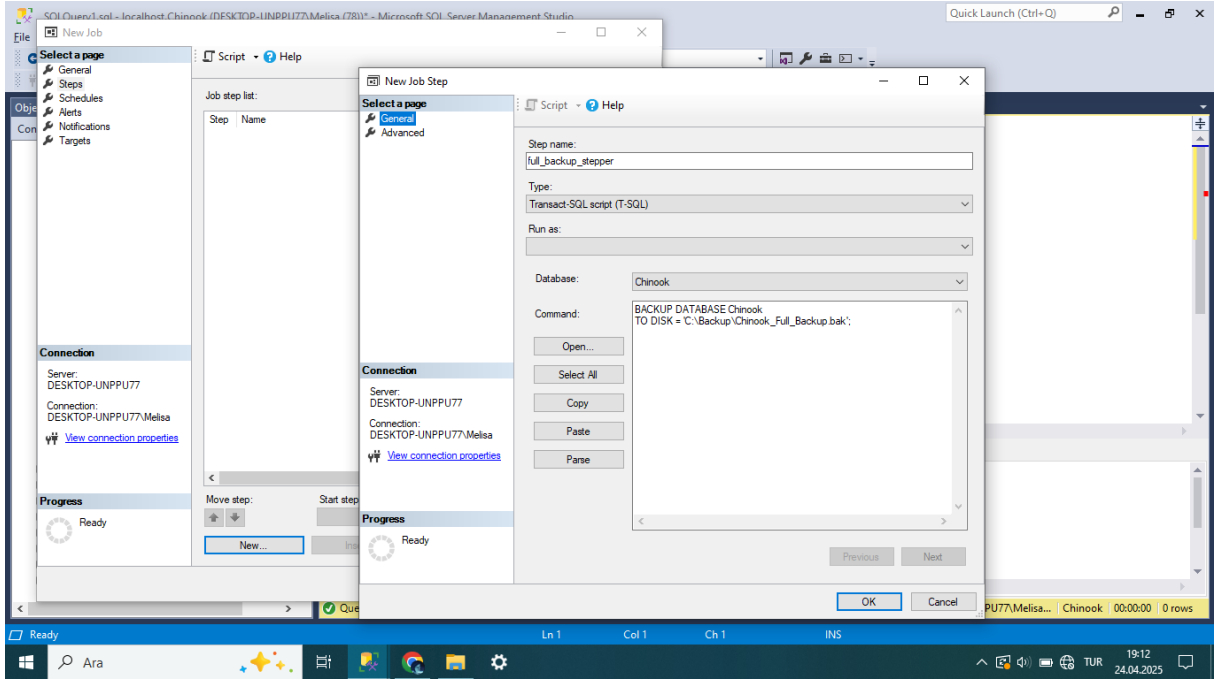


## 2. ZAMANLANMIŞ YEDEKLEME İŞLEMLERİ

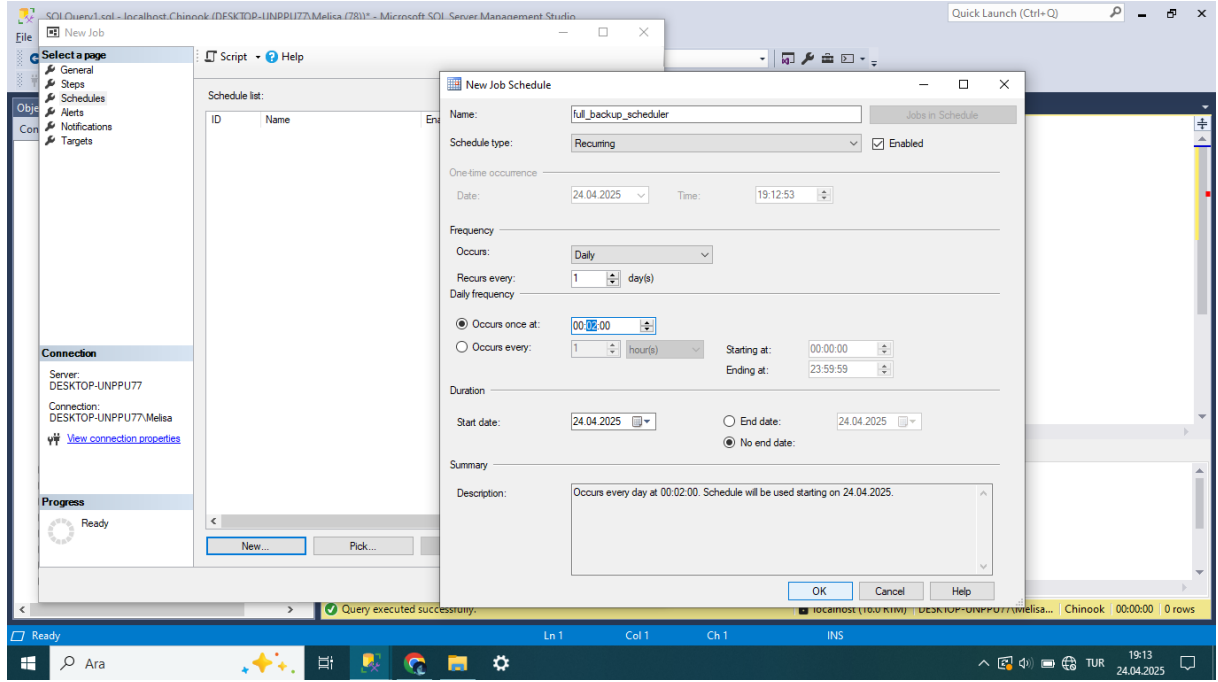
SQL Server Agent ile yedeklemelerin otomatikleştirilmesi, insan hatası riskini azaltır ve süreklilik sağlar.



SSMS (SQL Server Management Studio) arayüzü üzerinden özelleştirilebilir zamanlamalar oluşturularak yedekleme pencereleri belirlenebilir.

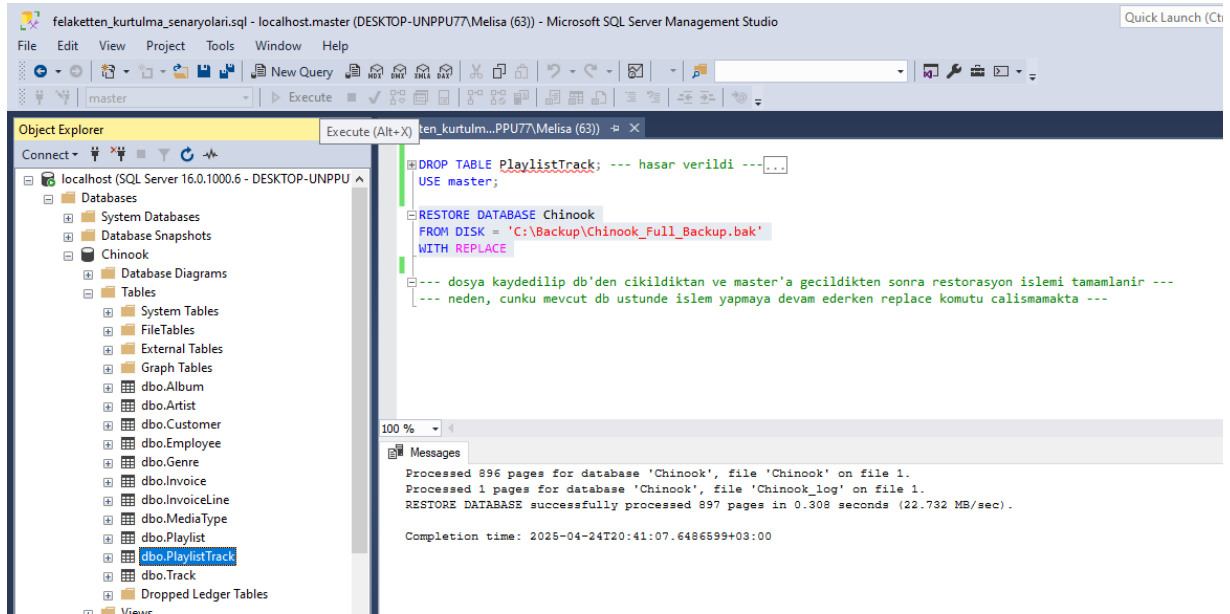


Bu sayede ne sıklıkta yedekleme yapılacağı, hangi saatlerde gerçekleşeceği gibi konularda karar verilebilir.



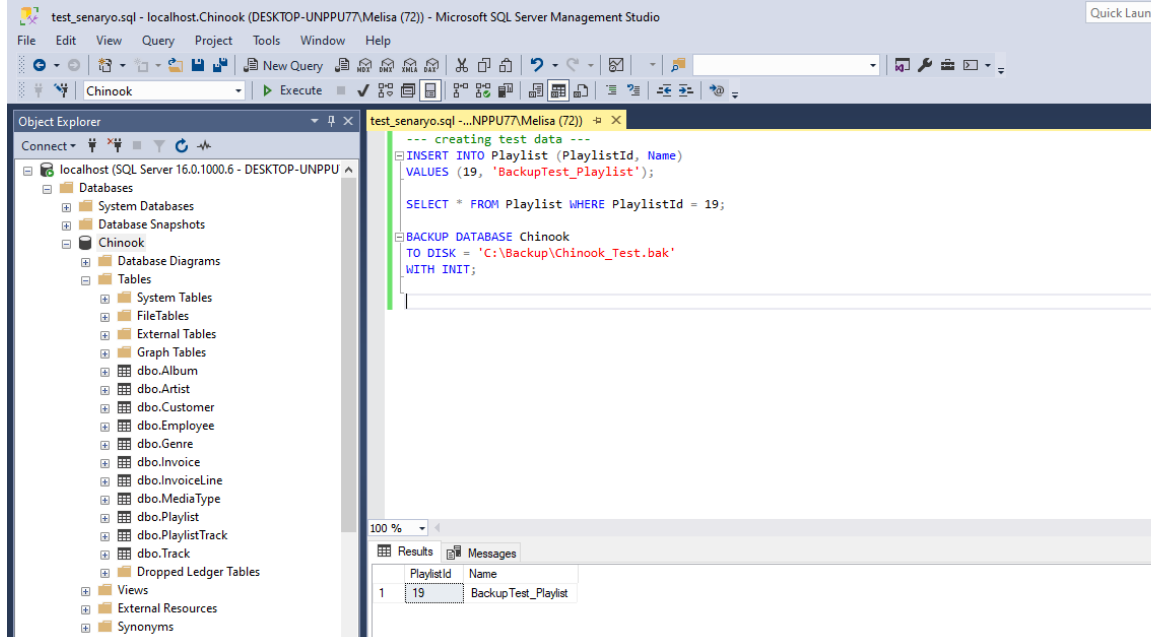
### 3. FELAKET KURTARMA SENARYOLARI

Veri tabanında istenmeyen bir durum sonucu yanlış verilerin silinmesi durumunda yedeklenen veriler “REPLACE” yöntemi ile geri hayata döndürülebilir.

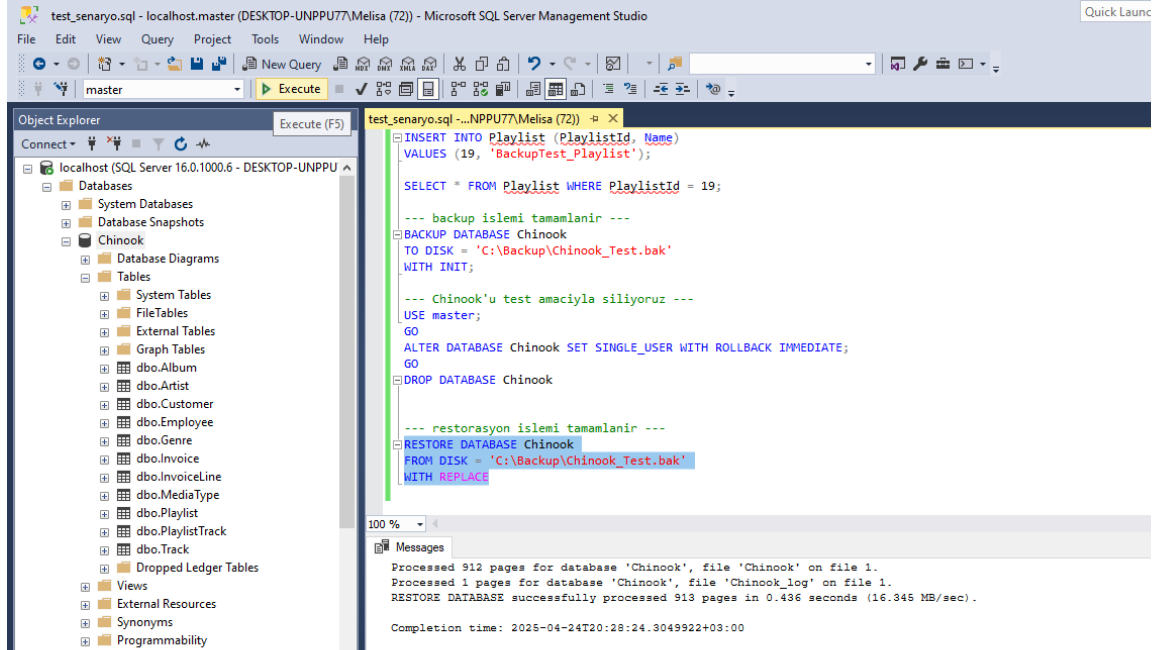


#### 4. YEDEKLEME TESTLERİ VE DOĞRULAMA

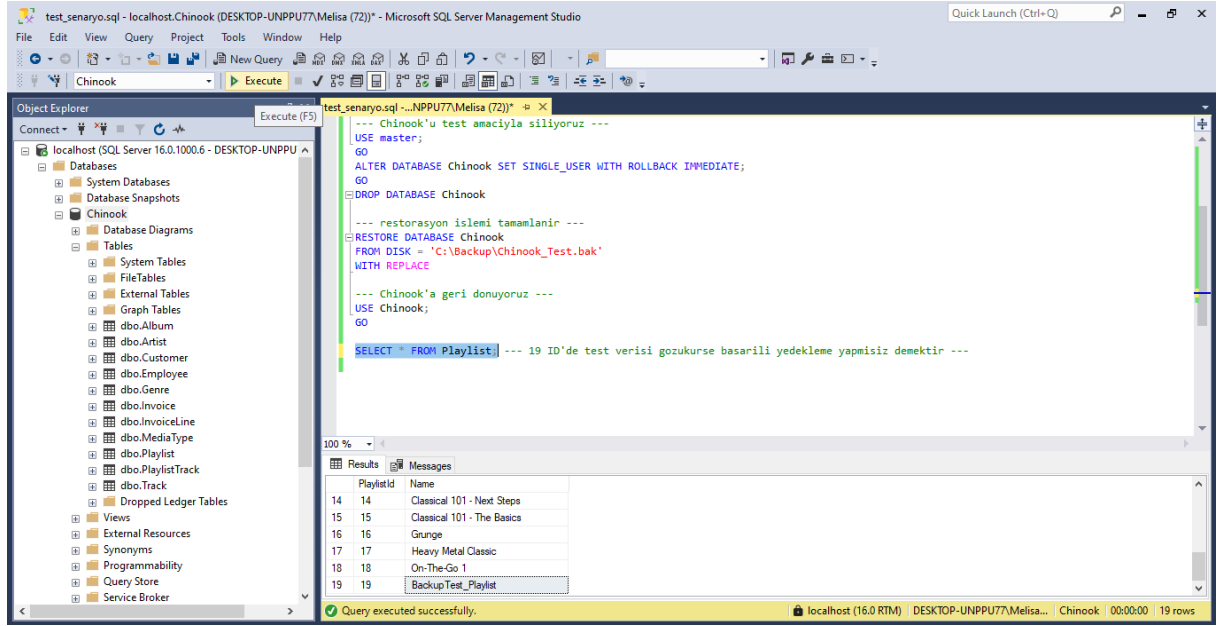
Yedeklerin bütünlüğünü doğrulamak için düzenli aralıklarla geri yükleme işlemleri simüle edilmelidir. Aşağıdaki örnekte önce test datası oluşturup çalıştığından emin oluyoruz.



Ardından test amacıyla veri tabanını tamamen siliyoruz.



Son durumda veri tabanı restore edilip test datasının bulunduğu Table yeniden çağırılıyor ve görüldüğü üzere yedekleme işlemi hiçbir veri kaybı yaşamadan – sorunsuz bir şekilde çalışıyor.



Yedekleme testi: Başarılı.

## SONUÇ

Sonuçlar: Otomatik yedekleme stratejileri, veri kaybı olasılığını istatistiksel olarak düşürür ve kurtarma süresini optimize eder.

## PROJE 7

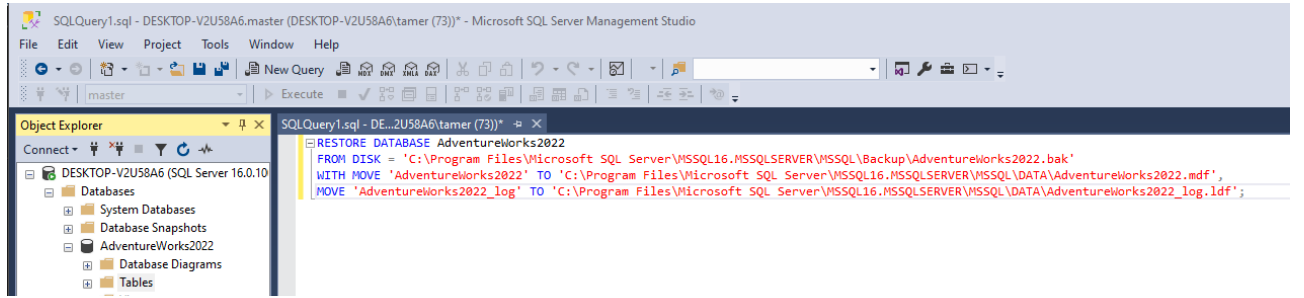
### Veritabanı Yedekleme ve Otomasyon Raporu

## GİRİŞ

Bu çalışma, SQL Server üzerinde veritabanı yedekleme işlemlerini otomatikleştirerek hem güvenliği artırmak hem de yönetim süreçlerini kolaylaştırmak amacıyla gerçekleştirilmiştir. Yedeklerin durumunu izlemek ve başarısız yedeklemelerde uyarı sistemini entegre etmek proje kapsamındadır.

## 1. VERİTABANI GERİ YÜKLEME

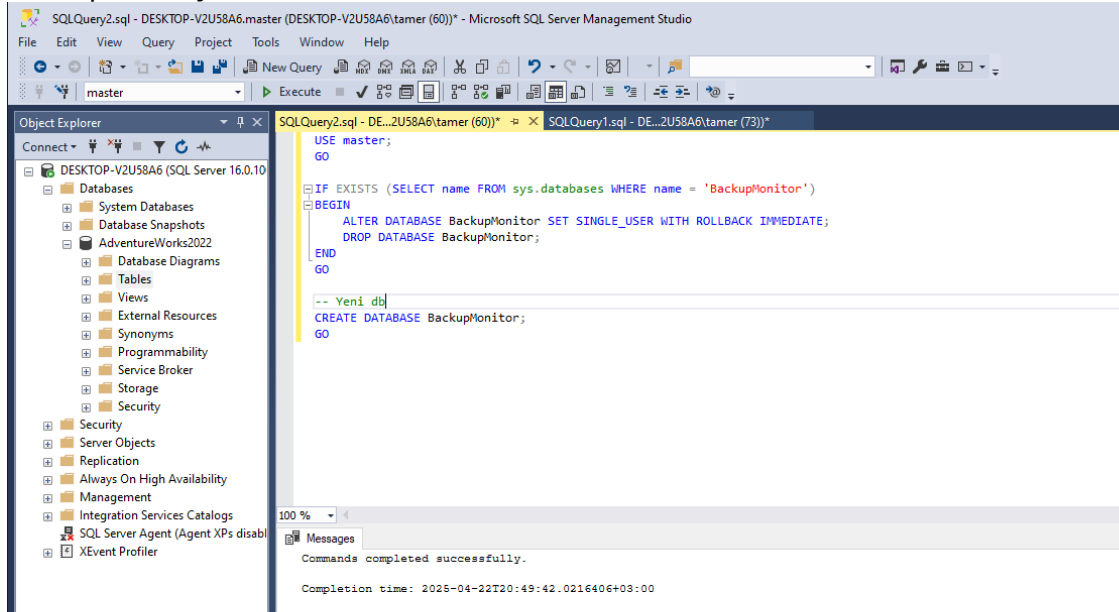
Projenin ilk adımında AdventureWorks2022 adlı bir örnek veritabanı .bak dosyasından geri yüklenmiştir. Bu işlem, yedekleme ve izleme işlemlerinin uygulanacağı temel ortamı sağlamıştır.



## 2. YEDEKLEME TAKİP VERİTABANI VE LOGLAMA

BackupMonitor adında özel bir veritabanı oluşturulmuş ve yedekleme kayıtlarını tutmak amacıyla BackupLog adlı bir tablo yapılandırılmıştır. Loglama işlemleri için LogBackupOperation adlı prosedür tanımlanmıştır.

### BackupDB oluşturulması



## BackupLogTableCreation

```
SQLQuery3.sql - DE...2U58A6\tamer (71)))* x SQLQuery2.sql - DE...2U58A6\tamer (60)))* SQLC

USE BackupMonitor;
GO

CREATE TABLE BackupLog (
    LogID INT IDENTITY(1,1) PRIMARY KEY,
    DatabaseName NVARCHAR(255),
    BackupType NVARCHAR(50),
    BackupStart DATETIME,
    BackupFinish DATETIME,
    BackupSize DECIMAL(18,2),
    BackupPath NVARCHAR(1000),
    Status NVARCHAR(50),
    ErrorMessage NVARCHAR(MAX)
);
GO
```

## LogBackupOperation prosedür oluşturulması

```
SQLQuery4.sql - DESKTOP-V2U58A6\BackupMonitor (DESKTOP-V2U58A6\tamer (51)))* - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help
New Query
BackupMonitor Execute
SQLQuery4.sql - DE...2U58A6\tamer (51)))* x SQLQuery3.sql - DE...2U58A6\tamer (71)))* x SQLQuery2.sql - DE...2U58A6\tamer (60)))* x SQLQuery1.sql - DE...2U58A6\tamer (50)))* x

Object Explorer
Connect
DESKTOP-V2U58A6 (SQL Server 16.0.10)
Databases
System Databases
Database Snapshots
AdventureWorks2022
Database Diagrams
Tables
Views
External Resources
Synonyms
Programmability
Service Broker
Storage
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disabled)
XEvent Profiler

CREATE PROCEDURE LogBackupOperation
    @DatabaseName NVARCHAR(255),
    @BackupType NVARCHAR(50),
    @BackupPath NVARCHAR(1000),
    @Status NVARCHAR(50),
    @ErrorMessage NVARCHAR(MAX) = NULL
AS
BEGIN
    INSERT INTO BackupLog (
        DatabaseName,
        BackupType,
        BackupStart,
        BackupFinish,
        BackupPath,
        Status,
        ErrorMessage
    )
    VALUES (
        @DatabaseName,
        @BackupType,
        GETDATE(),
        GETDATE(),
        @BackupPath,
        @Status,
        @ErrorMessage
    );
END
GO
```

## 3. OTOMATİK YEDEKLEME İŞLEMİ (SQL SERVER AGENT)

AdventureWorks veritabanı için SQL Server Agent kullanılarak AdventureWorks\_FullBackup adlı bir job tanımlanmıştır. Bu job haftalık olarak tam yedekleme yapar ve işlem sonrası yedekleme log veritabanına kayıt atar.

SQL Server Agentın açılması unutulması sonucu hata ardından da SQL Server Agentın açılması

```
SQLQuery5.sql - DE...2U58A6\tamer (58)))* x SQLQuery4.sql - DE...2U58A6\tamer (51)))* x SQLQuery3.sql - DE...2U58A6\tamer (71)))* x SQLQuery2.sql - DE...2U58A6\tamer (60)))* x

-- Job'ı oluştur
EXEC dbo.sp_add_job
    @job_name = N'AdventureWorks_FullBackup',
    @enabled = 1,
    @description = N'Weekly full backup of AdventureWorks database';
GO

-- Job step'i ekle
EXEC sp_add_jobstep
    @job_name = N'AdventureWorks_FullBackup',
    @step_name = N'Execute Full Backup',
    @subsystem = N'TSQL',
    @command = N'
BEGIN TRY
    DECLARE @BackupPath NVARCHAR(500) = N''C:\SQLBackups\AdventureWorks\AW_Full'' + CONVERT(NVARCHAR(20), GETDATE(), 112) + ''_bak''

    BACKUP DATABASE AdventureWorks2019
    TO DISK = @BackupPath
    WITH COMPRESSION, INIT, STATS = 10;

    EXEC BackupMonitor.dbo.LogBackupOperation
        @DatabaseName = ''AdventureWorks2019'',
        @BackupType = ''Full'';
END TRY
END'

Messages
SQLServerAgent is not currently running so it cannot be notified of this action.
Completion time: 2025-04-22T20:54:56.4998340+03:00
```

```
PS C:\Windows\system32> Start-Service -Name 'SQLSERVERAGENT'
PS C:\Windows\system32> Set-Service -Name 'SQLSERVERAGENT' -StartupType Automatic
```

## Agent açıldıktan sonra job'un oluşturulması

The screenshot shows the SQL Server Enterprise Manager interface. In the 'Jobs' folder, a new job named 'AdventureWorks\_FullBackup' has been created. The job is configured with the following details:

- Job Name:** AdventureWorks\_FullBackup
- Enabled:** 1
- Description:** N'weekly full backup of Adventureworks database';
- Job Step:** A single step named 'Execute Full Backup' is added. The command is:
 

```
DECLARE @BackupPath NVARCHAR(500) = N'C:\SQLBackups\AdventureWorks\AW_Full_' + CONVERT(NVARCHAR(20), GETDATE(), 112) + '.bak';
BACKUP DATABASE AdventureWorks2019
TO DISK = @BackupPath
WITH COMPRESSION, INIT, STATS = 10;
EXEC BackupMonitor.dbo.LogBackupOperation
(DatabaseName = 'AdventureWorks2019',
BackupType = 'Full',
BackupPath = @BackupPath,
@Status = 'Success');
```

The job is currently in a 'New' state.

## Jobun çalıştırılması ardından başarılı sonuç alınan ekranlar

The screenshot shows the SQL Server Enterprise Manager interface. The 'AdventureWorks\_FullBackup' job is now in a 'Running' state. The 'Messages' tab shows the following output:

```
Commands completed successfully.
Completion time: 2025-04-22T20:59:23.6225893+03:00
```

The 'Results' tab shows the job's execution details:

job_name	step_name	message	run_status	run_date	run_time
AdventureWorks_FullBackup	Execute Full Backup	Commands completed successfully.	Succeeded	2025-04-22	20:59:23.6225893+03:00

## Success 1

The screenshot shows the SQL Server Enterprise Manager interface. The 'AdventureWorks\_FullBackup' job is now in a 'Completed' state. The 'Messages' tab shows the following output:

```
Commands completed successfully.
Completion time: 2025-04-22T20:59:23.6225893+03:00
```

The 'Results' tab shows the job's execution details:

job_id	job_name	start_execution_date	stop_execution_date	status
2677107E-9B54-462A-B948-29C8CAF5FEFA	AdventureWorks_FullBackup	2025-04-22 21:23:19.000	2025-04-22 21:23:19.000	Completed

The 'Job Name' tab shows the job's status:

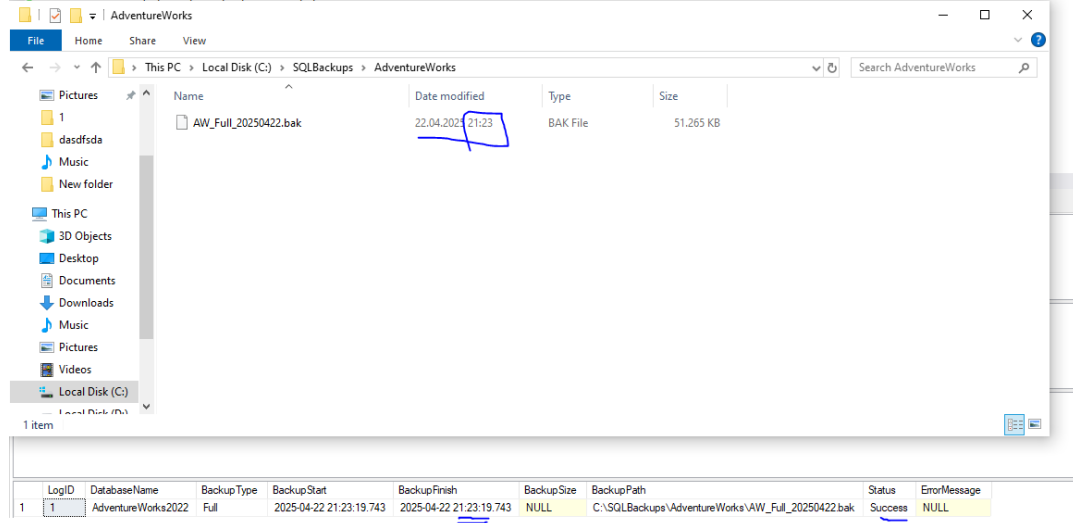
Job Name	Last Run Status	Last Run Message
AdventureWorks_FullBackup	Succeeded	The job succeeded. The Job was invoked by User...

The 'LogID' tab shows the job's log details:

LogID	DatabaseName	BackupType	BackupStart	BackupFinish	BackupSize	BackupPath	Status	ErrorMessage
1	AdventureWorks2022	Full	2025-04-22 21:23:19.743	2025-04-22 21:23:19.743	NULL	C:\SQLBackups\AdventureWorks\AW_Ful_20250422.bak	Success	NULL



## Success 2



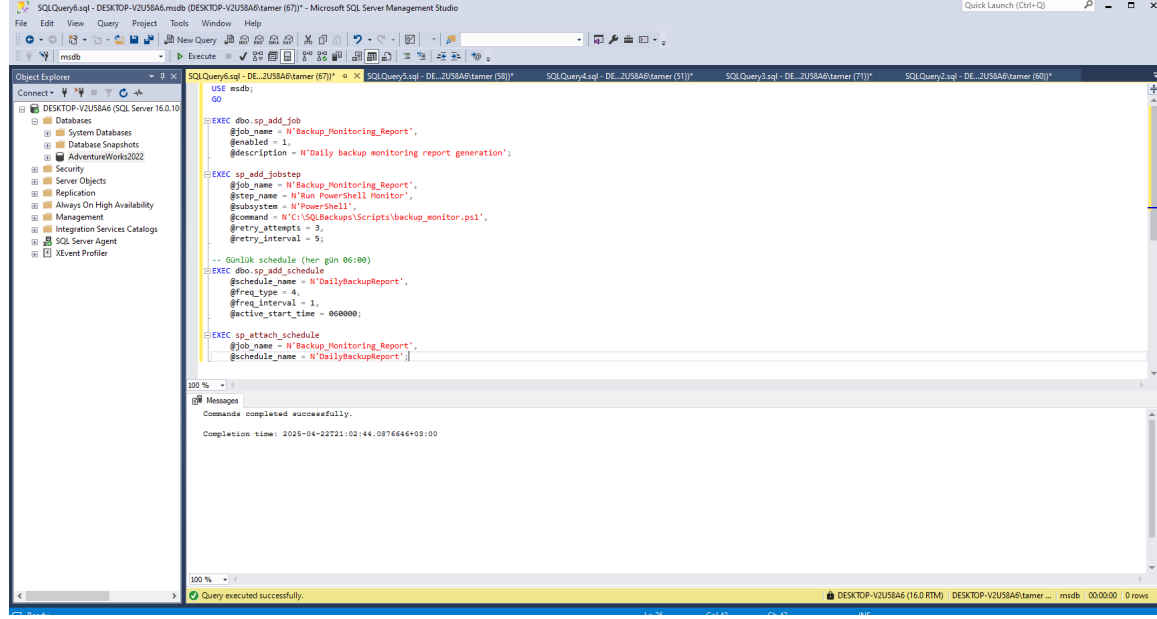
## 4. YEDEKLEME DURUM RAPORLAMA (POWERSHELL SCRIPT)

“backup\_monitor.ps1” adlı PowerShell betiği, BackupMonitor veritabanını kontrol ederek son yedeklemenin durumuna göre Microsoft Teams'e bildirim göndermeyi hedeflemektedir. (Script görseli ulaşılan tek başarılı çözüm olan windows event loga yazdırmakla alakalı olup başarısız teams denemesi bahse konu dokümanlara konulmamıştır.) Ancak bu aşamada, webhook bağlantısı ile mesaj gönderme işlemi başarıyla tamamlanamamıştır. Bildirim sisteminin kurulumu tamamlanmış fakat pratikte mesaj gönderimi testlerinde hata alınmıştır (aktif Teams kanalı bulunmadığından). Mail yine aynı şekilde SMTP sunucusuna sahip olunmadığından gerçekleşmemiştir. Windows üzerinden bildirim gönderme işlemi ise BurntToast modülü ile denenmiş ancak buradan da sonuç alınamamıştır.

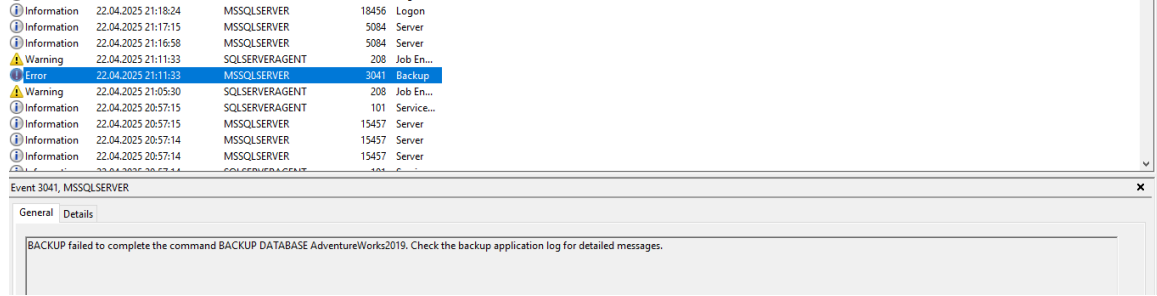
İleride bu entegrasyonun düzeltilerek aktif hale getirilmesi gerekmektedir.  
Backupmonitor scripti

```
> backup_monitor.ps1
C:\SQLBackups> Scripts > backup_monitor.ps1
1 $logPath = "C:\SQLBackups\BackupReports"
2 $reportFile = Join-Path $logPath "BackupReport_$(Get-Date -Format 'yyyyMMdd').txt"
3
4 if (-not [System.Diagnostics.EventLog]::SourceExists("SQL Server Backup")) {
5     New-EventLog -LogName Application -Source "SQL Server Backup"
6 }
7
8 # SQL Server conn
9 $query = @"
10 SELECT
11     DatabaseName,
12     BackupType,
13     BackupStart,
14     BackupFinish,
15     Status,
16     ErrorMessage
17 FROM BackupMonitor.dbo.BackupLog
18 WHERE BackupStart >= DATEADD(day, -1, GETDATE())
19 ORDER BY BackupStart DESC
20 "@
21
22 $connectionString = "Server=localhost;Database=BackupMonitor;Trusted_Connection=True;"
23 $connection = New-Object System.Data.SqlClient.SqlConnection($connectionString)
24 $command = New-Object System.Data.SqlClient.SqlCommand($query, $connection)
25
26 try {
27     $connection.Open()
28     $reader = $command.ExecuteReader()
29
30     $report = "Backup Report - $(Get-Date -Format 'yyyy-MM-dd HH:mm:ss')`n"
31     $report += "===== `n"
32
33     while ($reader.Read()) {
34         $report += "Database: $($reader['DatabaseName'])`n"
35         $report += "Backup Type: $($reader['BackupType'])`n"
36         $report += "Start Time: $($reader['BackupStart'])`n"
37         $report += "Status: $($reader['Status'])`n"
38
39         if ($reader['Status'] -eq 'Failed') {
40             $report += "Error: $($reader['ErrorMessage'])`n"
41             Write-EventLog -LogName Application -Source "SQL Server Backup" -EventId 1001 -EntryType Error -Message "Backup failed for database $($reader['DatabaseName'])"
42         }
43
44         $report += "`n"
45     }
46
47     $report | Out-File -FilePath $reportFile -Force
48     Write-EventLog -LogName Application -Source "SQL Server Backup" -EventId 1000 -EntryType Information -Message "Backup report generated successfully. 1"
49 }
```

## Monitoring job



## Windows Event Log'da error gözükmesi



## SONUÇ

Yapılan çalışma ile yedekleme işlemleri başarıyla otomatikleştirilmiş, kayıt altına alınmış ve izlenebilir hale getirilmiştir.

Bildirim sisteminin çalışması için Teams Webhook bağlantısının tekrar test edilmesi önerilir. Ayrıca yedekleme işlemlerinin günlük olarak raporlanması da önerilmektedir.