

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



**BLM4522 Final Raporu**

**İbrahim YILDIZ 21290706**

**Aymina YILIK 21290188**

**30.05.2025**

**Github Hesabım:**

İbrahim Yıldız: <https://github.com/ibrahimyz>

**Repo linkim:**

<https://github.com/ibrahimyz/BLM4522-21290706-Odevleri>

**Video için playlistim:**

[BLM4522 21290706 Proje Videoları](#)

Bu raporda yaptığımız projeleri tek tek açıklayalım.

**1. Veritabanı Yedekleme ve Felaketten Kurtarma Planı**

Bu projede öncelikle istenilen bir veritabanı yedekleme ve felaketten kurtarma planlarının tasarlanmasıydı. Bu planlar, bir organizasyonun verilerini korumak ve herhangi bir veri kaybı durumunda hızlı bir şekilde normal operasyonlara dönebilmesini sağlamak için kritik önem taşır.

Planımız şu şekilde olabilir:

1. Risk ve ihtiyaç analizinin yapılması gerekir. Aşağıdaki kavramların cevaplarının belirlenmesi gerekir.
  - RPO (Recovery Point Objective) yani “sistem çökmesi veya veri kaybı yaşandığında en fazla ne kadar eskiye dönmek kabul edilebilir (iş kaybı olmaz)?” sorusunun cevabının belirlenmesi
  - RTO (Recovery Time Objective) yani felaketten sonra servisin tekrar erişilebilir hale gelmesi için gereken maksimum sürenin belirlenmesi
  - Veritabanının iş için ne kadar kritik olduğunun belirlenmesi ve bunlara göre RPO ve RTO hedeflerinin ve yedekleme sıklıklarının belirlenmesi
2. Yedekleme stratejisinin belirlenmesi
  - Tam, artık ve fark yedeklemelerinin hangisinin ne sıklıkta kullanılacağı belirlenmeli
3. Otomasyon ve İzleme
  - SQL Server Agent ile zamanlayıcı atanıp otomasyona bağlanmalı ve olağanüstü durum ve yedekleme başarısızlıklarının takibi yapılmalı
4. Kurtarma Senaryoları
  - Silinen verinin getirilmesi ve Point-in-time restore kullanımı
5. Test ve Dokümantasyon
  - Düzenli yedek doğruluk testi ve adım adım kurtarma dokümantasyonlarının hazırlanması

Şimdi bunların nasıl yapılacağını bazı örnek senaryolar ve scriptler üzerinden gösterelim.

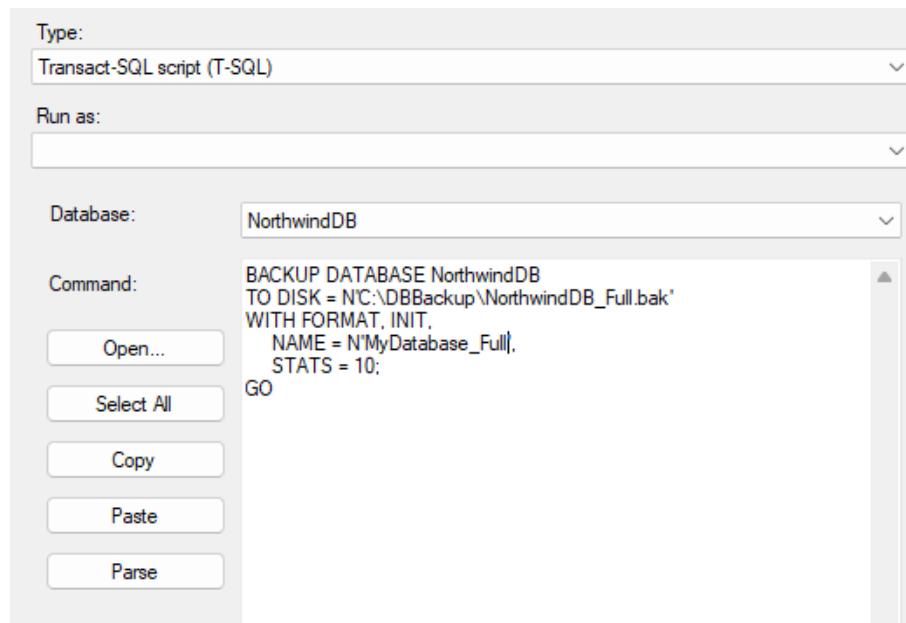
#### Point-in-time restore senaryosu:

Diyelim ki elimizdeki veri silindi ya da üzerinde işlem yaparken bozuldu. Bunu kurtarmanın yolu şu şekilde olabilir:

Önce point in time'a dönebilmek için log alınan Full recovery mode'unu aktifleştirmemiz gerekiyor.

```
ALTER DATABASE NorthwindDB  
SET RECOVERY FULL;  
GO
```

Burada SQL Server Agent ile otomatikleştirdiğimiz yedeklerin Step aşamasında eklediğimiz commandleri var. Full, differential (günlük), transaction log (saatlik) olarak yedeklemeleri yaptık.



Type: Transact-SQL script (T-SQL)

Run as:

Database: NorthwindDB

Command:

```
BACKUP DATABASE NorthwindDB  
TO DISK = N'C:\DBBackup\NorthwindDB_Full.bak'  
WITH FORMAT, INIT,  
NAME = N'MyDatabase_Full',  
STATS = 10;  
GO
```

Open... Select All Copy Paste Parse

Step name:  
DifferentialBU

Type:  
Transact-SQL script (T-SQL)

Run as:

Database: NorthwindDB

Command:

```
BACKUP DATABASE NorthwindDB  
TO DISK = N'C:\DBBackup\NorthwindDB_Diff.bak'  
WITH DIFFERENTIAL,  
NAME = N'NorthwindDB_Diff_20250421',  
STATS = 10;  
GO
```

Open...  
Select All  
Copy  
Paste  
Parse

Step name:  
Transaction Log

Type:  
Transact-SQL script (T-SQL)

Run as:

Database: NorthwindDB

Command:

```
BACKUP LOG NorthwindDB  
TO DISK = N'C:\DBBackup\NorthwindDB_Log.tm'  
WITH NOFORMAT, NOINIT,  
NAME = N'NorthwindDB',  
STATS = 5;  
GO
```

Open...  
Select All

Job Schedule Properties - Hourly

Name: Hourly Jobs in Schedule

Schedule type: Recurring ☒ Enabled

One-time occurrence

Date: 24.04.2025 Time: 20:56:08

Frequency

Occurs: Daily

Recurs every: 1 day(s)

Daily frequency

☐ Occurs once at: 00:00:00

☒ Occurs every: 1 hour(s) Starting at: 00:00:01 Ending at: 23:59:59

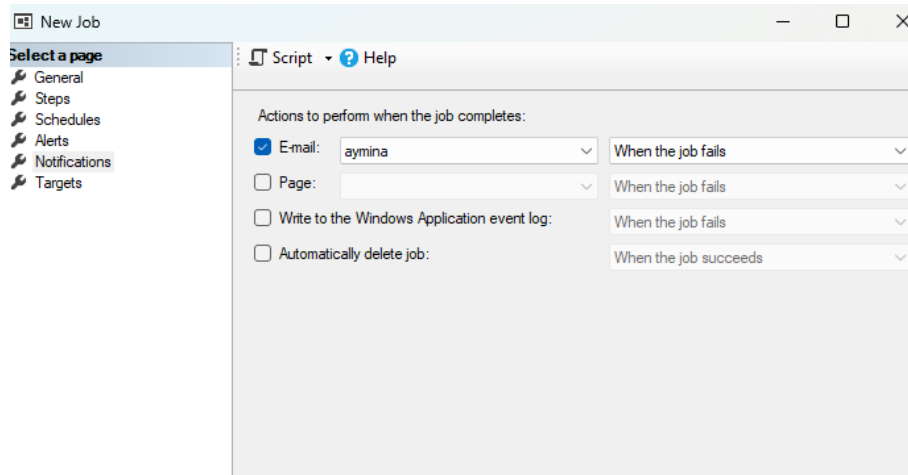
Duration

Start date: 14.04.2025 ☐ End date: 24.04.2025 ☒ No end date:

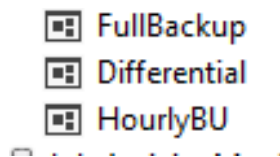
Summary

Description: Occurs every day every 1 hour(s) between 00:00:01 and 23:59:59. Schedule will be used starting on 14.04.2025.

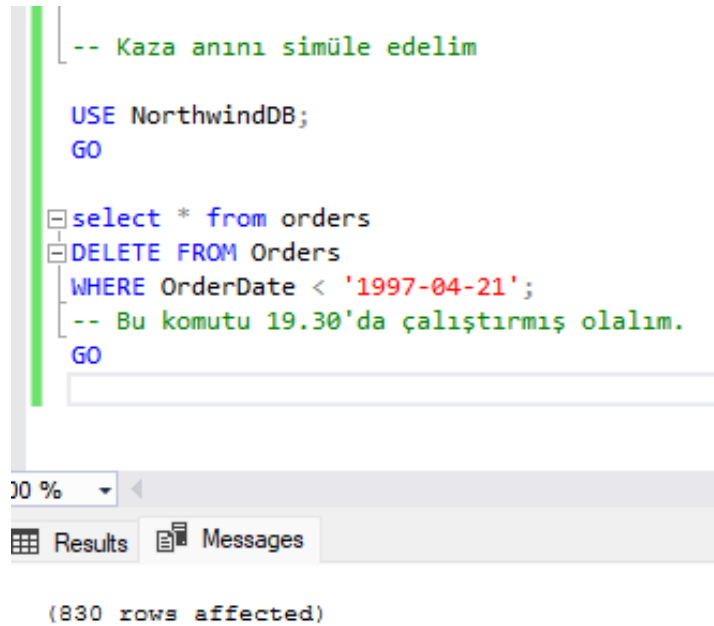
Bunları eklerken hata durumunda daha önce eklenen iletişim operatörü mailine mail atılması için bildirim ve uyarı ekledik:



Bu şekilde yedeklerimizi oluşturduk:



Yedeklerimizin sonrasında kaza anını simüle ettik. Verdiğimiz örnekte 24 Nisan 1977'den önceki orderları siliyor.



Daha sonra kaza öncesi logları almak için **tail-log** backup alıp veritabanını NORECOVERY moduna geçiriyoruz ki başka bağlantıları reddetsin.

```
-- Kaza öncesi olan tüm logları alalım, bunu yapmak için bağlantımızı değiştirmemiz gerekiyor

USE Master;
GO

BACKUP LOG NorthwindDB
TO DISK = N'C:\DBBackup\NorthwindDB_Tail.trn'
WITH NORECOVERY;
GO
```

00 %

Messages

Processed 2021 pages for database 'NorthwindDB', file 'NorthwindDB\_log' on file 1.  
BACKUP LOG successfully processed 2021 pages in 0.016 seconds (986.816 MB/sec).

Daha sonra yedekleri geri yüklüyoruz:

```
-- Full Backup'tan geri yükle , diğerleri de aynı şekilde yapılır.

RESTORE DATABASE NorthwindDB
FROM DISK = N'C:\DBBackup\NorthwindDB_Full.bak'
WITH REPLACE, NORECOVERY;
GO
```

%

Messages

Processed 1072 pages for database 'NorthwindDB', file 'NorthwindDB' on file 1.  
Processed 2 pages for database 'NorthwindDB', file 'NorthwindDB\_log' on file 1.  
RESTORE DATABASE successfully processed 1074 pages in 0.014 seconds (599.051 MB/sec).

Bu da Tail Log'un yükleniş şekli de bu şekilde.

```
-- Tail-Log yedeğini de kazadan hemen öncesine (STOPAT) getirecek şekilde RECOVERY ile yükleyebiliriz

RESTORE LOG NorthwindDB
FROM DISK = N'C:\DBBackup\NorthwindDB_Tail.trn'
WITH STOPAT = '2025-04-19 19:30:00', -- kaza öncesi son anı buraya yazıyoruz
RECOVERY;
GO
```

Testin doğruluğunu bir count query'si yazarak tespit edebiliriz.

```
-- Silinen kayıtların geri geldiğini doğrulayalım

USE NorthwindDB;
GO

SELECT COUNT(*)
FROM orders
WHERE OrderDate < '1997-04-21';
GO
```

00 %

Results Messages

(No column name)
1 830

## 2. Veritabanı Performans Optimizasyonu ve İzleme

Veritabanı olarak çok büyük bir veritabanı olmasa da daha önceki projelerimizde de kullandığımız Northwind'i seçtik.

Veritabanımızın durum analizini yaparak başlıyoruz:

```
-- Northwind boyutu
SELECT
    SUM(size) * 8.0/1024 AS [Size_MB]
FROM sys.database_files;
```

.00 %

Results Messages

	Size_MB
1	144.000000

```
-- Bellek kullanımı
SELECT
    object_name,
    counter_name,
    cntr_value
FROM sys.dm_os_performance_counters
WHERE counter_name = 'Page life expectancy';
```

0 %

Results Messages

object_name	counter_name	cntr_value
SQLServer:Buffer Manager	Page life expectancy	1109
SQLServer:Buffer Node	Page life expectancy	1109

Bunun için bazı DMV sorguları kullanıyoruz:

```
-- Bekleyen (wait) istatistiklerini inceleme
SELECT
    wait_type,
    waiting_tasks_count,
    wait_time_ms/1000.0 AS toplam_bekleme_saniye,
    max_wait_time_ms/1000.0 AS maksimum_bekleme_saniye
FROM sys.dm_os_wait_stats
WHERE waiting_tasks_count > 0
    AND wait_type NOT IN ('CLR_SEMAPHORE', 'LAZYWRITER_SLEEP', 'SLEEP_TASK', 'BROKER_TO_FLUSH', 'SQLTRACE_BUFFER_FLUSH')
ORDER BY wait_time_ms DESC;
```

wait_type	waiting_tasks_count	toplam_bekleme_saniye	maksimum_bekleme_saniye
SOS_WORK_DISPATCHER	21611	249401.862000	5047.441000
LOGMGR_QUEUE	74363	10355.130000	0.176000
DISPATCHER_QUEUE_SEMAPHORE	140	8232.012000	120.015000
DIRTY_PAGE_POLL	47407	5177.633000	0.128000
XE_TIMER_EVENT	1512	5177.173000	5.014000
HADR_FILESTREAM_IOMGR_IOCO...	10146	5177.093000	0.516000
SQLTRACE_INCREMENTAL_FLUS...	1292	5173.814000	4.015000
REQUEST_FOR_DEADLOCK_SEAR...	1034	5172.790000	5.016000
QDS_PERSIST_TASK_MAIN_LOOP...	87	5160.648000	60.015000
XE_DISPATCHER_WAIT	95	5134.693000	146.097000
SP_SERVER_DIAGNOSTICS_SLEEP	2567	5100.127000	300.014000
CHECKPOINT_QUEUE	26	5080.522000	1277.612000
QDS_ASYNC_QUEUE	5	2834.085000	1433.577000
BROKER_TASK_STOP	31	280.454000	10.014000
LCK_M_SCH_S	1	253.791000	253.791000
PWAIT_ALL_COMPONENTS_INITIA...	3	0.768000	0.261000
ASYNC_NETWORK_IO	3911	0.479000	0.026000
PREEMPTIVE_OS_QUERYREGISTRY	14863	0.276000	0.002000
CHKPT	1	0.206000	0.206000
SLEEP_SYSTEMTASK	1	0.206000	0.206000

```

-- En fazla CPU/çesitli kaynağı kullanan sorgularla listeleme
SELECT TOP 20
    qs.sql_handle,
    qs.execution_count,
    qs.total_worker_time AS toplam_cpu_ms,
    qs.total_elapsed_time/1000 AS toplam_sure_saniye,
    SUBSTRING(st.text, (qs.statement_start_offset/2)+1,
        ((CASE qs.statement_end_offset
            WHEN -1 THEN DATALength(st.text)
            ELSE qs.statement_end_offset END
            - qs.statement_start_offset)/2)+1) AS sql_text,
    qp.query_plan
FROM sys.dm_exec_query_stats AS qs
CROSS APPLY sys.dm_exec_sql_text(qs.sql_handle) AS st
CROSS APPLY sys.dm_exec_query_plan(qs.plan_handle) AS qp
ORDER BY qs.total_worker_time DESC;

```

sql_handle	execution_count	toplam_cpu_ms	toplam_sure_saniye	sql_text	query_plan
0x0200000009470F43029A71F8FD0F3062DA89572239149F26A0...	1	154280	190	SELECT OBJECT_NAME(i.object_id) AS TableName, i...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x0200000008604D91CA9F06799CE074F7300128092E060FD7...	17	152488	303	SELECT target_data FROM sys.dm_xe_session targ...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x02000000073088707696CE1D769EF33EEA30420BC967D37E...	1	116530	139	SELECT db_id() AS database_id, sm.[js_inlineable] ...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x020000000A89AC3784BFDEAC3131F07452A89ECFE8B034...	5	86450	86	SELECT udf.name AS [Name], udf.object_id AS [ID], udf cr...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x02000000017C0C514D261C5857504EF3C90BD18D0262453...	2	60816	88	SELECT SCHEMA_NAME(sp.schema_id) AS [Schema], sp...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x0200000007FC66090BCFA9CA236DBB95FC98D693CA4BE...	1	48059	59	SELECT db_id() AS database_id, c.system_type_id...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x0200000004067B0A35329594AE31B396523024097A05635A0...	2	31620	35	SELECT SCHEMA_NAME(id.schema_id) AS [Schema], u...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x020000000B1D4C10962D70F9A3E19D6D766282C0120B458D...	1	15706	15	select 1 as name, -- Temp DB case mtf type_desc w...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x0200000006F68472DCDEF218939F1D41A758F504CF6A73E1...	1	15568	18	insert into #tmp_sp_get_sqlagent_properties(auto_start, ms...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x020000000C204FB23C9CCDD1C87DF721D05EFAAF6A424...	1	14768	14	SELECT sp.name AS [Name], sp.object_id AS [ID], sp crea...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x020000000260B22946DAE2625FC2B5822730CA96E8DE521...	69	137693	13	SELECT clms.name AS [Name], clms.column_id AS [ID]...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x020000000024FE1E13BCB68AC3561D68420F69317C746EF...	1	12929	15	SELECT db_id() AS database_id, o.[type] AS object...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x02000000054807E08776C17369368D7ADE6383CA9FC89934...	1	12534	12	SELECT CASE WHEN name like '%msdtxpro.d%'...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x0200000008801B902D466ABF7D0F329665258CB800F76A8F3...	63	11751	253802	SELECT clms.name AS [Name], clms.column_id AS [ID]...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x0200000001FC20AD4496E6D1224D70B7121909772F6759F...	1	10437	10	select counter_name, cntr_value from sys.dm_os_performa...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x02000000020380D702A2CEFC358270CCE418CFAE8913419F...	1	10157	10	SET @JsonPerfCount = (select sum(cntr_value) from sys d...	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>
0x02000000039B1E24303AE8BDCB1A26642AC349BAAC135...	8	9064	157	SELECT O.OrderID, C.CompanyName, E.FirstName	<a href="#">ShowPlanXML xmlns="http://schemas.microsoft.com...</a>



Daha sonra izleme ve hata tespitine başlıyoruz. Bunu extended events ya da DMV sorguları ile yapabiliriz.

```
-- Extended events ile izleme

-- Oturumu oluşturalım
CREATE EVENT SESSION xe_nwind_slow_queries
ON SERVER
-- İzlemek istediğiniz event
ADD EVENT sqlserver.sql_batch_completed(
    WHERE duration > 5000000 -- 5 ms'den uzun süren batch'ler
)
-- Toplanan eventleri nereye yazacağımız
ADD TARGET package0.ring_buffer
WITH (
    MAX_MEMORY = 4096 KB,
    EVENT_RETENTION_MODE = ALLOW_SINGLE_EVENT_LOSS,
    MAX_DISPATCH_LATENCY = 30 SECONDS,
    TRACK_CAUSALITY = OFF,
    STARTUP_STATE = OFF
);
GO
-- Oturumu başlatalım
ALTER EVENT SESSION xe_nwind_slow_queries
ON SERVER
STATE = START;
GO
```

```
-- Tablolardaki indeks kırılma (fragmentation) oranlarını kontrol etme
SELECT
    OBJECT_NAME(ps.object_id) AS tablo,
    i.name AS indeks,
    ps.index_id,
    ps.avg_fragmentation_in_percent
FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'LIMITED') AS ps
JOIN sys.indexes AS i
    ON ps.object_id = i.object_id AND ps.index_id = i.index_id
WHERE ps.database_id = DB_ID()
    AND ps.avg_fragmentation_in_percent > 10
ORDER BY ps.avg_fragmentation_in_percent DESC;
```

100 %

Results Messages

	tablo	indeks	index_id	avg_fragmentation_in_percent
1	Orders	ShipPostalCode	9	75
2	Orders	CustomerID	2	66,6666666666667
3	Orders	CustomersOrders	3	66,6666666666667
4	Orders	ShippedDate	7	66,6666666666667
5	Orders	EmployeeID	4	50
6	Orders	EmployeesOrders	5	50
7	Orders	OrderDate	6	50
8	Orders	ShippersOrders	8	33,3333333333333
9	Customers	PK_Customers	1	33,3333333333333
10	OrderDetails	OrderID	2	20
11	OrderDetails	OrdersOrder_Details	3	20
12	OrderDetails	ProductID	4	20
13	OrderDetails	ProductsOrder_Details	5	20

Bu kırık indeksleri yeniden oluşturma (Rebuild) ve organize etme (Reorganize) de bu şekilde Cursor kullanarak yapalım:

```
-- Kırık indeksleri yeniden oluşturma (Rebuild) veya organize etme (Reorganize)
SET NOCOUNT ON;

DECLARE
    @SchemaName SYSNAME,
    @TableName SYSNAME,
    @IndexName SYSNAME,
    @Frag FLOAT;

DECLARE index_cursor CURSOR LOCAL FAST_FORWARD FOR
SELECT
    OBJECT_SCHEMA_NAME(ps.object_id, DB_ID()) AS SchemaName,
    OBJECT_NAME(ps.object_id, DB_ID()) AS TableName,
    i.name AS IndexName,
    ps.avg_fragmentation_in_percent AS Frag
FROM sys.dm_db_index_physical_stats(
    DB_ID(), NULL, NULL, NULL, 'LIMITED'
) AS ps
INNER JOIN sys.indexes AS i
ON ps.object_id = i.object_id
AND ps.index_id = i.index_id
WHERE ps.avg_fragmentation_in_percent > 10
AND i.type_desc <> 'HEAP'; -- HEAP için indeks yok
OPEN index_cursor;

FETCH NEXT FROM index_cursor
INTO @SchemaName, @TableName, @IndexName, @Frag;

WHILE @@FETCH_STATUS = 0
BEGIN
    IF @Frag > 30
    EXEC(
        'ALTER INDEX [' + @IndexName + '] ON ['
        + @SchemaName + '].[' + @TableName + '] REBUILD;');
    ELSE
    EXEC(
        'ALTER INDEX [' + @IndexName + '] ON ['
        + @SchemaName + '].[' + @TableName + '] REORGANIZE;');

    FETCH NEXT FROM index_cursor
    INTO @SchemaName, @TableName, @IndexName, @Frag;
END
CLOSE index_cursor;
DEALLOCATE index_cursor;
```

İndeks yönetimi için hiç kullanılmayan indeksleri inceleyelim:

```
-- Hiç kullanılmayan indeksleri tespit etme
SELECT
    OBJECT_NAME(s.object_id) AS tablo,
    i.name AS indeks,
    s.user_seeks + s.user_scans + s.user_lookups + s.user_updates AS kullanim_sayisi
FROM sys.dm_db_index_usage_stats AS s
JOIN sys.indexes AS i
ON s.object_id = i.object_id AND s.index_id = i.index_id
WHERE database_id = DB_ID()
AND s.user_seeks + s.user_scans + s.user_lookups = 0
AND i.is_primary_key = 0
ORDER BY kullanim_sayisi ASC;
```

	tablo	indeks	kullanim_sayisi
	OrderDetails	ProductsOrder_Details	0
	Suppliers	PostalCode	0
	Suppliers	CompanyName	0
	Products	SuppliersProducts	0
	Products	ProductName	0
	Products	CategoryID	0
	Employees	PostalCode	0
	Employees	LastName	0
	Categories	CategoryName	0
0	Orders	EmployeeID	2
1	Orders	OrderDate	2
2	Orders	ShippedDate	2
3	Orders	ShipPostalCode	2

Optimizasyon açısından kullanılmayan indeksler DROP'lanabilir.

Disk yönetiminde dosya boyutlarına ve istatistik güncellemelerine bakalım:

```
-- Veri tabanı ve dosya boyutlarını kontrol etme
EXEC sp_spaceused; -- genel özet
```

	database_name	database_size	unallocated space
1	NorthwindDB	144.00 MB	62.38 MB

	reserved	data	index_size	unused
1	9856 KB	2512 KB	2440 KB	4904 KB

Her bir dosyanın fiziksel boyutuna bakalım:

```
SELECT
    name AS mantıksal_ad,
    physical_name,
    size/128.0 AS boyut_MB,
    growth/128.0 AS büyüme_adimi_MB
FROM sys.master_files
WHERE database_id = DB_ID();
```

	mantıksal_ad	physical_name	boyut_MB	büyüme_adimi_MB
1	NorthwindDB	C:\Program Files\Microsoft SQL Server\MSSQL16.MSS...	72.000000	64.000000
2	NorthwindDB_log	C:\Program Files\Microsoft SQL Server\MSSQL16.MSS...	72.000000	64.000000

```
-- İstatistik güncelleme ile sorgu planlarını tazeleme
EXEC sp_updatestats;
```

Updating [dbo].[Shippers]  
[PK\_Shippers], update is not necessary...  
[\_WA\_Sys\_00000002\_59FA5E80], update is not necessary...  
0 index(es)/statistic(s) have been updated, 2 did not require update.

Updating [dbo].[Suppliers]  
[PK\_Suppliers] has been updated...  
[CompanyName] has been updated...  
[PostalCode] has been updated...  
3 index(es)/statistic(s) have been updated, 0 did not require update.

Updating [dbo].[Orders]  
[PK\_Orders] has been updated...  
[CustomerID], update is not necessary...  
[CustomersOrders], update is not necessary...  
[EmployeeID], update is not necessary...  
[EmployeesOrders], update is not necessary...  
[OrderDate], update is not necessary...  
[ShippedDate], update is not necessary...  
[ShippersOrders], update is not necessary...  
[ShipPostalCode], update is not necessary...  
[\_WA\_Sys\_0000000B\_5DCAEF64] has been updated...  
[\_WA\_Sys\_00000008\_5DCAEF64], update is not necessary...  
2 index(es)/statistic(s) have been updated, 9 did not require update.

Updating [dbo].[Products]  
[PK\_Products], update is not necessary...  
[CategoriesProducts] has been updated...

```
-- Belirli bir tablo için tam istatistik güncelleme
UPDATE STATISTICS dbo.Categories WITH FULLSCAN;
```

Burada da row bazlı veri sıkıştırma örneği var. Daha büyük bir database üzerinde partition ile de disk yönetimi sağlanabilir.

```
-- Tablo veri sıkıştırma
ALTER TABLE dbo.Orders
REBUILD WITH (DATA_COMPRESSION = ROW);
```

Bizim database'imiz yeterince büyük olmadığı için DMV bu sorguya bir şey öneremedi ama bu şekilde indeks önerisi alınabilir:

```
-- Filtreli ya da kapsayıcı (covering) indeks önerisi almak için DMV
SELECT TOP 20
    DB_NAME(mid.database_id) AS DatabaseName,
    OBJECT_NAME(mid.object_id, mid.database_id) AS TableName,
    mid.equality_columns AS EqualityColumns,
    mid.inequality_columns AS InequalityColumns,
    mid.included_columns AS IncludedColumns,
    migs.unique_compiles AS CompileCount,
    migs.user_seeks AS UserSeeks,
    migs.user_scans AS UserScans,
    migs.avg_total_user_cost AS AvgTotalCost,
    migs.avg_user_impact AS AvgUserImpact
FROM sys.dm_db_missing_index_details AS mid
JOIN sys.dm_db_missing_index_groups AS mig
    ON mid.index_handle = mig.index_handle
JOIN sys.dm_db_missing_index_group_stats AS migs
    ON mig.index_group_handle = migs.group_handle
ORDER BY migs.avg_user_impact DESC;
```

Yeni bir rol oluşturup ona yetki atama ve bunu bir kullanıcıya tanımlama da bu şekilde yapılıyor:

```
-- Yeni bir rol oluşturma
CREATE ROLE db_DataAnalyst;

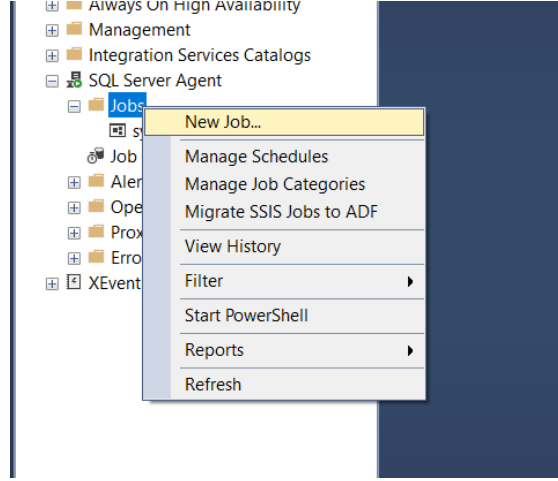
-- Role SELECT izni verme
GRANT SELECT ON SCHEMA :: dbo TO db_DataAnalyst;

-- Role EXECUTE izni verme (stored procedure'lar için)
GRANT EXECUTE ON SCHEMA :: dbo TO db_DataAnalyst;

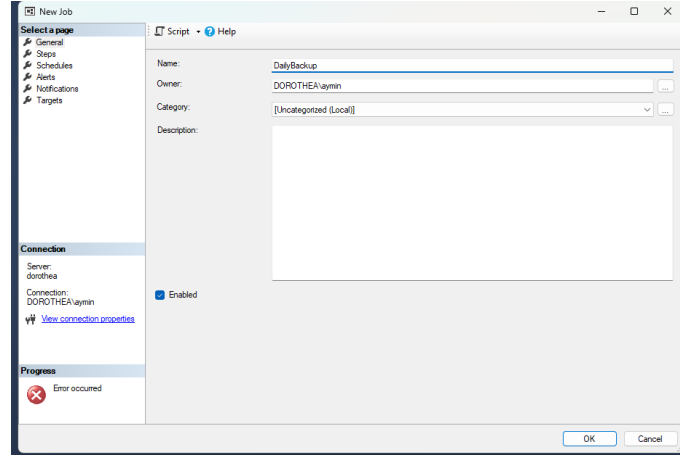
-- Bireysel kullanıcıyı role atama
ALTER ROLE db_DataAnalyst ADD MEMBER aymin;
```

### 3. Veritabanı Yedekleme ve Otomasyon Çalışması

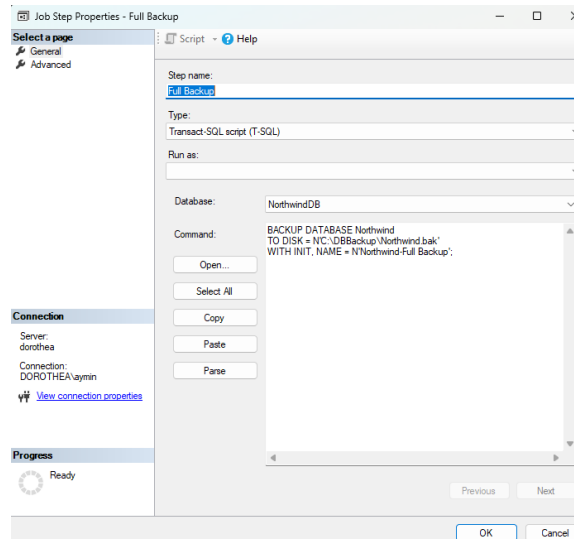
Öncelikle SQL Server üzerinde yeni bir job oluşturuyoruz.



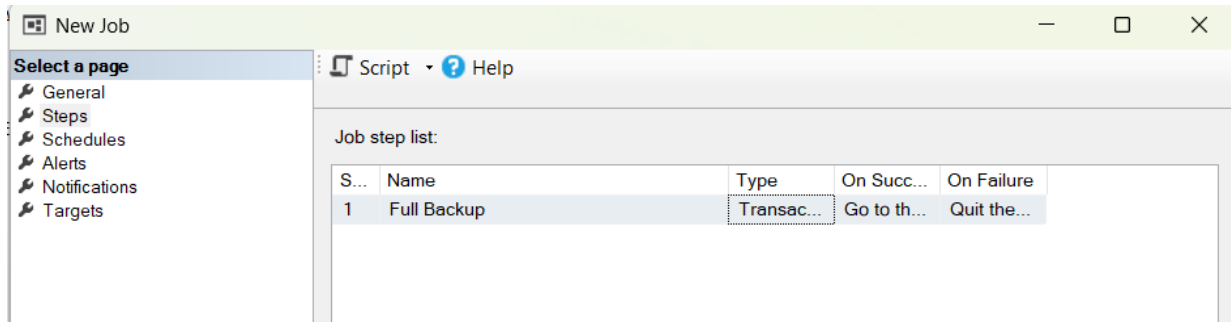
Daha sonra job adımızı general sekmesinden veriyoruz:



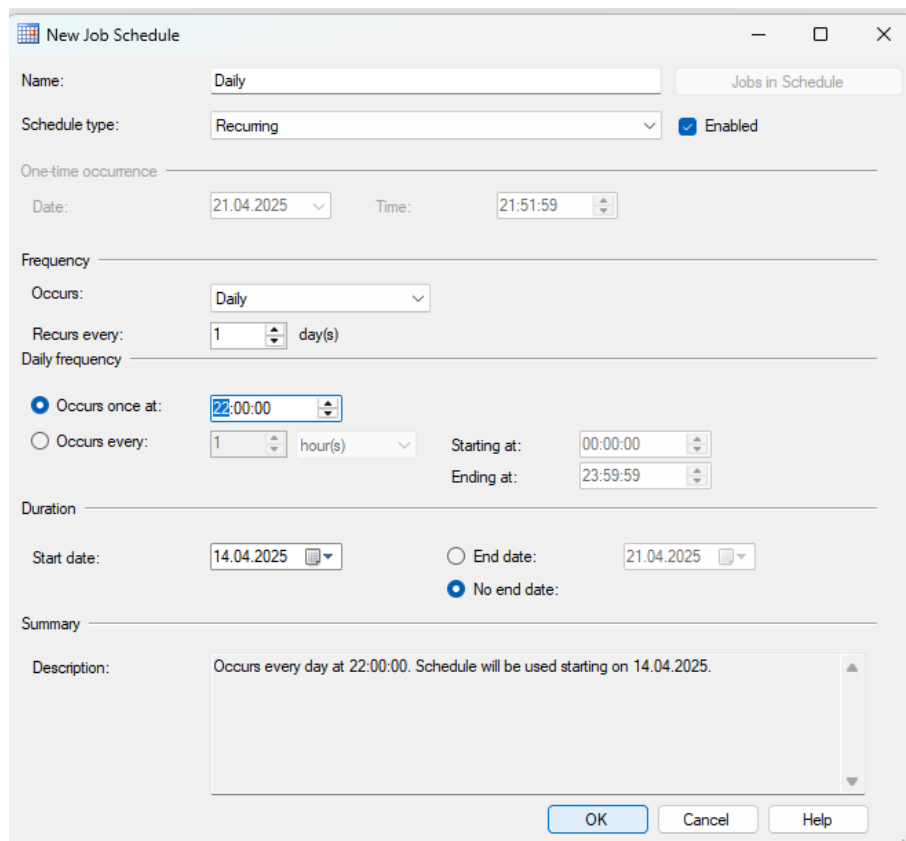
Sonra Step sekmesinden Command'imizi hangi database üzerinde çalışmasını istiyorsak belirtiyoruz.



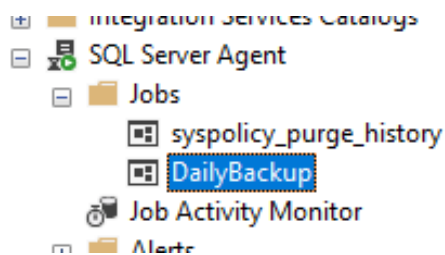
Step'imizi oluřturunca böyle g r n y r:



Daha sonra da yedeklememizin otomatik ger ekleřmesi i in Schedule sekmesinden ona bir zamanlama atıyoruz. G nl k saat her 22.00 olduėunda full backup yapacak bir schedule  rneėi :



Bu yedekleme otomasyonumuz oluřtuėunda da böyle g r n y r:



Daha sonra gelişmiş ayarları ve database mail xps'i aktifleştirip daha önce ürettiğimiz profillere ve maillere bakıyoruz:

```
-- Gelişmiş ayarlar
EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;

-- Database Mail XPs'i açalım
EXEC sp_configure 'Database Mail XPs', 1;
RECONFIGURE;

USE msdb;
GO

-- Var olan profiller
SELECT name, description
FROM sysmail_profile;

-- Var olan hesaplar
SELECT name, email_address
FROM sysmail_account;
```

Database Mail profilini seç

100 %

Results Messages

	name	description
1	aymina	DBA raporları için
2	Aymin	DBA raporları için
3	Aymina2	DBA raporları için
4	Aymina4	DBA raporları için

Sonra yeni bir database profili oluşturuyoruz:

```
-- Database Mail profilini seç
EXEC msdb.dbo.sysmail_add_profile_sp
    @profile_name = 'Aymin',
    @description = 'DBA raporları için';

EXEC msdb.dbo.sysmail_add_account_sp
    @account_name = 'DBMailAccount',
    @description = 'SQL Server raporlama mail hesabı',
    @email_address = 'ayminayilik@gmail.com',
    @display_name = 'SQL Rapor',
    @mailserver_name = 'smtp.domain.com';

EXEC msdb.dbo.sysmail_add_profileaccount_sp
    @profile_name = 'Aymin',
    @account_name = 'Aymin',
    @sequence_number = 1;
```

Ve test etmek için bir querynin rapor mailini gönderiyoruz:

```
-- Raporu mail ile gönder
EXEC msdb.dbo.sp_send_dbmail
    @profile_name = 'Aymin',
    @recipients = 'ayminayilik@gmail.com',
    @subject = 'Günlük Yedek Raporu',
    @body = 'Aşağıda son 24 saatte alınan yedekler yer almaktadır.',
    @body_format = 'HTML',
    @query = N'
        SELECT
            database_name,
            backup_start_date,
            backup_finish_date,
            CASE type WHEN ''D'' THEN ''Full''
                  WHEN ''I'' THEN ''Diff''
                  WHEN ''L'' THEN ''Log''
            END AS backup_type,
            backup_size/1024/1024 AS size_mb,
            physical_device_name
        FROM msdb.dbo.backupset bs
        JOIN msdb.dbo.backupmediafamily mf
            ON bs.media_set_id = mf.media_set_id
        WHERE backup_start_date >= DATEADD(day,-1,GETDATE())
        ORDER BY backup_start_date DESC;
    ',
    @attach_query_result_as_file = 0;
```

100 %

Messages

Mail (Id: 9) queued.

#### 4. Veritabanı Güvenliği ve Erişim Kontrolü

##### Erişim Yönetimi:

Farklı görevleri yapacak kullanıcılar oluşturularak, veritabanı üzerinde hangi kullanıcının hangi yetkilere sahip olacağını belirlemek güvenlik açısından önemlidir.

Server Kimlik Doğrulama modunu SQL Server ve Windows yapıyoruz.

Server authentication

☐ Windows Authentication mode

☒ SQL Server and Windows Authentication mode

Aşağıdaki görselde “reader” kullanıcısını oluşturuyoruz ve giriş bilgilerini SQL Server Authentication ile belirliyoruz.

Login - new

Select a page

- General
- Server Roles
- User Mapping
- Securables
- Status

Script Help

Login name: reader Search...

☐ Windows authentication

☐ Microsoft Entra ID authentication

☒ SQL Server authentication

Password: .....

Confirm password: .....

☐ Specify old password

Old password: .....

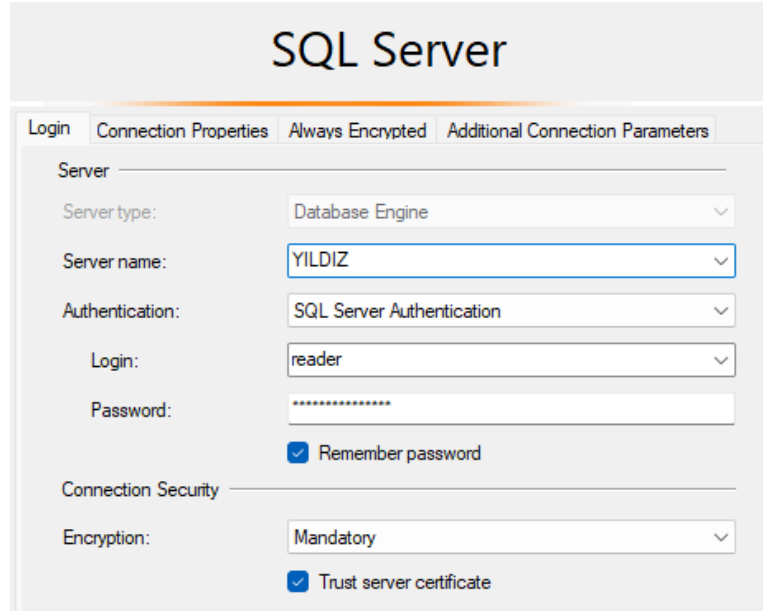
☒ Enforce password policy

☒ Enforce password expiration

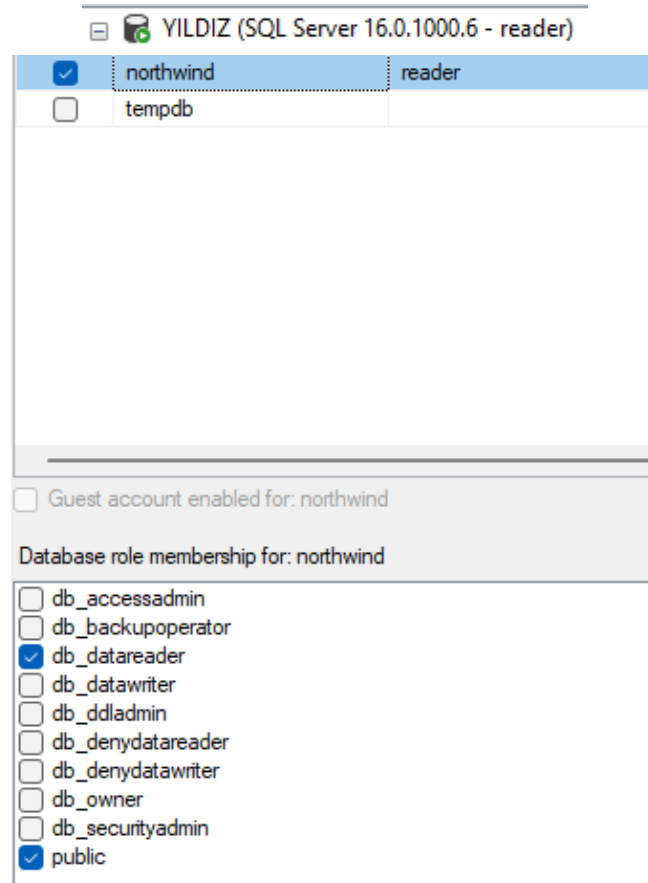
☒ User must change password at next login



SQL Server Login ekranında, oluşturulan bilgiler ile giriş yapıyoruz.



The image shows the SQL Server Login dialog box. It has four tabs: Login, Connection Properties, Always Encrypted, and Additional Connection Parameters. The Login tab is selected. The 'Server' section contains the following fields: 'Server type' (Database Engine), 'Server name' (YILDIZ), 'Authentication' (SQL Server Authentication), 'Login' (reader), and 'Password' (masked with dots). There is a 'Remember password' checkbox which is checked. The 'Connection Security' section contains the 'Encryption' field (Mandatory) and a 'Trust server certificate' checkbox which is checked.



The image shows a screenshot of the SQL Server Enterprise Manager. The title bar reads 'YILDIZ (SQL Server 16.0.1000.6 - reader)'. The left pane shows a tree view with 'northwind' and 'tempdb' databases. The 'northwind' database is selected. The right pane shows the 'Database role membership for: northwind' section. It lists several roles with checkboxes: db\_accessadmin, db\_backupoperator, db\_datareader (checked), db\_datawriter, db\_ddladmin, db\_denydatareader, db\_denydatawriter, db\_owner, db\_securityadmin, and public (checked). Below this list, there is a checkbox for 'Guest account enabled for: northwind' which is unchecked.

SQL Server Authentication ile giriş yapan ve “db\_datareader” rolüne sahip kullanıcı.  
(Sadece SELECT iznine sahiptir.)

```
INSERT INTO Suppliers (CompanyName, ContactName) VALUES ('Ankara LTD STI', 'Ali Vuralan');
```

100 %

Messages

Msg 229, Level 14, State 5, Line 8  
The INSERT permission was denied on the object 'Suppliers', database 'northwind', schema 'dbo'.

Completion time: 2025-05-29T21:16:59.4329979+03:00

```
SELECT * from Suppliers;
```

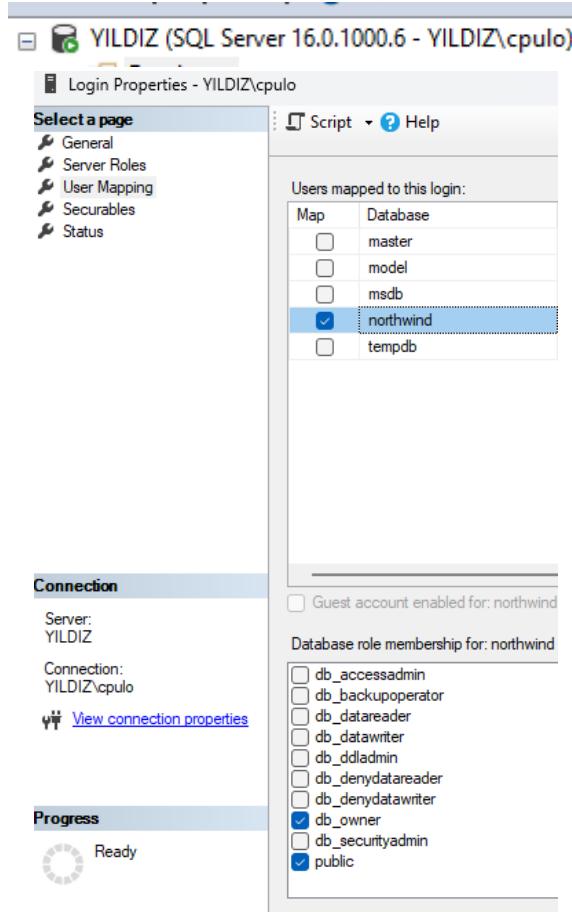
```
INSERT INTO Suppliers (CompanyName, ContactName) VALUES ('Ankara LTD STI', 'Ali Vuralan');
```

00 %

Results Messages

	SupplierID	CompanyName	ContactName	ContactTitle	Address
1	1	Exotic Liquids	Charlotte Cooper	Purchasing Manager	49 Gilbert St.
2	2	New Orleans Cajun Delights	Shelley Burke	Order Administrator	P.O. Box 78934
3	3	Grandma Kelly's Homestead	Regina Murphy	Sales Representative	707 Oxford Rd.

Bu kullanıcı ile veritabanına örnekte olduğu gibi INSERT, DELETE vb. işlemleri yapamıyoruz. Sadece SELECT işlemini yapabiliyoruz.



Windows Authentication ile giriş yapan ve “db\_owner” rolüne sahip kullanıcı. (Veritabanındaki tüm izinlere sahiptir.)

```
INSERT INTO Suppliers (CompanyName, ContactName) VALUES ('Ankara LTD STI', 'Ali Vuralan');

100 %
Messages

(1 row affected)

Completion time: 2025-05-29T20:50:14.6000936+03:00
```

Bu kullanıcı ile veritabanına örnekte olduğu gibi INSERT, DELETE vb. tüm işlemleri yapabiliriz.

### Veri Şifreleme:

Veritabanındaki verilerin fiziksel olarak şifrelemek (diskte .mdf, .ldf, .bak); disk çalınması, yetkisiz disk erişimi gibi durumlara karşı güvenlik sağlar. Bunu TDE (Transparent Data Encryption) ile yapabiliriz. Sertifika mutlaka güvenli bir yerde saklanmalıdır.

Bunun için öncelikle Database Master Key (DMK) gereklidir. Şifreleme zincirinin temelidir. Sertifikaları şifrelemek için gereklidir. Ve tam tersi sertifikalar da şifreleri korur.

```
USE master;
GO

CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Password123';
GO
```

Şimdi de TDE sertifikasını oluşturalım.

```
CREATE CERTIFICATE TDECert
WITH SUBJECT = 'Veri şifreleme için sertifika';
GO
```

Şimdi şifrelenecek veritabanına geçip, şifreleme algoritmasını ve şifrelenecek sertifikayı belirterek Database Encryption Key (DEK) oluşturalım.

```
USE northwind;
GO

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

Şimdi ise TDE'yi aktif edelim.

```
ALTER DATABASE northwind
SET ENCRYPTION ON;
```

Aktif mi diye kontrol ediyoruz. (State: 0 - DEK yok. | State: 3 - Veritabanı şifrelenmiş.)

	DatabaseName	encryption_state	percent_complete	key_algorithm	key_length
1	tempdb	3	0	AES	256
2	northwind	3	0	AES	256

Sertifikayı yedeklemek de önemli bir işlemdir.

```

BACKUP CERTIFICATE TDECert
TO FILE = 'C:\SQL2022\Backup\TDECert.cer'
WITH PRIVATE KEY (
    FILE = 'C:\SQL2022\Backup\TDECertPrivateKey.pvk',
    ENCRYPTION BY PASSWORD = 'Backup123'
);

```

Ad	Değiştirme tarihi	Tür	Boyut
TDECert	30.05.2025 01:37	Güvenlik Sertifikası	2 KB
TDECertPrivateKey.pvk	30.05.2025 01:37	PVK Dosyası	2 KB

(Bu işlemi yaparken, yedeklerin kaydolacağı dosyaya, güvenlik ayarlarından MSSQLSERVER’a gerekli izinlerin verilmesi gerekmektedir.)

### SQL Injection Testleri:

Kullanıcıdan alınan giriş bilgileri eğer doğru şekilde işlenmezse, bu durum güvenlik problemine yol açabilir. Bu şekilde veritabanını manipüle etmeye çalışıp değerli bilgilere erişimi engellemek gerekmektedir.

Bir örnek tablo üzerinden durumu değerlendirelim.

```

CREATE TABLE LoginTest (
    ID INT IDENTITY(1,1) PRIMARY KEY,
    Username NVARCHAR(100),
    Password NVARCHAR(100)
);

INSERT INTO LoginTest (Username, Password) VALUES ('admin', 'admin123');

```

Yukarıda kullanıcının bilgilerini oluşturduk ve aşağıda ise değişkenlere girilen bilgiler ile kullanıcının giriş yapması istendi. Eğer “username = 'admin' --” vb. bir şekilde girilirse SQL “--” ifadenin devamını yorum satırı olarak kabul edecek ve devamındaki sorgu (“password” kısmı) geçersiz kalacak. Örnekte görüldüğü üzere şifre yanlış olduğu halde sonuç kısmında “admin” kullanıcısının bilgileri gözükmemektedir.

```

DECLARE @username2 NVARCHAR(100) = 'admin' --';
DECLARE @password2 NVARCHAR(100) = 'yanlisSifre';

DECLARE @sql NVARCHAR(MAX);

SET @sql = 'SELECT * FROM LoginTest WHERE Username = ''' + @username2 + ''' AND Password = ''' + @password2 + '''';

PRINT @sql;

EXEC(@sql);

```

ID	Username	Password
1	admin	admin123

Bu gibi durumlardan kaçınmak için Stored Procedure kullanılabilir. Bu sayede sorgular direkt string olarak birleştirilmez ve dinamik ve daha güvenli hale getirilir. Injection engellenir.

```
CREATE PROCEDURE sp_LoginSecure
    @Username NVARCHAR(100),
    @Password NVARCHAR(100)
AS
BEGIN
    SELECT * FROM LoginTest
    WHERE Username = @Username AND Password = @Password;
END;

EXEC sp_LoginSecure @Username = 'admin' --', @Password = 'abc';
```

100 %

Results Messages

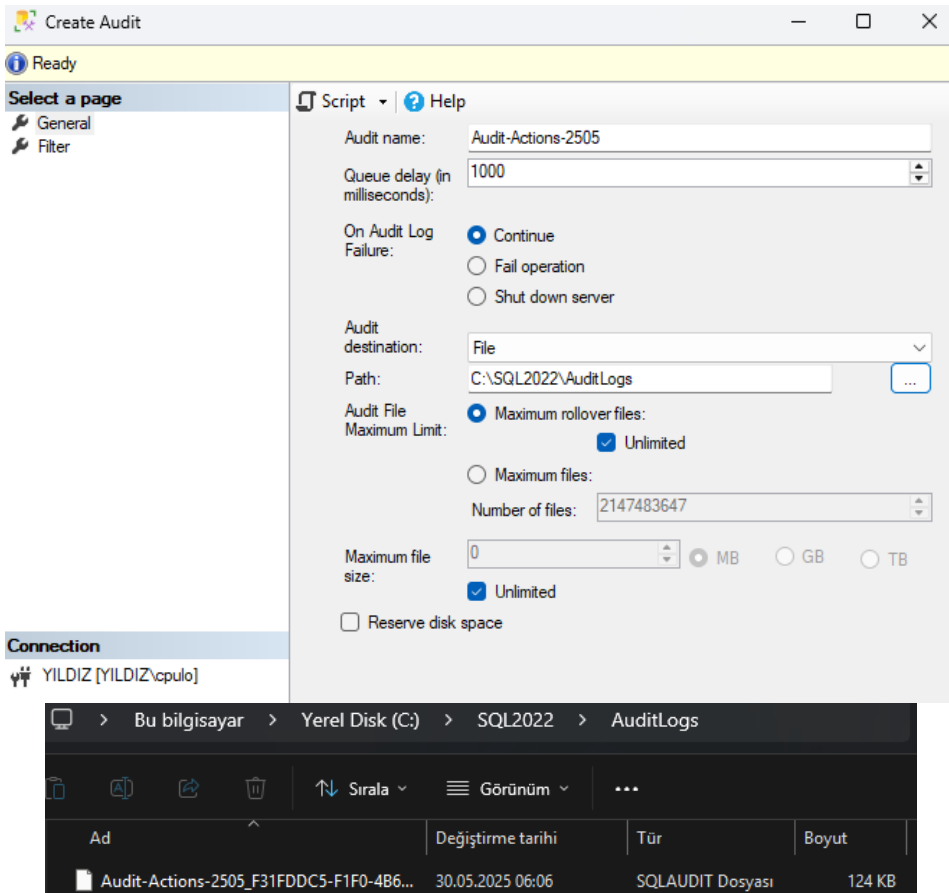
ID	Username	Password
----	----------	----------

Yukarıdaki örnekte gözüktüğü üzere sonuç boş dönmektedir çünkü "password" kontrol edilebilir durumdadır. Bu sayede güvenlik artırılmış ("OR 1=1" gibi saldırılar engellemiş.), kod okunabilirliği düzgün hale getirilmiş ve performansa katkıda sağlanılmıştır.

### Audit Logları:

Bu log'lar sayesinde SQL Server'da kimin, ne zaman ve ne yaptığını izleyebiliriz. Kullanıcı aktivitelerini kaydetmemizi sağlar. Bu güvenlik, uyumluluk vb. amaçlar için önemlidir.

Security > Audits > New Audit kısmından Audit oluşturulur. Bu işlem genel log dosyası oluşturmak içindir.



Şimdi sunucu düzeyinde, giriş bilgileri vb. hakkında log'lar tutan Audit nesnemizi oluşturalım.

Name: ServerAuditSpecification-20250530-060207

Audit: Audit-Actions-2505

Actions:

	Audit Action Type	Object Class	Object Schema	Object Name	Principal Name
1	FAILED_LOGIN_GROUP				
2	SUCCESSFUL_LOGIN_GROUP				
3					

Aşağıda ise bu durumun kayıtları gözükmemektedir. (İstenilen bilgiler ayrıca yazılmıştır.)

```
SELECT action_id, succeeded, server_principal_name, session_id, application_name, event_time
FROM sys.fn_get_audit_file('C:\SQL2022\AuditLogs\*.sqlaudit', DEFAULT, DEFAULT);
```

	action_id	succeeded	server_principal_name	session_id	application_name	event_time
28	LGIS	1	reader	75	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.010
29	LGIS	1	reader	74	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.011
30	LGIS	1	reader	72	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.011
31	LGIS	1	reader	74	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.012
32	LGIS	1	reader	65	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.117
33	LGIS	1	reader	65	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.151
34	LGIS	1	reader	65	Microsoft SQL Server Management Studio	2025-05-30 03:03:05.159
35	LGIF	0	reader	0	Microsoft SQL Server Management Studio	2025-05-30 03:03:10.813
36	LGIS	1	YILDIZ\cpulo	65	Microsoft SQL Server Management Studio	2025-05-30 03:03:15.033
37	LGIS	1	YILDIZ\cpulo	70	Microsoft SQL Server Management Studio	2025-05-30 03:03:15.094
38	LGIS	1	YILDIZ\cpulo	70	Microsoft SQL Server Management Studio	2025-05-30 03:03:15.099

Şimdi ise veritabanı için log'ları tutacak Audit nesnemizi oluşturalım.

Script Help

Name: DatabaseAuditSpecification-202505

Audit: Audit-Actions-2505

Actions:

	Audit Action Type	Object Class	Object Schema	Object Name	Principal Name
1	SELECT	OBJECT	dbo	Customers	db_owner
2	INSERT	OBJECT	dbo	Customers	db_owner

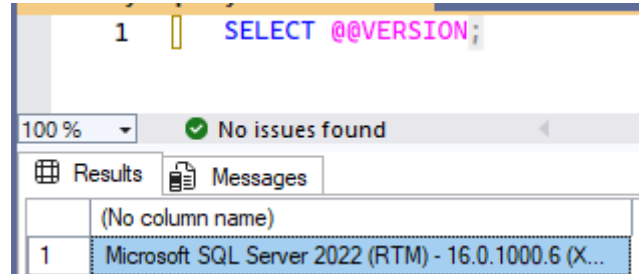
(Burada Action Type istenildiği gibi artırılabilir. İstenilen tabloya göre ayarlar yapılabilir.)

Aşağıda veritabanı hakkındaki log'larımız gözükmemektedir. Hangi veritabanı için, kim, neler yapmış, tarih vb. bilgilere ulaşabiliyoruz.

	action_id	statement	object_name	database_name	server_principal_name	event_time
1	AUSC				YILDIZ\cpulo	2025-05-30 02:33:17.0801355
2	SL	SELECT * FROM Customers WHERE CustomerID='ANATR'	Customers	northwind	YILDIZ\cpulo	2025-05-30 02:57:48.0136203
3	IN	INSERT INTO Customers (CustomerID, CompanyName, C...	Customers	northwind	YILDIZ\cpulo	2025-05-30 02:57:49.9260723
4	SL	SELECT * FROM Customers	Customers	northwind	YILDIZ\cpulo	2025-05-30 02:57:54.9454511

## 5. Veritabanı Yükseltme ve Sürüm Yönetimi

Öncelikle mevcut durum analizi yapalım. Yeni bir sürüme yükseltme öncesi, mevcut ortamın ayrıntılı bir envanterini çıkarmak, hangi sürümde çalıştığımızı ve veritabanılarının hangi uyumluluk düzeyinde olduğunu kesin olarak bilmek için Sql Server sürümümüzü ve veritabanı uyumluluk düzeylerimizi kontrol edelim.



Her bir veritabanı için `compatibility_level` değeri, çalışma zamanındaki dilbilgisi davranışlarını ve bazı optimizasyonları etkiler; dolayısıyla yeni sürümde oluşabilecek farklılıkları önceden hesaba katmamıza imkân tanır.

The screenshot shows a SQL query window with the command `SELECT name, compatibility_level FROM sys.databases;`. Below the query, the status bar indicates '100 %' and 'No issues found'. The 'Results' tab is active, showing a table with 2 columns: 'name' and 'compatibility\_level'. The table contains 5 rows of data.

	name	compatibility_level
1	master	160
2	tempdb	160
3	model	160
4	msdb	160
5	Northwind	160

Eski sürümden yeni sürüme geçişte hangi bileşenlerin ve özelliklerin etkileneceğini, hangi adımların atılacağını net biçimde tanımlayarak bir yükseltme stratejisi oluşturalım.

### Yöntem:

- Tüm veritabanı nesnelerini (tablo, view, stored procedure, fonksiyon vb.) envanterleyerek, deprecated (kaldırılmış) veya davranış değiştirmiş özellikleri belirlenmeli
- Uygulama katmanında, veritabanı bağlantı kitaplıklarının (ODBC, OLE DB, ADO.NET vs.) ve sorgu sentakslarının yeni sürümle uyumlu olup olmadığını test edilmeli.
- Performans karşılaştırmaları için, kritik rapor ve işlem setlerini hem eski hem yeni sürümde çalıştırarak sonuç ve süre farklarını ölçülmeli

Bu planın ardından üretim verilerinin eksiksiz bir güvenlik kopyasını oluşturmak ve yükseltme öncesi geri dönülebilir bir nokta sağlamak için tam yedek alalım. Daha önceki projelerde nasıl yapıldığını anlatmıştık.

New Job Step

Select a page: General, Advanced

Script Help

Step name: FullBUJ

Type: Transact-SQL script (T-SQL)

Run as:

Database: Northwind

Command: BACKUP DATABASE Northwind TO DISK = 'C:\Backups\NorthDB\_Full.bak' WITH FORMAT, INIT, NAME = 'Full Backup';

Open... Select All Copy Paste Parse

Connection: Server: thea, Connection: THEA\aymin, View connection properties

Progress: Error occurred

Previous Next OK Cancel

Tek seferlik yedek alınması için bunu seçiyoruz:

New Job Schedule

Name: One time Jobs in Schedule

Schedule type: One time Enabled

One-time occurrence: Date: 26.05.2025 Time: 21:42:09

The one-time occurrence date and time must be greater than the current date and time.

Frequency: Occurs: Monthly

Day: 1 of every 1 month(s)

The: first Monday of every 1 month(s)

Daily frequency: Occurs once at: 00:00:00

Occurs every: 1 hour(s) Starting at: 00:00:00 Ending at: 23:59:59

Duration: Start date: 29.05.2025 End date: 29.05.2025 No end date

Summary: Description: Occurs on 26.05.2025 at 21:42:09.

This PC > Local Disk (C:) > Backups

Name	Date modified	Type	Size
NorthDB_Full.bak	29.05.2025 21:42	BAK File	8.248 KB



Bunu test ortamında restore edebiliriz:

1	RESTORE FILELISTONLY
2	FROM DISK = N'C:\Backups\North08_Full.bak';

LogicalName	PhysicalName	Type	FileGroupName	Size	MaxSize	Field	CreateLSN	DropLSN	Uniqued	ReadOnlyLSN	ReadWriteLSN	BackupSizeInBytes	SourceBlockSize	FileGroupId	LogGroup
1	Northwind	D	PRIMARY	8388608	35184372080640	1	0	0	6F8C22E2-26E3-4B51-B67A-23EB5E39CF6F	0	0	8192000	4096	1	NULL
2	Northwind_Jog	L	NULL	75497472	2199023255552	2	0	0	27765892-1DF8-43C7-9EAF-74CDAB439F42	0	0	0	4096	0	NULL

Şema değişikliklerinin sürüm kontrolü de DDL trigger tabanlı değişiklik kaydı ile yapılır ve üretim veritabanı üzerinde doğrudan yapılan şema değişikliklerini gerçek zamanlı olarak izleme ve loglamayı sağlar.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure, including the 'Northwind' database and its tables. The right pane shows the SQL Query window with the following code:

```
-- 1. Audit Schema Tablosu
CREATE TABLE dbo.SchemaChangeLog
(
    ChangeID INT IDENTITY(1,1) PRIMARY KEY,
    EventTime DATETIME2 NOT NULL DEFAULT SYSUTCDATETIME(),
    EventType NVARCHAR(100) NOT NULL,
    ObjectName NVARCHAR(255),
    SqlCommand NVARCHAR(MAX),
    LoginName NVARCHAR(128),
    HostName NVARCHAR(128)
);

-- 2. DDL Trigger
CREATE TRIGGER trg_DDL_SchemaChange
ON DATABASE
AFTER DDL_DATABASE_LEVEL_EVENTS
AS
BEGIN
    SET NOCOUNT ON;

    INSERT INTO dbo.SchemaChangeLog(EventType, ObjectName, SqlCommand, LoginName, HostName)
    SELECT
        EVENTDATA().value('(/EVENT_INSTANCE/EventType)[1]', 'NVARCHAR(100)'),
        EVENTDATA().value('(/EVENT_INSTANCE/ObjectName)[1]', 'NVARCHAR(255)'),
        EVENTDATA().value('(/EVENT_INSTANCE/SQLCommand/CommandText)[1]', 'NVARCHAR(MAX)'),
        ORIGINAL_LOGIN(),
        HOST_NAME();
END;
```

The bottom of the screenshot shows the Messages pane with the message: "Commands completed successfully."

Bunu transaction bloğu ile deneyebiliriz. Şema veya veri güncellemeleri yapmadan önce işlemlerin tamamını veya hiçbirini uygulayarak güvenli bir test ortamı yaratmak için kullanılır. Değişikliği kalıcı yapmak için commit geri almak için rollback kullanılır.

```
SQLQuery2.sql (aymin (75))
1 BEGIN TRANSACTION;
2     -- Şema değişiklikleri
3     ALTER TABLE Orders ADD OrderStatus TINYINT NOT NULL DEFAULT(0);
4     -- Diğer ALTER / CREATE komutları
5     -- Sorun yoksa:
6     COMMIT;
7     -- Sorun varsa:
8     ROLLBACK;
```

100 % No issues found

Messages

Commands completed successfully.

Bunu örnek bir senaryoda test edelim. TestDB üzerinde gösterelim.

```
Connect THEA (SQL Server 16.0.1000.6 - thea\aymin)
Databases
  System Databases
  Database Snapshots
  Northwind
  Restored_DB
  TestDB
    Database Diagrams
    Tables
      System Tables
      FileTables
      External Tables
      Graph Tables
      dbo.Orders
      Dropped Ledger Tables
    Views
      System Views
      Dropped Ledger Views
    External Resources
    Synonyms
    Programmability
    Query Store
    Service Broker
    Storage
    Security
  Security
  Server Objects
  Replication
```

```
1 USE master;
2 GO
3 IF DB_ID('TestDB') IS NULL
4     CREATE DATABASE TestDB;
5 GO
6 USE TestDB;
7 GO
8
9 IF OBJECT_ID('dbo.Orders', 'U') IS NOT NULL
10     DROP TABLE dbo.Orders;
11 GO
12 CREATE TABLE dbo.Orders
13 (
14     OrderID INT IDENTITY(1,1) PRIMARY KEY,
15     OrderDate DATETIME2 NOT NULL DEFAULT SYSUTCDATETIME(),
16     OrderStatus TINYINT NOT NULL DEFAULT(0)
17 );
18 GO
19
20 -- Örnek birkaç satır ekleyelim
21 INSERT INTO dbo.Orders DEFAULT VALUES;
22 INSERT INTO dbo.Orders DEFAULT VALUES;
23 SELECT * FROM dbo.Orders;
24
```

100 % 1 0

Results Messages

	OrderID	OrderDate	OrderStatus
1	1	2025-05-29 23:10:52.2199305	0
2	2	2025-05-29 23:10:52.2199305	0

```

29 BEGIN TRANSACTION;
30 ALTER TABLE dbo.Orders
31 ADD ShipmentDate DATETIME2 NULL;
32
33 SELECT
34 COLUMN_NAME, DATA_TYPE
35 FROM INFORMATION_SCHEMA.COLUMNS
36 WHERE TABLE_NAME = 'Orders';
37

```

100 % No issues found

	COLUMN_NAME	DATA_TYPE
1	OrderID	int
2	OrderDate	datetime2
3	OrderStatus	tinyint
4	ShipmentDate	datetime2

```

38
39 ROLLBACK TRANSACTION;
40 GO
41
42 -- Değişiklik geri alındı mı kontrol edelim
43 SELECT
44 COLUMN_NAME
45 FROM INFORMATION_SCHEMA.COLUMNS
46 WHERE TABLE_NAME = 'Orders';
47

```

100 % No issues found

	COLUMN_NAME
1	OrderID
2	OrderDate
3	OrderStatus

Snapshot ile anlık görüntü alma da aynı şekilde anlık görüntüyü hızlıca alıp hata durumunda çalışır noktaya dönebilmek için kullanılır.

SQLQuery2.sql... \aymin (73)\*

```

1 -- Snapshot Oluşturma
2 CREATE DATABASE Northwind_Snap
3 ON
4 (NAME = Northwind, FILENAME = 'C:\Snapshots\Northwind_Snap.ss')
5 AS SNAPSHOT OF Northwind;

```

100 % No issues found

Messages

Commands completed successfully.

```

-- Değişiklikler yapıldıktan sonra geri dönme:
RESTORE DATABASE Northwind_Snap FROM DATABASE_SNAPSHOT = 'Northwind_Snap';

```

## 6. Veri Temizleme ve ETL Süreçleri Tasarımı

Eksik veya hatalı verileri düzenlemek sorgular ve genel işleyiş açısından önemli bir konudur. NULL, yanlış format veya tutarsız veriler olabilir.

Örnek bir NULL veri kontrolü:

```
SELECT * FROM Customers
WHERE Address IS NULL OR Fax IS NULL;
```

	Address	City	Region	PostalCode	Country	Phone	Fax
1	Mataderos 2312	México D.F.	NULL	05023	Mexico	(5) 555-3932	NULL
2	Fauntleroy Circus	London	NULL	EC2 5NT	UK	(171) 555-1212	NULL
3	Hauptstr. 29	Bern	NULL	3012	Switzerland	0452-076545	NULL
4	Av. dos Lusíadas, 23	Sao Paulo	SP	05432-043	Brazil	(11) 555-7647	NULL

İstenilen formata uygun olmayan telefon numarası olanlar:

```
SELECT CompanyName, Phone FROM Customers
WHERE LEN(Phone) < 10 OR Phone NOT LIKE '[0-9]%' ;
```

	CompanyName	Phone
1	Ana Trujillo Emparedados y helados	(5) 555-4729
2	Antonio Moreno Taquería	(5) 555-3932
3	Around the Horn	(171) 555-7788
4	Bólido Comidas preparadas	(91) 555 22 82
5	Bottom-Dollar Markets	(604) 555-4729

! 10 haneden küçük numaral için özel bilgi alınması gerekmektedir.

Telefon numarasından özel karakterleri vb. çıkarıp, en sade formata dönüştürmek.

```
UPDATE Customers
SET Phone = REPLACE(REPLACE(REPLACE(REPLACE(
REPLACE(Phone, '.', ''), '- ', ''), '(', ''), ')', ''), ' ', '');
```

Telefon numarasının başındaki 0 rakamını kaldırmak. (11 haneli numaralarda.)

```
UPDATE Customers
SET Phone = RIGHT(Phone, LEN(Phone) - 1)
WHERE LEFT(Phone, 1) = '0' AND LEN(Phone) = 11;
```

Sonuç:

Phone
55554729
55553932
062108460
88601531
915552282
91244540
11355555
55553392
115557647

Veri dönüştürmeyi de kaynak sistemler farklı yapıda veri gönderirken, hedef veri ambarının standardına uygun biçimde veriyi dönüştürmek ve iş kurallarını uygulamak için kullanırız. Customers tablosundaki Country alanını daha standart hale getirelim (örn: "USA" yerine "United States"). Ayrıca, telefon numaralarını tek bir formatta (örn: (XXX) XXX-XXXX) standardize edelim. Yeni bir CleanedCustomers tablosu oluşturup bu dönüştürülmüş verileri buraya aktaralım.

68  
69  
70

```

SELECT * FROM CleanedCustomers;
SELECT * FROM Customers;

```

00 % No issues found Lnc: 69 Ch: 1

	CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	PostalCode	CountryStandardized	PhoneStandardized	FaxStandardized
1	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	NULL	12209	Germany	0300074321	0300076545
2	ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de la Constitución 2222	México D.F.	NULL	05021	Mexico	55554729	55553745
3	ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Mataderos 2312	México D.F.	NULL	05023	Mexico	55553932	NULL
4	AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	NULL	WA1 1DP	United Kingdom	1715557788	1715556750
5	BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	Berguvsvägen 8	Luleå	NULL	S-958 22	Sweden	0921123465	0921123467
6	BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forsterstr. 57	Mannheim	NULL	68306	Germany	062108460	062108924
7	BLOMP	Blondiesdel père et fils	Frédérique Citeaux	Marketing Manager	24, place Kléber	Strasbourg	NULL	67000	France	88601531	88601532
8	BOLID	Bólido Comidas preparadas	Martin Sommer	Owner	C/ Araquil, 67	Madrid	NULL	28023	Spain	915552282	915559199

	CustomerID	CompanyName	ContactName	ContactTitle	Address	City	Region	PostalCode	Country	Phone	Fax
1	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	Obere Str. 57	Berlin	NULL	12209	Germany	030-0074321	030-0076545
2	ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	Avda. de la Constitución 2222	México D.F.	NULL	05021	Mexico	(5) 555-4729	(5) 555-3745
3	ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	Mataderos 2312	México D.F.	NULL	05023	Mexico	(5) 555-3932	NULL
4	AROUT	Around the Horn	Thomas Hardy	Sales Representative	120 Hanover Sq.	London	NULL	WA1 1DP	UK	(171) 555-7788	(171) 555-6750
5	BERGS	Berglunds snabbköp	Christina Berglun	Order Administrator	Berguvsvägen 8	Luleå	NULL	S-958 22	Sweden	0921-12 34 65	0921-12 34 67
6	BLAUS	Blauer See Delikatessen	Hanna Moos	Sales Representative	Forsterstr. 57	Mannheim	NULL	68306	Germany	0621-08460	0621-08924
7	BLOMP	Blondiesdel père et fils	Frédérique Cit...	Marketing Manager	24, place Kléber	Strasbourg	NULL	67000	France	88.60.15.31	88.60.15.32

CountryStandardized ifadesi kullanarak bazı ülke isimlerini daha standart bir forma getirdik. Gerçek bir ETL sürecinde bu, bir lookup tablosu veya daha karmaşık bir eşleme mantığı ile yapılabilir.

PhoneStandardized ve FaxStandardized: Bu örnekte, telefon ve faks numaralarındaki parantez, tire, boşluk gibi karakterleri REPLACE fonksiyonu ile kaldırdık. Gerçek dünyada, telefon numarası standartlaştırması çok daha karmaşık olabilir ve Regex (Regular Expression) desteği gerektirebilir. MSSQL Server'da doğrudan Regex desteği bulunmadığından CLR (Common Language Runtime) fonksiyonları veya harici uygulamalar kullanılabilir.

Veri yükleme aşamasını veri dönüştürme ile aynı anda sağladık:

```

-- 2. Veri Dönüştürme ve Yükleme (ETL'nin Transform ve Load kısmı bir arada)
INSERT INTO CleanedCustomers (
    CustomerID,
    CompanyName,
    ContactName,
    ContactTitle,
    Address,
    City,
    Region,
    PostalCode,
    CountryStandardized,
    PhoneStandardized,
    FaxStandardized
)
SELECT
    c.CustomerID,
    c.CompanyName,
    c.ContactName,
    c.ContactTitle,
    c.Address,
    c.City,
    c.Region,
    TRY_CAST(c.PostalCode) AS PostalCode, -- Temizlenmiş PostalCode
    CASE
        WHEN c.Country = 'USA' THEN 'United States'
        WHEN c.Country = 'UK' THEN 'United Kingdom'
        -- Diğer ülke standartlaştırmaları buraya eklenebilir
        ELSE c.Country
    END AS CountryStandardized,
    -- Telefon numarasını standardize etme (Rakamları alıp belirli bir formata sokma)
    -- Daha karmaşık telefon numarası standartlaştırması için regex veya CLR fonksiyonları ekleyebiliriz.
    CASE
        WHEN c.Phone IS NOT NULL AND c.Phone <> '' THEN
            REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(c.Phone, '(', ''), ')', ''), '-', ''), ' ', ''), '.', ''), '/', ''), '\', ''), '+')
            -- Bu kısım, daha gelişmiş bir regex veya CLR fonksiyonu ile daha sağlam yapılabilir.
            -- Basit bir örnek olarak, sadece ilk 10 rakamı alalım ve (XXX) XXX-XXXX formatına getirelim.
            -- SQL Server'da RegEx desteği olmadığından bu biraz manuel olacaktır.
        ELSE NULL
    END AS PhoneStandardized,
    CASE
        WHEN c.Fax IS NOT NULL AND c.Fax <> '' THEN
            REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(c.Fax, '(', ''), ')', ''), '-', ''), ' ', ''), '.', ''), '/', ''), '\', ''), '+')
        ELSE NULL
    END AS FaxStandardized
FROM
    Customers AS c;

```

Son olarak da veri kalitesi raporlarımız var. Bu raporlar, yapılan işlemlerin etkinliğini ve veri kalitesindeki iyileşmeyi göstermelidir. Aşağıda bazı örnek senaryoları verdik:

```
79
80 -- 2. Ülke standartlaştırma raporu
81 -- Kaç farklı ülke adı vardı ve şimdi kaç farklı standartlaştırılmış ülke adı var?
82 SELECT 'Before Standardization' AS ReportType, COUNT(DISTINCT Country) AS DistinctCountries
83 FROM Customers;
84
85 SELECT 'After Standardization' AS ReportType, COUNT(DISTINCT CountryStandardized) AS DistinctCountries
86 FROM CleanedCustomers;
87
88 -- Standartlaştırılan ülke dağılımı
89 SELECT CountryStandardized, COUNT(*) AS RecordCount
90 FROM CleanedCustomers
91 GROUP BY CountryStandardized
92 ORDER BY RecordCount DESC;
```

0% No issues found

Report Type	DistinctCountries
Before Standardization	21

Report Type	DistinctCountries
After Standardization	21

CountryStandardized	RecordCount
United States	13
France	11
Germany	11
Brazil	9
United Kingdom	7
Spain	5
Mexico	5
Venezuela	4
Italy	3
Canada	3
Argentina	3

```
94 -- 3. Telefon numarası format değişikliği raporu (örnek)
95 -- Temizlenmeden önce kaç telefon numarası geçersiz formattaydı?
96 SELECT
97     'Before Standardization' AS ReportType,
98     COUNT(*) AS InvalidPhoneFormatCount
99 FROM
100     Customers
101 WHERE
102     Phone IS NOT NULL AND Phone <> ''
103     AND (Phone LIKE '%[*0-9\-\(\)\ ext ]%' OR LEN(REPLACE(REPLACE(REPLACE(Phone, '(', ''), ')', ''), '-', '')) < 10);
104
105 -- Temizlendikten sonraki durum (daha standardize edilmiş olması beklenir)
106 SELECT
107     'After Standardization' AS ReportType,
108     COUNT(*) AS InvalidPhoneFormatCount
109 FROM
110     CleanedCustomers
111 WHERE
112     PhoneStandardized IS NOT NULL AND PhoneStandardized <> ''
113     AND LEN(PhoneStandardized) < 10; -- Eğer 10 rakamlı olmasını bekliyorsak
114
```

00% No issues found

Report Type	InvalidPhoneFormatCount
Before Standardization	82

Report Type	InvalidPhoneFormatCount
After Standardization	52

## 7. Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları

Bu projede amaçlarımız veritabanı sistemlerinin yüksek erişilebilirliğini sağlamak, performansı artırmak için iş yükünü sunucular arasında dağıtma ve felaket durumunda kesintisiz hizmet sağlamak.

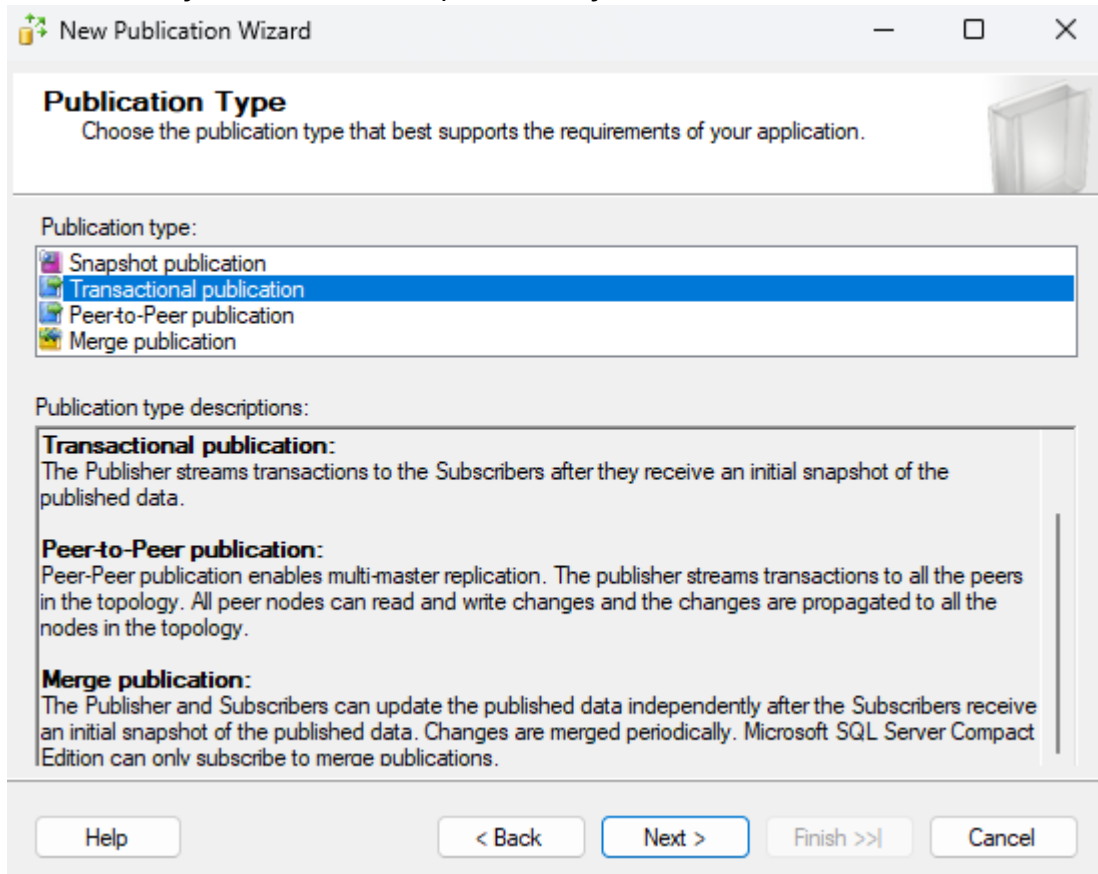
### Veritabanı Replikasyonu (SQL Server Replication)

Replication, bir veritabanındaki verileri birden fazla sunucuya kopyalamayı sağlar.

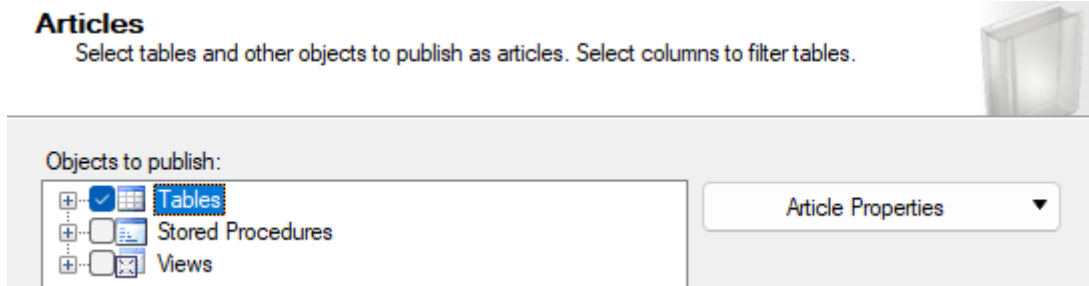
Publisher Ayarlama:

Replication → Local Publications → New Publication kısmından oluşturulur.

Veritabanı seçilir, transactional replication seçilir.




Çoğaltılacak tablolar belirlenir.





Sonucunda publication oluşturulur.

**Creating Publication**  
Click Stop to interrupt the operation.

 **Success** 3 Total 0 Error  
2 Success 1 Warning

Details:

Action	Status	Message
 Creating Publication 'Northwind_Pub'	Success	
 Adding article 14 of 14	Success	

Replication → Local Publications → “Oluşturulmuş Publication” → New Subscription kısmından kopyalanacak database için subscribe işlemi yapılır. Gerekli ayarlar seçilir.

### Subscribers

Choose one or more Subscribers and specify each subscription database.

Subscribers and subscription databases:

Subscriber ▲	Subscription Database
<input checked="" type="checkbox"/> YILDIZ	subDB


**Initialize Subscriptions**  
Specify whether to initialize each subscription with a snapshot of the publication data and schema.

Subscription properties:


Subscriber ▲	Memory Optimized	Initialize	Initialize When
YILDIZ	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Immediately

Sonucunda subscription oluşturulur.

**Creating Subscription(s)...**  
Click Stop to interrupt the operation.

 **Success** 2  
1

Details:

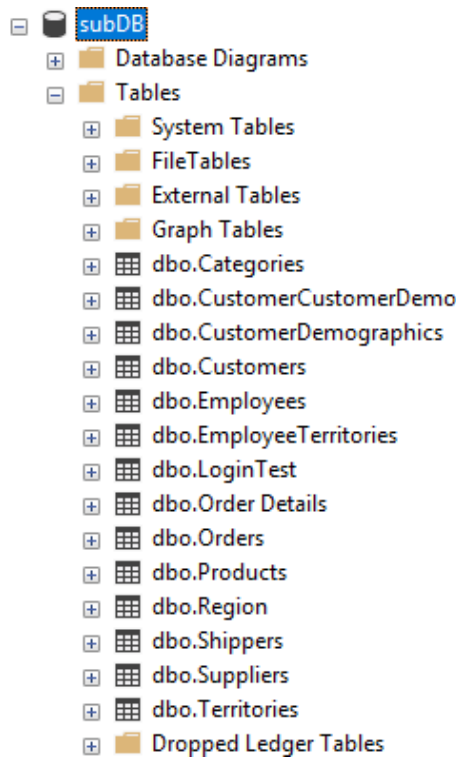
Action	Status
 Creating subscription for 'YILDIZ'	Success



Snapshot Agent çalıştırılarak kopyalama işlemi tamamlanır.



Aşağıda görüldüğü üzere “subDB” veritabanında “northwind” veritabanının tüm tabloları yer almaktadır.



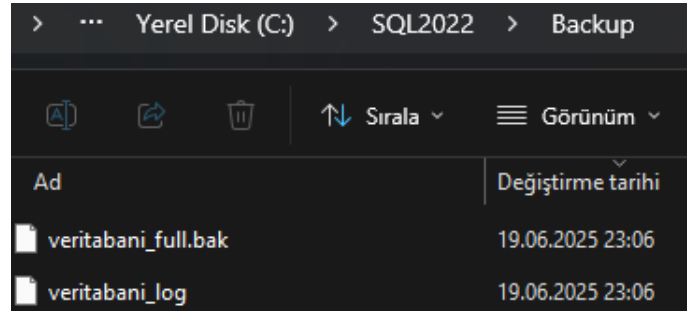
Sorgularla kontrol etmek istersek de aynı sonuçların döndüğünü görmekteyiz. Replication başarılı olmuştur.

replication.sql - YI...B (VILDIZ\cpulo (66))*					use northwind				
USE subDB; SELECT TOP 5 * FROM Customers;					use northwind SELECT TOP 5 * FROM Customers;				
Results					Results				
CustomerID	CompanyName	ContactName	ContactTitle		CustomerID	CompanyName	ContactName	ContactTitle	
1	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative	1	ALFKI	Alfreds Futterkiste	Maria Anders	Sales Representative
2	ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner	2	ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo	Owner
3	ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner	3	ANTON	Antonio Moreno Taquería	Antonio Moreno	Owner
4	AROUT	Around the Horn	Thomas Hardy	Sales Representative	4	AROUT	Around the Horn	Thomas Hardy	Sales Representative
5	BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator	5	BERGS	Berglunds snabbköp	Christina Berglund	Order Administrator

Bir ana veritabanı (principal), bir yedek veritabanı (mirror) içerir. Bu yapı ile veritabanı, bir sunucuda çalışamaz hale geldiğinde otomatik olarak diğerine geçebilir.

```
-- Veritabanını FULL recovery mode'a almak
ALTER DATABASE northwind SET RECOVERY FULL;

--FULL ve TRANSACTION BACKUP al (PRINCIPAL sunucuda)
BACKUP DATABASE northwind TO DISK = 'C:\SQL2022\Backup\veritabani_full.bak';
BACKUP LOG northwind TO DISK = 'C:\SQL2022\Backup\veritabani_log.trn';
```



Yerel Disk (C:) > SQL2022 > Backup	
Ad	Değiştirme tarihi
veritabani_full.bak	19.06.2025 23:06
veritabani_log	19.06.2025 23:06

```
--Mirror sunucuya bu yedekleri yükle (NORECOVERY ile!)
RESTORE DATABASE northwind
FROM DISK = 'C:\SQL2022\Backup\veritabani_full.bak'
WITH NORECOVERY;

RESTORE LOG northwind
FROM DISK = 'C:\SQL2022\Backup\veritabani_log.trn'
WITH NORECOVERY;
```

```
-- **Endpoint Oluştur (Her iki sunucuda da)**

-- Principal Sunucu
CREATE ENDPOINT MirroringEndpoint
STATE = STARTED
AS TCP (LISTENER_PORT = 5022)
FOR DATABASE_MIRRORING (
    ROLE = PARTNER
);

-- Mirror Sunucu
CREATE ENDPOINT MirroringEndpoint
STATE = STARTED
AS TCP (LISTENER_PORT = 5022)
FOR DATABASE_MIRRORING (
    ROLE = PARTNER
);

--Sunuculara login yetkisi ver
-- Principal'da:
CREATE LOGIN [MIRROR\sqlserviceaccount] FROM WINDOWS;
GRANT CONNECT ON ENDPOINT::MirroringEndpoint TO [MIRROR\sqlserviceaccount];

-- Mirror'da:
CREATE LOGIN [PRINCIPAL\sqlserviceaccount] FROM WINDOWS;
GRANT CONNECT ON ENDPOINT::MirroringEndpoint TO [PRINCIPAL\sqlserviceaccount];
```

```
-- Mirroring Başlat
ALTER DATABASE northwind
SET PARTNER = 'TCP://mirrorserver:5022';

ALTER DATABASE northwind
SET PARTNER = 'TCP://principalserver:5022';

--Durumu Kontrol Et
SELECT
    database_id,
    mirroring_state_desc,
    mirroring_role_desc,
    mirroring_partner_instance
FROM sys.database_mirroring
WHERE database_id = DB_ID('northwind');
```

Bu işlemler sonucunda ana veritabanında (Principal) yapılan her işlem, anında yedek veritabanına (Mirror) kopyalanır. Veritabanına Yüksek Erişilebilirlik (High Availability) sağlanır. Principal sunucu çökerse Mirror otomatik devreye girer. Replikasyondan farklıdır.