**ANKARA ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

****

**BLM 4522 FİNAL PROJE RAPORU**

**Furkan Uzunmehmetoğlu**

**21290176**

**30.05.2025**

**Git reposu:** [**https://github.com/Furkumo**](https://github.com/Furkumo/FurkiProject)

**Proje 1:**

[**https://drive.google.com/file/d/1QL1K6KOCWbwpC7G3NgKgywg8l7-HALq5/view?usp=drive\_link**](https://drive.google.com/file/d/1QL1K6KOCWbwpC7G3NgKgywg8l7-HALq5/view?usp=drive_link)

**Proje 2:**

[**https://drive.google.com/file/d/1j0nMWnNSpoTGr3CwCSx-09u\_gidZfPfX/view?usp=drive\_link**](https://drive.google.com/file/d/1j0nMWnNSpoTGr3CwCSx-09u_gidZfPfX/view?usp=drive_link)

**Proje 3:**

[**https://drive.google.com/file/d/1fuxJEJzloziLqEzPssJPjgzK5r3AV8H6/view?usp=drive\_link**](https://drive.google.com/file/d/1fuxJEJzloziLqEzPssJPjgzK5r3AV8H6/view?usp=drive_link)

**Proje 4:**[**https://drive.google.com/file/d/1Yrl8Ai5non8fPj02DLeAQx4yMOP8qwoX/view?usp=drive\_link**](https://drive.google.com/file/d/1Yrl8Ai5non8fPj02DLeAQx4yMOP8qwoX/view?usp=drive_link)

**Proje 5:** [**https://drive.google.com/file/d/1YkG-YGhFHNy0PFAPmyiXpVJbyskSIAaQ/view?usp=drive\_link**](https://drive.google.com/file/d/1YkG-YGhFHNy0PFAPmyiXpVJbyskSIAaQ/view?usp=drive_link)

**Proje 6:** [**https://drive.google.com/file/d/1PB3RogoHfmg2akbUnAcIdNHCT7KV6a4v/view?usp=drive\_link**](https://drive.google.com/file/d/1PB3RogoHfmg2akbUnAcIdNHCT7KV6a4v/view?usp=drive_link)

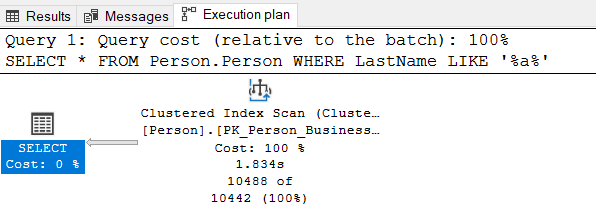
**Proje 7:**

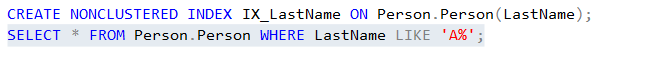
[**https://drive.google.com/file/d/1W\_nrSWg9wmzmWT1wl5fLXAxEcveMvIh9/view?usp=drive\_link**](https://drive.google.com/file/d/1W_nrSWg9wmzmWT1wl5fLXAxEcveMvIh9/view?usp=drive_link)

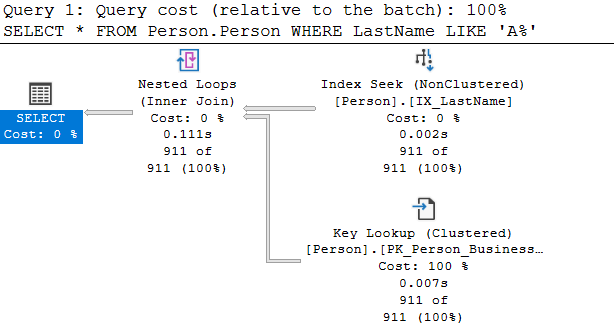
**PROJE 1: Veritabanı Performans Optimizasyonu ve İzleme**

**A)İndeks Yönetimi**

Öncelikle Veritabanı izlemesi yapıyoruz ve performansını analiz etmek için sorgumuzu yazıyoruz



Performansları karşılaştırmak için sorgumuzu yazıp tekrar çalıştırıyoruz

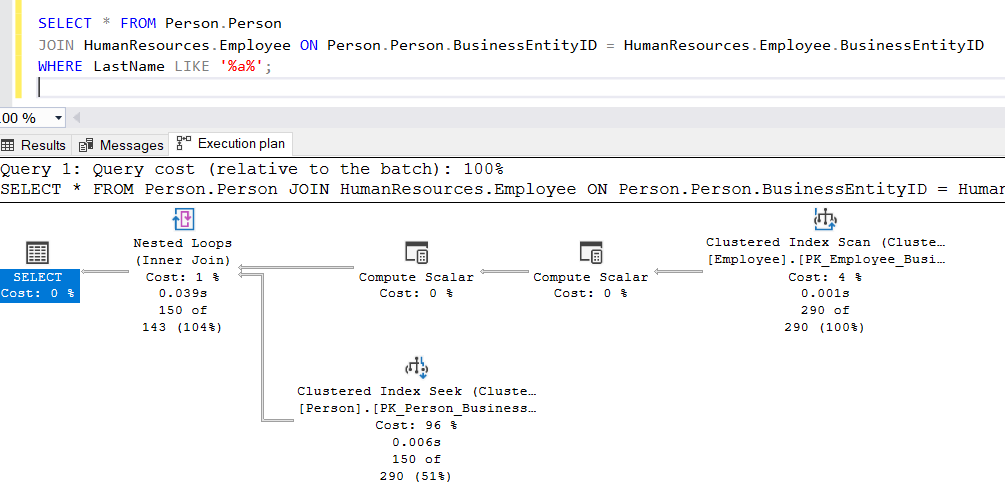
****

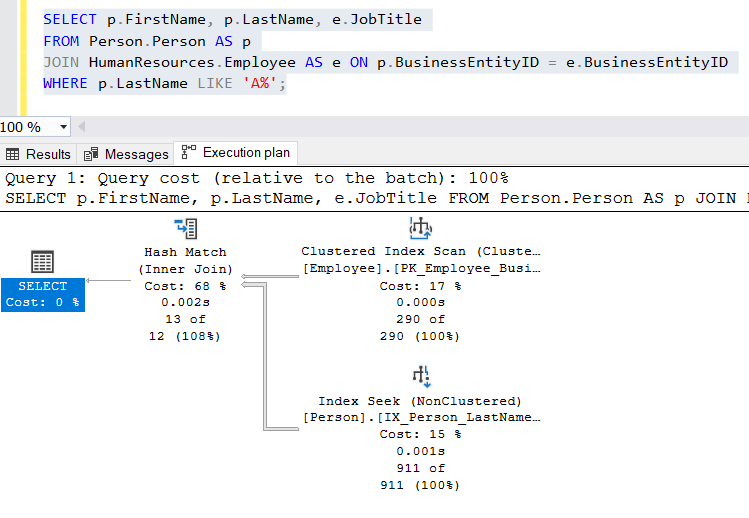
Person.Person tablosunda LastName sütununa yönelik oluşturulan IX\_LastName isimli NonClustered indeks, sorgu süresinde ciddi iyileşme sağlamıştır.

Başta yapılan sorgu Clustered Index Scan ile tüm tabloyu tararken, indeksin ardından Index Seek kullanımı ile sadece gerekli kayıtlar hedeflenmiştir.

Key Lookup işlemi, indeks dışında kalan alanların tabloda aranmasından kaynaklı olup performansa olumsuz etkisi minimumdur.

Bu örnek, doğru indeks kullanımının sorgu performansını ne derece etkilediğini açıkça ortaya koymaktadır.

**B)** **Sorgu İyileştirme**

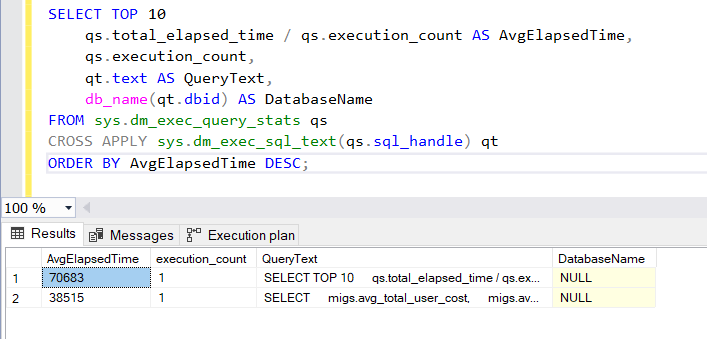


Bu adımda, AdventureWorks2019 veritabanında Person.Person ve HumanResources.Employee tabloları arasında yapılan birleştirme işlemi optimize edilmiştir.

Başlangıçta kullanılan SELECT \* ifadesi tüm sütunları çağırarak bellek yükünü artırmış, ayrıca LIKE '%a%' koşulu nedeniyle indeks kullanımı devre dışı kalmıştır.

Optimize edilen sorguda yalnızca ihtiyaç duyulan sütunlar çağrılmış (FirstName, LastName, JobTitle) ve LIKE 'A%' filtresi ile Index Seek kullanımı sağlanmıştır.

Execution Plan karşılaştırmasında, sorgu maliyetinin düştüğü ve veri erişiminin daha verimli hale geldiği gözlemlenmiştir.

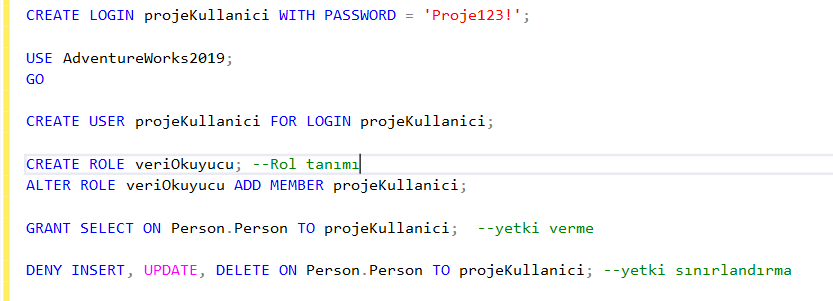
**C) Veritabanı İzleme**

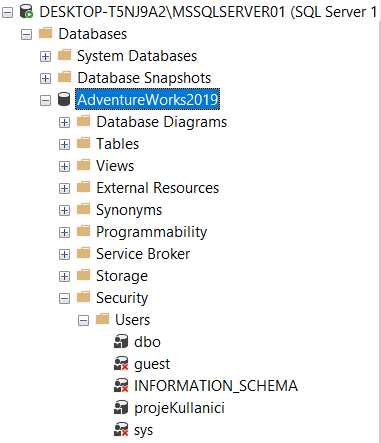
Tablomuz şunları gösteriyor:

AvgElapsedTime: Sorgunun ortalama çalışma süresi (milisaniye cinsinden)

execution\_count: Sorgunun kaç kere çalıştırıldığı

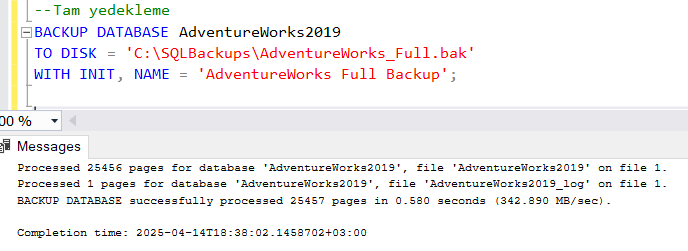
QueryText: Sorgunun metni

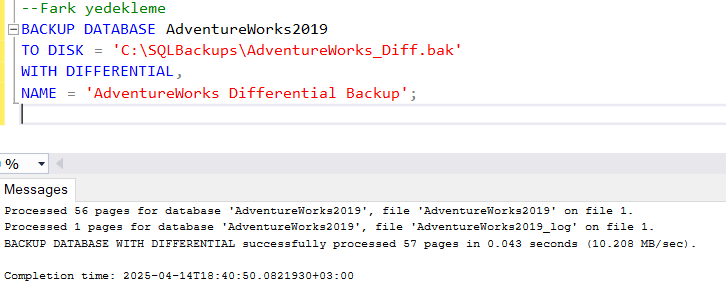
**D) Veri Yöneticisi Rolleri**

Kullanıcı girişimizi altta gösterebiliriz

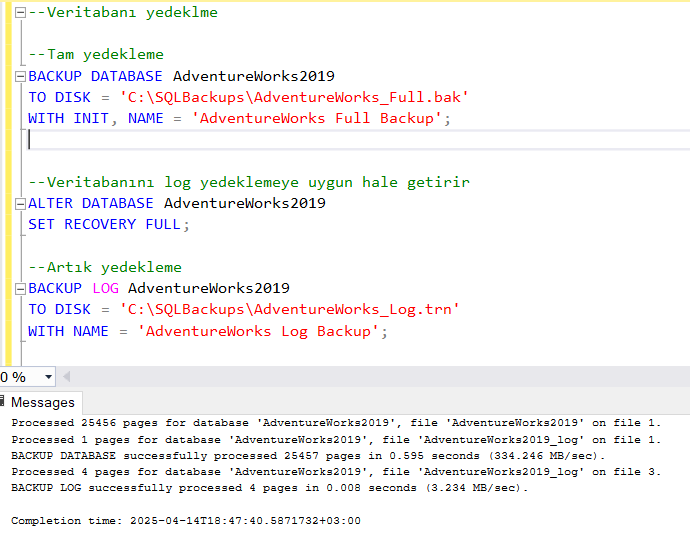
**Proje 2 : Veritabanı Yedekleme ve Felaketten Kurtarma Planı**

**A)Veritabanı Yedekleme işlemleri**

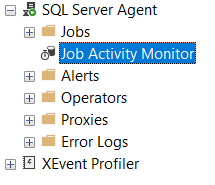




Artık yedekleme yapmadan önce tam yedekleme yapılması lazım ve yedeklemeye uygun hale getirmek için Simple Recovery Modedan Full moda getirme işlemi yapmamız gerekiyor.



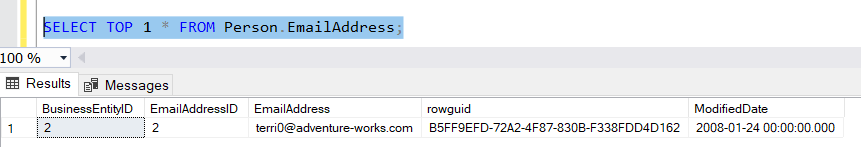
Zamanlayıcı yedekleme



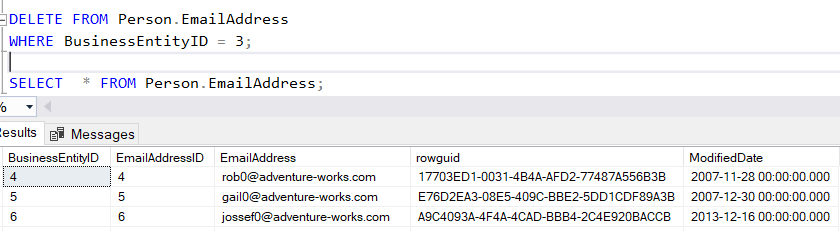
Yanda belirtildiği gibi SQL Server Agent üzerinden oluşturulan Tam\_Yedek adlı job, her hafta Pazar günü saat 00:00'da çalışacak şekilde zamanlanmıştır. Job içerisindeki Step'te AdventureWorks2019 veritabanının tam yedeği alınmaktadır.

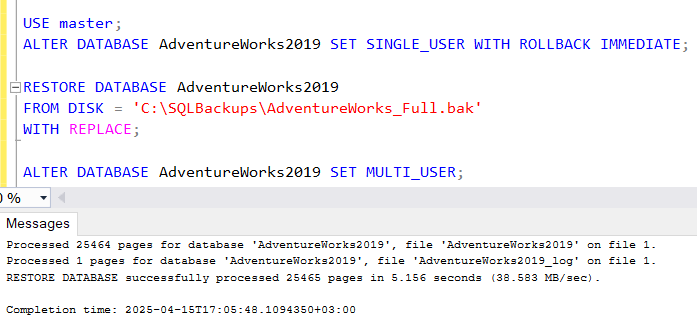
**B)Felaketten Kurtarma Senaryoları**

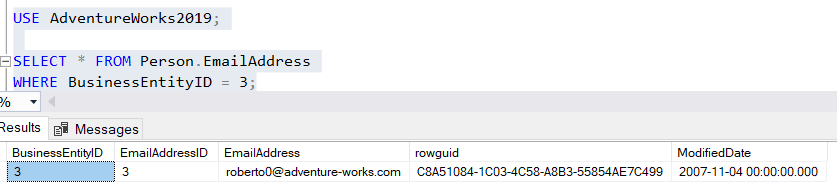
Sileceğimiz satırın ön izlemesini yapıyoruz



Silindiğini buradan görebiliyoruz



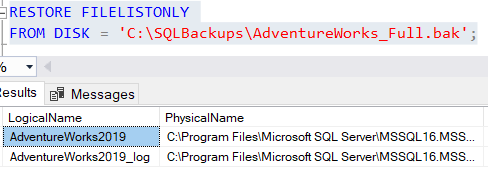
Geri yükleme başarılı

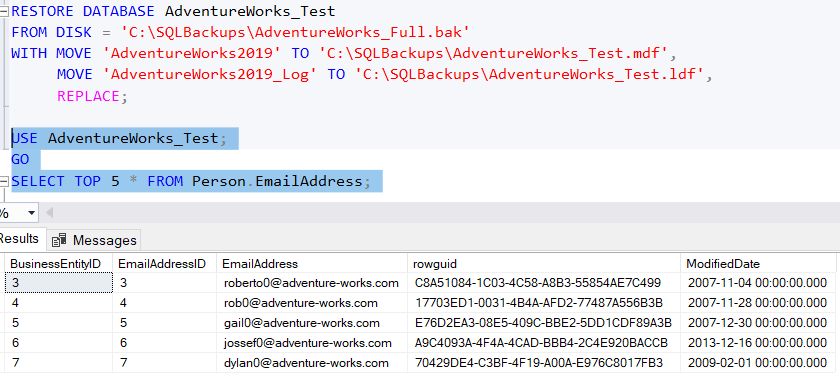
Geri yüklediğimiz satırı ön izliyoruz

Felaket senaryosu kapsamında Person.EmailAddress tablosundaki BusinessEntityID = 3 satırı manuel olarak silinmiştir.

Daha önce alınan tam yedek (.bak) dosyasından yapılan geri yükleme ile verinin kurtarıldığını göstermiş olduk.

**C)Test Yedekleme Senaryoları**

İşlemi yapacağımız satırın içeriğini öğreniyoruz

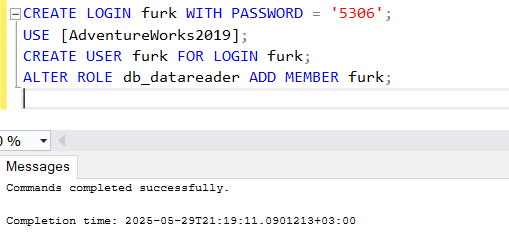


Yedekleme senaryosunun doğruluğunu test etmek amacıyla daha önce alınan tam yedek .bak dosyası farklı bir adla sisteme geri yükleme yaptık.  
Yükleme sonrası veritabanı içerikleri kontrol edilmiş, tabloların ve verilerin eksiksiz geldiği doğrulamış olduk.

**Proje Sonrası Değerlendirme**

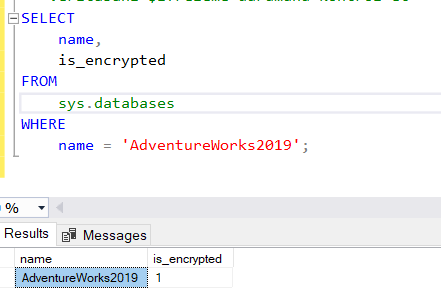
Proje sonunda görüldü ki, veri kayıplarına karşı alınabilecek en etkili önlem düzenli ve sistematik yedekleme stratejileridir. Tam yedekleme ve geri yükleme işlemleriyle verinin başarıyla kurtarılabildiği gösterilmiş, bu da felaket senaryolarında sistemin ne kadar hazırlıklı olduğunun bir göstergesi olmuştur. Ayrıca test senaryolarıyla planların işlerliği teyit edilmiştir.

**Proje 3: Veritabanı Güvenliği ve Erişim Kontrolü**

**a)Erişim Yönetim**

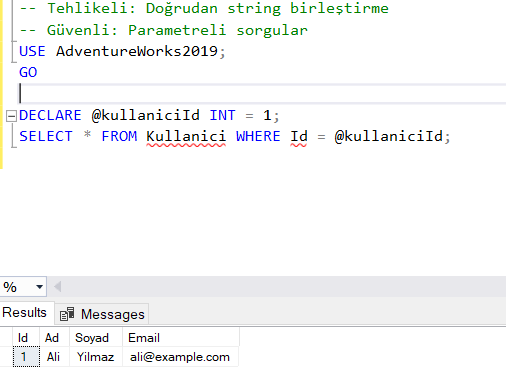
SQL Server’da kullanıcı erişim yönetimi sağlanarak, yalnızca yetkilendirilmiş kişilerin belirli verilere erişmesine izin verilmiştir. SQL Server Authentication ve Windows Authentication yöntemleri kullanılarak kullanıcıların kimlik doğrulaması yapılmıştır. Roller kullanılarak erişim seviyeleri yapılandırılmıştır.

**b) Veri Şifreleme**



Veri güvenliğini sağlamak amacıyla Transparent Data Encryption (TDE) yöntemi kullanılmıştır. Böylece veritabanı dosyaları diske yazılırken şifrelenmiş ve yetkisiz erişimlerin önüne geçilmiştir.

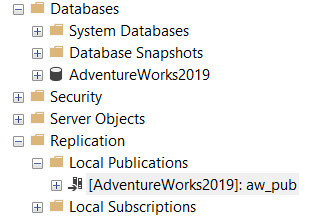
**c)** **SQL Injection Testi**



Veritabanı, SQL Injection saldırılarına karşı test edilmiştir. Parametreli sorgular ve giriş doğrulama mekanizmaları kullanılarak bu tür saldırıların önüne geçilmiştir.

**Proje Sonrası Değerlendirme:**Bu proje sayesinde, veritabanı güvenliğinin sadece fiziksel erişimle sınırlı olmadığı; kullanıcı yetkilendirme, veri şifreleme ve dış saldırılara karşı testlerle bütüncül bir güvenlik anlayışının gerekli olduğu görülmüştür. Geliştirilen sistem, veriye sadece yetkili kişilerin erişebilmesini sağlamış ve potansiyel saldırı risklerini en aza indirmiştir.

**Proje 4: Veritabanı Yük Dengeleme ve Dağıtık Veritabanı Yapıları**

**a)Veritabanı Replikasyonu**

**b)Yük Dengeleme**

Yük dengeleme stratejisi olarak SQL Server Database Mirroring yöntemi tercih edilmiştir.

İki farklı SQL Server instance’ı kullanılarak:

SQLDEV → Principal

MSSQLSERVER01 → Mirror

olarak yapılandırılmıştır. Veritabanı, FULL Recovery Model’e geçirilmiş ve tam yedek (full backup) ile log yedeği mirror sunucuya WITH NORECOVERY ile aktarılmıştır. Ardından mirror bağlantısı kurulmuş ve senkronizasyon başarılı şekilde sağlanmıştır.

**c)** **Failover Senaryoları**

Projenin devamında **manuel failover** senaryosu test edilmiştir. SQL Server Management Studio arayüzü üzerinden mirroring panelinden **“Failover”** işlemi gerçekleştirilmiş, rollerde değişim başarıyla gözlemlenmiştir:

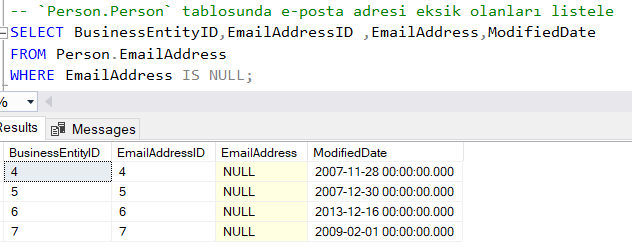
* Principal → Mirror
* Mirror → Principal

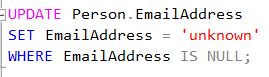
Failover sonrası sistem kesintisiz şekilde çalışmaya devam etmiştir. Veri bütünlüğü korunmuş ve herhangi bir kayıp yaşanmamıştır.

**Proje Sonrası Değerlendirme**

Yük dengeleme ve dağıtık yapıların büyük ölçekli sistemlerde ne kadar hayati olduğu bu projeyle anlaşılmıştır. Özellikle Database Mirroring kullanımıyla sistemde yedeklilik ve süreklilik sağlanmış; manuel failover testleriyle sistemin kesintisiz çalışabildiği gösterilmiştir. Bu uygulama, hem veri güvenliğini hem de sistemin erişilebilirliğini artırmıştır.

**Proje 5: Veri Temizleme ve ETL Süreçleri Tasarımı**

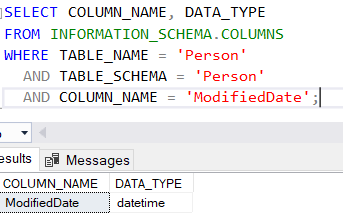
**A)** **Veri Temizleme**

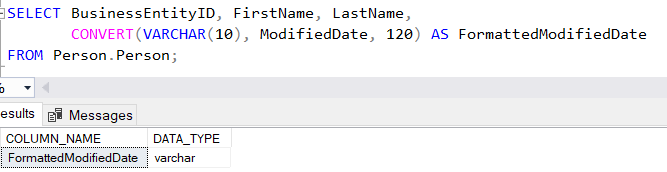




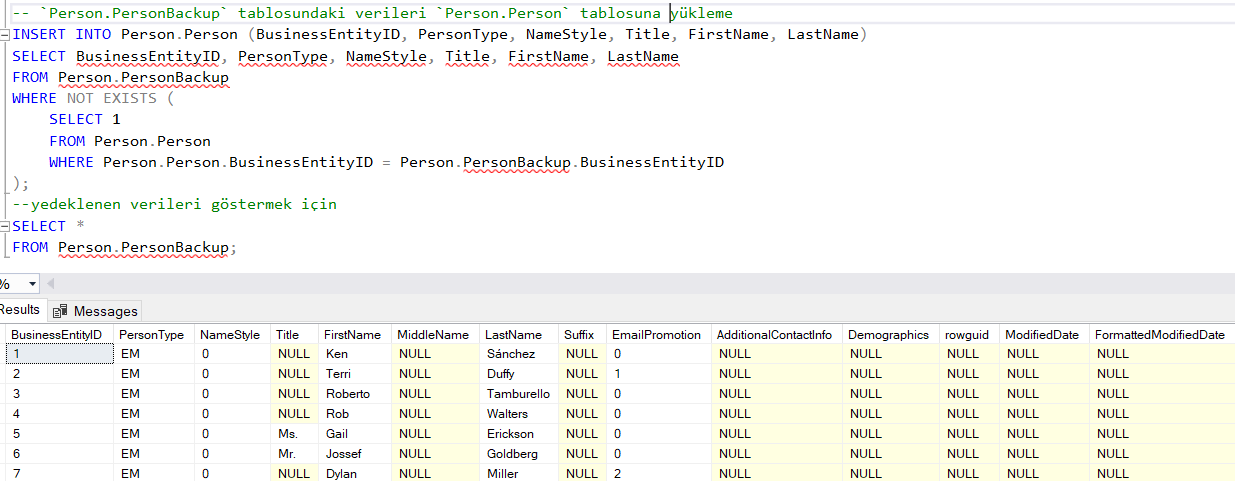
Veri temizleme süreci, eksik verilerin (NULL değerlerinin) düzenlenmesi için gerçekleştirilmiştir. Özellikle, Person.EmailAddress tablosundaki eksik e-posta adresleri tespit edilip, bu NULL değerler unknown ile güncellenmiştir.

**B) Veri Dönüştürme**

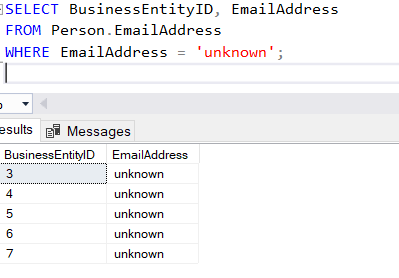
Dönüştürmeden önceki veri tipimiz



Veri dönüştürme süreci, tarih formatının doğru bir şekilde düzenlenmesi için yapılmıştır. ModifiedDate sütunundaki datetime türündeki veri, VARCHAR(10) türüne dönüştürülerek YYYY-MM-DD formatına getirilmiştir.

**C)Veri Yükleme**

**D)** **Veri Kalitesi Raporları**



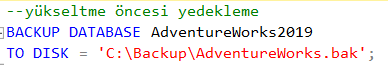
**Eksik E-posta Adresi Temizliği: E-posta adresi olmayan veriler** "unknown" ile güncellenmiştir.

**Proje Sonrası Değerlendirme**

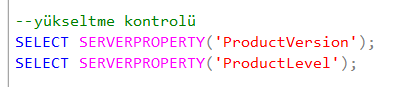
Veri kalitesinin yüksekliği, doğru kararlar alabilmek için hayati önemdedir. Bu proje kapsamında, eksik ve hatalı veriler temizlenmiş, veri dönüşümleri yapılmış ve sistemde kullanılmaya hazır hale getirilmiştir. ETL süreci boyunca uygulanan işlemler sayesinde verinin hem yapısı hem de okunabilirliği iyileştirilmiştir. Sonuç olarak, veri analizi için daha güvenilir ve anlamlı bir veri kümesi elde edilmiştir.

**Proje 6:** **Veritabanı Yükseltme ve Sürüm Yönetimi**

**A)Yükseltme Stratejisi**

Önce kullandığımız veri tabanını yedekliyoruz

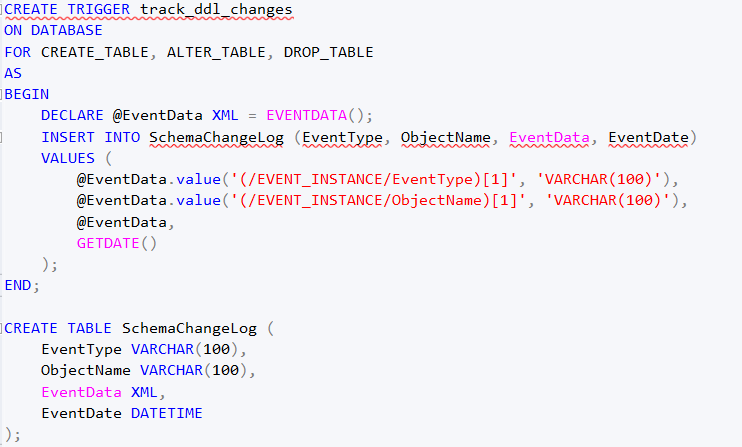
Yükseltme adımlarını gerçekleştiriyoruz.

Daha sonrasında control yapıyoruz

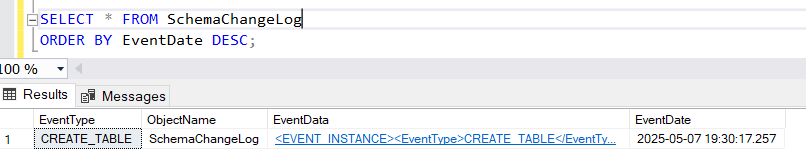
Var olan veritabanının yeni sürüme yükseltilmesi için önce tam yedekleme alınmıştır. Ardından yükseltme adımları kontrollü bir şekilde uygulanmıştır. Bu adımlar, sistemin bozulmadan yeni özelliklere kavuşmasını sağlamıştır.

**B)Sürüm Yönetimi**

DDL trigger'ı oluşturuyoruz. Bu trigger kullanarak her CREATE, ALTER veya DROP komutunu kaydediyoruz.

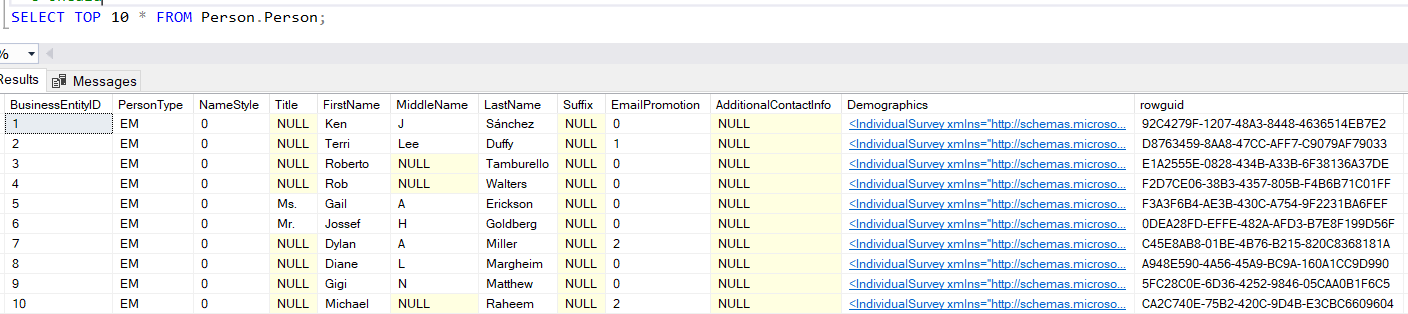


DDL trigger'ları kullanılarak veritabanındaki yapısal değişiklikler (CREATE, ALTER, DROP gibi komutlar) takip edilmiştir. Bu sayede şema değişiklikleri merkezi olarak izlenebilir hale gelmiştir.

Kontrol ediyoruz

**C) Test ve Geri Dönüş Planı**

**Test Planı**

Yükseltme sonrasında sistemin doğru çalıştığından emin olmak için test ediyoruz.

**Geri Dönüş Planı**

Yükseltme sırasında herhangi bir sorunla karşılaşırsak, yedeği geri yükleyerek eski sürüme dönebiliriz.

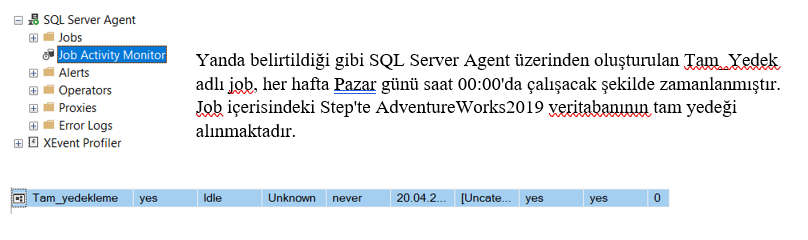


Yükseltme sonrası test senaryoları uygulanarak sistemin kararlılığı kontrol edilmiştir. Herhangi bir sorun durumunda geri dönüş için önceden alınan yedek kullanılarak sistem eski haline döndürülebilir yapıdadır.

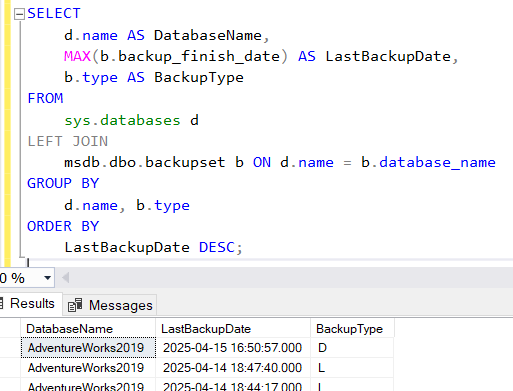
**Proje Sonrası Değerlendirme**

Yükseltme işlemi, yazılım geliştirme süreçlerinde sürekliliği sağlamak için kritik bir adımdır. Bu proje kapsamında gerçekleştirilen yükseltme ve sürüm yönetimi uygulamaları, veri bütünlüğünü koruyarak sistemin gelecekteki ihtiyaçlara daha esnek şekilde cevap verebilmesini sağlamıştır.

**Proje 7: Veritabanı Yedekleme ve Otomasyon Çalışması**

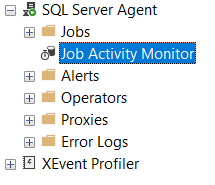
**A)SQL Server Agent Kullanarak**

**B)** **T-SQL Scripting**



T-SQL betikleriyle yedekleme geçmişi düzenli olarak kontrol edilmekte, hangi veritabanının ne zaman yedeklendiği raporlanmaktadır. Bu raporlar sayesinde sistem yöneticileri yedekleme durumunu kolaylıkla izleyebilmektedir.

**C) Otomatik Yedekleme Uyarıları**



Yanda belirtildiği gibi SQL Server Agent üzerinden oluşturulan Tam\_Yedek adlı job, her hafta Pazar günü saat 00:00'da çalışacak şekilde zamanlanmıştır. Job içerisindeki Step'te AdventureWorks2019 veritabanının tam yedeği alınmaktadır.



**Proje Sonrası Değerlendirme**

Veritabanı yedekleme ve otomasyon uygulamaları, kurumsal veri güvenliği açısından kritik öneme sahiptir. Bu proje ile düzenli, güvenli ve merkezi olarak takip edilebilir yedekleme sistemleri oluşturulmuş; veri kaybı riskleri büyük ölçüde azaltılmıştır.