**ANKARA ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

****

**BLM 4522 PROJE RAPORU**

**Furkan Uzunmehmetoğlu 21290176**

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

**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=sharing**](https://drive.google.com/file/d/1j0nMWnNSpoTGr3CwCSx-09u_gidZfPfX/view?usp=sharing)

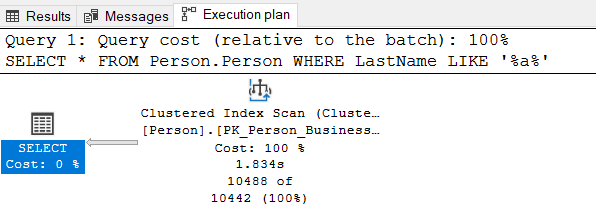
**Proje 3:** [**https://drive.google.com/file/d/1YkG-YGhFHNy0PFAPmyiXpVJbyskSIAaQ/view?usp=sharing**](https://drive.google.com/file/d/1YkG-YGhFHNy0PFAPmyiXpVJbyskSIAaQ/view?usp=sharing)

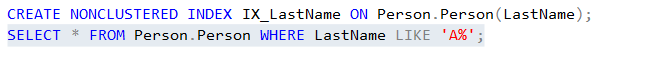
**22.04.2025**

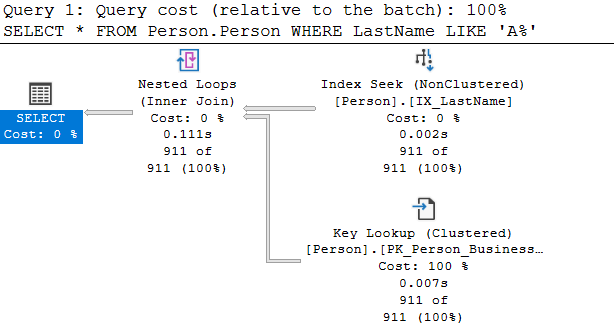
**PROJE 1: Veritabanı Performans Optimizasyonu ve İzleme**

**A)İndeks Yönetimi**

Öncelikle Veritabanı izlemesi yapıyoruz ve performansını analiz etmek için sorgumuzu yazıyoruz



Performansları karşılaştırmak için sorgumuzu yazıp tekrar çalıştırıyoruz

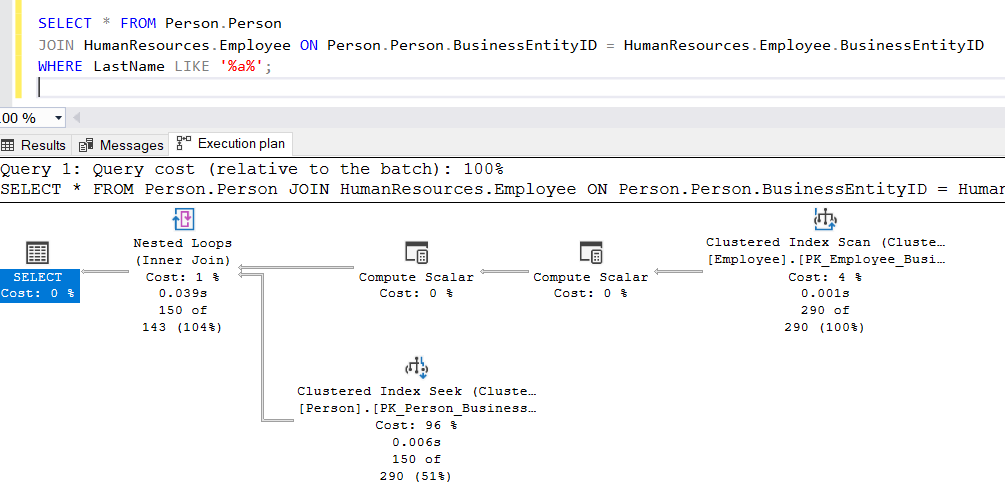
****

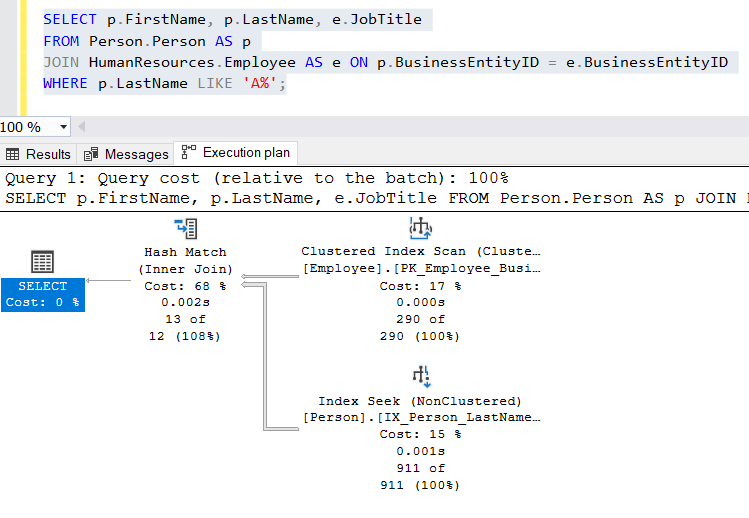
Person.Person tablosunda LastName sütununa yönelik oluşturulan IX\_LastName isimli NonClustered indeks, sorgu süresinde ciddi iyileşme sağlamıştır.

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**

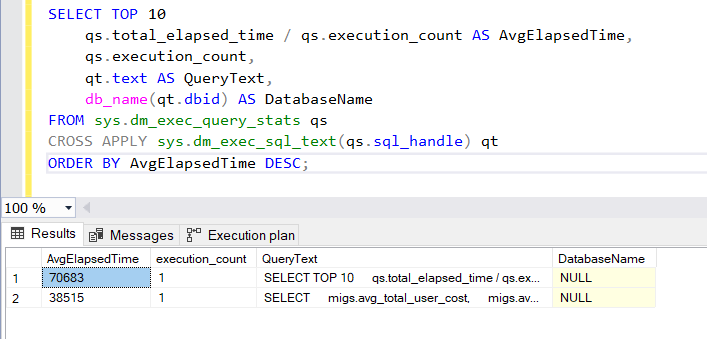


Bu adımda, AdventureWorks2019 veritabanında Person.Person ve HumanResources.Employee tabloları arasında yapılan birleştirme işlemi optimize edilmiştir.

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.

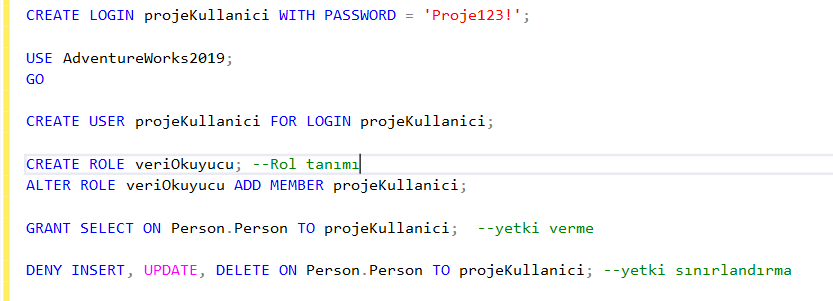
**C) Veritabanı İzleme**

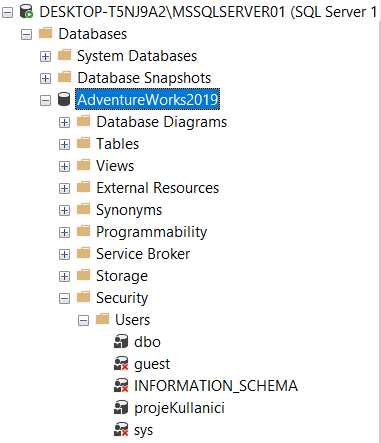
Tablomuz şunları gösteriyor:

AvgElapsedTime: Sorgunun ortalama çalışma süresi (milisaniye cinsinden)

execution\_count: Sorgunun kaç kere çalıştırıldığı

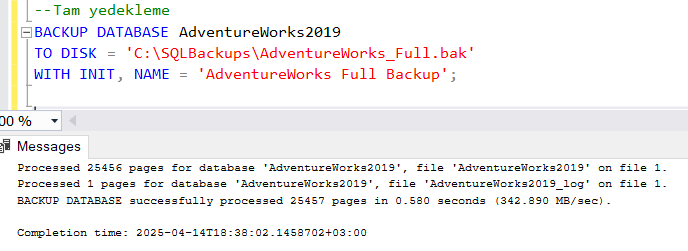
QueryText: Sorgunun metni

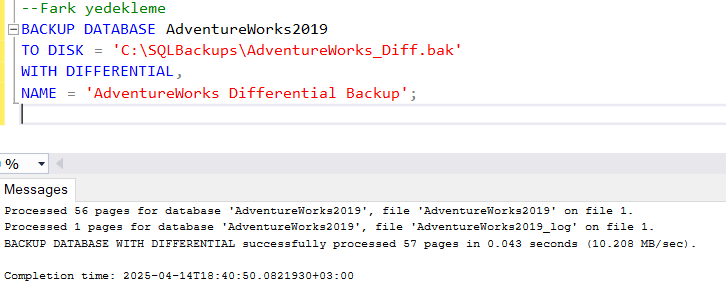
**D) Veri Yöneticisi Rolleri**

Kullanıcı girişimizi altta gösterebiliriz

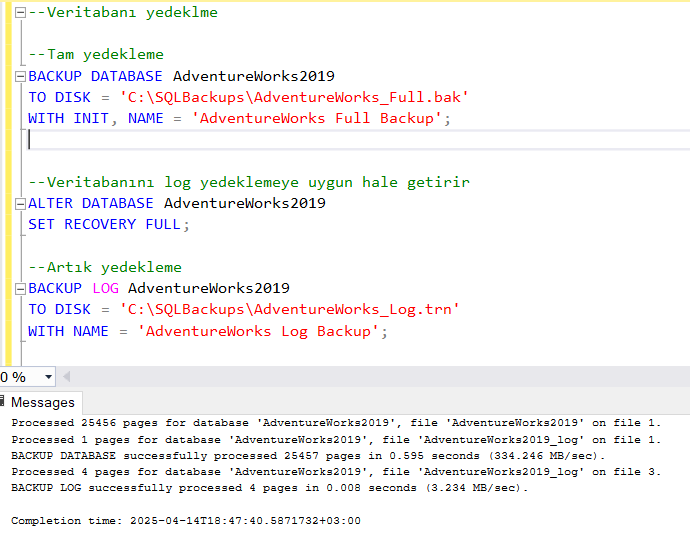
**Proje 2 : Veritabanı Yedekleme ve Felaketten Kurtarma Planı**

**A)Veritabanı Yedekleme işlemleri**

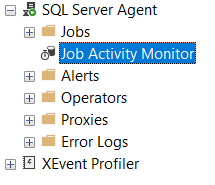




Artık yedekleme yapmadan önce tam yedekleme yapılması lazım ve yedeklemeye uygun hale getirmek için Simple Recovery Modedan Full moda getirme işlemi yapmamız gerekiyor.



Zamanlayıcı yedekleme

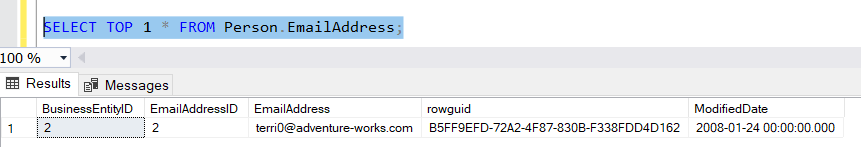


Yanda belirtildiği gibi SQL Server Agent üzerinden oluşturulan Tam\_Yedek adlı job, her hafta Pazar günü saat 00:00'da çalışacak şekilde zamanlanmıştır. Job içerisindeki Step'te AdventureWorks2019 veritabanının tam yedeği alınmaktadır.

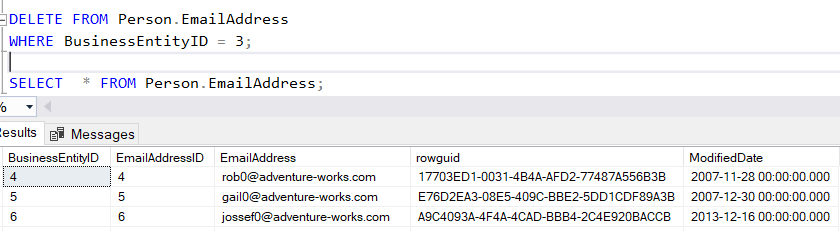


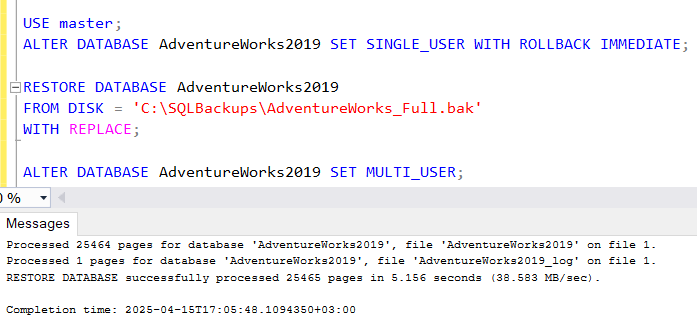
**B)Felaketten Kurtarma Senaryoları**

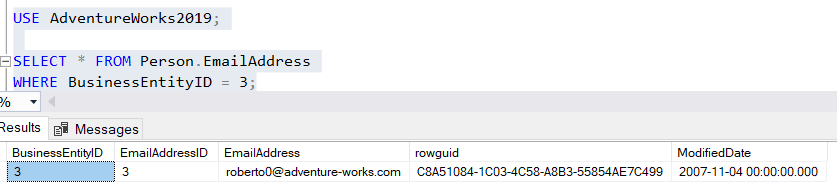
Sileceğimiz satırın ön izlemesini yapıyoruz



Silindiğini buradan görebiliyoruz



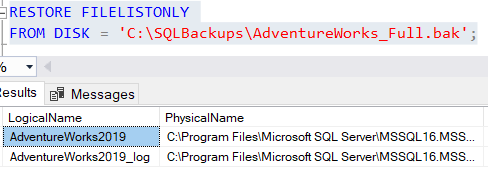
Geri yükleme başarılı

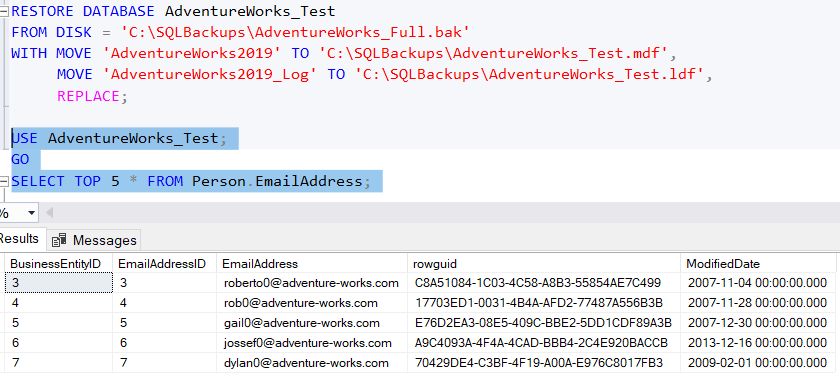
Geri yüklediğimiz satırı ön izliyoruz

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.

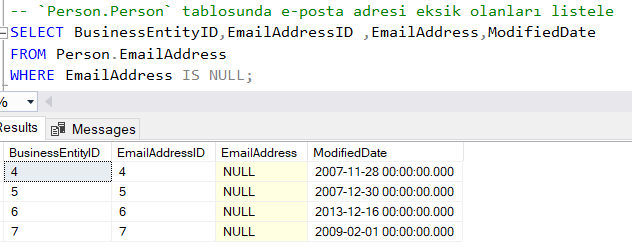
**C)Test Yedekleme Senaryoları**

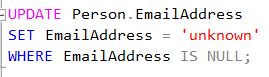
İşlemi yapacağımız satırın içeriğini öğreniyoruz



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 3: Veri Temizleme ve ETL Süreçleri Tasarımı**

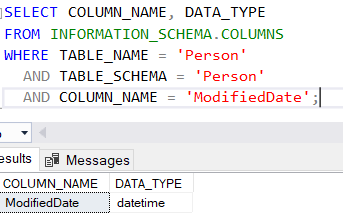
**A)** **Veri Temizleme**

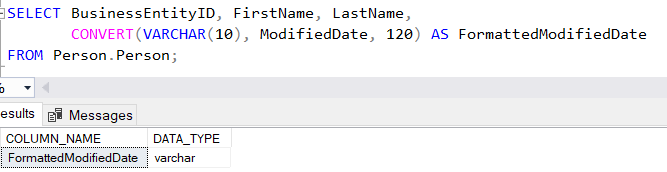




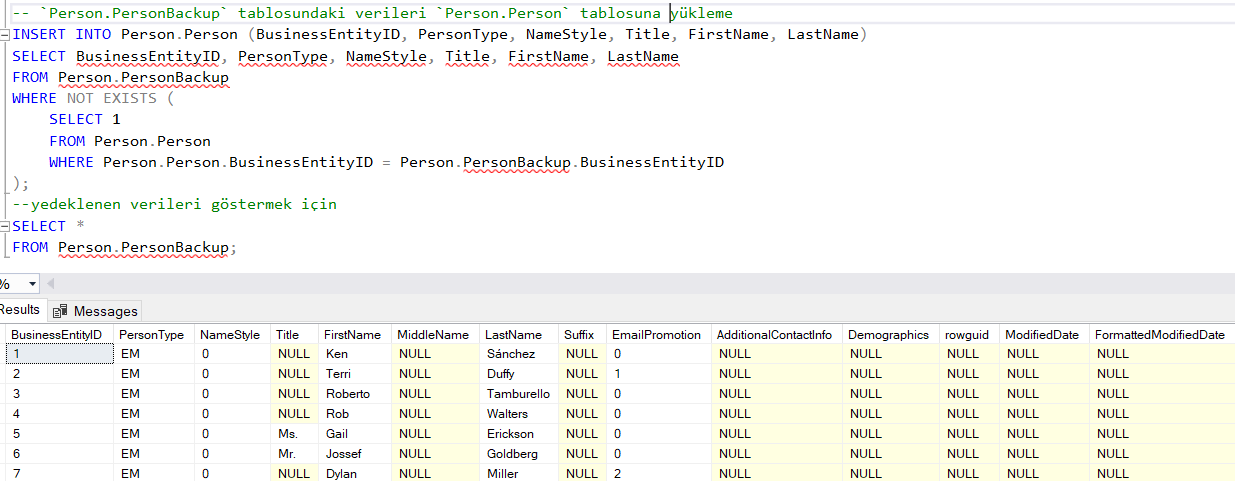
Veri temizleme süreci, eksik verilerin (NULL değerlerinin) düzenlenmesi için gerçekleştirilmiştir. Özellikle, Person.EmailAddress tablosundaki eksik e-posta adresleri tespit edilip, bu NULL değerler unknown ile güncellenmiştir.

**B) Veri Dönüştürme**

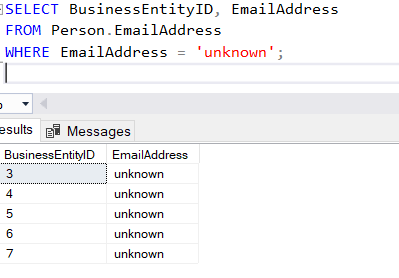
Dönüştürmeden önceki veri tipimiz



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.