# **Proje 1: Veritabanı Performans Optimizasyonu ve İzleme**

Hazırlayan: Mert Efe Kandemir, Kaan Kaya  
 Numara: 21290233, 21290436  
 Teslim: 25.04.2025   
 Github: https://github.com/kayakaan02/-BLM4522-A-Tabanl-Paralel-Da-t-m-Sistemleri-Proje-Videolari/blob/main/proje1.mp4

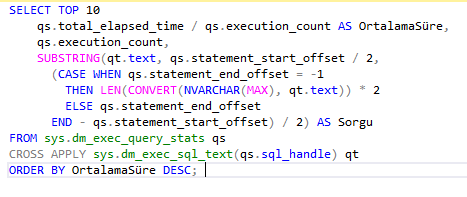
## **1. Giriş**

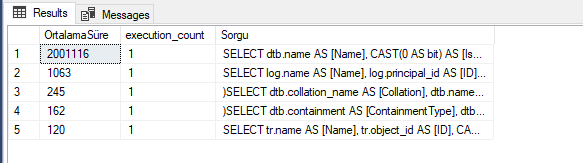
Bu proje, SQL Server ortamında büyük ölçekli veritabanlarında performans takibi ve iyileştirme stratejilerinin uygulanmasını kapsamaktadır. Yavaş çalışan sorguların optimize edilmesi, indeks yönetimi, sistem kaynaklarının izlenmesi ve erişim yönetimi gibi önemli performans faktörleri ele alınmıştır.

## **2. Veritabanı İzleme**

SQL Server Profiler ve Dynamic Management Views (DMV) gibi araçlar kullanılarak sistemde çalışan sorgular, kilitlenmeler ve yavaşlamalar tespit edilmiştir. Özellikle uzun süren sorguların CPU ve IO kullanımı detaylı olarak analiz edilmiştir. Sistem genelinde kaynak tüketimini etkileyen işlemler belirlenmiş ve işlem öncelikleri yeniden değerlendirilmiştir.

Aşağıdaki sorgu ile sistemdeki en çok kaynak tüketen sorgular listelenerek analiz edilmiştir:



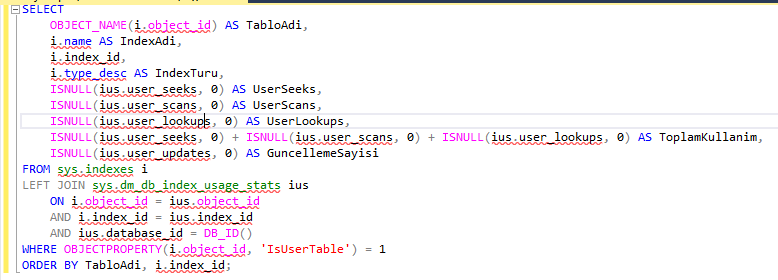


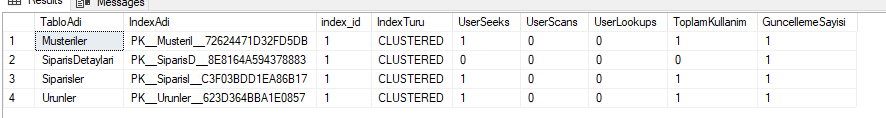
## **3. İndeks Yönetimi**

Veri erişim performansını artırmak için tablolar üzerinde yer alan indeksler analiz edilmiştir. Sık kullanılan sorgulara uygun biçimde yeni indeksler oluşturulmuş, kullanılmayan ve sisteme yük olan indeksler kaldırılmıştır. Clustered ve Non-Clustered indeks farkları dikkate alınarak doğru stratejiler belirlenmiştir.

### **Örnek Veritabanı Oluşturma** **İndeks Kullanım İstatistiklerini Analiz Etme**

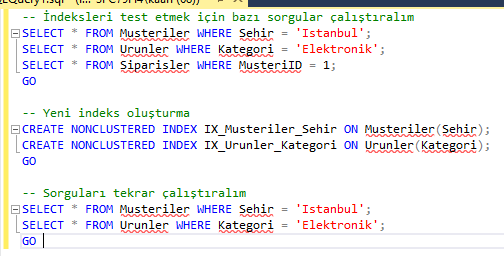
Aşağıdaki sorgu ile indeks kullanım istatistikleri listelenerek, az kullanılan indeksler tespit edilmiştir:

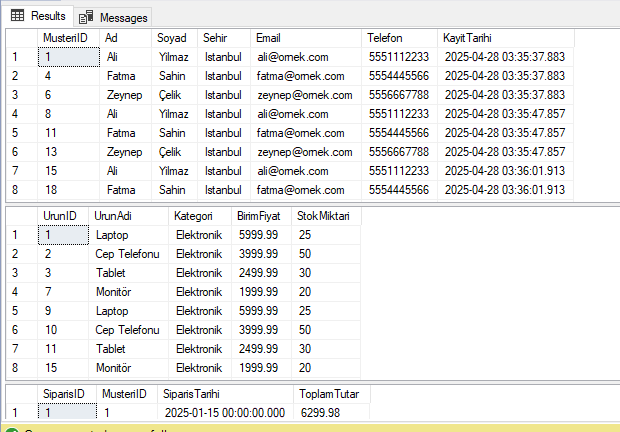




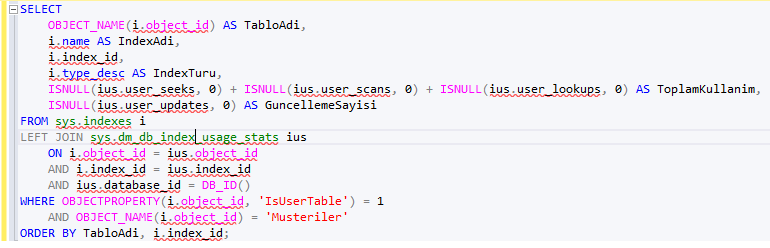
**İndeks Oluşturma ve Test Etme**

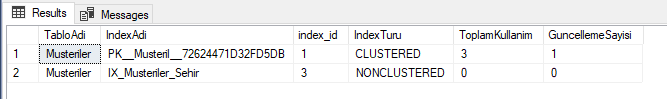
İndeksleri test etmek için bazı sorgular çalıştıralım ve ardından yeni indeksler oluşturalım:





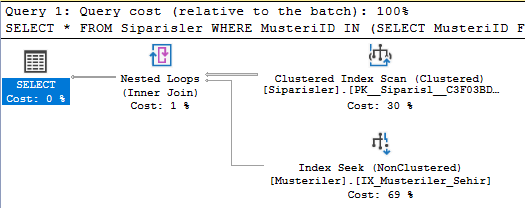
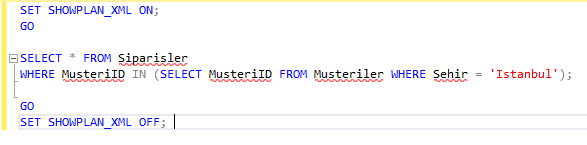
Şimdi Musteriler tablosu için indeks kullanım istatistiklerini kontrol edelim:



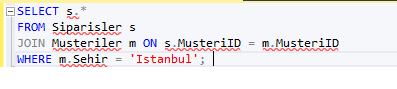


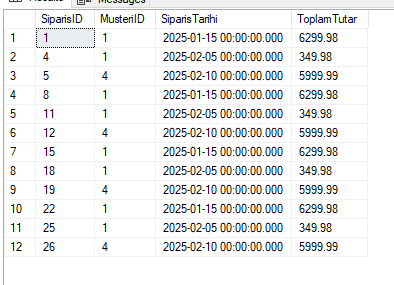
## **4. Sorgu İyileştirme**

Performans sorunu yaratan sorgular, Execution Plan yardımıyla analiz edilmiştir. Alt sorgular JOIN yapıları, filtreleme koşulları gözden geçirilerek daha az kaynak tüketen hâle getirilmiştir. Parametreli sorgular ve CTE (Common Table Expression) yapıları kullanılarak hem okunabilirlik hem de performans artırılmıştır.

Aşağıdaki sorgunun execution plan'i analiz edilerek iyileştirme yapılmıştır:

Subquery yerine JOIN kullanılarak sorgu optimize edilmiştir:

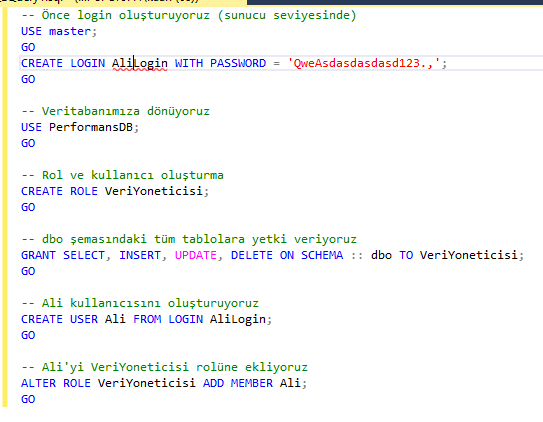




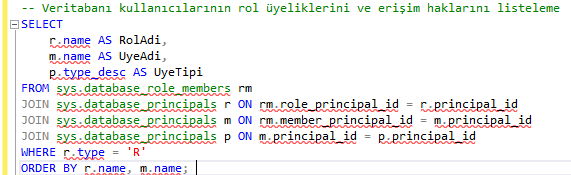
## **5. Veri Yöneticisi Rolleri**

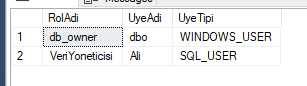
Sisteme erişen kullanıcıların rollerine göre veri yönetimi sınırlandırılmıştır. Veri yöneticileri için sadece gerekli yetkilerin tanımlandığı özel kullanıcı rolleri oluşturulmuştur. Bu sayede sistemde veri bütünlüğü ve güvenliği sağlanırken aynı zamanda operasyonel verimlilik artırılmıştır.

Yönetici rolü oluşturularak gerekli yetkiler tanımlanmıştır:



Ayrıca, veritabanı kullanıcılarının erişim hakları detaylı olarak listelenerek güvenlik kontrolü sağlanmıştır:





## **6. Sonuç**

Bu projede, SQL Server ortamında veritabanı performans optimizasyonu ve izleme teknikleri uygulamalı olarak ele alınmıştır. Sistemdeki yavaş çalışan sorgular tespit edilerek optimize edilmiş, indeks yapıları gözden geçirilmiş, veri erişim yetkileri düzenlenmiştir. Yapılan iyileştirmeler sonucunda, sistemin genel performansında önemli ölçüde artış sağlanmıştır.

Gelecek çalışmalarda, otomatik performans izleme mekanizmalarının kurulması ve düzenli bakım planlarının oluşturulması önerilmektedir.