T.C. SANAYİ VE TEKNOLOJİ BAKANLIĞI

YAPAY ZEKA UZMANLIK PROGRAMI – SENKRON EĞİTİM SONU BİTİRME PROJESİ RAPORU

POSTGRESQL VERİTABANI YAPILANDIRMASININ PERFORMANSA ETKİSİ: DOĞRU VE YANLIŞ KONFİGÜRASYONLARIN KARŞILAŞTIRMALI ANALİZİ

Hazırlayan: Eda Nur ARSLAN
Teslim Tarihi : 23/05/2025
Mentörler: Emrullah ARSEVEN, Mustafa ZAİMOĞLU

Bitirme Projesi Raporu

İÇİNDEKİLER

1. GİRİŞ

1.1 Problem Tanımı

Veritabanı performansında yapılandırma ayarları önemli bir rol oynar. Hatalı yapılandırmalarda sorgu süreleri uzar, sistem kaynakları verimsiz kullanılır. Bu proje kapsamında, aynı donanıma sahip iki sanal sunucuda biri doğru, diğeri hatalı yapılandırılmış PostgreSQL ortamları oluşturulmuştur.

1.2 Projenin Amacı

Amaç, PostgreSQL'in doğru ve yanlış yapılandırmalar altında nasıl performans gösterdiğini karşılaştırmalı olarak incelemektir. Yapılandırma parametrelerinin sistem performansı üzerindeki etkisi analiz edilerek verimli veritabanı yönetimi için öneriler sunulacaktır.

1.3 Neden Bu Karşılaştırma Yapıldı?

Verimli çalışan bir veritabanı sadece donanıma değil, yapılandırmaya da bağlıdır. Yanlış yapılandırmalar yüksek kaynak tüketimi ve düşük performansa yol açabilir. Bu nedenle farklı yapılandırmalara sahip iki PostgreSQL sisteminin kıyaslanması önemlidir.

1.4 Kullanılan Teknolojiler

- Veritabanı: PostgreSQL 14
- Programlama: Python 3
- İşletim Sistemi: Ubuntu 22.04 LTS
- Sanallaştırma: Oracle VirtualBox
- Donanım: Her bir VM için 8 GB RAM, 4 CPU

1.5 Ölçüm Yöntemi

Her iki sunucuda 10 milyon satır sahte veri ile aynı SQL sorguları çalıştırılmış, süreler ***\timing*** komutu ile ölçülmüştür. Sistem kaynak kullanımı ***top*** ve ***htop*** ile izlenmiştir. Paralel sorgular ***ThreadPoolExecutor*** kullanılarak gerçekleştirilmiştir.

1.6 Beklenen Katkı

Bu çalışma, yapılandırma ve sorgulama yöntemlerinin PostgreSQL performansı üzerindeki etkisini deneysel verilerle ortaya koymaktadır.

2. SİSTEM ORTAMI VE YAPILANDIRMALAR

2.1 Donanım ve Sanal Ortam Özellikleri

Her iki PostgreSQL sunucusu, ***Oracle VirtualBox*** üzerinde oluşturulmuş ***Ubuntu 22.04 LTS*** sanal makinelerinde yapılandırılmıştır. Her bir sunucu aynı donanımsal kaynaklara sahiptir:

- **RAM**: 8 GB
- **CPU**: 4 çekirdek
- **İşletim Sistemi**: Ubuntu 22.04 LTS (64-bit)
- **Sanallaştırma**: Oracle VirtualBox
- **PostgreSQL Sürümü**: PostgreSQL 14

Bu donanım konfigürasyonu, testlerin tutarlılığı açısından her iki ortamda da birebir korunmuştur.

2.2 Sunucu A – Doğru Yapılandırma

Sunucu A, sistem kaynaklarına uygun ve PostgreSQL performans rehberlerine uygun şekilde yapılandırılmıştır. Bazı parametreler şu şekildedir:

Parametre	Değer	Açıklama
shared_buffers	2 GB	Bellek tamponu (RAM'in yaklaşık %25'i)
work_mem	64 MB	Her sorgu işlemi için ayrılan bellek
effective_cache_size	6 GB	Önbelleklenebilir tahmini veri miktarı
max_connections	100	Maksimum eş zamanlı bağlantı sayısı
wal_buffers	16 MB	WAL tampon belleği
checkpoint_completion_target	0.9	Checkpoint işleminin ne kadar yayılacağı

Tablo 2.2.1

Bu yapılandırma PostgreSQL'in disk erişimi, sorgu işleme ve bağlantı yönetimi gibi performans faktörlerinde optimize çalışmasını sağlamıştır.

2.3 Sunucu B – Yanlış Yapılandırma

Sunucu B, bilinçli olarak verimsiz ve dengesiz parametrelerle yapılandırılmıştır. Amaç, yapılandırma hatalarının sistem performansına etkisini gözlemlemektir.

Parametre	Değer	Açıklama
shared_buffers	16 MB	Çok düşük tampon, bellek yetersiz kullanılır.
work_mem	1 MB	Gruplama ve sıralamada disk kullanımına yol açar.
effective_cache_size	512 MB	Yetersiz önbellek tahmini
max_connections	1000	Aşırı bağlantı izni, kaynak yoğunluğu
wal_buffers	256 kB	Çok düşük WAL tamponu
checkpoint_completion_target	0.1	Çok sık checkpoint işlemi

Tablo 2.3.1

---

3. VERİ SETİ VE TEST YÖNTEMİ

3.1 Tablo Yapısı

Performans testlerinde kullanılmak üzere her iki PostgreSQL sunucusunda aynı yapıya sahip bir tablo oluşturulmuştur. Tablo yapısı şu şekildedir:

```
CREATE TABLE kullanicilar (
    id SERIAL PRIMARY KEY,
    name TEXT,
    surname TEXT,
    eposta TEXT,
    dogum_tarihi DATE,
    olusturma_zamani TIMESTAMP
);
```

3.2 Veri Üretimi ve Yükleme

Tabloya her iki sunucuda da 10.000.000 satırdan oluşan sahte veri yüklenmiştir.
Veriler Python programlama dili ile ***Faker*** kütüphanesi kullanılarak rastgele olarak üretilmiş ve ***psycopg2*** kütüphanesi ile veritabanına aktarılmıştır. Veri yükleme işlemi batch (toplu) yöntemle her 10.000 satırda bir commit yapılarak gerçekleştirilmiştir.

Veri üretimi sırasında kullanılan Python modülleri:

- **faker**: Sahte ad, soyad, e-posta ve doğum tarihi üretmek için
- **psycopg2**: PostgreSQL bağlantısı ve veri aktarımı için

3.3 Test Sorguları

Veritabanı performansını karşılaştırmak için hem Sunucu A hem Sunucu B üzerinde aynı SQL sorguları çalıştırılmıştır. Test için seçilen sorgular aşağıdaki gibidir:

1. **ID ile kullanıcı arama:**

*SELECT * FROM kullanicilar WHERE id = 500000;*

2. **E-posta adresi ile kullanıcı arama:**

*SELECT * FROM kullanicilar WHERE eposta = 'ornek@example.com';*

3. **Doğum tarihi aralığında filtreleme:**

*SELECT * FROM kullanicilar WHERE dogum_tarihi BETWEEN '1980-01-01' AND '2000-12-31';*

4. **Soyadına göre gruplama ve sıralama:**

SELECT surname, COUNT(*) FROM kullanicilar GROUP BY surname ORDER BY COUNT(*) DESC;

3.4 Ölçüm Yöntemi

Her sorgunun çalıştırılma süresi PostgreSQL'in yerleşik zaman ölçüm aracı olan ***\timing*** komutu ile ölçülmüştür.

---

4. PERFORMANS SONUÇLARI

4.1 SQL Sorgularının Çalışma Süreleri

Performans testi kapsamında her iki sunucuda aynı SQL sorguları çalıştırılmış ve sorguların çalışma süreleri ***\timing*** komutu ile ölçülmüştür. Elde edilen sonuçlar aşağıdaki tabloda gösterilmektedir:

Test Sorgusu	Sunucu A (Doğru Konf.) [ms]	Sunucu B (Yanlış Konf.) [ms]	Fark (ms)	Performans Kaybı
ID ile kullanıcı arama	0.924	38.497	37.573	4066.341991
E-posta ile kullanıcı arama	524.458	1933.045	1408.587	268.5795621
Doğum tarihi aralığında filtreleme	2730.108	2958.511	228.403	8.366079291
Soyadına göre gruplama ve sıralama	775.976	792.507	16.531	2.130349392

Tablo 4.1.1

Sonuçlara göre, doğru yapılandırılmış Sunucu A, tüm sorgularda daha düşük sürelerle sonuç üretmiştir. Özellikle ID ile sorgulamada Sunucu B, Sunucu A'ya kıyasla yaklaşık **40 kat daha yavaş** çalışmıştır. En belirgin fark, bellek kullanımı gerektiren e-posta arama sorgusunda görülmüş; Sunucu B yaklaşık **4 kat daha fazla süre** harcamıştır.

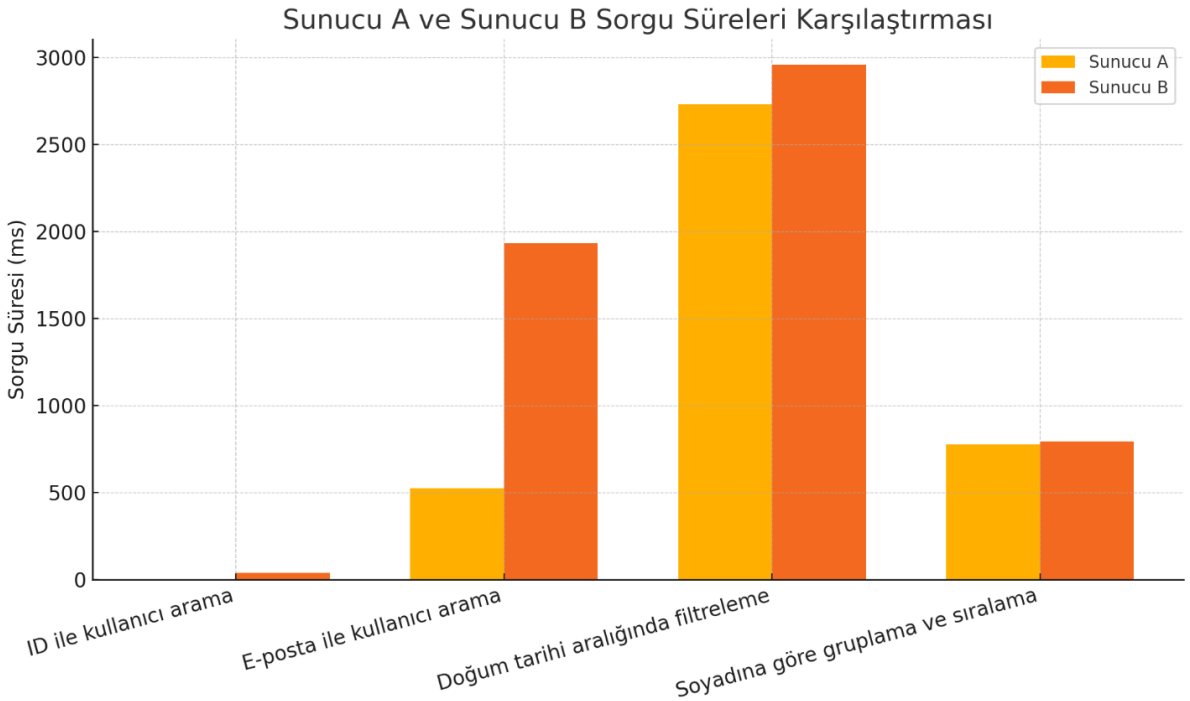
4.2 Sistem Kaynak Kullanımı (CPU ve RAM)

Sorgular çalıştırılırken ***htop*** ve ***top*** araçları kullanılarak sistem kaynak kullanımları gözlemlenmiştir. Sunucu A, belleği daha verimli kullanırken, Sunucu B'de RAM kullanımının hızla arttığı ve işlemcinin ani sıçramalarla yoğun yüke maruz kaldığı görülmüştür.

Gözlem Noktası	Sunucu A (Doğru Konfigürasyon)	Sunucu B (Yanlış Konfigürasyon)
Ortalama RAM kullanımı	1.5 GB	2.8–3.0 GB
Ortalama CPU kullanımı	%35–45	%80–90
Tepki süresi	Düşük	Yüksek

Tablo 4.2.1

Sunucu B'de yapılandırma hataları nedeniyle disk erişimi artmış, CPU sürekli olarak aktif kalmış ve belleğin yetersiz kullanımı sistem performansını olumsuz etkilemiştir.



Grafik 4.2.1

5. PARALEL PROGRAMLAMA TESTİ SONUÇLARI

5.1 Testin Amacı

Bu testte, Python dili kullanılarak sıralı ve paralel sorgu işleme yöntemlerinin veritabanı performansına etkisi incelenmiştir. Her iki yapılandırma altında PostgreSQL veritabanına yapılan sorguların süresi ölçülmüş ve elde edilen sonuçlar karşılaştırılmıştır.

5.2 Yöntem

Test senaryosu olarak 10 farklı kullanıcı id'sine karşılık gelen ***SELECT*** sorguları belirlenmiştir. Bu sorgular iki farklı Python programı aracılığıyla gerçekleştirilmiştir:

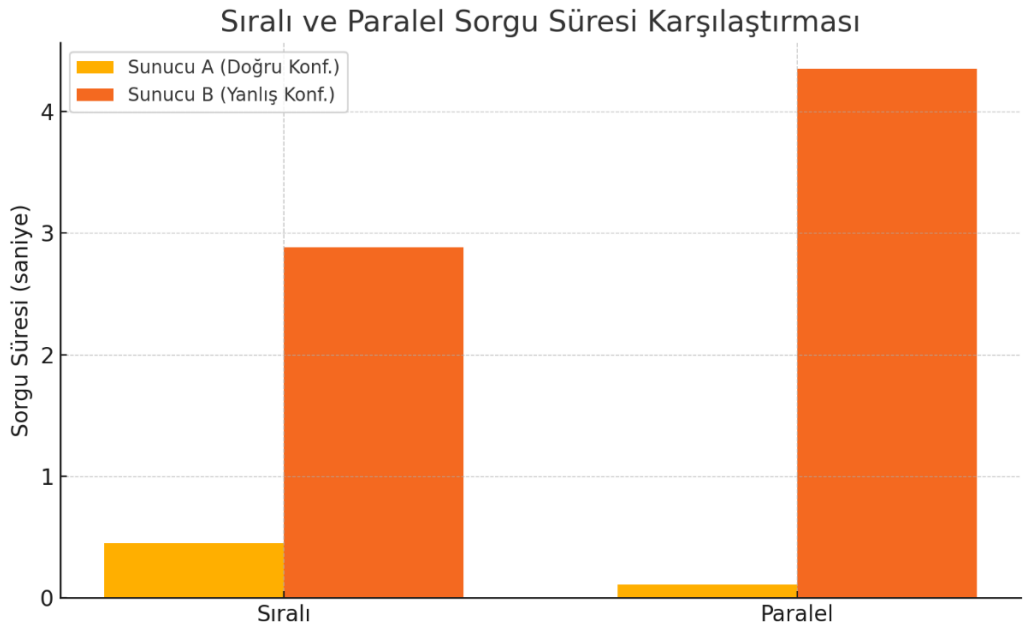
- **sequential_test.py**: Sorgular tek tek, sıralı olarak çalıştırılmıştır.
- **parallel_test.py**: Sorgular eş zamanlı olarak, 10 iş parçacığı (thread) ile çalıştırılmıştır.

Python'un ***concurrent.futures.ThreadPoolExecutor*** sınıfı kullanılarak paralel sorgulama gerçekleştirilmiştir. Her iki program da iki sunucu üzerinde çalıştırılmıştır.

5.3 Test Sonuçları

Test Türü	Sunucu A Süre (sn)	Sunucu B Süre (sn)
Sıralı(sequential_test.py)	0.45	2.88
Paralel(parallel_test.py)	0.11	4.35

Tablo 5.3.1



Grafik 5.3.1

5.4 Analizler

- **Sunucu A**'da paralel sorgulama yöntemi, sıralı çalışmaya göre yaklaşık **4 kat daha hızlı** sonuç vermiştir. Bu durum, yapılandırmanın sistem kaynaklarını verimli kullandığını ve eş zamanlı bağlantılara uygun olduğunu göstermektedir.
- **Sunucu B**'de ise paralel sorgulama beklenenin aksine **daha uzun sürede** tamamlanmıştır. Bunun temel nedeni, yetersiz ***work_mem, shared_buffers ve yüksek max_connections*** değerlerinin sistem kaynaklarını zorlaması ve thread'lerin çakışmasına neden olmasıdır.
- *Bu test, yalnızca donanım değil, **yapılandırma kalitesinin** paralel programlamada doğrudan etkili olduğunu göstermiştir.*

---

6. SONUÇ VE DEĞERLENDİRME

Bu projede, PostgreSQL veritabanı yönetim sisteminin doğru ve yanlış yapılandırmalar altında gösterdiği performans farklılıkları gözlemlenmiş; sıralı ve paralel sorgulama teknikleri ile karşılaştırma yapılmıştır:

- Doğru yapılandırılmış Sunucu A, her sorguda istikrarlı ve düşük sürelerle yanıt verirken; yanlış yapılandırılmış Sunucu B, özellikle kaynak yoğun sorgularda yüksek gecikme ve bellek tüketimi göstermiştir.
- Sıralı sorgulama testlerinde, yapılandırma farkı doğrudan sorgu sürelerine yansımıştır. Paralel sorgulama testlerinde ise, Sunucu A çok daha verimli çalışırken; Sunucu B'de paralel işlemler verimi düşürmüş ve kaynak darboğazına neden olmuştur.

Sistem kaynaklarının yüksek olması tek başına yeterli değildir; yapılandırma parametrelerinin bu kaynaklara uygun şekilde ayarlanması kritik öneme sahiptir. Ayrıca, paralel programlama yöntemleri yalnızca doğru yapılandırılmış sistemlerde etkili sonuçlar vermektedir. Yetersiz yapılandırmalarda, paralel yapı avantaj sağlamadığı gibi sistem kararlılığını da olumsuz etkileyebilmektedir.

Öneriler:

- PostgreSQL kurulumu sonrası varsayılan yapılandırma ayarları mutlaka optimize edilmelidir.
- ***shared_buffers, work_mem, effective_cache_size, wal_buffers*** gibi temel parametreler sistem kaynaklarına göre ayarlanmalıdır.
- Paralel sorgulama gibi gelişmiş kullanım senaryolarında yapılandırma, kaynak planlama ve izleme araçlarıyla desteklenmelidir.

7. KAYNAKÇA

1. PostgreSQL Global Development Group. (2024). *PostgreSQL 14 Documentation*. Erişim adresi: https://www.postgresql.org/docs/
2. Python Software Foundation. (2024). *The Python Standard Library — concurrent.futures*. Erişim adresi: https://docs.python.org/3/library/concurrent.futures.html
3. Faker Community. (2024). *Faker Documentation*. Erişim adresi: https://faker.readthedocs.io/en/master/
4. Psycopg Development Team. (2024). *psycopg2 - PostgreSQL adapter for Python*. Erişim adresi: https://www.psycopg.org/docs/
5. The htop Project. (2024). *htop - Interactive Process Viewer*. Erişim adresi: https://htop.dev/
6. The Linux Foundation. (2024). *top Command in Linux with Examples*. Erişim adresi: https://man7.org/linux/man-pages/man1/top.1.html
7. Oracle Corporation. (2024). *VirtualBox Documentation*. Erişim adresi: https://www.virtualbox.org/manual/
8. Ubuntu Documentation. (2024). *Ubuntu Server Guide: PostgreSQL*. Erişim adresi: https://ubuntu.com/server/docs/databases-postgresql