T.C SANAYİ VE TEKNOLOJİ BAKANLIĞI

MİLLİ TEKNOLOJİ AKADEMİSİ

YAPAY ZEKA UZMANLIK PROGRAMI BİTİRME PROJESİ

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

Atalay BEKTAŞ

Giriş

Veritabanı performansı, yapılandırma ayarlarından ve sorguların işlenme yöntemlerinden doğrudan etkilenir.Yanlış yapılandırılmış sistemlerde, sorgular daha yavaş çalışır ve kaynaklar verimli kullanılamaz. Bu çalışmada, aynı donanım özelliklerine sahip iki sanal sunucuya PostgreSQL kurulmuş bunlardan biri doğru, diğeri ise kasıtlı olarak hatalı yapılandırılmıştır..

Veritabanı sistemlerinin etkin ve verimli çalışabilmesi yalnızca donanım özelliklerine değil, aynı zamanda yapılandırma ayarlarının doğruluğuna da bağlıdır. Uygun şekilde yapılandırılmış bir veritabanı, sistem kaynaklarının daha verimli kullanılmasını sağlar ve genel performansı önemli ölçüde artırır.

Bu çalışmada PostgreSQL veritabanının doğru ve yanlış yapılandırmalar altındaki performansını karşılaştırmalı olarak analiz edilmiştir ve paralel programlamanın sistem performansına etkisi incelenmiştir.

Sunucu Özellikleri

Oracle VirtualBox üzerinden 2 adet sanal sunucu ortamı oluşturulmuş ve PostgreSQL 17 kurulmuştur.

- 8 GB RAM, 4 CPU

-Ubuntu 22.04 LTS

-100 GB SSD

Konfigürasyon Detayları

PostgreSQL yapılandırması PostgreSQL performans rehberlerine uygun şekilde PGTune (https://pgtune.leopard.in.ua/) aracı kullanılarak optimize edilmiştir,

Fakat en iyi performans için diğer parametrelerin de dikkate alınması gerekmektedir.

PGTune, PostgreSQL için donanım özelliklerine dayanarak en iyi performansı elde etmek amacıyla yapılandırma ayarlarını önerir. Ancak, bu öneriler ayarların etkisi sadece donanımla değil, aynı zamanda veritabanının büyüklüğü, eşzamanlı kullanıcı sayısı ve sorguların karmaşıklığı gibi faktörlerle de ilgilidir bu faktörler çalışmanın kapsamında değildir.

Sunucu A – Doğru Konfigürasyon

Sunucu A doğru yapılandırılma parametreleriyle yapılandırılmıştır. Sunucu A için bazı parametler aşağıdadır. Bu ayarlar, PostgreSQL'in bellek kullanımı, bağlantı kapasitesi, veri önbellekleme ve veri yazma performansını optimize ederek sistemin verimli ve stabil çalışmasını sağlar.

Parametre	Değer	Açıklama
shared_buffers	2 GB	Bellek tamponu

work_mem	64 MB	Her sorgu işlemi için ayrılan bellek
effective_cache_size	6 GB	Önbelleklenebilir 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

Sunucu B – Yanlış Konfigürasyon

Sunucu B Ynalış yapılandırılma paremetreleriyle yapılandırılmıştır. Performansı olumsuz etkileyebilecek, verimsiz değerler bilinçli olarak tercih edilmiştir.

Parametre	Değer	Açıklama
shared_buffers	16 MB	Düşük tampon, bellek yetersiz kullanılır.
work_mem	1 MB	Her sorgu için yetersiz düşük bellek disk kullanımına yol açar.
effective_cache_size	512 MB	Düşük önbellek tahmini
max_connections	1000	Aşırı büyük bağlantı sayısı
wal_buffers	256 kB	Çok düşük WAL tamponu
checkpoint_completion_target	0.1	Çok sık checkpoint işlemi

Veri Seti Oluşturma

PostgreSQL veritabanına testlerde kullanılması için sahte verilerden oluşan tablo oluşturulmuştur.

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

Veri ekleme işlemi, PostgreSQL'in generate_series fonksiyonu kullanılarak 1'den 10 milyona kadar döngü ile gerçekleştirilmiştir. Bu sayede tabloya büyük miktarda, gerçekçi yapıya benzer test verisi hızlıca eklenmiştir.

Ölçüm Yöntemi

Her iki sunucuda 10 milyon satır sahte veri ile aynı SQL sorguları çalıştırılmıştır. Sistem kaynak kullanımı top ve htop ile izlenmiştir. Paralel sorgular ThreadPoolExecutor kullanılarak gerçekleştirilmiştir.

Test Sorguları

Test için hem Sunucu A hem Sunucu B üzerinde çalıştırılan SQL sorguları aşağıdadır.

1. **ID ile kullanıcı arama**:
   ```
   SELECT * FROM kullanicilar
   WHERE id=817;
   ```
2. **E-posta adresi ile kullanıcı arama**:
   ```
   SELECT * FROM kullanicilar
   WHERE eposta = 'eposta_4475276@example.com';
   ```
3. **Doğum tarihi aralığında filtreleme**:
   ```
   SELECT * FROM kullanicilar
   WHERE dogum_tarihi BETWEEN '2024-09-24' AND '2024-09- 25';
   ```
4. **Soyadına göre gruplama ve sıralama**:
   ```
   SELECT surname, COUNT(*) as count
    FROM kullanicilar
   WHERE surname= 'soyad_5227846
   GROUP BY surname
   ORDER BY count DESC'
   ```

Sonuçlar

Sorgu	Sunucu A Süresi (sn)	Sunucu B Süresi (sn)	Fark (sn)	Yüzdesel Fark (%)
Id ile getirme	0.115	0.311	0.196	170.43%
E-posta ile getirme	0.533	0.781	0.248	46.52%

Doğum tarihi aralığı filtreleme	0.360	0.629	0.269	74.72%
Surname'e göre gruplama ve sıralama	0.325	0.680	0.355	109.23%

Yapılan sorgu testlerinde, optimize edilen sunucu (sunucu A) tüm sorgularda daha hızlı yanıt vermiştir.

Sistem Kaynak Kullanımı (CPU ve RAM)

CPU ve RAM kullanımı top ve htop komutuyla gözlemlendi. Linux'un top komutu, sistemde çalışan işlemlerin anlık CPU, RAM ve diğer kaynak kullanımlarını gerçek zamanlı olarak gösteren temel bir performans izleme aracıdır.

Sorguların çok hızlı çalışması nedeniyle daha sağlıklı gözlemler yapabilmek amacıyla sorgular defalarca tekrarlandı.

Sunucu A'da RAM kullanımı dengeli ve stabil olarak gözlemlenmiştir. CPU kullanımı ise dengeli ve yoğun bir şekilde gerçekleşmektedir. Buna karşın, Sunucu B'de RAM kullanımı düşük, dalgalı ve ani yükselişler, CPU kullanımı ise dalgalı ve ani yoğun yüke maruz seviyelerde gözlemlenmiştir.

Sunucu A'da RAM ve CPU kullanımı daha verimli, dengeli olarak gözlemlenmiştir. Sunucu B de RAM kulanımında düşük seviyelerden ani artışlarla dengesiz dalgalı ve ani yükselişler gözlenmiştir. CPU'da ise düşük kullanım seviyesinden ani yükselişlerle sunucu A ya göre daha yüksek CPU kullanımı seviyelerine çıktığı gözlemlenmiştir.

Kaynak	Sunucu A (Doğru Konfigürasyon)	Sunucu B (Yanlış Konfigürasyon)
RAM kullanımı	%20	%40
CPU kullanımı	%30	%50

Sunucu A daha yüksek veri tabanını performansını sistem kaynaklarını daha verimli kullanarak sağlamıştır

Paralel Programlama Testi

PostgreSQL veritabanında yapılan sorguların paralel olarak çalıştırılması performans açısından test edilmiştir. Bu amaçla Python programı kullanılarak veritabanına bağlantı sağlanmış ve test sorguları gerçekleştirilmiştir. Sorgular, Python'da concurrent.futures kütüphanesinde ThreadPoolExecutor kullanılarak threadlerle paralel biçimde çalıştırılmıştır. Test kapsamında 10 adet kullanıcıya ait ID'ler tek tek sorgulanmıştır. Bu sorgular önce sıralı olarak çalıştırılmış ve geçen süre ölçülmüştür. Ardından aynı sorgular paralel olarak çalıştırılmış ve süreler karşılaştırılmıştır.

test_sıralı.py ile tek tek, sıralı olarak sorgular çalıştırılmıştır.

test_paralel.py ile paralel olarak 10 thread ile sorgular çalıştırılmıştır.

Test	Sunucu A Süre (sn)	Sunucu B Süre (sn)
Sıralı	0.692	1.67
Paralel	0.316	2.92

Sunucu A'da paralel sorgulama yöntemi, daha iyi sonuç vermiştir. Yapılandırma sistem kaynaklarının eş zamanlı paralel sorgulama için optimize edilmiştir.

Sunucu B'de ise paralel sorgulama daha kötü performans vermiştir. yetersiz work_mem, shared_buffers ve yüksek max_connections gibi paremetler eş zamanlı paralel programlamada CPU çekirdeklerinin verimsiz kullanımına yol açarak threadlerin sağlıklı çalışamamasına yol açmıştır.

Değerlendirme

Doğru ve yanlış yapılandırılmış PostgreSQL veritabanı performansı, sıralı ve paralel sorgulama yöntemleriyle karşılaştırılmıştır. Doğru yapılandırılmış Sunucu A daha iyi performans göstermiştir. Yanlış yapılandırılmış Sunucu B ise yüksek gecikme ile birlikte daha yüksek RAM ve CPU tüketimi sergilemiştir.

Eş zamanlı paralel sorgulama yöntemlerinde, Sunucu A'da paralel sorgulamalar performansı artırırken; Sunucu B'de yanlış yapılandırma, paralel sorguların sıralı sorgulardan daha kötü performans göstermesine ve sistem kaynaklarında darboğaza yol açmıştır.

Yapılandırma parametrelerinin, sistem kaynaklarına uygun şekilde seçilmesi, donanım kaynaklarının etkin kullanımı açısından önemlidir. Eş zamanlı paralel programlama yöntemleri için yapılandırma parametrelerinin optimize edilmesi kritiktir; aksi hâlde, sistem performansını olumlu etkilemek yerine olumsuz etkileyebilir.