

PostgreSQL Sıralama ve Relevance (Alaka Düzeyi) Özeti

Klasik senaryolarda `ORDER BY` ifadesi, sayısal veya tarihsel verileri **deterministik** (kesin) bir şekilde sıralamak için yeterlidir. Ancak metin tabanlı aramalarda "en doğru sonucu en üste getirme" ihtiyacı doğduğunda standart yöntemler yetersiz kalır. `LIKE` veya `ILIKE` gibi operatörler yalnızca filtreleme yapar, bir alaka skoru üretmezler.

Gelişmiş metin araması için PostgreSQL iki temel yol sunar:

1. **Full-Text Search (ts_rank):** Yerleşik bir çözümdür. Kelime frekansına göre skor üretir ancak büyük veri setlerinde önce tüm sonuçları bulup sonra sıraladığı için performans kaybına yol açabilir.
2. **Modern BM25 Yaklaşımı (<@>):** `pg_textsearch` gibi eklentilerle kullanılan bu yöntem, arama motoru standartlarında (BM25 algoritması) skorlama yapar. En büyük avantajı, `ORDER BY` ve `LIMIT` ile birlikte kullanıldığında tüm satırları taramak yerine doğrudan en alakalı sonuçlara odaklanarak yüksek performans sunmasıdır.

Sonuç olarak; basit listelemeler için standart `ORDER BY` idealdir; ancak kullanıcı deneyimini ön plana çıkaran "akıllı" bir arama sistemi için BM25 operatörü hem hız hem de alaka düzeyi açısından en verimli modern yaklaşımdır.

PostgreSQL'de metin sıralama ekosistemi, sadece tam kelime eşleşmeleriyle sınırlı değildir. Yazım hatalarını tolere eden (fuzzy), okunuş tabanlı (phonetic) ve yüksek performanslı (block-based) algoritmalarla bu yelpaze genişler. İşte araştırmanıza dahil edebileceğiniz ileri seviye sıralama yöntemleri:

1. Benzerlik ve Mesafe Odaklı Sıralama (Trigram & Levenshtein)

Bu yöntemler, kelimelerin birbirine görsel veya yapısal olarak ne kadar benzediğine odaklanır.

- **pg_trgm (Trigram Similarity):** Metni üçer karakterlik bloklara böler. `similarity(col, query)` fonksiyonu veya `%` operatörü ile kullanılır. Özellikle **yazım hatalarında** ve "içerir" sorgularında `ORDER BY similarity(...) DESC` şeklinde en yakın sonucu bulmak için idealdir.
- **Levenshtein Mesafesi (fuzzystrmatch):** Bir metni diğerine dönüştürmek için gereken minimum karakter değişikliği sayısını hesaplar. `ORDER BY levenshtein(name, 'target')` ASC şeklinde kullanılır. Kısa metinlerde (isim, soyisim) çok hassas sonuç verir ancak işlem maliyeti yüksektir.

2. Fonetik (Okunuş) Tabanlı Sıralama

Kullanıcının kelimeyi nasıl yazacağını bilmediği, sadece duyduğu gibi yazdığı senaryolarda kullanılır.

- **Soundex & Metaphone (fuzzystmatch):** Kelimeleri telaffuz kodlarına dönüştürür. Örneğin, "Smith" ve "Smyth" aynı kodu üretir. `WHERE metaphone(name, 10) = metaphone('smyth', 10)` filtresiyle beraber ikincil bir benzerlik puanlamasıyla sıralama yapılabilir.

3. İleri Seviye Optimizasyon: Block-Max WAND

Büyük veri setlerinde "top-k" (en alakalı ilk k sonuç) sıralamasını hızlandırmak için kullanılan bir algoritmadır.

- **Çalışma Mantığı:** Geleneksel yöntemler tüm eşleşen dokümanları skorlayıp sonra sıralarken; Block-Max WAND, skor potansiyeli düşük olan blokları (chunk) hiç incelemiden atlar.
- **Avantajı:** PostgreSQL'in modern eklentilerinde (ve Lucene tabanlı motorlarda) kullanılır. `ORDER BY + LIMIT` sorgularında, sadece en yüksek skor ihtimali olan veri bloklarını işleyerek disk I/O ve CPU kullanımını radikal şekilde düşürür.

Özet Karşılaştırma

Yöntem	Kullanım Amacı	Avantajı
Trigram	Yazım yanlışları, eksik kelime	GIN/GIST index desteği ile hızlıdır.
Levenshtein	Karakter bazlı tam değişim	Çok yüksek hassasiyet.
Metaphone	Okunuş benzerliği	"Duyulduğu gibi" aramalarda rakipsiz.
BM25 / WAND	Alaka düzeyi ve Performans	Milyonlarca satırda en alakalı ilk 10 sonuç.

Bu yöntemleri kullanarak, sisteminizde hem "ali" yazınca "aliye"yi (trigram), hem "jon" yazınca "john"u (levenshtein), hem de "feyzbuk" yazınca "facebook"u (metaphone) en üste getirecek hibrit bir sıralama kurgulayabilirsiniz.

Özellik	Klasik Postgres FTS	pg_textsearch (BM25) Block-Max WAND (🔥)	Elasticsearch
Relevance kalitesi	Orta	Yüksek (BM25)	Çok yüksek
ORDER BY + LIMIT optimizasyonu	Zayıf	Yüksek	Var
Top-k performansı	Orta	Çok iyi	Çok iyi
Infra maliyeti	Yok	Yok	Yüksek
Operasyonel karmaşıklık	Düşük	Düşük	Yüksek
Dil desteği	Var	Postgres config'leri	Var
Partition awareness	Kısıtlı	Var (ama dikkat)	Var
GA olgunluğu	Çok olgun	🚧 (v1.0-dev)	Çok olgun