Laboratorio 04: Comparación de Rendimiento entre BRIN y B+ Tree

Prof. Heider Sanchez
ACLs: Ana María Accilio, Sebastián Loza

Introducción

En sistemas que manejan grandes volúmenes de datos con ordenamiento natural como registros de sensores IoT, los índices tradicionales B+ Tree pueden no ser óptimos en espacio ni velocidad. PostgreSQL ofrece índices BRIN, más ligeros y eficientes para datos que presentan correlación física, especialmente cuando las consultas son por **rangos extensos**.

En este laboratorio se le pide analizar y comparar el rendimiento de los índices **BRIN** y **B+ Tree** en PostgreSQL mediante un caso práctico que simule la lectura de sensores. Se utilizarán **dos tablas idénticas** con distinto tipo de índice sobre la columna "fecha", evaluando su impacto en el rendimiento de consultas por rango.

Diseño del Experimento

Se le pide crear una tabla para almacenar datos de sensores de temperatura. Simulación de sensores que registran temperatura cada minuto durante un año. El total será mas de 500 mil registros por tabla, representando lecturas de múltiples sensores.

1. Preparación de Tablas

```sql
-- Tabla con índice B+ Tree
CREATE TABLE temperatura_btree (
    id SERIAL PRIMARY KEY,
    sensor_id INT,
    fecha TIMESTAMP,
    temperatura DECIMAL(5,2)
);

-- Tabla con índice BRIN
CREATE TABLE temperatura_brin (
    id SERIAL PRIMARY KEY,
    sensor_id INT,
    fecha TIMESTAMP,
    temperatura DECIMAL(5,2)
);

-- Población de ambas tablas con los mismos datos
INSERT INTO temperatura_btree (sensor_id, fecha, temperatura)
SELECT
    (RANDOM()*9 + 1)::INT,
    TIMESTAMP '2024-01-01' + (i || ' minutes')::INTERVAL,
    ROUND(CAST(20 + RANDOM()*10 AS NUMERIC), 2)
FROM generate_series(1, 525600) AS s(i);

INSERT INTO temperatura_brin SELECT * FROM temperatura_btree;
```

2. Creación de Índices

```sql
-- Índice B+ Tree
CREATE INDEX idx_fecha_btree ON temperatura_btree(fecha);

-- Índice BRIN
CREATE INDEX idx_fecha_brin ON temperatura_brin USING BRIN(fecha);
```

> ¿Qué índice se creo más rápido?

Consultas para Medir Rendimiento

Consultas por rangos pequeños

```
-- B+ Tree
EXPLAIN ANALYZE
SELECT * FROM temperatura_btree
WHERE fecha BETWEEN '2024-06-01 00:00:00' AND '2024-06-01 01:00:00';

-- BRIN
EXPLAIN ANALYZE
SELECT * FROM temperatura_brin
WHERE fecha BETWEEN '2024-06-01 00:00:00' AND '2024-06-01 01:00:00';
```

> ¿Quien obtuvo mejores resultados?

Consultas por rangos medianos

```
-- 1 día de datos
EXPLAIN ANALYZE
SELECT * FROM temperatura_btree
WHERE fecha BETWEEN '2024-06-01' AND '2024-06-02';

EXPLAIN ANALYZE
SELECT * FROM temperatura_brin
WHERE fecha BETWEEN '2024-06-01' AND '2024-06-02';
```

> ¿Quien obtuvo mejores resultados?

Consultas por rangos grandes

```
-- 1 mes de datos
EXPLAIN ANALYZE
SELECT * FROM temperatura_btree
WHERE fecha BETWEEN '2024-06-01' AND '2024-07-01';

EXPLAIN ANALYZE
SELECT * FROM temperatura_brin
WHERE fecha BETWEEN '2024-06-01' AND '2024-07-01';
```

> ¿Quien obtuvo mejores resultados?

Ejercicios Propuestos para el Alumno

1. (5 pts) Comparar tiempos de ejecución

- Ejecute varias consultas con `EXPLAIN ANALYZE` y registre el tiempo promedio de ejecución:
    - Rango de 1 minuto

- Rango de 1 hora
- Rango de 1 día
- Rango de 1 mes
- Rango de 2 meses

- Grafique los resultados para comparar el rendimiento entre B+ Tree y BRIN. Gráfico sugerido:

Rango	B+ Tree (ms)	BRIN (ms)
1 minuto	5	95
1 hora	25	80
1 día	60	40
1 mes	450	70
2 meses	850	60

- **Nota:** Aplique `VACUUM ANALYZE` o reinicie el servidor para asegurarse de que las lecturas sean desde disco. El objetivo es tener tiempos realistas de acceso a disco en lugar de caché.

2. (1 pts) Medir tamaño de índices

- Usar la función `pg_relation_size` para obtener el tamaño de los índices creados.
- **Comando:**

```
SELECT pg_size_pretty(pg_relation_size('idx_fecha_btree')) AS btree,
       pg_size_pretty(pg_relation_size('idx_fecha_brin')) AS brin;
```

- Insertar la misma cantidad de registros inicial pero para todo el año 2025 en ambas tablas.
- Vuelva a ejecutar el comando y registre los tamaños de los índices.
- Explique los resultados al comparar el espacio ocupado por B+ Tree y BRIN.

3. (5 pts) Optimización del índice BRIN

- Modifica el parámetro `pages_per_range` del índice BRIN y evalúa su impacto en el rendimiento.

```
DROP INDEX idx_fecha_brin;
CREATE INDEX idx_fecha_brin ON temperatura_brin
USING BRIN(fecha) WITH (pages_per_range = 64);
```

- Variar la cantidad de páginas por rango con los siguientes valores: 64, 32, 16, 8 y 4.
- Construya una tabla comparativa y explique lo que esta pasando. En la primera fila de la tabla coloque los tamaños del índice.

Rango	B+ Tree (ms)	BRIN-64(ms)	BRIN-32(ms)	BRIN-16(ms)	BRIN-08(ms)	BRIN-04(ms)
Space (mb)						
1 minuto						
1 hora						
1 día						
1 mes						
2 meses						

4. (1 pts) Consulta con filtro compuesto

- Agregar un indice en la columna `sensor_id` .
- Realizar una consulta por `fecha` y `sensor_id` al mismo tiempo. Observa si los índices se usan correctamente.

```sql
SELECT * FROM temperatura_btree
WHERE fecha BETWEEN '2024-06-01' AND '2024-06-02'
  AND sensor_id = 5;
```

- Registre su observación en base al resultado obtenido.

5. (6 pts) Forzar desorden en datos BRIN

- Insertar datos aleatorios (las fechas en desorden) en una nueva tabla `temperatura_brin_desordenada` , crear un índice BRIN, y compare los resultados con la tabla ordenada.
- Elabore el cuadro comparativo
- Registre su observación en base al resultado obtenido.

6. (2 pts) Responder las siguientes preguntas

- ¿En qué escenarios gana el índice BRIN? ¿Por qué?
- ¿Qué desventajas tiene BRIN frente a B+ Tree?
- ¿Qué pasaría si se mezclan los datos en la tabla con BRIN?
- ¿Es recomendable usar ambos índices al mismo tiempo?
- ¿En qué otros casos reales conviene usar BRIN? Fundamente.