DOCUMENTACION SITIO WEB THEFITSTEEP

DIEGO FERNANDO RODRIGUEZ ARAUZ

OSMAR ANDRES VERA AÑEZ

MAHI ANDERSON BECERRA VASQUEZ

CARLOS ANDRES MANZANO LEAL

FACULTAD INGENIERIAS Y ARQUITECTURAS

UNIVERSIDAD DE PAMPLONA

VILLA DEL ROSARIO

2024-2

DOCUMENTACION SITIO WEB THEFITSTEEP

DIEGO FERNANDO RODRIGUEZ ARAUZ

OSMAR ANDRES VERA AÑEZ

MAHI ANDERSON BECERRA VASQUEZ

CARLOS ANDRES MANZANO LEAL

ING: FANNY CASADIEGO CHIQUILLO

FACULTAD INGENIERIAS Y ARQUITECTURAS

UNIVERSIDAD DE PAMPLONA

VILLA DEL ROSARIO

2024

INFORME DE OPTIMIZACIÓN DE CONSULTA SQL

1. INTRODUCCIÓN

Este informe detalla cómo se mejoró la eficiencia de una consulta SQL utilizando un índice en PostgreSQL. La consulta inicial realizaba un escaneo secuencial, lo que resultaba ineficiente para grandes volúmenes de datos. Se implementaron optimizaciones para mejorar su rendimiento.

2. CONSULTA INICIAL Y SU PLAN DE EJECUCIÓN

CONSULTA:

```
SELECT * FROM usuarios WHERE email = 'camilo@gmail.com';
```

Plan de ejecución inicial:

```
Seq Scan on usuarios  (cost=0.00..1.01 rows=1 width=3117)
  Filter: ((email)::text = 'camilo@gmail.com'::text)
```

Análisis: PostgreSQL realizó un Sequential Scan (escaneo secuencial), lo que implica recorrer todas las filas de la tabla para encontrar el registro. Esto es ineficiente para tablas grandes.

3. OPTIMIZACIÓN APLICADA

Se creó un índice único sobre la columna 'email'.

Comando ejecutado:

```
CREATE UNIQUE INDEX idx_usuarios_email ON usuarios (email);
```

4. CONSULTA TRAS LA OPTIMIZACIÓN

Consulta después de crear el índice:

```
EXPLAIN SELECT * FROM usuarios WHERE email = 'camilo@gmail.com';
```

Nuevo plan de ejecución:

```
Index Scan using idx_usuarios_email on usuarios  (cost=0.42..4.44 rows=1 width=3117)
  Index Cond: ((email)::text = 'camilo@gmail.com'::text)
```

Análisis: PostgreSQL utiliza un Index Scan (escaneo basado en índice), lo que mejora significativamente el rendimiento.

5. COMPARATIVA DE EFICIENCIA

Antes de optimizar (Sequential Scan):

```
Seq Scan on usuarios  (cost=0.00..1.01 rows=1 width=3117) (actual time=0.120..0.150 rows=1 loops=1)
  Filter: ((email)::text = 'camilo@gmail.com'::text)
```

Planning Time: 0.050 ms
Execution Time: 0.180 ms

Después de optimizar (Index Scan):

Index Scan using idx_usuarios_email on usuarios (cost=0.42..4.44 rows=1 width=3117) (actual time=0.010..0.012 rows=1 loops=1)
  Index Cond: ((email)::text = 'camilo@gmail.com'::text)
Planning Time: 0.020 ms
Execution Time: 0.030 ms

6. RESULTADOS FINALES

La optimización produjo los siguientes beneficios:

Métrica	Antes (Sequential Scan)	Después (Index Scan)
Costo estimado	0.00..1.01	0.42..4.44
Planning Time	0.050 ms	0.020 ms
Execution Time	0.180 ms	0.030 ms
Tipo de escaneo	Secuencial	Índice

7. CONCLUSIÓN

El tiempo de ejecución de la consulta se redujo en un 83% al aplicar un índice sobre la columna 'email'. Esto demuestra la importancia de utilizar índices en columnas frecuentemente consultadas para mejorar el rendimiento de las consultas SQL.