

Crear índices

Antonio Espín Herranz

Indexación en PostgreSQL

- Nos ayudan a buscar datos de una forma más rápida
- Pero también añade consumo de recursos en escrituras y almacenamiento.
- Desde postgresql podemos:
 - Crear índices
 - Eliminar índices
 - Listar los índices disponibles en postgresql

Indexación en PostgreSQL

- Tipos de índices
 - Índices únicos
 - Índices basados en una expresión
 - Índices parciales
 - Reindexar índices para reconstruir uno o más índices.
 - Índices multicolumnas

Crear índices

Create index [if not exists] nombreIndice

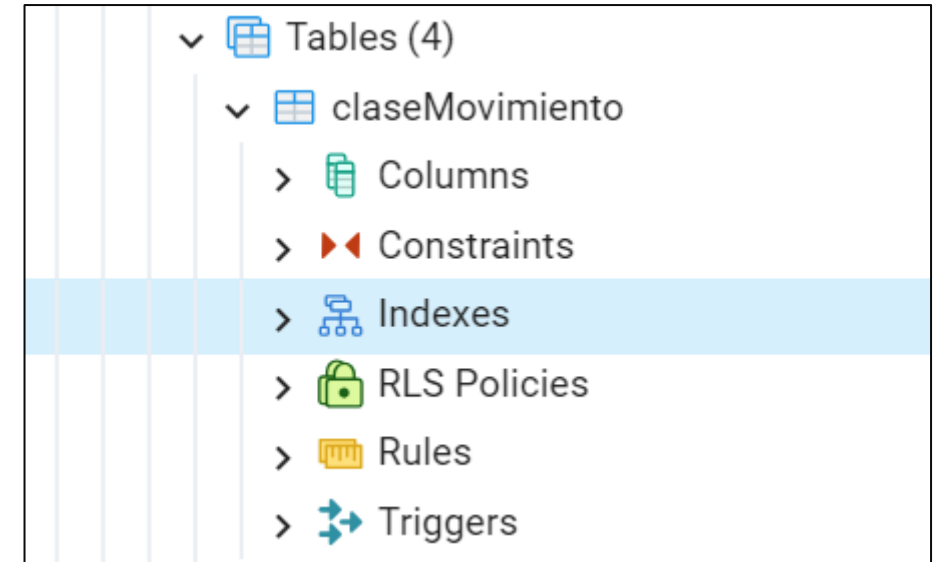
On nombreTabla

(columna-name [ASC | DESC] ...)

Pueden ser varias columnas.

El orden se puede especificar por cada columna

- Los índices se pueden consultar a nivel de la tabla en pgAdmin 4
- El índice tiene que ser único dentro de la BD de postgresql.
- Si el nombre del índice lleva mayúsculas utilizar ""
- Por defecto el tipo de índice es btree: árbol B



Ejemplo

- Create index nombre_index
 - On esquema.tabla
 - Using btree (campo);
- Al crear el índice se puede indicar el tablespace, por defecto utiliza: **pg_default** y no es necesario indicarlo.

Borrar índices

- Drop index
- [concurrently] → sin bloquear otras consultas sobre la tabla. El servidor espera a que completen las TX en conflicto.
- [if exists] nombresesquema.nombre_índice
- [cascade | restrict] → borra objeto relacionado | evita que se borren objetos relacionados.

Listar los índices

- Disponemos de la vista: **pg_indexes** para listar los índices.
- Podemos listar todos los índices de un schema, utilizar el campo **schemaname** de la vista.
- Campos de la vista:
 - Schemaspace
 - Tablename
 - Indexname
 - Tablespace
 - Indexdef: El comando de la definición del index.

Tipos de índices. Sintaxis completa

```
CREATE [ UNIQUE ] INDEX [ CONCURRENTLY ] [ [ IF NOT EXISTS ] nmIndice ]  
  ON [ ONLY ] nmTabla [ USING metodo ]  
    ( ( column_name | ( expresion ) )  
      [ COLLATE colacion ]  
      [ opclass [ ( param_opclass = valor [, ... ] ) ] ]  
      [ ASC | DESC ]  
      [ NULLS ( FIRST | LAST ) ]  
      [, ... ]  
    )  
  [ INCLUDE ( nmColumna [, ...] ) ]  
  [ WITH ( param_almacenamiento [= valor] [, ... ] ) ]  
  [ TABLESPACE nmTableSpace ]  
  [ WHERE predicado ]
```


Tipos de índices

- La cláusula **using método** permite definir el tipo de índice.
- Btree: por defecto → árbol B
- Hash
- GiST
- SP-GiST
- GIN
- BRIN

Tipos de índices - Btree y los operadores para elegir este índice

B-tree es un árbol de autoequilibrio que mantiene los datos ordenados y permite búsquedas, inserciones, eliminaciones y acceso secuencial en tiempo logarítmico.

Operadores:

<

<=

=

>=

BETWEEN

IN

IS NULL

IS NOT NULL

Btree

Además, el planificador de consultas puede usar un índice **B-tree** para consultas que involucren un operador de coincidencia de patrones LIKE y ~ si el patrón es una constante y está anclado al principio del patrón.

Ejemplo
nombreCol like 'cadena%'
nombreCol ~ '^cadena'

Indice Hash

- Los índices Hash solo pueden manejar solo una comparación de igualdad simple (=)
- Create index nombre_index
- On nombre_tabla using '**HASH**' (nombre_col ,...)

Indice GiST

- GiST significa “generalized Search Tree” (árbol de búsqueda generalizado).
- Los índices GiST permiten la construcción de estructuras de árbol generales.
- Create index nombre_index
- On nombre_tabla using '**GI**ST' (nombre_col ,...)
- Sirven para indexar **tipos de datos geométricos**

Indice SP-GiST

- SP-Gist significa “space - partitioned Gist”, Gist con particiones espaciales. Admite árboles de búsqueda particionados que facilitan el desarrollo de una amplia gama de diferentes estructuras de datos no equilibradas.
- Create index nombre_index
- On nombre_tabla using **SPGIST** (nombre_col ,...)

Indice GIN

- GIN significa “Generalized Inverted Indexes” (índices invertidos generalizados).
- Create index nombre_index
- On nombre_tabla using '**GIN**' (nombre_col ,...)
- Son útiles cuando tenemos varios valores almacenados en una sola columna. Por ejemplo: **un array, jsonb**, etc.
- Permite los operadores: <@ @> = &&
- **Ojo el índice tiene un crecimiento muy rápido.**

Indice BRIN

- BRIN significa “Block Ranges Indexes” (índices de rango de bloque)
- BRIN es mucho más pequeño y menos costoso de mantener en comparación con un índice B-tree.
- Create index nombre_index
- On nombre_tabla using '**BRIN**' (nombre_col ,...)
- Permite el uso de índice en una tabla muy grande.
- Por ejemplo, se puede aplicar en una columna que sea de tipo fecha, pero que sea lineal, por ejemplo: **la fecha de creación de un registro.**

Indices únicos

- Obliga a que los valores utilizados para el índice ya sea una columna o varias sean únicos en toda la tabla.
- Create **unique** index nombre_índice on nombre_tabla (col_name ...)
- El único tipo de índice que se puede declarar como **Unique** es **B-TREE**
- PostgreSQL trata **NULL** como un valor distintivo, por lo tanto, puede tener **múltiples valores NULL en una columna con un índice UNICO.**

Indices en expresiones

- Son índices basados en una expresión que involucre columnas de una tabla
- Create index nombre_índice on nombre_tabla (**expresión**)

Indices en expresiones


Una vez que defina una expresión de índice, PostgreSQL considerará usar ese índice cuando la expresión que define el índice aparezca en la cláusula WHERE o en la cláusula ORDER BY de la declaración SQL.

Tenga en cuenta que los índices de expresiones son bastante costosos de mantener porque PostgreSQL tiene que evaluar la expresión para cada fila cuando se inserta o actualiza y usa el resultado para indexar.

Indice de expresiones

- Podemos utilizar **explain** de postgresql para ver si está utilizando o no los índices cuando realiza una consulta.

```
explain
select * from productos where descripcion = 'Cuna para Bebe';
```

	QUERY PLAN text	
1	Bitmap Heap Scan on productos (cost=4.18..12.64 rows=4 width=80)	
2	Recheck Cond: ((descripcion)::text = 'Cuna para Bebe'::text)	
3	-> Bitmap Index Scan on idxproductos (cost=0.00..4.18 rows=4 width=0)	
4	Index Cond: ((descripcion)::text = 'Cuna para Bebe'::text)	

Indice de expresiones

- Si añadimos una expresión y utilizamos **explain** veremos que se va por el modo secuencial para buscar los datos.

```
explain
select * from productos where lower(descripcion) = 'cuna para bebe';
```

Output Messages Notifications



QUERY PLAN

text



Seq Scan on productos (cost=0.00..21.25 rows=4 width=...

Filter: (lower((descripcion)::text) = 'cuna para bebe':te...

Indice de expresiones

- Definir el índice para una expresión:
- `Create index idx2productos on productos(lower(descripción));`
- Si comprobamos con explain y la expresión lower, continúa haciendo una búsqueda secuencial.
- Hay que desactivar una directiva para que no utilice las búsquedas secuenciales.
- **`Set enable_seqscan=false;`**

QUERY PLAN
text
Index Scan using idx2productos on productos (cost=0.14..8.15 rows=1 width=...
Index Cond: (lower((descripcion)::text) = 'cuna para bebe'::text)

Ya utiliza el nuevo índice.

Indices parciales

- Permiten indicar que filas se añadirán al índice:
- `Create index nombre_index on nombre_tabla (columnas) where condición;`
- Hacer pruebas comparando el query plan cuando se utiliza en la query el mismo filtro que se puso en el índice parcial u otra distinta.

Reindexación

- En la práctica un índice puede corromperse y ya no contiene datos válidos debido a fallas de HW o errores de SW.
- **Reindex** [(verbose)] { index | table | schema | database | system } nombre;
- Verbose: nos muestra el proceso de reindexación cuando lo realiza.

Reindexación

- `Reindex index nombre_index`
 - La forma más habitual para reindexar un índice.
- `Reindex table nombre_tabla`
 - Para reindexar todos los índices de la tabla
- `Reindex schema nombre_esquema;`
 - Reindexa todos los índices de un esquema
- `Reindex database nombre_bd;`
 - Para reindexar todos los índices de una BD
- `Reindex system nombre_bd`
 - Todos los índices del catálogo.

Indexación multicolumna

- Se pueden utilizar múltiples columnas para la creación de índices.
- Create index nombre_índice on tabla using '**método**' (nombre de las cols)
 - Mínimo tenemos que tener 2 columnas, si no, no es multi-columna.
 - Máximo 32 columnas en una sola tabla
 - Sólo se puede aplicar a tipos de índice B-tree, GIST, GIN y BRIN
 - No se pueden definir índices multicolumna de tipo: HASH, SP-GIST

Indexación multicolumna

- Cuando creamos el índice multicolumna importa el orden en que se colocan las columnas del índice.
- Primero hay que colocar las columnas que más se utilizan y al final las que menos se utilizan.
- A la hora de lanzar las consultas es importante el orden en el que se colocan los campos en la cláusula where con respecto a la posición que colocamos los campos al crear el índice.