Optimización

Administración de Bases de Datos

Departamento de Lenguajes y Sistemas Informáticos

eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

Contenido

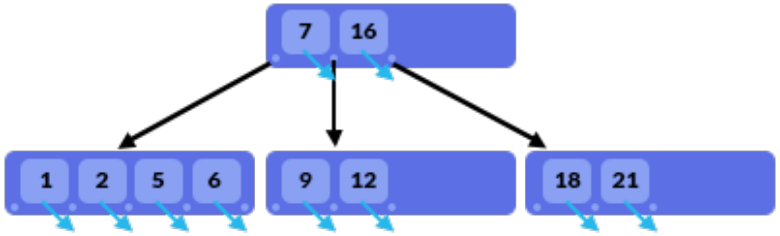
1. Índices
2. Procesamiento de consultas
3. Estimación de costes

Índice

- Estructura de datos que acelera la obtención de registros
- Se utiliza en ciertas condiciones de búsqueda
- Se puede construir sobre 1 o varios campos de una tabla
- Pueden ser de distintos tipos
  - B-Tree, Hash, GiST, ...
  - La variedad depende del SGBD

Índices

- El tipo más común es el B-Tree[1]:
  - Ordena los valores de una columna en forma de árbol:



  - Evita tener que recorrer todos los valores de una columna.
    - Esta operación se conoce como “full table scan”.
  - Funciona adecuadamente para la mayoría de tipos de datos.

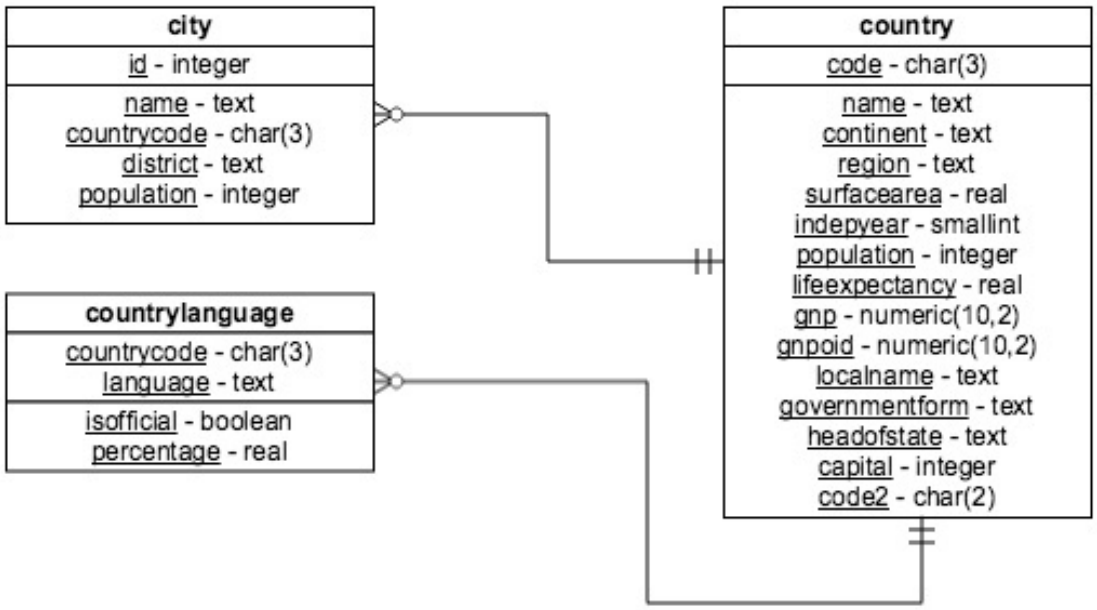
[1]Imagen: https://www.vertabelo.com/blog/all-about-indexes-part-2-mysql-index-structure-and-performance

Índices

- Usar índices tiene desventajas:
- Los índices ocupan espacio.
  - Se guardan en ficheros de disco.
  - Cuantos más datos en una columna indexada, mayor el índice.
- Retrasa las operaciones insert/update/delete.
  - El índice tiene que actualizarse cuando hay cambios en la tabla.
- Puede que no se utilice.
  - Algunas consultas serán más rápidas sin usar índices.

Índices

- Base de datos de ejemplo “World”:
  - Descarga: https://downloads.mysql.com/docs/world-db.zip
  - Esquema[1]:



[1]Obtenido de : https://gist.github.com/mdang/9a4a8063ebea3b829b8025746643ade1

Índices

- Crear un índice:

```
ALTER TABLE <tabla> ADD INDEX <nombre> (columna)
```

  - Donde:
    - ‹tabla›: Tabla donde se crea el índice
    - ‹nombre›: Nombre del índice creado
    - ‹columna›: Columna a la que se le asigna el índice

  - Ejemplo:

```
ALTER TABLE ciudades ADD INDEX idx_poblacion (poblacion)
```

Índices

- Consultar índices creados:
  - Para una tabla concreta:

```
SHOW INDEX FROM <tabla>
```

    - donde ‹tabla› es la tabla a consultar.

  - Para todas las tablas de una BD:

```
select table_name, index_name, index_type from
INFORMATION_SCHEMA.STATISTICS WHERE TABLE_SCHEMA = "<nombre-BD>"
```

    - donde ‹nombre-BD› es la BD a consultar.

Índices

- Comprobar el espacio utilizado por los índices:

```
show table status from <nombre-bd>
```

  - donde ‹nombre-db› es el nombre de la BD.
  - La columnas muestran:
    - Data_length: Espacio utilizado por cada tabla, en bytes.
    - Index_length: Espacio utilizado por los índices en cada tabla, en bytes.

- Borrar un índice:

```
ALTER TABLE <tabla> DROP INDEX <nombre>
```

  - donde:
    - ‹tabla›: Tabla objetivo.
    - ‹nombre›: Nombre del índice.

Índices

- Medir el tiempo de las consultas con más precisión:
  - Activar el profiling de consultas:

```
set profiling=1;
```

  - Mostrar el tiempo de las consultas registradas:

```
show profiles;
```

    - Por defecto, muestra las 15 últimas como máximo.

  - Más información:
    - https://dev.mysql.com/doc/refman/8.0/en/show-profile.html

Ejercicio 1

- *Utilizando la BD "World":*

- Crear una consulta que obtenga el nombre de los países en los que se hable inglés por un 50% de la población al menos.
  - Mostrar el nombre del país junto con el porcentaje de hablantes.

- Crear un índice en la tabla "countrylanguage" y comparar el tiempo de ejecución de la consulta con/sin índice.
  - Elegir la(s) columna(s) más apropiadas.

Índices B-Tree

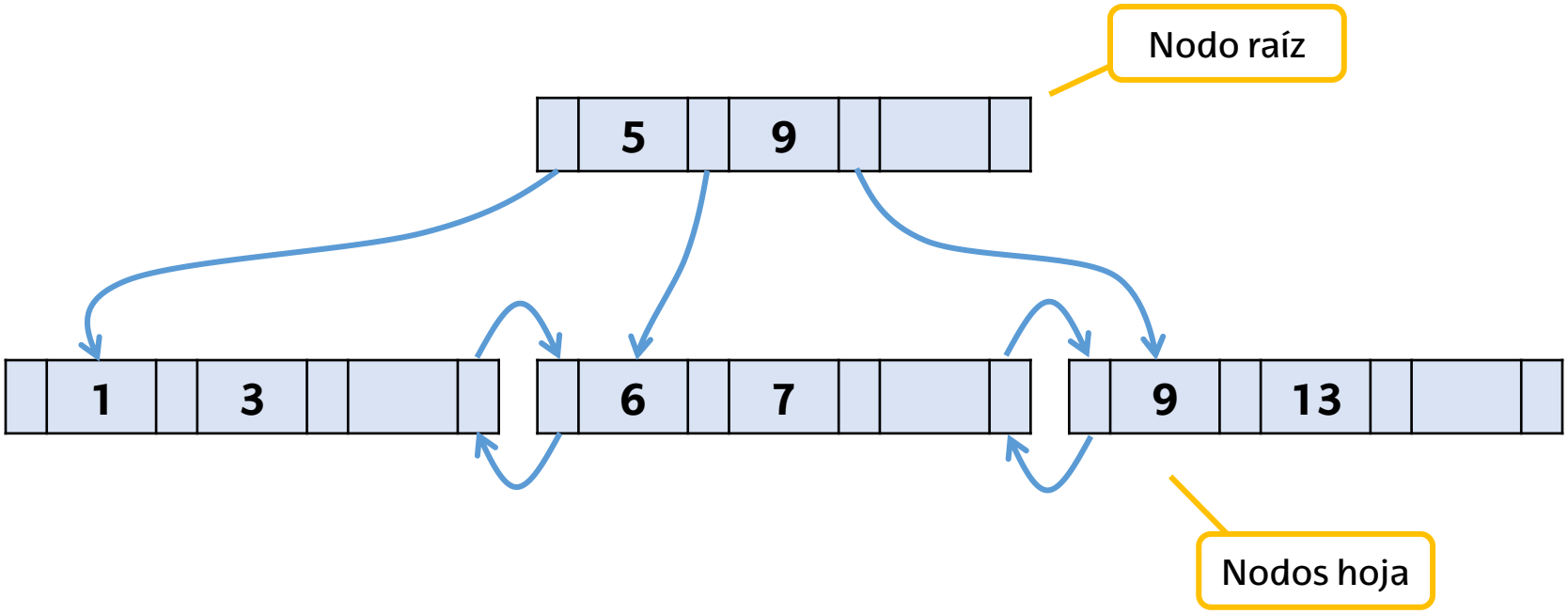
- B-Tree se refiere a una estructura de datos concreta.
- B-Tree también se usa como referencia a una familia de estructuras de datos:
  - B-Tree (1971)
  - B+Tree (1973)
    - La más utilizada
  - B*Tree (~1977)
  - B^{link}-Tree (1981)

Índices B+Tree

- Estructura de datos ordenada que permite búsquedas, acceso secuencial, inserciones y borrado en $O(\log n)$.
- Generalización del árbol de búsqueda binario.
- Se considera una estructura de M-caminos, con las siguientes propiedades:
  - Balanceado (todos los nodos hoja están a la misma profundidad).
  - Todos los nodos (excepto la raíz) tienen $M/2-1$ a $M-1$ elementos.
  - Todo nodo intermedio con k elementos tiene $k+1$ hijos no nulos.

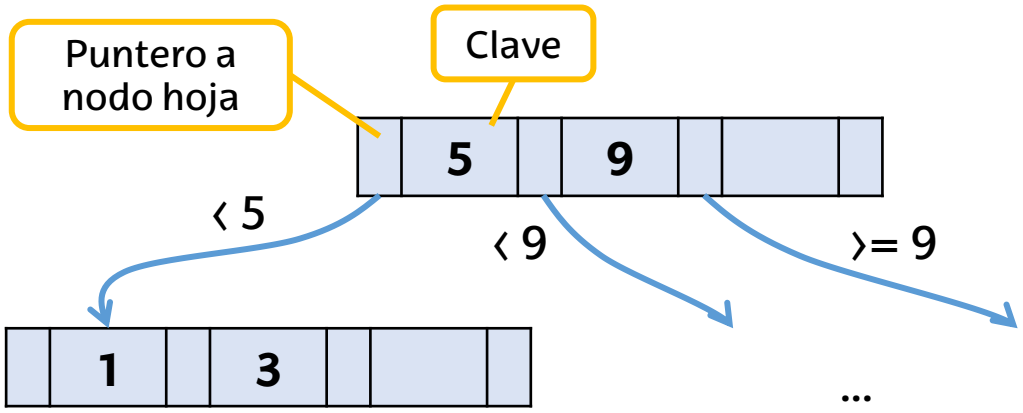
Índices B+Tree

- Ejemplo:



Índices B+Tree

- Ejemplo:
  - Mostrando sólo 1 de los nodos hoja:



  - En los nodos hoja, por cada clave, hay un puntero a la tupla correspondiente a la entrada de índice.
    - En algunos SGBDs, se almacena la tupla en lugar del puntero.

Índices B+Tree

- Algoritmo para insertar un nuevo elemento E.

1) Encontrar el nodo hoja H que le corresponda.
2) Insertar E en H.
3) ¿H tiene suficiente espacio para contener E?
   - Si: Fin.
   - No: Dividir las claves de H entre H y un nuevo nodo hoja H2.
     - Distribuir las claves de forma uniforme.
     - Insertar elemento apuntando a H2 en el padre de H.

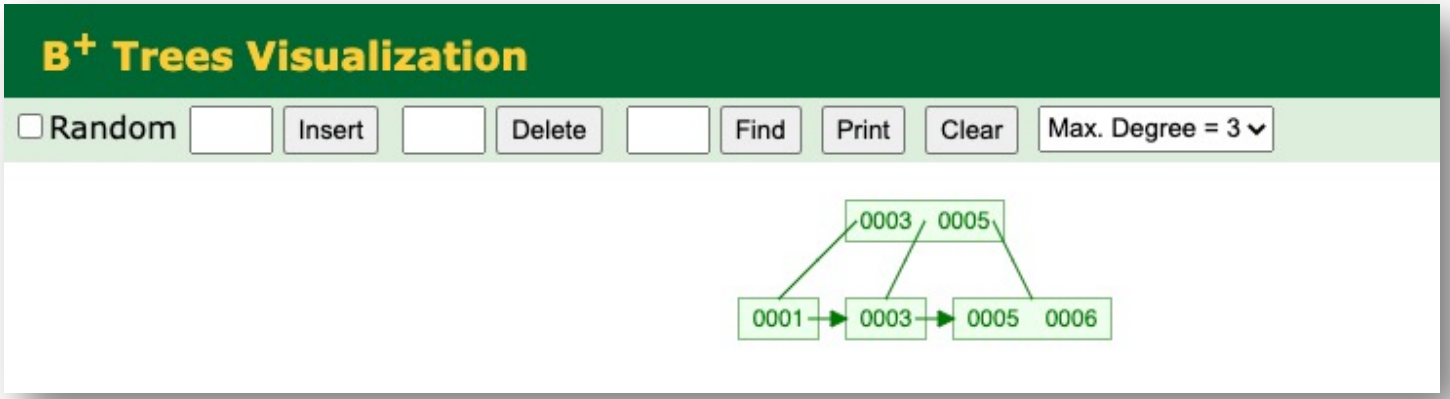
Índices B+Tree

- Algoritmo para eliminar un elemento E.

1) Encontrar el nodo hoja H que contiene E.
2) Eliminar E.
3) ¿H contiene más de M/2 - 1 elementos?
   - Si: Fin.
   - No: Re-distribuir elementos, cogiendo del nodo hoja cercano.
     - Si no es posible re-distribuir, fusionar H con el nodo hoja cercano.

Índices B+Tree

- Visualizador:
  - URL: https://cmudb.io/btree
  - Autor: David Galles, Universidad de San Francisco
  - El valor *Max. Degree* representa el nº máximo de elementos que puede contener un nodo + 1.



Procesamiento de consultas

- No es recomendable añadir índices sin entender su impacto en el SGBD.

- SQL es un lenguaje declarativo.
  - Una consulta define **qué** datos recuperar/modificar.
  - No define **cómo** se deben recuperar/modificar.

Procesamiento de consultas

- Cuando emitimos una consulta a un SGBD, esta pasa por diferentes fases:

1) Análisis gramatical (“parse”)
2) Optimización
3) Ejecución

    - El SGBD elige la estrategia de ejecución más optima.

Procesamiento de consultas

- El análisis gramatical divide la consulta en palabras y verifica que:
  - La sintaxis SQL es correcta.
  - Hace referencia a objetos existentes en la BD/SGBD.

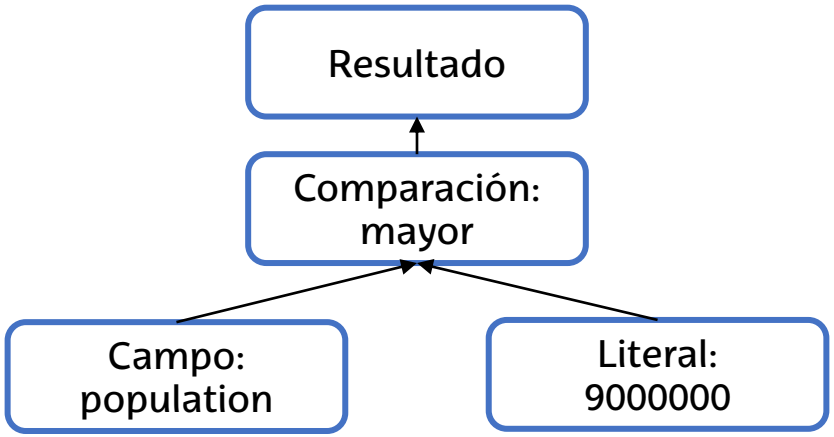
- Si la consulta es correcta, se crea un árbol sintáctico.
  - Representa los objetos y operaciones.

Procesamiento de consultas

- Ejemplo:
  - Consulta para la BD “World”:

```
select * from city where population > 9000000;
```

  - El analizador gramatical:
    1) Comprueba que la sintaxis y objetos usados son correctos.
    2) Crea árbol sintáctico:



Procesamiento de consultas

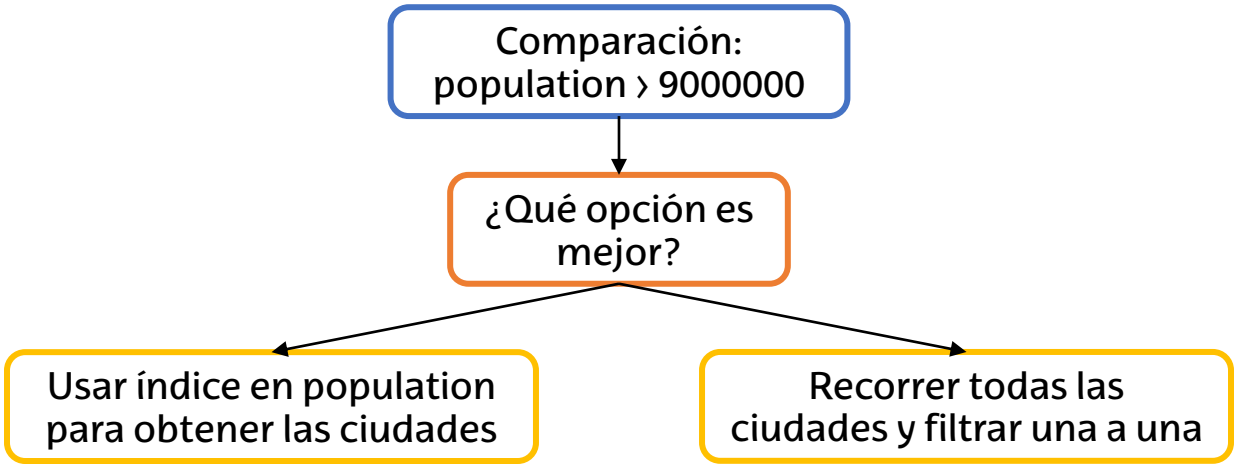
- Después, el optimizador usa el árbol sintáctico para:
  - Determinar qué operaciones Join se deben hacer.
  - Determinar si es mejor usar índices o “table scan”.
  - Determinar el orden de las operaciones.
  - …
- El resultado del optimizador es un plan de ejecución.
  - Secuencia de operaciones a realizar.

Procesamiento de consultas

- Ejemplo:
  - Consulta para la BD “World”:

```
select * from city where population > 9000000;
```

  - Se asume un índice en el campo “population”.
  - El optimizador plantea:



Procesamiento de consultas

- El optimizador estima el coste de cada alternativa para decidir qué ejecutar.

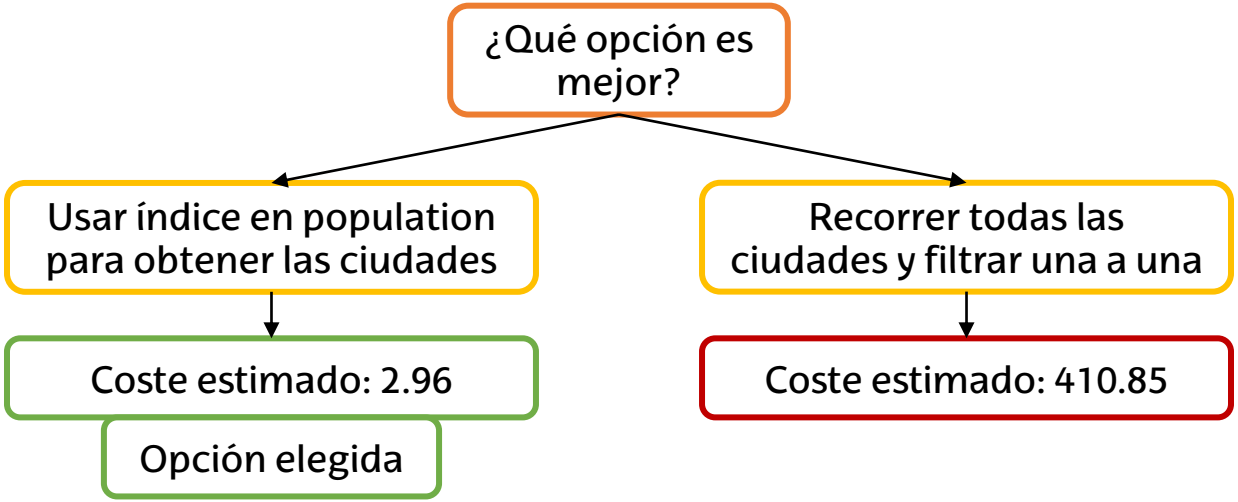
- El coste de una operación es un valor que sirve para comparar la complejidad entre diferentes planes.
  - No tiene unidad.
  - Se usa crear el plan de ejecución.

Procesamiento de consultas

- Ejemplo:
  - Consulta para la BD “World”:

```
select * from city where population > 9000000;
```

  - El optimizador calcula:



Procesamiento de consultas

- Podemos obtener información sobre las decisiones que toma el optimizador:
  - Mostrar el plan de ejecución para una consulta **sin** ejecutar la consulta.

```
explain <formato> <consulta>
```

    - donde:
      - ‹formato›: Indica el formato, p.e. “format=tree”. Opcional.
      - ‹consulta›: Consulta SQL a analizar.

  - Ejemplo:

```
explain select * from city where population > 9000000;
```

Procesamiento de consultas

- Podemos obtener información sobre las decisiones que toma el optimizador:
  - Mostrar el plan de ejecución **y** ejecutar una consulta:

```
explain analyze <consulta>
```

    - Ejemplo:

```
explain analyze select * from city where population > 9000000;
```

  - Este comando debe usarse sólo para evaluación de rendimiento, nunca en entornos de producción.

Procesamiento de consultas

- Cómo interpretar la salida de Explain Analyze
  - *Ejemplo para la consulta anterior:*

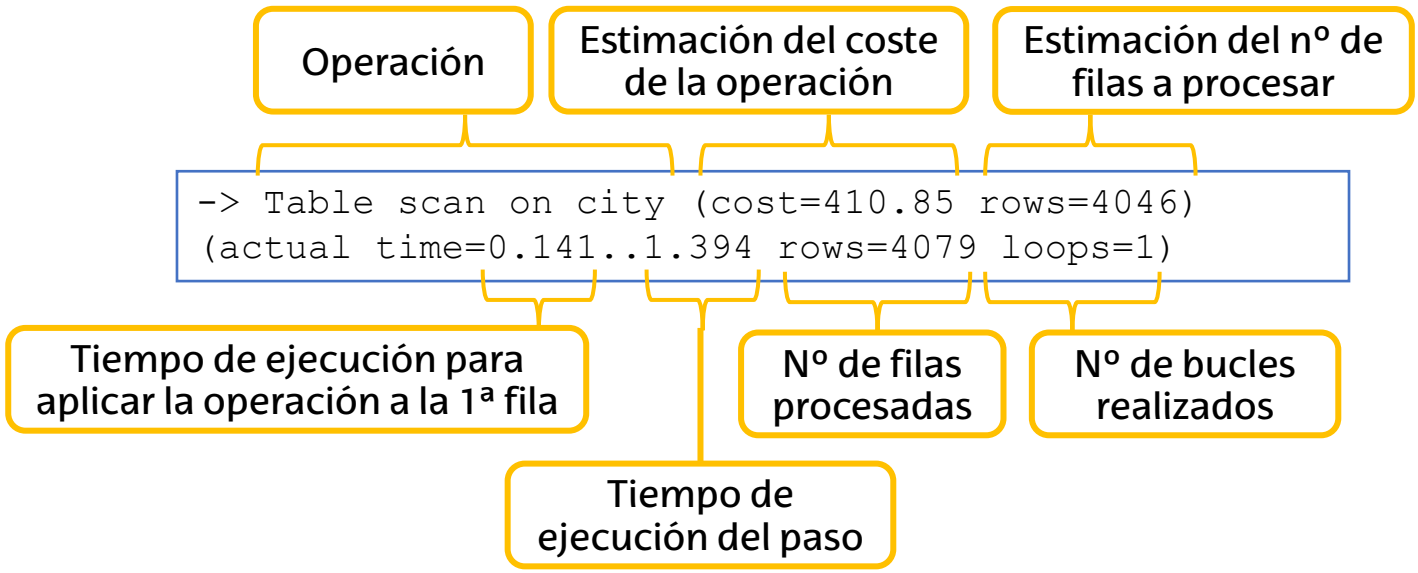
```
-> Filter: (city.Population > 9000000) (cost=410.85 rows=1349) (actual time=0.239..1.629 rows=6 loops=1)
  -> Table scan on city  (cost=410.85 rows=4046) (actual time=0.141..1.394 rows=4079 loops=1)
```

  - Cada línea con -› es un “nodo de consulta”
    - Paso en el que se ha accedido o procesado datos

  - El primer paso es el más interior en el árbol.
  - Cada paso se ejecuta y pasa su resultado al paso más cercano hacia arriba.
  - El resultado final es el paso más exterior y mas arriba en el árbol.

Procesamiento de consultas

- Cómo interpretar la salida de Explain Analyze
  - *Ejemplo para un paso de la consulta anterior:*



Ejercicio 2

- *Utilizando la BD "World":*

- Crear las siguientes consultas:
  1) Obtener el nombre de los países en los que se hable inglés por un 50% de la población al menos.
     - Mostrar el nombre del país junto con el porcentaje de hablantes.
  2) Mostrar los idiomas oficiales hablados en los países con esperanza de vida superior a 80 años.
  3) Mostrar el nombre y población de las 10 ciudades más pobladas que NO sean capitales de país.

- *Continúa en la siguiente diapositiva*

Ejercicio 2

- Completar la siguiente tabla:
  - Obtener el coste de las consultas sin crear índices adicionales.
  - Crear índices en las columnas que veáis convenientes.
  - Obtener el coste de las consultas con índices adicionales.

Consulta	Coste estimado	
	Sin índices	Con índices
Consulta 1		
Consulta 2		
Consulta 3		

  - ¿Qué consulta obtiene la mayor reducción de coste al usar un índice?

Estimación de costes

- El coste estimado de una operación es una aproximación de la complejidad de su cómputo.
  - Depende de los datos de entrada.
- Se basa en:
  - Estadísticas de las operaciones y datos involucrados en las consultas.
  - Valores heurísticos del SGBD.

Estimación de costes

- Los datos estadísticos necesarios para calcular costes se obtienen del diccionario de datos.
  - Tablas del SGBD con estadísticas que se actualizan tras cada operación.

- En MySQL es la BD llamada *Information_Schema*
  - P.e. consultar estadísticas sobre los índices en las tablas de "world"

```
select * from information_schema.statistics where table_schema="world";
```

  - "index_name" muestra el nombre de cada índice creado.
  - "cardinality" muestra un estimación del nº de valores únicos en el índice.
  - Más información: https://dev.mysql.com/doc/refman/8.0/en/information-schema-statistics-table.html

Estimación de costes

- En este tema, consideraremos 3 operaciones para estimar el coste de un plan.
  - Representadas como:

Operación	Equivalente SQL	Símbolo algebraico
Proyección	Select	π
Restricción	Where	σ
Fusión	Join	$\triangleright\triangleleft$

  - Utilizaremos una representación algebraica para describir las consultas de forma genérica.
  - En un SGBD real se tienen en cuenta más operaciones.

Estimación de costes

- En este tema, consideraremos 3 operaciones para estimar el coste de un plan.
  - Ejemplos de conversión a representación algebraica:

```
SELECT name FROM city
```

$$\pi_{\text{name}}(\text{city})$$

```
SELECT name FROM city
WHERE population > 9000000
```

$$\pi_{\text{name}}(\sigma_{\text{population}>9000000}(\text{city}))$$

```
SELECT ci.name, co.name
FROM city ci INNER JOIN country co
ON ci.CountryCode=co.Code;
```

$$\pi_{\text{ci.name, co.name}}(\text{city} \underset{\text{countryCode=Code}}{\triangleright\!\triangleleft} \text{country})$$

Estimación de costes

- En este tema, consideraremos 3 operaciones para estimar el coste de un plan.
  - Ejemplo de conversión a representación algebraica :

```
SELECT name
FROM Employee AS E, Work_in AS W
WHERE E.eid=W.weid and W.wid=2;
```

  - Representación algebraica:

$$\pi_{name}\left(\sigma_{wid=2}\left(Employee \underset{eid=weid}{\triangleright\!\triangleleft} Work\_in\right)\right)$$

Proyección (Select)

Restricción (Where)

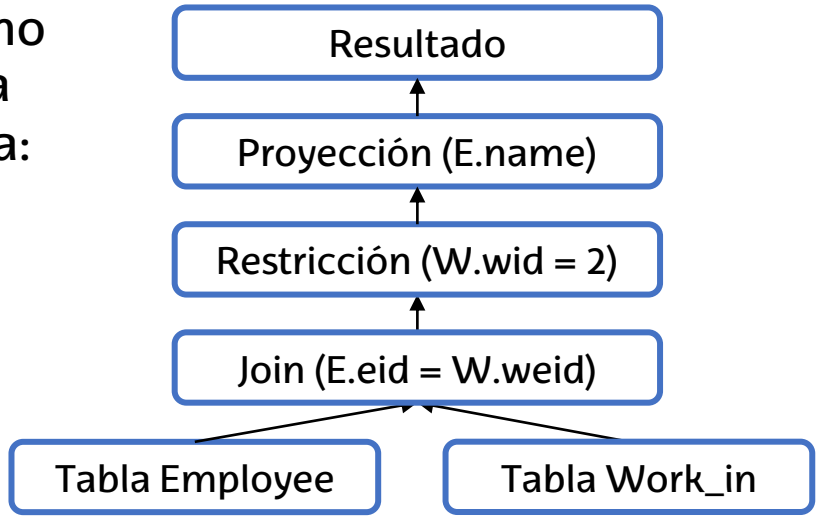
Fusión (Join)

Estimación de costes

- La representación algebraica se utiliza para construir el árbol sintáctico.
  - Ejemplo:

$$\pi_{name}\left(\sigma_{wid=2}\left(EMPLOYEE \underset{eid=weid}{\triangleright\triangleleft} WORK\_IN\right)\right)$$

  - Árbol sintáctico como representación de la expresión algebraica:



Ejercicio 3

- Obtener la representación algebraica de las siguientes consultas:
  - Parte 1/2

a)

```
SELECT ci.name, ci.population, co.name
FROM city ci INNER JOIN country co
ON ci.CountryCode=co.Code
WHERE co.Name="France";
```

b)

```
SELECT A.a2, B.b1, B.b2, C.c2
FROM A INNER JOIN B INNER JOIN (SELECT c1, c2 FROM C WHERE c3 > 0) AS C
ON A.a1 = B.b1 AND B.b1 = C.c1
WHERE A.a3 > 0 AND B.b2 > C.c2
```

Ejercicio 3

- Obtener la representación algebraica de las siguientes consultas:
  - Parte 2/2

c)

```
SELECT S.id, R.course, R.student, C.code, C.name
FROM Student as S INNER JOIN Registration as R ON S.id = R.Student
INNER JOIN Course as C on R.course=C.code
WHERE S.level = 3 and R.student > C.coordinator
```

d)

```
SELECT T.id, T.first_name, C.code, C.coordinator, R.course, R.year
FROM Teacher as T INNER JOIN Course as C INNER JOIN
  (SELECT course, year FROM Registration WHERE year<2010) as R
ON T.id=C.coordinator AND C.code = R.course
WHERE R.mark = 5 and T.first_name like 'A%'
```

Estimación de costes

- Una misma consulta puede tener múltiples planes de ejecución.
  - El optimizador ejecutará el plan con el menor coste.

- El coste de un plan será la suma de los costes sus operaciones.

$$cost(plan) = \sum cost(operation_i)$$

- Consideraremos el tamaño resultante de cada operación.
  - Nº de tuplas resultantes.
  - Se representará como: *size(operation)*

Estimación de costes: operaciones

- Operación Restricción (Where)

$$\sigma_c(A)$$

  - donde:
    - A: Tabla objetivo
    - C: Condición

- 2 técnicas posibles para su implementación:
  - 1: Búsqueda lineal:
    - No existe índice en el atributo o no se utiliza.

$$\text{cost}(\sigma_c(A)) = \text{size}(A)$$

    - donde size(A) es el tamaño (nº de tuplas) de A.

Estimación de costes: operaciones

- Operación Restricción (Where)

$$\sigma_c(A)$$

  - donde:
    - A: Tabla objetivo
    - C: Condición

- 2 técnicas posibles para su implementación:
  - 2: Búsqueda por índice:
    - Se utiliza un índice para buscar en la columna.

$$\text{cost}(\sigma_c(A)) = t * \log_2(\text{size}(A))$$

    - donde **t** es el promedio de veces que se repite un valor en la columna.
      - Este valor se obtiene del diccionario de datos.

Estimación de costes: operaciones

- Operación Restricción (Where)

$$\sigma_c(A)$$

  - donde:
    - A: Tabla objetivo
    - C: Condición

- Tamaño:
  - Se cumple que: $\text{size}(\sigma_c(A)) \leq \text{size}(A)$

- La operación $\sigma_c(A)$ hereda todos los índices de A

Estimación de costes: operaciones

- Operación Proyección (Select)

$$\pi_p(A)$$

  - Donde:
    - A: tabla objetivo
    - p: Atributos a proyectar

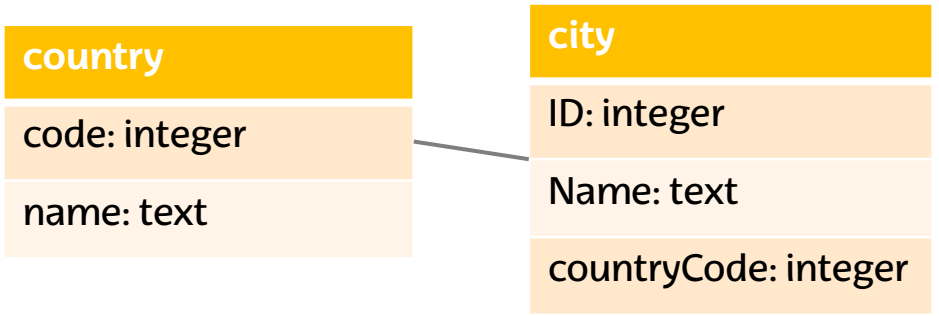
- Coste: $\text{cost}(\pi_p(A)) = \text{size}(A)$

- Tamaño: $\text{size}(\pi_p(A)) = \text{size}(A)$

- La operación $\pi_p(A)$ hereda todos los índices de A.

Estimación de costes: operaciones

- Ejercicio de ejemplo:
  - Esquema:



  - Diccionario de datos:
    - Hay 20 países
    - Hay 40 ciudades
    - No hay ningún índice creado
  - Consultas:
    a) $\pi_{name}(country)$
    b) $\pi_{ID,name}(\sigma_{name="Washington"}(city))$

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\!\triangleleft} S$$

  - En SQL hay diferentes tipos de Join:
    - Considerando las siguientes 2 tablas de ejemplo[1]:

User Table – Table 1

ID (Primary Key)	Name	Address
1	Sally Select	123 Join Dr
2	Frank From	25 Where St

Event Table – Table 2

User_ID (Foreign Key)	ID (Primary Key)	Action
1	A	LOGIN
3	B	VIEW PAGE
4	C	LOGIN

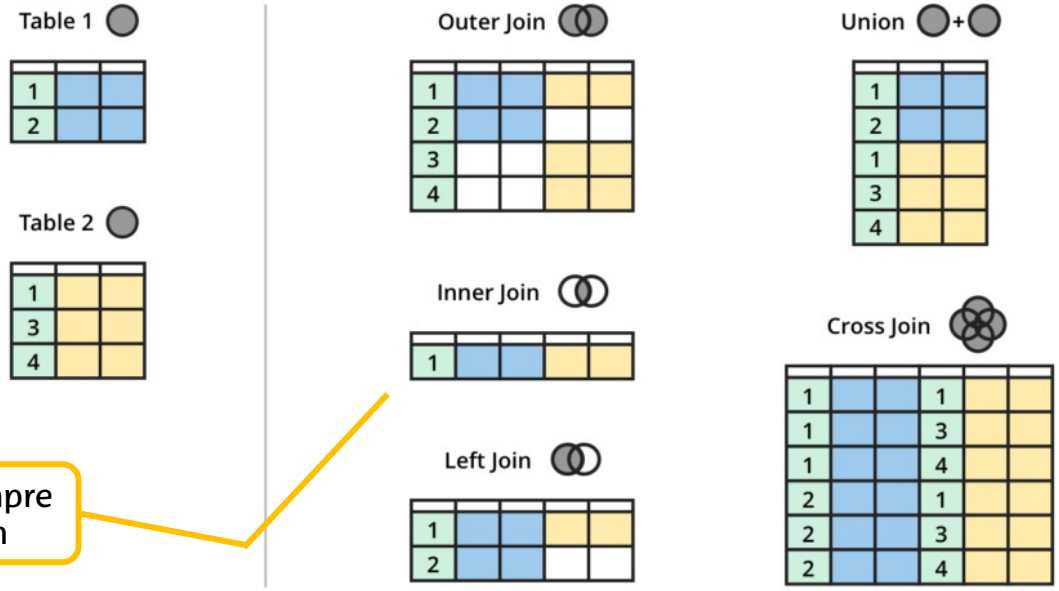
[1]Imagen: https://dataschool.com/how-to-teach-people-sql/sql-join-types-explained-visually/

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\!\triangleleft} S$$

  - En SQL hay diferentes tipos de Join:
    - Considerando las 2 tablas de ejemplo[1]:



En este tema siempre serán Inner Join

[1]Imagen: https://dataschool.com/how-to-teach-people-sql/sql-join-types-explained-visually/

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\triangleleft} S$$

  - donde:
    - R y S: Tablas a fusionar
    - r y s: Atributos de R y S utilizados para indicar la fusión.

- 3 técnicas posibles para su implementación
  - 1: Fuerza bruta
    - No se usan índices. Se examinan todas las combinaciones posibles entre las tuplas de R y S.

$$cost(R \triangleright\triangleleft S) = size(R) * size(S)$$

    - $R \triangleright\triangleleft S$ hereda todos los índices de R.

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\triangleleft} S$$

  - donde:
    - R y S: Tablas a fusionar
    - r y s: Atributos de R y S utilizados para indicar la fusión.

- 3 técnicas posibles para su implementación
  - 2: Búsqueda por índice
    - Existe un índice sobre el atributo s.

$$cost(R \triangleright\triangleleft S) = size(R) * t * \log_2(size(S))$$

    - R $\triangleright\triangleleft$ S hereda todos los índices de R.

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\triangleleft} S$$

  - donde:
    - R y S: Tablas a fusionar
    - r y s: Atributos de R y S utilizados para indicar la fusión.

- 3 técnicas posibles para su implementación
  - 3: Búsqueda por mezcla
    - Existe un índice sobre el atributo r y otro índice sobre s.

$$cost(R \triangleright\triangleleft S) = size(R) + size(S)$$

    - $R \triangleright\triangleleft S$ hereda los índices sobre los atributos r y s.

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\triangleleft} S$$

  - donde:
    - R y S: Tablas a fusionar
    - r y s: Atributos de R y S utilizados para indicar la fusión.

- Tamaño:

$$size(R \triangleright\triangleleft S) \leq size(R) * size(S)$$

Estimación de costes: operaciones

- Operación Fusión (Join)

$$R \underset{r=s}{\triangleright\!\triangleleft} S$$

  - donde:
    - R y S: Tablas a fusionar
    - r y s: Atributos de R y S utilizados para indicar la fusión.

- Propiedades:
  - Asociatividad: $R \triangleright\!\triangleleft (S \triangleright\!\triangleleft T) = (R \triangleright\!\triangleleft S) \triangleright\!\triangleleft T$
  - Semi-conmutatividad: $R \triangleright\!\triangleleft S \simeq S \triangleright\!\triangleleft R$

Estimación de costes: operaciones

- Como parte de la optimización de un plan, se pueden ordenar los valores de una relación R.
- No implica la creación permanente de un índice.
  - Los valores ordenados se destruyen tras ejecutar el plan.
- El coste es:

$$sort(R) = size(R) * \log_2(size(R))$$

  - Este coste se suma al coste total del plan.

Estimación de costes: Resumen I

- Restricción
  - Coste de búsqueda lineal: $cost(\sigma_c(A)) = size(A)$
  - Coste de búsqueda por índice: $cost(\sigma_c(A)) = t * \log_2(size(A))$
  - Tamaño: $size(\sigma_c(A)) \leq size(A)$
- Proyección
  - Coste: $cost(\pi_p(A)) = size(A)$
  - Tamaño: $size(\pi_p(A)) = size(A)$

Estimación de costes: Resumen II

- Join
  - Coste de fuerza bruta: $cost(R \rhd\lhd S) = size(R) * size(S)$
  - Coste de búsqueda por índice: $cost(R \rhd\lhd S) = size(R) * t * \log_2(size(S))$
  - Coste de búsqueda por mezcla: $cost(R \rhd\lhd S) = size(R) + size(S)$
  - Tamaño: $size(R \rhd\lhd S) \leq size(R) * size(S)$
- Ordenación
  - Coste: $sort(R) = size(R) * \log_2(size(R))$

Bibliografía

- R. Elmasri, S. B. Navathe. “Fundamentals of Database Systems”, 7th edition, Pearson, 2017.
- S. Pachev. “Understanding MySQL Internals”, 1st edition, O’Reilly, 2007.
- S. Grider. “SQL and PostgreSQL: The Complete Developer's Guide”, Udemy, 2022.
- A. Pavlo. “Database Systems”, 2022
  - CMU 15-445: https://15445.courses.cs.cmu.edu/fall2022/