# TEMA 6 OPTIMIZACION

**Objetivos:**

* Comprender las tareas de procesamiento y optimización de consultas realizadas por un SGBD relacional
* Conocer reglas heurísticas y de transformaciones de expresiones del álgebra relacional y cómo aplicarlas para mejorar la eficiencia de una consulta
* Conocer diferentes estrategias de implementación de operaciones relacionales, en particular del join, y cómo evaluar sus costes
* Identificar la información estadística de la BD necesaria para estimar el coste de ejecución de las operaciones del álgebra relacional

# Procesamiento y optimización de consultas

* Los primeros sistemas relacionales ofrecían un bajo rendimiento en las consultas
* Este hecho promovió́ el estudio de algoritmos eficientes para procesar consultas
* SQL es un lenguaje declarativo:
  + Una consulta SQL define qué datos se tienen que recuperar
  + Sin embargo, una consulta SQL no indica cómo se han de recuperar los datos
* El SGBD es el encargado de seleccionar la mejor estrategia de ejecución:
  + El sistema decide las operaciones necesarias y su orden de ejecución
* Definición
  + Procesamiento de consultas:
    - Actividades involucradas en la recuperación de datos de la BD
  + Optimización de consultas:
    - Elección de una estrategia de ejecución eficaz para procesar cada consulta sobre la BD
    - La optimización automática es obligatoria si se quiere lograr un tiempo de consulta aceptable
* Objetivos
  + Objetivos del procesamiento de consultas:
    - Transformar una consulta SQL en una estrategia de ejecución eficaz expresada en un lenguaje de bajo nivel
      * Existen muchas transformaciones equivalentes
    - Ejecutar dicha estrategia para recuperar los datos requeridos
  + Objetivos de la optimización de consultas:
    - Elegir la estrategia de ejecución que minimiza el uso de los recursos
    - En general, el SGBD no garantiza que la estrategia elegida sea óptima, aunque seguro que es una bastante eficiente
* Ventajas de la optimización automática
  + El usuario no se preocupa de cómo formular la consulta
  + El módulo optimizador trabaja “mejor” que el programador:
    - Dispone de información estadística en el diccionario de datos
      * Mayor precisión al estimar la eficiencia de cada posible estrategia
      * Con mayor probabilidad elegirá́ la más eficiente
      * Si cambian las estadísticas (tras reorganización del esquema de la BD), puede elegir otra estrategia
    - Considera más estrategias que un programador
    - Contempla toda la experiencia sobre el tema

# Etapas del procesamiento de consultas

## Análisis léxico, sintáctico y validación semántica

### Análisis léxico

Identificar los componentes léxicos en el texto de la consulta SQL

### Análisis sintáctico

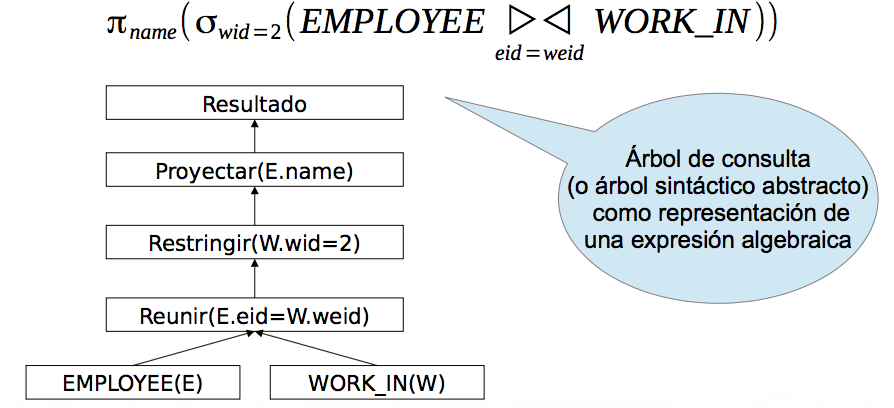
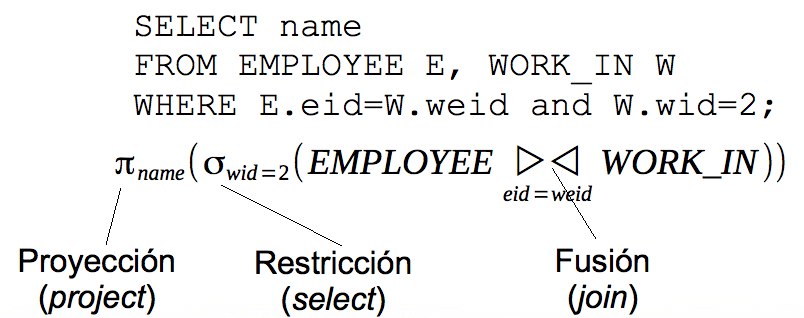
Revisar la corrección gramatical de la consulta SQL

### Validación semántica

Verificar la validez de los nombres de las relaciones y atributos

* Resultado:
  + Traducción de la consulta a una representación interna
    - El formalismo usado debe ser:
      * Rico, para representar toda consulta posible
      * Neutral, sin predisponer a ciertas opciones de optimización
  + Álgebra relacional
* Ejemplo:

Obtener los nombres de los empleados que trabajan en el proyecto de código 2



## Optimización

* El optimizador de consultas combina diferentes técnicas:
  + Optimización heurística
    - Ordenar las operaciones en una estrategia de ejecución
  + Estimación de costes
    - Estimar sistemáticamente el coste de cada estrategia
    - Elegir el plan (estrategia) con el menor coste estimado
* El analizador sintáctico genera un árbol de consulta inicial
  + Sin optimización y, por tanto, ineficiente
* El optimizador de consultas transforma el árbol de consulta inicial en un árbol de consulta final equivalente pero más eficiente
  + Mediante la aplicación de reglas de transformación guiadas por heurísticos
* La consulta se mantiene siempre en forma canónica
* El optimizador decide cómo evaluar la consulta en forma canónica
* Estimación sistemática de costes:
  + Estimación y comparación de los costes de ejecutar una consulta con diferentes estrategias, y elegir la de menor coste estimado
* El punto de partida es considerar la consulta como una serie de operaciones interdependientes

## Optimización heurística

* Aplicación de reglas heurísticas modificando la representación interna de una consulta (álgebra relacional o árbol de consulta) para mejorar su rendimiento
* Varias expresiones del álgebra relacional pueden corresponder a una misma consulta
* En SQL:
* Una misma consulta se puede expresar de distintas formas
* Pero el rendimiento no debería depender de cómo sea expresada la consulta

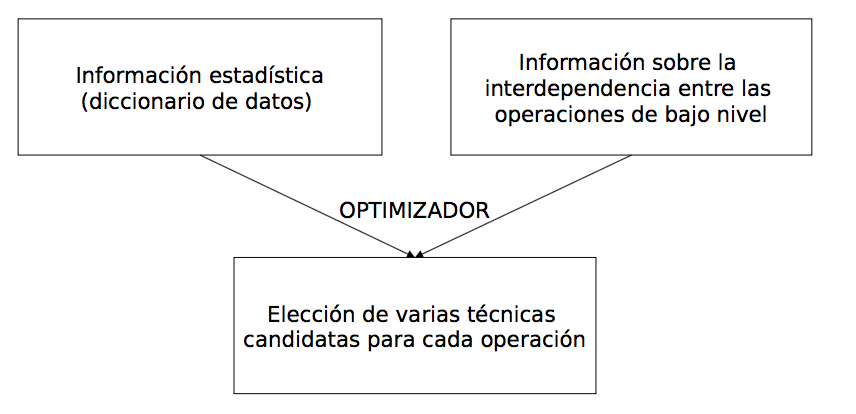
## Álgebra relacional

* Operaciones del álgebra relacional
  + JOIN, PROJECT, SELECT, UNION, INTERSECTION, SET DIFFERENCE, ...
* El optimizador tiene un conjunto de técnicas para implementar cada operación
  + Por ejemplo, la operación σ:
* Búsqueda lineal
* Búsqueda binaria
* Empleo de índice primario o clave de dispersión
* Empleo de índice de agrupamiento
* Empleo de índice secundario

## Estimación de coste

* Cada procedimiento tiene asociada una estimación del coste
  + Número de accesos a bloque de disco necesarios
  + Se debe estimar el tamaño actual de las tablas

## Esquema



## Información estadística

* El éxito de la estimación estadística del tamaño y coste de las operaciones incluidas en una consulta depende de la cantidad y actualidad de la información almacenada en el diccionario de datos
* Para cada relación:
  + Cardinalidad (número de tuplas)
  + Factor de bloque (número de tuplas que caben en cada bloque)
  + Número de bloques ocupados
  + Método de acceso primario y otras estructuras de acceso (índices, hash, etc.)
  + Atributos indexados, de dispersión, de ordenamiento (físicos o no), etc.
* Para cada atributo:
  + Número de valores distintos almacenados
  + Valores máximo y mínimo
* Número de niveles de los índices de múltiples niveles

## Plan de ejecución

* Es una combinación de técnicas candidatas, una por cada operación de bajo nivel de la consulta
* El optimizador genera varios planes de ejecución y elige el plan más económico
* Función de coste a minimizar
  + Medida en número de accesos a bloque o transferencias de bloques de memoria a disco

cost(plan)=Σi cost(techniquei)

* En general, existen muchos (demasiados) posibles planes de ejecución.
  + La tarea de elegir el plan más económico tiene un coste prohibitivo
  + Se utilizan técnicas heurísticas para mantener el conjunto de planes de consulta generados dentro de unos límites razonables:
    - Se pretende una reducción del espacio de evaluación

# Reglas de transformación

## Reglas generales

* Una secuencia de restricciones sobre una relación A puede transformarse en una sola restricción:

σc2(σc1(A))⇒σc1∧c2(A)

* Una secuencia de proyecciones sobre una relación A puede transformarse en la última, si cada atributo mencionado en la última también aparece en las demás:

πp2(πp1(A))⇒πp2(A) iff p2⊆p1

* Una restricción de una proyección puede transformarse en una proyección de una restricción:

σc(πp(A))⇒πp(σc(A))

Es mejor hacer restricción antes que proyección, pues la restricción reduce el número de filas a considerar

* SELECT es distributiva con respecto a UNION, INTERSECTION y SET DIFFERENCE:

σc(A∪B)⇒σc(A)∪σc(B)

σc(A∩B)⇒σc(A)∩σc(B)

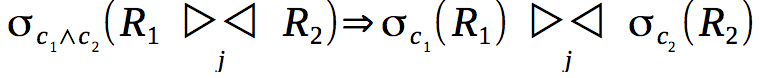
σc(A−B)⇒σc(A)−σc(B)

* PROYECT es distributiva con respecto a UNION:

πp(A∪B)⇒πp(A)∪πp(B)

* Distribución de SELECT con respecto a JOIN

Se cumple

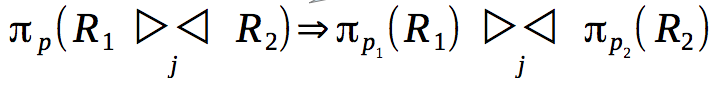


si en c1 (resp. c2) solo aparecen atributos de R1 (resp. R2)

Se reduce el número de tuplas a examinar en la siguiente operación

* Distribución de PROJECT con respecto a JOIN

Se cumple



sii todo atributo que aparece en j aparece bien en p1 o bien en p2, y se cumple que p = (p1 union p2)

También se reduce el nº de tuplas a examinar en la siguiente operación.

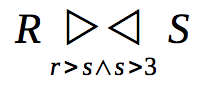
* Commutatividad: A op B = B op A
  + UNION, INTERSECTION y JOIN son commutativas
  + SET DIFFERENCE no es commutativa
* Asociatividad: A op (B op C) = (A op B) op C
  + UNION, INTERSECTION y JOIN son asociativas
  + SET DIFFERENCE no es asociativa
* Idempotencia: for all A: A op A = A
  + UNION, INTERSECTION y JOIN son idempotentes

## Expresiones de cómputo escalar

* El optimizador debe conocer reglas de transformación de expresiones aritméticas
* Reglas de transformación basadas en las propiedades conmutativa, asociativa y distributiva

## Expresiones condicionales

* El optimizador debe saber aplicar reglas generales a operadores:
  + De comparación (<,>, ...)
  + Lógicos (and, or, ...)
* Ejemplo:



La condición de JOIN r>s∧s>3 es equivalente a r>s∧s>3∧r>3

La condición r > 3 permite realizar una restricción de R antes del JOIN de R y S

## Forma nomal conjuntiva (CNF)

* Una fórmula (condición) c está en forma normal conjuntiva (CNF) sii
  + c=c1∧c2∧...∧cn
  + Cada ci no incluye ninguna conjunción
* La fórmula c es equivalente a true si todo ci es equivalente a true
* La fórmula c es equivalente a false si algún ci es equivalente a false
* Ventajas de la forma normal conjuntiva:
  + Como la conjunción es commutativa, el optimizador puede evaluar las subcondiciones en cualquier orden. Por ejemplo, en orden creciente de dificultad
    - En cuanto una subcondición es equivalente a false, el proceso concluye
  + Además, en un entorno de procesamiento paralelo, es posible evaluar todas las subcondiciones a la vez

## Reglas heurísticas

* Ejecutar las operaciones de restricción σ tan pronto como sea posible
* Ejecutar primero las restricciones más restrictivas (las que producen menor número de tuplas)
* Ejecutar las operaciones de proyección π tan pronto como sea posible

# Implementación de operaciones relaciones

## Operaciones:

## Macintosh HD:Users:aidagonzalezgonzalez:Desktop:Captura de pantalla 2018-05-21 a las 23.56.10.pngSELECT

* Técnicas para realizar una restricción
  + Búsqueda lineal:
  + Búsqueda por índice: si existe un índice en A para algún atributo usado en la condición c
* Coste asintótico:
  + Búsqueda lineal: cost(σc(A))=size(A)
* Búsqueda por índice: cost(σc(A))=t×log size(A)
* Tamaño:
  + Se calcula por estimación en base a la información del diccionario
  + Siempre se cumple que:

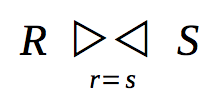
size(σc(A))≤size(A)

* σc(A)hereda todos los índices de A (\*)

(\*) Aunque puede suponer un aumento del coste de la39 operación

## Macintosh HD:Users:aidagonzalezgonzalez:Desktop:Captura de pantalla 2018-05-22 a las 0.07.26.pngPROJECT

* Técnicas para realizar una proyección
  + Búsqueda lineal
* Coste asintótico: cost(πp(A))=size(A)
* Tamaño: size(πp(A))=size(A)
* πp(A) hereda todos los índices de A



## JOIN

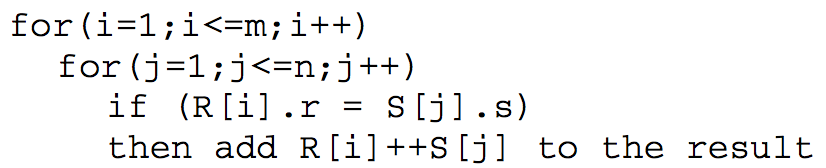
* Técnicas para realizar una fusión
  + Fuerza bruta
  + Búsqueda por índice
  + Mezcla (búsqueda doble)
* Tamaño:
  + Se calcula por estimación en base a:
    - Claves únicas y restricciones referenciales
    - Información del diccionario
  + Siempre se cumple que:

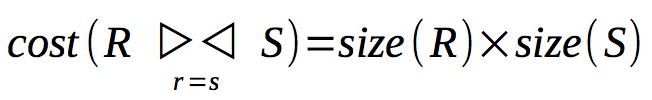
size(R ▷◁ S)≤size(R)×size(S)

r=s

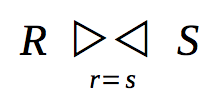
## Macintosh HD:Users:aidagonzalezgonzalez:Desktop:Captura de pantalla 2018-05-21 a las 23.30.40.pngJOIN por fuerza bruta

* Se examinan todas las combinaciones posibles de tuplas R y S:





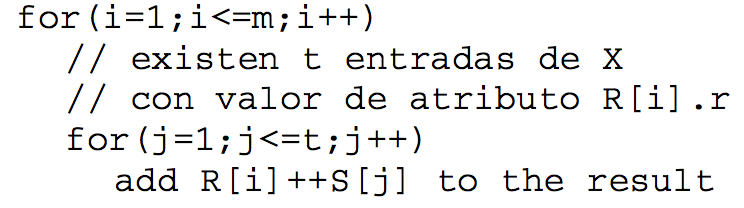
* Coste asintótico:

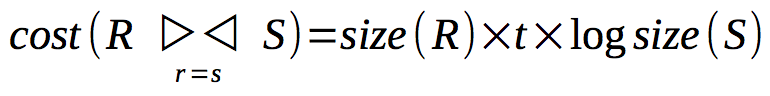


* hereda todos los índices de R

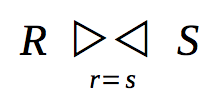
## Macintosh HD:Users:aidagonzalezgonzalez:Desktop:Captura de pantalla 2018-05-21 a las 23.30.40.pngJOIN por búsqueda por índice

* Existe un índice X sobre el atributo s:



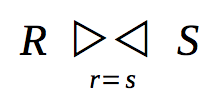
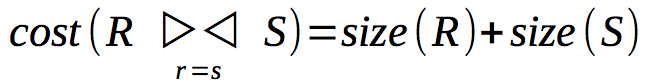
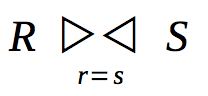
* Coste asintótico:

donde asumimos que size(S)≫t×log size(S)

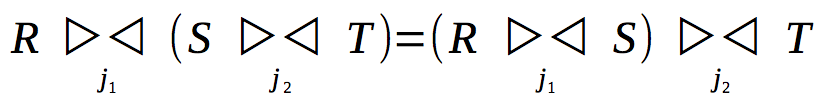


* hereda todos los índices de R

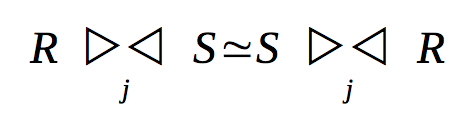
## JOIN por mezcla

* Si existe un índice para el atributo r y otro para s, entonces ambas relaciones pueden examinarse simultáneamente en una única pasada
* Coste asintótico:
*  hereda los índices sobre los atributos r y s

## Propiedades del JOIN



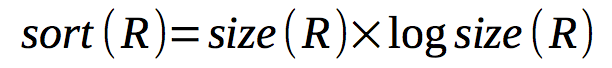
* Asociatividad:



* Semi-commutatividad:

## Ordenación

* El coste de ordenar una relación R según cualquier criterio X es:



* Se suma al coste asintótico de las operaciones sobre la relación R

## Planes

* Hay que considerar todas las opciones posibles
  + Incluyendo la ordenación de relaciones

