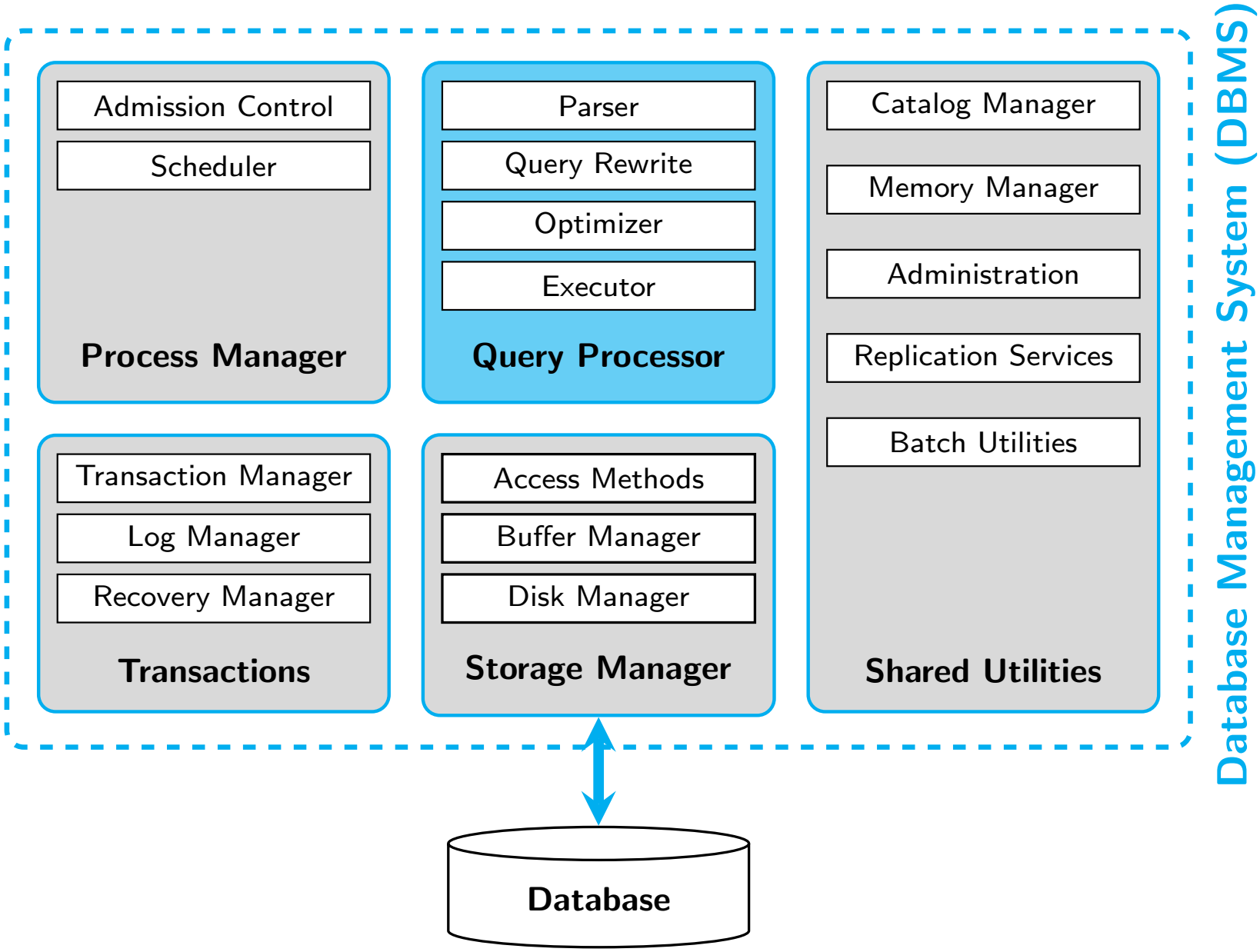
# Complejidad de consultas

Clase 22

IIC 3413

Prof. Cristian Riveros

# Complejidad de consultas relacionales



# Complejidad de consultas relacionales

- ¿es posible mejorar más nuestro **optimizador**?
- ¿existe una estrategia **mejor** para evaluar consultas?
- ¿cuáles son las consultas más **difíciles**?

# Outline

Evaluación de consultas

Optimización de consultas

Consultas conjuntivas (CQ)

Evaluación de CQ

Optimización de CQ

# Outline

# ¿qué tan complejo es evaluar una consulta SQL?

Problema de **enumeración**:

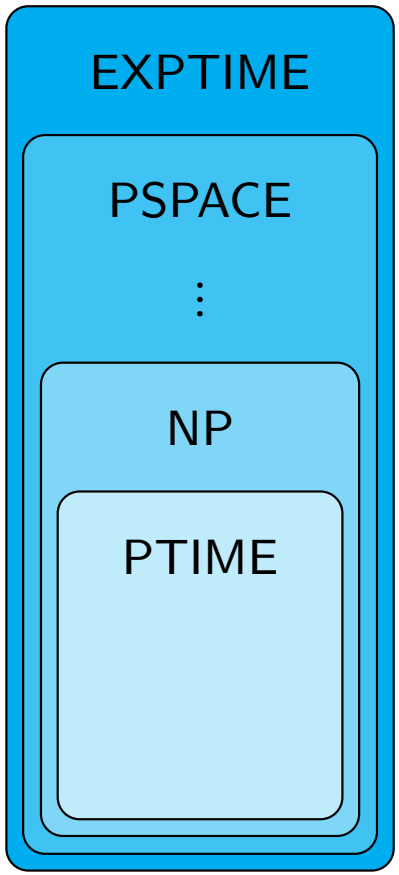
| | |
|---|---|
| `PROBLEMA:` | Evaluación de consultas en `SQL` (SQL-Enum). |
| `INPUT:` | una consulta $Q$ en `SQL`, |
| | una BD relacional $\mathcal{D}$. |
| `OUTPUT:` | $Q(\mathcal{D})$. |

Queremos un **algoritmo de enumeración** que sea polinomial en $Q$ y $\mathcal{D}$:

- tiempo polinomial en $Q$ y $\mathcal{D}$ para entregar la **primera tupla** de $Q(\mathcal{D})$, y
- tiempo polinomial en $Q$ y $\mathcal{D}$ entre cada **siguiente tupla** de $Q(\mathcal{D})$.

¿cómo medimos la complejidad de SQL-Enum?

# Micro-curso de complejidad computacional




- **PTIME:** problemas que pueden ser resueltos en **tiempo polinomial** en el tamaño del input.
- **NP:** problemas cuya solución puede ser **verificada** en **tiempo polinomial** en el tamaño del input/solución.
- **PSPACE:** problemas que pueden ser resultos en **espacio polinomial** en el tamaño del input.
- **EXPTIME:** problemas que pueden ser resueltos en **tiempo exponencial** en el tamaño del input.

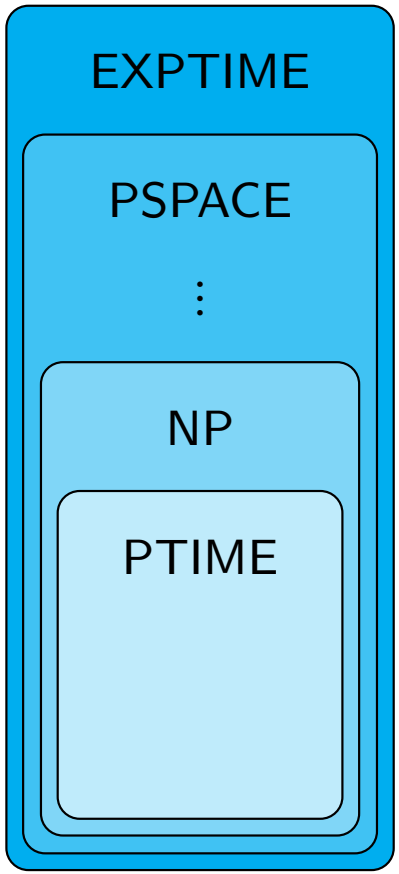
## Definición

- Un problema $P$ es **hard** para una clase de complejidad $\mathcal{C}$ si todos los problemas $P' \in \mathcal{P}$ se pueden reducir (en tiempo polinomial) a $P$.

- Un problema $P$ es **completo** para una clase de complejidad $\mathcal{C}$ si:
  1. $P \in \mathcal{C}$.
  2. $P$ es hard para $\mathcal{C}$.

# Micro-curso de complejidad computacional

Problemas completos para cada clase:




- **PTIME:** programación lineal, horn-SAT, circuit-eval.
- **NP:** SAT, problemas en grafo.
- **PSPACE:** QBF-SAT, juegos/puzzles.
- **EXPTIME:** ajedrez.

# ¿qué tan complejo es evaluar una consulta SQL?

Problema de **enumeración**:

| | |
|---|---|
| `PROBLEMA:` | Evaluación de consultas en `SQL` (SQL-Enum). |
| `INPUT:` | una consulta $Q$ en `SQL`, |
| | una BD relacional $\mathcal{D}$. |
| `OUTPUT:` | $Q(\mathcal{D})$. |

Necesitamos un **problema de decisión** asociado a SQL-Enum!

# Problema de decisión asociado a SQL-Enum

`PROBLEMA:` Resultado no-vacío de consultas `SQL` (SQL-Emptyness).

`INPUT:` una consulta $Q$ en `SQL`,

una BD relacional $\mathcal{D}$

`OUTPUT:` `TRUE` ssi $Q(\mathcal{D}) \neq \emptyset$.

1. Si SQL-Emptyness no esta en PTIME (ej. es NP-hard),
   ¿implica que SQL-Enum NO se puede enumerar en tiempo polinomial?

2. Si SQL-Emptyness esta en PTIME,
   ¿implica que SQL-Enum se puede enumerar en tiempo polinomial?

SQL-Emptyness **solo nos puede dar evidencia** si el problema es difícil

# ¿qué tan complejo es evaluar una consulta SQL?

## Teorema

El problema SQL-EMPTYNESS es PSPACE-completo.

A menos que P = PSPACE, **no existe** un algoritmo de enumeración eficiente (en tiempo polinomial) para SQL-ENUM

¿cuáles son las consultas `SQL` **difíciles** de evaluar?

- Consultas de la forma: `NOT EXIST... EXIST... NOT EXIST...`
- Consultas con negación anidadas.

# Outline

# Problemas asociados a optimización de consultas en SQL

Para la optimización de consultas en SQL,
nos interesan **algoritmos eficientes** para los siguientes problemas:

| | |
|---|---|
| PROBLEMA: | Satisfabilidad de SQL (SQL-SAT). |
| INPUT: | una consulta $Q$ en SQL, |
| OUTPUT: | TRUE ssi existe $\mathcal{D}$ tal que $Q(\mathcal{D}) \neq \emptyset$. |

| | |
|---|---|
| PROBLEMA: | Igualdad de consultas SQL (SQL-Equivalence). |
| INPUT: | consultas $Q_1$ y $Q_2$ en SQL, |
| OUTPUT: | TRUE ssi para todo $\mathcal{D}$ se cumple $Q_1(\mathcal{D}) = Q_2(\mathcal{D})$. |

¿para que nos serviría resolver estos problemas?

# Es imposible tener un optimizador perfecto para SQL

## Teorema

Para SQL, los siguientes problemas son **indecidibles**:

- SQL-Equivalence
- SQL-SAT

**indecidible** = no existe algoritmo alguno que solucione el problema

¿es posible hacer "algo" para mejorar la evaluación/optimización en SQL?

# Outline

# Fragmento más sencillo: consultas conjuntivas

## Definición

Una **consulta conjuntiva** (CQ) es una consulta en AR que solo contiene:

- proyección ($\pi$)
- selección sencilla ($\sigma_{A=B}$ o $\sigma_{A=v}$)
- Equality joins ($\bowtie_{A=B}$)
- Renaming ($\rho_{A \to B}$)

## Ejemplo

```
SELECT   P.name, M.goals
FROM     Players AS P, Matches AS M, Players_Matches AS PM
WHERE    P.pId = PM.pId AND PM.mId = M.mId AND
         P.name = 'Alexi' AND M.year = 2001
```

En otras palabras, una consulta `SELECT-FROM-WHERE`.

# Fragmento más sencillo: consultas conjuntivas

## Definición

Una **consulta conjuntiva** (CQ) es una consulta en AR que solo contiene:

- proyección ($\pi$)
- selección sencilla ($\sigma_{A=B}$ o $\sigma_{A=v}$)
- Equality joins ($\bowtie_{A=B}$)
- Renaming ($\rho_{A\to B}$)

## Sin perdida de generalidad

Desde ahora en adelante consideraremos consultas conjuntivas solo con:

- proyección $\pi$.
- selección $\sigma_{A=v}$.
- natural joins $\bowtie$.

$\sigma_{A=B}$, $\bowtie_{A=B}$ y $\rho_{A\to B}$ no cambian la complejidad del problema.

# Fragmento más sencillo: consultas conjuntivas

## Proposición

Para toda consulta conjuntiva $Q$, existe una consulta $Q'$ tal que $Q(\mathcal{D}) = Q'(\mathcal{D})$ para toda BD $\mathcal{D}$ y $Q'$ es de la forma:

$$\pi_I\big(\sigma_{c_1}(R_1) \bowtie \ldots \bowtie \sigma_{c_n}(R_n)\big)$$

con cada $c_i$ una conjunción filtros $A = v$.

Demostración: use las reglas de reescritura.

# Representación simplificada de consultas conjuntivas

Sea **V** un conjunto de variables y **C** un conjunto de constantes.

## Simplificación

Desde ahora una consulta conjuntiva la representaremos como:

$$ans(\bar{y}) \ := \ R_1(\bar{x}_1), R_2(\bar{x}_2), \ldots, R_n(\bar{x}_n)$$

1. $\bar{x}_1, \ldots, \bar{x}_n$ son variables en **V** o constantes en **C**,
2. $\bar{y}$ es un subconjunto de variables en $\bar{x}_1, \ldots, \bar{x}_n$.

### Ejemplo

$$ans(x, z) \ := \ P(x, \text{'Alexi'}), PM(x, y), M(y, 2001, z)$$

- $x, y, z$ son variables.
- 'Alexi' y 2001 son constantes.

# Representación simplificada de consultas conjuntivas

Sea **V** un conjunto de variables y **C** un conjunto de constantes.

## Simplificación

Desde ahora una consulta conjuntiva la representaremos como:

$$ans(\bar{y}) \ := \ R_1(\bar{x}_1), R_2(\bar{x}_2), \ldots, R_n(\bar{x}_n)$$

1. $\bar{x}_1, \ldots, \bar{x}_n$ son variables en **V** o constantes en **C**,
2. $\bar{y}$ es un subconjunto de variables en $\bar{x}_1, \ldots, \bar{x}_n$.

## Notación

- $R_1(\bar{x}_1), \ldots, R_n(\bar{x}_n)$ es el **cuerpo** de $Q$ y $ans(\bar{y})$ es la **cabeza** de $Q$.
- cada $R_i(\bar{x}_i)$ es un **átomo** de $Q$.
- si $\bar{y}$ es **vacía**, entonces hablamos de una **consulta booleana**.

# Homomorfismo de consultas conjuntivas

## Definición

Un **homomorfismo** de $Q$ a $\mathcal{D}$ es una función $h : (\mathbf{V} \cup \mathbf{C}) \to \mathbf{C}$ tal que:

- $h(c) = c$ para toda $c \in \mathbf{C}$ y
- si $R(d_1, \ldots, d_k)$ es un átomo de $Q$,

  entonces $(h(d_1), \ldots, h(d_k)) \in \mathcal{D}(R)$.

¿cuál es un homomorfismo de $Q$ a $\mathcal{D}$?

$$Q: \quad anx(x, z) := P(x, \text{'Alexi'}, y), M(x, z, \text{'3'})$$

$\mathcal{D}$:

**Players** (P):

| Id | Name | Year |
|---|---|---|
| 1 | Alexi | 1987 |
| 2 | Gary | 1990 |
| 3 | Arturo | 1985 |

**Matches** (M):

| Id | Stadium | Goals |
|---|---|---|
| 1 | Nacional | 3 |
| 1 | Monumental | 3 |
| 2 | San Carlos | 4 |

# Homomorfismo de consultas conjuntivas

## Definición

Un **homomorfismo** de $Q$ a $\mathcal{D}$ es una función $h : (\mathbf{V} \cup \mathbf{C}) \to \mathbf{C}$ tal que:

- $h(c) = c$ para toda $c \in \mathbf{C}$ y
- si $R(d_1, \ldots, d_k)$ es un átomo de $Q$,

  entonces $(h(d_1), \ldots, h(d_k)) \in \mathcal{D}(R)$.

## Proposición

Para toda base de datos $\mathcal{D}$ y toda consulta conjuntiva $Q$ de la forma:

$$ans(y_1, \ldots, y_k) \ := \ R_1(\bar{x}_1),\ R_2(\bar{x}_2),\ \ldots,\ R_n(\bar{x}_n)$$

se tiene que $t \in Q(\mathcal{D})$ si, y solo si, existe un homomorfismo $h$ de $Q$ a $\mathcal{D}$ con

$$t = (h(y_1), \ldots, h(y_k)).$$

Demostración: ejercicio.

# Outline

# ¿qué tan complejo es evaluar una consulta conjuntiva?

Problema de **decisión**:

| | |
|---|---|
| `PROBLEMA:` | Resultado no-vacío de consultas conjuntivas (CQ-EMPTYNESS). |
| `INPUT:` | una consulta conjuntiva $Q$, |
| | una BD relacional $\mathcal{D}$ |
| `OUTPUT:` | `TRUE` ssi $Q(\mathcal{D}) \neq \emptyset$. |

## Teorema

El problema CQ-EMPTYNESS es NP-completo.

Demostración: ejercicio.

# ¿estamos modelando el problema correctamente?

`PROBLEMA:` Resultado no-vacío de consultas conjuntivas (CQ-Emptyness).

`INPUT:` una consulta conjuntiva $Q$,

una BD relacional $\mathcal{D}$

`OUTPUT:` `TRUE` ssi $Q(\mathcal{D}) \neq \varnothing$.

En la práctica tenemos que:

$$|Q| \ << \ |\mathcal{D}|$$

Consultas son muchísimo más pequeñas que los datos.

# Complejidad en término de los datos

- **Combined**-complexity: consulta y datos son parte del input.
- **Data**-complexity: solo los datos son parte del input (consulta esta fija).

`PROBLEMA:` Resultado no-vacío de consultas conjuntivas $Q$ ($\text{CQ-EVAL}_Q$).

`INPUT:` una BD relacional $\mathcal{D}$

`OUTPUT:` $t \in Q(\mathcal{D})$.

# Complejidad en término de los datos

**Teorema**

El problema $\textsc{ConjSQL-Eval}_Q$ esta en PTIME para todo consulta $Q \in \texttt{SQL}$.

¿es posible hacer una análisis mas fino?

# Outline

# Equivalencia y satisfiabilidad de consultas conjuntivas

## Definición

Un **homomorfismo** de $Q_1$ a $Q_2$ es una función $h : (\mathbf{V} \cup \mathbf{C}) \to (\mathbf{V} \cup \mathbf{C})$:

- $h(c) = c$ para toda $c \in \mathbf{C}$,
- si $R(d_1, \ldots, d_k)$ es un átomo de $Q_1$,
  entonces $R(h(d_1), \ldots, h(d_k))$ es un átomo de $Q_2$,
- si $ans(y_1, \ldots, y_k)$ es el cuerpo de $Q_1$,
  entonces $ans(h(y_1), \ldots, h(y_k))$ es el cuerpo de $Q_2$.

## Proposición

Para todo par de consultas conjuntivas $Q_1$ y $Q_2$ se tiene que:

1. $Q_1(\mathcal{D}) \subseteq Q_2(\mathcal{D})$ para toda $\mathcal{D}$ si, y solo si,
2. existe un homomorfismo de $Q_2$ a $Q_1$.

# Equivalencia y satisfiabilidad de consultas conjuntivas

`PROBLEMA:` Satisfabilidad de consultas conjuntivas. (CQ-SAT).

`INPUT:` una consulta conjuntiva $Q$,

`OUTPUT:` `TRUE` ssi existe $\mathcal{D}$ tal que $Q(\mathcal{D}) \neq \varnothing$.

`PROBLEMA:` Igualdad de consultas conjuntivas (CQ-Equivalence).

`INPUT:` consultas conjuntivas $Q_1$ y $Q_2$,

`OUTPUT:` `TRUE` ssi para todo $\mathcal{D}$ se cumple $Q_1(\mathcal{D}) = Q_2(\mathcal{D})$.

## Teorema

- CQ-SAT es un problema trivial (siempre es satisfacible).
- CQ-Equivalence es NP-completo.