DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Integración de datos - Consultas

Fernando Berzal, berzal@acm.org

Integración de datos

- Descripción de fuentes de datos
- Emparejamiento de cadenas [string matching]
- Integración de esquemas
  - Emparejamiento de esquemas [schema matching]
  - Correspondencias entre esquemas [schema mapping]
  - Gestión de modelos
- Emparejamiento de datos [data matching]
- Wrappers
- Apéndice: Procesamiento de consultas

Procesamiento de consultas

- Procesamiento de consultas en bases de datos
  - Datalog
  - Query unfolding
  - Query containment
  - Usando vistas para responder consultas
  - Procesamiento de consultas en un DBMS
- Procesamiento de consultas distribuidas
- Consultas en integración de datos

Procesamiento de consultas

Dada una base de datos D, al ejecutar una consulta Q se obtiene un conjunto de datos Q(D).

Lenguajes de especificación de consultas

- SQL [Structured Query Language]
- Datalog

Procesamiento de consultas

EJEMPLO

Interview
(candidate, date, recruiter, hireDecision, grade)
EmployeePerf
(empID, name, reviewQuarter, grade, reviewer)

- Clave primaria $PK_{EmployeePerf}$ = { EmpID, reviewQuarter }
- Clave externa FK: EmployeePerf.EmpID → Employee.ID

Procesamiento de consultas

Datalog en 5 minutos

Reuniones

Consultas conjuntivas en las que aparecen múltiples ocurrencias de la misma variable, p.ej.

```
Q(X,T) :-   Interview(X,D,Y,H,F),
            EmployeePerf(E,Y,T,W,Z),
            W < 2.5.
```

SQL

```
select recruiter, candidate
from Interview, EmployeePerf
where recruiter=name
AND grade < 2.5
```

Procesamiento de consultas

Datalog en 5 minutos

Negaciones

Restricción: Todas las variables de la consulta deben aparecer en un subobjetivo positivo (no negado), p.ej.

```
Q(X,T) :-   Interview(X,D,Y,H,F),
            EmployeePerf(E,Y,T,W,Z),
            ¬OfferMade(X, date).
```


Procesamiento de consultas

Datalog en 5 minutos

Uniones

Múltiples reglas para el mismo predicado:

```
Q(R,C) :-   Interview(X,D,Y,H,F),
            EmployeePerf(E,Y,T,W,Z),
            W < 2.5.

Q(R,C) :-   Interview(X,D,Y,H,F),
            EmployeePerf(E,Y,T,W,Z),
            Manager(y),
            W > 3.9.
```

Procesamiento de consultas

Datalog en 5 minutos

Recursividad

Si edge(X,Y) describe las aristas de un grafo, la siguiente consulta encuentra todos los caminos en el grafo:

```
path(X,Y) :-  edge(X,Y).

path(X,Y) :-  edge(X,Z),
              path(Z,Y).
```


Procesamiento de consultas

Query unfolding
[despliegue/"desenrollado" de consultas]

La composición de consultas es una forma habitual de elaborar consultas complejas,

p.ej. consultas a partir de vistas

"Query unfolding" deshace la composición:

- Comparación entre consultas sobre vistas.
- Optimización de consultas.
- Identificación de consultas no satisfacibles (compuestas de consultas satisfacibles).

Procesamiento de consultas

Query unfolding
[despliegue/"desenrollado" de consultas]

EJEMPLO

$$Q_1(X,Y) :- Flight(X,Z), Hub(Z), Flight(Z,Y)$$
$$Q_2(X,Y) :- Hub(Z), Flight(Z,X), Flight(X,Y)$$
$$Q_3(X,Z) :- Q_1(X,Y), Q_2(Y,Z)$$

$$Q'_3(X,Z) :- Flight(X,U), Hub(U), Flight(U,Y),$$
$$Hub(W), Flight(W,Y), Flight(X,Z)$$

Procesamiento de consultas

Query containment

Intuitivamente, el despliegue [unfolding]
de las dos consultas siguientes es equivalente:

$$Q_5(X,Z) :- Flight(X,U), Hub(U), Flight(U,Y),$$
$$Hub(Y), Flight(Y,Z)$$

$$Q'_3(X,Z) :- Flight(X,U), Hub(U), Flight(U,Y),$$
$$Flight(X,Z)$$

Procesamiento de consultas

Query containment

- La consulta Q_1 está contenida en la consulta Q_2 si, para **toda** base de datos D, $Q_1(D) \subseteq Q_2(D)$
- La consulta Q_1 es equivalente a la consulta Q_2 si $Q_1(D) \subseteq Q_2(D)$ y $Q_2(D) \subseteq Q_1(D)$

¿Por qué nos interesa saberlo?

- Al describir fuentes de datos como vistas, nos sirve para comparar las fuentes de datos.
- Si eliminamos subobjetivos de un consulta, podemos evaluarla de forma más eficiente (optimización).

Procesamiento de consultas

Query containment

EJEMPLOS

$$Q_1(X,Z) :- p(X,Y,Z)$$
$$Q_2(X,Z) :- p(X,X,Z)$$

$$Q_1 \supseteq Q_2$$

$$Q_1(X,Y) :- p(X,Z), p(Z,Y)$$
$$Q_2(X,Y) :- p(X,Z), p(Z,Y), p(X,W)$$

Procesamiento de consultas

Query containment

EJEMPLO COMPLETO

- Relaciones: Flight(source, destination) & Hub(city)
- Vistas:
  $Q_1(X,Y)$:- Flight(X,Z), Hub(Z), Flight(Z,Y).
  $Q_2(X,Y)$:- Hub(Z), Flight(Z,X), Flight(X,Y).
- Consulta:
  $Q_3(X,Z)$:- $Q_1(X,Y)$, $Q_2(Y,Z)$.
- Query unfolding:
  $Q'_3(X,Z)$:- Flight(X,U), Hub(U), Flight(U,Y),
  Hub(W), Flight(W,Y), Flight(Y,Z).
- Eliminación de objetivos redundantes:
  $Q''_3(X,Z)$:- Flight(X,U), Hub(U), Flight(U,Y), Flight(Y,Z).

Procesamiento de consultas

Usando vistas para responder consultas

¿Cómo decide un sistema de integración de datos qué fuentes de datos son relevantes para una consulta? ¿cúales son redundantes? ¿cómo combinar múltiples fuentes de datos para responder la consulta?

Razonando sobre los contenidos de las fuentes de datos.

Las fuentes de datos
se describen mediante vistas (i.e. consultas).

Procesamiento de consultas

Usando vistas para responder consultas

EJEMPLO 1

```
Movie(ID,title,year,genre)
Director(ID,director)
Actor(ID, actor)
```

$$Q(T,Y,D) :- Movie(I,T,Y,G), Y \geq 1950, G = "comedy"$$
$$Director(I,D), Actor(I,D)$$

$$V_1(T,Y,D) :- Movie(I,T,Y,G), Y \geq 1940, G = "comedy"$$
$$Director(I,D), Actor(I,D)$$

$$V_1 \supseteq Q \Rightarrow \quad Q'(T,Y,D) :- V_1(T,Y,D), Y \geq 1950$$

V_1 puede utilizarse para responder la consulta Q :-)

Procesamiento de consultas

Usando vistas para responder consultas

EJEMPLO 2

```
Movie(ID,title,year,genre)
Director(ID,director)
Actor(ID, actor)
```

$$Q(T,Y,D) :- Movie(I,T,Y,G), Y \geq 1950, G = "comedy"$$
$$Director(I,D), Actor(I,D)$$

$$V_2(I,T,Y) :- Movie(I,T,Y,G), Y \geq 1950, G = "comedy"$$

$$V_3(I,D) :- Director(I,D), Actor(ID,D)$$

En este caso, la consulta no está incluida en una vista, pero V_2 y V_3 pueden utilizarse para resolver la consulta:

$$Q''(T,Y,D) :- V_2(I,T,Y), V_3(I,D)$$

Procesamiento de consultas

Usando vistas para responder consultas

Dada una consulta Q y un conjunto de vistas $V_1..V_n$, una **reescritura [rewriting]** de Q es otra consulta Q' que sólo hace referencias a las vistas $V_1..V_n$.

Una **reescritura equivalente** de Q es una reescritura Q' tal que Q' $\Leftrightarrow$ Q.

¿Para qué?

- Integración de datos LAV.
- Optimización de consultas (DBMS comerciales).
- Diseño físico de bases de datos

Procesamiento de consultas

Usando vistas para responder consultas

ALGORITMOS

Búsqueda eficiente en el espacio de reescrituras

- **Bucket algorithm** [VLDB'1996]
  - Producto cartesiano de los subobjetivos de la consulta.
  - Útil si la consulta incluye predicados interpretados (condiciones).
- **MiniCon algorithm** [VLDB Journal'2001]
  - Detecta interacciones entre los subobjetivos de la consulta.
  - Algoritmo más eficiente en la práctica (satisfacción de restricciones).

Enfoque "lógico"

- **Inverse-Rules algorithm** [PODS'1997]
  - Conceptualmente elegante.
  - Puede producir reescrituras ineficientes.

Procesamiento de consultas

Usando vistas para responder consultas

COMPLEJIDAD COMPUTACIONAL

Problema NP-completo (o peor, según el caso particular):

Grafo G(V,E):

$$v_1(X) :- color(X,Y) \qquad I(V_1) = V$$
$$v_2(Y) :- color(X,Y) \qquad I(V_2) = \{red, green, blue\}$$
$$v_3(X,Y) :- edge(X,Y) \qquad I(V_3) = E$$

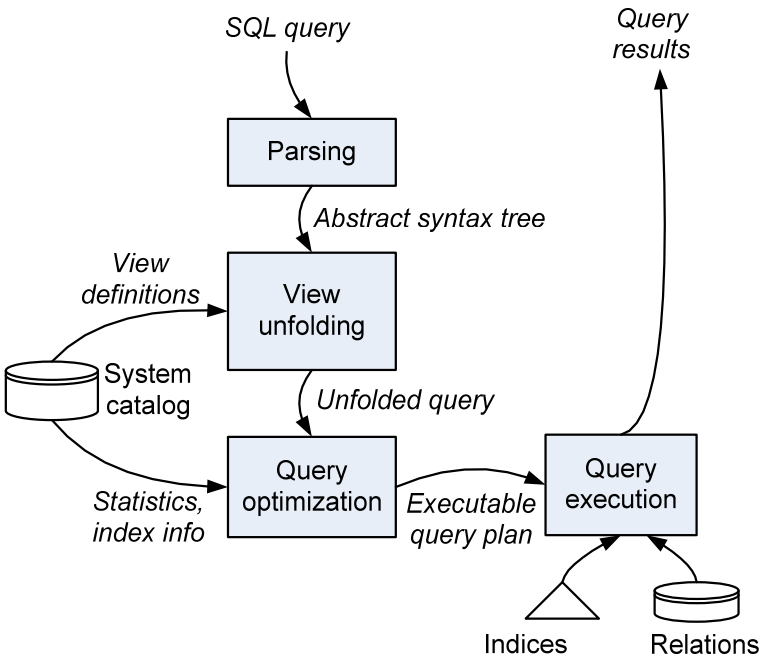
$$q() :- edge(X,Y), color(X,Z), color(Y,Z)$$

La respuesta incluye una tupla si y sólo si G es 3-coloreable.

En la práctica no suele suponer un problema.

Procesamiento de consultas

Procesamiento de consultas en un DBMS

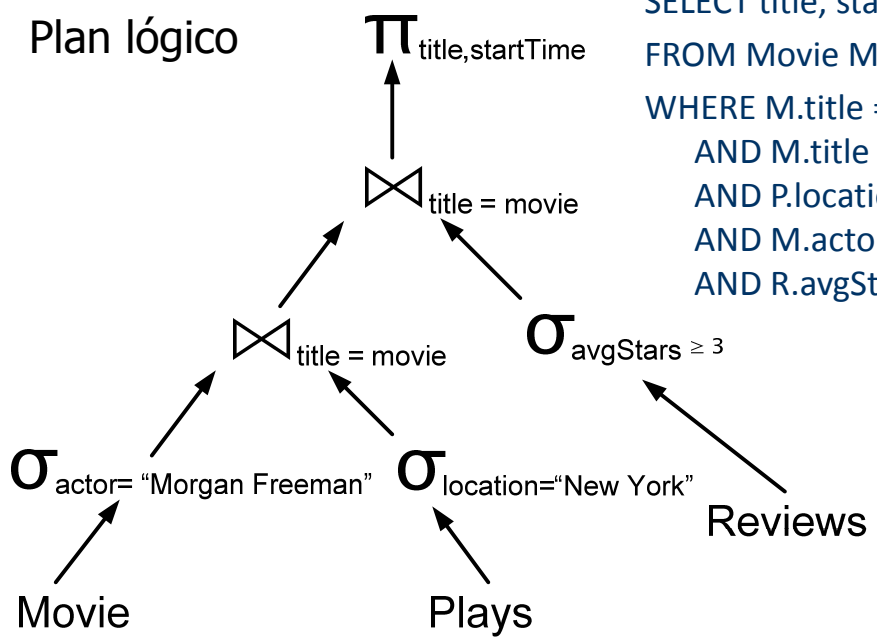


Procesamiento de consultas

Procesamiento de consultas en un DBMS

EJEMPLO

Plan lógico



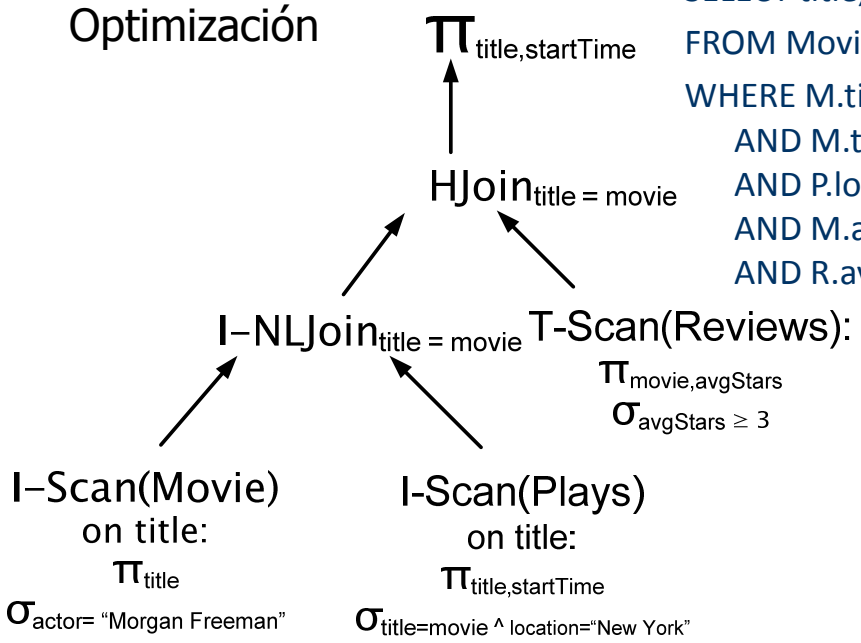

```
SELECT title, startTime
FROM Movie M, Plays P, Reviews R
WHERE M.title = P.movie
    AND M.title = R.movie
    AND P.location ="New York "
    AND M.actor = "Morgan Freeman"
    AND R.avgStars >= 3
```


Procesamiento de consultas

Procesamiento de consultas en un DBMS

EJEMPLO

Optimización




```
SELECT title, startTime
FROM Movie M, Plays P, Reviews R
WHERE M.title = P.movie
    AND M.title = R.movie
    AND P.location ="New York "
    AND M.actor = "Morgan Freeman"
    AND R.avgStars >= 3
```

Procesamiento de consultas

Procesamiento de consultas en un DBMS

OPTIMIZACIÓN DE CONSULTAS

Enumeración de planes (búsqueda)

- Expresiones equivalentes (álgebra relacional):
  $\sigma_a(R \bowtie S) = (\sigma_a(R) \bowtie S)$
- Selección y proyección lo antes posible.
- Reuniones (programación dinámica):
  $R \bowtie S \bowtie T = (R \bowtie S) \bowtie T = (R \bowtie T) \bowtie S = (S \bowtie T) \bowtie R$

Estimación de costes

- Tamaño de los conjuntos de datos.
- Coste de cada algoritmo (E/S, CPU…).
- Parámetros de calibración para el servidor concreto.

Procesamiento de consultas

Procesamiento de consultas en un DBMS

EJECUCIÓN DE CONSULTAS

Granularidad

- Operadores procesan relaciones.
- Operadores procesan tuplas [pipelining]:
  No siempre posible, p.ej. GROUP BY sobre tuplas no ordenadas o HJoin [hash join], que necesita un buffer para una de sus entradas.

Flujo de control

- Basado en iteradores
  (desde la raíz del árbol del plan físico de la consulta)
- Basado en flujo de datos
  (desde las hojas, conforme llegan tuplas).

Consultas distribuidas

Procesamiento des consultas cuando tenemos los datos distribuidos en múltiples ordenadores:

- **DBMS paralelos**
  (nodos homogéneos y redes de interconexión rápidas, puede que con memoria compartida).

- **DBMS distribuidos**
  (nodos potencialmente heterogéneos y redes de comunicación más lentas, con algunos recursos sólo disponibles en determinadas máquinas).

Consultas distribuidas

El problema de la localización de los datos

El rendimiento del sistema
depende de la forma de distribuir los datos:

- Partición por relaciones (tablas en máquinas diferentes)
- Partición vertical (por columnas)
- Partición horizontal (por filas)

Además de la posibilidad de replicar particiones
(trade-off consultas vs. actualizaciones)

Consultas distribuidas

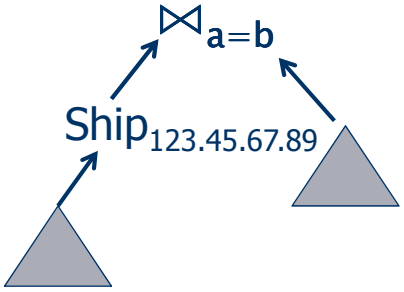
Muchas más opciones de ejecución de planes con diferentes costes, que en el caso tradicional.

Operadores nuevos que se incorporan en los planes de ejecución de consultas:

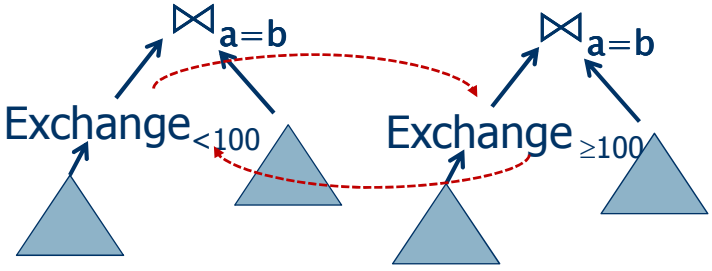
- **Ship** para enviar tuplas de una máquina a otra.
- **Rehash/exchange** para redistribuir tuplas.
- **Reunión en dos fases** [two-phase join] para reducir la comunicación en reuniones distribuidas.

Consultas distribuidas

Ship



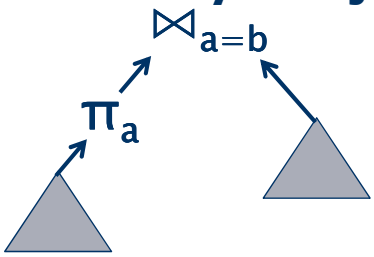
Rehash/exchange



Consultas distribuidas

Reunión en dos fases [two-phase join]

Two-way semijoin



- Take R, project join key attributes
- Ship to S
- Fetch all matching S tuples
- Ship to destination
- Join R tuples with S tuples

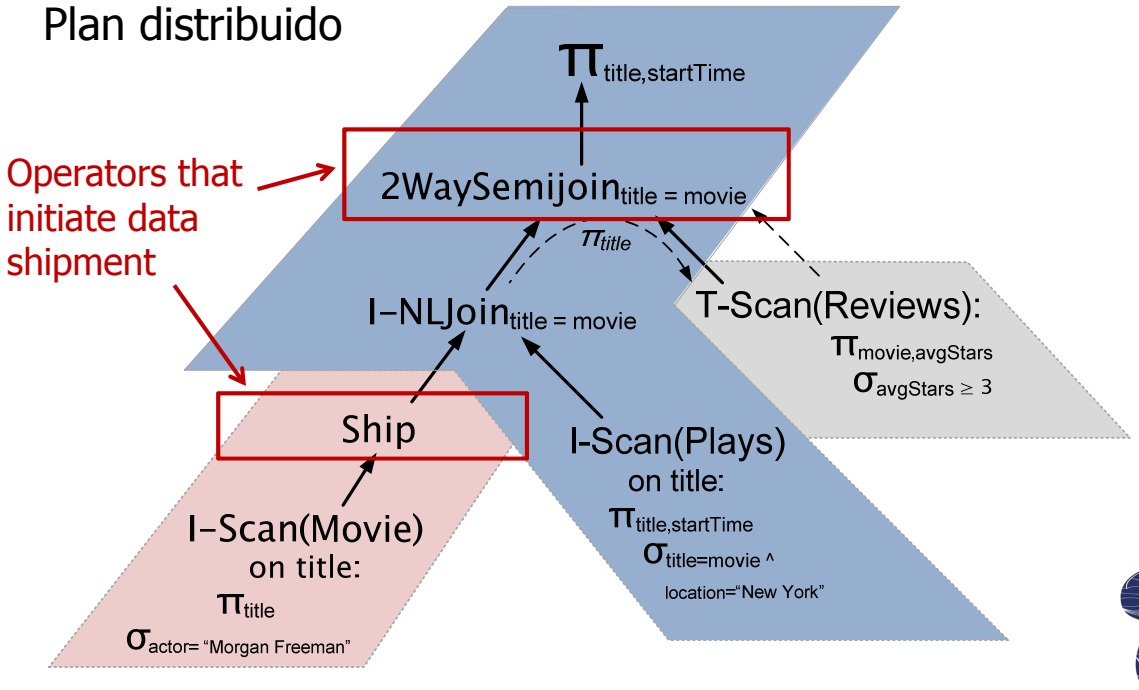
Bloomjoin

- Take R, build *Bloom filter*
  - Build a large bit vector
  - Take each value of R.a, hash with multiple functions $h_i(a)$
  - Set bits for each $h_i(a)$
- Ship to S
- Fetch all matching S tuples
  - For each S.b, ensure a match to each $h_i(b)$ in Bloom filter
- Ship to destination
- Join R tuples with S tuples

Consultas distribuidas

EJEMPLO

Plan distribuido



Integración de datos

Queremos procesar consultas distribuidas pero…

- No tenemos índices ni estadísticas fiables.
- Todos los datos están alojados de forma remota.
- Tenemos restricciones en la forma de acceder a los datos.
- Los resultados queremos mostrarlos conforme lleguen.

Integración de datos

El procesador de consultas recibe
una consulta reformulada y poco más
(no podemos utilizar técnicas de optimización):

- Plan de consulta inicial.
- Ejecución de la consulta [pipelining].
- Procesamiento adaptativo de la consulta.

Integración de datos

Plan de consulta inicial

Como en la optimización de consultas para bases de datos distribuidas, pero sin estimaciones fiables de los costes:

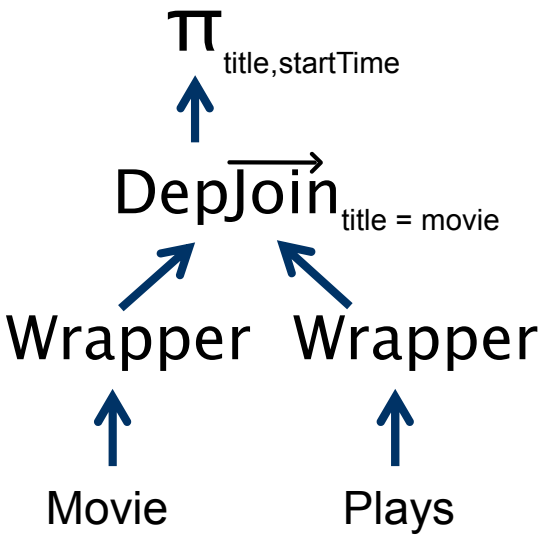
- Uso de heurísticas para establecer perfiles de las fuentes de datos (rendimiento y ancho de banda).
- Estimaciones conservadoras de los tamaños de los resultados intermedios, i.e. sobreestimaciones.

Restricciones adicionales introducidas por los wrappers (patrones de acceso y capacidades de consulta).

Integración de datos

Plan de consulta inicial

Restricciones en los patrones de acceso

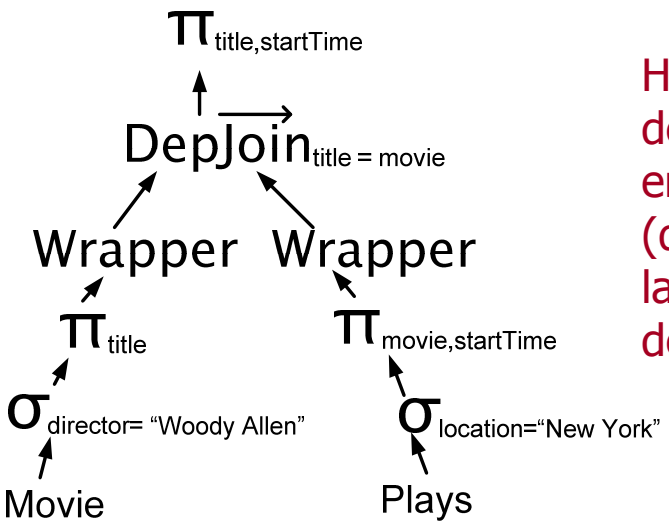


Reunión dependiente
Suministra datos
de su primera entrada
al wrapper
de su segunda entrada

Integración de datos

Plan de consulta inicial

Restricciones en las capacidades de consulta

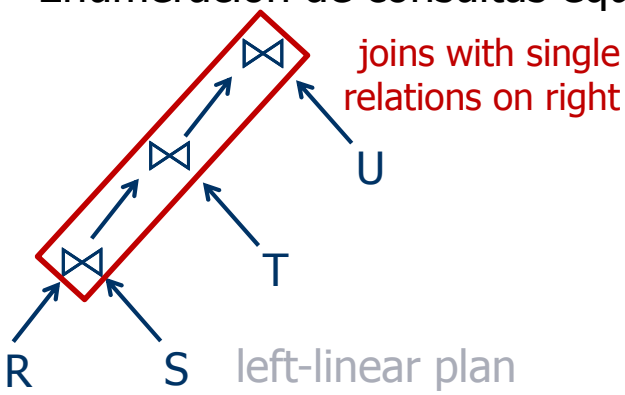


Hay que estimar el coste de delegar operaciones en las fuentes de datos (que depende de su carga, la cual puede que también desconozcamos).

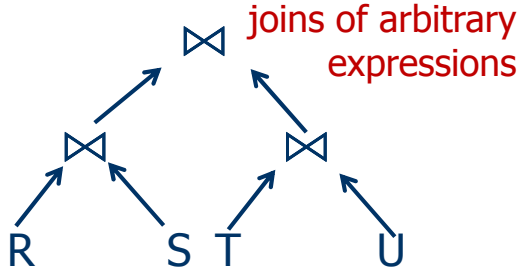
Integración de datos

Plan de consulta inicial

- Enumeración de consultas equivalentes:



left-linear plan



non-linear ("bushy") plan

- Estimaciones de costes diferentes, adaptadas a los wrappers y las fuentes individuales.

Integración de datos

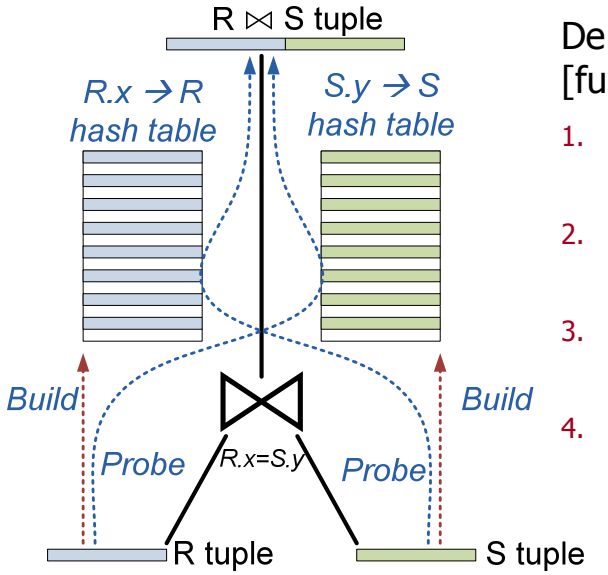
Ejecución de consultas

- Las fuentes de datos en Internet presentan algunos desafíos prácticos: tasas de transferencia de datos bajas e impredecibles, problemas de acceso (fallos)…
- Cada fuente de datos manda sus datos a distinto ritmo: no nos sirve una ejecución secuencial.
- Implementación concurrente (múltiples hebras: una para cada fuente de datos más otra para la consulta): **pipelined hash join**.

Integración de datos

Ejecución de consultas

PIPELINED HASH JOIN [A.K.A. SYMMETRIC HASH JOIN]

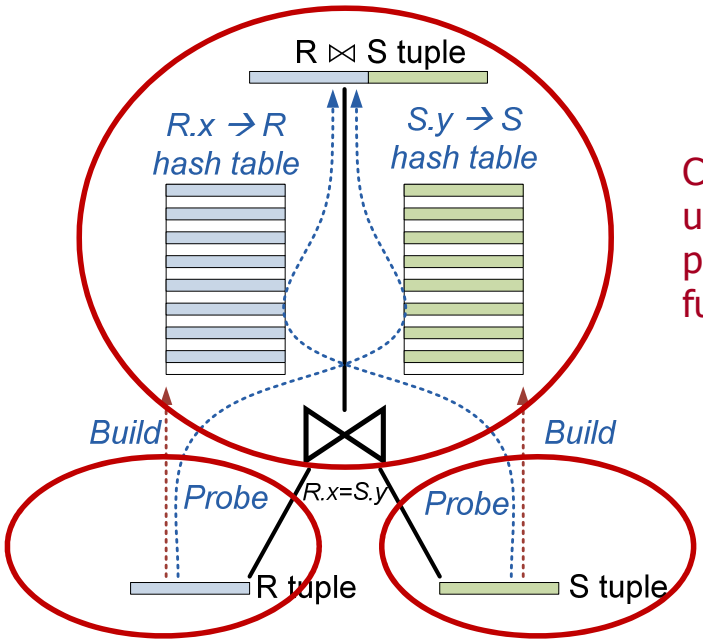


De forma simétrica [fully pipelined]:

1. Leer de cualquiera de las entradas.
2. Añadir la tupla a la tabla hash de la entrada recibida.
3. Consultar la tabla hash de la otra entrada.
4. Devolver la salida obtenida.

Ejecución de consultas

PIPELINED HASH JOIN [A.K.A. SYMMETRIC HASH JOIN]

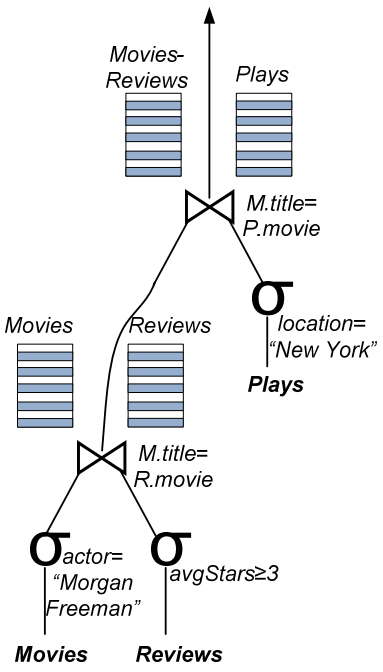


Cada elipse roja representa una hebra diferente, planificada por la CPU en función de su disponibilidad.

Ejecución de consultas

PLANES DE CONSULTA (PIPELINED JOINS)

Uso extensivo de **tablas hash** en lugar de índices o algoritmos de ordenación



Integración de datos

Procesamiento adaptativo de consultas

Es habitual que las fuentes de datos no siempre estén disponibles, por los que los operadores en el plan de ejecución de una consulta pueden generar excepciones: eventos gestionados por el procesador de consultas.

p.ej. retrasos en la llegada de datos,
fallos en las fuentes de datos,
ineficiencias (conjuntos de datos enormes causados por una estimación de costes errónea)

Integración de datos

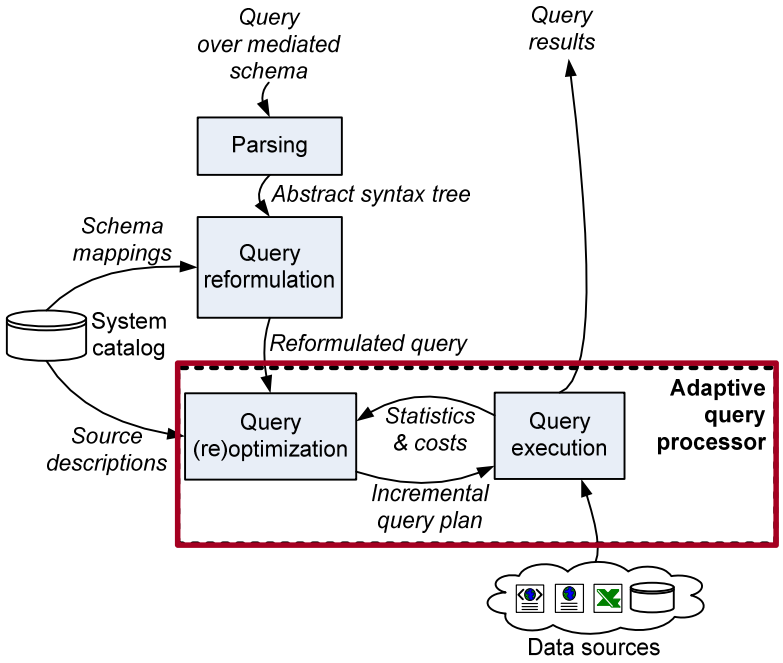
Procesamiento adaptativo de consultas

Exige combinar los módulos de optimización y ejecución de consultas (tradicionalmente separados en un DBMS):

Se comienza a ejecutar un plan
capaz de adaptarse a condiciones cambiantes:

Replanificación

Procesamiento adaptativo de consultas



Procesamiento adaptativo de consultas

ADAPTACIÓN DIRIGIDA POR EVENTOS

on event **if** condition **then** take action

Posibles respuestas si falla una fuente de datos:

- Encontrar una fuente de datos alternativa:
  on timeout(wrapper1, 10msec) **if** true **then** activate(coll1, A)
- Replanificar/suspender un subplan:
  on timeout(wrapper1, 10msec) **if** true **then** reschedule(op2)

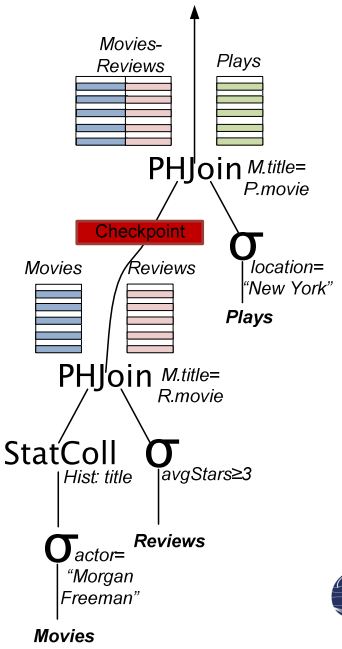
Integración de datos

Procesamiento adaptativo de consultas

ADAPTACIÓN DIRIGIDA POR EVENTOS

Descomposición del plan en varios pipelines y re-optimización si se detecta un error de estimación:

- **StatColl**: Recolección de estadísticas
- **Checkpoint**: Punto de comprobación (vista materializada).



Integración de datos

Procesamiento adaptativo de consultas

ADAPTACIÓN DIRIGIDA POR RENDIMIENTO

El uso de checkpoints es costoso e impide la ejecución segmentada [pipelined] de las consultas.

Podemos cambiar el plan de consulta a mitad de su ejecución sin releer los datos, ¿cómo?

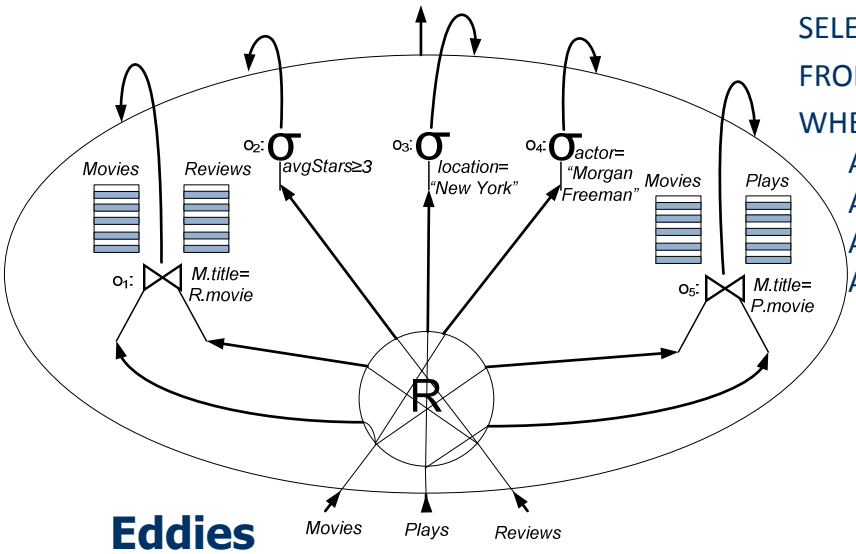
- Descomponemos los datos en segmentos o fases.
- Ejecutamos planes diferentes sobre cada fase.
- Combinamos los resultados.

En Álgebra Relacional: $\Pi(\sigma(\mathbf{R} \bowtie \mathbf{S})) = \cup_i \Pi(\sigma(\mathbf{R}_i \bowtie \mathbf{S}_i))$

Integración de datos

Procesamiento adaptativo de consultas

ADAPTACIÓN DIRIGIDA POR RENDIMIENTO




```
SELECT title, startTime
FROM Movie M, Plays P, Reviews R
WHERE M.title = P.movie
    AND M.title = R.movie
    AND P.location ="New York "
    AND M.actor = "Morgan Freeman"
    AND R.avgStars >= 3
```

Eddies

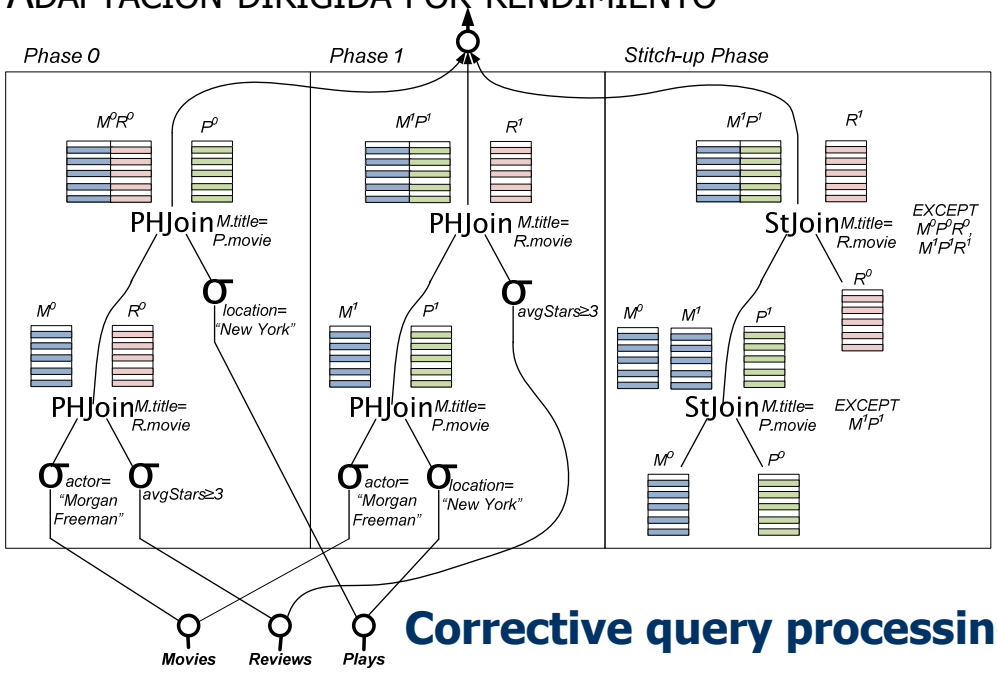
done bits

(title,actor,P.movie,location,R.movie,avgStats, o1,o2,o3,o4,o5)

Integración de datos

Procesamiento adaptativo de consultas

ADAPTACIÓN DIRIGIDA POR RENDIMIENTO



Corrective query processing

Integración de datos

El procesamiento de consultas en integración de datos extiende técnicas ya existentes:

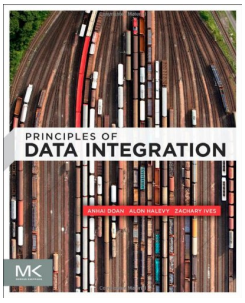
- DBMS centralizados:
  enumeración de planes, modelado de costes…
- DBMS distribuidos:
  costes de comunicación, distribución de datos…

Pero con un nuevo énfasis en:

- Wrappers
  (patrones de acceso y capacidades de consulta).
- Algoritmos segmentados [pipelining].
- Capacidad de adaptación (eventos & rendimiento).

Bibliografía recomendada

- Hai Doan, Alon Halevy & Zachary Ives:
  Principles of Data Integration
  Morgan Kaufmann, 1st edition, 2012.
  ISBN 0124160441
  http://research.cs.wisc.edu/dibook/



Chapter 2: Manipulating Query Expressions

Chapter 8: Query Processing