Optimización de Consultas

Una vez resuelta una consulta en álgebra relacional, debería buscar una consulta o expresión equivalente que sea lo mas óptima posible, es decir en la que se mejore el rendimiento (por ejemplo disminución en el volumen de datos para operar). Esta expresión equivalente será la **FORMA CANÓNICA** de dicha consulta.

Una consulta puede ser expresada en forma de árbol. Un árbol de consultas es una estructura de árbol que corresponde a la expresión en el álgebra relacional de una consulta; donde las hojas del árbol, representan los esquemas que intervienen en la consulta y los nodos internos son operadores del álgebra. La ejecución de una consulta consiste en la ejecución de un nodo interno cada vez que sus operandos están disponibles y reemplazar ese nodo interno por el resultado de la operación. La ejecución termina cuando se ejecuta el nodo raíz.
El árbol que expresa a una consulta en forma canónica, se denomina **ÁRBOL CANÓNICO**.

Reglas de transformación de consultas en álgebra relacional

Para obtener la forma canónica de una consulta, se aplican reglas de transformación de operaciones del álgebra relacional, esto se basa en el hecho de que se pueden expresar consultas equivalentes.
Para definir las reglas de transformación, usaremos los siguientes esquemas:

EMPLEADO (#Emp, nYAp, dir, #DeptoTrab)
DEPARTAMENTO (#Depto, nombre, fechaCreación, #EmpDir)
PROYECTO (#Proy, nomProy, ubicación, #DeptoRespons)

Tenga en cuenta que por cada departamento existe un único director (#EmpDir)

Regla 1 - Cascada de selecciones: Una selección con una condición compuesta puede ser descompuesta en una secuencia de selecciones individuales.

$$\sigma_{c1 \text{ and } c2 \text{ and} \ldots \text{and } cn}(R) \equiv \sigma_{c1}(\sigma_{c2}(\ldots(\sigma_{cn}(R))\ldots)$$

Ejemplo:

$(\sigma_{\text{ubicación}=\text{"La Plata" and nomProy} = \text{"liq\_tarifas"}}(\text{ PROYECTO})$

$\equiv$

$(\sigma_{\text{ubicación}=\text{"La Plata"}} (\sigma_{\text{nomProy} = \text{"liq\_tarifas"}}(\text{ PROYECTO}))$

Importante: en caso de que la condición de selección tuviese conectores lógicos or, no es válida esta regla.

Regla 2 - Conmutatividad de la selección: La operación de selección es conmutativa.

$$\sigma_{c1}(\sigma_{c2}(R)) \equiv \sigma_{c2}(\sigma_{c1}(R))$$

Ejemplo:

$\sigma_{ubicación=\text{"La Plata"}} (\sigma_{nomProy =\text{"liq\_tarifas"}}(\text{PROYECTO}))$
$\equiv$
$(\sigma_{nomProy =\text{"liq\_tarifas"}} (\sigma_{ubicación=\text{"La Plata"}} (\text{PROYECTO}))$

Regla 3 - Cascada de proyecciones: En una cascada de proyecciones todas, excepto la última pueden ser ignoradas.

$$\pi_{L1} (\pi_{L2} (...(\pi_{Ln}(R))...)) \equiv \pi_{L1}(R)$$

Ejemplo:

$\pi_{ubicación} (\pi_{ubicación,\ nombProy} (\text{PROYECTO})) \equiv \pi_{ubicación} (\text{PROYECTO})$

Regla 4 - Conmutatividad de la selección con la proyección: Si una condición *c* de una selección involucra sólo atributos A1,...,An en la lista de la proyección, las dos operaciones pueden conmutarse.

$$\pi_{A1,A2,...,An} (\sigma_{c} (R)) \equiv \sigma_{c} (\pi_{A1,A2,...,An} (R))$$

Ejemplo:

$\pi_{ubicación} (\sigma_{ubicación=\text{"La Plata"}} (\text{PROYECTO}))$
$\equiv$
$\sigma_{ubicación=\text{"La Plata"}} (\pi_{ubicación} (\text{PROYECTO}))$

Nota: el orden en el que se conmutan las proyecciones y las selecciones dependerá de la naturaleza de los datos, ya que en el caso de tablas con muchos atributos, es aconsejable bajar las proyecciones mientras que en el caso de tablas con miles de tuplas insertadas es aconsejable bajar la selección, es decir hacerla lo antes posible.

Regla 5 - Conmutatividad del producto natural: el producto natural es conmutativo

$$R\ |X|_c\ S \equiv S\ |X|_c\ R$$

Ejemplo:

(PROYECTO) $|X|_{\#DeptoRespons=\#Depto}$ (DEPARTAMENTO)
≡
(DEPARTAMENTO) $|X|_{\#DeptoRespons=\#Depto}$ (PROYECTO)

Regla 6 - Conmutatividad de la selección con el producto cartesiano: Si todos los atributos de la condición de la selección involucran sólo atributos de una de las relaciones del producto cartesiano (por ejemplo R) las dos operaciones pueden conmutarse como sigue.

$$\sigma_c (R \times S) \equiv (\sigma_c (R)) \times S$$

Ejemplo:

$\sigma_{ubicación=\text{"La Plata"}}$ (PROYECTO X DEPARTAMENTO)
≡
($\sigma_{ubicación=\text{"La Plata"}}$ (PROYECTO)) X DEPARTAMENTO

Alternativamente, si la condición *c* de la selección pude escribirse como *c1* and *c2* donde *c1* involucra sólo atributos de R y *c2* sólo de S las operaciones se conmutan como:

$$\sigma_c (R \times S) \equiv (\sigma_{c1} (R)) \times (\sigma_{c2} (S))$$

Ejemplo:

$\sigma_{ubicación=\text{"La Plata" and nombre="contable"}}$ (PROYECTO X DEPARTAMENTO)
≡
$\sigma_{ubicación=\text{"La Plata"}}$ (PROYECTO) X $\sigma_{nombre=\text{"contable"}}$ (DEPARTAMENTO)

La misma regla se aplica si la operación es un producto natural

Regla 7 - Conmutatividad de la proyección con el producto natural (*ver limitante con el producto cartesiano*) : Si la lista de la proyección es L= {A1,...,An, B1, ..., Bm} donde A1,...,An son atributos de R y B1,..., Bm son atributos de S. Si la condición del producto natural involucra sólo atributos de L, las dos operaciones pueden conmutarse como:

$$\pi_L(R\ |X|_c\ S) \equiv ((\pi_{A1,A2,\ldots,An}(R))\ |X|_c\ (\pi_{B1,B2,\ldots,Bm}(S))$$

Ejemplo:

π ubicación, nombre, #Depto (PROYECTO |X| #DeptoRespons= #Depto (DEPARTAMENTO))
≡
(π ubicación,# DeptoRespons (PROYECTO) |X| #DeptoRespons = #Depto (π nombre, #Depto(DEPARTAMENTO)))

Nota: en este caso en el producto natural se indica explícitamente cual es el atributo por el cual se hace el producto. Esto se debe a que su nombre en las tablas no es el mismo. Este mismo recurso se usa a lo largo del documento

Si la condición c del producto natural contiene atributos adicionales a L, estos pueden ser agregados a la lista y entonces una proyección al final será necesaria. Por ejemplo si los atributos C1,...Ck de R y D1,...Dj de S están involucrados en la condición c pero no en la lista de la proyección las operaciones se conmutan como:

$$\pi_L(R\ |X|_c\ S) \equiv \pi_L((\pi_{A1,A2,\ldots,An,C1,\ldots,Ck}(R))\ |X|_c\ (\pi_{B1,B2,\ldots,Bm,D1,\ldots Dj}(S))$$

Ejemplo:

π ubicación, nombre (PROYECTO |X| #DeptoRespons =#Depto (DEPARTAMENTO))
≡
π ubicación, nombre(π ubicación, #DeptoRespons (PROYECTO)
|X| #DeptoRespons =#depto (π nombre, #depto(DEPARTAMENTO)))

En el caso del producto cartesiano, al no haber condición se aplica la primera parte de la regla.

Regla 8 - Conmutatividad de operaciones de conjunto: Las operaciones de Unión e Intersección son conmutativas. La resta no lo es.

$$\pi_L(R) \cup \pi_L(S) \equiv \pi_L(S) \cup \pi_L(R)$$

$$\pi_L(R) \cap \pi_L(S) \equiv \pi_L(S) \cap \pi_L(R)$$

Ejemplo:

(π #DeptoRespons(PROYECTO)) ∪ (π #Depto(DEPARTAMENTO))
≡
(π #Depto(DEPARTAMENTO)) ∪ (π #DeptoRespons(PROYECTO))

Regla 9 - Asociatividad del producto natural, producto cartesiano, unión e intersección: Estas cuatro operaciones son individualmente asociativas. Esto es si θ es cualquiera de ellas:

$$(R\ \theta\ S)\ \theta\ T \equiv R\theta\ (S\ \theta\ T)$$

Ejemplo:

$\pi_{nYAp,dir,\#Proy,\#Depto}((\sigma_{ubicación=\text{"La Plata"}}(\text{PROYECTO})$
$|X|_{\#DeptoRespons=\#Depto}(\text{DEPARTAMENTO}))$
$|X|_{\#EmpDir=\#Emp}(\text{EMPLEADO}))$
$\equiv$

$\pi_{nYAp,dir,\#Proy,\#Depto}(\sigma_{ubicación=\text{"La Plata"}}(\text{PROYECTO})$
$|X|_{\#DeptoRespons=\#Depto}(\text{DEPARTAMENTO}$
$|X|_{\#EmpDir=\#Emp}(\text{EMPLEADO})))$

Regla 10 - Conmutatividad de la selección con las operaciones de conjunto: La selección conmuta con la unión, la intersección y la diferencia. Si θ es una de las tres operaciones de conjuntos :

$$\sigma_c\ (R\ \theta\ S) \equiv (\sigma_c\ (R))\ \theta\ (\sigma_c\ (S))$$

Regla 11 - La proyección conmuta con la Unión:

$$\pi_L\ (R \cup S) \equiv (\pi_L\ (R)) \cup (\pi_L\ (S))$$

Nota: Tener en cuenta que R y S deben ser de unión compatible para que la parte de la regla $(\pi_L\ (R)) \cup (\pi_L\ (S)) \rightarrow \pi_L\ (R \cup S)$ valga.

Regla 12 - Otras transformaciones: una condición *c* se puede convertir a una condición equivalente usando las leyes de DeMorgan:

c = NOT (c1 AND c2) = (NOT c1) OR (NOT c2)
c = NOT (c1 OR c2) = (NOT c1) AND (NOT c2)

Algoritmo de transformación de consultas en álgebra relacional:

Una vez definidas las reglas de transformación para optimizar una consulta en álgebra relacional, describiremos los pasos de un algoritmo de transformación de consultas en álgebra relacional que usa las reglas de transformación. Tener en cuenta que este algoritmo no es secuencial. Puede ser que al agregar nuevas proyecciones, se deban rever las reglas propuestas.

a- Usando la regla 1- separar las condiciones compuestas de la selección en una cascada de selecciones
b- Usando las reglas 2, 4, 6, y 10 mover cada operación de selección lo más abajo posible en el árbol de acuerdo con los atributos involucrados en la selección.
c- Usando la regla 9 reacomodar las hojas del árbol de manera que las relaciones con selecciones mas restrictivas sean ejecutadas primero; por más restrictivas se entiende aquellas que produzcan la relación más chica en cantidad de tuplas o en tamaño absoluto.
d- Transformar los productos naturales en productos cartesianos con selecciones cuyas condiciones representan la condición del producto natural. Es decir si tengo $A|X|_{cond}$ B $\equiv \sigma_{cond}$ (A |X| B) $\equiv \sigma_{cond}$ (σ_c (A X B)) donde $_c$ representa la condición del producto natural y $_{cond}$ es alguna condición adicional que haya puesto en el producto natural. Tener en cuenta que $_{cond}$, podría no existir.
e- Usar las reglas 3, 4, 7, y 11 para mover las proyecciones lo más abajo posible en el árbol y *crear nuevas proyecciones cuando sea posible* (nota: agregar proyecciones ayuda a reducir la cantidad de datos que se manipula en las consultas)

Proceso de optimización de consultas en álgebra relacional:

Las etapas principales del proceso de optimización, dada una consulta escrita en álgebra relacional, serán las siguientes:

1. Escribir el árbol de la consulta.
2. Aplicar el algoritmo de optimización, indicando reglas de optimización usadas y pasos del algoritmo realizados para hallar el árbol canónico de la consulta.
3. Explicitar cual es el árbol canónico y escribir la consulta resultante del proceso de optimización (consulta canónica).

Ejemplos

Una vez introducidos los conceptos de optimización, veremos ejemplos de aplicación del proceso propuesto anteriormente.

Ejemplo 1:

Sea el siguiente esquema que modela una biblioteca.

LIBRO(titulo, autor, eNom, nroInv)
EDITORIAL(eNom,eDir,eCiudad)
SOCIO(nom, dir, ciudad, nroSocio)
PRESTAMO(nroSocio, nroInv, fecha)

Supongamos que tenemos una consulta que genera la información completa sobre cada uno de los préstamos (datos del socio y del libro)

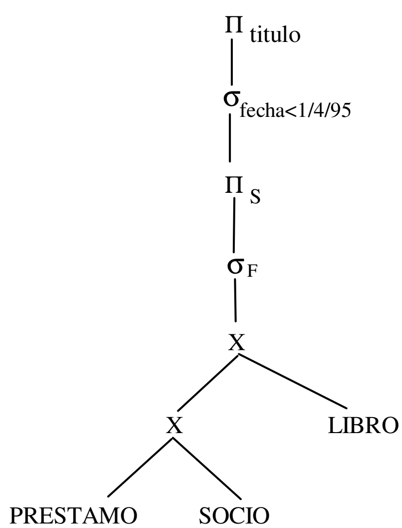
π_S (σ_F (PRESTAMO x SOCIO x LIBRO) donde

F = PRESTAMO.nroSocio=SOCIO.nroSocio and PRESTAMO.nroInv=LIBRO.nroInv
S = título, autor, eNom,nroInv, nom, dir, ciudad, nroSocio, fecha

Ahora queremos refinar esa consulta, de modo de obtener una lista de títulos de libros prestados antes del 1/4/95

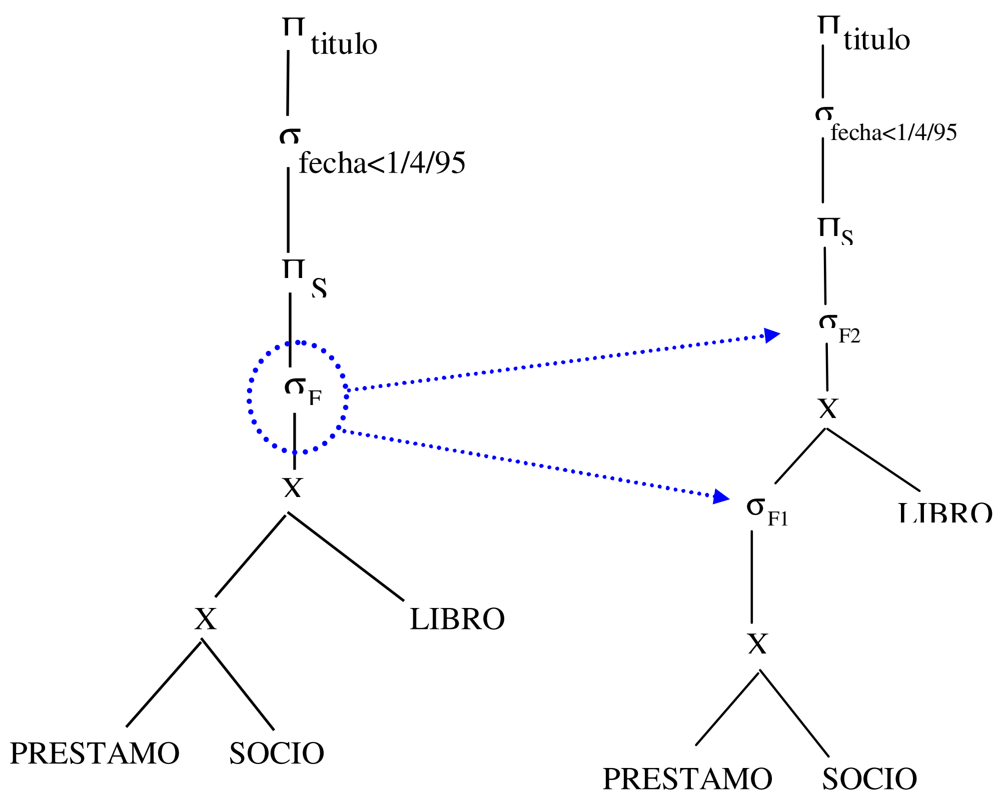
π_{titulo} ($\sigma_{fecha<1/4/95}$ (π_S (σ_F (PRESTAMO x SOCIO x LIBRO))))

El árbol de consulta es el siguiente:



Una vez escrito el árbol de la consulta, debemos hallar el árbol canónico equivalente.
Realizamos las selecciones lo antes posible. Se aplican las siguientes reglas:

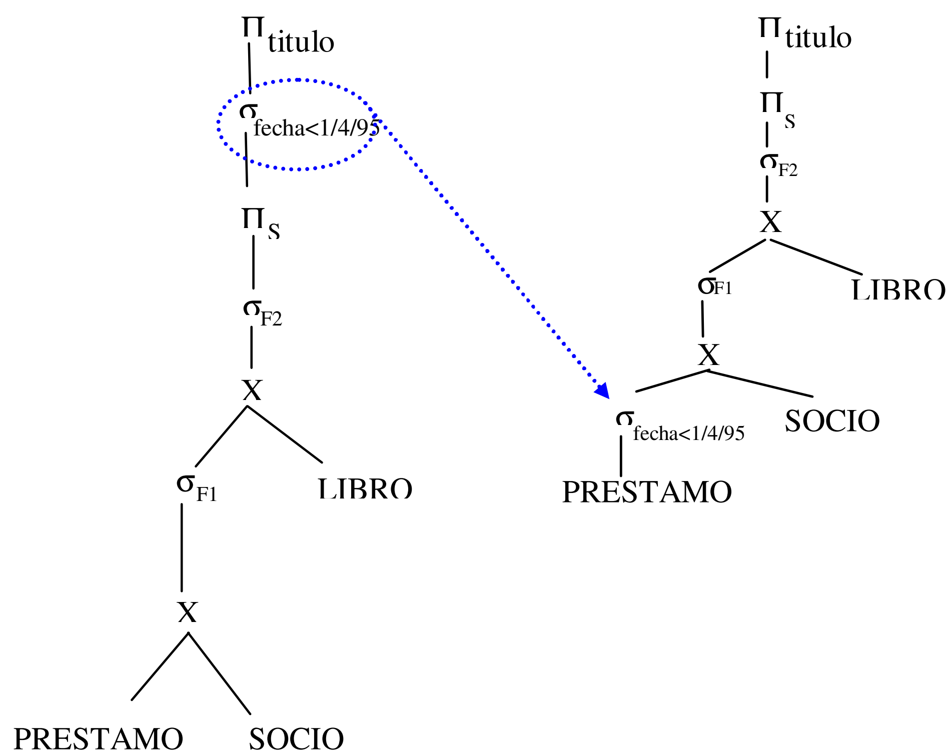
Regla 1: Cascada de Selecciones (transformo F en F1 y F2),
Regla 2: Conmutatividad de la Selección (primero selecciono con F2 y luego con F1)
Regla 6: Conmutatividad de la selección con el producto cartesiano (la selección con F1 conmuto con el producto cartesiano)



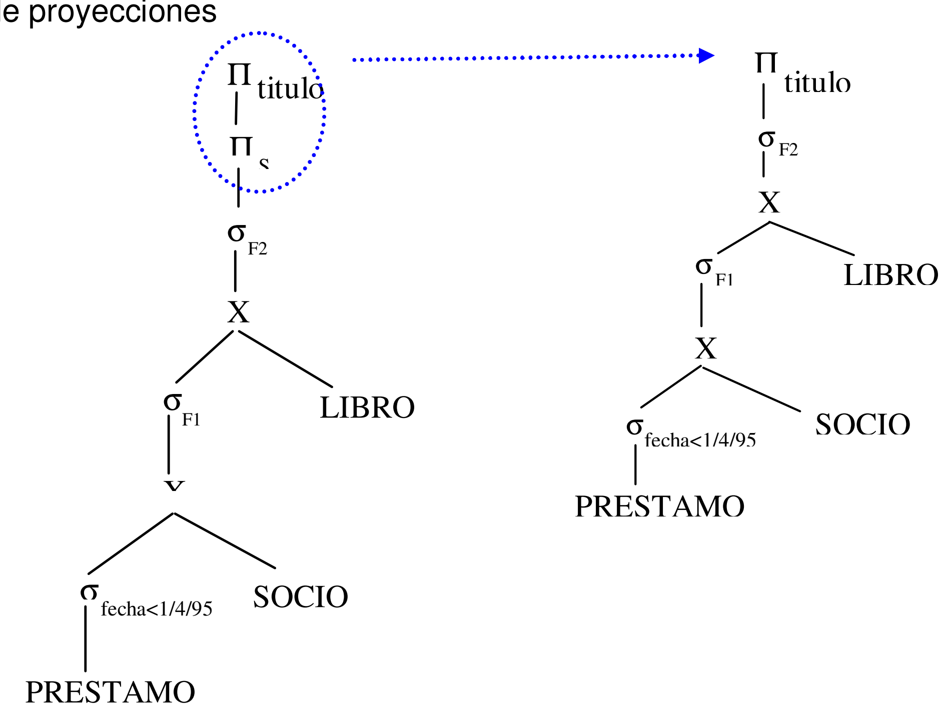
F1= PRESTAMO.nroSocio=SOCIO.nroSocio
F2= PRESTAMO.nroInv=LIBRO.nroInv

Para llevar lo más abajo posible la selección con la fecha, se aplican las siguientes reglas:

Regla 4: Conmutatividad de la selección con la proyección
Regla 2: Conmutatividad de la selección
Regla 6: Conmutatividad de la selección con el producto cartesiano
Regla 2: Conmutatividad de la selección
Regla 6: Conmutatividad de la selección con el producto cartesiano

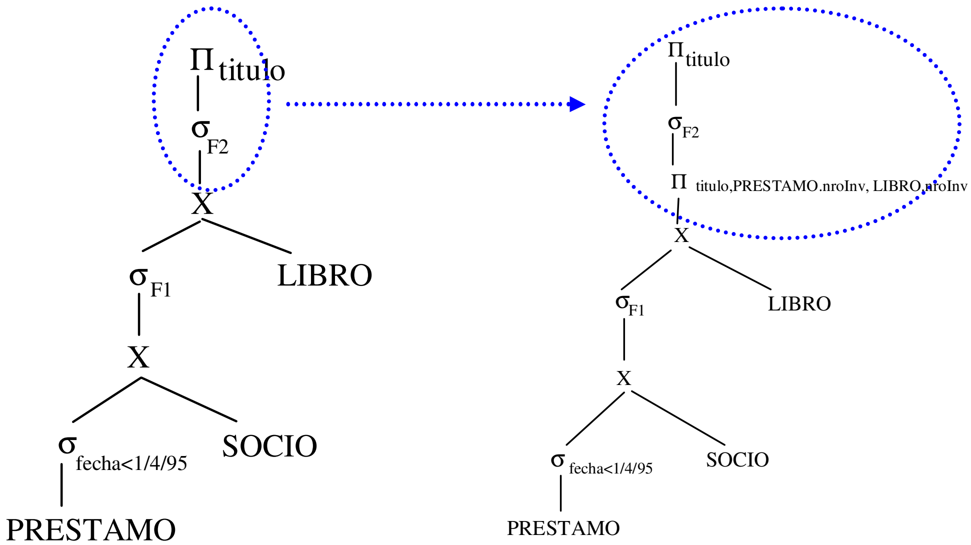


Ahora se desean eliminar proyecciones innecesarias, entonces se aplica la Regla 3: Cascada de proyecciones

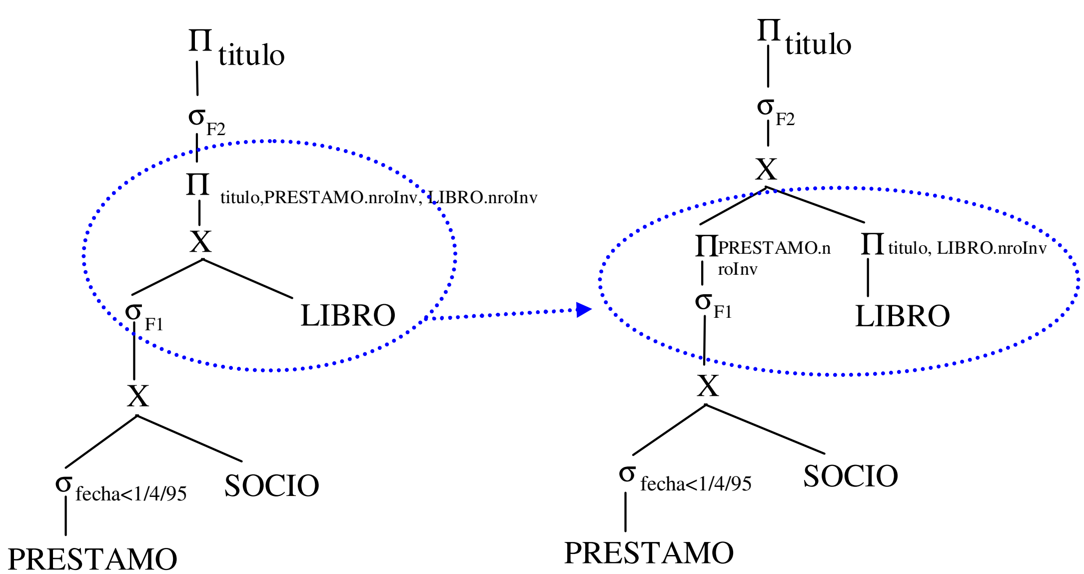


Supongamos que se decide agregar una proyección y no se hace lo más restrictivamente posible, sino que se va a agregar en un lugar de la consulta que requerirá realizar pasos de optimización adicionales.

Por el paso e del algoritmo propuesto se agregan proyecciones

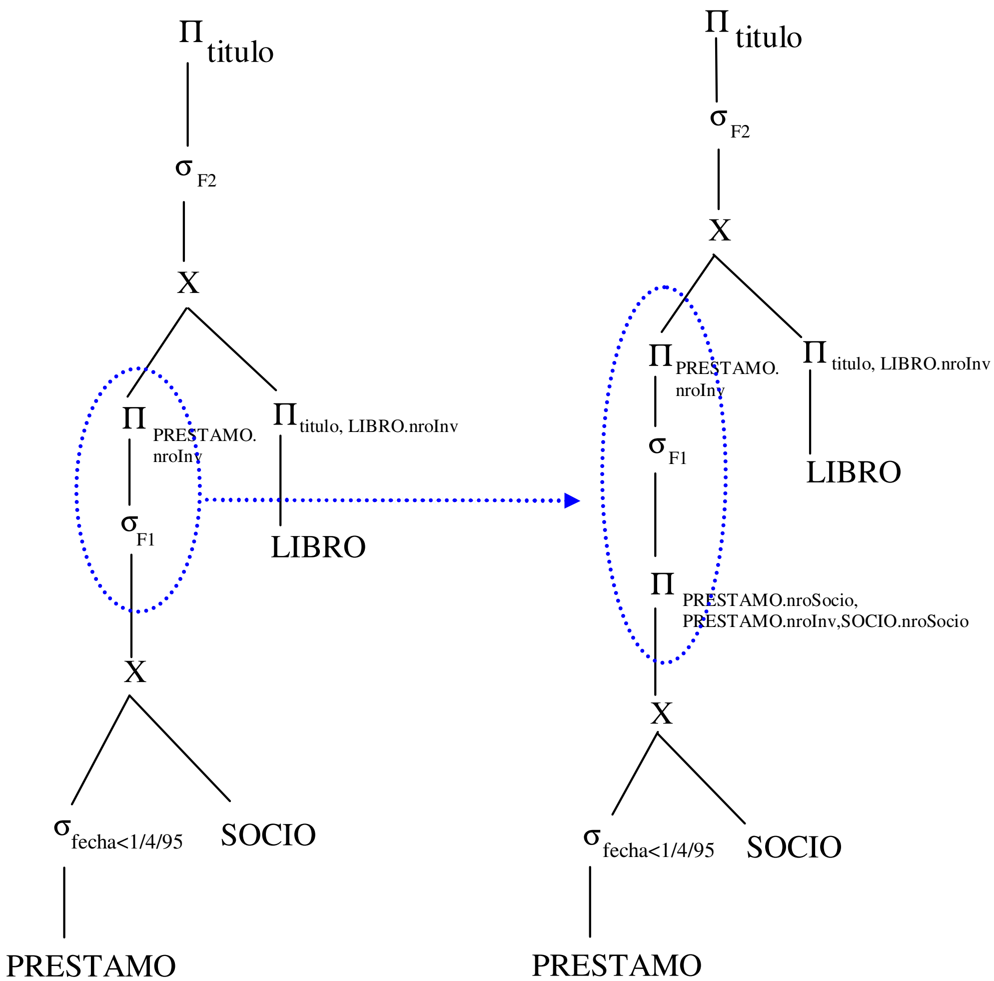


Para reducir el espacio de búsqueda de la consulta, se aplica la Regla 7: Conmutatividad de la proyección con el producto cartesiano

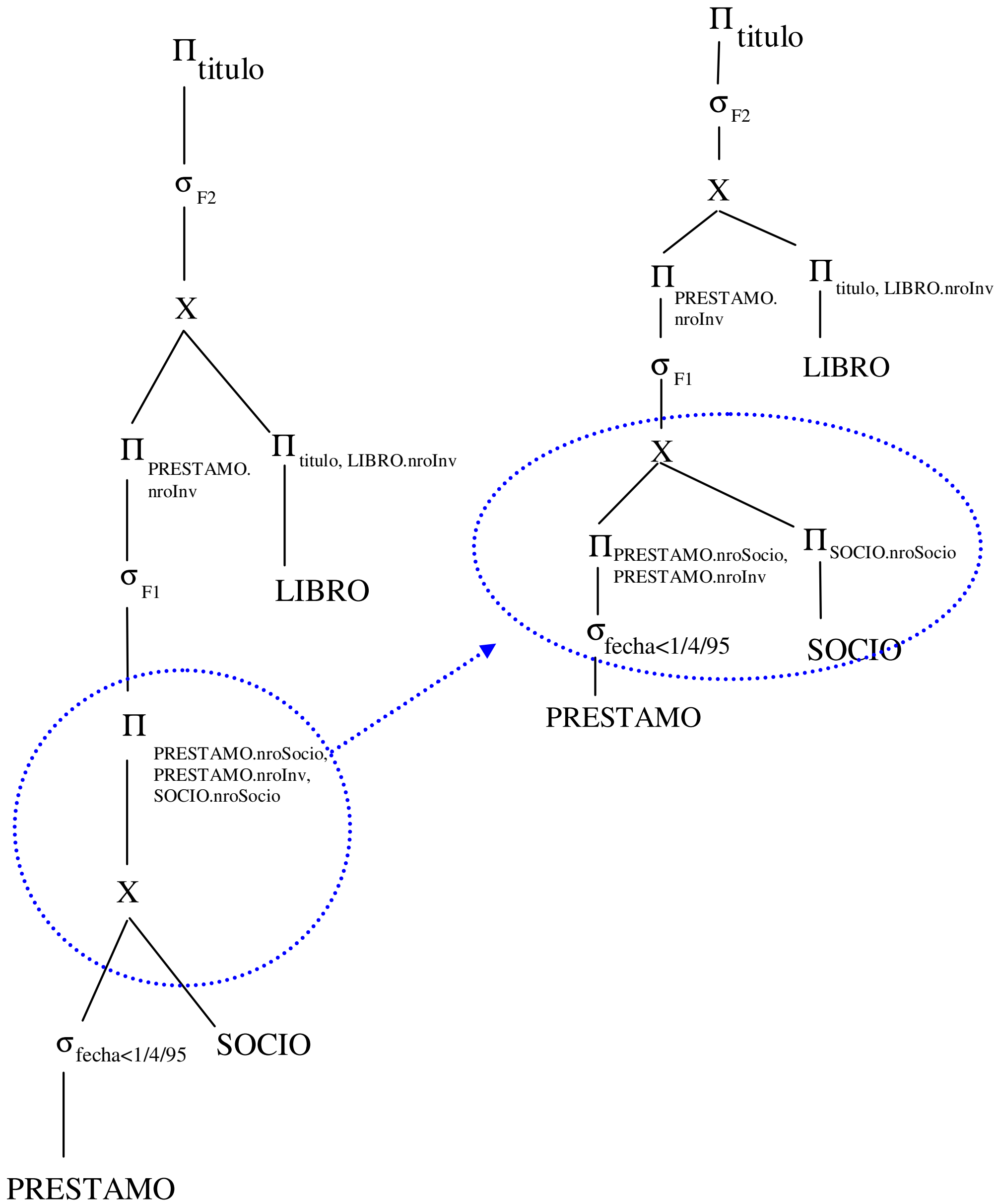


Supongamos que se decide agregar una proyección y no se hace lo más restrictivamente posible, sino que se va a agregar en un lugar de la consulta que requerirá realizar pasos de optimización adicionales.

Por el paso e del algoritmo propuesto se agregan proyecciones



Para reducir el espacio de búsqueda de la consulta, se aplica la Regla 7: Conmutatividad de la proyección con el producto cartesiano



Luego de aplicar todas las reglas descriptas y pasos del algoritmo, se obtiene el árbol canónico.

La consulta canónica es la siguiente:

π_{titulo}

$(\sigma_{F2}$

$((\pi_{PRESTAMO.nroInv}$

$(\sigma_{F1} ($

$(\pi_{nroSocio,nroInv}$

$(\sigma_{fecha<1/4/95} PRESTAMO)$

$)$

$X\ (\pi_{nroSocio}\ SOCIO)$

$)$

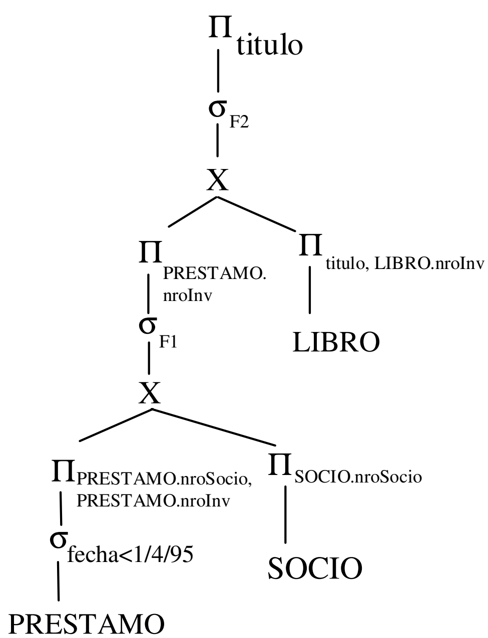
$))$

$X\ (\pi_{titulo,nroInv}\ LIBRO)$

$))$

F1= PRESTAMO.nroSocio=SOCIO.nroSocio
F2= PRESTAMO.nroInv=LIBRO.nroInv

Mientras que el árbol canónico es el siguiente:



De esta manera, se finaliza con el proceso propuesto.

Ejemplo 2:

Suponga que tiene el siguiente esquema que representa un conjunto de departamentos con sus respectivos proyectos.

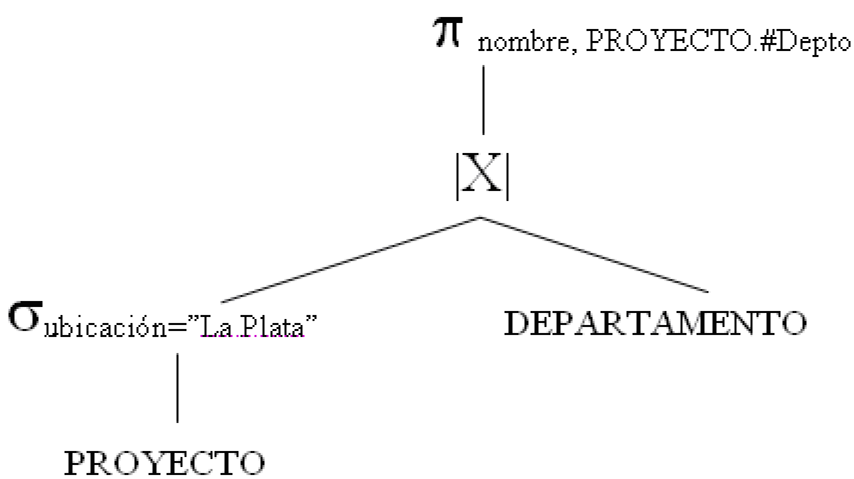
DEPARTAMENTO (#Depto, nombre, fechaCreación)
PROYECTO (#Proy, nomProy, ubicación, #Depto)

La consulta que analizaremos será: ***Obtener el nombre y el número de los departamentos que poseen proyectos ubicados en la ciudad de La Plata***.

Proponemos la siguiente expresión en álgebra relacional:

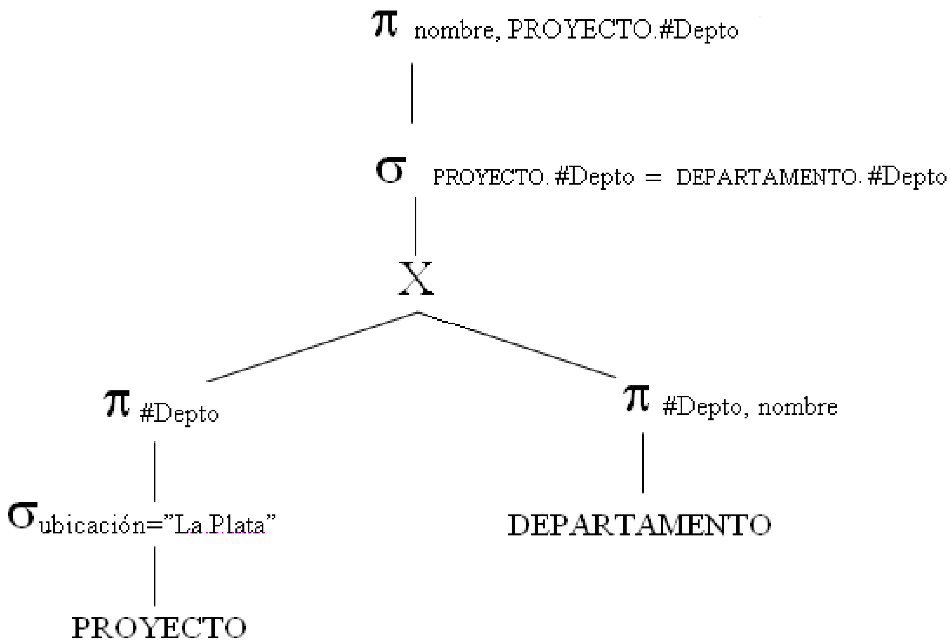
$$\pi_{\text{nombre, PROYECTO.\#Depto}}(\sigma_{\text{ubicación="La Plata"}} (\text{PROYECTO}) \;|X|\; \text{DEPARTAMENTO})$$

El árbol de la consulta es el siguiente:



Una vez escrito el árbol de la consulta, debemos hallar el árbol canónico equivalente.

Por las características del árbol de la consulta, lo que podemos hacer es aplicar el paso d del algoritmo propuesto. Transformar los productos naturales en productos cartesianos con selecciones cuyas condiciones representan la condición del producto natural. Se aplica además el paso e, es decir se agregan nuevas proyecciones.



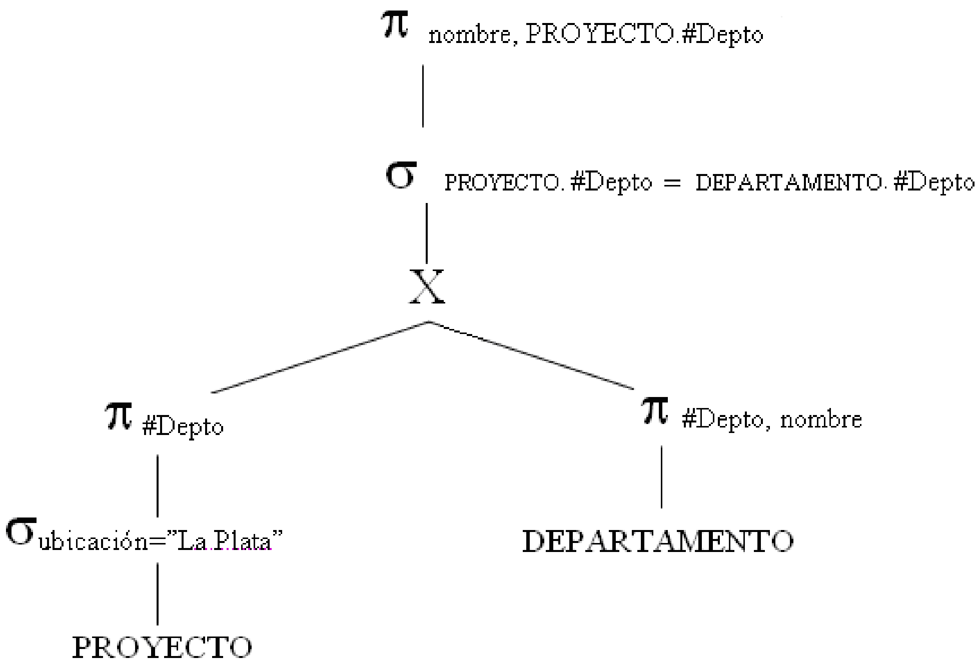
Nota: Al momento de realizar la operación de selección se debe desambigüar la tabla de origen del atributo #Depto.

Luego de aplicar todos los pasos del algoritmo descripto, se obtiene el árbol canónico.

La consulta canónica es la siguiente:

$\pi_{\text{nombre, PROYECTO.\#Depto}} (\sigma_{\text{PROYECTO.\#Depto=DEPARTAMENTO.\#Depto}} (\pi_{\text{\#Depto}} (\sigma_{\text{ubicación="La Plata"}}(\text{PROYECTO})) \text{ X } \pi_{\text{\#Depto,nombre}} (\text{DEPARTAMENTO})))$

Mientras que el árbol canónico es el siguiente:



De esta manera, se finaliza con el proceso propuesto.