Foundations of Databases

Query Processing and Optimization

Free University of Bozen – Bolzano, 2004–2005

Thomas Eiter

Institut für Informationssysteme

Arbeitsbereich Wissensbasierte Systeme (184/3)

Technische Universität Wien

`http://www.kr.tuwien.ac.at/staff/eiter`

(Most of the slides based on material by Leonid Libkin)

Query Processing and Optimization

- *Query optimization*: finding a good way to evaluate a query
- Queries are declarative, and can be translated into procedural languages in more than one way
- Hence one has to choose the best (or at least good) procedural query
- This happens in the context of *query processing*
- A query processor turns queries and updates into sequences of operations on the database

Query processing and optimization stages

- Which relational algebra expression, equivalent to a given declarative query, will lead to the most efficient algorithm?
- For each algebraic operator, what algorithm do we use to compute that operator?
- How do operations pass data (main memory buffer, disk buffer)?

Issues:

- Finding equivalent relational algebra expressions (“query plans”)
- Assessing efficiency of their evaluation: We need to know how data is stored, and how it is accessed, etc.

Use general guidelines and statistics information

Overview of query processing

- Start with a declarative query:

```
SELECT R.A, S.B, T.E
FROM R,S,T
WHERE R.C=S.C AND S.D=T.D AND R.A>5 AND S.B<3 AND T.D=T.E
```

- Translate into an algebra expression:

$$\pi_{R.A,S.B,T.E}(\sigma_{R.A>5 \wedge S.B<3 \wedge T.D=T.E}(R \bowtie S \bowtie T))$$

- Optimization step: rewrite to an equivalent but more efficient expression:

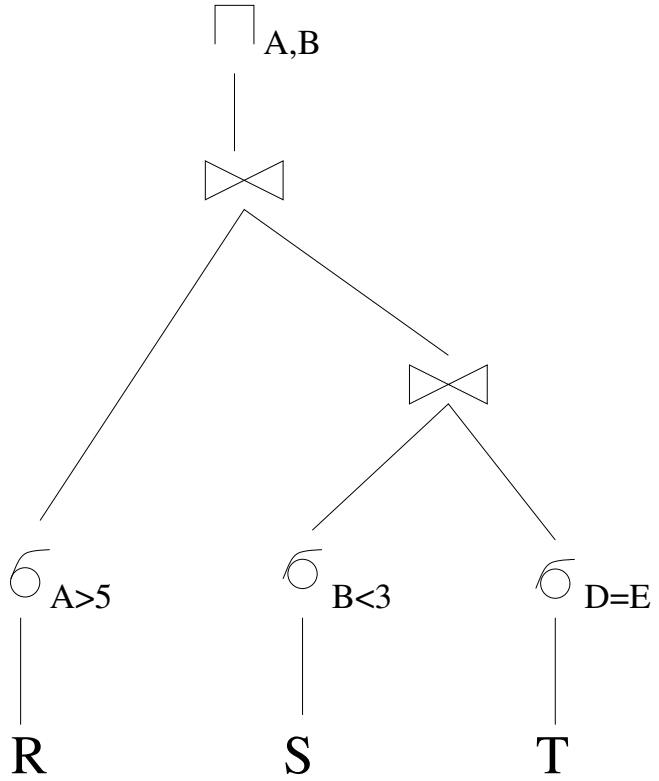
$$\pi_{R.A,S.B,T.E}(\sigma_{A>5}(R) \bowtie \sigma_{B<3}(S) \bowtie \sigma_{D=E}(T)))$$

- Why is it more efficient?

  Because selections are evaluated early, and joined relations are not as large as R, S, T.

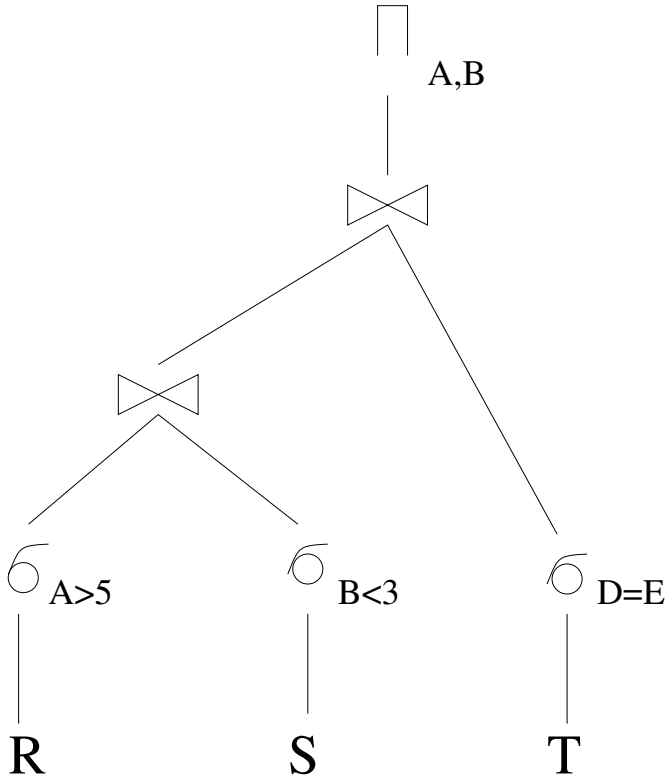
Overview of query processing cont'd

- Evaluating the optimized expression. Choices to make: order of joins.
- First *query plan* (out of two):



first join S and T, and then join the result with R.

- Alternative query plan:



  It first joins S, T, and then joins the result with R.

- Both query plans produce the same result.

- How to choose one?

Optimization by algebraic manipulations

- Given a relational algebra expression e, find another expression e' equivalent to e that is easier (faster) to evaluate.
- Basic question: Given two relational algebra expressions e_1, e_2, are they equivalent?
- This is the same as asking if an expression e always produces the empty answer:

$$e_1 = e_2 \quad \Leftrightarrow \quad e_1 - e_2 = \emptyset \text{ and } e_2 - e_1 = \emptyset$$

- Problem: testing $e = \emptyset$ is undecidable for relational algebra expressions.
- Good news:

  We can still list some useful equalities, and

  It is decidable for very important classes of queries (SPJ queries)

Optimization by Algebraic Equivalences

Systematic way of query optimization: Apply equivalences

- $\bowtie$ and $\times$ are commutative and associative, hence applicable in any order
- Cascaded projections might simplified: If the attributes $A_1, \ldots, A_n$ are among $B_1, \ldots, B_m$, then

$$\pi_{A_1,\ldots,A_n}(\pi_{B_1,\ldots,B_m}(E)) \;=\; \pi_{A_1,\ldots,A_n}(E)$$

- Cascaded selections might be merged:

$$\sigma_{c_1}(\sigma_{c_2}(E)) \;=\; \sigma_{c_1 \wedge c_2}(E)$$

- Commuting selection with join. If c only involves attributes from E_1, then

$$\sigma_c(E_1 \bowtie E_2) \;=\; \sigma_c(E_1) \bowtie E_2$$

- etc

We will not treat this here.

Optimization of conjunctive queries

- Reminder:

  conjunctive queries

  = SPJR queries

  = simple SELECT-FROM-WHERE SQL queries

  (only AND and (in)equality in the WHERE clause)

- Extremely common, and thus special optimization techniques have been developed
- Reminder: for two relational algebra expressions e_1, e_2, $e_1 = e_2$ is undecidable.
- But for conjunctive queries, even $e_1 \subseteq e_2$ is decidable.
- Main goal of optimizing conjunctive queries: reduce the number of joins.

Optimization of conjunctive queries: an example

- Given a relation R with two attributes A, B
- Two SQL queries:

Q1

```
SELECT R1.B, R1.A
FROM R R1, R R2
WHERE R2.A=R1.B
```

Q2

```
SELECT R3.A, R1.A
FROM R R1, R R2, R R3
WHERE R1.B=R2.B AND R2.B=R3.A
```

- Are they equivalent?
- If they are, we saved one join operation.
- In relational algebra:

$$Q_1 = \pi_{2,1}(\sigma_{2=3}(R \times R))$$

$$Q_2 = \pi_{5,1}(\sigma_{2=4 \wedge 4=5}(R \times R \times R))$$

Optimization of conjunctive queries cont'd

- Are Q_1 and Q_2 equivalent?
- If they are, we cannot show it by using equivalences for relational algebra expression.
- Because: they don't decrease the number of $\bowtie$ or $\times$ operators, but Q_1 has 1 join, and Q_2 has 2.
- But Q_1 and Q_2 are equivalent. How can we show this?
- But representing queries as databases. This representation is very close to rule-based queries.

$$\begin{aligned} Q_1(x,y) &\text{ :– } R(y,x), R(x,z) \\ Q_2(x,y) &\text{ :– } R(y,x), R(w,x), R(x,u) \end{aligned}$$

Conjunctive queries into tableaux

- Tableau: representation of a conjunctive query as a database
- We first consider queries over a single relation

$$Q_1(x, y) \text{ :- } R(y, x), R(x, z)$$

$$Q_2(x, y) \text{ :- } R(y, x), R(w, x), R(x, u)$$

- Tableaux:

A	B	
y	x	
x	z	
x	y	← answer line

A	B	
y	x	
w	x	
x	u	
x	y	← answer line

- Variables in the answer line are called distinguished

Tableau homomorphisms

- A homomorphism of two tableaux $f : T_1 \rightarrow T_2$ is a mapping

$$f : \{\text{variables of } T_1\} \rightarrow \{\text{variables of } T_2\} \bigcup \{\text{constants}\}$$

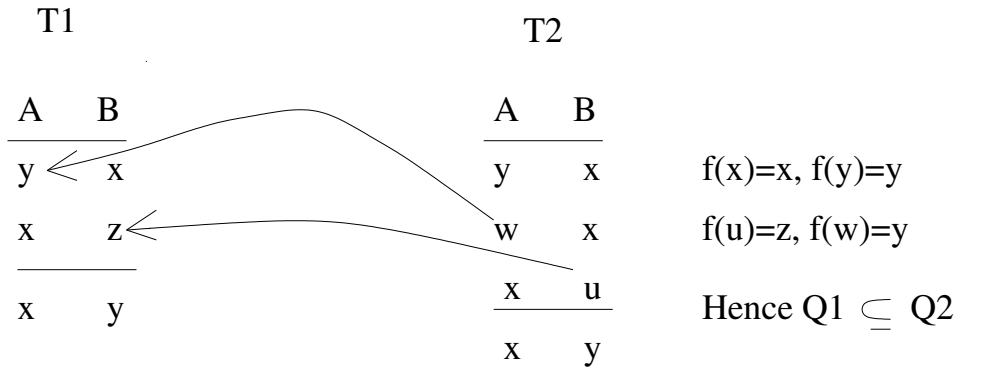
- For every distinguished x, $f(x) = x$
- For every row $x_1, \ldots, x_k$ in T_1, $f(x_1), \ldots, f(x_k)$ is a row of T_2

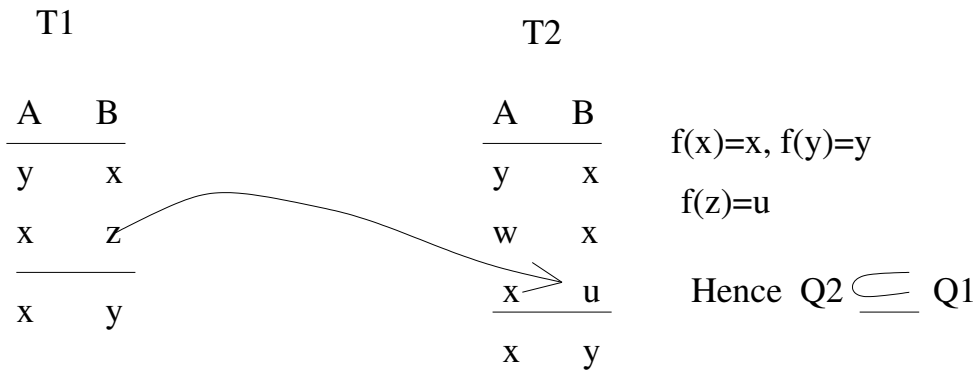
Defn. Query containment: $Q \subseteq Q'$ if $Q(\mathbf{I}) \subseteq Q'(\mathbf{I})$ for every database instance $\mathbf{I}$.

Homomorphism Theorem: Let Q, Q' be two conjunctive queries, and T, T' their tableaux. Then

$$Q \subseteq Q' \Leftrightarrow \text{there exists a homomorphism } f : T' \rightarrow T$$

Applying the Homomorphism Theorem: $Q_1 = Q_2$





Applying the Homomorphism Theorem: Complexity

- Given two conjunctive queries, how hard is it to test if $Q_1 = Q_2$?
- it is easy to transform them into tableaux, from either SPJ relational algebra queries, or SQL queries, or rule-based queries.
- However, a polynomial algorithm for deciding "equivalence" is unlikely to exist:

  Theorem. Given two tableaux, deciding the existence of a homomorphism between them is NP-complete.

- In practice, query expressions are small, and thus conjunctive query optimization is nonetheless feasible in polynomial time

Minimizing Conjunctive Queries

- Goal: Given a conjunctive query Q, find an equivalent conjunctive query Q' with the minimum number of joins.
- Assume Q is

$$Q(\vec{x}) \text{ :- } R_1(\vec{u}_1), \ldots, R_k(\vec{u}_k)$$

- Assume that there is an equivalent conjunctive query Q' of the form

$$Q'(\vec{x}) \text{ :- } S_1(\vec{v}_1), \ldots, S_l(\vec{v}_l), \qquad l < k.$$

- Then Q is equivalent to a query of the form

$$Q'(\vec{x}) \text{ :- } R_{i_1}(\vec{u}_{i_1}), \ldots, R_l(\vec{u}_{i_l})$$

- In other words, to minimize a conjunctive query, one has to delete some subqueries on the right of :-

Minimizing conjunctive queries cont'd

- Given a conjunctive query Q, transform it into a tableau T.
- Let Q' be a minimal conjunctive query equivalent to Q. Then its tableau T' is a subset of T.
- Minimization algorithm:

  $T' := T$;

  repeat until no change

  choose a row t in T';

  if there is a homomorphism $f : T' \to T' \setminus \{t\}$

  then $T' := T' \setminus \{t\}$

  end.

- Note: If a homomorphism $T' \to T' \setminus \{t\}$ exists, then T', $T' \setminus \{t\}$ define equivalent queries, as a homomorphism from $T' \setminus \{t\}$ to T' exists. (Why?)

Minimizing SPJ/conjunctive queries: example

- R with three attributes A, B, C
- SPJ query

$$Q \;=\; \pi_{AB}(\sigma_{B=4}(R)) \bowtie \pi_{BC}(\pi_{AB}(R) \bowtie \pi_{AC}(\sigma_{B=4}(R)))$$

- Translate into relational calculus:

$$\big(\exists z_1\, R(x,y,z_1) \wedge y = 4\big) \wedge \exists x_1 \Big(\big(\exists z_2\, R(x_1,y,z_2)\big) \wedge \big(\exists y_1\, R(x_1,y_1,z) \wedge y_1 = 4\big)\Big)$$

- Simplify, by substituting the constant, and putting quantifiers forward:

$$\exists x_1, z_1, z_2\, (R(x,4,z_1) \wedge R(x_1,4,z_2) \wedge R(x_1,4,z) \wedge y = 4)$$

- Conjunctive query:

$$Q(x,y,z) \;\text{:–}R(x,4,z_1), R(x_1,4,z_2), R(x_1,4,z), y = 4$$

Minimizing SPJ/conjunctive queries cont'd

- Tableau T:

A	B	C
x	4	z_1
x_1	4	z_2
x_1	4	z
x	4	z

- Minimization, step 1: Is there a homomorphism from T to

A	B	C
x_1	4	z_2
x_1	4	z
x	4	z

- Answer: No. For any homomorphism f, $f(x) = x$ (why?), thus the image of the first row is not in the small tableau.

Minimizing SPJ/conjunctive queries cont'd

- Step 2: Is T equivalent to

A	B	C
x	4	z_1
x_1	4	z
x	4	z

- Answer: Yes. Homomorphism f: $f(z_2) = z$, all other variables stay the same.
- The new tableau is not equivalent to

A	B	C
x	4	z_1
x	4	z

or

A	B	C
x_1	4	z
x	4	z

- Because $f(x) = x$, $f(z) = z$, and the image of one of the rows is not present.

Minimizing SPJ/conjunctive queries cont'd

- Minimal tableau:

A	B	C
x	4	z_1
x_1	4	z
x	4	z

- Back to conjunctive query:

$$Q'(x, y, z) \text{ :- } R(x, y, z_1), R(x_1, y, z), y = 4$$

- An SPJ query:

$$\sigma_{B=4}(\pi_{AB}(R) \bowtie \pi_{BC}(R))$$

- Pushing selections:

$$\pi_{AB}(\sigma_{B=4}(R)) \bowtie \pi_{BC}(\sigma_{B=4}(R))$$

Review of the journey

- We started with

$$\pi_{AB}(\sigma_{B=4}(R)) \bowtie \pi_{BC}(\pi_{AB}(R) \bowtie \pi_{AC}(\sigma_{B=4}(R)))$$

- Translated into a conjunctive query
- Built a tableau and minimized it
- Translated back into conjunctive query and SPJ query
- Applied algebraic equivalences and obtained

$$\pi_{AB}(\sigma_{B=4}(R)) \ \bowtie \ \pi_{BC}(\sigma_{B=4}(R))$$

- Savings: one join.

All minimizations are equivalent

- Let Q be a conjunctive query, and Q_1, Q_2 two conjunctive queries equivalent to Q
- Assume that Q_1 and Q_2 are both minimal, and let T_1 and T_2 be their tableaux.
- Then T_1 and T_2 are isomorphic; that is, T_2 can be obtained from T_1 by renaming of variables.
- That is, all minimizations are equivalent.
- In particular, in the minimization algorithm, the order in which rows are considered, is irrelevant.

Equivalence of conjunctive queries: multiple relations

- So far we assumed that there is only one relation R, but what if there are many?
- Construct tableaux as before:

$$Q(x, y) \text{:–} B(x, y), R(y, z), R(y, w), R(w, y)$$

- Tableau:

B:

A	B
x	y

R:

A	B
y	z
y	w
w	y

x	y

- Formally, a tableau is just a database where variables can appear in tuples, plus a set of distinguished variables.

Tableaux and multiple relations

- Given two tableaux T_1 and T_2 over the same set of relations, and the same distinguished variables, a homomorphism $f : T_1 \rightarrow T_2$ is a mapping

$$f : \{\text{variables of } T_1\} \rightarrow \{\text{variables of } T_2\}$$

such that

- $f(x) = x$ for every distinguished variable, and
- for each row $\vec{t}$ in R in T_1, $f(\vec{t})$ is in R in T_2.

- **Homomorphism theorem**: let Q_1 and Q_2 be conjunctive queries, and T_1, T_2 their tableaux. Then

$$Q_2 \subseteq Q_1 \Leftrightarrow \text{there exists a homomorphism } f : T_1 \rightarrow T_2$$

Minimization with multiple relations

- The algorithm is the same as before, but one has to try rows in different relations. Consider homomorphism $f(z) = w$, and f is the identity for other variables. Applying this to the tableau for Q yields

B:

A	B
x	y

R:

A	B
y	w
w	y

x	y

- This can't be further reduced, as for any homomorphism f, $f(x) = x$, $f(y) = y$.
- Thus Q is equivalent to

$$Q'(x, y) \text{ :- } B(x, y), R(y, w), R(w, y)$$

- One join is eliminated.

Conjunctive Queries with Equalities and Inequalities

- Equality / Inequality atoms $x = y$, $x = a$, $x \neq z$, etc
- Let T_1, T_2 be the tableaux of the parts of conjunctive queries Q_1 and Q_2 with ordinary relations
- $Q_2 \subseteq Q_1$ if there exists a homomorphism $f : T_1 \rightarrow T_2$ such that for each (in)equality atom $t_1 \theta t_2$ in Q_1, $f(t_1) \theta f(t_2)$ is logically implied by the equality and inequality atoms in Q_2
- The converse does not hold in general
- It holds under certain conditions, though

Note: Deciding whether a set of equality / inequality atoms A logically implies an equality / inequality atom is easy.

Query optimization and integrity constraints

- Additional equivalences can be inferred if integrity constraints are known
- Example: Let R have attributes A, B, C. Assume that R satisfies $A \rightarrow B$.
- Then it holds that

$$R \;=\; \pi_{AB}(R) \bowtie \pi_{AC}(R)$$

- Tableaux can help with these optimizations!
- $\pi_{AB}(R) \bowtie \pi_{AC}(R)$ as a conjunctive query:

$$Q(x, y, z) \text{:–} R(x, y, z_1), R(x, y_1, z)$$

Query optimization and integrity constraints cont'd

- Tableau:

A	B	C
x	y	z_1
x	y_1	z
x	y	z

- Using the FD $A \rightarrow B$ infer $y = y_1$
- Next, minimize the resulting tableau:

A	B	C
x	y	z_1
x	y	z
x	y	z

$\rightarrow$

A	B	C
x	y	z
x	y	z

- And this says that the query is equivalent to $Q'(x, y, z)$:–$R(x, y, z)$, that is, R.

Query optimization and integrity constraints cont'd

- General idea: simplify the tableau using functional dependencies and then minimize.
- Given: a conjunctive query Q, and a set of FDs F
- Algorithm:

  Step 1. Compute the tableau T for Q.

  Step 2. Apply algorithm CHASE(T, F).

  Step 3. Minimize output of CHASE(T, F).

  Step 4. Construct a query from the tableau produced in Step 3.

The CHASE

Assume that all FDs are of the form $X \rightarrow A$, where A is an attribute.

for each $X \rightarrow A$ in F do
  for each t_1, t_2 in T such that $t_1.X = t_2.X$ and $t_1.A \neq t_2.A$ do
    case $t_1.A, t_2.A$ of
    both nondistinguished $\Rightarrow$
      replace one by the other
    one distinguished, one nondistinguished $\Rightarrow$
      replace nondistinguished by distinguished
    one constant, one variable $\Rightarrow$
      replace variable by constant
    both constants $\Rightarrow$
      output $\emptyset$ and STOP
  end
end.

Query optimization and integrity constraints: example

- R is over A, B, C; F contains $B \to A$
- $Q = \pi_{BC}(\sigma_{A=4}(R)) \bowtie \pi_{AB}(R)$
- Q as a conjunctive query:

$$Q(x, y, z) \text{ :- } R(4, y, z), R(x, y, z_1)$$

- Tableau:

A	B	C
4	y	z
x	y	z_1
x	y	z

$\xrightarrow{\text{CHASE}}$

A	B	C
4	y	z
4	y	z_1
4	y	z

$\xrightarrow{\text{minimize}}$

A	B	C
4	y	z
4	y	z

- Final result: $Q(x, y, z)$:- $R(x, y, z), x = 4$, that is, $\sigma_{A=4}(R)$.

Query optimization and integrity constraints: example

- Same R and F; the query is:

$$Q \;=\; \pi_{BC}(\sigma_{A=4}(R)) \bowtie \pi_{AB}(\sigma_{A=5}(R))$$

- As a conjunctive query:

$$Q(x, y, z) \text{ :- } R(4, y, z), R(x, y, z_1), x = 5$$

- Tableau:

A	B	C
4	y	z
5	y	z_1
5	y	z

$\overset{\text{CHASE}}{\longrightarrow}$ $\emptyset$

- Final result: $\emptyset$
- This equivalence is not true *without* the FD $B \rightarrow A$

Query optimization and integrity constraints: example

- Sometimes simplifications are quite dramatic
- Same R, FD is $A \rightarrow B$, the query is

$$Q \;=\; \pi_{AB}(R) \bowtie \pi_{A}(\sigma_{B=4}(R)) \bowtie \pi_{AB}(\pi_{AC}(R) \bowtie \pi_{BC}(R))$$

- Convert into conjunctive query:

$$Q(x, y) \text{ :- } R(x, y, z_1), R(x, y_1, z), R(x_1, y, z), R(x, 4, z_2)$$

- Tableau:

A	B	C
x	y	z_1
x	y_1	z
x_1	y	z
x	4	z_2
x	y	

$\xrightarrow{\text{CHASE}}$

A	B	C
x	4	z_1
x	4	z
x_1	4	z
x	4	z_2
x	4	

$\xrightarrow{\text{minimize}}$

A	B	C
x	4	z
x	4	

Query optimization and integrity constraints: example cont'd

A	B	C
x	4	z
x	4	

  is translated into $Q(x, y)$:– $R(x, y, z), y = 4$

- or, equivalently $\pi_{AB}(\sigma_{B=4}(R))$.
- Thus,

  $$\pi_{AB}(R) \bowtie \pi_A(\sigma_{B=4}(R)) \bowtie \pi_{AB}(\pi_{AC}(R) \bowtie \pi_{BC}(R)) \;=\; \pi_{AB}(\sigma_{B=4}(R))$$

  in the presence of FD $A \rightarrow B$.
- Savings: 3 joins!
- This cannot be derived by algebraic manipulations, nor conjunctive query minimization without using CHASE.

Bibliography

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] H. Garcia-Molina, J. D. Ullman, and J. Widom. *Database Systems – The Complete Book*. Prentice Hall, 2002.

[3] D. Maier. *The Theory of Relational Databases*. Computer Science Press, Rockville, Md., 1983.

[4] J. D. Ullman. *Principles of Database and Knowledge Base Systems*. Computer Science Press, 1989.