Semantical and Syntactical Optimization

Knowledge Objectives

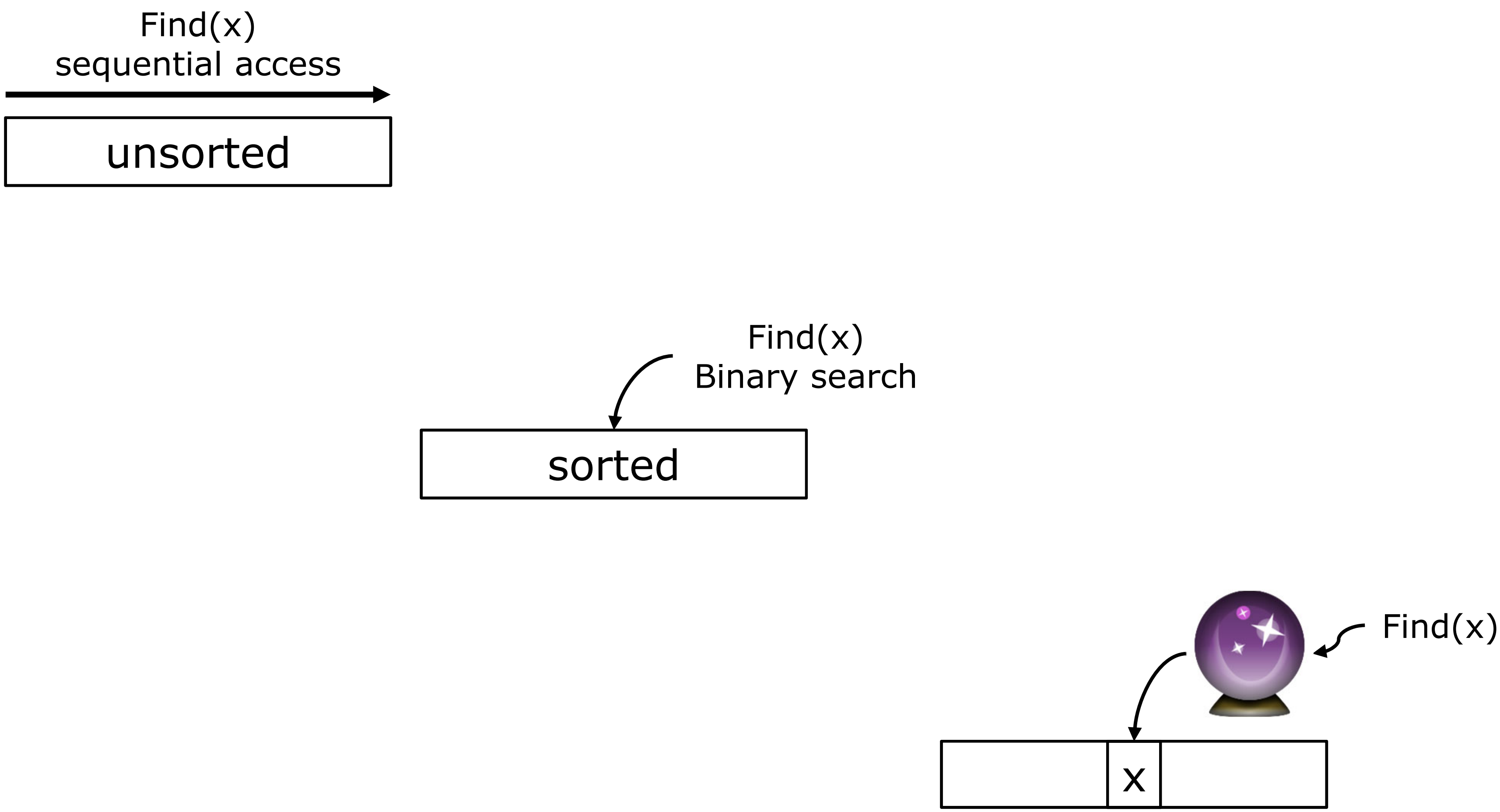
1. Enumerate the three phases of query optimization
2. Define semantic query optimization
3. Define syntactic query optimization
4. Explain two syntactic optimization heuristics

Understanding Objectives

1. Perform simple semantic optimizations over a query
2. Optimize a syntactic tree considering the heuristics about the order of operations

QUERY OPTIMIZATION

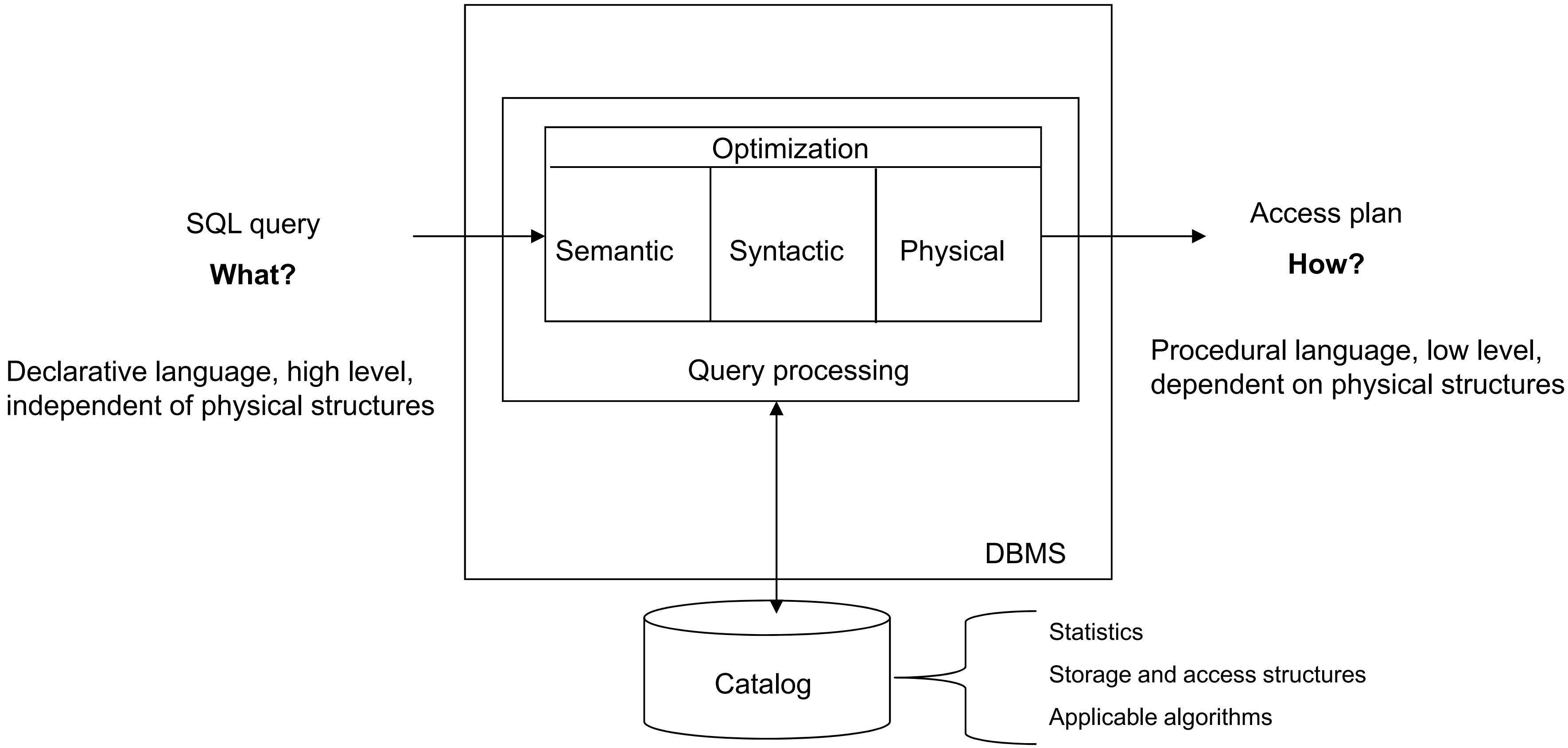
Cost efficiency



Preliminary considerations

- It is the last step in query processing
- Input: An SQL query over tables (or views), syntactically correct and authorized
- Output: An algorithm (access plan) that must be followed by the DBMS in order to get the result
- Goal: Minimize the use of resources
  - In general, a DBMS does not find the optimal access plan, but it obtains an approximation (in a reasonable time)

Architecture



SEMANTIC OPTIMIZATION

Semantic optimization

Consists of transforming the SQL sentence into an equivalent one with a lower cost, by considering:

- Integrity constraints
- Logics

Examples of semantic optimization

```
CREATE TABLE students (
    id CHAR(8) PRIMARY KEY,
    mark FLOAT CHECK (mark>3)
    );
```

```
SELECT *
FROM students
WHERE mark<2;

SELECT *
FROM students
WHERE mark<6 AND mark>8;

SELECT *
FROM students
WHERE mark<6
        AND mark<7;
```

Example of semantic optimization (ORACLE)

```
SELECT *
FROM employees e, departments d
WHERE e.dpt=d.code AND d.code>5;
```

⇩

```
SELECT *
FROM employees e, departments d
WHERE e.dpt=d.code AND d.code>5
      AND e.dpt>5;
```

Example of semantic optimization (DB2)

```
SELECT *
FROM students
WHERE mark=5 OR mark=6;
```

↓

```
SELECT *
FROM students
WHERE mark IN [5, 6];
```

SYNTACTIC OPTIMIZATION

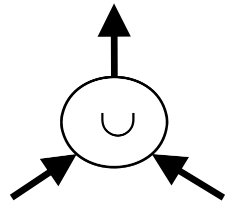
Syntactic optimization

Consists of translating the sentence from SQL into a sequence of algebraic operations in the form of syntactic tree, with minimum cost, by means of heuristics (there is more than one solution)

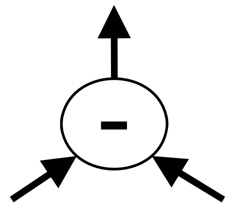
- Nodes
  - Leaves: Relations
  - Internal: Algebraic operations
  - Root: Result
- Edges
  - Denote direct usage

Internal nodes of the syntactic tree

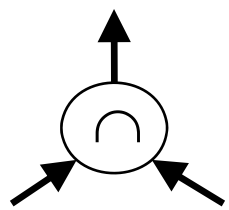
Union



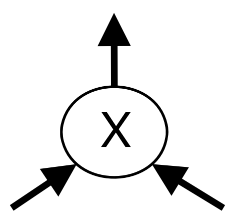
Difference



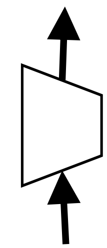
Intersection



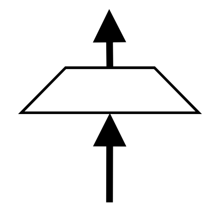
Cross product



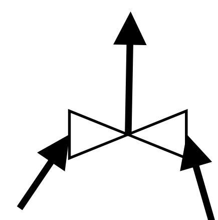
Selection



Projection



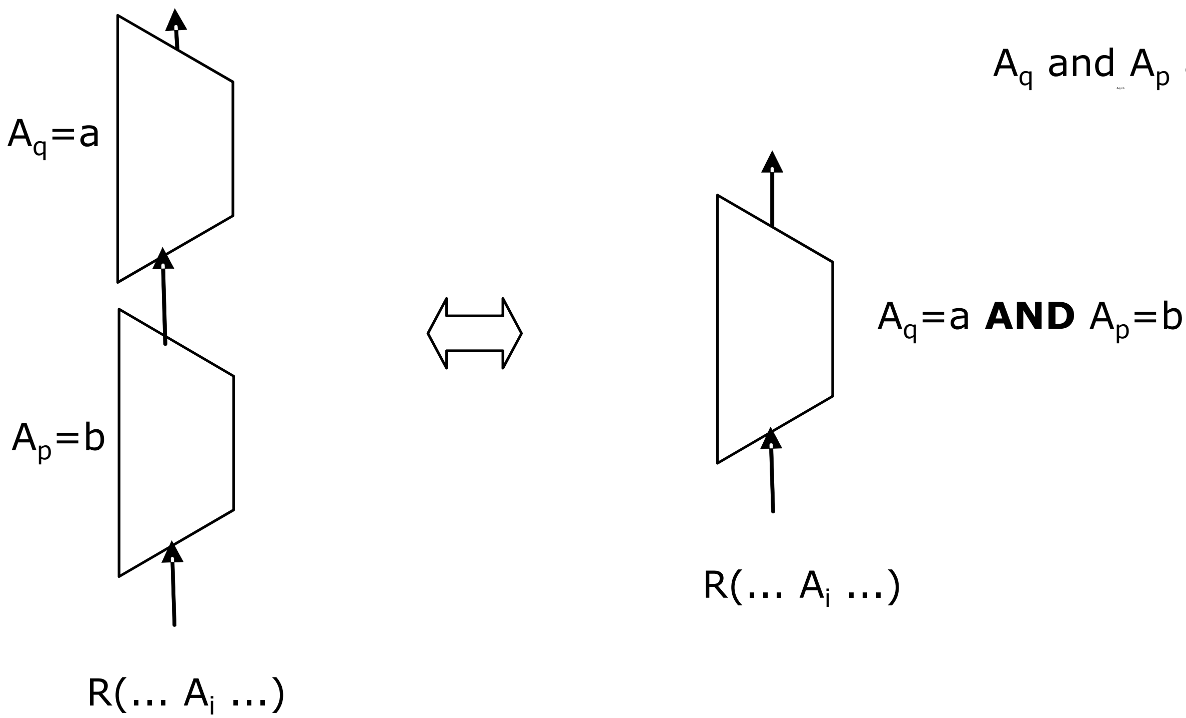
Join



Equivalence rules (I)

- Splitting/grouping selections

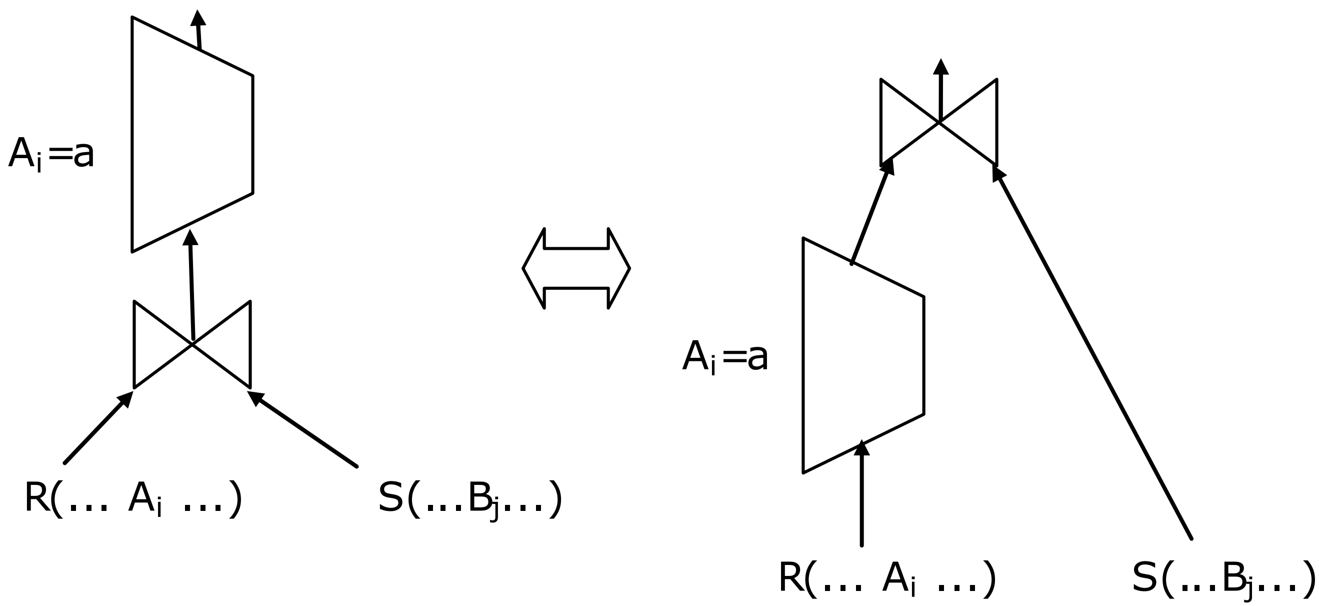
A_q and A_p are attributes of R



Elena Rodríguez

Equivalence rules (II)

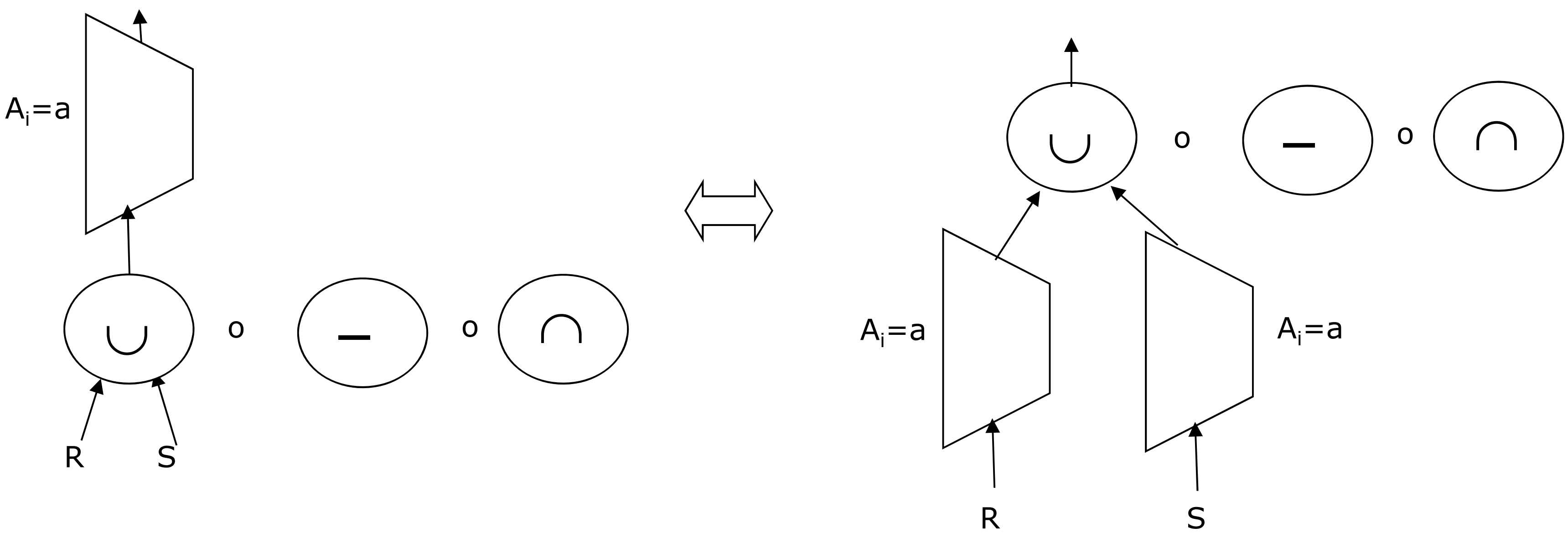
- Commuting the precedence of selection & join



Elena Rodríguez

Equivalence rules (III)

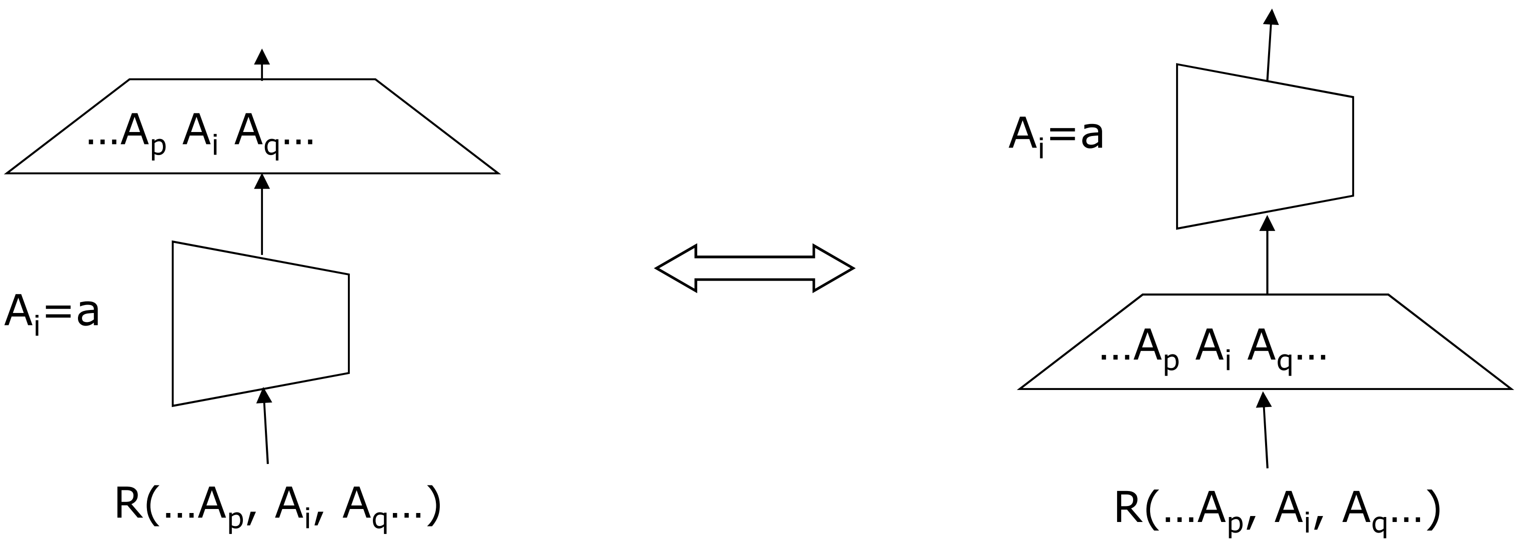
- Commuting the precedence of selection & set operation



Elena Rodríguez

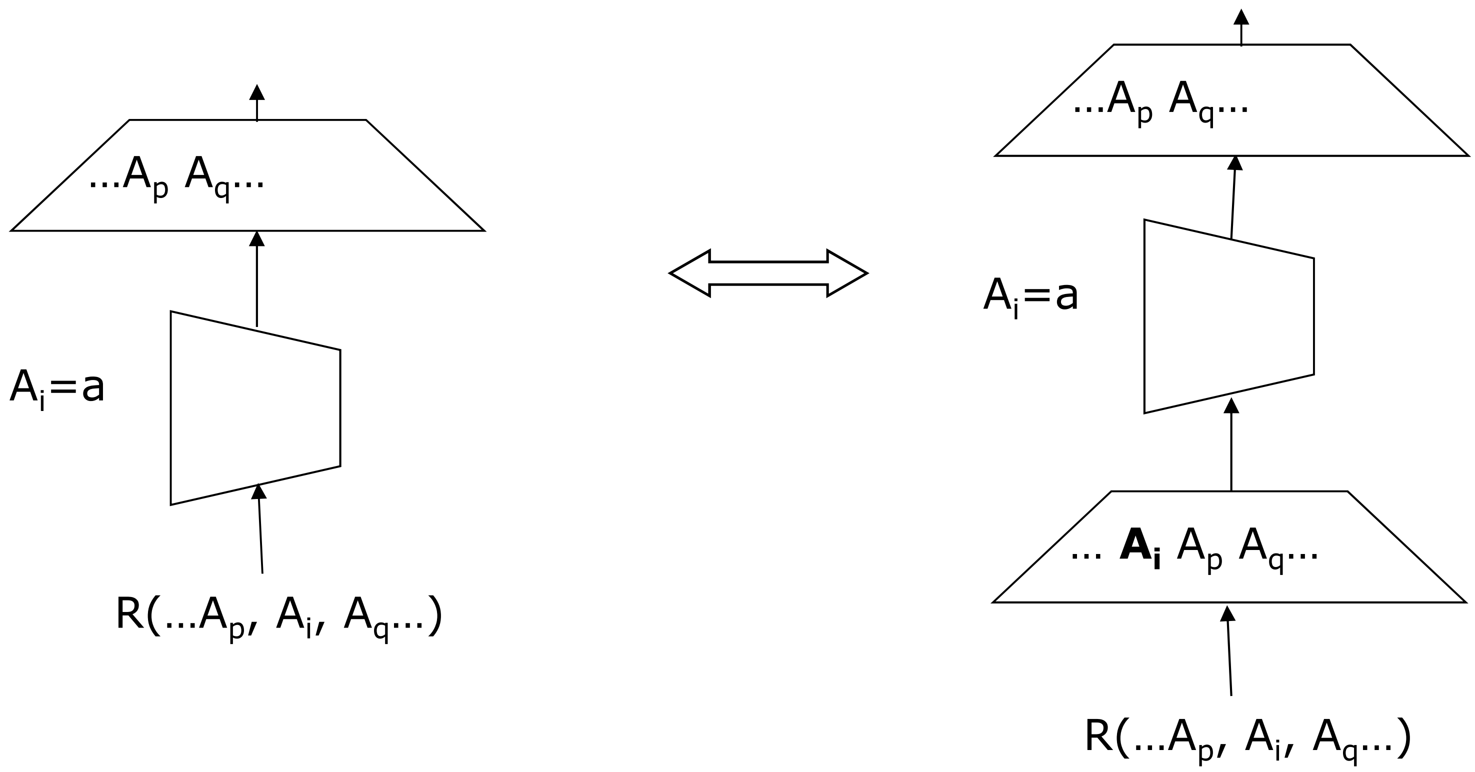
Equivalence rules (IV)

- Commuting the precedence of selection & projection ($A_i \in \{\ldots A_p, A_i, A_q \ldots\}$)



Equivalence rules (V)

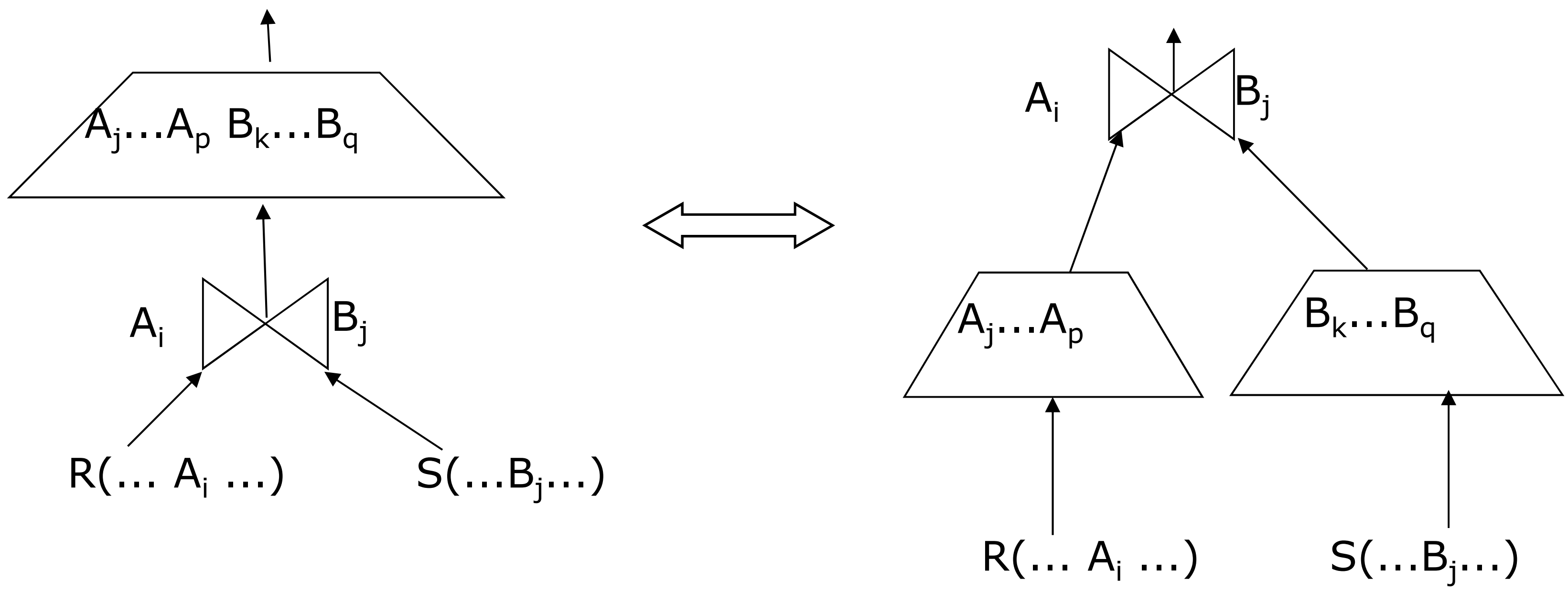
- Commuting the precedence of selection & projection ($A_i \notin \{\ldots A_p, A_q \ldots\}$)



Elena Rodríguez

Equivalence rules (VI)

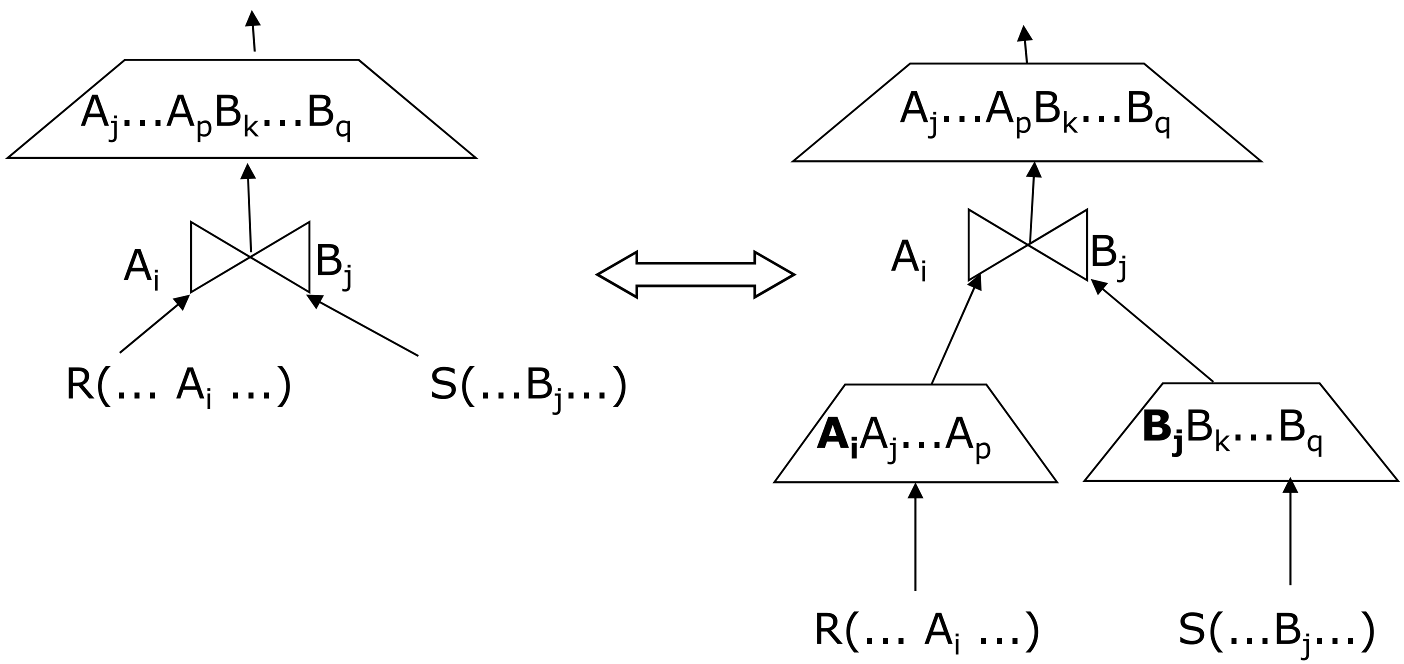
- Commuting the precedence of projection & join (A_i i $B_j \in \{A_j...A_p, B_k...B_q\}$)



Elena Rodríguez

Equivalence rules (VII)

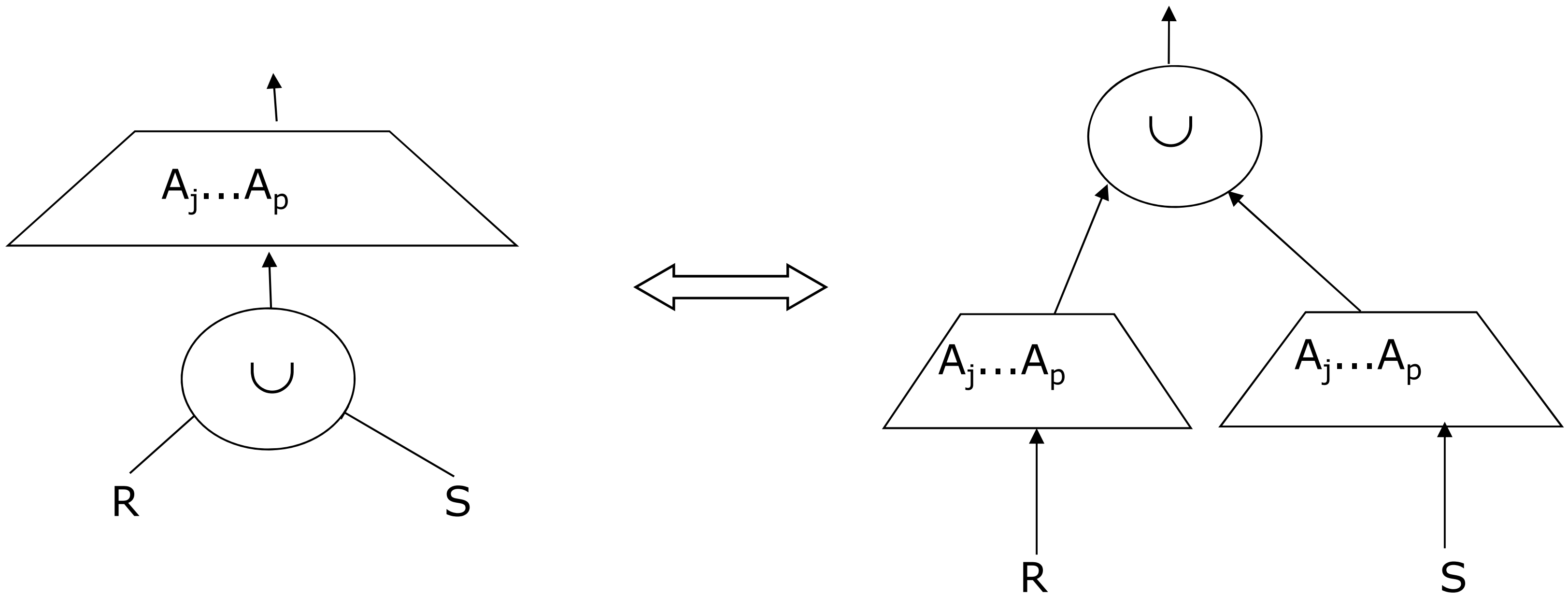
- Commuting the precedence of projection & join (A_i or B_j (or both) $\notin \{A_j \ldots A_p, B_k \ldots B_q\}$)



Elena Rodríguez

Equivalence rules (VIII)

- Commuting the precedence of projection & union



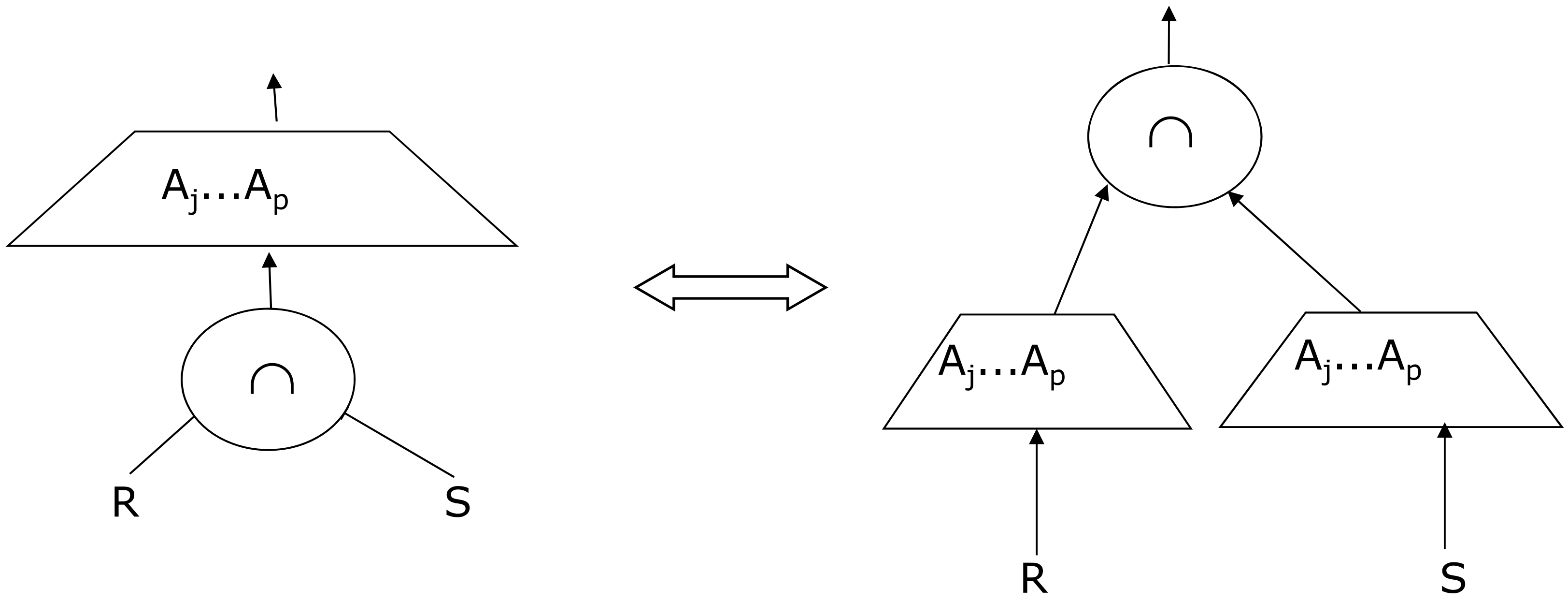
Important:

- Projection & intersection precedence cannot be freely commuted
- Projection & difference precedence cannot be freely commuted

Elena Rodríguez

Equivalence rules (IX)

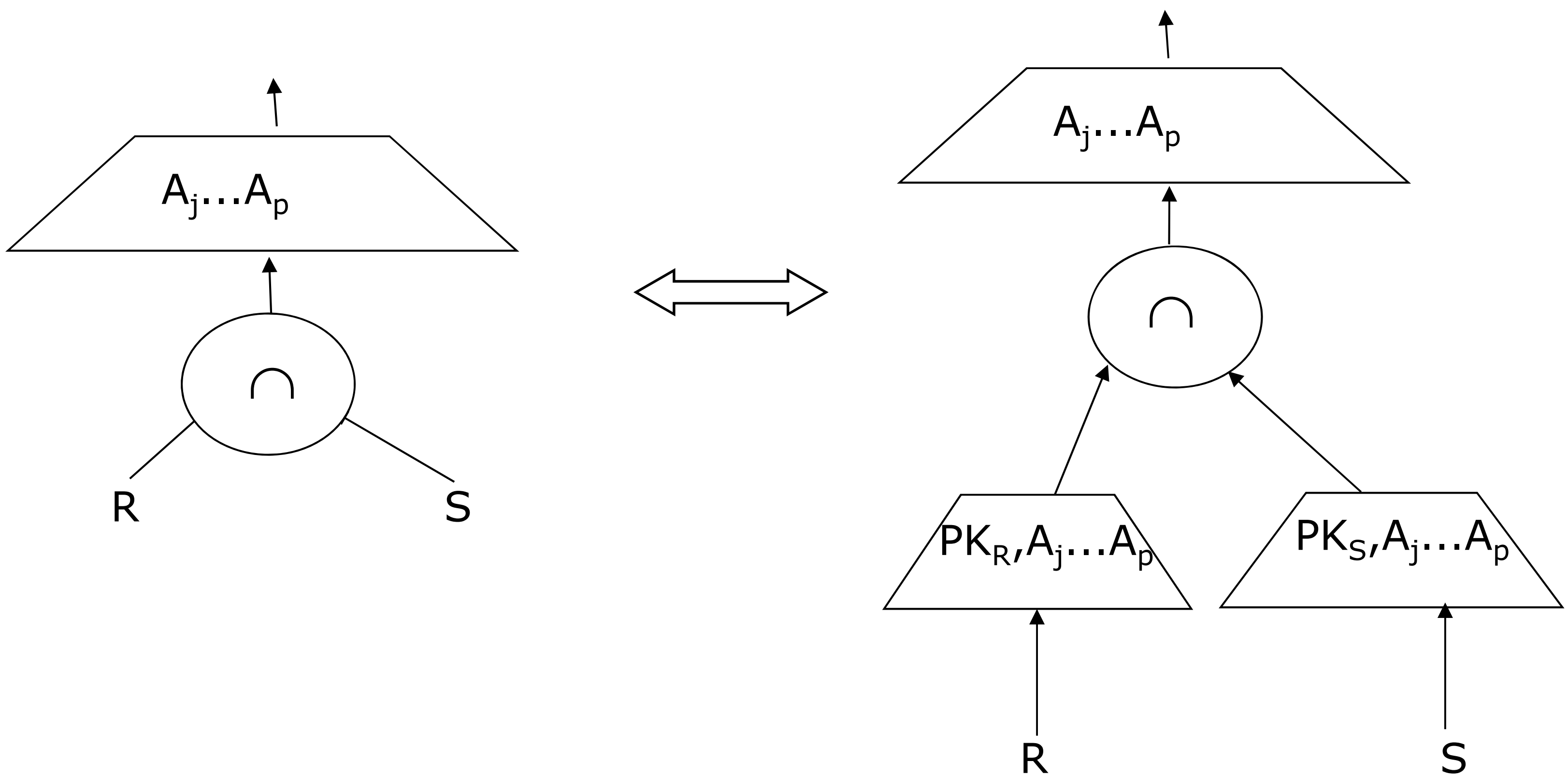
- Commuting the precedence of projection & intersection/difference (PK_R, $PK_S \in \{A_j,\ldots, A_p\}$)



Elena Rodríguez

Equivalence rules (X)

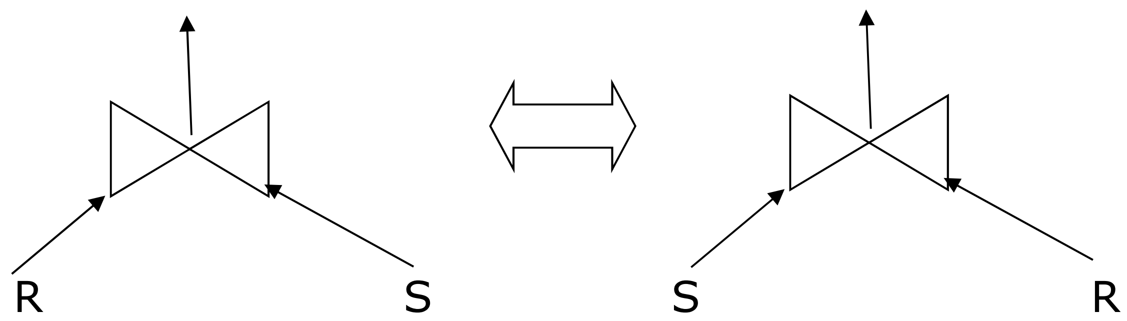
- Commuting the precedence of projection & intersection/difference (PK_R, $PK_S \notin \{A_j,\ldots, A_p\}$)



Elena Rodríguez

Equivalence rules (XI)

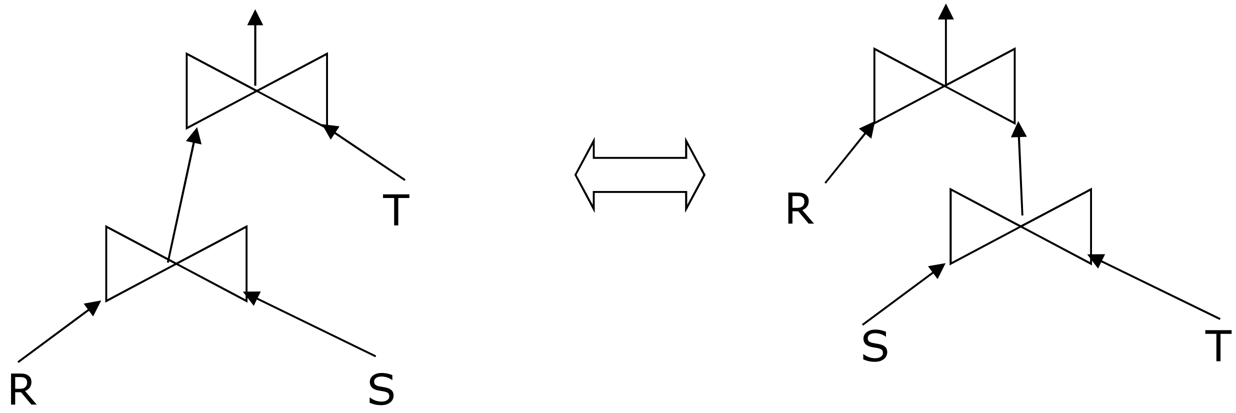
- Commuting join branches



Elena Rodríguez

Equivalence rules (XII)

- Associating join tables

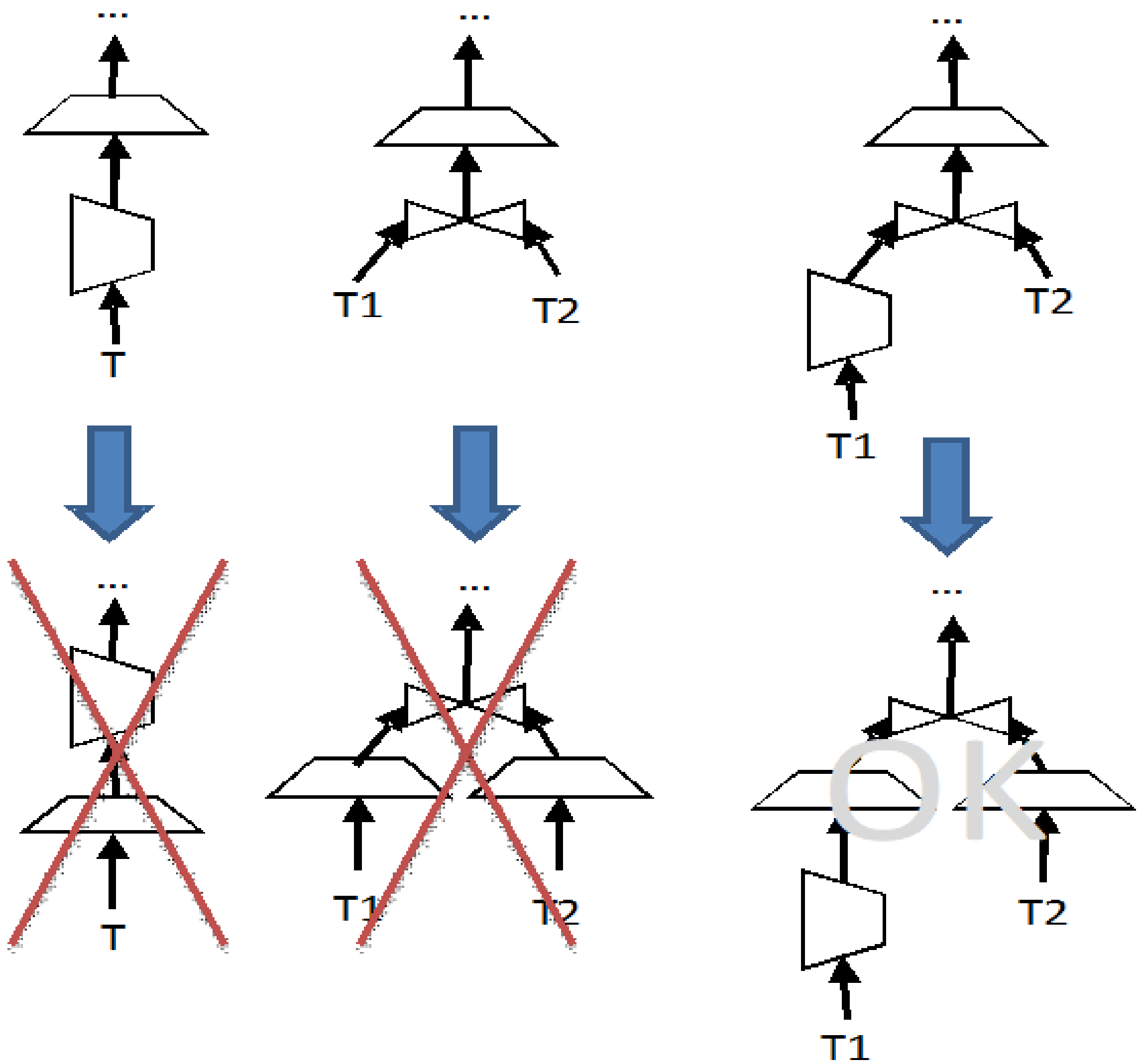


Elena Rodríguez

Transforming the syntactic tree

- Objective:
  - Reduce the size of intermediate nodes
- Steps:
  1. Split the selection predicates into simple clauses
  2. Lower selections as much as possible
  3. Group consecutive selections
     - Simplify them if possible
  4. Lower projections as much as posible
     - Do not leave them just on a table
  5. Group consecutive projections
     - Simplify them if possible

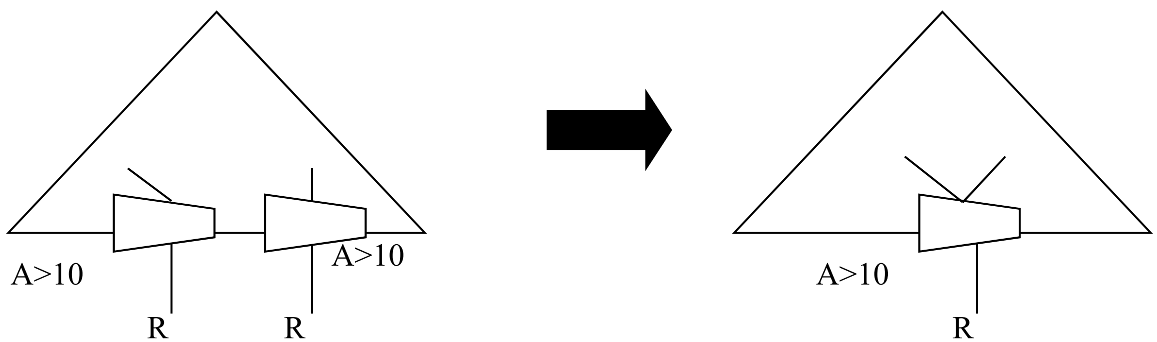
Projections over tables



Simplification of the syntactic tree

- Removal of disconnected components

- Fusion of common branches



- Removal of tautologies:

  $R \cap \emptyset = \emptyset$, $R - R = \emptyset$, $\emptyset - R = \emptyset$

  $R \cap R = R$, $R \cup R = R$, $R \cup \emptyset = R$, $R - \emptyset = R$

CLOSING

Summary

- Query optimization phases
  - **Semantic**
  - **Syntactic**
  - Physical
- Relational algebra equivalence rules
- Heuristics for the order of operations

Bibliography

- Y. Ioannidis. *Query Optimization*. ACM Computing Surveys, vol. 28, num. 1, March 1996
- R. Ramakrishnan and J. Gehrke. *Database Management Systems.* McGraw-Hill, 3rd Edition, 2003
- S. Lightstone, T. Teorey and T. Nadeau. *Physical Database Design*. Morgan Kaufmann, 2007