8 Query Optimization

- The success of relational database technology is largely due to the systems' ability to automatically find **evaluation plans** for declaratively specified queries.

- Given some (SQL) query Q, the system

  ① parses and analyzes Q,
  ② derives a **relational algebra** expression E that computes Q,
  ③ transforms and simplifies E, and
  ④ annotates the operators in E with **access methods** and **operator algorithms** to obtain an evaluation plan P.

- Discuss here: ③ + ④

From query to plan

- **Example**:

  List the airports from which flights operated by Swiss (airline code LX) fly to any german (DE) airport.

Airport :

code	*country*	*name*
'FRA'	'DE'	'Frankfurt'
'ZHR'	'CH'	'Zurich'
'MUC'	'DE'	'Munich'
⋮		

Flight :

from	*to*	*airline*
'FRA'	'ZHR'	'LX'
'ZHR'	'MUC'	'LX'
'FRA'	'MUC'	'US'
⋮		

SQL query Q:

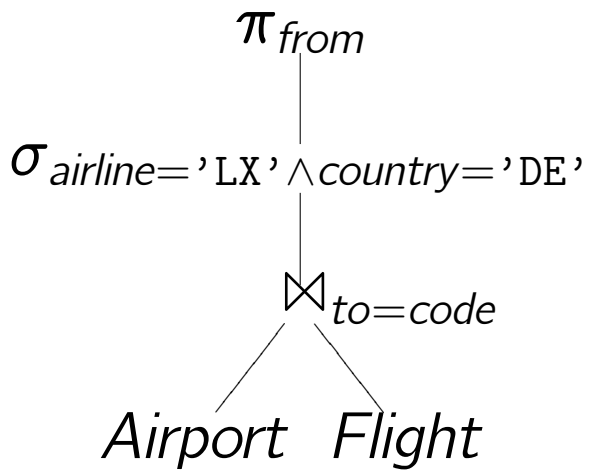
```
SELECT  f.from
  FROM  Flight f, Airport a
 WHERE  f.to = a.code AND f.airline = 'LX' AND a.country = 'DE'
```

From query to plan

- SQL query Q:

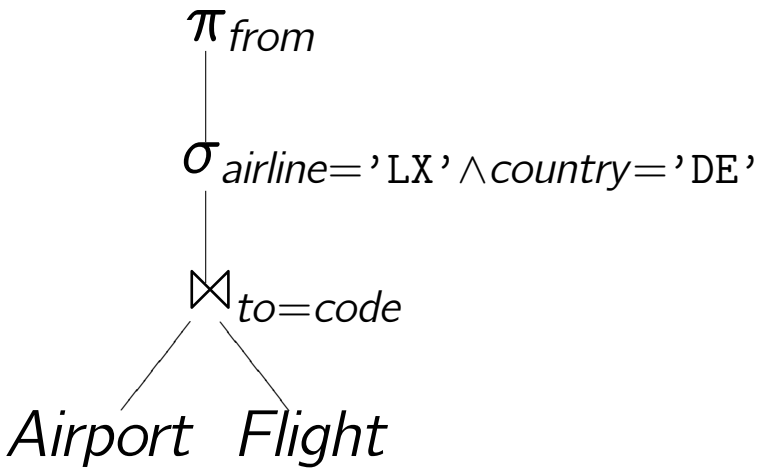
```
SELECT  f.from
  FROM  Flight f, Airport a
 WHERE  f.to = a.code AND f.airline = 'LX' AND a.country = 'DE'
```

- Relational algebra expression E that computes Q:

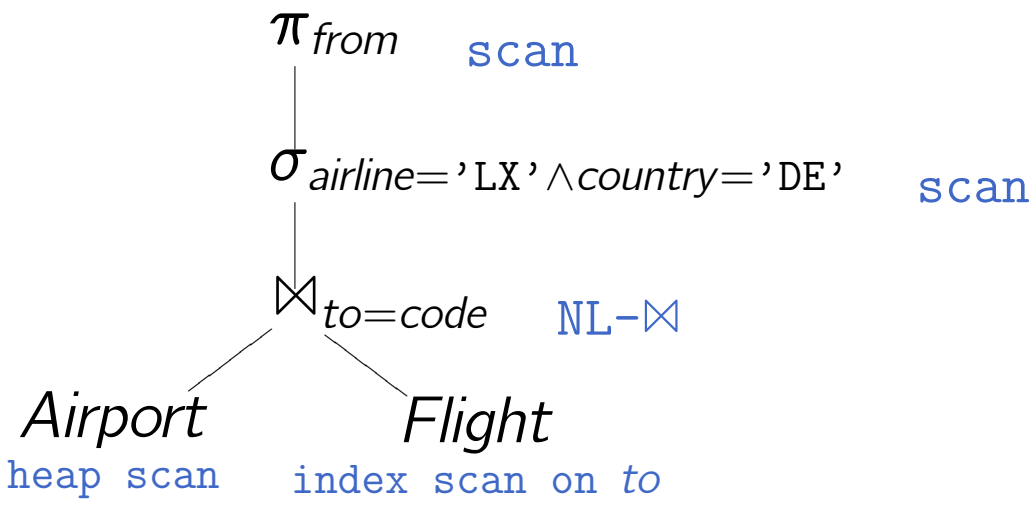


From query to plan

- Relational algebra expression E that computes Q:



- One (of **many**) plan P to evaluate Q:



Equivalence in the relational algebra

- Two relational algebra expressions E_1, E_2 are **equivalent** if—on every legal database instance—the two expressions generate the same set of tuples.
  - **Note**: the order of tuples is irrelevant 
- Such equivalences are denoted by **equivalence rules** of the form

$$E_1 \quad \equiv \quad E_2$$

(such a rule may be applied by the system in both directions $\rightarrow, \leftarrow$).

CASCADES OF SELECTION

$$\sigma_{\bigwedge_i F_i}(R) = \sigma_{F_1}(\ldots \sigma_{F_n}(R)\ldots)$$

Proof

$$\sigma_{\bigwedge_i F_i}(R) \overset{\text{def}}{=} \left\{ t \,\middle|\, t \in R \wedge \left(\bigwedge_i F_i\right) \right\} =$$

$$= \left\{ t \,\middle|\, t \in R \wedge \left(F_1 \wedge \left(\bigwedge_{i=2}^{n} F_i\right)\right) \right\} \overset{\wedge \text{ is assoc.}}{=}$$

$$= \left\{ t \,\middle|\, (t \in R \wedge F_1) \wedge \left(\bigwedge_{i=2}^{n} F_i\right) \right\} \overset{\text{def } \sigma}{=}$$

$$= \sigma_{F_1}\left(\sigma_{\bigwedge_{i=2}^{n} F_i}(R)\right) = \ldots =$$

$$= \sigma_{F_1}(\ldots \sigma_{F_n}(R)\ldots)$$

Equivalence rules

① Conjunctive selections can be deconstructed into a sequence of individual selections:

$$\sigma_{p_1 \wedge p_2}(E) \quad \equiv \quad \sigma_{p_1}(\sigma_{p_2}(E))$$

② Selection operations are commutative:

$$\sigma_{p_1}(\sigma_{p_2}(E)) \quad \equiv \quad \sigma_{p_2}(\sigma_{p_1}(E))$$

③ Only the last projection in a sequence of projections is needed, the others can be omitted:

$$\pi_{L_1}(\pi_{L_2}(\cdots \pi_{L_n}(E) \cdots)) \quad \equiv \quad \pi_{L_1}(E)$$

Equivalence rules

④ Selections can be combined with cartesian prodcuts and joins:

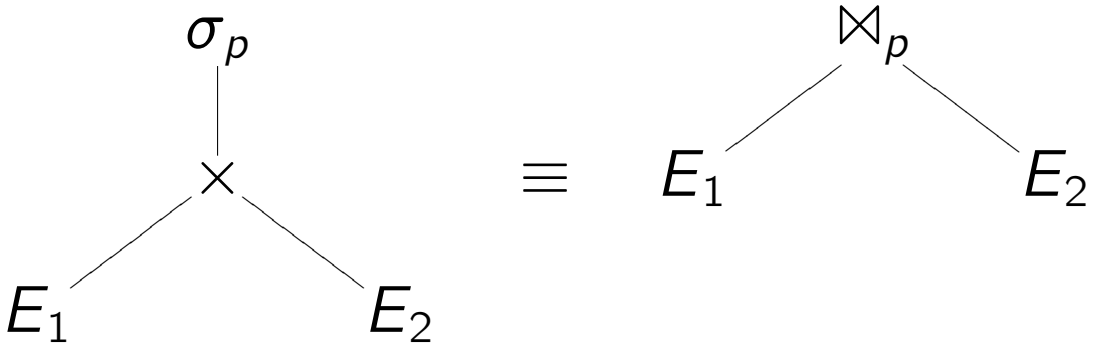
(a)

$$\sigma_p(E_1 \times E_2) \quad \equiv \quad E_1 \bowtie_p E_2$$

(b)

$$\sigma_p(E_1 \bowtie_q E_2) \quad \equiv \quad E_1 \bowtie_{p \wedge q} E_2$$

- Pictorial description of ④ (a):



Equivalence rules

⑤ Join operations are commutative:

$$E_1 \bowtie_p E_2 \quad \equiv \quad E_2 \bowtie_p E_1$$

⑥ (a) Natural joins (equality of common attributes) are associative:

$$(E_1 \bowtie E_2) \bowtie E_3 \quad \equiv \quad E_1 \bowtie (E_2 \bowtie E_3)$$

(b) Generals joins are associative in the following sense:

$$(E_1 \bowtie_p E_2) \bowtie_{q \wedge r} E_3) \quad \equiv \quad E_1 \bowtie_{p \wedge q} (E_2 \bowtie_r E_3)$$

where predicate r involves attributes of E_2, E_3 only.

Equivalence rules

⑦ Selection distributes over joins in the following ways:

(a) If predicate p involves attributes of E_1 only:

$$\sigma_p(E_1 \bowtie_q E_2) \quad \equiv \quad \sigma_p(E_1) \bowtie_q E_2$$

(b) If predicate p involves only attributes of E_1 and q involves only attributes of E_2:

$$\sigma_{p \wedge q}(E_1 \bowtie_r E_2) \quad \equiv \quad \sigma_p(E_1) \bowtie_r \sigma_q(E_2)$$

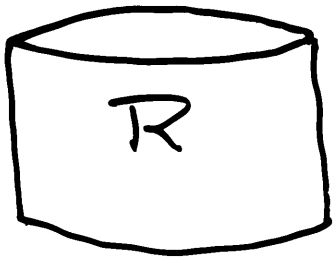
(this is a consequence of rules ⑦ (a) and ①).

EXAMPLE

$R = R(AB)$ $S = S(CD)$

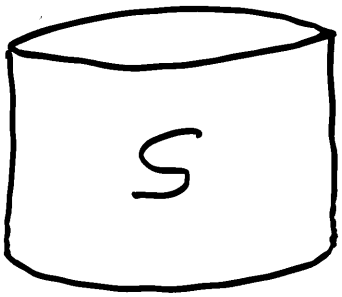
Query: PRINT NAMES (A) OF TREES (B, C) AGED 100 (D)

$$\pi_A \left(\sigma_{B=C \wedge D=100} (R \times S) \right)$$



N_1 records (AB)

B_1 records/block



N_2, B_2

Access time: 20 blocks/sec

Evaluate $R \times S$ first

$\frac{N_1}{B_1}$ accesses to R, $\frac{N_2}{B_2}$ accesses to S

$$\frac{N_1}{B_1}\left(B_1 \frac{N_2}{B_2}\right) = \frac{N_1 N_2}{B_2} \text{ accesses to disk}$$

$N_1 = 50\,000$; $B_1 = 10$; $N_2 = 100\,000$; $B_2 = 5$

total access time $= \frac{5 \cdot 10^9}{5} = 10^9 \longrightarrow$

5×10^7 sec

Using Relational Algebra properties

$$\sigma_{F_1 \wedge F_2}(R \times S) = \sigma_{F_2}(\sigma_{F_1}(R) \times S)$$

$$\pi_A(\sigma_{B=C \wedge D=100}(R \times S)) =$$

$$= \pi_A(\sigma_{B=C}(\sigma_{D=100}(S) \times R))$$

index on D (key)

$\propto$ accesses to R, $\propto < B_1$

Hence: $\propto \frac{N_2}{B_2}$ for $(\sigma_{D=100}(S) \times R)$

$$\propto \frac{N_2}{B_2} < B_1 \frac{N_2}{B_2} \quad !!$$

! Less time !

Equivalence rules

⑧ Projection distributes over join as follows:

$$\pi_{L_1 \cup L_2}(E_1 \bowtie_p E_2) \quad \equiv \quad \pi_{L_1}(E_1) \bowtie_p \pi_{L_2}(E_2)$$

if p involves attributes in $L_1 \cup L_2$ only and L_i contains attributes of E_i only.

⑨ The set operations union and intersection are commutative:

$$\begin{aligned} E_1 \cup E_2 &\equiv E_2 \cup E_1 \\ E_1 \cap E_2 &\equiv E_2 \cap E_1 \end{aligned}$$

⑩ The set operations union and intersection are associative:

$$\begin{aligned} (E_1 \cup E_2) \cup E_3 &\equiv E_1 \cup (E_2 \cup E_3) \\ (E_1 \cap E_2) \cap E_3 &\equiv E_1 \cap (E_2 \cap E_3) \end{aligned}$$

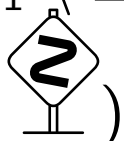
Equivalence rules

⑪ The selection operation distributes over $\cup, \cap$ and $\setminus$:

$$\begin{aligned}
\sigma_p(E_1 \cup E_2) &\equiv \sigma_p(E_1) \cup \sigma_p(E_2) \\
\sigma_p(E_1 \cap E_2) &\equiv \sigma_p(E_1) \cap \sigma_p(E_2) \\
\sigma_p(E_1 \setminus E_2) &\equiv \sigma_p(E_1) \setminus \sigma_p(E_2)
\end{aligned}$$

Also:

$$\begin{aligned}
\sigma_p(E_1 \cap E_2) &\equiv \sigma_p(E_1) \cap E_2 \\
\sigma_p(E_1 \setminus E_2) &\equiv \sigma_p(E_1) \setminus E_2
\end{aligned}$$

(this does not apply for $\cup$ )

⑫ The projection operation distributes over $\cup$:

$$\pi_L(E_1 \cup E_2) \equiv \pi_L(E_1) \cup \pi_L(E_2)$$

Heuristic optimization

- Query optimizers use the equivalence rules of relational algebra to improve the expected performance of a given query in *most cases*.

- The optimization is guided by the following **heuristics**:

  (a) **Break apart conjunctive selections** into a sequence of simpler selections
  (rule ①—preparatory step for (b)).

  (b) **Move σ down the query tree** for the earliest possible execution
  (rules ②, ⑦, ⑪—reduce number of tuples processed).

  (c) **Replace σ–$\times$ pairs by $\bowtie$**
  (rule ④ (a)—avoid large intermediate results).

  (d) **Break apart and move as far down the tree as possible lists of projection attributes**, create new projections where possible
  (rules ③, ⑧, ⑫—reduce tuple widths early).

  (e) **Perform the joins with the smallest expected result first.**