Databases 1

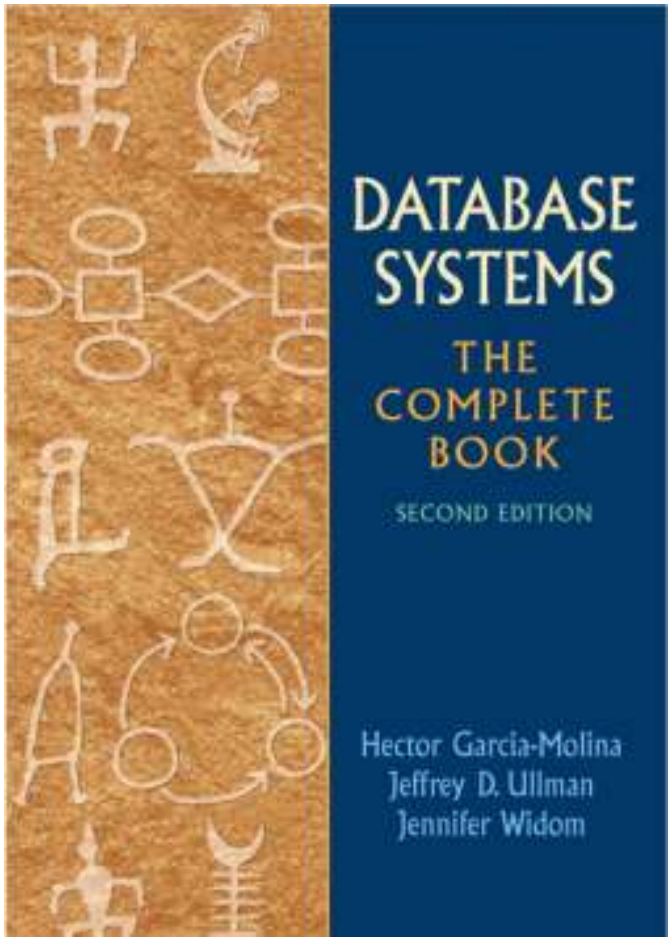
Exercises 2.4. Products Queries

Textbook

- A First Course in Database Systems (3rd ed.)
  by Jeff Ullman and Jennifer Widom

same material and sections as

- Database Systems: The Complete Book (2nd ed)
  by Garcia-Molina, Jeff Ullman and Jennifer Widom



Defining a Database Schema

- A database schema comprises declarations for the relations ("tables") of the database.
- *Relation schema* = relation name + attributes, in order (+ types of attributes).
  - Example: Beers(name, manf) or Beers(name: string, manf: string)
- *Relation = set of tuples (n-values)*
- *Database* = collection of relations.
- *Database schema* = set of all relation schemas in the database.

Core Relational Algebra

- Union, intersection, and difference.
  - Usual set operations, but require both operands have the same relation schema.
- Selection: picking certain rows.
- Projection: picking certain columns.
- Products and joins: compositions of relations.
- Renaming of relations and attributes.

Set operations: Union, intersection, difference

- To apply these operators the relations must have the same attributes.
- Union (R1∪R2): all tuples from R1 or R2
- Intersection (R1∩R2): common tuples from R1 and R2
- Difference (R1\R2): tuples occuring in R1 but not in R2

Projection and Selection

- R1 := π_L(R2)
  - *L* is a list of attributes from the schema of R2.
  - R1 is constructed by looking at each tuple of R2, extracting the attributes on list *L*, in the order specified, and creating from those components a tuple for R1.
  - Eliminate duplicate tuples, if any.
- R1 := σ_C(R2)
  - *C* is a condition (as in "if" statements) that refers to attributes of R2.
  - R1 is all those tuples of R2 that satisfy *C*.

Product and Natural Join

- R3 := R1 x R2
  - Pair each tuple t1 of R1 with each tuple t2 of R2.
  - Concatenation t1t2 is a tuple of R3.
  - Schema of R3 is the attributes of R1 and R2, in order.
  - But beware attribute *A* of the same name in R1 and R2: use R1.*A* and R2.*A*.
- A frequent type of join connects two relations by:
  - Equating attributes of the same name, and
  - Projecting out one copy of each pair of equated attributes.
  - Called *natural* join.
  - Denoted R3 := R1 ⋈ R2.

Renaming

- The RENAME operator gives a new schema to a relation.
- R1 := $\rho_{1(A1,\ldots,An)}$(R2) makes R1 be a relation with attributes A1,…,A*n* and the same tuples as R2.
- Simplified notation: R1(A1,…,A*n*) := R2.

Building Complex Expressions

- Algebras allow us to express sequences of operations in a natural way
  - Example: in arithmetic --- $(x + 4)*(y - 3)$.
- Relational algebra allows the same.
- Three notations, just as in arithmetic:
  1. Sequences of assignment statements.
  2. Expressions with several operators.
  3. Expression trees.

Expression Trees

- Precedence of relational operators:
  1. Unary operators --- select, project, rename --- have highest precedence, bind first.
  2. Then come products and joins.
  3. Then intersection.
  4. Finally, union and set difference bind last.
- But you can always insert parentheses to force the order you desire.
- Leaves are operands --- either variables standing for relations or particular, constant relations.
- Interior nodes are operators, applied to their child or children.

Exercises 2.4.1.

- The database schema consists of four relations, whose schemas are:

  Product(maker, model, type)

  PC(model, speed, ram, hd, price)

  Laptop(model, speed, ram, hd, screen, price)

  Printer(model, color, type, price)

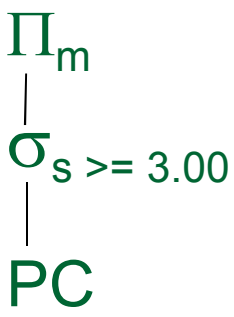
- create table:
  http://people.inf.elte.hu/sila/eduAB/create_products.txt

(a.)

a.) What PC models have a speed of at least 3.00 GHz?

Relational algebra:

$\Pi_m(\sigma_{s \geq 3.00}(PC))$



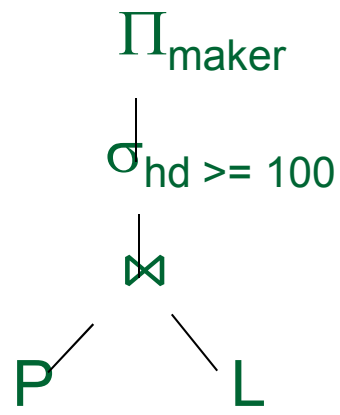
SQL SELECT:

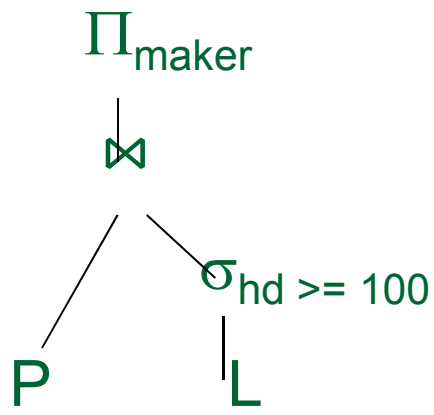
```
SELECT model
  FROM PC
  WHERE speed>=3.00;
```

(b.)

b.) Which manufacturers make laptops with a hard disk (hd) of at least 100 GB?

$\Pi_{maker}(\sigma_{hd \geq 100} (P \bowtie L))$ or ekv. $\Pi_{maker}(P \bowtie (\sigma_{hd \geq 100}(L))$






```
SELECT maker
    FROM Product P, Laptop L
  WHERE P.model=L.model
     AND hd>=100;
```

(c.)

c.) Find the model number and price of products (of any type) made by manufacturer B.

--- **BP :=** $\Pi_{m}\, \sigma_{gy='B'}$**(P)** --->> $\Pi_{m,\, ár}$**(BP** $\bowtie$ **PC)** $\cup$
$\cup\, \Pi_{m,\, ár}$**(BT** $\bowtie$ **Laptop)** $\cup$
$\cup\, \Pi_{m,\, ár}$**(BT** $\bowtie$ **Printer)**

with BP as

(select model from product where maker='B')

```
select model, price from pc natural join BP
        union
select model, price from laptop natural join BP
        union
select model, price from printer natural join BP;
```

(d.)

d.) Find the model numbers of all color laser printers.

$$\Pi_{m}(\sigma_{sz='i'}(Ny)) \cap \Pi_{m}(\sigma_{t='lézer'}(Ny))$$

-- elvégezhető más módon is: $\Pi_{m}(\sigma_{sz='i' \wedge t='lézer'}(Ny)) =$

$$= \Pi_{m}(\sigma_{sz='i'}\,\sigma_{t='lézer'}(Ny)) = \Pi_{m}(\sigma_{t='lézer'}\,\sigma_{sz='i'}(Ny))$$

(e.)

e) Find those manufacturers that sell Laptops, but not PC's
(ha laptop gyártó több pc-t gyárt, akkor az eredménytábla csökken, nem monoton művelet: R - S)

$$\Pi_{gy}(T \bowtie L) - \Pi_{gy}(T \bowtie PC)$$

(f.)

! f) Find those hard-disk sizes that occur in two or more PC's.

(táblát önmagával szorozzuk)

-- segédváltozót vezetek be, legyen **PC_1 := PC**

$\Pi_{PC.ml}(\sigma_{PC_1.m \neq PC.m \wedge PC_1.ml=PC.ml}(PC_1 \times PC))$

(g.)

! g) Find those pairs of PC models that have both the same cpu speed and RAM, the size of memory. A pair should be listed only once, e.g., list (I,j) but not (j,i)

$$\Pi_{PC_1.m,\ PC.m}(\sigma_{PC_1.m<PC.m \wedge PC_1.s=PC.s \wedge PC_1.me=PC.me}(PC_1 \times PC))$$

(h.)

- !! h) Find those manufacturers of at least two different computers (PC's or laptops) with speeds of at least 2.80 GHz.

  -- segédváltozó: **Gyors** := $\Pi_m(\sigma_{s\geq 2.8}(PC)) \cup \Pi_m(\sigma_{s\geq 2.8}(L))$

  -- és ezzel legyen: T_1 := T $\bowtie$ **Gyors** és T_2 := T $\bowtie$ **Gyors**

  $\Pi_{T_1.gy}(\sigma_{T_1.gy = T2.gy \wedge T_1.m \neq T_2.m} (T_1 \times T_2))$

(i.)

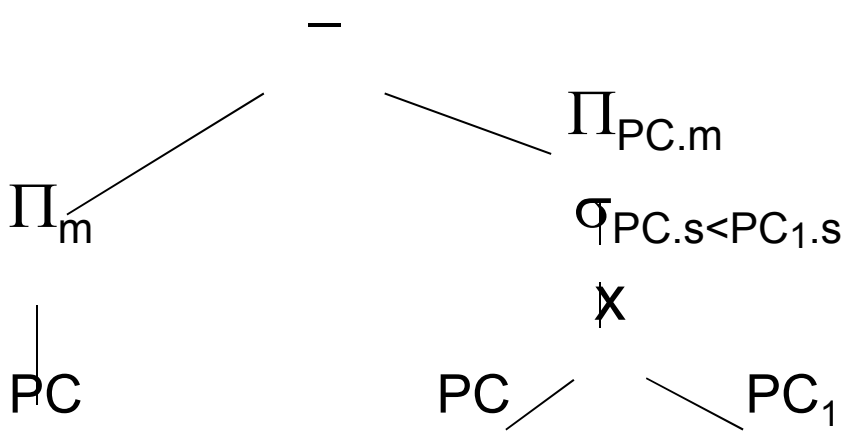
- !!i) Find the manufacturers of the computer (PC or laptop) with the highest available speed.

  Kiválasztjuk azokat a PC-ket, amelyiknél van gyorsabb, ha ezt kivonjuk a PC-ékből megkapjuk a leggyorsabbat:

  EnnélVanNagyobb = $\Pi_{PC.m}(\sigma_{PC.s<PC_1.s}(PC \times PC1))$

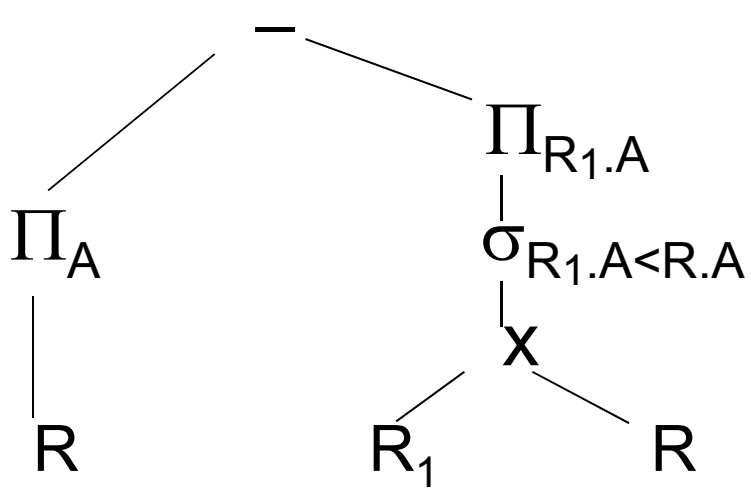
  Leggyorsabb: $\Pi_m(PC) - EnnélVanNagyobb$

Ehhez rajzoljuk fel a kiértékelő fát is: (folyt.: PC helyett számítógép kell és a válaszban is a gyártó kell…)



MAX with relational algebra

- R(A,B). Feladat: Adjuk meg MAX(A) értékét! (Ez majd átvezet az új témára, aggregáló függvényekre, illetve csoportosításra).
- $\pi_A(R) - \pi_{R1.A}(\sigma_{R1.A<R.A}(\rho_{R_1}(R) \times R))$
- tree:



From relational algebra to SQL

- Kiértékelő fa szerinti átírás SQL-be:

```
(SELECT A FROM R)
      EXCEPT
(SELECT R1.A AS A
      FROM R R1, R R2
      WHERE R1.A<R2.A);
```

- Nézzük meg korrelált (függő) alkérdéssel is:

```
SELECT A FROM R MAXA
      WHERE NOT EXISTS
            (SELECT A FROM R
             WHERE A > MAXA.A);
```

Példák rel.algebrai kif. átírása (j.)

!! j) Find the manufacturers of PC's with at least three different cpu speeds.

mint a legalább kettő, csak ott 2x, itt 3x kell a táblát önmagával szorozni. Legyenek $\mathbf{S, S_1, S_2 := T \bowtie \Pi_{m,s}(PC)}$

$$\Pi_{S.gy}(\sigma_{S_1.gy=S.gy \wedge S_2.gy=S.gy \wedge S_1.s \neq S.s \wedge S_2.s \neq S.s \wedge S_1.s \neq S_2.s}(S \times S_1 \times S_2))$$

!! k) Find the manufacturers who sell exactly three different models of PC. legalább 3-ból - legalább 4-t kivonni