JACOBS UNIVERSITY

Relational Algebra

Molina, Ullman, Widom

Database Management: Complete Book,

Chapters 2 & 5

Selection

- $R1 := \sigma_C (R2)$
  - *C* : condition on attributes of R2.
  - R1 is all those tuples of R2 that satisfy *C*.

sid	name	login	gpa
53666	Jones	jones@cs	3.4
53688	Smith	smith@eecs	3.2
53650	Smith	smith@math	3.8

$\sigma_{gpa<3.8}$(Students):

sid	name	login	gpa
53666	Jones	jones@cs	3.4
53688	Smith	smith@eecs	3.2

Selection: Observations

- unary operation: 1 table
- conditions apply to each tuple individually
  - condition cannot span tuples (how to do that?)
- degree of $\sigma_C (R)$ = degree of R
  - Cardinality?
- Select is commutative: $\sigma_{C1} (\sigma_{C2}(R)) = \sigma_{C2} (\sigma_{C1}(R))$

Projection

JACOBS UNIVERSITY

- $R1 := \pi_{attr}(R2)$
  - *attr* : list of attributes from R2 schema
- For each tuple of R2:
  - extract attributes from list attr in order specified (!) → R1 tuple
- Eliminate duplicate tuples

```
sid   name  login        gpa
----------------------------
53666 Jones jones@cs     3.4
53688 Smith smith@eecs   3.2
53650 Smith smith@math   3.8
```

$\pi_{name,login}(Students) =$

```
name  login
----------------
Jones jones@cs
Smith smith@eecs
```

Projection: Observations

JACOBS UNIVERSITY

- Unary operation: 1 table
- removes duplicates in result
  - Cardinality?
  - Degree?
- Project is not commutative
- Sample algebraic law: $\pi_{L1}(\pi_{L2}(R)) = \pi_{L1}(R)$ if $L1 \subseteq L2$
  - else incorrect expression, syntax error

Cartesian Product

- project, select operators operate on single relation
- Cartesian product combines two:
  R3 = R1 x R2
  - Pair each tuple t1 $\in$ R1 with each tuple t2 $\in$ R2
  - Concatenation t1,t2 is a tuple of R3
  - Schema of R3 = attributes of R1 and then R2, in order
  - beware attribute *A* of the same name in R1 and R2: use R1.*A* and R2.*A*

Natural Join

- R3 = R1 $\bowtie$ R2
- connect two relations:
  - Equate attributes of same name
  - Project out redundant attiribute(s)
- Ex: Sailors $\bowtie_{bid}$ Reserves

Theta Join

- Generalization of equi-join: A θ B
  - θ one of =, <, ...
- R3 = R1 $\bowtie_C$ R2
  - R1 x R2, then apply σ_C
- Today, more general: σ_C can be any predicate

Relational Algebra: Summary

JACOBS UNIVERSITY

- Mathematical definition of relations + operators
  - Query = Algebraic expression
- Relational algebra $A = (R,OP)$ with relation $R = A_1 \times \ldots \times A_n$, $OP=\{\pi,\sigma,\times\}$
  - Projection: $\pi_{attr}(R) = \{ r.attr \mid r \in R \}$
  - Selection: $\sigma_p(R) = \{ r \mid r \in R, p(r) \}$
  - Cross product: $R_1 \times R_2 = \{(r_{11}, r_{12}, \ldots, r_{21}, r_{22}, \ldots) \mid (r_{11}, r_{12}, \ldots) \in R_1, (r_{21}, r_{22}, \ldots) \in R_2 \}$
  - Further: set operations, join, ...

Relational Calculus

- Tuple variable = variable over some relation schema
- Query $Q = \{ T \mid T \in R, p(T) \}$
  - R relation schema, p(T) predicate over T
- Example 1: "sailors with rating above 8"
  - Sailors = sid:int $\times$ sname:string $\times$ rating:int $\times$ age:float
  - $\{ S \mid S \in \text{Sailors} \wedge S.\text{rating} > 8 \}$
- Example 2: "names of sailors who have reserved boat #103":
  - Reserves = sid:int $\times$ bid:int $\times$ day:date
  - $\{ P.\text{sname} \mid \exists S \in \text{Sailors}\ \exists R \in \text{Reserves}:$
    $R.\text{sid}=S.\text{sid} \wedge R.\text{bid}=103 \wedge P.\text{sname}=S.\text{sname} \}$

Comparison of Relational Math

- Relational algebra
  - set-based formalization of selection, projection, cross product (no aggregation!)
  - Operation oriented = procedural = imperative
- Relational calculus
  - Same, but in predicate logic
  - Describing result = declarative; therefore basis of SQL
- Equally powerful
  - proven by Codd in 1970