# Database Systems

10.8, 2008
Lecture #4



# Course Administration

- Assignment #1 is due next Wed (outside 336/338).
- Course slides will now have black background
  - Printer friendly: set the printing color to "black-white"
- Next week reading:
  - Chapter 5 SQL

# Mischief (Video, MIT/Stanford)

30 mice & 1 PC in a classroom

# Reflection: DB design

- Step 1: Requirements Analysis
  - What data to store in the database?
- Step 2: Conceptual Database Design
  - Come up with the design: Entity-Relation (ER) model
  - Sketch the design with entity-relationship diagrams
- Step 3: Logical Database Design
  - Implement the design: relational data model
  - Map ER diagrams to relational tables
- What's next (after creating all these nice tables)?

4

# What's next?

- How to ask questions about the [relational] database?
  - How much money in account XYZ?
  - Who are valuable customers [∑ deposits > 1M]?
  - Find the better 3-4 combination in MLB that is better (on-based percentage) than Ramirez and Ortiz (Boston Red Sox).
- Two query languages
  - Relational algebra [CH4] : a Math query language
  - SQL [CH5] : a Real query Language

5

# Relational Algebra

Chapter 4.1 – 4.2

6

# Relational Query Languages

- What are query languages?
  - For asking questions about the database
- Relational Algebra
  - Mathematical Query Languages form the basis for "real" languages (e.g. SQL) and for implementation.
  - A relational query is composed using a small set of operators ($\pi$, $\sigma$, $\infty$, $X$, …)
    - Like +, -, *, / operators in algebra

7

# Preliminaries

- A query is applied to table(s), and the result of a query is also a table.
  - Schema of input table(s) for a query is fixed.
  - Schema for the result of a given query is also fixed! Determined by definition of query language constructs.
- Example of a *relational algebra expression*:
  - Find the names of sailors who have reserved boat 103

$$\pi_{sname}((\sigma_{bid = 103} Reserves) \infty Sailors)$$

8

# Example Tables

- Sailors and Reserves are tables.
- Can refer to the fields by their positions or names:
- Assume names of fields in the result table are inherited from names of fields in input tables.

***R1***

| sid | bid | day |
|---|---|---|
| 22 | 101 | 10/10/96 |
| 58 | 103 | 11/12/96 |

***S1***

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

***S2***

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

9

# Relational Algebra

- Basic relational algebra operators:
  - Selection (σ, pronounced sigma): Select a subset of rows from a table.
  - Projection (π): Delete unwanted columns from a table.
  - Cross-product ( *X* ): Combine two tables.
  - Set-difference ( - ): Tuples in table 1, but not in table 2.
  - Union ( *U* ): Tuples in tables 1 or 2.
- Each operator takes one or two input table(s), and returns one table.

10

# Relational Algebra (more)

- Additional relational algebra operators:
  - Intersection ($\cap$) : Tuples in tables 1 and 2.
  - Join ($\infty$): conditional cross product
  - Division (/): will explain later
  - Renaming ($\rho$ *, pronounced "row"*)
- Operations can be composed to form a very complex query

$$\pi_{sid}\,(\sigma_{age > 20}\,\text{Sailors}) \; - \; \pi_{sid}\,((\sigma_{color = \text{'red'}}\,\text{Boats}) \infty \text{Reserves} \infty \text{Sailors})$$

11

# Relational Operators

- Projection
- Selection
- Union
- Intersection
- Set difference
- Cross product
- Rename operator
- Join
- Division

12

## Projection

- Delete attributes not in projection list.
- Duplicates eliminated
- Find ages of sailors in S2
- Find names of sailors in S2

| sname | rating |
|---|---|
| yuppy | 9 |
| lubber | 8 |
| guppy | 5 |
| rusty | 10 |

$\pi_{sname,rating}(S2)$

***S2***

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

| age |
|---|
| 35.0 |
| 55.5 |

$\pi_{age}(S2)$

13

## Relational Operators

- Projection
- Selection
- Union
- Intersection
- Set difference
- Cross product
- Rename operator
- Join
- Division

14

## Selection

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 58 | rusty | 10 | 35.0 |

- Selects rows satisfying selection condition.
  - with duplicate removal
- Result table can be fed into other operations
- Find (names, ratings) of sailors whose ratings are greater than 8.
- Find names of sailors whose ages are greater than 40.

$$\sigma_{rating>8}(S2)$$

| sname | rating |
|---|---|
| yuppy | 9 |
| rusty | 10 |

S2

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

$$\pi_{sname,rating}(\sigma_{rating>8}(S2))$$

15

## Relational Operators

- Projection
- Selection
- Union
- Intersection
- Set difference
- Cross product
- Rename operator
- Join
- Division

16

# Union

- Take two input tables, which must be union-compatible:
  - Same number of fields.
  - `Corresponding' fields have the same type.
- What is the schema of result?
- Find sailors in S1 or S2.

***S1***

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

***S2***

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

$S1 \cup S2$

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |
| 44 | guppy | 5 | 35.0 |
| 28 | yuppy | 9 | 35.0 |



# Intersection

***S1***

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

***S2***

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

$S1 \cap S2$

| sid | sname | rating | age |
|---|---|---|---|
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

# Set-Difference

*S1*

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

*S2*

| sid | sname | rating | age |
|---|---|---|---|
| 28 | yuppy | 9 | 35.0 |
| 31 | lubber | 8 | 55.5 |
| 44 | guppy | 5 | 35.0 |
| 58 | rusty | 10 | 35.0 |

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |

$S1 - S2$

19

# Relational Operators

- Projection
- Selection
- Union
- Intersection
- Set difference
- Cross product
- Rename operator
- Join
- Division

20

# Cross-Product

- Each row of S1 is paired with each row of R1.
- Result schema has one field per field of S1 and R1, with field names `inherited' if possible.
  - Conflict: Both S1 and R1 have a field called sid.

*S1*

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

*R1*

| sid | bid | day |
|---|---|---|
| 22 | 101 | 10/10/96 |
| 58 | 103 | 11/12/96 |

*S1 x R1*

| (sid) | sname | rating | age | (sid) | bid | day |
|---|---|---|---|---|---|---|
| 22 | dustin | 7 | 45.0 | 22 | 101 | 10/10/96 |
| 22 | dustin | 7 | 45.0 | 58 | 103 | 11/12/96 |
| 31 | lubber | 8 | 55.5 | 22 | 101 | 10/10/96 |
| 31 | lubber | 8 | 55.5 | 58 | 103 | 11/12/96 |
| 58 | rusty | 10 | 35.0 | 22 | 101 | 10/10/96 |
| 58 | rusty | 10 | 35.0 | 58 | 103 | 11/12/96 |

Renaming operator: $\rho(C(1 \to sid1, 5 \to sid2), S1 \times R1)$

# Condition Joins

$S1 \bowtie_{S1.sid < R1.sid} R1$

$R \bowtie_C S = \sigma_C (R \times S)$

| (sid) | sname | rating | age | (sid) | bid | day |
|---|---|---|---|---|---|---|
| 22 | dustin | 7 | 45.0 | 58 | 103 | 11/12/96 |
| 31 | lubber | 8 | 55.5 | 58 | 103 | 11/12/96 |

*R1*

| sid | bid | day |
|---|---|---|
| 22 | 101 | 10/10/96 |
| 58 | 103 | 11/12/96 |

*S1*

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

- Cross-product, followed by a selection
- *Result schema* same as that of cross-product.
- Fewer tuples than cross-product, reduce tuples not meeting the condition.

22

# Equi-Joins

- A special case of condition join where the condition *c* contains only equalities.
- Result schema similar to cross-product, but only one copy of fields for which equality is specified.
- Natural Join (∞): Equi-join on all common fields.

***R1***

| sid | bid | day |
|---|---|---|
| 22 | 101 | 10/10/96 |
| 58 | 103 | 11/12/96 |

***S1***

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

$S1 \bowtie_{sid} R1$

| sid | sname | rating | age | bid | day |
|---|---|---|---|---|---|
| 22 | dustin | 7 | 45.0 | 101 | 10/10/96 |
| 58 | rusty | 10 | 35.0 | 103 | 11/12/96 |

23

# Example of Join

- Find the names of sailors who have reserved at least a boat.
  - $\pi_{sname}(R1 \infty S1)$
  - $\pi_{sname}(R1 \infty_{R1.sid = S1.sid} S1)$
  - $\pi_{sname}(R1 \infty_{sid} S1)$

***R1***

| sid | bid | day |
|---|---|---|
| 22 | 101 | 10/10/96 |
| 58 | 103 | 11/12/96 |

***S1***

| sid | sname | rating | age |
|---|---|---|---|
| 22 | dustin | 7 | 45.0 |
| 31 | lubber | 8 | 55.5 |
| 58 | rusty | 10 | 35.0 |

24

## Relational Operators

- Projection
- Selection
- Union
- Intersection
- Set difference
- Cross product
- Rename operator
- Join
- Division

25

## Division: analogy to integer division

- Two integers: *A, B*
  - *A/B = Q, Q* is the largest integer such that *Q * B ≤ A*
- How about relational tables *A, B*?
  - *A/B = Q, Q* is the largest table such that *Q X B ≤ A*
- Look at *Q X B in A*
  - *Q* must be a subset of attributes in *A*
  - *Q*'s attributes + B's attributes = A's attributes
  - *A's* tuples must contain all pairings <q in Q, b in B>

26

# Division

- Reserves(sailor_name, boat_id); Boats (boat_id)
  - Useful for expressing queries like:

  *Find sailors who have reserved* all *boats => Reserves / Boats*
- Let *A* have 2 fields, *x* and *y*; *B* have only field *y*:
  - *A/B* = $\{\langle x \rangle \mid \exists \langle x, y \rangle \in A \ \ \forall \langle y \rangle \in B\}$
  - *A[xy]/B[y]* contains all *x* tuples (sailor_name) such that for *every y* tuple (boat_id) in *B*, there is an *xy* tuple in *A*.

27

# Examples of Division A/B

| X | Y |
|---|---|
| X1 | Y1 |
| X1 | Y2 |
| X1 | Y3 |
| X1 | Y4 |
| X2 | Y1 |
| X2 | Y2 |
| X3 | Y2 |
| X4 | Y2 |
| X4 | Y4 |

*A*

| Y |
|---|
| Y2 |

*B1*

| X |
|---|
| X1 |
| X2 |
| X3 |
| X4 |

*A/B1*

| Y |
|---|
| Y2 |
| Y4 |

*B2*

| X |
|---|
| X1 |
| X4 |

*A/B2*

| Y |
|---|
| Y1 |
| Y2 |
| Y4 |

*B3*

| X |
|---|
| X1 |

*A/B3*

28

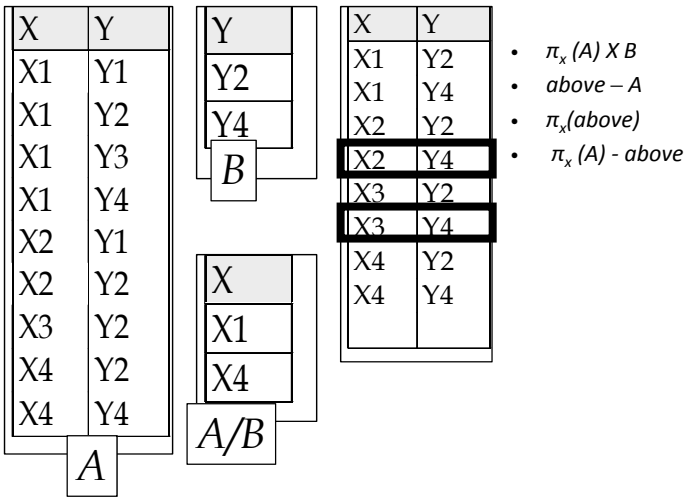
# Expressing A/B Using Basic Operators

- Idea: For *A(xy)/B(y)*, compute all *x* values that are not disqualified by some *y* value in *B*.
  - *x* value is disqualified if by attaching *y* value from *B*, we obtain an *xy* tuple that is not in *A*.
  - 1) Iterate through each *x* value
  - 2) Check: combined with each *y* value, *xy* in *A*? If not, disqualify.

- Disqualified *x* values:
  *A/B =*

29

# Examples of Division A/B




- $\pi_x$ *(A) X B*
- *above – A*
- $\pi_x$*(above)*
- $\pi_x$ *(A) - above*

30

# Practices with Relational Operators



# (Q1) Find names of sailors who've reserved boat #103

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*

- Solution 1:
  $\pi_{sname}(\sigma_{bid = 103}$ *(Reserves ∞ Sailors))*
- Solution 2 (more efficient)
  $\pi_{sname}((\sigma_{bid = 103}$ *Reserves) ∞ Sailors)*
- Solution 3 (using rename operator)
  *P(Temp1,* $\sigma_{bid = 103}$ *Reserves)*
  *P(Temp2, Temp1 ∞ Sailors)*
  $\pi_{sname}$*(Temp2)*

## (Q2) Find names of sailors who've reserved a red boat

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Solution?

$\pi_{sname}((\sigma_{color = 'red'} Boats) \infty Reserves \infty Sailors )$

- A more efficient solution (# of disk access)?

$\pi_{sname}(\pi_{sid}((\pi_{bid}\sigma_{color = 'red'} Boats) \infty Reserves ) \infty Sailors )$

*A query optimizer can find this, given the first solution!*

33

## (Q3) Find the colors of boats reserved by Lubber

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Solution?

$\boldsymbol{\pi_{color}((\sigma_{sname = 'Lubber'} Sailor) \infty Reserves \infty Boats )}$

34

## (Q4) Find the names of sailors who have reserved at least one boat

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Solution?

$\pi_{sname}(Sailor \infty Reserves)$

35

## (Q5) Find the names of sailors who have reserved a red or a green boat

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Solution?

$\pi_{sname}(\sigma_{color='red' \text{ or } color = 'green'} Boats \infty Reserves \infty Sailors)$

36

## (Q6) Find the names of sailors who have reserved a red and a green boat

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Wrong solution:

$\pi_{sname}(\sigma_{color='red' \text{ and } color='green'}$ *Boats ∞ Reserves ∞ Sailors)*

- What's wrong with the above?
- A correct solution?

$\pi_{sname}(\sigma_{color='red'}$ ***Boats ∞ Reserves ∞ Sailors)*** $\cap \pi_{sname}(\sigma_{color='green'}$ ***Boats ∞ Reserves ∞ Sailors)***

37

## (Q7) Find the names of sailors who have reserved at least two different boats

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Strategy?
  - *Join a table (sid, bid, sname): sailors reserving at least one boat*
  - *Cross-product the table with itself*
  - *Select sailors with two different boats reserved*

*P (Reservations, C(1->sid1, 2->sid2, 3->bid1, 4->bid2)*
$\pi_{sid,sname,bid}$ *(Sailors ∞ Reserves))*
$\pi_{sname}(\sigma_{(sid1=sid2) \,\&\, (bid1 \neq bid2)}$ *Reservations x Reservations)*

38

## (Q8) Find the sids of sailors with age over 20 who have not reserved a red boat

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Strategy
  - Find all sailors (sids) with age over 20
  - Find all sailors (sids) who have reserved a red boat
  - Take their set differences

$$\pi_{sid}\,(\sigma_{age>20}\,Sailors) - \pi_{sid}\,((\sigma_{color='red'}\,Boats) \infty Reserves)$$

39

## (Q9A) Find the names of sailors who have reserved all boats

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Strategy
  - *Use division; all = division*
  - *what divides by what?Solution*

$$\boldsymbol{\pi_{sname}\,((\pi_{sid,bid}\,(Reserves) / \pi_{bid}\,(Boats)) \infty Sailors)}$$

40

## (Q9') Find the names of sailors who have reserved boats with all different colors

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Strategy
  - *what divides by what?*
    - *(sid, color) / (color)*
- Solution

$$\pi_{sname}((\pi_{sid,color}(Reserves \infty Boats) / \pi_{color}(Boats)) \infty Sailors)$$

41

## (Q10) Find the names of sailors who have reserved all boats called "Interlake"

*Reserves(sid, bid, day)*
*Sailors(sid, sname, rating, age)*
*Boats(bid, bname, color)*

- Previous solution?

$$\pi_{sname}((\pi_{sid,bid}(Reserves) / \pi_{bid}(Boats)) \infty Sailors)$$

- How to modify it?

$$\pi_{sname}((\pi_{sid,bid}(Reserves) / \pi_{bid}(\sigma_{bname='Interlake'} Boats)) \infty Sailors)$$

42

# More Exercises



*employee(person-name, street, city)*
*works(person-name, company-name, salary)*
*company(company-name, city)*
*manages(person-name, manager-name)*

*(a) Find the names of all employees who work for First Bank Corporation.*

*(b) Find the names and cities of residence of all employees who work for First Bank Corporation.*

*(c) Find the names, street address, and cities of residence of all employees who work for First Bank Corporation and earn more than $10,000 per annum.*

*employee(person-name, street, city)*
*works(person-name, company-name, salary)*
*company(company-name, city)*
*manages(person-name, manager-name)*

*(d) Find the names of all employees in this database who live in the same city as the company locates for which they work.*

(e) *Find the names of all employees who live in the same city and on the same street as do their managers.*

*(f) Find the names of all employees in this database who do not work for the First Bank Corporation.*

45

*employee(person-name, street, city)*
*works(person-name, company-name, salary)*
*company(company-name, city)*
*manages(person-name, manager-name)*

*(g) Find the names of all employees who earn more than every employee of Small Bank Corporation.*

*(h) Assume the companies may be located in several cities. Find all companies located in every city in which Small Bank Corporation is located.*

*(i) Find the names of employees who work for more than 3 (included) companies.*

46