Database Systems

SQL: Data Manipulation (DML)
Relational Algebra Concepts

Relational Algebra

- ◆ Theoretical language in nature
- ◆ Operations that work on one or more relations (tables) to give another relation (table) as a result
- ◆ Usefulness:
  - Procedural alternative to solving queries
  - Essential to understanding SQL query processing and optimization

---

Relational Algebra

- ◆ Relational algebraic operations:
  - **Projection**
  - **Restriction**

    Unary table operation

  - **Product**
  - Join
  - **Union**
  - Intersection
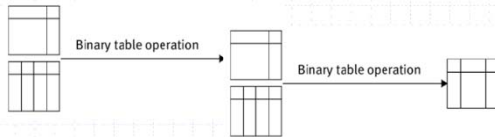 - **Difference**
  - Division

    Binary table operation

1

Relational Algebra

◆ Property of Closure:
- The result of a relational operation on relations (tables) is also a relation (table)
- The allows the nesting of operations to perform complex queries

| | Binary table operation | | Binary table operation | |

Relational Algebra

◆ Fundamental operations:
- **Projection**
- **Restriction (Selection)**
- **Product**
- **Union**
- **Difference**

◆ Additional operations: can be derived
- Join
- Intersection
- Division

Fundamental Operations

- **Projection**
  - T1 <= $\pi_{col\ 1,\ col\ 2,\ ...}$ (R)
- **Restriction (Selection)**
  - T2 <= $\sigma_{predicate}$ (R)
- **Product**
  - T3 <= R × S
- **Union**
  - T4 <= R ∪ S
- **Difference**
  - T5 <= R − S

Additional Operations

- **Intersection**
  - T6 <= R ∩ S
  - T6 <= R − (R − S)
- **Join**
  - T7 <= R ⋈$_{conditions}$ S
    where join conditions is of form R.coln = S.coln ...
  - T7 <= $\pi_{col\ 1,\ col\ 2,\ ...}$ ($\sigma_{R.coln\ =\ S.coln}$ (R x S))
- **Division**
  - T8 <= R ÷ S
  - See the textbook

Basic SQL SELECT Statement

```
SELECT [DISTINCT | ALL]
    {* | [columnExpression [AS newName]]
       [,...] }
 FROM TableName [alias] [, ...]
 [WHERE   condition]
 [GROUP BY    columnList]
 [HAVING condition]
 [ORDER BY    columnList];
```

- Syntax order of the clauses 'cannot' be changed
- Only SELECT and FROM are mandatory (Oracle); SQL Server – only SELECT is mandatory

---
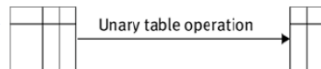
Relational Algebra

- Unary operators:

  Unary table operation

  - **Projection** – pick out columns
  - SQL: SELECT <columns>

  - **Restriction** – pick out rows meeting certain conditions
  - SQL: WHERE <conditions>

Slide 10

$T1 <= \pi_{Set\_ID, Option, Team\_ID} (R)$

Student_ID	LName	FName	Set_ID	Option	Team_ID	Grade
000001234	Simpson	Arthur	3A	IS	03	0
000002345	Clinton	Cliff	3E	DC	02	0
000003456	Graham	William	3A	IS	03	0
000004567	Powell	Diana	3A	IS	03	0
000012345	Koch	Michael	3F	DB	01	0
000023456	Lake	John	3F	DB	01	0
000034567	Miller	Jane	3F	DB	01	0
000045678	Newman	Paul	3E	DC	02	0
000123456	Martin	James	3E	DC	02	0

Set_ID	Option	Team ID
3A	IS	03
3E	DC	02
3F	DB	01

Keith Tang   COMP2714   10

Slide 11

$T2 <= \sigma_{Scheduled\_Date < DATE'2012-09-23'} (R)$

Milestone	Description	Scheduled_Date
PPR	Project Plan Reviewed	2012-09-22
PSC	Project Selection Complete	2012-09-08
SAT	Software Accepted & Tested	2012-12-01
SCC	Supervisor & Client Contacted	2012-09-16
SCR	Software Coded & Reviewed	2012-11-03
SDR	Software Design Reviewed	2012-10-14
SIT	Software Integrated & Tested	2012-11-07
SRR	Software Requirements Reviewed	2012-10-01

Milestone	Description	Scheduled Date
PPR	Project Plan Reviewed	2012-09-22
PSC	Project Selection Complete	2012-09-08
SCC	Supervisor & Client Contacted	2012-09-16

Keith Tang   COMP2714   11

Slide 12

Relational Algebra

◆ Binary SET operators
Union Compatible

- A **Union** B :
  All rows from
  both tables
- A **Intersect** B :
  Rows existed
  in both tables
- A **Difference** B :
  Rows in one but
  not in other table

Keith Tang   COMP2714   12

Set Operations

◆ SQL2 defines set operations:
- Union of two tables, A and B, is table containing all rows in either A or B or both
- Intersection is table containing all rows common to both A and B
- Difference (Except) is table containing all rows in A but not in B

◆ Two tables must be **union compatible**
- Same number of corresponding attributes / columns with matching domain / data types
- How to make tables union compatible?

T4 <= R ∪ S

Faculty_ID	Faculty_Name
BA	Brian Anderson
DW	Dion Walsh
JC	John Chase
KT	Keith Tang

Faculty_ID	Faculty_Name
BA	Brian Anderson
JC	John Chase
KT	Keith Tang

Faculty_ID	Faculty_Name
BA	Brian Anderson
DW	Dion Walsh

T5 <= R − S

Faculty_ID	Faculty_Name
JC	John Chase
KT	Keith Tang

Faculty_ID	Faculty_Name
BA	Brian Anderson
JC	John Chase
KT	Keith Tang

Faculty_ID	Faculty_Name
BA	Brian Anderson
DW	Dion Walsh

T6 <= R ∩ S

Faculty_ID	Faculty_Name
BA	Brian Anderson
JC	John Chase
KT	Keith Tang

Faculty_ID	Faculty_Name
BA	Brian Anderson

Faculty_ID	Faculty_Name
BA	Brian Anderson
DW	Dion Walsh

Relational Algebra

◆ Binary operators

Binary table operation
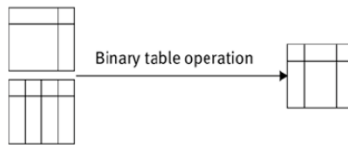
- **Product** – SQL: CROSS JOIN
- **Join**   – SQL: INNER JOIN, OUTER JOIN
  SQL: FROM clause
- **Division** – no single SQL counterpart (later)

T3 <= R × S

Faculty_ID	Faculty_Name
BA	Brian Anderson
JC	John Chase
KT	Keith Tang

Faculty_ID	Faculty_Name	Faculty_ID	Team_ID
BA	Brian Anderson	BA	01
BA	Brian Anderson	JC	02
BA	Brian Anderson	JC	03
JC	John Chase	BA	01
JC	John Chase	JC	02
JC	John Chase	JC	03
KT	Keith Tang	BA	01
KT	Keith Tang	JC	02
KT	Keith Tang	JC	03

Faculty_ID	Team_ID
BA	01
JC	02
JC	03

Slide 19

$$T7 <= R \bowtie_{R.Faculty\_ID=S.Faculty\_ID} S$$

Faculty_ID	Faculty_Name	Faculty_ID	Team_ID
BA	Brian Anderson	BA	01
BA	Brian Anderson	JC	02
BA	Brian Anderson	JC	03
JC	John Chase	BA	01
JC	John Chase	JC	02
JC	John Chase	JC	03
KT	Keith Tang	BA	01
KT	Keith Tang	JC	02
KT	Keith Tang	JC	03

Faculty_ID	Faculty_Name
BA	Brian Anderson
JC	John Chase
KT	Keith Tang

Faculty_ID	Team_ID
BA	01
JC	02
JC	03

Keith Tang   COMP2714   19

Slide 20

$$T7 <= R \bowtie_{R.Faculty\_ID=S.Faculty\_ID} S$$

Faculty_ID	Faculty_Name	Faculty_ID	Team_ID
BA	Brian Anderson	BA	01
JC	John Chase	JC	02
JC	John Chase	JC	03

Faculty_ID	Faculty_Name
BA	Brian Anderson
JC	John Chase
KT	Keith Tang

Faculty_ID	Team_ID
BA	01
JC	02
JC	03

Keith Tang   COMP2714   20