Chapter 4A.

Relational Algebra I

CSIS0278 / COMP3278

Introduction to

Database Management Systems

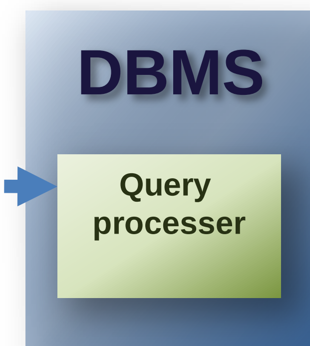
Department of Computer Science, The University of Hong Kong

Slides prepared by - **Dr. Chui Chun Kit**, http://www.cs.hku.hk/~ckchui/ For other uses, please email : ckchui@cs.hku.hk

Motivation

- (Query 5 in Chapter 3B) Find the dept. names where employees named Smith work.

```
SELECT D.name
FROM Employees E, Works_in W, Departments D
WHERE E.name = 'Smith'  AND
        E.employee_id = W.employee_id AND
        W.department_id = D.department_id;
```



How does the DBMS execute this SQL query?
Join which two tables first?
Which **constraint** is applied first?

Employees

employee_id	name	salary
1	Jones	26000
2	Smith	28000
3	Parker	35000
4	Smith	24000

Works_in

employee_id	department_id	since
1	1	2001-1-1
2	1	2002-4-1
2	2	2005-2-2
3	3	2003-1-1
4	3	2005-1-1

Departments

department_id	name	budget
1	Toys	122000
2	Tools	239000
3	Food	100000

Relational Algebra

- **Relational Algebra** is similar to normal algebra (as in 2+3*x-y), except it uses **relations (tables)** as **operand**, and a new set of **operators**.
- The inner, lower-level operations of a relational DBMS are (or are similar to) relational algebra operations.
- **We need to know about relational algebra to understand query execution and optimization in a relational DBMS.**

Section 1

Basic operators

Basic operators

- Select (σ)
- Project (π)
- Union (∪)
- Set difference (-)
- Cartesian product (×)
- Rename (ρ)

That is to say, there should be programs (or functions) implemented in the DBMS for each of these relational operators

5

Selection

- $\sigma_p(R) = \{ t \mid t \in R \wedge p(t) \}$
- Example
  - Consider the relation **Author** (<u>authorID</u>, name, date of birth)
  - Select all authors called “May”.

```
SELECT *
FROM Author
WHERE name = "May";
```

SQL

→ Query processer → $\sigma_{name=\text{“May”}}(\textbf{Author})$

Relational algebra

Author

authorID	name	date of birth
101	May	Nov 16
102	Bonnie	Jan 15
103	May	Jul 11
104	Raymond	Apr 30
105	Tiffany	Oct 10

$\sigma_{name=\text{“May”}}(\textbf{Author})$

authorID	name	date of birth
101	**May**	Nov 16
103	**May**	Jul 11

Projection

- $\pi_{A1, A2, \ldots, Ak}(R)$
  - A copy of R with only listed attributes A1 to Ak.
- Example
  - Consider the relation **Book** (<u>bookID</u>, title, publisher)
  - Report only the **bookID** and **title** of all the books.

```
SELECT bookID, title
FROM Book;
```

SQL

Query processer

$\pi_{bookID,\ title}(\textbf{Book})$

Relational algebra

Book

bookID	title	publisher
115	Stuffy doll	ABC
116	Angel's feather	MTG
117	Little girl	MGH
118	Myr	ABC

$\pi_{bookID,\ title}(\textbf{Book})$

bookID	title
115	Stuffy doll
116	Angel's feather
117	Little girl
118	Myr

Union

- $R \cup S = \{ t \mid t \in R \vee t \in S \}$
  - R and S must have the **same number of attributes** and **attribute domains compatible**.
- Example
  - Find the name of all products in Audio_CD and DVD tables.

```
SELECT name
FROM Audio_CD
UNION
SELECT name
FROM DVD;
```

SQL

Query processer

$$\pi_{name}\,(\textbf{Audio\_CD}) \cup \pi_{name}\,(\textbf{DVD})$$

Relational algebra

Audio_CD

name	#tracks
One Heart	14
Miracle	14

DVD

name	length	subtitle
Prince of Persia	110	English, Chinese
Villon’s Wife	90	Japanese
Legend is born: Ip Man	90	Chinese

Union

π_{name} (Audio_CD) $\cup$ π_{name} (DVD)

name
One Heart
Miracle
Prince of Persia
Villon's Wife
Legend is born: Ip Man

Note that the left and right part of the Union operator has to be **comparable**. (i.e. same **type**, same number of attributes)

π_{name} (Audio_CD)

name
One Heart
Miracle

π_{name} (DVD)

name
Prince of Persia
Villon's Wife
Legend is born: Ip Man

Audio_CD

name	#tracks
One Heart	14
Miracle	14

DVD

name	length	subtitle
Prince of Persia	110	English, Chinese
Villon's Wife	90	Japanese
Legend is born: Ip Man	90	Chinese

Set difference

- $R - S = \{ t \mid t \in R \wedge t \notin S \}$
  - R and S must have the **same number of attributes** and **attribute domains compatible**.
- Example
  - Find the ID of the students who haven't submitted the assignment.

```sql
SELECT student_id
FROM Student
EXCEPT
SELECT student_id
FROM Submit;
```

SQL

→ Query processer →

$\pi_{student\_id}(\textbf{Student}) - \pi_{student\_id}(\textbf{Submit})$

Relational algebra

Student

student_id	name	gender	major
123	Kit	M	CS
456	Yvonne	F	CS
789	Paul	M	CS

Submit

student_id	assignment_id	date
456	1	28/9
789	1	25/9

Set difference

$\pi_{student\_id}(\textbf{Student}) - \pi_{student\_id}(\textbf{Submit})$

student_id
123

Note that the left and right part of the Set difference operator has to be **comparable**. (i.e. same **type**, same number of attributes)

$\pi_{student\_id}(\textbf{Student})$

student_id
123
456
789

$\pi_{student\_id}(\textbf{Submit})$

student_id
456
789

Student

student_id	name	gender	major
123	Kit	M	CS
456	Yvonne	F	CS
789	Paul	M	CS

Submit

student_id	assignment_id	date
456	1	28/9
789	1	25/9

Cartesian product

- $R \times S = \{ t\,q \mid t \in R \wedge q \in S \}$
  - No attributes with a common name in R and S.
- Example
  - Display the date of the tutorials of the course “Introduction to Database Management Systems”.

```
SELECT Tutorial.date
FROM Course, Tutorial
WHERE
Coruse.name=“Introduction to
Database Management Systems” AND
Course.course_id = Tutorial.course_id;
```

SQL

Query processer

$$
\pi_{\text{Tutorial.date}}\Big(\sigma_{\text{Course.name=“Introduction to Database Management Systems”}}\big(\sigma_{\text{Course.course\_id=Tutorial.course\_id”}}(\textbf{Course} \times \textbf{Tutorial})\big)\Big)
$$

Relational algebra

Course

course_id	name
c1119	Data Structures and Algorithms
c0278a	Introduction to Database Management Systems

Tutorial

tutorial_id	course_id	date
1	c1119	5/9
1	c0278a	7/9
2	c0278a	15/9

Cartesian product

Course × Tutorial

Course.course_id	Course.name	Tutorial .tutorial_id	Tutorial .course_id	Tutorial .date
c1119	Data Structures and Algorithms	1	c1119	5/9
c1119	Data Structures and Algorithms	1	c0278a	7/9
c1119	Data Structures and Algorithms	2	c0278a	15/9
c0278a	Introduction to Database Management Systems	1	c1119	5/9
c0278a	Introduction to Database Management Systems	1	c0278a	7/9
c0278a	Introduction to Database Management Systems	2	c0278a	15/9

Course

course_id	name
c1119	Data Structures and Algorithms
c0278a	Introduction to Database Management Systems

Tutorial

tutorial_id	course_id	date
1	c1119	5/9
1	c0278a	7/9
2	c0278a	15/9

Cartesian product

Course × Tutorial

Course.course_id	Course.name	Tutorial .tutorial_id	Tutorial .course_id	Tutorial .date
c1119	Data Structures and Algorithms	1	c1119	5/9
c1119	Data Structures and Algorithms	1	c0278a	7/9
c1119	Data Structures and Algorithms	2	c0278a	15/9
c0278a	Introduction to Database Management Systems	1	c1119	5/9
c0278a	Introduction to Database Management Systems	1	c0278a	7/9
c0278a	Introduction to Database Management Systems	2	c0278a	15/9

$\sigma_{Course.course\_id=Tutorial.course\_id}$ **(Course × Tutorial)**

Course.course_id	Course.name	Tutorial .tutorial_id	Tutorial .course_id	Tutorial .date
c1119	Data Structures and Algorithms	1	c1119	5/9
c0278a	Introduction to Database Management Systems	1	c0278a	7/9
c0278a	Introduction to Database Management Systems	2	c0278a	15/9

Cartesian product

$\pi_{\text{Tutorial.date}}\ ($
$\sigma_{\text{Course.name="Introduction to Database Management Systems"}}($
$\sigma_{\text{Course.course\_id=Tutorial.course\_id"}}\ (\textbf{Course} \times \textbf{Tutorial})$
$)$
$)$

Tutorial.date
7/9
15/9

$\sigma_{\text{Course.name="Introduction to Database Management Systems"}}(\ \sigma_{\text{Course.course\_id=Tutorial.course\_id"}}\ (\textbf{Course} \times \textbf{Tutorial}))$

Course.course_id	Course.name	Tutorial .tutorial_id	Tutorial .course_id	Tutorial .date
c0278a	Introduction to Database Management Systems	1	c0278a	7/9
c0278a	Introduction to Database Management Systems	2	c0278a	15/9

$\sigma_{\text{Course.course\_id=Tutorial.course\_id}}\ (\textbf{Course} \times \textbf{Tutorial})$

Course.course_id	Course.name	Tutorial .tutorial_id	Tutorial .course_id	Tutorial .date
c1119	Data Structures and Algorithms	1	c1119	5/9
c0278a	Introduction to Database Management Systems	1	c0278a	7/9
c0278a	Introduction to Database Management Systems	2	c0278a	15/9

Rename

- Notation: $\rho_X(E)$
  - Rename operator allows us to name and refer to the results of relational-algebra expressions
- $\rho_X(E)$ returns the expression E under the name X

```
SELECT Tutorial.date
FROM Course, Tutorial
WHERE
Coruse.name="Introduction to Database
Management Systems" AND
Course.course_id = Tutorial.course_id;
```

SQL

$$\pi_{\text{Tutorial.date}}(\sigma_{\text{Course.name="Introduction to Database Management Systems"}}(\sigma_{\text{Course.course\_id=Tutorial.course\_id"}}(\textbf{Course} \times \textbf{Tutorial})))$$

Relational algebra (Without rename)

```
SELECT T.date
FROM Course C, Tutorial T
WHERE
C.name="Introduction to Database
Management Systems" AND
C.course_id = T.course_id;
```

SQL

$$\pi_{\textbf{T}\text{.date}}(\sigma_{\textbf{C}\text{.name="Introduction to Database Management Systems"}}(\sigma_{\textbf{C}\text{.course\_id=}\textbf{T}\text{.course\_id"}}(\rho_{\textbf{C}}(\textbf{Course}) \times \rho_{\textbf{T}}(\textbf{Tutorial}))))$$

Relational algebra (With Course rename to C, Tutorial rename to T)

Section 2

Exercises

Question 1

- Given the following relational schema:
  - **Student** (UID, name, age).
  - **Course** (CID, title).
  - **Enroll** (UID, CID) with UID referencing **Student** and CID referencing **Course**.

*UID and CID, age are **interger; name** and **title** are **varchar**.

- Which of the following is (are) valid Relational Algebra expression(s)?
  - π_{UID}(**Student**)-π_{CID}(**Course**) ✔
  - **Course**-π_{UID}(**Enroll**) ✗

The left and right parts of Set difference have to be **comparable** (same number of attributes).

Question 1

- Given the following relational schema:
  - **Student** (<u>**UID**</u>, **name**, **age**).
  - **Course** (<u>**CID**</u>, **title**).
  - **Enroll** (<u>**UID**</u>, <u>**CID**</u>) with **UID** referencing **Student** and **CID** referencing **Course**.

*****UID** and **CID**, **age** are **interger; name** and **title** are **varchar**.

- Which of the following is (are) valid Relational Algebra expression(s)?
  - $\sigma_{age<18}(\textbf{Student} \cup \textbf{Course})$ ✗ Student and Course have different attributes.
  - $\sigma_{age<18}(\pi_{UID,\ name}(\textbf{Student}))$ ✗ No attribute “age” for selection.

Question 2

```
SELECT employee_id,name
FROM Employee
WHERE employee_id < 100;
```

SQL

- Find the employeeIDs and names of employees whose employeeID < 100.

Employee

employeeID	name	start_date
97	May	Nov 16, 1997
98	Felix	Jun 30, 2003
99	May	Sep 18, 2007
100	George	Jan 1, 2008

Employee → Query processer (Option 1) | Employee → Query processer (Option 2)

Option 1

$\sigma_{employeeID<100}$(**Employee**)

employeeID	name	start_date
97	May	Nov 16, 1997
98	Felix	Jun 30, 2003
99	May	Sep 18, 2007

$\pi_{name,\ employeeID}(\sigma_{employeeID<100}$(**Employee**))

employeeID	name
97	May
98	Felix
99	May

Option 2

$\pi_{name,\ employeeID}$(**Employee**)

employeeID	name
97	May
98	Felix
99	May
100	George

$\sigma_{employeeID<100}(\pi_{name,\ employeeID}$(**Employee**))

employeeID	name
97	May
98	Felix
99	May

Which one is better?

Question 3

- (Query 4 in Chapter 3B) Find the dept. id(s) where employees named Smith work.

Option 1

$$\pi_{W.department\_id}(\sigma_{E.name=\text{“Smith”}}(\sigma_{E.employee\_id=W.employee\_id}(\rho_E(\textbf{Employees}) \times \rho_W(\textbf{Works\_in}))))$$

Now we compute the Cartesian product between **Employees** and **Works_in** first, which creates an **intermediate relation** with 10,000*5 = 50,000 tuples! Can we reduce the size of intermediate relation?

Employees

employee_id	name	salary
1	Jones	26000
2	Smith	28000
…	…	…
10000	…	…

Employees (10000 tuples)

Works_in

employee_id	department_id	since
1	1	2001-1-1
2	1	2002-4-1
2	2	2005-2-2
3	3	2003-1-1
4	3	2005-1-1

Works_in (5 tuples)

```
SELECT W.department_id
FROM Employees E, Works_in W
WHERE E.name = 'Smith' AND
        E.employee_id = W.employee_id;
```

Question 3

- (Query 4 in Chapter 3B) Find the dept. id(s) where employees named Smith work.

Option 1

$$\pi_{W.department\_id}(\sigma_{E.name=\text{"Smith"}}(\sigma_{E.employee\_id=W.employee\_id}(\rho_E(\textbf{Employees}) \times \rho_W(\textbf{Works\_in}))))$$

Option 2

$$\pi_{W.department\_id}(\sigma_{E.employee\_id=W.employee\_id}(\sigma_{E.name=\text{"Smith"}}(\rho_E(\textbf{Employees})) \times \rho_W(\textbf{Works\_in})))$$

We apply selection on Employees before the Cartesian product. If there is only two employee named Smith, the **intermediate relation** can be reduced to 2*5 = 10 tuples!

Employees

employee_id	name	salary
1	Jones	26000
2	Smith	28000
…	…	…
10000	…	…

Employees (10000 tuples)

Works_in

employee_id	department_id	since
1	1	2001-1-1
2	1	2002-4-1
2	2	2005-2-2
3	3	2003-1-1
4	3	2005-1-1

Works_in (5 tuples)

```
SELECT W.department_id
FROM Employees E, Works_in W
WHERE E.name = 'Smith' AND
      E.employee_id = W.employee_id;
```

Banking example

Foreign key

Foreign key

Branch

branch_id	name	asset
B1	Central	7100000
B2	Causeway Bay	9000000
B3	Aberdeen	400000
B4	North Point	3700000

Loan

branch_id	loan_id	amount
B3	L1	900
B1	L2	1500
B1	L3	1000

Borrower

customer_id	loan_id
C1	L3
C4	L2
C2	L1

Foreign key

Foreign key

Foreign key

Account

branch_id	account_id	balance
B1	A1	500
B2	A2	400
B2	A3	900
B1	A4	700

Depositor

account_id	customer_id
A1	C1
A1	C2
A2	C2
A3	C4
A4	C4

Foreign key

Customer

customer_id	name	address
C1	Kit	CB320
C2	Ben	CB326
C3	Jolly	CB311
C4	Yvonne	CB415

Question 4

- Find the names of all customers who have a loan, an account, or both in a bank.

```
SELECT customer_id
FROM Borrower
UNION
SELECT customer_id
FROM Depositor
```

$$\pi_{customer\_id}\ (Borrower) \cup \pi_{customer\_id}\ (Depositor)$$

- Find the names of all customers who have both a loan and an account in a bank.

```
SELECT customer_id
FROM Borrower
INTERSECT
SELECT customer_id
FROM Depositor
```

$$\pi_{customer\_id}\ (Borrower) \cap \pi_{customer\_id}\ (Depositor)$$

Wait! Do we have set intersection in relational algebra ☺?

$$\pi_{customer\_id}\ (Borrower) - (\pi_{customer\_id}\ (Borrower) - \pi_{customer\_id}\ (Depositor))$$

Additional operators

- These operations do not add any power to relational algebra, but simplify common queries.
  - **Set intersection** ($\cap$)
  - **Natural join** ($\bowtie$)
  - **Division** ($\div$)
  - **Assignment** ($\leftarrow$)

Chapter 4A.

END

CSIS0278 / COMP3278

Introduction to

Database Management Systems

Slides prepared by - **Dr. Chui Chun Kit**, http://www.cs.hku.hk/~ckchui/ for students in CSIS0278 / COMP3278

For other uses, please email : ckchui@cs.hku.hk