Relational Algebra

SWEN304/SWEN435

Lecturer: Dr Hui Ma

Engineering and Computer Science

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

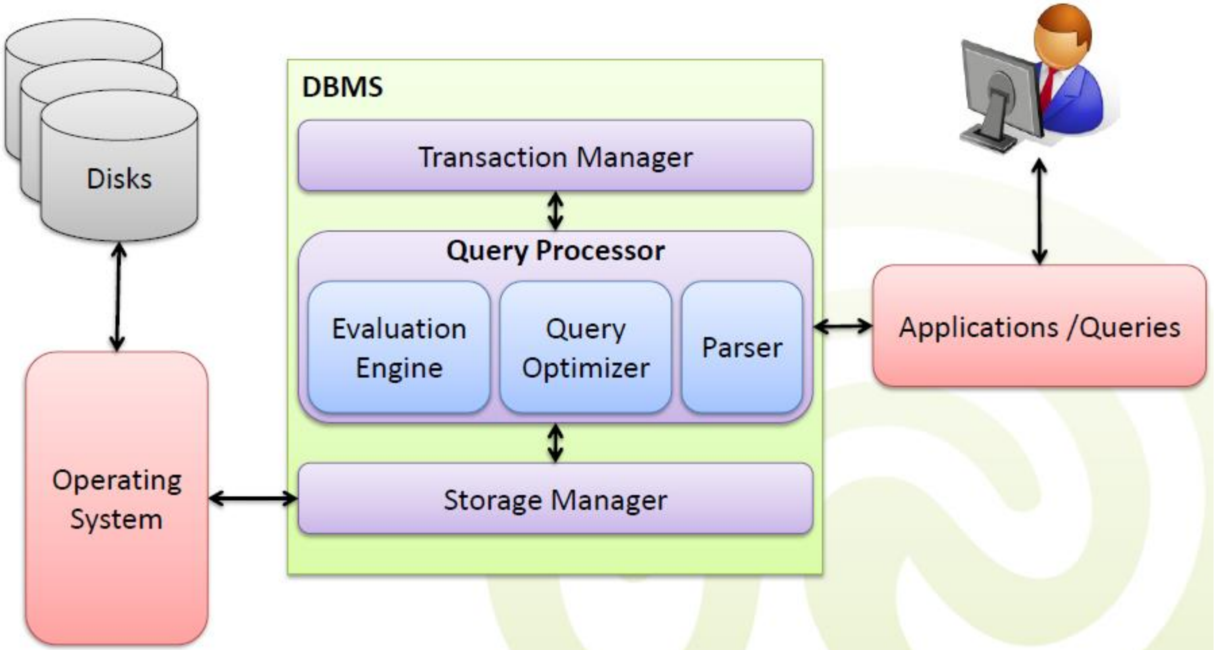
Outline

- Basic relational algebra operations
- Set theoretic operations
- Additional operations

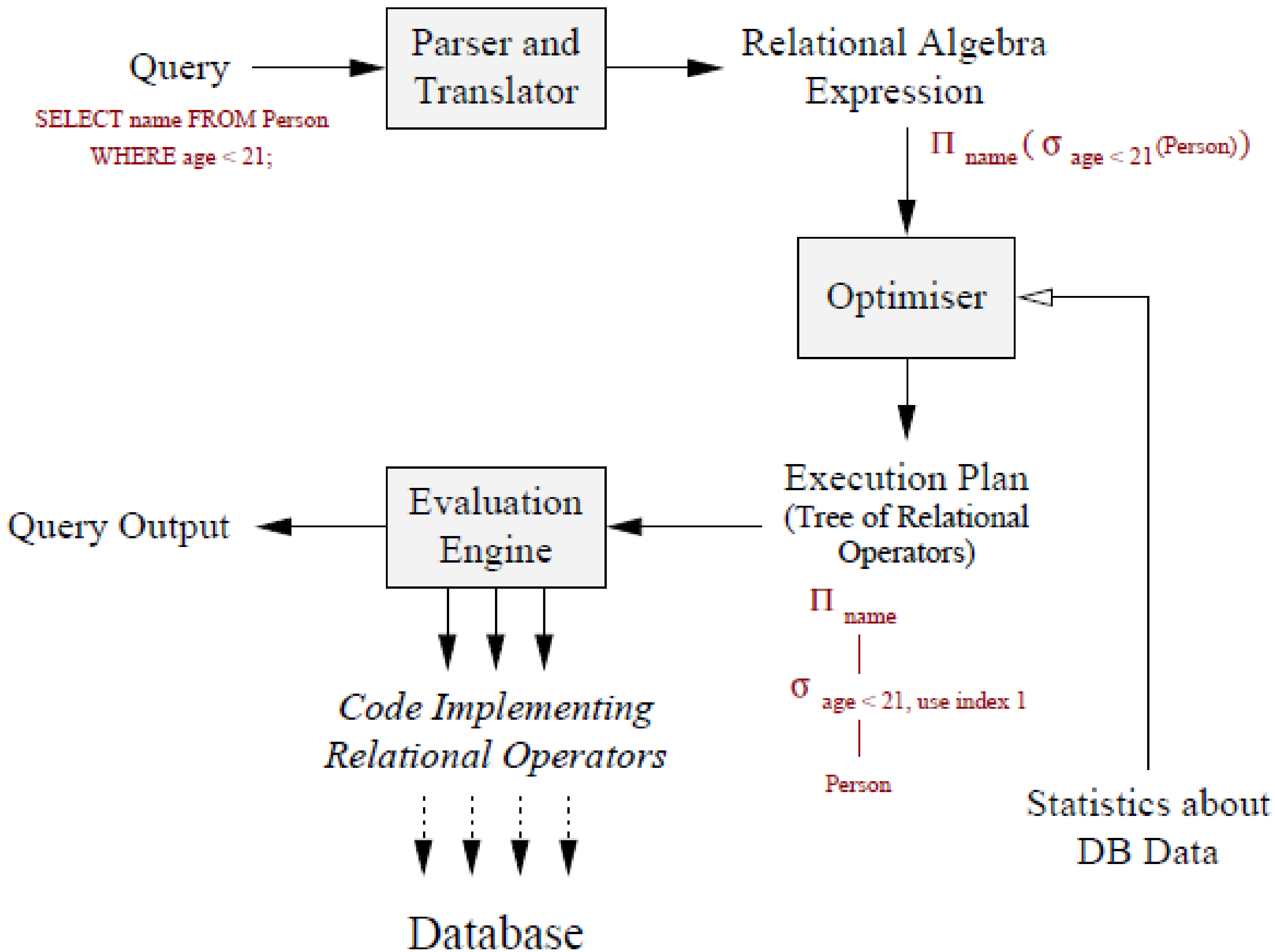
- *Reading: Chapters 6 of the textbook*

Query Processing in DBMS

- Users/applications submit queries to the DBMS
- The DBMS processes queries before evaluating them
  - Recall: DBMS mainly use declarative query languages (such as SQL)
  - Queries can often be evaluated in different ways
  - SQL queries do not determine how to evaluate them

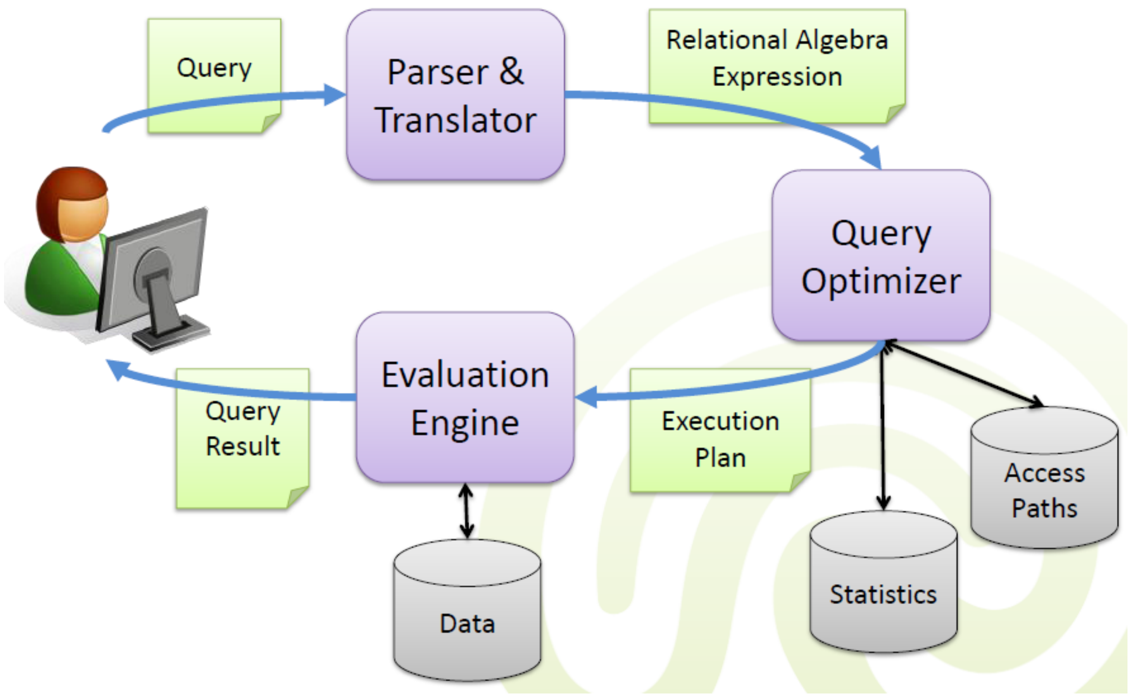


Query Processing in DBMS[4,5]



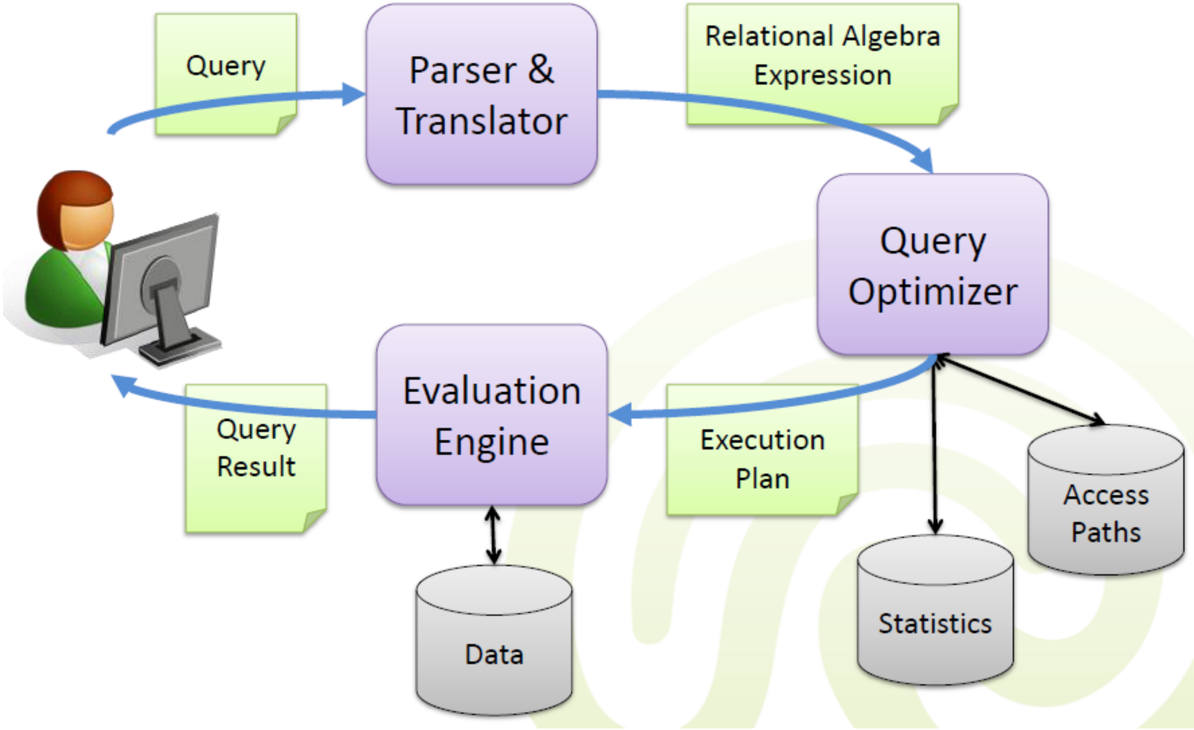
Query Processing in DBMS

- The parser checks the syntax, e.g., verifies table names, data types
  - A scanner tokenizes the query (tokens for SQL commands, names, ...)
  - Either the query is executable or an error message is generated
    - (SQLCODE/SQLSTATE)



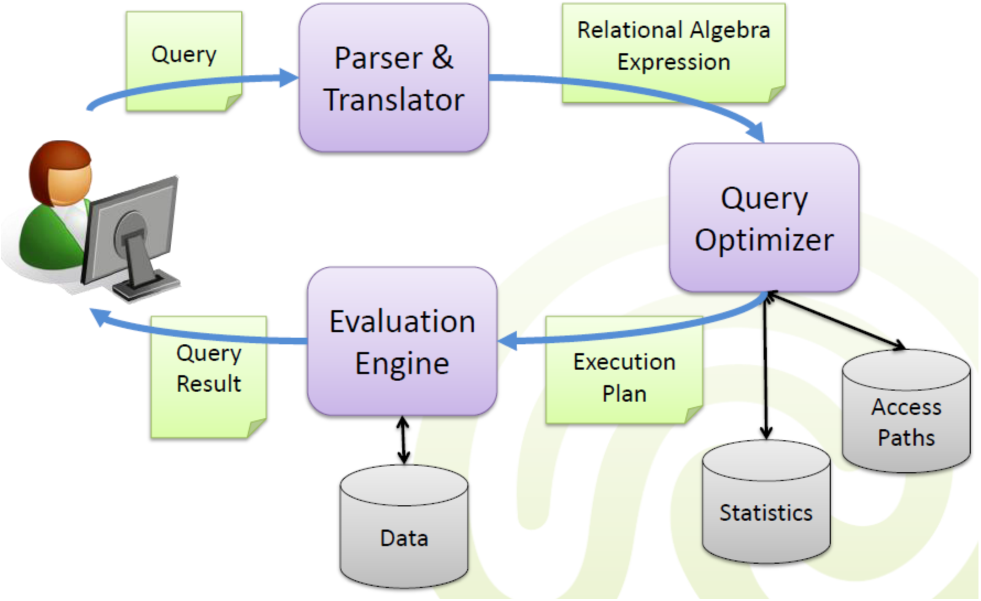
Query Processing in DBMS

- The translator translates the query into relational algebra
  - Internal exchange format between DBMS components
  - Allows for symbolic calculation



Query Processing in DBMS

- Relational Algebra was introduced by Codd (1970) with the relational data model
  - Provides formal foundations for relational model operations
  - Used as basis for implementing and optimizing queries in RDBMSs
  - Some of the concepts are incorporated into the SQL standard query language



Relational Algebra

- A set of operations to manipulate (query and update) a relational database
  - Operations are applied onto relations
  - The result is a new relation
- Basic operations:
  - project , select, rename, and join
- Set theoretic operations:
  - union, intersect, set difference,
  - Cartesian product
- Additional relational operations:
  - aggregate operations (SUM, COUNT, AVERAGE), grouping, and
  - outer join

A Sample Relational Database

Student

Lname	Fname	StudId	Major
Smith	Susan	131313	Comp
Bond	James	007007	Math
Smith	Susan	555555	Comp
Cecil	John	010101	Math

Course

Cname	CourId	Points	Dept
DB Systems	C302	15	Comp
Software Engineering	C301	15	Comp
Discrete Math	M214	22	Math
Programmes	C201	22	Comp

Grades

StudId	CourId	Grad
007007	C302	A+
555555	C302	ω
007007	C301	A
007007	M214	A+
131313	C201	B-
555555	C201	C
131313	C302	ω
007007	C201	A
010101	C201	ω

Project Operation

- Notation: $\pi_{AL}(N)$

  where AL is a subset of attributes from R in $N\ (R, C)$.

  Note: for simplicity we also use N to refer to relation r over N
- Project operation produces a new relation by retaining columns in AL and dropping all the others
- If $AL = (A_l, \ldots, A_k)$, then $\pi_{AL}(N) = N[A_l, \ldots, A_k]$
- Example: StudentName= $\pi_{\text{LName, FName}}$(Student):

StudentName

LName	FName
Smith	Susan
Bond	James
Cecil	John

Select Operation

- It is used to select such a subset of tuples from a relation that satisfies a given condition
- Notation: $\sigma_c(N)$
  - Condition c is a Boolean expression on attributes of R in $N(R, C)$
  - Boolean expression is made up of clauses of the form $A\ \theta\ a$ or $A\ \theta\ B$, where
    - $a \in dom(A)$,
    - $\theta \in \{ =, <, >, \leq, \geq, \neq \}$, and
    - $A, B \in R$
  - Clauses can be connected by Boolean operators $\neg, \wedge, \vee$ to form new clauses

Select Operation: Examples

Student

Lname	Fname	StudId	Major
Smith	Susan	131313	Comp
Bond	James	007007	Math
Smith	Susan	555555	Comp
Cecil	John	010101	Math

- Student2 = $\sigma_{\text{StudId} = 007007}$(Student)

Student2

LName	FName	StudId	Major
Bond	James	007007	Math

Student3 = $\sigma_{\text{FName} = \text{'Susan'}}$ (Student)

Student3

LName	FName	StudId	Major
Smith	Susan	131313	Comp
Smith	Susan	555555	Comp

Numeric Properties of Select and Project

- Since we want to use relational algebra expressions in query optimization, we need numeric properties of relational algebra operations
- Relation $\pi_{AL}(N)$ is produced from relation N by retaining columns in AL and dropping duplicate tuples, hence:
  - $degree(\pi_{AL}(N)) = |AL| \leq |R|$ (number of attributes)
  - $|\pi_{AL}(N)| \leq |N|$ (number of tuples)
- Relation $\sigma_C(N)$ contains those tuples of $r(N)$ that evaluate true for C, hence:
  - $degree(\sigma_C(N)) = degree(N)$ (number of attributes)
  - $\sigma_C(N) \subseteq N$ and $|\sigma_C(N)| \leq |N|$ (number of tuples)

Combining Select and Project Operators

- $\pi_{AL}(\sigma_C(N))$ or $\sigma_C(\pi_{AL}(N))$
- For example,

$$\text{Student4} = \pi_{\text{FName, LName}}(\sigma_{\text{StudId} = 007007}(\text{Student}))$$

In SQL:

```
SELECT FName, LName
FROM Student
WHERE StudentId = 007007;
```

Student

Lname	Fname	StudId	Major
Smith	Susan	131313	Comp
Bond	James	007007	Math
Smith	Susan	555555	Comp
Cecil	John	010101	Math

Student4

FName	LName
James	Bond

Rename Operation

- Notation: $\rho_{A1\to B1,\dots,Ak\to Bk}(N)$

  with $dom(B_i) = dom(A_i)$ for $i = 1, \dots, k$
- A unary operation defined on relations $r(N)$ with $A_1, \dots, A_k \in R$
- schema: $(R - \{A_1, \dots, A_k\}) \cup \{B_1, \dots, B_k\}$
- Example: $\rho_{\text{FName}\to\text{FirstName},\text{LName}\to\text{LastName}}(\text{Student4})$
- In SQL:

```
SELECT FName As FirstName, LName As LastName
FROM Student4;
```

Student

Lname	Fname	StudId	Major
Smith	Susan	131313	Comp
Bond	James	007007	Math
Smith	Susan	555555	Comp
Cecil	John	010101	Math

Student5

FirstName	LastName
James	Bond

Join Operation

- Join operation merges those tuples from two relations that satisfy a given condition
  - The condition is defined on attributes belonging to both of the relations to be joined
- Theta, equi, and natural join operations
- Theta, equi, and natural join are collectively called INNER joins
- In each of inner joins, tuples with null valued join attributes do not appear in the result
- OUTER joins include tuples with null valued join attributes into the result

Theta Join Operation

- Notation: $N = N_1 \bowtie_{JC} N_2$
  - N is the result of joining relation N_1 over $N_1(R_1, C_1)$ with relation N_2 over $N_2(R_2, C_2)$
  - Join condition $JC = jc_1 \wedge \ldots \wedge jc_n$
    - $jc_i = A \theta B, A \in R_1, B \in R_2,$
    - $\theta \in \{=, \neq, <, >, \leq, \geq\},$
    - $Dom(N_1, A) \subseteq Dom(N_2, B),$
    - $Range(N_1, A) \subseteq Range(N_2, B)$
  - $R_1 = \{A_1, \ldots, A_m\}, R_2 = \{B_1, \ldots, B_n\},$
    $R = \{A_1, \ldots, A_m, B_1, \ldots, B_n\}$
  - $degree(R) = degree(R_1) + degree(R_2)$
  - $|N| \leq |N_1| \times |N_2|$

Equijoin Operation

- A special case of the theta join, when $\theta \in \{=\}$
  - Notation: $N = N_1 \bowtie_{JC} N_2$

    where $JC = jc_1 \wedge \ldots \wedge jc_n$

    $jc_i \equiv A = B,\ A \in R_1,\ B \in R_2,$

Student

Lname	Fname	StudId	Major
Smith	Susan	131313	Comp
Bond	James	007007	Math
Smith	Susan	555555	Comp
Cecil	John	010101	Math

Grades

StudId	CourId	Grad
007007	C302	A+
555555	C302	ω
007007	C301	A
007007	M214	A+
131313	C201	B-
555555	C201	C
131313	C302	ω
007007	C201	A
010101	C201	ω

- For example,

  $\text{Student} \bowtie_{\text{StudId} = \text{StudId}} \text{Grades}$

  In SQL:

```
SELECT *
FROM Student s, Grades g
WHERE s.StudId = g.StudId;
```

Equijoin Operation: Example

Superfluous column

Student_Grades

Lname	Fname	StudId	StudId	Major	CourId	Grade
Smith	Susan	131313	131313	Comp	C201	B-
Smith	Susan	131313	131313	Comp	C302	ω
Bond	James	007007	007007	Math	C302	A+
Bond	James	007007	007007	Math	C301	A
Bond	James	007007	007007	Math	M214	A+
Bond	James	007007	007007	Math	C201	A
Smith	Susan	555555	555555	Comp	C201	C
Smith	Susan	555555	555555	Comp	C302	ω
Cecil	John	010101	010101	Math	C201	ω

Natural Join Operation

- A special case of an equijoin operation, when join attributes have the same name ($N_1.X = N_2.X$)
  - Notation: $N = N_1 * N_2$
  - Formal definition:

$$N_1 * N_2 = \{\ t\ [R_1 \cup R_2] \mid t\ [R_1] \in N_1 \wedge t\ [R_2] \in N_2 \}$$

  - $degree(r\ ) = degree(r_1) + degree(r_2) - |X\ |$ (number of attributes)
  - $0 \leq |N_1 * N_2| \leq |N_1| \bullet |N_1|$ (number of tuples)
  - }

where $|N_i|$ denotes the number of elements in a relation N_i

Natural Join Operation: Example

- Query: Retrieve information of students and their grades
- Relational Algebra:

  Student * Grades

- In SQL:

```
SELECT * FROM Student NATURAL JOIN Grades;
```

Natural Operation: Example

Student * Grades

Lname	Fname	StudId	Major	CourId	Grade
Smith	Susan	131313	Comp	C201	B-
Smith	Susan	131313	Comp	C302	ω
Bond	James	007007	Math	C302	A+
Bond	James	007007	Math	C301	A
Bond	James	007007	Math	M214	A+
Bond	James	007007	Math	C201	A
Smith	Susan	555555	Comp	C201	C
Smith	Susan	555555	Comp	C302	ω
Cecil	John	010101	Math	C201	ω

Set Theoretic Operations

- Union, Intersect, Difference, Cartesian product

$$N = N_1 \, \Theta \, N_2$$

  where $R_1 = (A_1, \ldots, A_n)$, $R_2 = (B_1, \ldots, B_m)$ are lists of attributes, and

  $\Theta \in \{ \cup, \cap, -, \times \}$

- i.e.
  - $N = N_1 \cup N_2$
  - $N = N_1 \cap N_2$
  - $N = N_1 \times N_2$
  - $N = N_1 - N_2$

Set Theoretic Operations

- For union, intersect and difference, attribute sets R_1 and R_2 have to be union compatible:
  - $|R_1| = |R_2|$,
  - $(\forall i \in \{1,\ldots, n\})(Dom(N_1, A_i) = Dom(N_2, B_i))$, and
  - $(\forall i \in \{1,\ldots, n\})(Range(N_1, A_i) = Range(N_2, B_i))$

- For cartesian product

  $R = R_1 \cup R_2$,

  $degree(N_1 \times N_2) = degree(r(N_1)) + degree(N_2)$,

  $| N_1 \times N_2 | = |N_1| \bullet |N_2|$

Question For You

- Consider the following relations

N_1

A	B
1	2
3	3
4	4

N_2

B	C
2	7
4	9
ω	0

- How many tuples will the Cartesian product $N_1 \times N_2$ return?

a) 6

b) 9

Question For You

- Consider the following relations

N_1

A	B
1	2
3	3
4	4

N_2

B	C
2	7
4	9
ω	0

- How many tuples will the natural join $N_1 * N_2$ return?

a) 2

b) 6

c) 9

Outer Join

- Introduced to include those tuples that don't match, or contain null values for join attributes into join relation
- Notations:

LEFT: ⟕ RIGHT: ⟖ and FULL outer join: ⟗

- Example:

N_1

A	B
1	2
5	6

$⟕_{B=C}$

N_2

C	D
2	7
2	9
ω	1

=

$N_1 ⟕_{B=C} N_2$

A	B	C	D
1	2	2	7
1	2	2	9
5	6	ω	ω

Relational Algebra & SQL

- Each relational algebra query (except union) can be easily rewritten in SQL (for simplicity: assume global attribute names)
  - attribute selection $\sigma_{A=B}(N)$:

    SELECT $*$ FROM N WHERE $A = B$;
  - constant selection $\sigma_{A=c}(N)$:

    SELECT $*$ FROM N WHERE A = c;
  - projection $\pi_{A1,\ldots,Ak}\ (N)$:

    SELECT DISTINCT $A_1, \ldots, A_k$ FROM N;

Relational Algebra & SQL

- rename $\rho_{A_1 \to B_1, \ldots, A_k \to B_k}(N)$:

  SELECT A_1 AS B_1 ,. . . , A_k AS B_k FROM N ;

- natural join $N_1 * N_2$ (with common attributes A_1, . . . ,A_k):

  SELECT ∗ FROM N_1 NATURAL JOIN N_2;

- equijoin $N_1 \bowtie_{A_1 = B_1, \ldots, A_k = B_k} N_2$:

  SELECT ∗ FROM N_1, N_2 WHERE $N_1.A_1 = N_2.B_1$ AND . . . AND $N_1.A_k = N_2.B_k$;

- difference $N_1 - N_2$:

  SELECT ∗ FROM N_1 EXCEPT SELECT ∗ FROM N_2 ;

Relational Algebra and SQL: Examples

- Project operation:
  - $\pi_{LName,\ FName}$ (Student)
  - SELECT **DISTINCT** LName, FName FROM Student;
- Selection operation:
  - $\sigma_{FName\ =\ 'Susan'}$ (Student)
  - SELECT * FROM Student WHERE FName = 'Susan';

Summary

- Relational Algebra consists of several groups of operations
  - Unary Relational Operations
    - SELECT (symbol: σ (sigma))
    - PROJECT (symbol: π (pi))
    - RENAME (symbol: ρ (rho))
  - Binary Relational Operations
    - JOIN (several variations of JOIN exist)
  - Relational Algebra Operations From Set Theory
    - UNION ($\cup$), INTERSECTION ($\cap$), DIFFERENCE (or MINUS, **–**)
    - CARTESIAN PRODUCT (**x**)
  - Additional Relational Operations
    - OUTER JOINS,
    - AGGREGATE FUNCTIONS (These compute summary of information: for example, SUM, COUNT, AVG, MIN, MAX)

References

1. Elmasri, Navathe. Fundamentals of database systems. Pearson, 2010
2. Ramakrishnan, Gehrke. Database Management Systems. McGraw-Hill, 2003
3. Silberschatz, Korth, Sudarshan. Database Systems Concepts. McGraw-Hill, 2002
4. Abiteboul, Hull, Vianu. Foundations of Databases. Addison Wesley, 1995
5. Connolly, Begg. Database Systems - A Practical Approach to Design, Implementation, and Management. Addison Wesley, 2002

Next topic

- Query Optimization
  - Heuristic optimization
  - Cost-based optimization
- Readings
  - Chapter 19: Algorithms for Query Processing and Optimization
  - Chapters 17: Disk Storage, Basic File Structures, and Hashing
    (Sections: 13.2, to 13.8)
  - Chapter 18: Indexing Structures for Files
    (Sections: 14.1 to 14.5)
  - File Organization – COMP261