CO527: Advanced Database Systems

The Relational Algebra

Sampath Deegalla
Senior Lecturer, Department of Computer Engineering
University of Peradeniya
dsdeegalla@pdn.ac.lk

11th February 2020

Relational Algebra

- Relational algebra is a query language that allows us to retrieve data from DBs.
- It consists of a set of operations that take one or two relations as input and produce a new relation as output.
- The result of a retrieval is a new relation, which may have been formed from one or more relations. The algebra operations thus produce new relations, which can be further manipulated using operations of the same algebra.
- A sequence of relational algebra operations forms a relational algebra expression, whose result will also be a relation that represents the result of a database query (or retrieval request).

Relational Algebra

- Relational algebra is a query language that allows us to retrieve data from DBs.
- It consists of a set of operations that take one or two relations as input and produce a new relation as output.
- The result of a retrieval is a new relation, which may have been formed from one or more relations. The algebra operations thus produce new relations, which can be further manipulated using operations of the same algebra.
- A sequence of relational algebra operations forms a relational algebra expression, whose result will also be a relation that represents the result of a database query (or retrieval request).

Relational Algebra

- Relational algebra is a query language that allows us to retrieve data from DBs.
- It consists of a set of operations that take one or two relations as input and produce a new relation as output.
- The result of a retrieval is a new relation, which may have been formed from one or more relations. The algebra operations thus produce new relations, which can be further manipulated using operations of the same algebra.
- A sequence of relational algebra operations forms a relational algebra expression, whose result will also be a relation that represents the result of a database query (or retrieval request).

Relational Algebra

- Relational algebra is a query language that allows us to retrieve data from DBs.
- It consists of a set of operations that take one or two relations as input and produce a new relation as output.
- The result of a retrieval is a new relation, which may have been formed from one or more relations. The algebra operations thus produce new relations, which can be further manipulated using operations of the same algebra.
- A sequence of relational algebra operations forms a relational algebra expression, whose result will also be a relation that represents the result of a database query (or retrieval request).

Relational Algebra

- Relational algebra is a query language that allows us to retrieve data from DBs.
- It consists of a set of operations that take one or two relations as input and produce a new relation as output.
- The result of a retrieval is a new relation, which may have been formed from one or more relations. The algebra operations thus produce new relations, which can be further manipulated using operations of the same algebra.
- A sequence of relational algebra operations forms a relational algebra expression, whose result will also be a relation that represents the result of a database query (or retrieval request).

The Select Operation

- The SELECT operation is used to select a subset of the tuples from a relation that satisfy a selection condition.
- It is a filter that keeps only those tuples that satisfy a qualifying condition.
- Those satisfying the condition are selected while others are not included in the result.
- In general, the select operation is denoted by

$$\sigma_{<\text{selection condition}>}(R)$$

where σ (sigma) denotes the select operator, and the selection condition is a Boolean expression specified on the attributes of relation R.

The Select Operation

- The SELECT operation is used to select a subset of the tuples from a relation that satisfy a selection condition.
- It is a filter that keeps only those tuples that satisfy a qualifying condition.
- Those satisfying the condition are selected while others are not included in the result.
- In general, the select operation is denoted by

$$\sigma_{<\text{selection condition}>}(R)$$

where σ (sigma) denotes the select operator, and the selection condition is a Boolean expression specified on the attributes of relation R.

The Select Operation

- The SELECT operation is used to select a subset of the tuples from a relation that satisfy a selection condition.
- It is a filter that keeps only those tuples that satisfy a qualifying condition.
- Those satisfying the condition are selected while others are not included in the result.
- In general, the select operation is denoted by

$$\sigma_{<\text{selection condition}>}(R)$$

where σ (sigma) denotes the select operator, and the selection condition is a Boolean expression specified on the attributes of relation R.

The Select Operation

- The SELECT operation is used to select a subset of the tuples from a relation that satisfy a selection condition.
- It is a filter that keeps only those tuples that satisfy a qualifying condition.
- Those satisfying the condition are selected while others are not included in the result.
- In general, the select operation is denoted by

$$\sigma_{<\text{selection condition}>}(R)$$

where σ (sigma) denotes the select operator, and the selection condition is a Boolean expression specified on the attributes of relation R.

The Select Operation

- The SELECT operation is used to select a subset of the tuples from a relation that satisfy a selection condition.
- It is a filter that keeps only those tuples that satisfy a qualifying condition.
- Those satisfying the condition are selected while others are not included in the result.
- In general, the select operation is denoted by

$$\sigma_{<\text{selection condition}>}(R)$$

where σ (sigma) denotes the select operator, and the selection condition is a Boolean expression specified on the attributes of relation R.

Select Examples

- Example: Select the EMPLOYEE tuples whose department number is four

$$\sigma_{\text{DNO}=4}(EMPLOYEE)$$

- Example: Select employees whose salary is greater than $30,000

$$\sigma_{\text{SALARY}>30,000}(EMPLOYEE)$$

Select Examples

- Example: Select the EMPLOYEE tuples whose department number is four

$$\sigma_{\text{DNO}=4}(EMPLOYEE)$$

- Example: Select employees whose salary is greater than $30,000

$$\sigma_{\text{SALARY}>30,000}(EMPLOYEE)$$

Select Examples

- Example: Select the EMPLOYEE tuples whose department number is four

$$\sigma_{\mathrm{DNO}=4}(EMPLOYEE)$$

- Example: Select employees whose salary is greater than $30,000

$$\sigma_{\mathrm{SALARY}>30,000}(EMPLOYEE)$$

Select Examples

- Example: Select the EMPLOYEE tuples whose department number is four

$$\sigma_{\text{DNO}=4}(EMPLOYEE)$$

- Example: Select employees whose salary is greater than $30,000

$$\sigma_{\text{SALARY}>30{,}000}(EMPLOYEE)$$

Selection Conditions

- A selection condition is a set of clauses connected by the Boolean operators AND, OR, and NOT
- Each clause has the format
  <attribute name> <comparison op> < constant value>
  or
  <attribute name> <comparison op> <attribute name>
- The comparison operators are $=$, $<$, $\leq$, $>$, $\geq$ and $\neq$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond }1>}(\sigma_{<\text{cond }2>}(R)) = \sigma_{<\text{cond }2>}(\sigma_{<\text{cond }1>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond }1>}(\sigma_{<\text{cond }2>}(\sigma_{<\text{cond }3>}(R))) = \sigma_{<\text{cond }1>\text{ AND }<\text{cond }2>\text{ AND }<\text{cond }3>}(R)$$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond }1>}(\sigma_{<\text{cond }2>}(R)) = \sigma_{<\text{cond }2>}(\sigma_{<\text{cond }1>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond }1>}(\sigma_{<\text{cond }2>}(\sigma_{<\text{cond }3>}(R))) = \sigma_{<\text{cond }1>\text{ AND }<\text{cond }2>\text{ AND }<\text{cond }3>}(R)$$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(R)) = \sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 1}>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 3}>}(R))) = \sigma_{<\text{cond 1}>\text{ AND }<\text{cond 2}>\text{ AND }<\text{cond 3}>}(R)$$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(R)) = \sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 1}>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 3}>}(R))) = \sigma_{<\text{cond 1}>\ \text{AND}\ <\text{cond 2}>\ \text{AND}\ <\text{cond 3}>}(R)$$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond } 1>}(\sigma_{<\text{cond } 2>}(R)) = \sigma_{<\text{cond } 2>}(\sigma_{<\text{cond } 1>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond } 1>}(\sigma_{<\text{cond } 2>}(\sigma_{<\text{cond } 3>}(R))) = \sigma_{<\text{cond } 1> \text{ AND } <\text{cond } 2> \text{ AND } <\text{cond } 3>}(R)$$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(R)) = \sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 1}>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 3}>}(R))) = \sigma_{<\text{cond 1}>\ \text{AND}\ <\text{cond 2}>\ \text{AND}\ <\text{cond 3}>}(R)$$

More on Select

- SELECT is a unary operator that takes one relation as input and produces another relation as output
- The SELECT operation $\sigma_{<\text{selection condition}>}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is commutative; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(R)) = \sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 1}>}(R))$$

- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,

$$\sigma_{<\text{cond 1}>}(\sigma_{<\text{cond 2}>}(\sigma_{<\text{cond 3}>}(R))) = \sigma_{<\text{cond 1}>\ \text{AND}\ <\text{cond 2}>\ \text{AND}\ <\text{cond 3}>}(R)$$

Select Example

To select the tuples for all employees who either work in department 4 and make \$25,000 per year, or work in department 5 and make over \$30,000:

Select Example

To select the tuples for all employees who either work in department 4 and make \$25,000 per year, or work in department 5 and make over \$30,000:

(a) $\sigma_{\text{(DNO=4 AND SALARY>25000) OR (DNO=5 AND SALARY>30000)}}(EMPLOYEE)$

Select Example

To select the tuples for all employees who either work in department 4 and make \$25,000 per year, or work in department 5 and make over \$30,000:

(a) $\sigma_{(\text{DNO}=4 \text{ AND SALARY}>25000) \text{ OR } (\text{DNO}=5 \text{ AND SALARY}>30000)}(\mathit{EMPLOYEE})$

(a)

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
Franklin	T	Wong	333445555	1955-12-08	638 Voss,Houston,TX	M	40000	888665555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 FireOak,Humble,TX	M	38000	333445555	5

The Project Operation

- PROJECT is a unary operation that returns a relation containing only specified attributes of its operand relation R
    - The general form of the project operation is

$$\pi_{<\text{attribute list}>}(R)$$

where π (pi) is the symbol used to represent the project operation and $<$attribute list$>$ is the desired list of attributes from the attributes of relation R.

- The project operation removes any duplicate tuples, so the result of the project operation is a set of tuples and hence a valid relation.

Project Example

- Example: To list each employee's first and last name and salary, the following is used:

$$\pi_{\text{LNAME, FNAME, SALARY}}(EMPLOYEE)$$

  - The number of tuples in the result of projection $\pi_{<\text{list}>}(R)$ is always less or equal to the number of tuples in R.
  - If the list of attributes includes a key of R, then the number of tuples is equal to the number of tuples in R.
  - $\pi_{<\text{list1}>}(\pi_{<\text{list2}>}(R)) = \pi_{<\text{list1}>}(R)$ as long as <list2> contains the attributes in <list1>

Project Example

$$\text{(b) } \pi_{\text{LNAME, FNAME, SALARY}}(\mathit{EMPLOYEE})$$

$$\text{(c) } \pi_{\text{SEX, SALARY}}(\mathit{EMPLOYEE})$$

Project Example

$$\text{(b) } \pi_{\text{LNAME, FNAME, SALARY}}(EMPLOYEE)$$

$$\text{(c) } \pi_{\text{SEX, SALARY}}(EMPLOYEE)$$

(b)

LNAME	FNAME	SALARY
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

(c)

SEX	SALARY
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

Renaming of Relational Operations

- We may want to apply several relational algebra operations one after the other.
- Either we can write the operations as a single relational algebra expression by nesting the operations, or we can apply one operation at a time and create intermediate result relations. In the latter case, we must give names to the relations that hold the intermediate results.

Renaming of Relational Operations

- Example: To retrieve the first name, last name, and salary of all employees who work in department number 5, we must apply a select and a project operation. We can write a single relational algebra expression as follows:

$$\pi_{\text{FNAME, LNAME, SALARY}}(\sigma_{\text{DNO}=5}(EMPLOYEE))$$

- OR We can explicitly show the sequence of operations, giving a name to each intermediate relation:

$$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO}=5}(EMPLOYEE)$$

$$\text{RESULT} \leftarrow \pi_{\text{FNAME, LNAME, SALARY}}(DEP5\_EMPS)$$

- We can also rename the attributes, if desired, by specifying new attribute names when we name a partial result, e.g.,RESULT (F, L, S)

Examples

$$(a)\pi_{\text{FNAME, LNAME, SALARY}}(\sigma_{\text{DNO}=5}(EMPLOYEE))$$

$$(b)\ \text{TEMP} \leftarrow \sigma_{\text{DNO}=5}(EMPLOYEE)$$

$$\text{R(FIRSTNAME, LASTNAME, SALARY)} \leftarrow \pi_{\text{FNAME, LNAME, SALARY}}(TEMP)$$

(a)

FNAME	LNAME	SALARY
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

(b)

TEMP	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren,Houston,TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss,Houston,TX	M	40000	888665555	5
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak,Humble,TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice,Houston,TX	F	25000	333445555	5

R	FIRSTNAME	LASTNAME	SALARY
	John	Smith	30000
	Franklin	Wong	40000
	Ramesh	Narayan	38000
	Joyce	English	25000

Exercise

1. Which projects are located in Houston?
2. What are the names of the departments?
3. Find out everything about all of the employees who were born before 1950-01-01.
4. What are the names of the employees who were born before 1950-01-01?
5. When did the manager of the Research department begin managing that department?

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

DEPT_LOCATIONS	DNUMBER	DLOCATION
	1	Houston
	4	Stafford
	5	Bellaire
	5	Sugarland
	5	Houston

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

WORKS_ON	ESSN	PNO	HOURS
	123456789	1	32.5
	123456789	2	7.5
	666884444	3	40.0
	453453453	1	20.0
	453453453	2	20.0
	333445555	2	10.0
	333445555	3	10.0
	333445555	10	10.0
	333445555	20	10.0
	999887777	30	30.0
	999887777	10	10.0
	987987987	10	35.0
	987987987	30	5.0
	987654321	30	20.0
	987654321	20	15.0
	888665555	20	null

PROJECT	PNAME	PNUMBER	PLOCATION	DNUM
	ProductX	1	Bellaire	5
	ProductY	2	Sugarland	5
	ProductZ	3	Houston	5
	Computerization	10	Stafford	4
	Reorganization	20	Houston	1
	Newbenefits	30	Stafford	4

DEPENDENT	ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP
	333445555	Alice	F	1986-04-05	DAUGHTER
	333445555	Theodore	M	1983-10-25	SON
	333445555	Joy	F	1958-05-03	SPOUSE
	987654321	Abner	M	1942-02-28	SPOUSE
	123456789	Michael	M	1988-01-04	SON
	123456789	Alice	F	1988-12-30	DAUGHTER
	123456789	Elizabeth	F	1967-05-05	SPOUSE

Set Theoretic Relational Operators

- The set theoretic operators are union $(R \cup S)$, intersection $(R \cap S)$ and difference $(R - S)$.
- Since relations are sets of tuples, we can borrow established operators that work on sets.
- These are all binary operators. They each take two relations as operands and produce one relation as their result.
- They all require that their input relations are union compatible.

Union Compatibility

- For two operand relations, $R(A_1, A_2, \ldots, A_n)$ and $S(B_1, B_2, \ldots, B_n)$ to be union compatible,
  - they must have the same number of attributes, and
  - the domains of their corresponding attributes must be the same; that is, $dom(A_i) = dom(B_i)$ for $i = 1, 2, \ldots, n$.
- The relation that results from a set theoretic operation will also be union compatible with the two input relations.
- The names of the corresponding attributes do not have to be the same.

The Union Operation

- The result of the UNION operation, denoted by $R \cup S$, is a relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated.
  - Example: To retrieve the social security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5, we can use the union operation as follows:

$$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO}=5}(EMPLOYEE)$$

$$\text{RESULT1} \leftarrow \pi_{\text{SSN}}(DEP5\_EMPS)$$

$$\text{RESULT2(SSN)} \leftarrow \pi_{\text{SUPERSSN}}(DEP5\_EMPS)$$

$$RESULT \leftarrow RESULT1 \cup RESULT2$$

  - The union operation produces the tuples that are in either RESULT1 or RESULT2 or both.

The Union Operation

- The result of the UNION operation, denoted by $R \cup S$, is a relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated.
  - Example: To retrieve the social security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5, we can use the union operation as follows:

$$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO}=5}(EMPLOYEE)$$

$$\text{RESULT1} \leftarrow \pi_{\text{SSN}}(DEP5\_EMPS)$$

$$\text{RESULT2(SSN)} \leftarrow \pi_{\text{SUPERSSN}}(DEP5\_EMPS)$$

$$RESULT \leftarrow RESULT1 \cup RESULT2$$

  - The union operation produces the tuples that are in either RESULT1 or RESULT2 or both.

The Union Operation

- The result of the UNION operation, denoted by $R \cup S$, is a relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated.
  - Example: To retrieve the social security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5, we can use the union operation as follows:

$$\mathrm{DEP5\_EMPS} \leftarrow \sigma_{\mathrm{DNO}=5}(EMPLOYEE)$$

$$\mathrm{RESULT1} \leftarrow \pi_{\mathrm{SSN}}(DEP5\_EMPS)$$

$$\mathrm{RESULT2(SSN)} \leftarrow \pi_{\mathrm{SUPERSSN}}(DEP5\_EMPS)$$

$$RESULT \leftarrow RESULT1 \cup RESULT2$$

  - The union operation produces the tuples that are in either RESULT1 or RESULT2 or both.

The Union Operation

- The result of the UNION operation, denoted by $R \cup S$, is a relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated.
  - Example: To retrieve the social security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5, we can use the union operation as follows:

$$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO} = 5}(EMPLOYEE)$$

$$\text{RESULT1} \leftarrow \pi_{\text{SSN}}(DEP5\_EMPS)$$

$$\text{RESULT2(SSN)} \leftarrow \pi_{\text{SUPERSSN}}(DEP5\_EMPS)$$

$$RESULT \leftarrow RESULT1 \cup RESULT2$$

  - The union operation produces the tuples that are in either RESULT1 or RESULT2 or both.

The Intersection Operation

- The result of the INTERSECTION operation, denoted by $R \cap S$, is a relation that includes all tuples that are in both R and S. The two operands must be union compatible.
  - Example: The result of the intersection operation includes only those who are both students and instructors.
  - STUDENT $\cap$ INSTRUCTOR

(a)

STUDENT	FN	LN
	Susan	Yao
	Ramesh	Shah
	Johnny	Kohler
	Barbara	Jones
	Amy	Ford
	Jimmy	Wang
	Ernest	Gilbert

INSTRUCTOR	FNAME	LNAME
	John	Smith
	Ricardo	Browne
	Susan	Yao
	Francis	Johnson
	Ramesh	Shah

(c)

FN	LN
Susan	Yao
Ramesh	Shah

The Set Difference Operation

- The result of the SET DIFFERENCE operation, denoted by $R - S$, is a relation that includes all tuples that are in R but not in S. The two operands must be union compatible.
  - Example: The figure shows the names of students who are not instructors, and the names of instructors who are not students.
  - `(d) STUDENT - INSTRUCTOR (e)INSTRUCTOR - STUDENT`

(a)

STUDENT	FN	LN
	Susan	Yao
	Ramesh	Shah
	Johnny	Kohler
	Barbara	Jones
	Amy	Ford
	Jimmy	Wang
	Ernest	Gilbert

INSTRUCTOR	FNAME	LNAME
	John	Smith
	Ricardo	Browne
	Susan	Yao
	Francis	Johnson
	Ramesh	Shah

(d)

FN	LN
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

(e)

FNAME	LNAME
John	Smith
Ricardo	Browne
Francis	Johnson

Examples on UNION, INTERSECTION and MINUS

(b) STUDENT ∪ INSTRUCTOR. (c) STUDENT ∩ INSTRUCTOR. (d) STUDENT − INSTRUCTOR. (e) INSTRUCTOR − STUDENT

(a)

STUDENT	FN	LN
	Susan	Yao
	Ramesh	Shah
	Johnny	Kohler
	Barbara	Jones
	Amy	Ford
	Jimmy	Wang
	Ernest	Gilbert

INSTRUCTOR	FNAME	LNAME
	John	Smith
	Ricardo	Browne
	Susan	Yao
	Francis	Johnson
	Ramesh	Shah

(b)

FN	LN
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

(c)

FN	LN
Susan	Yao
Ramesh	Shah

(d)

FN	LN
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

(e)

FNAME	LNAME
John	Smith
Ricardo	Browne
Francis	Johnson

Set Theoretic Relational Operators

- Notice that both union and intersection are commutative operations; that is

$$R \cup S = S \cup R, \text{ and } R \cap S = S \cap R$$

- Both union and intersection can be treated as n-ary operations applicable to any number of relations as both are associative operations; that is

$$R \cup (S \cup T) = (R \cup S) \cup T, \text{ and}$$

$$(R \cap S) \cap T = R \cap (S \cap T)$$

- The minus operation is not commutative; that is, in general

$$R - S \neq S - R$$

Exercise

1. Find the social security numbers of the managers of departments who work on project 30.
2. Find the social security numbers of all employees who have dependents or who work on project 2.
3. Find the social security numbers of all employees who do not have any dependents.

The Cartesian Product (Cross Product)

- The CARTESION PRODUCT operation (also known as Cross Product or Cross Join) is used to combine tuples from two relations in a combinatorial fashion. In general, the result of

  $$R(A_1, A_2, \ldots, A_n) \times S(B_1, B_2, \ldots, B_m)$$

  is a relation Q with degree $n + m$ attributes

  $$Q(A_1, A_2, \ldots, A_n, B_1, B_2, \ldots, B_m)$$

  in that order.
  - The resulting relation Q has one tuple for each combination of tuples – one from R and one from S.
- Hence, if R has n_R tuples (denoted as $|R| = n_R$), and S has n_S tuples, then $|R \times S|$ will have $n_R * n_S$ tuples.
- The two operands do NOT have to be union compatible

Cartesian Product Example

Example: To retrieve a list of names of each female employee's dependents

$$FEMALE\_EMPS \leftarrow \sigma_{SEX = 'F'}(EMPLOYEE)$$

$$EMPNAMES \leftarrow \pi_{FNAME,\ LNAME,\ SSN}(FEMALE\_EMPS)$$

$$EMP\_DEPENDENTS \leftarrow EMPNAMES \times DEPENDENT$$

Cartesian Product Example

Example: To retrieve a list of names of each female employee's dependents

$$\text{FEMALE\_EMPS} \leftarrow \sigma_{\text{SEX} = \text{'F'}}(\mathit{EMPLOYEE})$$

$$\text{EMPNAMES} \leftarrow \pi_{\text{FNAME, LNAME, SSN}}(\mathit{FEMALE\_EMPS})$$

$$\text{EMP\_DEPENDENTS} \leftarrow \mathit{EMPNAMES} \times \mathit{DEPENDENT}$$

Cartesian Product Example

Example: To retrieve a list of names of each female employee's dependents

$$\text{FEMALE\_EMPS} \leftarrow \sigma_{\text{SEX = 'F'}}(\textit{EMPLOYEE})$$

$$\text{EMPNAMES} \leftarrow \pi_{\text{FNAME, LNAME, SSN}}(\textit{FEMALE\_EMPS})$$

$$\text{EMP\_DEPENDENTS} \leftarrow \textit{EMPNAMES} \times \textit{DEPENDENT}$$

Cartesian Product Example

Example: To retrieve a list of names of each female employee's dependents

$$\text{FEMALE\_EMPS} \leftarrow \sigma_{\text{SEX} = \text{'F'}}(\mathit{EMPLOYEE})$$

$$\text{EMPNAMES} \leftarrow \pi_{\text{FNAME, LNAME, SSN}}(\mathit{FEMALE\_EMPS})$$

$$\text{EMP\_DEPENDENTS} \leftarrow \mathit{EMPNAMES} \times \mathit{DEPENDENT}$$

Example

FEMALE_EMPS	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle,Spring,TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry,Bellaire,TX	F	43000	888665555	4
	Joyce	A	English	453453453	1972-07-31	5631 Rice,Houston,TX	F	25000	333445555	5

EMPNAMES	FNAME	LNAME	SSN
	Alicia	Zelaya	999887777
	Jennifer	Wallace	987654321
	Joyce	English	453453453

Result

EMP_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
	Alicia	Zelaya	999887777	333445555	Alice	F	1986-04-05	• • •
	Alicia	Zelaya	999887777	333445555	Theodore	M	1983-10-25	• • •
	Alicia	Zelaya	999887777	333445555	Joy	F	1958-05-03	• • •
	Alicia	Zelaya	999887777	987654321	Abner	M	1942-02-28	• • •
	Alicia	Zelaya	999887777	123456789	Michael	M	1988-01-04	• • •
	Alicia	Zelaya	999887777	123456789	Alice	F	1988-12-30	• • •
	Alicia	Zelaya	999887777	123456789	Elizabeth	F	1967-05-05	• • •
	Jennifer	Wallace	987654321	333445555	Alice	F	1986-04-05	• • •
	Jennifer	Wallace	987654321	333445555	Theodore	M	1983-10-25	• • •
	Jennifer	Wallace	987654321	333445555	Joy	F	1958-05-03	• • •
	Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	• • •
	Jennifer	Wallace	987654321	123456789	Michael	M	1988-01-04	• • •
	Jennifer	Wallace	987654321	123456789	Alice	F	1988-12-30	• • •
	Jennifer	Wallace	987654321	123456789	Elizabeth	F	1967-05-05	• • •
	Joyce	English	453453453	333445555	Alice	F	1986-04-05	• • •
	Joyce	English	453453453	333445555	Theodore	M	1983-10-25	• • •
	Joyce	English	453453453	333445555	Joy	F	1958-05-03	• • •
	Joyce	English	453453453	987654321	Abner	M	1942-02-28	• • •
	Joyce	English	453453453	123456789	Michael	M	1988-01-04	• • •
	Joyce	English	453453453	123456789	Alice	F	1988-12-30	• • •
	Joyce	English	453453453	123456789	Elizabeth	F	1967-05-05	• • •

Result

EMP_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
	Alicia	Zelaya	999887777	333445555	Alice	F	1986-04-05	• • •
	Alicia	Zelaya	999887777	333445555	Theodore	M	1983-10-25	• • •
	Alicia	Zelaya	999887777	333445555	Joy	F	1958-05-03	• • •
	Alicia	Zelaya	999887777	987654321	Abner	M	1942-02-28	• • •
	Alicia	Zelaya	999887777	123456789	Michael	M	1988-01-04	• • •
	Alicia	Zelaya	999887777	123456789	Alice	F	1988-12-30	• • •
	Alicia	Zelaya	999887777	123456789	Elizabeth	F	1967-05-05	• • •
	Jennifer	Wallace	987654321	333445555	Alice	F	1986-04-05	• • •
	Jennifer	Wallace	987654321	333445555	Theodore	M	1983-10-25	• • •
	Jennifer	Wallace	987654321	333445555	Joy	F	1958-05-03	• • •
	Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	• • •
	Jennifer	Wallace	987654321	123456789	Michael	M	1988-01-04	• • •
	Jennifer	Wallace	987654321	123456789	Alice	F	1988-12-30	• • •
	Jennifer	Wallace	987654321	123456789	Elizabeth	F	1967-05-05	• • •
	Joyce	English	453453453	333445555	Alice	F	1986-04-05	• • •
	Joyce	English	453453453	333445555	Theodore	M	1983-10-25	• • •
	Joyce	English	453453453	333445555	Joy	F	1958-05-03	• • •
	Joyce	English	453453453	987654321	Abner	M	1942-02-28	• • •
	Joyce	English	453453453	123456789	Michael	M	1988-01-04	• • •
	Joyce	English	453453453	123456789	Alice	F	1988-12-30	• • •
	Joyce	English	453453453	123456789	Elizabeth	F	1967-05-05	• • •

Result

$$\text{ACTUAL\_DEPENDENTS} \leftarrow \sigma_{\text{SSN}=\text{ESSN}}(\mathit{EMP\_DEPENDENTS})$$

$$\text{RESULT} \leftarrow \pi_{\text{FNAME, LNAME, DEPENDENT\_NAME}}(\mathit{ACTUAL\_DEPENDENTS})$$

ACTUAL_DEPENDENTS	FNAME	LNAME	SSN	ESSN	DEPENDENT_NAME	SEX	BDATE	• • •
	Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	• • •

RESULT	FNAME	LNAME	DEPENDENT_NAME
	Jennifer	Wallace	Abner

Join Operations

- We don't usually want a Cartesian product per se. We usually want some subset of it.
- So, we typically need to use the select operation to get to the part of the Cartesian product we want.
- We use JOIN operations for this. There are three types of JOIN operations, the theta join, the equijoin, and the natural join.
- The general form of a join operation on two relations $R(A_1, A_2, \ldots, A_n)$ and $S(B_1, B_2, \ldots, B_m)$ is:

$$R \bowtie_{<\text{join condition}>} S$$

The Theta Join

- The theta join produces all combinations of tuples from two relations, R and S, that satisfy a join condition
- A join condition is similar to a select condition, except that you can not use the Boolean OR and NOT operators. All clauses are ANDed together
- If you need a more general condition, you can use select with a Cartesian product

$$R \bowtie_{<\text{condition}>} S = \sigma_{<\text{condition}>}(R \times S)$$

Join Example

- Suppose that we want to retrieve the name of the manager of each department. To get the manager's name, we need to combine each DEPARTMENT tuple with the EMPLOYEE tuple whose SSN value matches the MGRSSN value in the department tuple. We do this by using the join $\bowtie$ operation.

$$\text{DEPT\_MGR} \leftarrow \mathit{DEPARTMENT} \bowtie_{\text{MGRSSN = SSN}} \mathit{EMPLOYEE}$$

$$\text{RESULT} \leftarrow \pi_{\text{DNAME, LNAME, FNAME}}(\mathit{DEPT\_MGR})$$

Join Example

- Suppose that we want to retrieve the name of the manager of each department. To get the manager's name, we need to combine each DEPARTMENT tuple with the EMPLOYEE tuple whose SSN value matches the MGRSSN value in the department tuple. We do this by using the join $\bowtie$ operation.

$$\text{DEPT\_MGR} \leftarrow DEPARTMENT \bowtie_{\text{MGRSSN} = \text{SSN}} EMPLOYEE$$

$$\text{RESULT} \leftarrow \pi_{\text{DNAME, LNAME, FNAME}}(DEPT\_MGR)$$

Equi-joins

- The most common use of join involves join conditions with equality comparisons only. Such a join, where the only comparison operator used is $=$, is called an EQUIJOIN. In the result of an EQUIJOIN we always have one or more pairs of attributes (whose names need not be identical) that have identical values in every tuple.

Natural Joins

- Because one of each pair of attributes with identical values is superfluous, a new operation called natural join - denoted by * - was created to get rid of the second (unnecessary) attribute in an EQUIJOIN condition.
- The standard definition of natural join requires that the two join attributes, or each pair of corresponding join attributes, have the same name in both relations. If this is not the case, a renaming operation is applied first.

Natural Join Example

- To apply a natural join on the DNUMBER attributes of DEPARTMENT and DEPT_LOCATIONS, we write:

$$DEPT\_LOCS \leftarrow DEPARTMENT * DEPT\_LOCATIONS$$

The Outer Join Operation

- In NATURAL JOIN tuples without a matching (or related) tuple are eliminated from the join result. Tuples with null in the join attributes are also eliminated. Sometimes, as a practical matter, we want to keep this information.
- A set of operations, called outer joins, can be used when we want to keep all the tuples in R, or all those in S, or all those in both relations in the result of the join, regardless of whether or not they have matching tuples in the other relation.

The Outer Join Operation

- The left outer join operation keeps every tuple in the first or left relation R in R $⟕$ S; if no matching tuple is found in S, then the attributes of S in the join result are filled or “padded” with null values.
- A similar operation, right outer join, keeps every tuple in the second or right relation S in the result of R $⟖$ S.
- A third operation, full outer join, denoted by $⟗$ keeps all tuples in both the left and the right relations when no matching tuples are found, padding them with null values as needed.

Outer Join Example

- To find the names of all employees, along with the names of the departments they manage, we could use:

$$TEMP \leftarrow EMPLOYEE \;⟕_{\mathrm{SSN}\,=\,\mathrm{MGRSSN}} DEPARTMENT$$

$$RESULT \leftarrow \pi_{\mathrm{FNAME,MINIT,LNAME,DNAME}}(TEMP)$$

Complete Set of Relational Operations

- The set of operations including select σ, project π, union $\cup$, set difference $-$, and Cartesian product $\times$ is called a complete set because any other relational algebra expression can be expressed by a combination of these five operations.
- For example:

$$R \cap S = (R \cup S) - ((R - S) \cup (S - R))$$

$$R \bowtie_{<\text{join condition}>} S = \sigma_{<\text{join condition}>}(R \times S)$$

Additional Relational Operations

- Even though we could, in principle, get by with only five relational algebra operations, in practice, we use more.
- Sometimes, these are convenient shortcuts, like the join operation.
- Sometimes, we want to extend the relational algebra to enable us to make some types of queries that were not originally supported.

The Division Operation

- DIVISION is another shortcut operation that could be expressed using only π, $\times$, and $-$
- It is applied to two relations, $R(Z) \div S(X)$, where the attributes X are a subset of the attributes Z. Let $Y = Z - X$ (and hence $Z = X \cup Y$); that is, let Y be the set of attributes of R that are not attributes of S.
- The result of DIVISION is a relation $T(Y)$. For a tuple t to appear in the result T, the values in t must appear in R in combination with every tuple in S.

Division (cont.)

To get R ÷ S, you could use:

- Project out Y, the attributes of R that are not in S

$$T_1 \leftarrow \pi_Y(R)$$

- T2 now contains all the tuples you do not want

$$T_2 \leftarrow \pi_Y((T_1 \times S) - R)$$

- Set difference removes these tuples, leaving just the tuples you do want

$$RESULT \leftarrow T_1 - T_2$$

Division Example

R

A1	A2	A3	A4
a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
e	d	e	f
a	b	d	e

S

A3	A4
c	d
e	f

$R \div S$

A1	A2
a	b
e	d