Relational Algebra (Part 2)

Summary of Relational Operators

Operator	Notation	Meaning
Selection	$\sigma_{\varphi}(R)$	choose rows
Projection	$\pi_{A_1,\ldots,A_n}(R)$	choose columns
Union **Intersection** **Difference**	$R_1 \cup R_2$ $R_1 \cap R_2$ $R_1 - R_2$	set operations
Cartesian product **Join** **Natural-join**	$R_1 \times R_2$ $R_1 \bowtie_{\varphi} R_2$ $R_1 \bowtie R_2$	combine tables
Renaming	$\rho_{R'(A_1,\ldots,A_n)}(R)$ $\rho_{R'}(R)$ $\rho_{(A_1,\ldots,A_n)}(R)$	rename relation and attributes

A Complete Set of Relational Operators

- The following six operators constitute **a complete set**:
  - **selection** σ;
  - **projection** π;
  - **renaming** ρ;
  - **union** $\cup$;
  - **difference** $-$;
  - **Cartesian product** $\times$.

A Complete Set of Relational Operators

- Six operators (i.e., **selection** σ, **projection** π, **renaming** ρ, **union** $\cup$, **difference** - and **Cartesian product** $\times$) constitute **a complete set**.
- It means that the other RA operators like **intersection** and **join** are **not necessary** and can be expressed by these six operators.
  - **join**: $R_1 \bowtie_\varphi R_2 = \sigma_\varphi(R_1 \times R_2)$
  - **intersection**: $R_1 \cap R_2 = R_1 - (R_1 - R_2)$
- Hence, **intersection** and **join** do not increase the expressive power of RA.
- Nonetheless it is important to include **intersection** and **join** because they are convenient to use and commonly applied in database applications.

Relational Algebra Queries

- The output of each RA operation is a relation, which can be used again as the input for another RA operation.
- RA operations **can be nested to arbitrary depth** for expressing complex queries, as in arithmetic.
  - Parentheses and precedence rules define the order of evaluation: from highest to lowest: $\{\sigma, \pi, \rho\}$, $\{\times, \bowtie\}$, $\{\cap\}$, $\{\cup, -\}$
  - Operators with the same precedence are evaluated from left to right.
  - Use brackets if you are not sure.
- A **query** in RA is a sequence of RA operations and each RA operation takes one or two relations as its input and produces one relation as its output.
- Different from SQL, RA considers **relations as sets** (not **multisets** as in SQL). Hence, relations produced by an RA operation **have no duplicate tuples**.

Hints for Writing RA Queries

1. Firstly, identify which relations need to be involved, while ignoring the rest.
2. Then break the answer down by considering intermediate relations, i.e., queries may be expressed as **a sequence of assignment statements**.

   Example: $R := \pi_{HTeam,GTeam}(\sigma_{HScore=1}(\rho_{(HTeam,HScore,GScore,GTeam)}(\text{SOCCER})))$

   - Use good names for intermediate relations;
   - Keep track of attributes you have at each step.
3. When combining relations, check attribute names and make sure that:
   - attributes that should match are to match.
   - attributes that shouldn't match are not to match.
4. When using set operations, make sure that two relations of an operation have the same type (i.e., type compatibility).

RA Queries – Exercises (Self Join)

- Given the following relation schema:

  STUDENT={StudentID, Name, DoB}

- **Query 1:** Find **pairs of** students who have the same birthday. Show their names.

STUDENT		
StudentID	Name	DoB
457	Lisa	18-Oct-1993
458	Mike	16-May-1990
459	Peter	18-Oct-1993

RA Queries – Exercises (Self Join)

- Given the following relation schema:

$$\text{STUDENT}=\{\text{StudentID, Name, DoB}\}$$

- **Query 1:** Find **pairs of** students who have the same birthday. Show their names.

$$\pi_{R_1.Name, R_2.Name}(\sigma_{R_1.StudentID<R_2.StudentID}(\sigma_{R_1.DoB=R_2.DoB}(\rho_{R_1}(\text{STUDENT}) \times \rho_{R_2}(\text{STUDENT})))).$$

```
SELECT R1.name, R2.name
  FROM Student AS R1, Student AS R2
 WHERE R1.DoB = R2.DoB AND R1.StudentID < R2.StudentID;
```

- **Why do we need $\sigma_{R_1.StudentID<R_2.StudentID}$ in the above query?**
- **Why do we need to use renaming in the above query?**

RA Queries – Exercises (Self Join)

- Given the following relation schema:

$$\textsc{Student}=\{\text{StudentID, Name, DoB}\}$$

- **Query 1:** Find **pairs of** students who have the same birthday. Show their names.

Two different solutions:

(1). $\pi_{R_1.Name, R_2.Name}(\sigma_{R_1.StudentID<R_2.StudentID}(\sigma_{R_1.DoB=R_2.DoB}(\rho_{R_1}(\textsc{Student}) \times \rho_{R_2}(\textsc{Student})))$

(2). $\pi_{Name, Name'}(\sigma_{StudentID<StudentID'}(\textsc{Student} \bowtie \rho_{S(StudentID', Name', DoB)}(\textsc{Student}))$

RA Queries – Exercises (Self Join)

- **Query 1:** Find **pairs of** students who have the same birthday. Show their names.

  (1). $\pi_{R_1.Name, R_2.Name}(\sigma_{R_1.StudentID < R_2.StudentID}(\sigma_{R_1.DoB = R_2.DoB}(\rho_{R_1}(\text{STUDENT}) \times \rho_{R_2}(\text{STUDENT}))))$

  (2). $\pi_{Name, Name'}(\sigma_{StudentID < StudentID'}(\text{STUDENT} \bowtie \rho_{S(StudentID', Name', DoB)}(\text{STUDENT})))$

- If evaluating our queries over the following relation, what will be the result?

STUDENT		
StudentID	Name	DoB
457	Lisa	18-Oct-1993
458	Mike	16-May-1990
459	Peter	18-Oct-1993

RA Queries – Exercises (Self Join)

- **Query 1 (solution 1):** $\pi_{R_1.Name, R_2.Name}(\sigma_{R_1.StudentID<R_2.StudentID}(\sigma_{R_1.DoB=R_2.DoB}(\rho_{R_1}(\text{STUDENT}) \times \rho_{R_2}(\text{STUDENT}))))$.

STUDENT		
StudentID	Name	DoB
457	Lisa	18-Oct-1993
458	Mike	16-May-1990
459	Peter	18-Oct-1993

$\rho_{R_1}(\text{STUDENT}) \times \rho_{R_2}(\text{STUDENT})$

R_1.StudentID	R_1.Name	R_1.DoB	R_2.StudentID	R_2.Name	R_2.DoB
457	Lisa	18-Oct-1993	457	Lisa	18-Oct-1993
457	Lisa	18-Oct-1993	458	Mike	16-May-1990
457	Lisa	18-Oct-1993	458	Peter	18-Oct-1993
458	Mike	16-May-1990	457	Lisa	18-Oct-1993
458	Mike	16-May-1990	458	Mike	16-May-1990
458	Mike	16-May-1990	458	Peter	18-Oct-1993
458	Peter	18-Oct-1993	457	Lisa	18-Oct-1993
458	Peter	18-Oct-1993	458	Mike	16-May-1990
458	Peter	18-Oct-1993	458	Peter	18-Oct-1993

RA Queries – Exercises (Self Join)

- **Query 1 (solution 1):** $\pi_{R_1.Name, R_2.Name}(\sigma_{R_1.StudentID<R_2.StudentID}(\sigma_{R_1.DoB=R_2.DoB}(\rho_{R_1}(\text{STUDENT}) \times \rho_{R_2}(\text{STUDENT})))$.

STUDENT		
StudentID	Name	DoB
457	Lisa	18-Oct-1993
458	Mike	16-May-1990
459	Peter	18-Oct-1993

$R' = \sigma_{R_1.DoB=R_2.DoB}(\rho_{R_1}(\text{STUDENT}) \times \rho_{R_2}(\text{STUDENT}))$

R_1.StudentID	R_1.Name	R_1.DoB	R_2.StudentID	R_2.Name	R_2.DoB
457	Lisa	18-Oct-1993	457	Lisa	18-Oct-1993
457	Lisa	18-Oct-1993	459	Peter	18-Oct-1993
458	Mike	16-May-1990	458	Mike	16-May-1990
459	Peter	18-Oct-1993	457	Lisa	18-Oct-1993
459	Peter	18-Oct-1993	459	Peter	18-Oct-1993

$\pi_{R_1.Name, R_2.Name}(\sigma_{R_1.StudentID<R_2.StudentID}(R'))$

R_1.Name	R_2.Name
Lisa	Peter

RA Queries – Exercises (Self Join)

- **Query 1 (solution 2):** $\pi_{Name,Name'}(\sigma_{StudentID<StudentID'}($ $\textsc{Student} \bowtie \rho_{S(StudentID',Name',DoB)}(\textsc{Student}))$.

STUDENT		
StudentID	Name	DoB
457	Lisa	18-Oct-1993
458	Mike	16-May-1990
459	Peter	18-Oct-1993

$R' = \textsc{Student} \bowtie \rho_{S(StudentID',Name',DoB)}(\textsc{Student})$

StudentID	Name	DoB	StudentID'	Name'
457	Lisa	18-Oct-1993	459	Peter
459	Peter	18-Oct-1993	457	Lisa
459	Peter	18-Oct-1993	459	Peter
457	Lisa	18-Oct-1993	457	Lisa
458	Mike	16-May-1990	458	Mike

$\pi_{Name,Name'}(\sigma_{StudentID<StudentID'}(R'))$

Name	Name'
Lisa	Peter

RA Queries – Exercises (Difference 1)

- Given the following relation schemas:

STUDENT={StudentID, Name, DoB}

ENROL={StudentID, CourseNo, Semester, EnrolDate}

- **Query 2:** Which students have **never** enrolled in any course? Show their IDs and names.

STUDENT		
StudentID	Name	DoB
456	Tom	02-Jan-1991
457	Lisa	18-Oct-1993
458	Mike	16-May-1990

ENROL			
StudentID	CourseNo	Semester	EnrolDate
456	COMP2400	2010 S2	02-Jul-2010
458	COMP2400	2010 S2	23-Jun-2010
458	COMP2600	2010 S2	05-Aug-2010

RA Queries – Exercises (Difference 1)

- Given the following relation schemas:

  STUDENT={StudentID, Name, DoB}

  ENROL={StudentID, CourseNo, Semester, EnrolDate}

- **Query 2:** Which students have **never** enrolled in any course? Show their IDs and names.

Hints:

(1) All the students

(2) Students who have enrolled in at least one course

Answer: Students in **the result (1) but not in the result (2)**.

RA Queries – Exercises (Difference 1)

- Given the following relation schemas:

$$\text{STUDENT}=\{\text{StudentID, Name, DoB}\}$$
$$\text{ENROL}=\{\text{StudentID, CourseNo, Semester, EnrolDate}\}$$

- **Query 2:** Which students have **never** enrolled in any course? Show their IDs and names.

  (1) All the students

$$R_1 := \pi_{StudentID}(\text{STUDENT})$$

  (2) Students who have enrolled in at least one course

$$R_2 := \pi_{StudentID}(\text{ENROL})$$

  Answer: Students in **the result (1) but not in the result (2)**

$$\pi_{StudentID,Name}((R_1 - R_2) \bowtie \text{STUDENT})$$

RA Queries – Exercises (Difference 1)

- **Query 2:** Which students have **never** enrolled in any course? Show their IDs and names.
- If evaluating our query over the following relations, what will be the result?
  - $R_1 := \pi_{StudentID}(\text{STUDENT})$
  - $R_2 := \pi_{StudentID}(\text{ENROL})$
  - $\pi_{StudentID,Name}((R_1 - R_2) \bowtie \text{STUDENT})$

STUDENT		
StudentID	Name	DoB
456	Tom	02-Jan-1991
457	Lisa	18-Oct-1993
458	Mike	16-May-1990

ENROL			
StudentID	CourseNo	Semester	EnrolDate
456	COMP2400	2010 S2	02-Jul-2010
458	COMP2400	2010 S2	23-Jun-2010
458	COMP2600	2010 S2	05-Aug-2010

RA Queries – Exercises (Difference 1)

- **Query 2:** Which students have **never** enrolled in any course? Show their IDs and names.
- If evaluating our query over the following relations, what will be the result?
  - $R_1 := \pi_{StudentID}(\textsc{Student})$
  - $R_2 := \pi_{StudentID}(\textsc{Enrol})$
  - $\pi_{StudentID,Name}((R_1 - R_2) \bowtie \textsc{Student})$

R_1

StudentID
456
457
458

R_2

StudentID
456
458

$\pi_{StudentID,Name}((R_1 - R_2) \bowtie \textsc{Student})$

StudentID	Name
457	Lisa

RA Queries – Exercises (Difference 2)

- Given the following relation schemas:

  STUDENT={StudentID, Name, DoB}

  ENROL={StudentID, CourseNo, Semester, EnrolDate}

- **Query 3:** Which students have **only** enrolled in the course COMP2400? Show their IDs and names.

STUDENT

StudentID	Name	DoB
456	Tom	02-Jan-1991
457	Lisa	18-Oct-1993
458	Mike	16-May-1990

ENROL

StudentID	CourseNo	Semester	EnrolDate
456	COMP2400	2010 S2	02-Jul-2010
457	COMP2400	2010 S2	08-Jul-2010
458	COMP2400	2010 S2	23-Jun-2010
458	COMP2600	2010 S2	05-Aug-2010

RA Queries – Exercises (Difference 2)

- Given the following relation schemas:

  STUDENT={StudentID, Name, DoB}

  ENROL={StudentID, CourseNo, Semester, EnrolDate}

- **Query 3:** Which students have **only** enrolled in the course COMP2400? Show their IDs and names.

Hints:

(1) Students who have enrolled in the course COMP2400.

(2) Students who have enrolled in a course but not COMP2400.

Answer: Students in **the result (1) but not in the result (2)**.

RA Queries – Exercises (Difference 2)

- Given the following relation schemas:

$$\textsc{Student}=\{\text{StudentID, Name, DoB}\}$$
$$\textsc{Enrol}=\{\text{StudentID, CourseNo, Semester, EnrolDate}\}$$

- **Query 3:** Which students have **only** enrolled in the course COMP2400? Show their IDs and names.

  (1) Students who have enrolled in the course COMP2400.

$$R_1 := \pi_{StudentID}(\sigma_{CourseNo=\text{`}COMP2400\text{'}}(\textsc{Enrol}))$$

  (2) Students who have enrolled in a course other than COMP2400.

$$R_2 := \pi_{StudentID}(\sigma_{CourseNo\neq\text{`}COMP2400\text{'}}(\textsc{Enrol}))$$

  Answer: Students in **the result (1) but not in the result (2)**.

$$\pi_{StudentID,Name}((R_1 - R_2) \bowtie \textsc{Student})=$$
$$\pi_{StudentID,Name}((\pi_{StudentID}(\sigma_{CourseNo=\text{`}COMP2400\text{'}}(\textsc{Enrol})) - \pi_{StudentID}(\sigma_{CourseNo\neq\text{`}COMP2400\text{'}}(\textsc{Enrol}))) \bowtie \textsc{Student})$$

RA Queries – Exercises (Difference 2)

- **Query 3:** Which students have **only** enrolled in the course COMP2400? Show their IDs and names.
- If evaluating our query over the following relations, what will be the result?
  - $R_1 := \pi_{StudentID}(\sigma_{CourseNo=\text{'}COMP2400\text{'}}(\textsc{Enrol}))$
  - $R_2 := \pi_{StudentID}(\sigma_{CourseNo\neq\text{'}COMP2400\text{'}}(\textsc{Enrol}))$
  - $\pi_{StudentID,Name}((R_1 - R_2) \bowtie \textsc{Student})$

STUDENT		
StudentID	Name	DoB
456	Tom	02-Jan-1991
457	Lisa	18-Oct-1993
458	Mike	16-May-1990

ENROL			
StudentID	CourseNo	Semester	EnrolDate
456	COMP2400	2010 S2	02-Jul-2010
457	COMP2400	2010 S2	08-Jul-2010
458	COMP2400	2010 S2	23-Jun-2010
458	COMP2600	2010 S2	05-Aug-2010

RA Queries – Exercises (Difference 2)

- **Query 3:** Which students have **only** enrolled in the course COMP2400? Show their IDs and names.
- If evaluating our query over the following relations, what will be the result?
  - $R_1 := \pi_{StudentID}(\sigma_{CourseNo=`COMP2400'}(\text{ENROL}))$
  - $R_2 := \pi_{StudentID}(\sigma_{CourseNo\neq`COMP2400'}(\text{ENROL}))$
  - $\pi_{StudentID,Name}((R_1 - R_2) \bowtie \text{STUDENT})$

R_1

StudentID
456
457
458

R_2

StudentID
458

$\pi_{StudentID,Name}((R_1 - R_2) \bowtie \text{STUDENT})$

StudentID	Name
456	Tom
457	Lisa

More Hints for Writing RA Queries

- Pay attention to keywords like **not**, **never**, **only**, **always**, **exactly**, etc. which often indicates the use of **difference** in the corresponding RA queries.
- To show "never":
  - Find all the (combinations of) tuples that are involved.
  - Use difference to subtract those that have occurred.
- To show "only" and "always":
  - Find all the (combinations of) tuples that are involved.
  - Use difference to subtract those that didn't always occur.

Equivalence of RA and SQL Queries (1)

- Each RA query can be easily re-written in SQL, or vice versa.
  - **Selection:** $\sigma_{\varphi}(R)$ corresponds to
    `SELECT DISTINCT * FROM R WHERE` φ`;`
  - **Projection:** $\pi_{A_1,\ldots,A_n}(R)$ corresponds to
    `SELECT DISTINCT` $A_1,\ldots,A_n$ `FROM R;`
  - **Renaming:** $\rho_{S(B_1,\ldots,B_n)}(R)$ (with attributes $A_1,\ldots,A_n$ in R) corresponds to
    `SELECT` A_1 `AS` $B_1,\ldots,A_n$ `AS` B_n `FROM` R `AS` S`;`

Equivalence of RA and SQL Queries (2)

- **Union:** $R_1 \cup R_2$ corresponds to

  `SELECT * FROM` R_1 `UNION SELECT * FROM` R_2

- **Intersection:** $R_1 \cap R_2$ corresponds to

  `SELECT * FROM` R_1 `INTERSECT SELECT * FROM` R_2

- **Difference:** $R_1 - R_2$ (with attributes $A_1, \ldots, A_n$) corresponds to

  `SELECT * FROM` R_1 `EXCEPT SELECT * FROM` R_2

  `SELECT DISTINCT * FROM` R_1 `WHERE NOT EXISTS`
  `(SELECT * FROM` R_2
  `WHERE` $R_1.A_1 = R_2.A_1$ `AND ... AND` $R_1.A_n = R_2.A_n$`)`

SQL eliminates duplicate tuples in the resulting relations of set operations `UNION`, `INTERSECT` and `EXCEPT`.

Equivalence of RA and SQL Queries (3)

- **Cartesian Product:** $R_1 \times R_2$ corresponds to

  `SELECT * FROM` R_1`,` R_2`;`

- **Join:** $R_1 \bowtie_\varphi R_2$ corresponds to

  `SELECT DISTINCT * FROM` R_1 `INNER JOIN` R_2 `ON` φ`;`

  (φ may contain $=, <, \leq, >, \geq, \neq$)

- **Natural-Join:** $R_1 \bowtie R_2$ corresponds to

  `SELECT DISTINCT * FROM` R_1 `NATURAL JOIN` R_2`;`

Outer joins are not considered in the traditional relational algebra, as well as aggregation.

Summary

- RA is a **procedural query language** defined in the relational model.

  An RA query itself suggests a procedure for constructing the result (i.e., implement the query).

- RA is **not used as a query language by users**.

- RA is **used for the internal representation and processing of SQL queries** in relational DBMSs, which is a basis of query optimisation techniques.

- Thus, to understand how SQL queries are processed and how they can be optimised, we **first need to understand relational algebra**.