Database Design and Programming

Tutorial 5: Relational Algebra

Biswadeep Sen
School of Computing
National University of Singapore
biswadeep@u.nus.edu

1.(a) Find the **different departments** in School of Computing.

1.(a) Find the different depts in School of Computing.

Projection Operator

Selection Operator

Rename Operator

$$\underbrace{\pi_{[\texttt{d.department}]}}(\underbrace{\sigma_{[\texttt{d.faculty='School of Computing'}]}}(\underbrace{\rho(\texttt{department,d}))})$$

π (projection) is used to *pick which columns* to display. Here only department

σ filters rows: keep only tuples where `d.faculty = 'School of Computing'` (evaluate after rename, before projection)

ρ (rename) just renames the tables to shorter aliases. Here department to d.

1.(a) Find the different depts in School of Computing.

$$\pi_{[\texttt{d.department}]}(\sigma_{[\texttt{d.faculty='School of Computing'}]}(\rho(\texttt{department,d})))$$

```
SELECT d.department

FROM department d

WHERE d.faculty = 'School of Computing';
```

1.(b) Let us check the integrity of the data. Find the emails of the students who **borrowed** or **lent** a copy of a book **before** they joined the university.

1.(b) Who borrowed OR lent **before** joining?

$$\overbrace{\pi_{[\texttt{s.email}]}}^{\text{Show only } \texttt{s.email}.}\Big(\overbrace{\sigma_{[(\texttt{s.email}=\texttt{l.borrower}\lor\texttt{s.email}=\texttt{l.owner})\land(\texttt{l.borrowed}<\texttt{s.year})]}}^{\text{Keep rows where } \texttt{s.email} \text{ = borrower } \textbf{OR} \text{ owner, } \textbf{and } \texttt{l.borrowed} < \texttt{s.year}.}\Big(\underbrace{\rho(\texttt{student},\texttt{s})\times\rho(\texttt{loan},\texttt{l})}_{\text{Rename } \texttt{student} \to \texttt{s}, \texttt{loan} \to \texttt{l}; \text{ '×' makes all } \texttt{s}\text{–}\texttt{l} \text{ pairs.}}\Big)\Big)$$

📌**NOTE:**

- RA uses set semantics → duplicates removed.
- Evaluation order: ρ → × → σ → π (inside-out).
- $\lor$ = OR; $\land$ = AND
- In this dataset, student.year stores the exact join date (DATE), e.g., 2021-08-01—it is **not** just a year number.

1.(b) Who borrowed OR lent **before** joining?

$$\pi_{[\texttt{s.email}]}(\sigma_{[(\texttt{s.email=l.borrower}\lor\texttt{s.email=l.owner})\land(\texttt{l.borrowed<s.year})]}(\\ \quad \rho(\texttt{student},\texttt{s}) \times \rho(\texttt{loan},\texttt{l})\\ ))$$

```sql
SELECT s.email

FROM student s, loan l

WHERE (s.email = l.borrower OR s.email = l.owner)

AND l.borrowed < s.year;
```

Alternative Solution: INNER JOIN

$$\pi_{[\texttt{s.email}]}\Big(\rho(\texttt{student},\texttt{s}) \bowtie_{[(\texttt{s.email}=\texttt{l.borrower} \vee \texttt{s.email}=\texttt{l.owner}) \wedge (\texttt{l.borrowed}<\texttt{s.year})]} \rho(\texttt{loan},\texttt{l}) \Big)$$

Inner Join

📌 *NOTE:*

Inner join = keep only `s−l` pairs where the student is the borrower **or** owner **and** the loan date is earlier than the student’s year; everything else is discarded, then we project `s.email`

```sql
SELECT s.email

FROM student s

INNER JOIN loan l ON (s.email = l.borrower OR s.email = l.owner)

AND l.borrowed < s.year;
```

Alternative: UNION

$$\overbrace{\pi_{[\texttt{s1.email}]}(\sigma_{[\texttt{s1.email=l1.borrower}\wedge\texttt{l1.borrowed<s1.year}]}(\rho(\texttt{student},\texttt{s1})\times\rho(\texttt{loan},\texttt{l1})))}^{\text{Borrowed a book before joining university}}$$

$$\cup \longleftarrow \text{Union}$$

$$\underbrace{\pi_{[\texttt{s2.email}]}(\sigma_{[\texttt{s2.email=l2.owner}\wedge\texttt{l2.borrowed<s2.year}]}(\rho(\texttt{student},\texttt{s2})\times\rho(\texttt{loan},\texttt{l2})))}_{\text{Lent a book before joining university}}$$

Alternative: UNION

```
SELECT s1.email
FROM loan l1, student s1
WHERE s1.email = l1.borrower
AND l1.borrowed < s1.year
UNION
SELECT s2.email
FROM loan l2, student s2
WHERE s2.email = l2.owner
AND l2.borrowed < s2.year
```

Borrowed a book before joining university

Lent a book before joining university

1.(c) Print the emails of the students who **borrowed** but **did not lend** a copy of a book **on the day** that they joined the university.

1.(c) Borrowed on join day—did not lend

$$\pi_{[\texttt{s1.email}]}(\sigma_{[\texttt{s1.email=l1.borrower} \wedge \texttt{l1.borrowed=s1.year}]}(\rho(\texttt{student}, \texttt{s1}) \times \rho(\texttt{loan}, \texttt{l1})))$$

$$- \longleftarrow \text{Non-symmetric Set Difference}$$

$$\pi_{[\texttt{s2.email}]}(\sigma_{[\texttt{s2.email=l2.owner} \wedge \texttt{l2.borrowed=s2.year}]}(\rho(\texttt{student}, \texttt{s2}) \times \rho(\texttt{loan}, \texttt{l2})))$$

📌**NOTE:** Non-symmetric set difference: A – B keeps the elements in A that aren’t in B — order matters (generally A – B ≠ B – A)

1.(c) Borrowed on join day—did not lend

```sql
SELECT s1.email
FROM loan l1, student s1
WHERE s1.email = l1.borrower
AND l1.borrowed < s1.year
EXCEPT
SELECT s2.email
FROM loan l2, student s2
WHERE s2.email = l2.owner
AND l2.borrowed < s2.year
```

2.(a) Print the emails and the names of the different students who borrowed **all** the books authored **by Adam Smith**.

(2.a) Students who borrowed *all* Adam Smith books

```
SELECT s.email, s.name
FROM student s
WHERE NOT EXISTS (
    SELECT *
    FROM book b
    WHERE authors = 'Adam Smith'
      AND NOT EXISTS (
          SELECT *
          FROM loan l
          WHERE l.book = b.isbn13
            AND l.borrower = s.email  ));
```

📌 *NOTE:*
NOT EXISTS is not directly translatable to relational algebra.

Tiny Example

Q1 – Adam-Smith ISBNs

`{A1, A2, A3}`

Students

`Ana, Ben, Chen, Divya`

Actual borrowings (who borrowed which AS books)

- Ana → `{A1}`
- Ben → `{A1, A2}`
- Chen → `{A1, A2, A3}`
- Divya → `{ }` (none)

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name='ana')`

`loan(…)`: **borrower = student.email**, **book = book.ISBN13**

Tiny Example

Q2 = all possible (student, AS-ISBN) pairs

Students × `{A1,A2,A3}` = 12 pairs:

`(Ana,A1) (Ana,A2) (Ana,A3)`

`(Ben,A1) (Ben,A2) (Ben,A3)`

`(Chen,A1) (Chen,A2) (Chen,A3)`

`(Divya,A1) (Divya,A2) (Divya,A3)`

Q3 = actual borrowed pairs

`(Ana,A1)`

`(Ben,A1) (Ben,A2)`

`(Chen,A1) (Chen,A2) (Chen,A3)`

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name=’ana’)`

`loan(…)`: **borrower = student.email**, **book = book.ISBN13**

Tiny Example

Q2 − Q3 = missing pairs (did NOT borrow)

`(Ana,A2) (Ana,A3)`

`(Ben,A3)`

`(Divya,A1) (Divya,A2) (Divya,A3)`

Q4 = project (email,name) from Q2 − Q3

Drop the ISBN and de-duplicate ⇒ students with **at least one** missing AS book:

`{Ana, Ben, Divya}`

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name='ana')`

`loan(…)`: **borrower = student.email**, **book = book.ISBN13**

Tiny Example

Q5 = all students − Q4

People who borrowed ALL Adam Smith books

$$\overbrace{\{\texttt{Ana, Ben, Chen, Divya}\}}^{\text{All students}} - \overbrace{\{\texttt{Ana, Ben, Divya}\}}^{\text{Missed at least one AS book}} = \{\texttt{Chen}\}$$

Result: only **Chen** borrowed **every** Adam-Smith book.

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name='ana')`

`loan(…)`: **borrower = student.email**, **book = book.ISBN13**

Key idea: `Q2 – Q3` finds *which* book each student still lacks; projecting to students and removing duplicates (**Relational Algebra removes duplicates by default!**) turns that into “who is missing **at least one**,” and subtracting from all students leaves those who are missing **none**.

(2.a) What to do instead of DOUBLE NOT EXISTS

What we can do is to break down the problem into the following subproblems.

1. Find students who missed at least one Adam-Smith book.

2. Subtract them from the full student set → students who borrowed *all* Adam-Smith books.

Step 1 can be split even further:

A. Build all pairs (student, Adam-Smith ISBN).
B. Subtract actual borrowings from A → gives the "missed" pairs.
C. Project to just (student email, name).

2.(a) continued…

$$Q_1 := \pi_{[\texttt{b1.ISBN13}]}(\sigma_{[\texttt{b1.authors='Adam Smith'}]}(\rho(\texttt{book}, \texttt{b1})))$$

Explanation: All ISBN13 of books written by Adam Smith.

$$Q_2 := \pi_{[\texttt{s1.email}, \texttt{s1.name}]}(\rho(\texttt{student}, \texttt{s1})) \times Q_1$$

Explanation: Every student paired with every Adam-Smith ISBN.
The columns are [email,name, ISBN13]

Tiny Example

Q2 = all possible (student, AS-ISBN) pairs

Students × `{A1,A2,A3}` = 12 pairs:

`(Ana,A1) (Ana,A2) (Ana,A3)`

`(Ben,A1) (Ben,A2) (Ben,A3)`

`(Chen,A1) (Chen,A2) (Chen,A3)`

`(Divya,A1) (Divya,A2) (Divya,A3)`

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name='ana')`

`loan(…)`: **borrower = student.email**, **book = book.ISBN13**

2.(a) continued…

$$Q_3 := \overbrace{\pi_{[\texttt{s2.email},\texttt{s2.name},\texttt{b2.ISBN13}]}}^{3}\Big(\overbrace{\rho(\texttt{loan},\texttt{l2}) \bowtie_{[\texttt{l2.book}=\texttt{b2.ISBN13} \wedge \texttt{b2.authors}=\texttt{'Adam Smith'}]} \rho(\texttt{book},\texttt{b2})}^{1} \underbrace{\bowtie_{[\texttt{l2.borrower}=\texttt{s2.email}]} \rho(\texttt{student},\texttt{s2})\Big)}_{2}$$

Explanation: Build all **student–ISBN** pairs that correspond to a **real borrowing** of an **Adam-Smith** book.

1. Join **loan l2** to **book b2** on `l2.book = b2.ISBN13`, then **filter** `b2.authors = 'Adam Smith'` → only Adam-Smith loans remain.
2. Join that result to **student s2** on `l2.borrower = s2.email` → attach the borrower's identity.
3. Finally **project** to `[s2.email, s2.name, b2.ISBN13]`.

Schema at this step: `[email, name, ISBN13]` (set semantics ⇒ duplicates removed).

Tiny Example

Q3 = actual borrowed pairs

`(Ana,A1)`

`(Ben,A1) (Ben,A2)`

`(Chen,A1) (Chen,A2) (Chen,A3)`

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name='ana')`

`loan(…)`: **borrower = student.email**, **book = book.ISBN13**

2.(a) continued…

$$Q_4 := \pi_{[\texttt{s2.email},\texttt{s2.name}]}(Q_2 - Q_3)$$

Explanation: Subtract borrowed pairs from all pairs, then project to students → anyone who **did not** borrow at least one Adam-Smith book.

$$Q_5 := \pi_{[\texttt{s3.email},\texttt{s3.name}]}(\rho(\texttt{student},\texttt{s3})) - Q_4$$

Explanation: All students minus those who missed at least one → students who borrowed **every** Adam-Smith book.

Tiny Example

Q4 = project (email,name) from Q2 − Q3

Drop the ISBN and de-duplicate ⇒ students with **at least one** missing AS book:

`{Ana, Ben, Divya}`

Q5 = all students − Q4

People who borrowed ALL Adam Smith books

`{Ana, Ben, Chen, Divya} - {Ana, Ben, Divya} = {Chen}`

```
Final Result = Chen ≡ { ( chen@u.edu, Chen) }
```

📝 **SHORTHANDS:**

Ana ≡ `student(email='ana@u.edu', name='ana')`

Q2 − Q3 = missing pairs (did NOT borrow)

```
(Ana,A2) (Ana,A3)
(Ben,A3)
(Divya,A1) (Divya,A2) (Divya,A3)
```

(2.a) Students who borrowed *all* Adam Smith books

```sql
SELECT s3.email, s3.name
FROM student s3
EXCEPT
SELECT tmp.email, tmp.name
FROM (
SELECT s1.email, s1.name, b1.ISBN13
FROM student s1, book b1
WHERE b1.authors = 'Adam Smith’
EXCEPT
SELECT s2.email, s2.name, b2.ISBN13
FROM student s2, book b2, loan l2
WHERE b2.authors = 'Adam Smith'
AND s2.email = l2.borrower
AND b2.ISBN13 = l2.book
) AS tmp;
```

(2.a) Students who borrowed *all* Adam Smith books

```
SELECT s3.email, s3.name
FROM student s3
EXCEPT
SELECT tmp.email, tmp.name
FROM (
SELECT s1.email, s1.name, b1.ISBN13
FROM student s1, book b1
WHERE b1.authors = 'Adam Smith'
EXCEPT
SELECT s2.email, s2.name, b2.ISBN13
FROM student s2, book b2, loan l2
WHERE b2.authors = 'Adam Smith'
AND s2.email = l2.borrower
AND b2.ISBN13 = l2.book
) AS tmp;
```

Q2: builds **all possible (student, Adam-Smith-ISBN)** pairs.

Q3: builds **actual** (student, Adam-Smith ISBN) **borrowings** (joins student↔loan and loan↔book).

(2.a) Students who borrowed *all* Adam Smith books

```
SELECT s3.email, s3.name
FROM student s3
EXCEPT
SELECT tmp.email, tmp.name
FROM (
SELECT s1.email, s1.name, b1.ISBN13
FROM student s1, book b1
WHERE b1.authors = 'Adam Smith'
EXCEPT
SELECT s2.email, s2.name, b2.ISBN13
FROM student s2, book b2, loan l2
WHERE b2.authors = 'Adam Smith'
AND s2.email = l2.borrower
AND b2.ISBN13 = l2.book
) AS tmp;
```

Q4: Q2 – Q3, then project to **(email, name)** ⇒ students **missing at least one** “Adam Smith” book.

(2.a) Students who borrowed *all* Adam Smith books

```sql
SELECT s3.email, s3.name
FROM student s3
EXCEPT
SELECT tmp.email, tmp.name
FROM (
SELECT s1.email, s1.name, b1.ISBN13
FROM student s1, book b1
WHERE b1.authors = 'Adam Smith’
EXCEPT
SELECT s2.email, s2.name, b2.ISBN13
FROM student s2, book b2, loan l2
WHERE b2.authors = 'Adam Smith'
AND s2.email = l2.borrower
AND b2.ISBN13 = l2.book
) AS tmp;
```

Q5: All students − Q4 = students who borrowed every “Adam Smith” book.

Thank you for joining!

Got questions? Post them on the forum or email me:
biswadeep@u.nus.edu
(I reply **within 2 working days** — *faster if coffee is strong* ☕)

Because your learning matters to me! 😊

NUS
National University
of Singapore