CZ2007 Introduction to Database Systems (Week 5)

Topic 5: Relational Algebra (1)

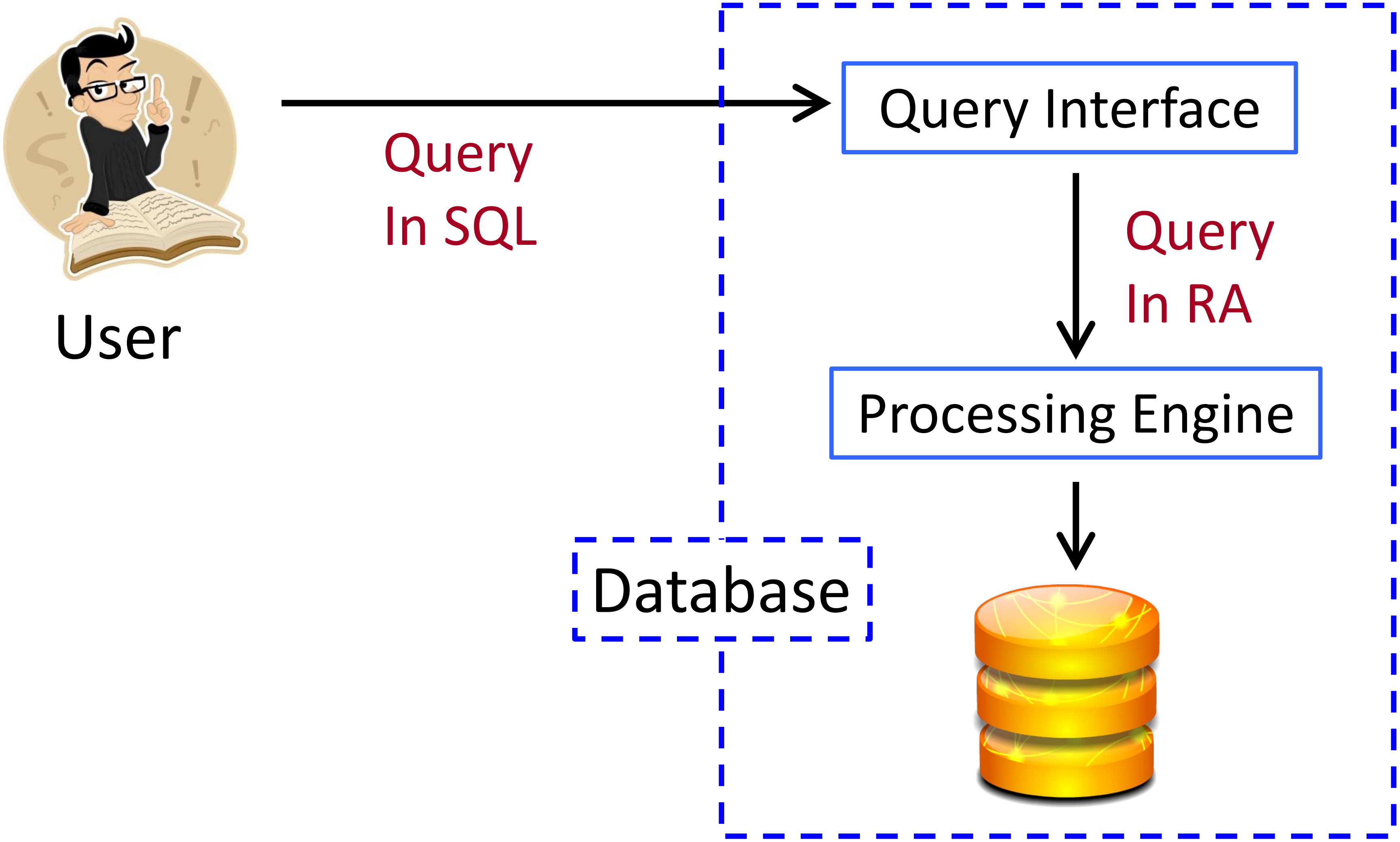
This Lecture

- Motivation for relational algebra
- Relational algebraic operators
  - Selection: $\sigma_{A > 100} R_1$
  - Projection: $\Pi_{A,\ B} R_1$
  - Union: $R_1 \cup R_2$
  - Intersection: $R_1 \cap R_2$
  - Difference: $R_1 - R_2$
  - Natural Join: $R_1 \bowtie R_2$
  - Theta Join: $R_1 \bowtie_{R1.A=R2.A \text{ AND } R1.B<R2.B} R_2$

Relational Algebra: Motivation

- We have specification of an DB application
- We use ER-diagram for a conceptual design of database
- We transform ER-diagram into database schema (i.e., the schemas of a set of tables)
- We normalize the schema, and then insert some tuples into the tables
- Now what?
- How do we perform queries on those tables?
  - Database side: Relational Algebra (RA)
  - User side: Structured Query Language (SQL)

Relational Algebra: Motivation



User

Query
In SQL

Query Interface

Query
In RA

Processing Engine

Database

Relational Algebra

- A mathematical way to formulate queries on relations (i.e., tables)
- Has numerous operators for query formulation
- Example
  - Given: Two relations $R_1(A, B, C)$, $R_2(A, B, C)$
  - Selection: $\sigma_{A > 100} R_1$
  - Projection: $\Pi_{A, B} R_1$
  - Union: $R_1 \cup R_2$
  - Intersection: $R_1 \cap R_2$
  - And a few others…

Selection σ (row-wise operation)

Students

<u>ID</u>	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: "Find me the <u>student</u> named <u>Alice</u>"
- $\sigma_{\text{Name = 'Alice'}}$ Students

Results

<u>ID</u>	Name	Age	School
1234	Alice	20	SCSE

Selection σ

Students

ID	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: "Find the students in SCSE"
- $\sigma_{\text{School = 'SCSE'}}$ Students

Results

ID	Name	Age	School
1234	Alice	20	SCSE
3742	Cathy	22	SCSE

Selection σ

Students

ID	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: "Find the SCSE students under 21"
- $\sigma_{\text{School = 'SCSE' AND Age < 21}}$ Students

Results

ID	Name	Age	School
1234	Alice	20	SCSE

Selection σ

Students

<u>ID</u>	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: "Find the <u>students</u> who are either in <u>SCSE</u> or <u>under</u> <u>21</u>"
- $\sigma_{\text{School = 'SCSE' OR Age < 21}}$ Students

Projection Π (column-wise)

Students

ID	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: “Find the IDs and Names of all students”
- $\Pi_{\text{ID, Name}}$ Students

Results

ID	Name
1234	Alice
5678	Bob
3742	Cathy
9413	David

Combining Operators

Students

ID	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: “Find the IDs and Names of all students in SCSE”
- $\Pi_{\text{ID, Name}} (\sigma_{\text{School = ‘SCSE’}} \text{ Students})$

Results

ID	Name
1234	Alice
3742	Cathy

Combining Operators

Students

ID	Name	Age	School
1234	Alice	20	SCSE
5678	Bob	20	EEE
3742	Cathy	22	SCSE
9413	David	21	CEE

- Query: “Find the IDs and Names of all students in SCSE”
- How about $\sigma_{\text{School = 'SCSE'}} (\Pi_{\text{ID, Name}}$ Students)?
- Wrong
- The projection goes before the selection here
- Since the projection eliminates “School’, the selection cannot be performed

Union ∪

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name	Age
Cathy	22
David	21
Eddie	43
Fred	35

Results

Name	Age
Alice	20
Bob	21
Cathy	22
David	21
Cathy	22
David	21
Eddie	43
Fred	35

- Query: “Find the persons who are eithe volunteers”
- Students ∪ Volunteer
-

Union ∪

Students

<u>Name</u>	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

<u>Name</u>	Age
Cathy	22
David	21
Eddie	43
Fred	35

Results

Name	Age
Alice	20
Bob	21
Cathy	22
David	21
Eddie	43
Fred	35

- Query: “Find the persons who are either <u>students</u> or <u>volunteers</u>”
- Students ∪ Volunteer
- Note 1: Duplicate tuples are automatically removed

Union ∪

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name	Age
Cathy	22
David	21
Eddie	43
Fred	35

Results

Name
Alice
Bob
Cathy
David
Eddie
Fred

- Query: “Find the names of the persons who are either students or volunteers”
- Π_{Name} (Students ∪ Volunteer)
- (Π_{Name} Students) ∪ (Π_{Name} Volunteer)

Union ∪

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name
Cathy
David
Eddie
Fred

Results

Name
Alice
Bob
Cathy
David
Eddie
Fred

- Query: “Find the persons who are either students or volunteers”
- Students ∪ Volunteer ?
- Wrong
- Note 2: The two sides of a union must have the same schema (i.e., the same set of attributes)
- Correct solution: (Π_{Name} Students) ∪ Volunteer

Intersection ∩

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name	Age
Cathy	22
David	21
Eddie	43
Fred	35

Results

Name	Age
Cathy	22
David	21

- Query: “Find the persons who are both students and volunteers”
- Students ∩ Volunteer
- Note 1: Duplicate tuples are automatically removed

Intersection $\cap$

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name
Cathy
David
Eddie
Fred

Results

Name
Cathy
David

- Query: “Find the persons who are both students and volunteers”
- $(\Pi_{\text{Name}}\ \text{Students}) \cap \text{Volunteer}$
- Note 2: The two sides of an intersection must have the same schema (i.e., the same set of attributes)

Difference –

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name	Age
Cathy	22
David	21
Eddie	43
Fred	35

Results

Name	Age
Alice	20
Bob	21

- Query: "Find the persons who are students but not volunteers"
- Students – Volunteer
- Note 1: Duplicate tuples are automatically removed

Difference –

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name	Age
Cathy	22
David	21
Eddie	43
Fred	35

Results

Name	Age
Eddie	43
Fred	35

- Query: “Find the persons who are volunteers but not students”
- Volunteer – Students

Difference –

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name
Cathy
David
Eddie
Fred

Results

Name
Alice
Bob

- Query: “Find the persons who are students but not volunteers”
- $(\Pi_{\text{Name}}$ Students) – Volunteer
- Note 2: The two sides of a difference must have the same schema (i.e., the same set of attributes)

Exercise

Name	Course	Grade
Alice	DB	A
Bob	DB	B

Name	Course	Grade
Alice	DM	C

Name	Course	Grade
Bob	AI	B
Cathy	CG	A

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

- Query: “Find the students who have taken DB and DM, but not AI or CG”
- $((\underline{\sigma_{Course\ =\ 'DB'}\ Grades}) \cap (\underline{\sigma_{Course\ =\ 'DM'}\ Grades})) - ((\underline{\sigma_{Course\ =\ 'AI'}\ Grades) \cup (\sigma_{Course\ =\ 'CG'}\ Grades}))$
- Result is empty set
- Wrong

Exercise

Name	Course	Grade
Alice	DB	A
Bob	DB	B

Name	Course	Grade
Alice	DM	C

Name	Course	Grade
Bob	AI	B
Cathy	CG	A

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

- Query: “Find the students who have taken DB and DM, but not AI or CG”
- $((\Pi_{\text{Name}}\ \sigma_{\text{Course = 'DB'}}\ \text{Grades}) \cap (\Pi_{\text{Name}}\ \sigma_{\text{Course = 'DM'}}\ \text{Grades})) - ((\Pi_{\text{Name}}\ \sigma_{\text{Course = 'AI'}}\ \text{Grades}) \cup (\Pi_{\text{Name}}\ \sigma_{\text{Course = 'CG'}}\ \text{Grades}))$

Exercise

Name	Course	Grade
Alice	DB	A
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

- Query: “Find the students who have never taken DM”
- $\sigma_{\text{Course} \neq \text{‘DM’}}$ Grades
- Alice has taken DM but still appear in the result
- Wrong

Exercise

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

Name	Course	Grade
Alice	DB	A
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

Name	Course	Grade
Alice	DM	C

- Query: “Find the students who have never taken DM”
- Grades – $(\sigma_{\text{Course = ‘DM’}}$ Grades)
- Alice has taken DM but still appear in the result
- Wrong

Exercise

Name
Alice
Bob
Bob
Cathy
David

Name
Alice

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	AI	B
Cathy	CG	A
David	NN	C

- Query: “Find the students who have never taken DM”
- $(\Pi_{\text{Name}}\ \text{Grades}) - (\Pi_{\text{Name}}\ \sigma_{\text{Course = 'DM'}}\ \text{Grades})$

Natural Join ⋈

Students

NRIC	Name
11	Alice
2	Bob
33	Cathy
4	David

Phones

NRIC	Number
11	9123234
11	8635168
33	8213654
5	9653154

Results

NRIC	Name	Number
11	Alice	9123234
11	Alice	8635168
33	Cathy	8213654

- Query: “Find the NRIC, Name, and Phone of each student, omitting those without a phone” (those without phone will not appear in table)
- Students ⋈ Phones
- Note 1: The join is performed based on the common attributes of the two relations
- Note 2: Each common attribute appears only once in the result

Natural Join ⋈

Students

Name	School
Alice	SCSE
Bob	EEE
Cathy	CEE
David	SCSE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
Cathy	CEE	100
David	SCSE	200

- Students ⋈ Donations
- Meaning: “For those students who have made donation, find their names, schools, and amounts of their donations”

Natural Join ⋈

Students

Name	School
Alice	SCSE
Bob	EEE
Cathy	CEE
David	SCSE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
David	SCSE	200

- $(\sigma_{\text{School = 'SCSE'}}$ Students) ⋈ Donations
- Meaning: “For those SCSE students who have made a donation, find their names, schools, and amounts of their donations”

Exercise

Grades

<u>Name</u>	<u>Course</u>	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	NN	B
Cathy	SP	B
Cathy	NN	A

CrsSch

<u>Course</u>	School
DB	SCSE
DM	SCSE
AI	SCSE
NN	EEE
SP	EEE

Results

<u>Name</u>
Alice

- Query: "Find the students who have taken SCSE courses but not EEE courses"
- $(\Pi_{\text{Name}}\ \text{Grades} \bowtie (\sigma_{\text{School = 'SCSE'}}\ \text{CrsSch})) -$
  $(\Pi_{\text{Name}}\ \text{Grades} \bowtie (\sigma_{\text{School = 'EEE'}}\ \text{CrsSch}))$

Exercise

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	NN	B
Cathy	SP	B
Cathy	NN	A

CrsSch

Course	School
DB	SCSE
DM	SCSE
AI	SCSE
NN	EEE
SP	EEE

- Query: “Find the students who have only taken EEE courses”
- How to eliminate Bob who has taken SCSE courses?

Exercise

Grades

Name	Course	Grade
Alice	DB	A
Alice	DM	C
Bob	DB	B
Bob	NN	B
Cathy	SP	B
Cathy	NN	A

CrsSch

Course	School
DB	SCSE
DM	SCSE
AI	SCSE
NN	EEE
SP	EEE

Results

Name
Cathy

- Query: “Find the students who have only taken EEE courses”
- Π_{Name} Grades – (Π_{Name} Grades ⋈ ($\sigma_{\text{School <> 'EEE'}}$ CrsSch))

Theta Join $\bowtie_{\text{condition}}$

Students

SName	School
Alice	SCSE
Bob	EEE
Cathy	CEE
David	SCSE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

SName	Name	School	Amount
Cathy	Cathy	CEE	100
David	David	SCSE	200

- Query: “For those students who have made donations, find their names, schools, and amounts of their donations”
- Students $\bowtie_{\text{Sname=Name}}$ Donations
- Difference from natural join: Duplicate attributes will NOT be removed from the results
- In general, the join condition in a theta join can also be inequalities

Theta Join $\bowtie_{\text{condition}}$

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Quiz2

Name	Score
Alice	80
Bob	90
Cathy	90
David	70

Results

Name	Score	Name	Score
Alice	70	Alice	80
Cathy	80	Cathy	90

- Query: “Find the students who score higher in quiz 2 than quiz 1”
- Quiz1 $\bowtie_{\text{Quiz1.Name = Quiz2.Name AND Quiz1.Score < Quiz2.Score}}$ Quiz2
- Note: In the join condition, whenever there are ambiguous attribute names (e.g., Score), we need to add the table names along with the attribute names to eliminate the ambiguity (e.g., by using Quiz1.Score instead of Score)

Cartesian Product ×

Students

Name	Age
Alice	19
Bob	22

Courses

ID	Name
C1	DB
C2	Algo

Results

Name	Age	ID	Name
Alice	19	C1	DB
Alice	19	C2	Algo
Bob	22	C1	DB
Bob	22	C2	Algo

- Effect: Theta join without a condition
- Query: “Create a table that provides all possible student-course combinations”
- Students × Donations

Next lecture:

Topic 5: Relational Algebra (2)