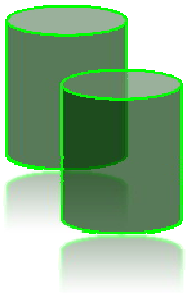

Database Systems Part-4

Relational Algebra

Arijit Khan

Associate Professor
Department of Computer Science
Aalborg University, Denmark

arijitk@cs.aau.dk

Schedule for first half

Intro & ER	Sep 5 Monday (12:30-14:15), Lecture - FRB 7H	No exercise
Relation, FD, Keys	Sep 12 Monday (12:30-14:15), Lecture - FRB 7H	Exercise-1 on Intro & ER (14:30-16:30)
Normalization	Sep 19 Monday (12:30-14:15), Lecture - FRB 7H	Exercise-2 on Relation, FD, Keys (14:30-16:30)
Relational Algebra	Exercise-3 on Normalization (8:15-10:00)	Sep 23 Friday (10:15-12:00), Lecture - NOVI8
Release of Self-Study-1	Sep 26 Monday, No lecture, No exercise	
SQL – Part I	Oct 3 Monday (12:30-14:15), Lecture - FRB 7H	Exercise-4 on Relational Algebra (14:30-16:30)
Feedback of Self-Study-1	Oct 10 Monday (12:30-14:15), During lecture - FRB 7H	Exercise-5 on SQL – Part I (14:30-16:30)

Ask Questions!

The important thing is not to stop questioning.

Albert Einstein

Relational Algebra: Motivation

- We have the specification of an DB application
- We use an ER-diagrams for a conceptual design of our database
- We transform the ER-diagram into a 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

Roadmap

- Relational Algebra
  - This lectures
- SQL
  - Lectures that follow

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

Students

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

- Query: “Find me the student named Alice”
- $\sigma_{\text{Name = 'Alice'}}$ Students

Results

ID	Name	Age	School
1234	Alice	20	SCE

Selection σ

Students

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

- Query: “Find the students in SCE”
- $\sigma_{\text{School} = \text{'SCE'}}$ Students

Results

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

Selection σ

Students

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

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

Results

ID	Name	Age	School
1234	Alice	20	SCE

Selection σ

Students

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

- Query: “Find the students who are either in SCE or under 21”
- $\sigma_{\text{School} = \text{'SCE'} \text{ OR } \text{Age} < 21}$ Students

Projection Π

Students

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

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

Results

ID	Name
1234	Alice
3742	Cathy

Combining Operators

Students

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

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

Questions?

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
Eddie	43
Fred	35

- Query: “Find the persons who are either students or volunteers”
- 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: $(\Pi_{\text{Name}}$ Students) ∪ 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

In order to be used in a UNION, the tables must have the same attribute characteristics, that it the attributes and their domains must be compatible. **When two or more tables share the same number of columns and when their corresponding columns share the same or compatible domains**, they are said to be union-compatible.

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 ∩

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)

Intersection ∩

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name
Cathy
David
Eddie
Fred

Results

Name
Cathy
David

In order to be used in an INTERSECTION, the tables must have the same attribute characteristics, that it the attributes and their domains must be compatible. **When two or more tables share the same number of columns and when their corresponding columns share the same or compatible domains**, they are said to be intersection-compatible.

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)

Difference –

Students

Name	Age
Alice	20
Bob	21
Cathy	22
David	21

Volunteer

Name
Cathy
David
Eddie
Fred

Results

Name
Alice
Bob

In order to be used in a DIFFERENCE, the tables must have the same attribute characteristics, that it the attributes and their domains must be compatible. **When two or more tables share the same number of columns and when their corresponding columns share the same or compatible domains**, they are said to be DIFFERENCE-compatible.

Questions?

Classroom Exercise-1

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”
- $((\sigma_{\text{Course = 'DB'}} \text{Grades}) \cap (\sigma_{\text{Course = 'DM'}} \text{Grades})) - ((\sigma_{\text{Course = 'AI'}} \text{Grades}) \cup (\sigma_{\text{Course = 'CG'}} \text{Grades}))$
- Wrong!

Classroom Exercise-1: Solution

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”
- $((\Pi_{\text{Name}}\ \sigma_{\text{Course = 'DB'}}\ \text{Grades}) \cap (\Pi_{\text{Name}}\ \sigma_{\text{Course = 'DM'}}\ \text{Grades})) - ((\Pi_{\text{Name}}\ \sigma_{\text{Course = 'AI'}}\ \text{Grades}) \cup (\Pi_{\text{Name}}\ \sigma_{\text{Course = 'CG'}}\ \text{Grades}))$

Classroom Exercise-2: Solution

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”
- 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	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
Cathy	CEE	100
David	SCE	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	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
David	SCE	200

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

Classroom Exercise-3

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	SCE
DM	SCE
AI	SCE
NN	EEE
SP	EEE

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

Classroom Exercise-4

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	SCE
DM	SCE
AI	SCE
NN	EEE
SP	EEE

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

Questions?

Theta Join $\bowtie_{condition}$

Students

SName	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

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

- Query: "For those students who have made donations, find their names, schools, and amounts of their donations"
- Students $\bowtie_{Sname=Name}$ 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 $\bowtie_{condition}$

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

Assignment :=

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Evaluation1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Over85

Name	Score
Bob	90
David	100

- Conceptually: Make another copy of the table and give it a new name
- Example
  - Evaluation1 := Quiz1
  - Over85 := $\sigma_{Score > 85}$ Quiz1
- Note: All attribute names are retained

Assignment :=

- Useful for breaking down steps
- Example:
  - $(\Pi_{\text{Name}}\ \text{Students}) \cup (\Pi_{\text{Name}}\ \text{Volunteer})$
- Equivalent Representation
  - R1 := Π_{Name} Students
  - R2 := Π_{Name} Volunteer
  - R1 $\cup$ R2
- This makes your solution easier to write and easier for others to understand

Rename ρ

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Evaluation1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Eval1

SName	QScore
Alice	70
Bob	90
Cathy	80
David	100

- Similar to assignment, but allows change of attribute names
- Example
  - $\rho_{Evaluation1}$ Quiz1
  - $\rho_{Eval1(SName,\ QScore)}$ Quiz1

5 Minutes Interval

The important thing is not to stop questioning.

Albert Einstein

Classroom Exercise-5

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

R1

Name	Score
Cathy	80

R2

Name1	Score1	Name2	Score2
Bob	90	Cathy	80
David	100	Cathy	80

R3

Name1	Score1
Bob	90
David	100

- Find the students who score higher than Cathy in Quiz1
- R1 := $\sigma_{\text{Name='Cathy'}}$ Quiz1
- $\rho_{\text{R2(Name1, Score1, Name2, Score2)}}$ Quiz1 $\bowtie_{\text{Quiz1.Score > R1.Score}}$ R1
- R3 := $\Pi_{\text{Name1, Score1}}$ R2

Classroom Exercise-6

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80

R1

Name	Score
Alice	70
Bob	90
Cathy	80

R2

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

R4

Name
Bob

- Find the students who score the highest in Quiz1
- R1 := Quiz1
- R2 := Quiz1 $\bowtie_{\text{Quiz1.Name <> R1.Name AND Quiz1.Score < R1.Score}}$ R1
- R3 := $\Pi_{\text{Quiz1.Name}}$ R2
- R4 := $\Pi_{\text{Quiz1.Name}}$ Quiz1 – R3

Duplicate Elimination δ

Purchase

Name	Product	Date
Alice	iPhone	2016.01.01
Bob	Xbox	2016.01.01
Cathy	iPhone	2016.01.01
David	Xbox	2016.02.17

R1

Product
iPhone
Xbox
iPhone

R2

Product
iPhone
Xbox

- Effect: Eliminate duplicate tuples
- Query: Find the list of products sold on 2016.01.01
- R1 := Π_{Product} ($\sigma_{\text{Date='2013.01.01'}}$ Purchase)
- R2 := δ(R1)

Extended Projection Π

Scores

Name	Quiz1	Quiz2
Alice	70	90
Bob	90	80
Cathy	80	100
David	100	90

Results

Name	Total
Alice	160
Bob	170
Cathy	180
David	190

- Similar to ordinary projection, but allows the creation of new attributes via arithmetic
- Query: "For each student, find his/her total score in Quiz 1 and 2"
- $\Pi_{\text{Name, Quiz1 + Quiz2} \rightarrow \text{Total}}$ Scores
- The left hand side of "→" gives the arithmetic performed
- The right hand side gives an attribute name to the result

Extended Projection Π

Scores

Name	Quiz1	Quiz2
Alice	70	90
Bob	90	80
Cathy	80	100
David	100	90

Results

Name	Average
Alice	80
Bob	85
Cathy	90
David	95

- Similar to ordinary projection, but allows the creation of new attributes via arithmetic
- Query: “For each student, find his/her average score in Quiz 1 and 2”
- $\Pi_{\text{Name, (Quiz1 + Quiz2)/2} \rightarrow \text{Average}}$ Scores

Grouping and Aggregation γ

Quiz1

Name	School	Score
Alice	SCE	90
Bob	EEE	80
Cathy	EEE	100
David	SCE	90

Results

MaxScore
100

- Query: “Find the highest score in Quiz1”
- $\gamma_{\text{MAX(Score)} \rightarrow \text{MaxScore}}$ Quiz1
- The attribute name on right hand side of “→” can be arbitrary

Grouping and Aggregation γ

Quiz1

Name	School	Score
Alice	SCE	90
Bob	EEE	80
Cathy	EEE	100
David	SCE	90

Results

MinScore
80

- Query: “Find the lowest score in Quiz1”
- $\gamma_{\text{MIN(Score)} \rightarrow \text{MinScore}}$ Quiz1

Grouping and Aggregation γ

Quiz1

Name	School	Score
Alice	SCE	90
Bob	EEE	80
Cathy	EEE	100
David	SCE	90

Results

AvgScore
90

- Query: “Find the average score in Quiz1”
- $\gamma_{\text{AVG(Score)} \rightarrow \text{AvgScore}}$ Quiz1

Grouping and Aggregation γ

Quiz1

Name	School	Score
Alice	SCE	90
Bob	EEE	80
Cathy	EEE	100
David	SCE	90

Results

SumScore
360

- Query: "Find the sum of scores in Quiz1"
- $\gamma_{\text{SUM(Score)} \rightarrow \text{SumScore}}$ Quiz1

Grouping and Aggregation γ

Quiz1

Name	School	Score
Alice	SCE	90
Bob	EEE	80
Cathy	EEE	100
David	SCE	90

Results

NumStu
4

- Query: “Find the number of students in Quiz1”
- $\gamma_{\text{COUNT(Name)} \rightarrow \text{NumStu}}$ Quiz1
- $\gamma_{\text{COUNT(School)} \rightarrow \text{NumStu}}$ Quiz1
- $\gamma_{\text{COUNT(Score)} \rightarrow \text{NumStu}}$ Quiz1
- All three queries above give the number of tuples in Quiz1

Aggregate Functions

- MAX(...)
- MIN(...)
- AVG(...)
- SUM(...)
- COUNT(...)

Grouping and Aggregation γ

Quiz1

Name	School	GPA
Alice	SCE	4
Bob	EEE	3
Cathy	EEE	3.4
David	SCE	3.6

Results

School	AvgGPA
SCE	3.8
EEE	3.2

- Query: "Find the average GPA in each school"
- $\gamma_{\text{School, AVG(GPA)} \rightarrow \text{AvgGPA}}$ Quiz1
- Effect: Divide tuples into separate groups based on their "School" value, and then compute the average GPA in each group

Grouping and Aggregation γ

Quiz1

Name	School	GPA
Alice	SCE	4
Bob	EEE	3
Cathy	EEE	3.4
David	SCE	3.6

Results

School	AvgGPA	MaxGPA
SCE	3.8	4
EEE	3.2	3.4

- Query: "Find the average GPA and highest GPA in each school"
- $\gamma_{\text{School, AVG(GPA)} \rightarrow \text{AvgGPA, MAX(GPA)} \rightarrow \text{MaxGPA}}$ Quiz1

Grouping and Aggregation γ

Quiz1

Name	School	Year	GPA
Alice	SCE	3	4
Bob	EEE	1	3
Cathy	EEE	2	3.4
David	SCE	3	3.6

Results

School	Year	GPA
SCE	3	3.8
EEE	1	3
EEE	2	3.4

- $\gamma_{School, Year, AVG(GPA) \rightarrow AvgGPA}$ Quiz1
- Effect: Divide tuples into separate groups based on their "School, year" value combination, and then compute the average GPA in each group

Common Mistake

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

R1

MaxScore
100

R2

Name	Score	MaxScore
David	100	100

R3

Name
David

- Query: “Find the student that scores the highest in Quiz1”
- $\sigma_{\text{Score} = \text{MAX(Score)}}$ Quiz1 ?
- Wrong: Aggregate functions can only be used with the aggregation operation γ
- R1 := $\gamma_{\text{MAX(Score)} \rightarrow \text{MaxScore}}$(Quiz1)
- R2 := Quiz1 $\bowtie_{\text{Score} = \text{MaxScore}}$ R1
- R3 := Π_{Name}(R2)

Questions?

Division ÷

Owns

Name	Product
Alice	iPad
Alice	iPhone
Bob	iPhone
Cathy	iPad

AppleP

Product
iPhone
iPad

Results

Name
Alice

- Query: “Find each person that owns all Apple products”
- Owns ÷ AppleP
- In general, $R_1(A, B) \div R_2(B)$ returns a table that contains only A
- The table contains each A value in R_1 that is associated with every B value in R_2
- Intuitive interpretation: “Find the A that R_1 all the B in R_2”
- Example: “Find the Name that Owns all the Product in AppleP”

Division ÷

Joins

Name	Club
Alice	ABC
Bob	DEF
Bob	ABC
Cathy	DEF

Clubs

Club
ABC
DEF

Results

Name
Bob

- Query: “Find each person that has joined all clubs”
- Joins ÷ Clubs

Division ÷

Joins

Name	Club
Alice	ABC
Bob	DEF
Bob	ABC
Cathy	DEF

Clubs

Name
ABC
DEF

Results

Club

- Joins ÷ Clubs ?
- No result.
- In general, $R_1(A, B) \div R_2(B)$ returns a table that contains only A

Division ÷

Owns

Name	Product
Alice	iPad
Alice	iPhone
Bob	iPhone
Cathy	iPad

AppleP

Product	Price
iPhone	999
iPad	699

Results

Name
Alice

- Query: “Find each person that owns all Apple products”
- Owns ÷ AppleP ?
- Wrong, since “Price” does not appear in “Owns”
- In general, $R_1(A, B) \div R_2(B)$ returns a table that contains only A
- Correct answer: Owns ÷ $(\Pi_{Product}$AppleP)

Division ÷

Owns

Name	Since	Product
Alice	2013	iPhone
Alice	2013	iPad
Bob	2013	iPhone
Bob	2010	iPad

AppleP

Product
iPhone
iPad

Results

Name	Since
Alice	2013

- Owns ÷ AppleP
- The result is a table with attributes in Owns but not in AppleP, i.e., Name and Since
- The table contains every {Name, Since} combination that is associated with all Product in AppleP

Questions?

Left Outerjoin $\overset{\circ}{\bowtie}_L$ condition

Students

Name	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
Alice	SCE	NULL
Bob	EEE	NULL
Cathy	CEE	100
David	SCE	200

- Query: "For each student, find the amount of his/her donation"
- Students $\overset{\circ}{\bowtie}_L$ Donations
- All tuples in Students are retained in the results
- For each student who has not made a donation, a "NULL" value is given as his/her Amount

Left Outerjoin $\overset{\circ}{\bowtie}_{L\ \text{condition}}$

Students

SName	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

SName	Name	School	Amount
Alice	NULL	SCE	NULL
Bob	NULL	EEE	NULL
Cathy	Cathy	CEE	100
David	David	SCE	200

- Query: "For each student, find the amount of his/her donation"
- Students $\overset{\circ}{\bowtie}_{L\ \text{Sname = Name}}$ Donations
- Similar to theta joins in that all attributes are retained

Right Outerjoin $\overset{\circ}{\bowtie}_R$ condition

Students

Name	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
Cathy	SCE	100
David	EEE	200
Eddie	NULL	300
Fred	NULL	400

- Query: "For each donator, find the school he/she is in"
- Students $\overset{\circ}{\bowtie}_R$ Donations
- All tuples in Donations are retained in the results
- For each donator who is not students, a "NULL" value is given as his/her School

Right Outerjoin $\overset{\circ}{\bowtie}_{R\ \text{condition}}$

Students

SName	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

SName	Name	School	Amount
Cathy	Cathy	SCE	100
David	David	EEE	200
NULL	Eddie	NULL	300
NULL	Fred	NULL	400

- Query: "For each donator, find the school he/she is in"
- Students $\overset{\circ}{\bowtie}_{R\ \text{Sname = Name}}$ Donations
- Similar to theta joins in that all attributes are retained

Full Outerjoin $\overset{\circ}{\bowtie}_{\text{condition}}$

Students

Name	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

Name	School	Amount
Alice	SCE	NULL
Bob	EEE	NULL
Cathy	SCE	100
David	EEE	200
Eddie	NULL	300
Fred	NULL	400

- The combination of left and right outerjoins
- Students $\overset{\circ}{\bowtie}$ Donations

Full Outerjoin $\overset{\circ}{\bowtie}_{\text{condition}}$

Students

SName	School
Alice	SCE
Bob	EEE
Cathy	CEE
David	SCE

Donations

Name	Amount
Cathy	100
David	200
Eddie	300
Fred	400

Results

SName	Name	School	Amount
Alice	NULL	SCE	NULL
Bob	NULL	EEE	NULL
Cathy	Cathy	SCE	100
David	David	EEE	200
NULL	Eddie	NULL	300
NULL	Fred	NULL	400

- Students $\overset{\circ}{\bowtie}_{\text{Sname = Name}}$ Donations

Questions?

Summary

- **Relational Algebra**

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

R1

Name	Score
Cathy	80

R2

Name1	Score1	Name2	Score2
Bob	90	Cathy	80
David	100	Cathy	80

R3

Name1	Score1
Bob	90
David	100

- Find the students who score higher than Cathy in Quiz1
- R1 := $\sigma_{\text{Name='Cathy'}}$ Quiz1
- $\rho_{\text{R2(Name1, Score1, Name2, Score2)}}$ Quiz1 $\bowtie_{\text{Quiz1.Score > R1.Score}}$ R1
- R3 := $\Pi_{\text{Name1, Score1}}$ R2

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80

R1

Name	Score
Alice	70
Bob	90
Cathy	80

R2

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

R4

Name
Bob

- Find the students who score the highest in Quiz1
- R1 := Quiz1
- R2 := Quiz1 $\bowtie_{\text{Quiz1.Name <> R1.Name AND Quiz1.Score < R1.Score}}$ R1
- R3 := $\Pi_{\text{Quiz1.Name}}$ R2
- R4 := $\Pi_{\text{Quiz1.Name}}$ Quiz1 – R3

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Quiz2

Name	Score
Alice	80
Bob	90
Cathy	90
David	70

Quiz3

Name	Score
Alice	90
Bob	90
Cathy	80
David	70

- Query: “Find the students whose scores in Quizzes 1, 2, and 3 keep increasing”
- $(\text{Quiz1} \bowtie_{\text{Quiz1.Name} = \text{Quiz2.Name AND Quiz1.Score} < \text{Quiz2.Score}} \text{Quiz2}) \bowtie_{\text{Quiz2.Name} = \text{Quiz3.Name AND Quiz2.Score} < \text{Quiz3.Score}} \text{Quiz3}$

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Quiz2

Name	Score
Alice	80
Bob	90
Cathy	90
David	70

Quiz3

Name	Score
Alice	90
Bob	90
Cathy	80
David	70

- Query: “Find the students who score worse in Quiz 3 than either Quiz 1 or 2”
- $(\text{Quiz1} \bowtie_{\text{Quiz1.Name = Quiz3.Name AND Quiz1.Score > Quiz3.Score}} \text{Quiz3}) \cup (\text{Quiz2} \bowtie_{\text{Quiz2.Name = Quiz3.Name AND Quiz2.Score > Quiz3.Score}} \text{Quiz3})$
- Wrong!

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Quiz2

Name	Score
Alice	80
Bob	90
Cathy	90
David	70

Quiz3

Name	Score
Alice	90
Bob	90
Cathy	80
David	70

- Query: “Find the students who score worse in Quiz 3 than either Quiz 1 or 2”
- $\Pi_{\text{Quiz3.Name}}(\text{Quiz1} \bowtie_{\text{Quiz1.Name = Quiz3.Name AND Quiz1.Score > Quiz3.Score}} \text{Quiz3})$
  $\cup$
  $\Pi_{\text{Quiz3.Name}}(\text{Quiz2} \bowtie_{\text{Quiz2.Name = Quiz3.Name AND Quiz2.Score > Quiz3.Score}} \text{Quiz3})$

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

R1

Name	Score
Cathy	80

R2

Name1	Score1	Name2	Score2
Bob	90	Cathy	80
David	100	Cathy	80

R3

Name1	Score1
Bob	90
David	100

- Find the students who score higher than Cathy in Quiz1
- R1 := $\sigma_{\text{Name='Cathy'}}$ Quiz1
- $\rho_{\text{R2(Name1, Score1, Name2, Score2)}}$ Quiz1 $\bowtie_{\text{Quiz1.Score > R1.Score}}$ R1
- R3 := $\Pi_{\text{Name1, Score1}}$ R2

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80

R1

Name	Score
Alice	70
Bob	90
Cathy	80

R2

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

R4

Name
Bob

- Find the students who score the highest in Quiz1
- R1 := Quiz1
- R2 := Quiz1 $\bowtie_{\text{Quiz1.Name <> R1.Name AND Quiz1.Score < R1.Score}}$ R1
- R3 := $\Pi_{\text{Quiz1.Name}}$ R2
- R4 := $\Pi_{\text{Quiz1.Name}}$ Quiz1 – R3

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

R1

MaxScore
100

R2

Name	Score	MaxScore
David	100	100

R3

Name	Score
David	100

- Query: “Find the student that scores the second highest in Quiz1”
- R1 := $\gamma_{MAX(Score) \rightarrow MaxScore}(Quiz1)$
- R2 := Quiz1 $\bowtie_{Score = MaxScore}$ R1
- R3 := $\Pi_{Name, Score}(R2)$
- R4 := Quiz1 – R3
- R5 := $\gamma_{MAX(Score) \rightarrow 2ndMaxScore}(R4)$
- R6 := R4 $\bowtie_{Score = 2ndMaxScore}$ R5

Additional Exercise

Quiz1

Name	Score
Alice	70
Bob	90
Cathy	80
David	100

Students

Name	School
Alice	SCE
Bob	EEE
Cathy	EEE
David	SCE

- Query: “For each school, find the student that scores the highest in Quiz1”
- R1 := Quiz1 $\bowtie$ Student
- R2 := $\gamma_{\text{School, MAX(Score)} \rightarrow \text{MaxScore}}(\text{R1})$
- R3 := R1 $\bowtie_{\text{R1.School = R2.School AND Score = MaxScore}}$ R2
- R3 := $\Pi_{\text{Name, Score}}(\text{R3})$

Additional 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	SCE
DM	SCE
NN	EEE
SP	EEE

- Query: "Find the students who have taken all courses from SCE"
- R1 := $\sigma_{\text{School = 'SCE'}}$ CrsSch
- R2 := Grades $\bowtie$ R1
- R3 := $\gamma_{\text{Name, COUNT(Course)} \rightarrow \text{CrsCNT}}$ (R2)
- R4 := $\gamma_{\text{COUNT(Course)} \rightarrow \text{SceCNT}}$ (R1)
- R5 := R3 $\bowtie_{\text{CrsCNT = SceCNT}}$ R2
- R6 := Π_{Name} (R5)

Additional 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	SCE
DM	SCE
NN	EEE
SP	EEE

- Query: "For each school, find the students who have taken all courses in the school"
- R1 := Grades ⋈ CrsSch
- R2 := $\gamma_{\text{Name, School, COUNT(Course)}\rightarrow\text{CrsCNT}}$ (R1)
- R3 := $\gamma_{\text{School, COUNT(Course)}\rightarrow\text{CrsCNT}}$ (CrsSch)
- R4 := R2 $\bowtie_{\text{R2.School = R3.School AND R2.CrsCNT = R3.CrsCNT}}$ R3

Questions ??

Thank You !