Consider it very seriously

Database

(Relational Model and Algebra)

Spring, 2023

Jaewook Byun

Ph.D., Assistant Professor, Department of Software, Sejong University

jwbyun@sejong.ac.kr

https://sites.google.com/view/jack-dfpl/home

https://www.youtube.com/channel/UC988e-Y8nto0LXVae0aqaOQ

Schedule

Week	Title		
1	Course Overview and Environment Setting	Environment Setting and Revisiting Java	-
2	Introduction to Database		CH1
3	**Relational Model**	Relational Algebra	CH2
4	Relational Algebra		CH2
5	Database Language: SQL (DDL, DML, DQL, DCL)		CH3-5
6	Database Language: SQL (DDL, DML, DQL, DCL)		CH3-5
7	Database Language: SQL (DDL, DML, DQL, DCL)		CH3-5
8	Midterm Examination		
9	Physical Database Design: Indexing		CH12
10	Physical Database Design: Indexing		CH12
11	Conceptual Database Design – E-R Data Model		CH6
12	Logical Database Design 1 – Schema Mapping		CH6
13	Logical Database Design 2 – Normalization		CH7
14	Query Processing and Optimization, or View (TBD)		CH13-14, CH3-5
15	Final Examination		

Subject to change

Calendar

- 수업일수는 요일별 15주 이상이며 수업 결손이 발생하지 않도록 진행

요 일 별 수업일수	월	화	수	목	금
	16일	15일	16일	16일	15일

나. 수업주차 : 한 주차는 목요일부터 수요일까지 임

수업주차	기간	수업주차	기간
1주차	03.02.(목) ~ 03.08.(수)	9주차	04.27.(목) ~ 05.03.(수)
2주차	03.09.(목) ~ 03.15.(수)	10주차	05.04.(목) ~ 05.10.(수)
3주차	03.16.(목) ~ 03.22.(수)	11주차	05.11.(목) ~ 05.17.(수)
4주차	03.23.(목) ~ 03.29.(수)	12주차	05.18.(목) ~ 05.24.(수)
5주차	03.30.(목) ~ 04.05.(수)	13주차	05.25.(목) ~ 05.31.(수)
6주차	04.06.(목) ~ 04.12.(수)	14주차	06.01.(목) ~ 06.07.(수)
7주차	04.13.(목) ~ 04.19.(수)	15주차	06.08.(목) ~ 06.14.(수)
8주차 (중간고사)	04.20.(목) ~ 04.26.(수)	16주차 (기말고사)	06.15.(목) ~ 06.21.(수)

Calendar

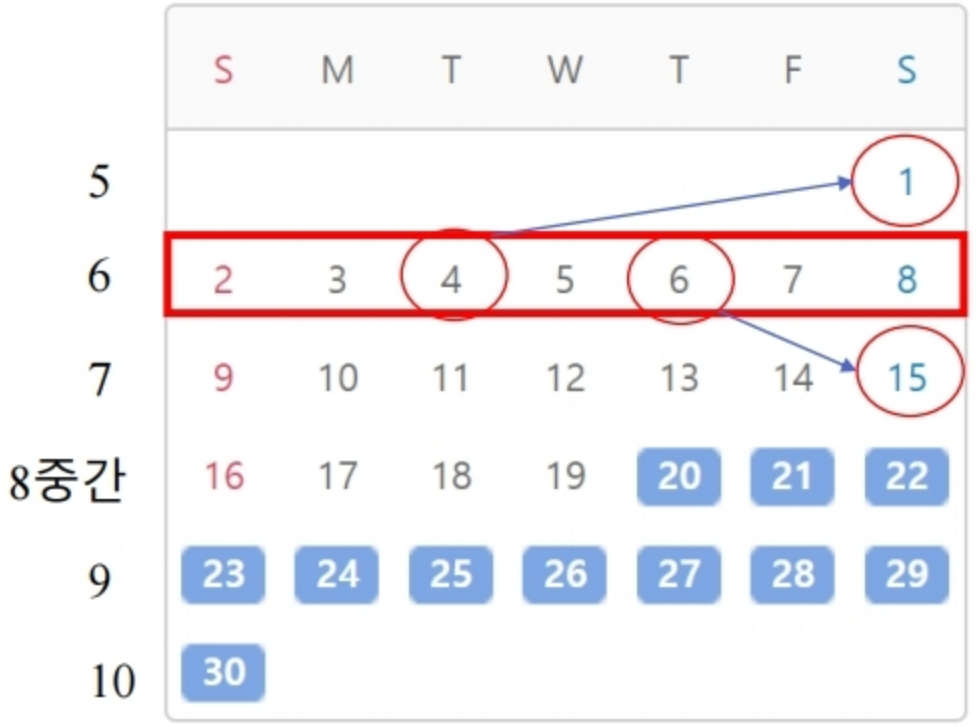
2023년 3월. March

	S	M	T	W	T	F	S
1				1	2	3	4
2	5	6	7	8	9	10	11
3	12	13	14	15	16	17	18
4	19	20	21	22	23	24	25
5	26	27	28	29	30	31	

- 2(목) 1학기 개강
- 3(금) - 8(수) 수강신청 과목 확인 및 변경
- 6(월) - 15(수) 교직신청
- 24(금) - 28(화) 수강신청과목 철회

Calendar

2023년 4월. April



	S	M	T	W	T	F	S
5							1
6	2	3	4	5	6	7	8
7	9	10	11	12	13	14	15
8중간	16	17	18	19	20	21	22
9	23	24	25	26	27	28	29
10	30						

- 20(목) - 26(수) 1학기 중간고사
- 27(목) - 5.1(월) 1학기 중간고사 성적 입력

IEEE ICDE 2023, Anaheim, California, US

Calendar

2023년 5월. May

	S	M	T	W	T	F	S
10		1	2	3	4	5	6
11	7	8	9	10	11	12	13
12	14	15	16	17	18	19	20
13	21	22	23	24	25	26	27
14	28	29	30	31			

- 2(화) - 7(일) 1학기 중간고사 성적 열람 및 정정
- 4(목) - 30(화) 복수·부전공, 연계융합전공 신청
- 5(금) 창립 83주년 기념휴일 (창립일 : 1940. 5. 20)
- 29(월) - 31(수) 하계 계절학기 수강신청

Calendar

2023년 6월. June

	S	M	T	W	T	F	S
14					1	2	3
	4	5	6	7	8	9	10
15기말	11	12	13	14	15	16	17
	18	19	20	21	22	23	24
	25	26	27	28	29	30	

- 9(금) - 26(월) 1학기 강의평가
- 15(목) - 21(수) 1학기 기말고사 및 수업결손 보충
- 22(목) - 26(월) 1학기 기말고사 성적 입력
- 22(목) 하계방학 시작 및 계절학기 개강
- 27(화) - 7.3(월) 1학기 기말고사 성적 열람 및 정정

Calendar

2023년 7월. July

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

- 4(화) - 5(수)
- 24(월) - 30(일)

1학기 기말고사 성적마감

2학기 복학, 휴학 신청

Table of Contents

- Relational Model
- Relational Algebra

Relational Model

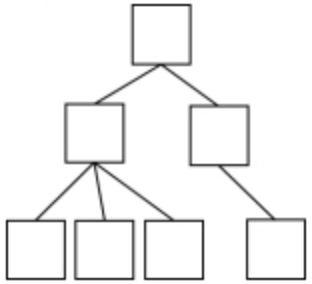
- Data Model
  - An abstract model that
    - organizes elements of data
    - defines a way of how they relate to one another and to the properties
  - Types

[Flat Model]

010-1234-5678	Database	6	98
010-1234-5678	Database programming	7	96
010-1234-5678	Database analysis	7	92
010-1234-5678	Advanced database	8	99

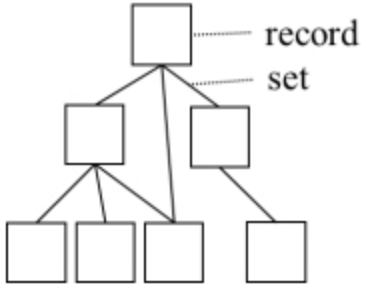
- A file of a uniform format
- No indexing
- No explicit relationship

[Tree Model]



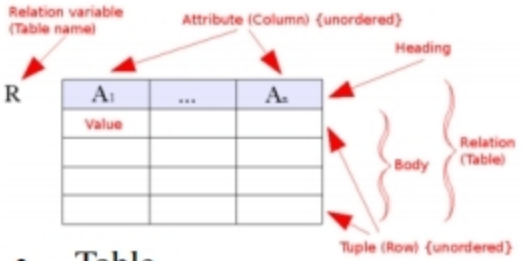
- Record and Set
- 1:1 1:n

[Network Model]



- Record and Set
- 1:1 1:n m:n

[Relational Model]

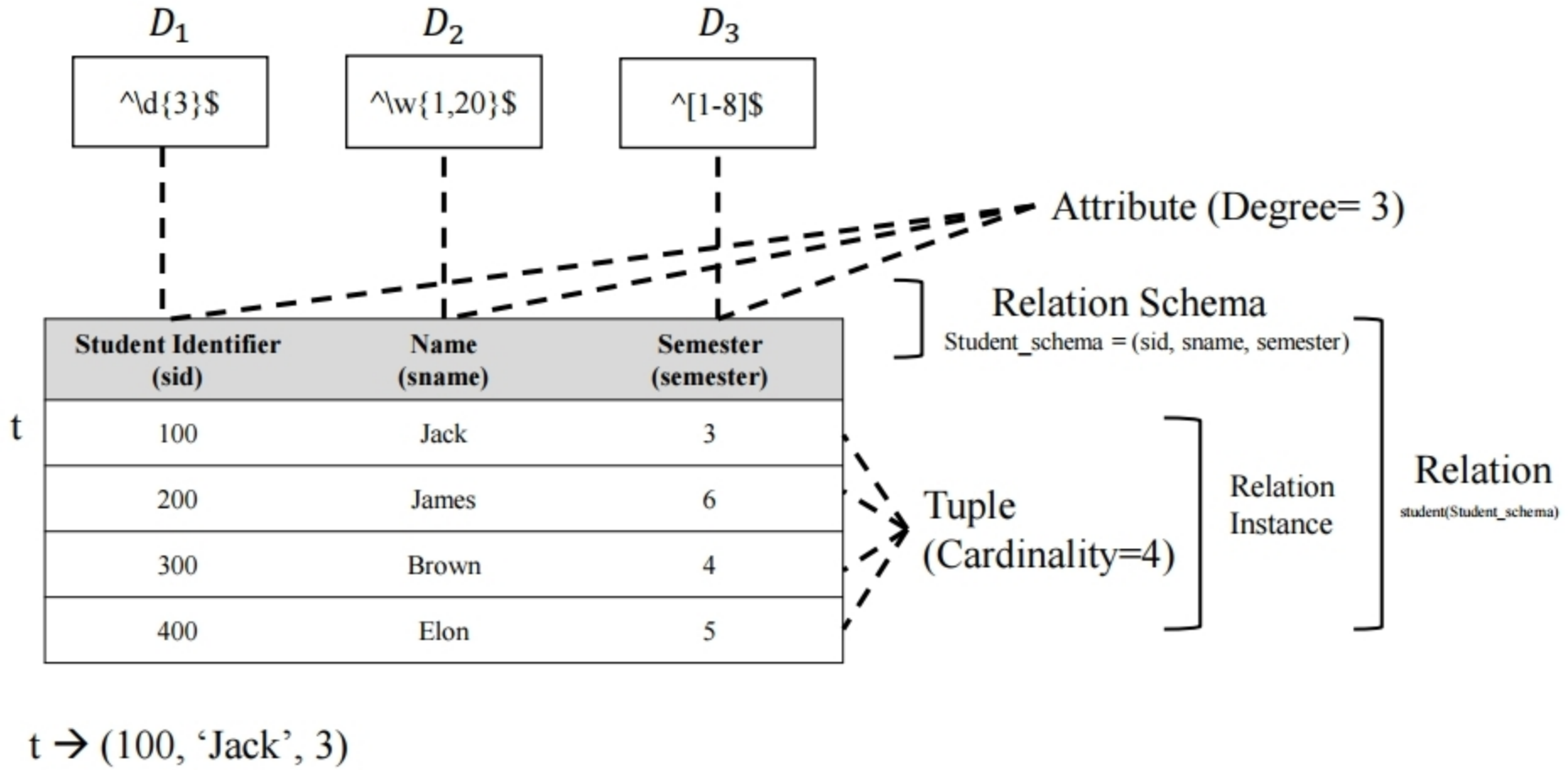


- Table
- Simple
- Data independence
- No insertion/deletion anomaly
- …

https://en.wikipedia.org/wiki/Data_model#Entity-relationship_model
https://www.geeksforgeeks.org/difference-between-hierarchical-network-and-relational-data-model/

Relational Model

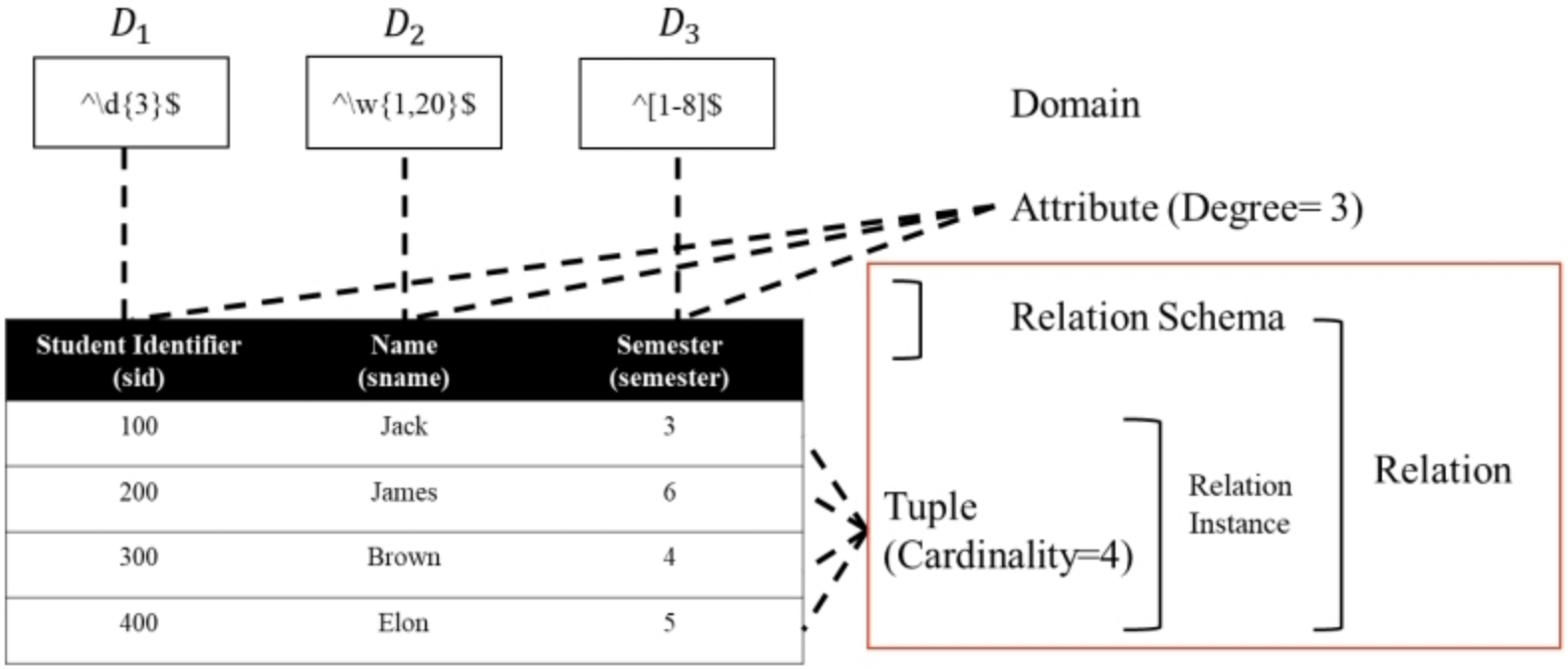
- Overview



t → (100, 'Jack', 3)

Relational Model

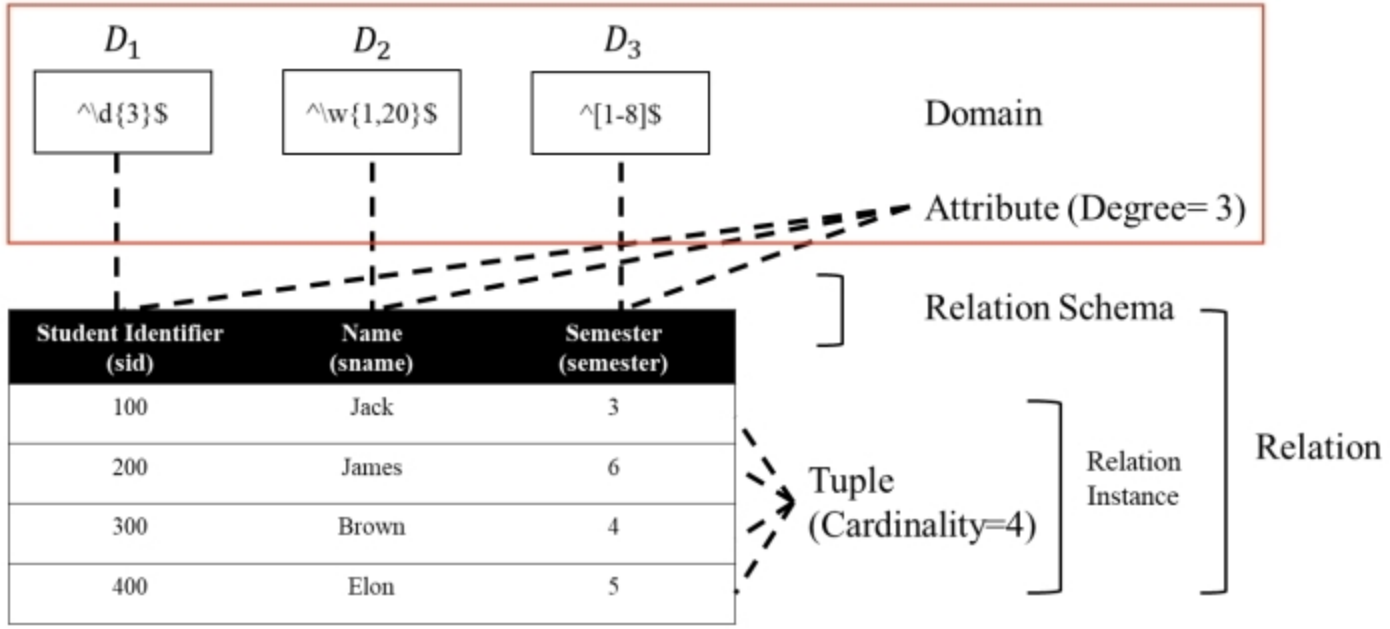
- Terms and Notations
  - Relation (a.k.a. Table) student(Student_schema)
    - Relation Schema (i.e., Relation intension)
      - The definition of the logical structure of relation
      - e.g., Student_schema = (sid, sname, semester)
    - Relation Instance (i.e., Relation extension)
      - A concrete occurrence of a relation



Relational Model

- Terms and Notations
  - Attribute
    - Relation schema consists of attributes
    - A name of column (e.g., student identifier, name, semester)
  - Degree
    - The number of attributes
  - Domain
    - A set of available values of an attribute
    - E.g., D_1: 3-digit numbers
    - Relation is a set of $D_1 \times D_2 \times \ldots \times D_n$ where n = degree

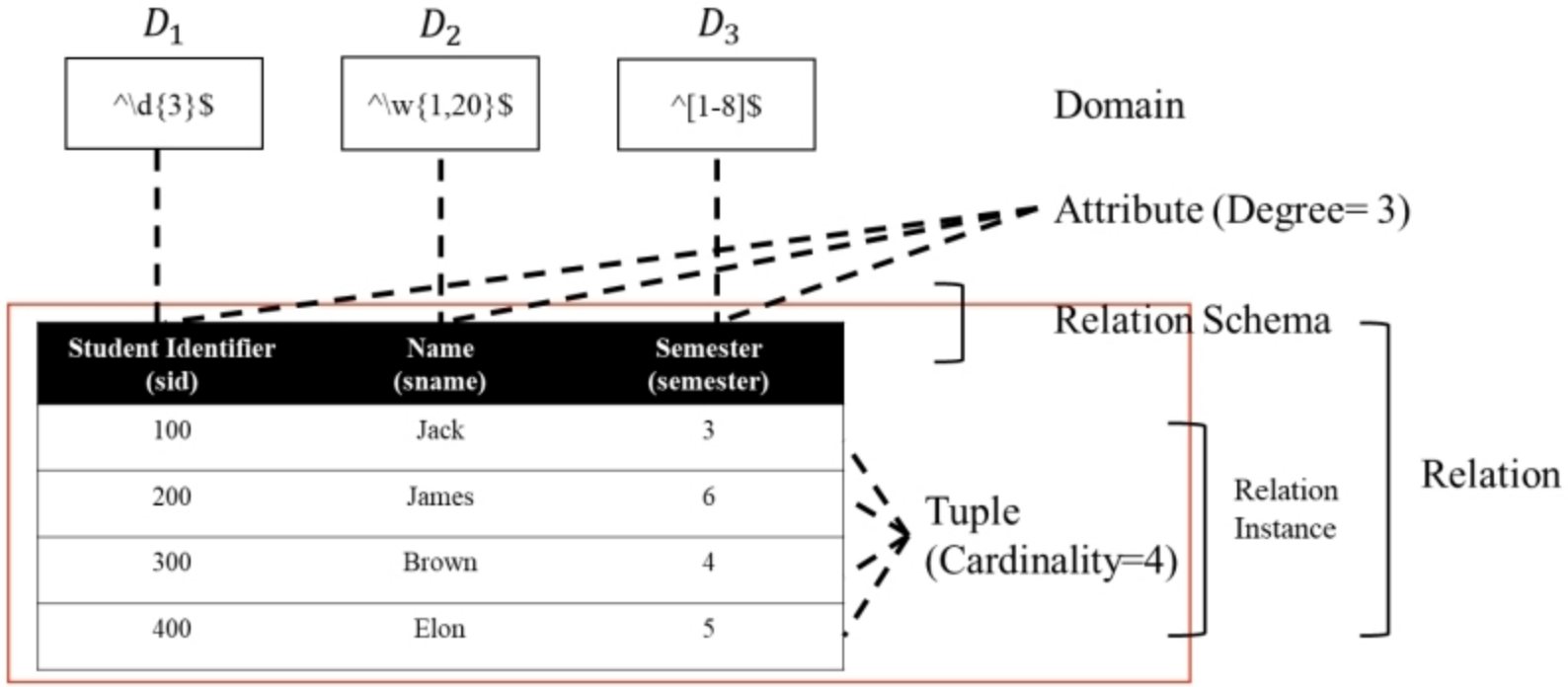
debuggex https://www.debuggex.com/



Relational Model

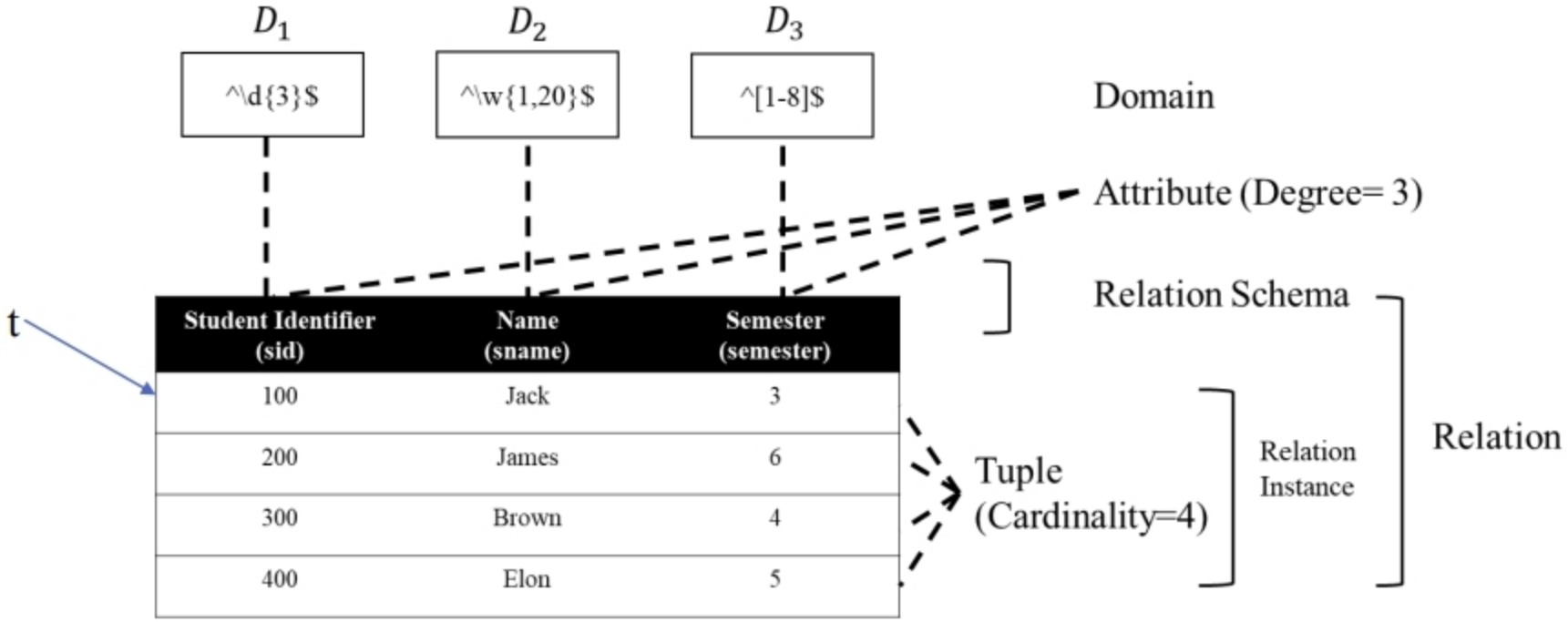
- Terms and Notations
  - Tuple
    - a.k.a. row
    - N-tuple has n values of the corresponding domain
    - E.g., 3-tuple (100, Jack, 3) where
      - $100 \in$ ^\d{3}$, $Jack \in$ ^\w{1,20}$, $3 \in$ ^[1 – 8]$
  - Cardinality: the number of tuples

Student_schema = (sid, sname, semester)
instantiates the following tuples
(100, “Jack”, 3), (200, “James”, 6),
(300, “Brown”, 4), (400, “Elon”, 5)



Relational Model

- Terms and Notations
  - Tuple variable
    - indicates a tuple in a relation
    - Assuming a tuple variable is *t* is a tuple (100, “Jack”, 3)
    - t[attribute] or t[index_attribute] indicates a value of tuple
    - e.g., t[sid] = 100, t[2] = “Jack”
    - $t \in r$, where r is a relation



Relational Model

- Characteristic of relational model
  1. A tuple is unique in a relation

ID	NAME	SEMESTER
100	Jack	3
300	Brown	4

  2. There is no order of each tuple

100	Jack	3
300	Brown	4

identical

300	Brown	4
100	Jack	3

  3. There is no order of each attribute

Student Identifier (sid)	Name (sname)	Semester (semester)
100	Jack	3

identical

Student Identifier (sid)	Semester (semester)	Name (sname)
100	3	Jack

  4. A domain is atomic

Student Identifier (sid)	Name (sname)	Semester (semester)
500	~~Gilldong, 길동~~	null

Non-atomic value is not allowable
Null value is allowed

Relational Model

- Key
  - A tuple is unique in a relation
  - Key is a set of attributes and a way to distinguish 'a' tuple from other tuples
  - $\{attr_1, attr_2, attr_3, \dots, attr_n\}$, degree = n

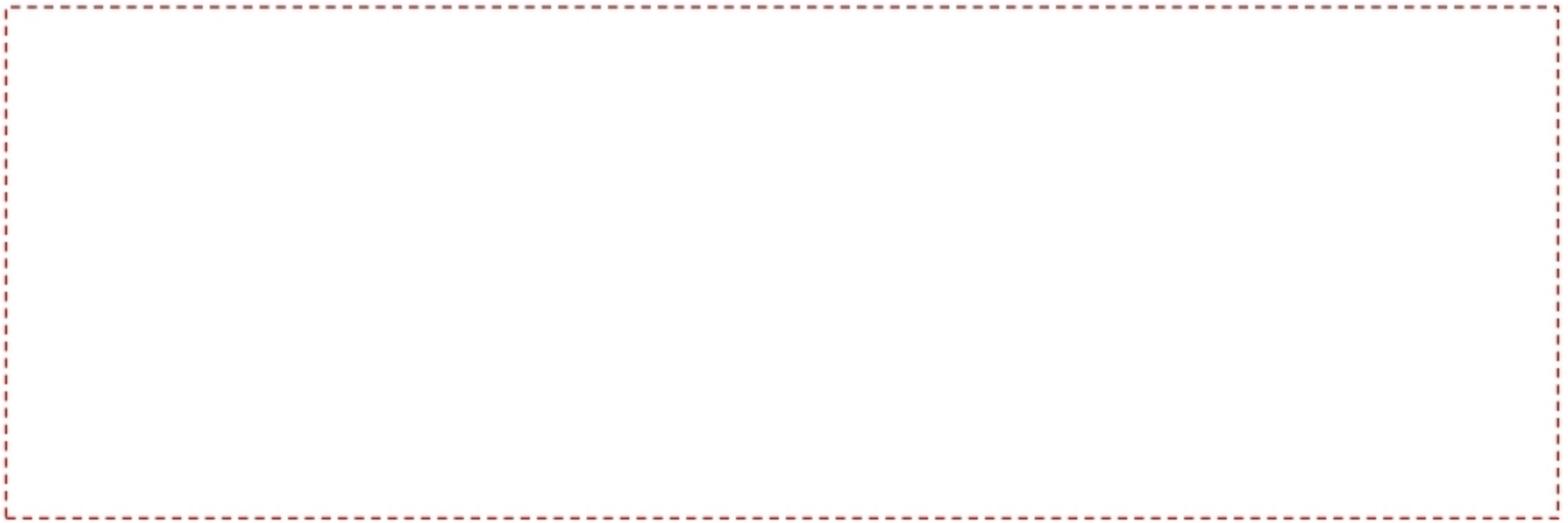
- A type of key
  - Super key
    - a key guarantees uniqueness **but could be non-minimal**
    - For each pair of $(t_1, t_2), t_1, t_2 \in r$,
      - If $t_1 \neq t_2 \rightarrow t_1[K] \neq t_2[K]$
        - K is a **super key** of R
  - Candidate key
    - a key guarantees uniqueness and be minimal
    - **a minimal super key**
  - Primary key
    - A candidate key selected by database manager
  - Alternate key
    - The other candidate keys, which could be an alternative for a primary key
  - Foreign key
    - A key to link two relations

Relational Model

- Example

1 2 3 4

Student_schema = (student_identifier, citizen_identifier, student_name, department)
with
(100, 031414, Jack, CS),
(200, 041412, Jack, SW),
(300, 055432, James, CS),
(400, 061412, Brown, IE)



Relational Model

Student_schema = (student_identifier, citizen_identifier, student_name, department, age, address)

- A type of key
  - Foreign key
    - A key to link two relations
    - A key of a relation R is a foreign key if the key is a primary key of other relation S
    - R: referencing relation
    - S: referenced relation
    - R and S can be same

Student_schema = (student_identifier, citizen_identifier, student_name, department)

Referencing relation

Person_schema = (citizen_identifier, age, address)

Referenced relation

Relational Model

- A type of key
  - Foreign key
    - A key to link two relations
    - A key of a relation R is a foreign key if the key is a primary key of other relation S
    - R: referencing relation
    - S: referenced relation
    - R and S can be same

Referenced and Referencing relation

Parent_child_schema = (my_identifier, parent_identifier)

Relational Model

- Integrity constraints
  - Entity integrity constraints
    - A primary key cannot contain null value
    - A value of a primary key should not be null
      - Null: a special domain value which is unknown or non-applicable
  - Referential integrity constraints
    - A value of a foreign key should be either null or a value existing in a primary key of a referenced relation

(100, **031414**, Jack, CS),
(200, **041412**, Jack, SW),
(300, **055432**, James, CS),
(400, **null**, Brown, IE)

(031414, 30, Seoul),
(041412, 32, Incheon),
(055432, 34, Jeju),

Query Language

- Query Language
  - **Relational algebra**
    - Procedural query language
    - Defines how and what
    - Input: relation → operator → Output: relation
  - Relational calculus
    - Declarative query language
    - Defines what
    - E.g., $\{S \mid F(S)\}, \{S \mid Staff(S) \wedge S.salary > 10000\}$
  - The power of relational algebra and relational calculus are equivalent

- Operators of relational algebra
  - Select, project, union, set difference, cartesian product, rename, set intersection, natural join, division, assignment, etc.

Relational Algebra

- Relational algebra
  - Procedural query language
  - Defines how and what
  - Input: relation → operator → Output: relation

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

$\sigma_{branch\_name="Perryridge"}(loan)$

$\sigma_{branch\_name="Perryridge"}(loan)$

loan_number	branch_name	Amount
L-15	Perryridge	1500
L-16	Perryridge	1300

- **Operators** of relational algebra
  - Select, project, union, set difference, cartesian product, rename, set intersection, natural join, division, assignment, etc.

Relational Algebra

- Operators of relational algebra
  - SELECT, σ (\sigma)
    - Syntax
      - $\sigma_{predicate}(relation)$
    - Select tuples that satisfy a given predicate using
      - $=, \neq, <, \leq, >, \geq$ and
      - $AND\ (\wedge), OR\ (\vee), NOT(\neg)$

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

$\sigma_{branch\_name="Perryridge"}(loan)$

$\sigma_{branch\_name="Perryridge"}(loan)$

loan_number	branch_name	Amount
L-15	Perryridge	1500
L-16	Perryridge	1300

Relational Algebra

- Operators of relational algebra
  - SELECT, σ (\sigma)
    - Practice
      - $\sigma_{amount>1200}(loan)$
      - $\sigma_{branch\_name="Perryridge" \wedge amount>1200}(loan)$

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

Relational Algebra

- Operators of relational algebra
  - PROJECT, π (\pi)
    - Syntax
      - $\pi_{attr1,attr2\ldots}(relation)$
    - Projects the list of attributes of interest to be appeared in the result
    - **Note:** a relation is a **set** of tuples (no redundancy)

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

$\pi_{branch\_name}(loan)$

$\pi_{branch\_name}(loan)$

branch_name
Round Hill
Downtown
Perryridge
Redwood
Mianus

Relational Algebra

- Operators of relational algebra
  - PROJECT, π (\pi)
    - Practice
      - $\pi_{branch\_name,amount}(loan)$
      - The relational algebra projects loan_number attribute of loan relation

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

Relational Algebra

- Operators of relational algebra
  - Nested relational algebra
    - Output of a relational algebra is a relation could be nested with ( )
    - E.g., $\pi_{attr1,attr2\ldots}\left(\sigma_{predicate}(relation)\right)$
    - $\sigma_{predicate}(\pi_{attr1,attr2\ldots}(relation))$

- Print loan_number where amount = 1000
  - $\pi_{loan\_number}\left(\sigma_{amount=1000}(loan)\right)$
  - ~~$\sigma_{amount=1000}(\pi_{loan\_number}(loan))$~~

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

$\pi_{branch\_name}(loan)$

branch_name
Round Hill
Downtown
Perryridge
Redwood
Mianus

$\sigma_{branch\_name="Mianus"}(loan)$

$\sigma_{branch\_name="Mianus"}(\pi_{branch\_name}(loan))$

branch_name
Mianus

Relational Algebra

- Operators of relational algebra
  - UNION, ∪ (\cup)
    - Syntax
      - $relation_1 \cup relation_2$
    - Returns all the tuples that are either in two relations
    - Two relations to be merged must have
      - the identical number of attributes (degree)
      - domain of each attribute on two relations is identical
    - **Note:** the result is also a relation of a set of tuples (no redundancy)
      - tuples in $relation_2$ are added to $relation_1$
  - Practice

$$\pi_{customer\_name}(borrower) \cup \pi_{customer\_name}(depositor)$$

$$\pi_{branch\_name}(\sigma_{balance>700}(account)) \cup \pi_{customer\_name}(\sigma_{customer\_name='Hayes'}(depositor))$$

borrower relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

$\pi_{customer\_name}(borrower) \cup \pi_{customer\_name}(depositor)$

?

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Relational Algebra

- Operators of relational algebra
  - INTERSECTION, ∩ (\cap)
    - Syntax
      - $relation_1 \cap relation_2$
    - Returns all the tuples that are in both two relations
    - Two relations to be merged must have
      - the identical number of attributes (degree)
      - domain of each attribute on two relations is identical
    - **Note:** the result is also a relation of a set of tuples (no redundancy)
      - tuples in $relation_2$ are added to $relation_1$
  - Practice

borrower1 relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14

borrower2 relation

customer_name	loan_number
Adams	L-16
Curry	L-37
Hayes	L-25
Jackson	L-14

borrower relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

$\pi_{customer\_name}(borrower) \cap \pi_{customer\_name}(depositor)$

?

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Relational Algebra

- Operators of relational algebra
  - DIFFERENCE, $-$
    - Syntax
      - $relation_1 - relation_2$
    - Returns all the tuples that are in a relation 1 but not in a relation 2
    - Two relations to be merged must have
      - the identical number of attributes (degree)
      - domain of each attribute on two relations is identical
    - **Note:** the result is also a relation of a set of tuples (no redundancy)
      - tuples in $relation_2$ are added to $relation_1$
  - Practice

borrower relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

$\pi_{customer\_name}(borrower) - \pi_{customer\_name}(depositor)$

?

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Syntax
      - $relation_1 \times relation_2$
      - $(relation_1.attr_1, relation_1.attr_2, \dots) \times (relation_2.attr_1, relation_2.attr_2, \dots)$
    - Returns a cross product of two relations
      - A cross product: combination of each tuple in a relation 1 with each tuple in a relation 2
      - $(relation_1.attr_1, relation_1.attr_2, \dots, relation_2.attr_1, relation_2.attr_2, \dots)$
        - Degree = degree of a relation 1 + degree of a relation 2
        - Cardinality = cardinality of a relation 1 * cardinality of a relation 2

Note: you can explicitly state an attribute in a specific relation

borrower relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

$borrower \times loan$

?

Degree?
Cardinality?

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Practice

$borrower \times loan$ relation

customer_name	borrower.loan_number	loan.loan_number	branch_name	Amount
Adams	L-16	L-11	Round Hill	900
Adams	L-16	L-14	Downtown	1500
Adams	L-16	L-15	Perryridge	1500
Adams	L-16	L-16	Perryridge	1300
Adams	L-16	L-17	Downtown	1000
Adams	L-16	L-23	Redwood	2000
Adams	L-16	L-93	Mianus	500
Curry	L-93	L-11	Round Hill	900
Curry	L-93	L-14	Downtown	1500
Curry	L-93	L-15	Perryridge	1500
Curry	L-93	L-16	Perryridge	1300
Curry	L-93	L-17	Downtown	1000
Curry	L-93	L-23	Redwood	2000
Curry	L-93	L-93	Mianus	500
...	...	...	...	....

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Practice
      - Goal: Finds all the names of customers who have a loan in 'Perryridge' branch

borrower relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Practice
      - Goal: Finds all the names of customers who have a loan in 'Perryridge' branch

1. Finds all the cross product information in both relations where the branch name is Perryridge

$$\sigma_{branch\_name='Perryridge'}(borrower \times loan)$$

customer_name	borrower.loan_number	loan.loan_number	branch_name	Amount
Adams	L-16	L-15	Perryridge	1500
Adams	L-16	L-16	Perryridge	1300
Curry	L-93	L-15	Perryridge	1500
Curry	L-93	L-16	Perryridge	1300
Hayes	L-15	L-15	Perryridge	1500
Hayes	L-15	L-16	Perryridge	1300
Jackson	L-14	L-15	Perryridge	1500
Jackson	L-14	L-16	Perryridge	1300
Jones	L-17	L-15	Perryridge	1500
Jones	L-17	L-16	Perryridge	1300
Smith	L-11	L-15	Perryridge	1500
Smith	L-11	L-16	Perryridge	1300
Smith	L-23	L-15	Perryridge	1500
Smith	L-23	L-16	Perryridge	1300
Williams	L-17	L-15	Perryridge	1500
Williams	L-17	L-16	Perryridge	1300

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Practice
      - Goal: Finds all the names of customers who have a loan in 'Perryridge' branch

2. Then, select tuples if two loan numbers are matched (valid)

$$\sigma_{borrower.loan\_number=loan.loan\_number}(\sigma_{branch\_name="Perryridge"}(borrower \times loan))$$

customer_name	borrower.loan_number	loan.loan_number	branch_name	Amount
Adams	L-16	L-16	Perryridge	1300
Hayes	L-15	L-15	Perryridge	1500

3. Project customer name(s) = goal

$$\pi_{customer\_name}(\sigma_{borrower.loan\_number=loan.loan\_number}(\sigma_{branch\_name="Perryridge"}(borrower \times loan)))$$

customer_name
Adams
Hayes

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Practice
      - Goal: Finds all the names of customers who have a loan in 'Perryridge' branch

1. Finds all the cross product information in both relations where the loan_number is same

Another way

$$\sigma_{borrower.loan\_number=loan.loan\_number}(borrower \times loan)$$

customer_name	borrower.loan_number	loan.loan_number	branch_name	Amount
Adams	L-16	L-16	Perryridge	1300
Curry	L-93	L-93	Mianus	500
Hayes	L-15	L-15	Perryridge	1500
Jackson	L-14	L-14	Downtown	1500
Jones	L-17	L-17	Downtown	1000
Smith	L-11	L-11	Round Hill	900
Smith	L-23	L-23	Redwood	2000
Williams	L-17	L-17	Downtown	1000

Relational Algebra

- Operators of relational algebra
  - CARTESIAN PRODUCT, × (\times)
    - Practice
      - Goal: Finds all the names of customers who have a loan in 'Perryridge' branch

2. Then, select tuples if the branch_name is 'Perryridge'

$$\sigma_{\text{branch\_}name='Perryridge'}(\sigma_{borrower.loan\_number=loan.loan\_number}(borrower \times loan))$$

customer_name	borrower.loan_number	loan.loan_number	branch_name	Amount
Adams	L-16	L-16	Perryridge	1300
Hayes	L-15	L-15	Perryridge	1500

3. Project customer name(s) = goal

$$\pi_{customer\_name}(\sigma_{\text{branch\_}name='Perryridge'}(\sigma_{borrower.loan\_number=loan.loan\_number}(borrower \times loan)))$$

customer_name
Adams
Hayes

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
    - Syntax
      - $\rho_{renamed\_relation}(original\_relation)$
    - Rename the name of original relation to renamed_relation
  - Practice 1
    - Goal: Find the maximum balance in account relation

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
    - Syntax
      - $\rho_{renamed\_relation}(original\_relation)$
    - Rename the name of original relation to renamed_relation
  - Practice 1
    - Goal: Find the maximum balance in account relation
    - Step1 : List all the cross product of account relation

$\rho_{a_1}(account)$

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

$\rho_{a_1}(account) \times \rho_{a_2}(account)$

?

$\rho_{a_2}(account)$

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
  - Practice 1
    - Step1: List all the cross product of account relation

$$\rho_{a_1}(account) \times \rho_{a_2}(account)$$

a1.account_number	a1.branch_name	a1.balance	a2.account_number	a2.branch_name	a2.balance
A-101	Downtown	500	A-101	Downtown	500
A-101	Downtown	500	A-102	Perryridge	400
A-101	Downtown	500	A-201	Brighton	900
A-101	Downtown	500	A-215	Mianus	700
A-101	Downtown	500	A-217	Brighton	750
A-101	Downtown	500	A-222	Redwood	700
A-101	Downtown	500	A-305	Round Hill	350
A-102	Perryridge	400	A-101	Downtown	500
A-102	Perryridge	400	A-102	Perryridge	400
A-102	Perryridge	400	A-201	Brighton	900
A-102	Perryridge	400	A-215	Mianus	700
A-102	Perryridge	400	A-217	Brighton	750
A-102	Perryridge	400	A-222	Redwood	700
A-102	Perryridge	400	A-305	Round Hill	350
...	...	...	...	...	...

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
  - Practice 1
    - Step 2: select a1's balances if a1's balance is less than a2's balance

$$\pi_{a1.balance}(\sigma_{a1.balance<a2.balance}(\rho_{a_1}(account) \times \rho_{a_2}(account)))$$

a1.balance
500
400
700
750
350

Note: it contains all the balances in account relation except the maximum value

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
  - Practice 1
    - Step 3: remove the non-maximum balance in account relation = goal

$\pi_{balance}(account)$

balance
500
400
900
700
750
700
350

−

$\pi_{a1.balance}(\sigma_{a1.balance<a2.balance}(\rho_{a_1}(account) \times \rho_{a_2}(account)))$

a1.balance
500
400
700
750
350

=

balance
350

balance
900

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
  - Practice 2
    - Goal: find the names of customers who lives in a street and a city that Smith lives

customer relation

customer	customer_street	customer_city
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Hayes	Main	Harrison
Johbnson	Alma	Palo Alto
Jones	MAIN	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Puthnam	Stamford
Williams	Nassau	Princeton

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
  - Practice 2
    - Goal: find the names of customers who lives in a street and a city that Smith lives
    - Step 1: Retrieves a street and a city that Smith lives as smith_address

$$\rho_{smith\_address}(\pi_{customer\_street,customer\_city}(\sigma_{customer\_name="Smith"}(customer)))$$

customer_street	customer_city
North	Rye

    - Step 2: List up all the cross product between customer and smith_address

$$customer \times \rho_{smith\_address}(\pi_{customer\_street,customer\_city}(\sigma_{customer\_name="Smith"}(customer)))$$

customer	customer_street	customer_city	smith_address.customer_street	smith_address.customer_city
Adams	Spring	Pittsfield	North	Rye
Brooks	Senator	Brooklyn	North	Rye
Curry	**North**	**Rye**	**North**	**Rye**
Glenn	Sand Hill	Woodside	North	Rye
Hayes	Main	Harrison	North	Rye
Johbnson	Alma	Palo Alto	North	Rye
Jones	MAIN	Harrison	North	Rye
Lindsay	Park	Pittsfield	North	Rye
Smith	**North**	**Rye**	**North**	**Rye**
Turner	Puthnam	Stamford	North	Rye
Williams	Nassau	Princeton	North	Rye

Relational Algebra

- Operators of relational algebra
  - RENAME, ρ (\rho)
  - Practice 2
    - Goal: find the names of customers who lives in a street and a city that Smith lives
    - Step 3: select tuples where a customer lives in a street and a city of Smith

$$\sigma_{customer.customer\_street=smith\_address.customer\_street \wedge customer.customer\_city=smith\_address.customer\_city}\Big(customer \times \rho_{smith\_address}\left(\pi_{customer\_street,customer\_city}\left(\sigma_{customer\_name="Smith"}(customer)\right)\right)\Big)$$

customer	customer.customer_street	customer.customer_city	smith_address.customer_street	smith_address.customer_city
Curry	North	Rye	North	Rye
Smith	North	Rye	North	Rye

    - Step 4: project the name of the customer

$$\pi_{customer\_name}\Big(\sigma_{customer.customer\_street=smith\_address.customer\_street \wedge customer.customer\_city=smith\_address.customer\_city}\Big(customer \times \rho_{smith\_address}\left(\pi_{customer\_street,customer\_city}\left(\sigma_{customer\_name="Smith"}(customer)\right)\right)\Big)\Big)$$

customer
Curry
Smith

Relational Algebra

- Operators of relational algebra
  - Motivation
    - Merge borrower and loan relations where its loan_number is matched

borrower relation

customer_name	loan_number
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

$$\pi_{customer\_name,loan.loan\_number,branch\_name,amount}(\sigma_{borrower.loan\_number=loan.loan\_number}(borrower \times loan))$$

?

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

customer_name	loan_number	branch_name	amount
Adams	L-16	Perryridge	1300
Curry	L-93	Mianus	500
Hayes	L-15	Perryridge	1500
Jackson	L-14	Downtown	1500
Jones	L-17	Downtown	1000
Smith	L-11	Round Hill	900
Smith	L-23	Redwood	2000
Williams	L-17	Downtown	1000

Relational Algebra

- Operators of relational algebra
  - JOIN, ⋈ (\bowtie)
    - Syntax
      - $relation_1 \bowtie relation_2$
    - Merge two relations based on their common attributes
    - Degree = degree of relation 1 + 2 – the number of common attributes

    - $r \bowtie s = \pi_{R \cup S}\left(\sigma_{r.A_1=s.A_1 \wedge r.A_2=s.A_2 \wedge \cdots \wedge r.A_n=s.A_n}(r \times s)\right)$
      - $R \cap S = \{A_1, A_2, \ldots, A_n\}$
        - R = schema of r, S = schema of s
        - $R \cup S$: the union of attributes on both relation schema
        - $R \cap S$: the intersection of attributes on both relation schema
        - $R - S, S - R$: the difference of attributes from a relation to another relation

    - Resolving motivation
      - $\pi_{customer\_name,loan.loan\_number,branch\_name,amount}(\sigma_{borrower.loan\_number=loan.loan\_number}(borrower \times loan))$
      - → $\boldsymbol{borrower \bowtie loan}$

Relational Algebra

$customer \bowtie account$

$=$

- Operators of relational algebra
  - JOIN, $\bowtie$ (\bowtie)
    - JOIN satisfies Associative rule
      - $(customer \bowtie depositor) \bowtie account$
      - $= customer \bowtie (depositor \bowtie account)$
    - Example
      - Customer_schema = (customer_name, customer_street, customer_city)
      - Depositor_schema = (customer_name, account_number)
      - Account_schema = (account_number, branch_name, balance)
      - Goal: Find all the branch_name of accounts of customers who live in 'Harrison' city

customer relation

customer_name	customer_street	customer_city
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Hayes	Main	Harrison
Johbnson	Alma	Palo Alto
Jones	MAIN	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Puthnam	Stamford
Williams	Nassau	Princeton

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Relational Algebra

- Operators of relational algebra
  - JOIN, ⋈ (\bowtie)
    - JOIN satisfies Associative rule
      - $(customer \bowtie depositor) \bowtie account$
      - $= customer \bowtie (depositor \bowtie account)$
    - Example
      - Goal: Find all the branch_name of customers who live in 'Harrison'
        - $\pi_{branch\_name}(\sigma_{customer\_city="harrison"}(customer \bowtie depositor \bowtie account))$

check below

$$\pi_{branch\_name}\left(\left(\sigma_{customer\_city='Harrison'}(customer)\right) \bowtie depositor \bowtie account\right)$$

customer	customer_street	customer_city	account_number	branch_name	balance
~~Adams~~	~~Spring~~	~~Pittsfield~~			Note:
~~Brooks~~	~~Senator~~	~~Brooklyn~~			just for your reference, not included
~~Curry~~	~~North~~	~~Rye~~			
~~Glenn~~	~~Sand Hill~~	~~Woodside~~			
Hayes	**Main**	**Harrison**	**A-102**	**Perryridge**	**400**
Johnson	Alma	Palo Alto	A-101	Downtown	500
Johnson	Alma	Palo Alto	A-201	Brighton	900
Jones	**MAIN**	**Harrison**	**A-217**	**Brighton**	**750**
Lindsay	Park	Pittsfield	A-222	Redwood	700
Smith	North	Rye	A-215	Mianus	700
Turner	Puthnam	Stamford	A-305	Round Hill	350
~~Williams~~	~~Nassau~~	~~Princeton~~		Note: just for your reference, not included	

Relational Algebra

- Operators of relational algebra
  - JOIN, ⋈ (\bowtie)
    - $r \bowtie s = r \times s, where\ R \cap S = \emptyset$
    - Example
      - The cardinality of $customer \bowtie account$

l relation

a	b
1	2

r relation

c	d
3	4
5	6

a relation = degree 3 cardinality 5
b relation = degree 4 cardinality 7

$$A \cap B = \emptyset$$

$$a \bowtie b = degree\ ?\ cardinality\ ?$$

customer relation

customer_name	customer_street	customer_city
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Hayes	Main	Harrison
Johbnson	Alma	Palo Alto
Jones	MAIN	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Putnam	Stamford
Williams	Nassau	Princeton

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Relational Algebra

- Operators of relational algebra
  - Theta Join, $\bowtie_\theta$, \bowtie_\theta
    - Syntax
      - $relation_1 \bowtie_\theta relation_2 = \sigma_\theta(relation_1 \times relation_2)$
    - Merge two relations based on the predicate (theta)
    - Degree = degree of relation 1 + 2
    - Example
      - $\pi_{customer,customer\_street,customer\_city}(customer \bowtie_{customer.customer=depositor.customer\_name} depositor)$
      - Now, we can join relations with semantically identical but differently named attributes

customer relation

customer	customer_street	customer_city
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Hayes	Main	Harrison
Johbnson	Alma	Palo Alto
Jones	MAIN	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Puthnam	Stamford
Williams	Nassau	Princeton

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - Syntax
      - $relation_1 \div relation_2$
    - Select tuples of relation 1 associated with all the tuples of relation 2
    - Degree = |degree of $Relation_1$ − degree of $Relation_2$|

  - Example

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	**Brighton**
Jones	**Brighton**
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton

=

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - Syntax
      - $relation_1 \div relation_2$
    - Select tuples of relation 1 associated with **all** the tuples of relation 2
    - Degree = |degree of $Relation_1$ − degree of $Relation_2$|

  - Example

customer_name	branch_name
Hayes	Perryridge
Johnson	**Downtown**
Johnson	**Brighton**
Jones	**Brighton**
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton
Downtown

=

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - $r \div s$
    - $= \pi_{R-S}(r) - \pi_{R-S}\left((\pi_{R-S}(r) \times s) - \pi_{R-S,S}(r)\right)$
      - $where\ S \subseteq R$

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton
Downtown

customer_name
Hayes
Johnson
Jones
Lindsay
Smith
Turner

customer_name	branch_name
Hayes	Brighton
Hayes	Downtown
Johnson	Brighton
Johnson	Downtown
Jones	Brighton
Jones	Downtown
Lindsay	Brighton
Lindsay	Downtown
Smith	Brighton
Smith	Downtown
Turner	Brighton
Turner	Downtown

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - $r \div s$
    - $= \pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - \pi_{R-S,S}(r))$

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton
Downtown

customer_name
Hayes
Johnson
Jones
Lindsay
Smith
Turner

customer_name	branch_name
Hayes	Brighton
Hayes	Downtown
Johnson	Brighton
Johnson	Downtown
Jones	Brighton
Jones	Downtown
Lindsay	Brighton
Lindsay	Downtown
Smith	Brighton
Smith	Downtown
Turner	Brighton
Turner	Downtown

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - $r \div s$
    - $= \pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - \pi_{R-S,S}(r))$

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton
Downtown

customer_name
Hayes
Johnson
Jones
Lindsay
Smith
Turner

customer_name	branch_name
Hayes	Brighton
Hayes	Downtown
Jones	Downtown
Lindsay	Brighton
Lindsay	Downtown
Smith	Brighton
Smith	Downtown
Turner	Brighton
Turner	Downtown

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - $r \div s$
    - $= \pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - \pi_{R-S,S}(r))$

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton
Downtown

customer_name
Hayes
Johnson
Jones
Lindsay
Smith
Turner

customer_name
Hayes
Jones
Lindsay
Smith
Turner

Relational Algebra

- Operators of relational algebra
  - Division, ÷, \div
    - $r \div s$
    - $= \pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - \pi_{R-S,S}(r))$

customer_name	branch_name
Hayes	Perryridge
Johnson	Downtown
Johnson	Brighton
Jones	Brighton
Lindsay	Redwood
Smith	Mianus
Turner	Round Hill

÷

branch_name
Brighton
Downtown

customer_name
Johnson

Relational Algebra

- Operators of relational algebra
  - Assignment, ←, \leftarrow
    - Syntax
      - $r_2 \leftarrow r_1$
    - Like assignment to a variable
    - Simplifies the complex relational algebra

  - Example

$$\pi_{R-S}(r) - \pi_{R-S}((\pi_{R-S}(r) \times s) - \pi_{R-S,S}(r))$$

    - $temp1 \leftarrow \pi_{R-S}(r)$
    - $temp2 \leftarrow \pi_{R-S}\left((temp1 \times s) - \pi_{R-S,S}(r)\right)$
    - $result = temp1 - temp2$

Relational Algebra

- Operators of relational algebra
  - Generalized-projection
    - Syntax
      - $\pi_{F_1,F_2,\ldots,F_n}(E)$
    - Extension of the projection with the allowance of arithmetic expression (F) and rename

id	point
a	100
b	90
c	90

  - Example

credit_info relation

customer_name	limit	credit_balance
Curry	2000	1750
Hayes	1500	1500
Jones	6000	700
Smith	2000	400

id	attendance	midterm	final	assignment
a	10	30	30	30
b	8	27	27	28
c	6	25	29	30

$$\pi_{id,(attendance+midterm+\ final+\ assignment)AS\ point}(grade)$$

$$\pi_{customer\_name,(limit-credit\_balance)\ \mathbf{as}\ credit\_available}\ (credit\_info)$$

customer_name	(limit-credit_balance)
Curry	250
Jones	5300
Smith	1600
Hayes	0

customer_name	credit_available
Curry	250
Jones	5300
Smith	1600
Hayes	0

Relational Algebra

- Operators of relational algebra
  - Aggregation function, $\mathcal{G}$, \scriptG
    - Syntax
      - $\mathcal{G}_{F_1(A_1),F_2(A_2),\ldots,F_n(A_n)}(r)$
      - F: aggregation function (e.g., avg, count, count-distinct, min, max)
      - A: attributes
    - Takes a set of attribute values and return a single value as a result

$$\mathcal{G}_{avg(salary)}(pt\_works)$$
$$\mathcal{G}_{count(employee\_name)}(pt\_works)$$
$$\mathcal{G}_{\boldsymbol{count-distinct}(salary)}(pt\_works)$$

- Example

pt_works relation

employee_name	branch_name	salary
Adams	Perryridge	1500
Brown	Perryridge	1300
Gopal	Perryridge	5300
Johnson	Downtown	1500
Loreena	Downtown	1300
Peterson	Downtown	2500
Rao	Austin	1500
Sato	Austin	1600

$\mathcal{G}_{count-distinct(branch\_name)}$(pt_works)

$count-distinct(branch\_name)$
3

$\mathcal{G}_{sum(salary)}$(pt_works)

$sum(salary)$
16500

Sum of salary for each branch?

Relational Algebra

- Operators of relational algebra
  - Aggregation function, $\mathcal{G}$, \scriptG
    - Syntax
      - $_{G_1,G_2,\ldots,G_m}\mathcal{G}_{F_1(A_1),F_2(A_2),\ldots,F_n(A_n)}(r)$
      - G: a list of attributes to be grouped
      - F: aggregation function (e.g., avg, count, count-distinct, min, max)
      - A: attribute name
    - Degree: m+n
    - Takes a set of attribute values and return a single value as a result
  - Example

A maximum salary for each branch

pt_works relation grouped by branch_name

employee_name	branch_name	salary
Adams	Perryridge	1500
Brown	Perryridge	1300
Gopal	Perryridge	5300
Johnson	Downtown	1500
Loreena	Downtown	1300
Peterson	Downtown	2500
Rao	Austin	1500
Sato	Austin	1600

$_{branch\_name}\mathcal{G}_{count(branch\_name)}(pt\_works)$

branch_name	$count(branch\_name)$
Perryridge	3
Downtown	3
Austin	2

$_{branch\_name}\mathcal{G}_{sum(salary)}(pt\_works)$

branch_name	$sum(branch\_name)$
Perryridge	8100
Downtown	5300
Austin	3100

Relational Algebra

- Operators of relational algebra
  - Motivation
    - Revisit a natural join $r \bowtie s$
    - We want all the information on employee in employee and ft-work relations
      - The yellow information is omitted in the joined relation
    - Introduces OUTER JOIN

employee relation

employee_name	Street	City
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

ft_works relation

employee_name	branch_name	salary
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

$employee \bowtie ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500

Relational Algebra

- Operators of relational algebra
  - LEFT OUTER JOIN, ⟕, (ALT+10197)
    - $r \mathbin{⟕} s = (r \bowtie s) \cup \left(r - \pi_R(r \bowtie s)\right) \times \{null, \ldots, null\}$
    - An extension of the natural join that avoids loss of information
    - Include tuples of r not matched in any values of common fields in tuples of s

employee relation

employee_name	Street	City
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

ft_works relation

employee_name	branch_name	salary
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

$employee \bowtie ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500

$employee \mathbin{⟕} ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500
Smith	Resolver	Death Valley	**null**	**null**

Relational Algebra

- Operators of relational algebra
  - RIGHT OUTER JOIN, ⋈, (ALT+10198)
    - $r ⟖ s$
    - An extension of the natural join that avoids loss of information
    - Include tuples of r not matched in any values of common fields in tuples of s

employee relation

employee_name	Street	City
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

ft_works relation

employee_name	branch_name	salary
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

$employee ⋈ ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500

$employee ⟖ ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500
Gates	null	null	Redmond	5300

Relational Algebra

- Operators of relational algebra
  - FULL OUTER JOIN, ⟗, (ALT+10199)
    - $r ⟗ s$
    - An extension of the natural join that avoids loss of information
    - Include tuples of r not matched in any values of common fields in tuples of s

employee relation

employee_name	Street	City
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

ft_works relation

employee_name	branch_name	salary
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

$employee ⟗ ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	Mesa	1500
Rabbit	Tunnel	Carrotville	Mesa	1300
Williams	Seaview	Seattle	Redmond	1500
Smith	Resolver	Death Valley	null	null
Gates	null	null	Redmond	5300

Relational Algebra

- Operators of relational algebra
  - OUTER UNION, $\cup^+$, (\cup^+)
    - $r \cup^+ s$,
    - $R(a_{c1}, a_{c2}, \dots, a_{cn}, a_{r1}, a_{r2}, \dots, a_{rm}), S(a_{c1}, a_{c2}, \dots, a_{cn}, a_{s1}, a_{s2}, \dots, a_{so})$
    - $\rightarrow (a_{c1}, a_{c2}, \dots, a_{cn}, a_{r1}, a_{r2}, \dots, a_{rm}, a_{cn}, a_{s1}, a_{s2}, \dots, a_{so})$, degree = n + m + o
    - **match attribute**
    - Cardinality = |r| + |s|

Union: Two relations to be merged must have
- the identical number of attributes (degree)
- domain of each attribute on two relations is identical

employee relation

employee_name	Street	City
Coyote	Toon	Hollywood
Rabbit	Tunnel	Carrotville
Smith	Revolver	Death Valley
Williams	Seaview	Seattle

ft_works relation

employee_name	branch_name	salary
Coyote	Mesa	1500
Rabbit	Mesa	1300
Gates	Redmond	5300
Williams	Redmond	1500

$employee \cup^+ ft\_works$

employee_name	Street	City	branch_name	Salary
Coyote	Toon	Hollywood	null	null
Rabbit	Tunnel	Carrotville	null	null
Smith	Revolver	Death Valley	null	null
Williams	Seaview	Seattle	null	null
Coyote	null	null	Mesa	1500
Rabbit	null	null	Mesa	1300
Gates	null	null	Redmond	5300
Williams	null	null	Redmond	1500

Relational Algebra

- Null value
  - Unknown or non-existing values
  - Let **K** be an applicable value or null

Expression	Result
K +,-,*,/ null	
null +,-,*,/ K	
K $<, \leq, \geq, >, \neq$ null	
null $<, \leq, \geq, >, \neq$ K	
$(true \wedge unknown)$	
$(false \wedge unknown)$	
$(unknown \wedge unknown)$	
$(true \vee unknown)$	
$(false \vee unknown)$	
$(unknown \vee unknown)$	
$\neg\, unknown$	

Relational Algebra

- Null value
  - Unknown or non-existing values

  - SELECT, $\sigma_P(r)$
    - Apply P on each tuple t
    - If the result of P on t is true, t will be added to the output relation
    - If the result of P on t is false or unknown, t won't be added

  - JOIN, r $\bowtie s = \pi_{R \cup S}(\sigma_{r.A_1=s.A_1 \wedge r.A_2=s.A_2 \wedge \cdots \wedge r.A_n=s.A_n} r \times s)$
    - $R \cap S = \{A_1, A_2, \ldots, A_n\}$
    - null value won't be added (see SELECT how to handle null)

a relation

a	c
1	1
2	2
3	NULL
4	NULL

b relation

b	c
5	1
6	2
7	NULL

a $\bowtie$ b relation

a	c	b
1	1	5
2	2	6

Relational Algebra

- Null value
  - Unknown or non-existing values

c relation

a	b
1	1
2	2
3	NULL
4	NULL

$\pi_b(c)$

b
1
2
NULL

  - PROJECT, $\pi_{a1,a2\ldots,an}(r)$
    - deduplicate NULL
    - UNION, INTERSECTION, DIFFERENCE, Generalized-projection are same
  - AGGREGATE, $\mathcal{G}$

x relation

val
1
2
3
4
NULL

$\mathcal{G}_{sum(val)}(x)$ $\mathcal{G}_{avg\ (val)}(x)$ $\mathcal{G}_{count\ (val)}(x)$

Relational Algebra

- Operators of relational algebra
  - DELETION, $-$
    - $r \leftarrow r - E$
    - Delete tuples (output relation, E) from a relation r (no output)
    - Tuple-level deletion

  - Example
    - Delete all the smith accounts
      - $depositor \leftarrow depositor - \sigma_{customer\_name="Smith"}(depositor)$
    - Delete the loans that their amount is between 1000 and 1300
      - $loan \leftarrow loan - \sigma_{amount \geq 1000 \wedge amount \leq 1300}(loan)$
    - Delete all the Brooklyn account

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

loan relation

loan_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

branch **relation**

branch_name	branch_city	assets
Brighton	Brooklyn	7100000
Downtown	Brooklyn	9000000
Mianus	Horseneck	400000
North Town	Rye	3700000
Perryridge	Horseneck	1700000
Pownal	Bennington	300000
Redwood	Palo Alto	2100000
Round Hill	Horseneck	8000000

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Relational Algebra

- Operators of relational algebra
  - INSERTION
    - $r \leftarrow r \cup E$
    - INSERT tuples (output relation, E) into a relation r (no output)

  - Example
    - $account \leftarrow account \cup \{(A-973, "Perryridge", 1200)\}$
    - $depositor \leftarrow depositor \cup \{("Smith", A-973)\}$

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Relational Algebra

- Operators of relational algebra
  - INSERTION
    - $r \leftarrow r \cup E$
    - INSERT tuples (output relation, E) into a relation r (no output)

  - Example: Open new accounts having $200 for people who has a loan in Perryridge
    -
    -
    -
    -

borrower relation

customer_name	**account_number**
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

loan relation

account_number	branch_name	Amount
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350
L-16	**Perryridge**	**200**
L-15	**Perryridge**	**200**

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305
Adams	**L-16**
Hayes	**L-15**

goal

Relational Algebra

- Operators of relational algebra
  - INSERTION
    - $r \leftarrow r \cup E$
    - INSERT tuples (output relation, E) into a relation r (no output)

  - Example: Open new accounts having $200 for people who has a loan in Perryridge
    - $r_1 \leftarrow \sigma_{branch\_name="Perryridge"}(borrower \bowtie loan)$
    - $r_2 \leftarrow \pi_{loan\_number,branch\_name}(r_1)$
    - $account \leftarrow account \cup (r_2 \times \{(200)\})$
    - $depositor \leftarrow depositor \cup \pi_{customer\_name,loan\_number}(r_1))$

$$\boldsymbol{borrower \bowtie loan}$$

customer_name	account_number	branch_name	amount
Adams	L-16	Perryridge	1300
Curry	L-93	Mianus	500
Hayes	L-15	Perryridge	1500
Jackson	L-14	Downtown	1500
Jones	L-17	Downtown	1000
Smith	L-11	Round Hill	900
Smith	L-23	Redwood	2000
Williams	L-17	Downtown	1000

Relational Algebra

- Operators of relational algebra
  - INSERTION
    - $r \leftarrow r \cup E$
    - INSERT tuples (output relation, E) into a relation r (no output)

  - Example: Open new accounts having $200 for people who has a loan in Perryridge
    - $r_1 \leftarrow \sigma_{branch\_name="Perryridge"}(borrower \bowtie loan)$
    - $r_2 \leftarrow \pi_{loan\_number,branch\_name}(r_1)$
    - $account \leftarrow account \cup (r_2 \times \{(200)\})$
    - $depositor \leftarrow depositor \cup \pi_{customer\_name,loan\_number}(r_1))$

$$r_1 \leftarrow \sigma_{branch\_name="Perryridge"} borrower \bowtie loan$$

customer_name	account_number	branch_name	amount
Adams	L-16	Perryridge	1300
Hayes	L-15	Perryridge	1500

$$r_2 \leftarrow \pi_{loan\_number,branch\_name}(r_1)$$

account_number	branch_name
L-16	Perryridge
L-15	Perryridge

Relational Algebra

- Operators of relational algebra
  - INSERTION
    - $r \leftarrow r \cup E$
    - INSERT tuples (output relation, E) into a relation r (no output)

  - Example: Open new accounts having $200 for people who has a loan in Perryridge
    - $r_1 \leftarrow \sigma_{branch\_name="Perryridge"}(borrower \bowtie loan)$
    - $r_2 \leftarrow \pi_{loan\_number,branch\_name}(r_1)$
    - $account \leftarrow account \cup (r_2 \times \{(200)\})$
    - $depositor \leftarrow depositor \cup \pi_{customer\_name,loan\_number}(r_1))$

$$r_2 \times \{(200)\}$$

account_number	branch_name	
L-16	Perryridge	200
L-15	Perryridge	200

$$account \leftarrow account \cup r_2 \times \{(200)\}$$

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350
L-16	**Perriridge**	**200**
L-15	**Perriridge**	**200**

Relational Algebra

- Operators of relational algebra
  - INSERTION
    - $r \leftarrow r \cup E$
    - INSERT tuples (output relation, E) into a relation r (no output)

  - Example: Open new accounts having $200 for people who has a loan in Perryridge
    - $r_1 \leftarrow \sigma_{branch\_name="Perryridge"}(borrower \bowtie loan)$
    - $r_2 \leftarrow \pi_{loan\_number,branch\_name}(r_1)$
    - $account \leftarrow account \cup (r_2 \times \{(200)\})$
    - $depositor \leftarrow depositor \cup \pi_{customer\_name,loan\_number}(r_1))$

r_1

customer_name	account_number
Adams	L-16
Hayes	L-15

depositor relation

customer_name	account_number
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305
Adams	**L-16**
Hayes	**L-15**

Relational Algebra

- Operators of relational algebra
  - UPDATE
    - update all: using generalized-projection
      - $r \leftarrow \pi_{F_1,F_2,\ldots,F_n}(r)$
    - update some tuples
      - $r \leftarrow \pi_{F_1,F_2,\ldots,F_n}(\sigma_P(r)) \cup (r - \sigma_p(r))$
  - Example
    - add interest 5% to all accounts
      -
    - add interest 6% to the accounts that exceeds $700 balance
      -

account relation

account_number	branch_name	balance
A-101	Downtown	500
A-102	Perryridge	400
A-201	Brighton	900
A-215	Mianus	700
A-217	Brighton	750
A-222	Redwood	700
A-305	Round Hill	350

Relational Algebra

- Relational Calculus?

Summary

- Relational Model
- Relational Algebra
  - SELECT
  - PROJECT
  - UNION
  - INTERSECTION
  - DIFFERENCE
  - CARTESIAN PRODUCT
  - RENAME
  - JOIN
  - DIVISION
  - ASSIGNMENT
  - AGGREGATION
  - DELETION
  - INSERTION
  - UPDATE

- Next Class
  - SQL