COSC 304
Introduction to Database Systems

Relational Model and Algebra

Dr. Ramon Lawrence
University of British Columbia Okanagan
ramon.lawrence@ubc.ca

Relational Model History

The relational model was proposed by E. F. Codd in 1970.

One of the first relational database systems, System R, developed at IBM led to several important breakthroughs:

- the first version of SQL
- various commercial products such as Oracle and DB2
- extensive research on concurrency control, transaction management, and query processing and optimization

Commercial implementations (RDBMSs) appeared in the late 1970s and early 1980s. Currently, the relational model is the foundation of the majority of commercial database systems.

Relational Model Definitions

A ***relation*** is a table with columns and rows.

An ***attribute*** is a named column of a relation.

A ***tuple*** is a row of a relation.

A ***domain*** is a set of allowable values for one or more attributes.

The ***degree*** of a relation is the number of attributes it contains.

The ***cardinality*** of a relation is the number of tuples it contains.

A ***relational database*** is a collection of normalized relations with distinct relation names.

The ***intension*** of a relation is the structure of the relation including its domains.

The ***extension*** of a relation is the set of tuples currently in the relation.

Relation Example

relation

attributes

tuples

Products : Table

Product ID	Product Name	Supplier	Category	Quantity Per Unit	Unit Price	Units In Stock
1	Chai	1	1	10 boxes x 20 bags	$18.00	39
2	Chang	1	1	24 - 12 oz bottles	$19.00	17
3	Aniseed Syrup	1	2	12 - 550 ml bottles	$10.00	13
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	$22.00	53
5	Chef Anton's Gumbo Mix	2	2	36 boxes	$21.35	0
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	$25.00	120
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	$30.00	15
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	$40.00	6
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	$97.00	29
10	Ikura	4	8	12 - 200 ml jars	$31.00	31
11	Queso Cabrales	5	4	1 kg pkg.	$21.00	22

Record: 1 of 77

Degree = 7

Cardinality = 77

Domain of Unit Price is *currency*.

Definition Matching Question

Question: Given the three definitions, select the ordering that contains their related definitions.

1) relation
2) tuple
3) attribute

A) column, row, table
B) row, column, table
C) table, row, column
D) table, column, row

Cardinality and Degree Question

Question: A database table has 10 rows and 5 columns. Select ***one*** true statement.

A) The table's degree is 50.

B) The table's cardinality is 5.

C) The table's degree is 10.

D) The table's cardinality is 10.

Relation Practice Questions

Order : Select Query

Order ID	Customer	Employee	Order Date	Shipped Date	Ship Via	Ship Name	Ship Address	Ship Postal Code
10248	VINET	5	04-Aug-94	16-Aug-94	3	Vins et alcools Chevalier	59 rue de l'Abbaye	51100
10249	TOMSP	6	05-Aug-94	10-Aug-94	1	Toms Spezialitäten	Luisenstr. 48	44087
10250	HANAR	4	08-Aug-94	12-Aug-94	2	Hanari Carnes	Rua do Paço, 67	05454-876
10251	VICTE	3	08-Aug-94	15-Aug-94	1	Victuailles en stock	2, rue du Commerce	69004
10252	SUPRD	4	09-Aug-94	11-Aug-94	2	Suprêmes délices	Boulevard Tirou, 255	B-6000
10253	HANAR	3	10-Aug-94	16-Aug-94	2	Hanari Carnes	Rua do Paço, 67	05454-876
10254	CHOPS	5	11-Aug-94	23-Aug-94	2	Chop-suey Chinese	Hauptstr. 31	3012
10255	RICSU	9	12-Aug-94	15-Aug-94	3	Richter Supermarkt	Starenweg 5	1204
10256	WELLI	3	15-Aug-94	17-Aug-94	2	Wellington Importadora	Rua do Mercado, 12	08737-363
10257	HILAA	4	16-Aug-94	22-Aug-94	3	HILARIÓN-Abastos	Carrera 22 con Ave. Carlos	5022
10258	ERNSH	1	17-Aug-94	23-Aug-94	1	Ernst Handel	Kirchgasse 6	8010
10259	CENTC	4	18-Aug-94	25-Aug-94	3	Centro comercial Moctezuma	Sierras de Granada 9993	05022
10260	OTTIK	4	19-Aug-94	29-Aug-94	1	Ottilies Käseladen	Mehrheimerstr. 369	50739

Record: 1 of 827

1) What is the name of the relation?

2) What is the cardinality of the relation?

3) What is the degree of the relation?

4) What is the domain of order date? What is the domain of order id?

5) What is larger the size of the intension or extension?

Relational Model Formal Definition

The relational model may be visualized as tables and fields, but it is formally defined in terms of sets and set operations.

A ***relation schema*** ***R*** with attributes ***A*** =<A_1, A_2, …, A_n> is denoted ***R*** (A_1, A_2, …, A_n) where each A_i is an attribute name that ranges over a domain D_i denoted dom(A_i).

Example: Product (id, name, supplierId, categoryId, price)

- ***R*** = Product (relation name)
- Set ***A*** = {id, name, supplierId, categoryId, price}
- dom(price) is set of all possible positive currency values
- dom(name) is set of all possible strings that represent people's names

Relation Schemas and Instances

A ***relation schema*** is a definition of a single relation.

- The relation schema is the *intension* of the relation.

A ***relational database schema*** is a set of relation schemas (modeling a particular domain).

A ***relation instance*** denoted $r(R)$ over a relation schema $\boldsymbol{R}(A_1, A_2, \ldots, A_n)$ is a set of n-tuples $<d_1, d_2, ..., d_n>$ where each d_i is an element of dom(A_i) or is **null**.

- The relation instance is the *extension* of the relation.
- A value of **null** represents a missing or unknown value.

Cartesian Product (review)

The ***Cartesian product*** written as $D_1 \times D_2$ is a set operation that takes two sets D_1 and D_2 and returns the set of all ordered pairs such that the first element is a member of D_1 and the second element is a member of D_2.

Example:

- $D_1 = \{1,2,3\}$
- $D_2 = \{A,B\}$
- $D_1 \times D_2 = \{(1,A), (2,A), (3,A), (1,B), (2,B), (3,B)\}$

Practice Questions:

- 1) Compute $D_2 \times D_1$.
- 2) Compute $D_2 \times D_2$.
- 3) If $|D|$ denotes the number of elements in set D, how many elements are there in $D_1 \times D_2$ in general.
  - ⇨ What is the *cardinality* of $D_1 \times D_2 \times D_1 \times D_1$? A) 27 B) 36 C) 54

Relation Instance

A ***relation instance*** **r(R)** can also be defined as a subset of the *Cartesian product* of the *domains* of all *attributes* in the *relation schema*. That is,

$$\mathbf{r(R)} \subseteq dom(A_1) \times dom(A_2) \times \ldots \times dom(A_n)$$

Example:

- R = Person(id, firstName, lastName)
- dom(id) = {1,2}, dom(firstName) = {Joe, Steve}
- dom(lastName) = {Jones, Perry}
- dom(id) × dom(firstName) × dom(lastName) =
  - ⇨ { (1,Joe,Jones), (1,Joe,Perry), (1,Steve,Jones), (1,Steve,Perry), (2,Joe,Jones), (2,Joe,Perry), (2,Steve,Jones), (2,Steve,Perry)}
- Assume our DB stores people Joe Jones and Steve Perry, then r(R) = { (1,Joe, Jones), (2,Steve,Perry)}.

Properties of Relations

A relation has several properties:

- 1) Each relation name is unique.
  - No two relations have the same name.
- 2) Each cell of the relation (value of a domain) contains exactly one atomic (single) value.
- 3) Each attribute of a relation has a distinct name.
- 4) The values of an attribute are all from the same domain.
- 5) Each tuple is distinct. There are no duplicate tuples.
  - This is because relations are sets. In SQL, relations are bags.
- 6) The order of attributes is not really important.
  - Note that this is different that a mathematical relation and our definitions which specify an ordered tuple. The reason is that the attribute names represent the domain and can be reordered.
- 7) The order of tuples has no significance.

Relational Keys

Keys are used to uniquely identify a tuple in a relation.

- Note that keys apply to the relational schema not to the relational instance. That is, looking at the current data cannot tell you for sure if the set of attributes is a key.

A ***superkey*** is a set of attributes that uniquely identifies a tuple in a relation.

A ***key*** is a *minimal* set of attributes that uniquely identifies a tuple in a relation.

A ***candidate key*** is one of the possible keys of a relation.

A ***primary key*** is the candidate key designated as the distinguishing key of a relation.

A ***foreign key*** is a set of attributes in one relation referring to the primary key of another relation.

- Foreign keys allow referential integrity to be enforced.

Keys and Superkeys Question

Question: **True or false:** A key is always a superkey.

A) true

B) false

Keys and Superkeys Question (2)

Question: **True or false:** It is possible to have more than one key for a table and the keys may have different numbers of attributes.

A) true

B) false

Keys and Superkeys Question (3)

Question: **True or false:** It is possible to always determine if a field is a key by looking at the data in the table.

A) true

B) false

Example Relations

Employee-Project Database:

- Employees have a unique number, name, title, and salary.
- Projects have a unique number, name, and budget.
- An employee may work on multiple projects and a project may have multiple employees. An employee on a project has a particular responsibility and duration on the project.

Relations:

Emp (<u>eno</u>, ename, title, salary)

Proj (<u>pno</u>, pname, budget)

WorksOn (<u>eno</u>, <u>pno</u>, resp, dur)

Underlined attributes denote keys.

Example Relation Instances

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Questions:

1) Is *ename* a key for *emp*?
2) Is *eno* a key for *WorksOn*?
3) List all the superkeys for *WorksOn*.

Practice Questions

Consider a relation storing driver information including:

- SSN, name, driver's license number and state (unique together)

Person Relation

SSN	name	LicNum	LicState
123-45-6789	S. Smith	123-456	IA
111-11-1111	A. Lee	123-456	NY
222-22-2222	J. Miller	555-111	MT
333-33-3333	B. Casey	678-123	OH
444-44-4444	A. Adler	456-345	IA

Questions:

1) List the candidate keys for the relation.
2) Pick a primary key for the relation.
3) Is *name* a candidate key for *Person*?
4) List all the superkeys for *Person*.

Assumptions:

1) A person has only one driver's license.
2) A driver's license uniquely identifies a person.

Relational Integrity

Integrity rules are used to insure the data is accurate.

Constraints are rules or restrictions that apply to the database and limit the data values it may store.

Types of constraints:

- ***Domain constraint*** - Every value for an attribute must be an element of the attribute's domain or be `null`.
  - `null` represents a value that is currently unknown or not applicable.
  - `null` is not the same as zero or an empty string.
- ***Entity integrity constraint*** - In a base relation, no attribute of a primary key can be null.
- ***Referential integrity constraint*** - If a foreign key exists in a relation, then the foreign key value must match a primary key value of a tuple in the referenced relation or be null.

Foreign Keys Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn.eno is FK to Emp.eno

WorksOn.pno is FK to Proj.pno

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Foreign Keys Example (2)

Proj Relation

pno	pname	budget	dno
P1	Instruments	150000	D1
P2	DB Develop	135000	D2
P3	CAD/CAM	250000	D3
P4	Maintenance	310000	null
P5	CAD/CAM	500000	D1

Proj.dno is FK to Dept.dno

Department Relation

dno	dname
D1	Management
D2	Consulting
D3	Accounting
D4	Development

Integrity Constraints Question

Question: What constraint says that a primary key field cannot be null?

A) domain constraint

B) referential integrity constraint

C) entity integrity constraint

Entity Integrity Constraint Question

Question: A primary key has three fields. Only one field is `null`. Is the entity integrity constraint violated?

A) Yes

B) No

Referential Integrity Constraint Question

Question: A foreign key has a `null` value in the table that contains the foreign key fields. Is the referential integrity constraint violated?

A) Yes

B) No

Integrity Questions

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	AS
E2	null	SA	50000
E3	A. Lee	12	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
null	L. Chu	EE	30000
E7	R. Davis	ME	null
E8	J. Jones	SA	50000

WorksOn Relation

eno	pno	resp	dur
E1	P0	null	12
E2	P1	Analyst	null
null	P2	Analyst	6
E3	P3	Consultant	10
E9	P4	Engineer	48
E4	P2	Programmer	18
E5	null	Manager	24
E6	P4	Manager	48
E7	P6	Engineer	36
E7	P4	Engineer	23
null	null	Manager	40

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	null	null

Question:

How many violations of integrity constraints?

A) 8 B) 9 C) 10 D) 11 E) 12

General Constraints

There are more general constraints that some DBMSs can enforce. These constraints are often called ***enterprise constraints*** or ***semantic integrity constraints***.

Examples:

- An employee cannot work on more than 2 projects.
- An employee cannot make more money than their manager.
- An employee must be assigned to at least one project.

Ensuring the database follows these constraints is usually achieved using triggers.

Relational Algebra

A ***query language*** is used to update and retrieve data that is stored in a data model.

Relational algebra is a set of relational operations for retrieving data.

- Just like algebra with numbers, relational algebra consists of operands (which are relations) and a set of operators.

Every relational operator takes as input one or more relations and produces a relation as output.

- Closure property - input is relations, output is relations
- Unary operations - operate on one relation
- Binary operations - have two relations as input

A sequence of relational algebra operators is called a ***relational algebra expression***.

Relational Algebra Operators

Relational Operators:

- Selection σ
- Projection Π
- Cartesian product $\times$
- Join $\bowtie$
- Union $\cup$
- Difference -
- Intersection $\cap$

Note that relational algebra is the foundation of ALL relational database systems. SQL gets translated into relational algebra.

Selection Operation

The ***selection operation*** is a unary operation that takes in a relation as input and returns a new relation as output that contains a subset of the tuples of the input relation.

- That is, the output relation has the same number of columns as the input relation, but may have less rows.

To determine which tuples are in the output, the selection operation has a specified condition, called a ***predicate***, that tuples must satisfy to be in the output.

- The predicate is similar to a condition in an `if` statement.

Selection Operation Formal Definition

The selection operation on relation *R* with predicate *F* is denoted by $\sigma_F(R)$.

$$\sigma_F(R)=\{t \mid t \in R \text{ and } F(t) \text{ is true}\}$$

where

- *R* is a relation, *t* is a tuple variable
- *F* is a formula (predicate) consisting of
  - operands that are constants or attributes
  - comparison operators: <, >, =, ≠, ≤, ≥
  - logical operators: AND, OR, NOT

Selection Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

$\sigma_{title = 'EE'}(Emp)$

eno	ename	title	salary
E1	J. Doe	EE	30000
E6	L. Chu	EE	30000

$\sigma_{salary > 35000 \ OR \ title = 'PR'}(Emp)$

eno	ename	title	salary
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

Selection Question

Question: Given this table and the query:

$$\sigma_{salary > 50000 \text{ or } title='PR'}(Emp)$$

How many rows are returned?

A) 0
B) 1
C) 2
D) 3

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

Selection Question (2)

Question: Given this table and the query:

$$\sigma_{salary > 50000 \text{ or } title='PR'}(Emp)$$

How many columns are returned?

A) 0
B) 2
C) 3
D) 4

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

Selection Questions

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

Write the relational algebra expression that:

1) Returns all rows with an employee working on project P2.

2) Returns all rows with an employee who is working as a manager on a project.

3) Returns all rows with an employee working as a manager for more than 40 months.

Show the resulting relation for each case.

Projection Operation

The ***projection operation*** is a unary operation that takes in a relation as input and returns a new relation as output that contains a subset of the attributes of the input relation and all non-duplicate tuples.

- The output relation has the same number of tuples as the input relation unless removing the attributes caused duplicates to be present.
- Question: When are we guaranteed to never have duplicates when performing a projection operation?

Besides the relation, the projection operation takes as input the names of the attributes that are to be in the output relation.

Projection Operation Formal Definition

The projection operation on relation R with output attributes $A_1,\ldots,A_m$ is denoted by $\Pi_{A_1,\ldots,A_m}(R)$.

$$\Pi_{A_1,\ldots,A_m}(R)=\{t[A_1,\ldots,A_m] \mid t \in R\}$$

where

- R is a relation, t is a tuple variable
- $\{A_1,\ldots,A_m\}$ is a subset of the attributes of R over which the projection will be performed.
- Order of $A_1,\ldots,A_m$ is significant in the result.
- Cardinality of $\Pi_{A_1,\ldots,A_m}(R)$ is not necessarily the same as R because of duplicate removal.

Projection Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

$\Pi_{eno,ename}(Emp)$

eno	ename
E1	J. Doe
E2	M. Smith
E3	A. Lee
E4	J. Miller
E5	B. Casey
E6	L. Chu
E7	R. Davis
E8	J. Jones

$\Pi_{title}(Emp)$

title
EE
SA
ME
PR

Projection Question

Question: Given this table and the query:

$$\Pi_{title}(\text{Emp})$$

How many rows are returned?

A) 0
B) 2
C) 4
D) 8

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

Projection Questions

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

Write the relational algebra expression that:

1) Returns only attributes *resp* and *dur*.
2) Returns only *eno*.
3) Returns only *pno*.

Show the resulting relation for each case.

Union

Union is a binary operation that takes two relations *R* and *S* as input and produces an output relation that includes all tuples that are either in *R* or in *S* or in both *R* and *S*. Duplicate tuples are eliminated.

General form:

$$R \cup S = \{t \mid t \in R \text{ or } t \in S\}$$

where *R*, *S* are relations, *t* is a tuple variable.

R and *S* must be union-compatible. To be ***union-compatible*** means that the relations must have the same number of attributes with the same domains.

- Note that attribute names in both relations do not have to be the same. Result has attributes names of first relation.

Union Example

Emp

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23

$\Pi_{eno}(\text{Emp}) \cup \Pi_{eno}(\text{WorksOn})$

eno
E1
E2
E3
E4
E5
E6
E7
E8

Set Difference

Set difference is a binary operation that takes two relations *R* and *S* as input and produces an output relation that contains all the tuples of *R* that are not in *S*.

General form:

$$R - S = \{t \mid t \in R \text{ and } t \notin S\}$$

where *R* and *S* are relations, *t* is a tuple variable.

Note that:

- $R - S \neq S - R$
- R and S must be union compatible.

Set Difference Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23

$\Pi_{eno}(Emp) - \Pi_{eno}(WorksOn)$

eno
E4
E8

Question: What is the meaning of this query?

Question: What is $\Pi_{eno}(WorksOn) - \Pi_{eno}(Emp)$?

Intersection

Intersection is a binary operation that takes two relations *R* and *S* as input and produces an output relation which contains all tuples that are in both *R* and *S*.

General form:

$$R \cap S = \{t \mid t \in R \text{ and } t \in S\}$$

where *R*, *S* are relations, *t* is a tuple variable.

- *R* and *S* must be union-compatible.

Note that $R \cap S = R - (R - S) = S - (S - R)$.

Intersection Example

Emp

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23

$\Pi_{eno}(\text{Emp}) \cap \Pi_{eno}(\text{WorksOn})$

eno
E1
E2
E3
E5
E6
E7

Set Operations
Union-compatible Question

Question: Two tables have the same number of fields in the same order with the same types, but the names of some fields are different. **True or false:** The two tables are union-compatible.

A) true

B) false

Cartesian Product

The ***Cartesian product*** of two relations R (of degree k_1) and S (of degree k_2) is:

$$R \times S = \{t \mid t\,[A_1,\ldots,A_{k_1}] \in R \text{ and } t\,[A_{k_1+1},\ldots,A_{k_1+k_2}] \in S\}$$

The result of $R \times S$ is a relation of degree $(k_1 + k_2)$ and consists of all $(k_1 + k_2)$-tuples where each tuple is a concatenation of one tuple of R with one tuple of S.

The cardinality of R $\times$ S is $|R| * |S|$.

The Cartesian product is also known as ***cross product***.

Cartesian Product Example

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000

Emp × *Proj*

eno	ename	title	salary	pno	pname	budget
E1	J. Doe	EE	30000	P1	Instruments	150000
E2	M. Smith	SA	50000	P1	Instruments	150000
E3	A. Lee	ME	40000	P1	Instruments	150000
E4	J. Miller	PR	20000	P1	Instruments	150000
E1	J. Doe	EE	30000	P2	DB Develop	135000
E2	M. Smith	SA	50000	P2	DB Develop	135000
E3	A. Lee	ME	40000	P2	DB Develop	135000
E4	J. Miller	PR	20000	P2	DB Develop	135000
E1	J. Doe	EE	30000	P3	CAD/CAM	250000
E2	M. Smith	SA	50000	P3	CAD/CAM	250000
E3	A. Lee	ME	40000	P3	CAD/CAM	250000
E4	J. Miller	PR	20000	P3	CAD/CAM	250000

Cartesian Product Question

Question: *R* is a relation with 10 rows and 5 columns. *S* is a relation with 8 rows and 3 columns.

What is the degree and cardinality of the Cartesian product?

A) degree = 8, cardinality = 80

B) degree = 80, cardinality = 8

C) degree = 15, cardinality = 80

D) degree = 8, cardinality = 18

θ-Join

Theta (θ) join is a derivative of the Cartesian product. Instead of taking all combinations of tuples from *R* and *S*, we only take a subset of those tuples that match a given condition *F*:

$$R \bowtie_F S = \{t \mid t[A_1,\ldots,A_{k_1}] \in R \text{ and } t[A_{k_1+1},\ldots,A_{k_1+k_2}] \in S \text{ and } F(t) \text{ is true}\}$$

where

- *R*, *S* are relations, *t* is a tuple variable
- *F*(*t*) is a formula defined as that of selection.

Note that $R \bowtie_F S = \sigma_F(R \times S)$.

θ-Join Example

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

WorksOn $\bowtie_{dur*10000 > budget}$ Proj

eno	pno	resp	dur	P.pno	pname	budget
E2	P1	Analyst	24	P1	Instruments	150000
E2	P1	Analyst	24	P2	DB Develop	135000
E3	P4	Engineer	48	P1	Instruments	150000
E3	P4	Engineer	48	P2	DB Develop	135000
E3	P4	Engineer	48	P3	CAD/CAM	250000
E3	P4	Engineer	48	P4	Maintenance	310000
E5	P2	Manager	24	P1	Instruments	150000
E5	P2	Manager	24	P2	DB Develop	135000
E6	P4	Manager	48	P1	Instruments	150000
E6	P4	Manager	48	P2	DB Develop	135000
E6	P4	Manager	48	P3	CAD/CAM	250000
E6	P4	Manager	48	P4	Maintenance	310000
E7	P3	Engineer	36	P1	Instruments	150000
E7	P3	Engineer	36	P2	DB Develop	135000
E7	P3	Engineer	36	P3	CAD/CAM	250000
E7	P4	Engineer	23	P1	Instruments	150000
E7	P4	Engineer	23	P2	DB Develop	135000

Types of Joins

The θ-Join is a general join in that it allows any expression in the condition *F*. However, there are more specialized joins that are frequently used.

A ***equijoin*** only contains the equality operator (=) in formula *F*.

- e.g. WorksOn $\bowtie_{WorksOn.pno = Proj.pno}$ Proj

A ***natural join*** over two relations *R* and *S* denoted by $R \bowtie S$ is the equijoin of *R* and *S* over a set of attributes common to both *R* and *S*.

- It removes the "extra copies" of the join attributes.
- The attributes must have the same name in both relations.

Equijoin Example

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

WorksOn $\bowtie_{WorksOn.pno = Proj.pno}$ Proj

eno	pno	resp	dur	P.pno	pname	budget
E1	P1	Manager	12	P1	Instruments	150000
E2	P1	Analyst	24	P1	Instruments	150000
E2	P2	Analyst	6	P2	DB Develop	135000
E3	P4	Engineer	48	P4	Maintenance	310000
E5	P2	Manager	24	P2	DB Develop	135000
E6	P4	Manager	48	P4	Maintenance	310000
E7	P3	Engineer	36	P3	CAD/CAM	250000
E7	P4	Engineer	23	P4	Maintenance	310000

What is the meaning of this join?

Natural join Example

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

WorksOn ⋈ Proj

eno	pno	resp	dur	pname	budget
E1	P1	Manager	12	Instruments	150000
E2	P1	Analyst	24	Instruments	150000
E2	P2	Analyst	6	DB Develop	135000
E3	P4	Engineer	48	Maintenance	310000
E5	P2	Manager	24	DB Develop	135000
E6	P4	Manager	48	Maintenance	310000
E7	P3	Engineer	36	CAD/CAM	250000
E7	P4	Engineer	23	Maintenance	310000

Natural join is performed by comparing *pno* in both relations.

Join Practice Questions

Emp Relation

eno	ename	title	salary
E1	J. Doe	EE	30000
E2	M. Smith	SA	50000
E3	A. Lee	ME	40000
E4	J. Miller	PR	20000
E5	B. Casey	SA	50000
E6	L. Chu	EE	30000
E7	R. Davis	ME	40000
E8	J. Jones	SA	50000

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Compute the following joins (counts only):

1) $Emp \bowtie_{title='EE' \text{ and } budget > 400000} Proj$
2) $Emp \bowtie WorksOn$
3) $Emp \bowtie WorksOn \bowtie Proj$
4) $Proj_1 \bowtie_{Proj1.budget > Proj2.budget} Proj_2$

Outer Joins

Outer joins are used in cases where performing a join "loses" some tuples of the relations. These are called *dangling tuples*.

There are three types of outer joins:

- 1) ***Left outer join*** - $R \overset{\subset}{\bowtie} S$ - The output contains all tuples of *R* that match with tuples of *S*. If there is a tuple in *R* that matches with no tuple in *S*, the tuple is included in the final result and is padded with nulls for the attributes of *S*.
- 2) ***Right outer join*** - $R \overset{}{\bowtie} \supset S$ - The output contains all tuples of *S* that match with tuples of *R*. If there is a tuple in *S* that matches with no tuple in *R*, the tuple is included in the final result and is padded with nulls for the attributes of *R*.
- 3) ***Full outer join*** - $R \overset{\subset}{\bowtie} \supset S$ - All tuples of *R* and *S* are included in the result whether or not they have a matching tuple in the other relation.

Right Outer Join Example

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

WorksOn $\overset{}{\bowtie}_{WorksOn.pno = Proj.pno}$ Proj

eno	pno	resp	dur	P.pno	pname	budget
E1	P1	Manager	12	P1	Instruments	150000
E2	P1	Analyst	24	P1	Instruments	150000
E2	P2	Analyst	6	P2	DB Develop	135000
E3	P4	Engineer	48	P4	Maintenance	310000
E5	P2	Manager	24	P2	DB Develop	135000
E6	P4	Manager	48	P4	Maintenance	310000
E7	P3	Engineer	36	P3	CAD/CAM	250000
E7	P4	Engineer	23	P4	Maintenance	310000
null	null	null	null	P5	CAD/CAM	500000

Outer Join Question

Question: Given this table and the query:

$$\text{WorksOn} \overset{\textit{WorksOn.pno = Proj.pno}}{\supset\!\bowtie} \text{Proj}$$

How many rows are returned?

A) 10
B) 9
C) 8
D) 7

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P4	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Semi-Join and Anti-Join

A ***semi-join*** between tables returns rows from the first table where one or more matches are found in the second table.

- Semi-joins are used in `EXISTS` and `IN` constructs in SQL.

An ***anti-join*** between two tables returns rows from the first table where ***no*** matches are found in the second table.

- Anti-joins are used with `NOT EXISTS`, `NOT IN`, and `FOR ALL`.
- Anti-join is the complement of semi-join: $R \triangleright S = R - R \ltimes S$

Semi-Join Example

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Proj $\ltimes_{Proj.pno = WorksOn.pno}$ WorksOn

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000

Anti-Join Example

WorksOn Relation

eno	pno	resp	dur
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P4	Engineer	48
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P4	Engineer	23

Proj Relation

pno	pname	budget
P1	Instruments	150000
P2	DB Develop	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

Proj ▷ *Proj.pno = WorksOn.pno* WorksOn

pno	pname	budget
P5	CAD/CAM	500000

Aside: Division Operator

There is also a division operator used when you want to determine if *all* combinations of a relationship are present.

- E.g. Return the list of employees who work on *all* the projects that 'John Smith' works on.

The division operator is not a base operator and is not frequently used, so we will not spend any time on it.

- Note that $R \div S = \Pi_{R-S}(R) - \Pi_{R-S}((\Pi_{R-S}(R) \times S) - R)$.

Combining Operations

Relational algebra operations can be combined in one expression by nesting them:

$$\Pi_{eno,pno,dur}(\sigma_{ename='J.\ Doe'}(Emp) \bowtie \sigma_{dur>16}(WorksOn))$$

- Return the eno, pno, and duration for employee 'J. Doe' when he has worked on a project for more than 16 months.

Operations also can be combined by using temporary relation variables to hold intermediate results.

- We will use the assignment operator $\leftarrow$ for indicating that the result of an operation is assigned to a temporary relation.

$$empdoe \leftarrow \sigma_{ename='J.\ Doe'}(Emp)$$
$$wodur \leftarrow \sigma_{dur>16}(WorksOn)$$
$$empwo \leftarrow empdoe \bowtie wodur$$
$$result \leftarrow \Pi_{eno,pno,dur}(empwo)$$

Rename Operation

Renaming can be applied when assigning a result:

$$\text{result(EmployeeNum, ProjectNum, Duration)} \leftarrow \Pi_{eno,pno,dur}(empwo)$$

Or by using the rename operator ρ (rho):

$$\rho_{\text{result(EmployeeName, ProjectNum, Duration)}}(empwo)$$

Operator Precedence

Just like mathematical operators, the relational operators have precedence.

The precedence of operators from highest to lowest is:

- unary operators - σ, Π, ρ
- Cartesian product and joins - X, $\bowtie$
- intersection, division
- union and set difference

Parentheses can be used to changed the order of operations.

Note that there is no universal agreement on operator precedence, so we ***always*** use parentheses around the argument for both unary and binary operators.

Complete Set of Relational Algebra Operators

It has been shown that the relational operators {σ, Π, $\times$, $\cup$, -} form a complete set of operators.

- That is, any of the other operators can be derived from a combination of these 5 basic operators.

Examples:

- Intersection - $R \cap S \equiv R \cup S - ((R - S) \cup (S - R))$
- We have also seen how a join is a combination of a Cartesian product followed by a selection.

Relational Algebra Query Examples

Consider the database schema

Emp (eno, ename, title, salary)

Proj (pno, pname, budget)

WorksOn (eno, pno, resp, dur)

Queries:

- List the names of all employees.
  - $\Pi_{ename}(Emp)$
- Find the names of projects with budgets over $100,000.
  - $\Pi_{pname}(\sigma_{budget>100000}(Proj))$

Practice Questions

Relational database schema:

branch (bname, address, city, assets)
customer (cname, street, city)
deposit (accnum, cname, bname, balance)
borrow (accnum, cname, bname, amount)

1) List the names of all branches of the bank.

2) List the names of all deposit customers together with their account numbers.

3) Find all cities where at least one customer lives.

4) Find all cities with at least one branch.

5) Find all cities with at least one branch or customer.

6) Find all cities that have a branch but no customers who live in that city.

Practice Questions (2)

branch (bname, address, city, assets)
customer (cname, street, city)
deposit (accnum, cname, bname, balance)
borrow (accnum, cname, bname, amount)

1) Find the names of all branches with assets greater than $2,500,000.

2) List the name and cities of all customers who have an account with balance greater than $2,000.

3) List all the cities with at least one customer but without any bank branches.

4) Find the name of all the customers who live in a city with no bank branches.

Practice Questions (3)

branch (bname, address, city, assets)
customer (cname, street, city)
deposit (accnum, cname, bname, balance)
borrow (accnum, cname, bname, amount)

1) Find all the cities that have both customers and bank branches.

2) List the customer name and loan and deposit amounts, who have a loan larger than a deposit account at the same branch.

3) Find the name and assets of all branches which have deposit customers living in Vancouver.

4) Find all the customers who have both a deposit account and a loan at the branch with name CalgaryCentral.

5) Your own?

Other Relational Algebra Operators

There are other relational algebra operators that we will not discuss. Most notably, we often need ***aggregate operations*** that compute functions on the data.

For example, given the current operators, we cannot answer the query:

- What is the total amount of deposits at the Kelowna branch?

We will see how to answer these queries when we study SQL.

Conclusion

The ***relational model*** represents data as relations which are sets of tuples. Each relational schema consists of a set of attribute names which represent a domain.

The relational model has several forms of ***constraints*** to guarantee data integrity including:

- domain, entity integrity and referential integrity constraints

Keys are used to uniquely identify tuples in relations.

Relational algebra is a set of operations for answering queries on data stored in the relational model.

- The 5 basic relational operators are: {σ, Π, $\times$, $\cup$, -}.
- By combining relational operators, queries can be answered over the base relations.

Objectives

- Define: relation, attribute, tuple, domain, degree, cardinality, relational DB, intension, extension
- Define: relation schema, relational database schema, relation instance, null
- Perform Cartesian product given two sets.
- List the properties of relations.
- Define: superkey, key, candidate key, primary key, foreign key
- Define: integrity, constraints, domain constraint, entity integrity constraint, referential integrity constraint
- Given a relation be able to:
  - identify its cardinality, degree, domains, keys, and superkeys
  - determine if constraints are being violated

Objectives (2)

- Define: relational algebra, query language
- Define and perform all relational algebra operators.
- List the operators which form the complete set of operators.
- Show how other operators can be derived from the complete set.

Given a relational schema and instance be able to translate English queries into relational algebra and show the resulting relation.