Relational Theory
Part 1

UAlbany ICSI 410
Fall 2016

Much of the material in these slides
is taken directly from

SQL and Relational Theory by C.J. Date

and

Database System Concepts by Silbershatz et al.

Why study **Relational Theory** in a database course?

Why study **Relational Theory** in a database course?

Those who are enamoured of practice without science are like a pilot who goes into a ship without rudder or compass and never has any certainty of where he is going. Practice should always be based upon a sound knowledge of theory.

(Leonardo da Vinci)

Why study **Relational Theory** in a database course?

- **Relational theory is concerned with principles…**
- **Principle:**
  - **A source, root, origin**
  - **That which is fundamental**
  - **Essential nature**
  - **Theoretical basis**
  - **A foundational truth on which others are founded or from which they spring.**

Why study **Relational Theory** in a database course?

- **Principles *endure*.**
  - **Products and technologies change all the time.**
- **Knowledge of principles is *transferable*.**
  - **If you know the relational model, you will have knowledge and skills that you will be able to apply in every environment, and will never be obsolete.**

Why study **Relational Theory** in a database course?

- Even when you need to make compromise and tradeoffs in real world applications, knowing the principles enables you to do so from a position of **conceptual strength**.

The Relational Model

- Originally invented by E.F. Codd in 1968.

  Codd, a mathematician by training, realized that the discipline of mathematics could be used to ***inject some solid principles and rigor into a database management***.

The Relational Model

- The relational model is a *data model*.
- “Data model” has two distinct meanings in the database world.

The Relational Model

“Data Model” definitions

1. An abstract, self-contained, logical definition of the data structures, data operators, and so forth that make up *the abstract machine with which users interact*.

2. A model of the data--especially the persistent data--of some particular enterprise.

The Relational Model

“Data Model” definitions

1. Like a programming language, whose constructs can be used to solve many specific problems but in and of themselves have no direct connection with any specific problem.
2. Like a specific program written in that language--it uses the facilities provided by the model (in the first sense) to solve some specific problem.

The Relational Model

“Data Model” definitions

1. An abstract, self-contained, logical definition of the data structures, data operators, and so forth that make up *the abstract machine with which users interact*.

We are concerned with this definition for this lecture.

The second definition will be covered later in the semester.

The Relational Model

Keep in mind!

- The topic of this lecture is Relational Model and Relational Algebra.
- None of what follows concerns the implementation of either.
  - We will not be concerned with the physical realization of the model on a real computer.
  - We will not be concerned with the algorithms that might be used to implement the operations of the algebra.
- Furthermore, everything that has to do with performance is fundamentally an implementation issue.
- It is precisely the separation of model and implementation that allows us to achieve ***data independence***.

The Relational Model (Structures)

Relation (Informally)

- Primary construct of the relational model.
- Relational databases consist of a set of relations, each of which has a name.
- Relations are generally thought of as “tables”.

From Relational Database - Relational Model By Jennifer Widom

The Relational Model (Structures)

A table, or a *picture* of a relation.

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

The Relational Model (Structures)

A table, or a *picture* of a relation.

DEPT		
DNUM	NAME	MAXEMP
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

NOTE: There is a “logical difference” between a table and a relation… More on this later.
Just keep in mind that this simple representation of a relation does suggest things that are not true.

The Relational Model (Structures)

Relation (A bit more formally)

- Every relation has a **heading** and a **body**.
- The heading is a ***set*** of **attributes**.
    - An attribute is an attribute-name/type-name pair.
    - "**Types**" are equivalent to "**domains**" (according to C.J. Date)
        - A type is a named, set of values--all possible value of some specific kind.
        - Actual attributes in actual relations can take their actual values only from their types set of values.
        - Every value is of some type--in fact, exactly one type (unless type inheritance is supported.)

The Relational Model (Structures)

NOTE:
Throughout these slides, unless otherwise stated, we will use Date's convention of treating the attributes as unordered. Another convention is to treat them as sequences.

Relation (A bit more formally)

- Every relation has a **heading** and a **body**.
- The heading is a ***set*** of **attributes**.
  - An attribute is an attribute-name/type-name pair.
  - "**Types**" are equivalent to "**domains**" (according to C.J. Date)
    - A type is a named, set of values--all possible value of some specific kind.
    - Actual attributes in actual relations can take their actual values only from their types set of values.
    - Every value is of some type--in fact, exactly one type (unless type inheritance is supported.)

The Relational Model (Structures)

Relation (A bit more formally)

- Every relation has a **heading** and a **body**.
- The heading is a *set* of **attributes**.
  - No two attributes have the same attribute name.
  - The number of attributes in the heading is the **degree** (or **arity**).
  - Because the heading is a set, the attributes of a relation are unordered.

The Relational Model (Structures)

Relation (A bit more formally)

- Every relation has a **heading** and a **body**.
- The heading is a *set* of **attributes**.
  - No two attributes have the same attribute name.
  - The number of attributes in the heading is the **degree** (or **arity**).
  - Because the heading is a set, the attributes of a relation are unordered.
- A relation can be thought of as a subset of the Cartesian product of the attribute domains.

The Relational Model (Structures)

Relation (A bit more formally)

- Every relation has a **heading** and a **body**.
- The body is a *set* of **tuples** that conform to the heading.
  - The number of tuples in the body is the **cardinality**.
  - Because the body is defined to be a set
    - Relations *never* contain duplicate tuples.
    - The tuples of a relation are unordered.

The Relational Model (Structures)

Relation (A bit more formally)

Still a *picture* of a relation. Why?

DEPT		
DNUM: INTEGER	NAME: VARCHAR	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

The Relational Model (Structures)

Relation (A bit more formally)

How many such pictures can represent the relation DEPT?

DEPT		
DNUM: INTEGER	NAME: VARCHAR	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

The Relational Model (Structures)

Tuple (informally)

- Row in a RDBMS table.
- A record in a relational database.

The Relational Model (Structures)

Tuple (informally)

- Row in a RDBMS table.
- A record in a relational database.

A row, or a *picture* of a tuple.

DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

The Relational Model (Structures)

Tuple (somewhat formally)

- Each tuple in a relation represents an *n*-ary relationship, in the ordinary natural language sense of that term, *interrelating a set of n values* (one such value for each tuple attribute).

The Relational Model (Structures)

Tuple (formally)

- Definition
  - Let *T1, T2, …, Tn* (*n* ≥ 0) be type names, not necessarily all distinct.
  - Associate with each T_i a distinct attribute name, A_i ; each of the *n* attribute-name/type-name combinations that results is an ***attribute***.
  - Associate with each attribute an attribute value, vi, of type Ti ; each of the n attribute/value combinations that results is a *component*.
  - Then the set of all n components thus defined, t say, is a *tuple value* (or just a tuple for short) over the attributes *A1, A2, …, An*.
  - The value *n* is the *degree* of *t*.
  - The set of all *n* attributes is the *heading* of *t.*

The Relational Model: (Structures)

Tuple

DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- Degree:
- Type names:
- Corresponding attribute names:
- Corresponding attribute values:
- Heading:

The Relational Model: (Structures)

Tuple

<u>DNUM: INTEGER</u>	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- Degree: 3
- Type names:
- Corresponding attribute names:
- Corresponding attribute values:
- Heading:

The Relational Model: (Structures)

Tuple

DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- Degree: 3
- Type names: INTEGER, STRING, INTEGER
- Corresponding attribute names:
- Corresponding attribute values:
- Heading:

The Relational Model: (Structures)

Tuple

<u>DNUM: INTEGER</u>	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- Degree: 3
- Type names: INTEGER, STRING, INTEGER
- Corresponding attribute names: DNUM, NAME, MAXEMP
- Corresponding attribute values:
- Heading:

The Relational Model: (Structures)

Tuple

<u>DNUM: INTEGER</u>	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- Degree: 3
- Type names: INTEGER, STRING, INTEGER
- Corresponding attribute names: DNUM, NAME, MAXEMP
- Corresponding attribute values: 4132, “Data Processing”, 310
- Heading:

The Relational Model: (Structures)

Tuple

DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- Degree: 3
- Type names: INTEGER, STRING, INTEGER
- Corresponding attribute names: DNUM, NAME, MAXEMP
- Corresponding attribute values: 4132, “Data Processing”, 310
- Heading:

NAME: STRING	DNUM: INTEGER	MAXEMP: INTEGER

The Relational Model: (Structures)

Tuple

Note:

- Every subset of a tuple is a tuple.
- Every subset of a heading is a heading.
- A tuple containing a single value is not the same thing as that value.
    - They are of different types.

The Relational Model: (Structures)

Tuple Equality (formally)

- Definition
  - Tuples *t* and *t′* are equal if and only if they have the same attributes *A1, A2, …, An* and for all *i (i = 1, 2, …, n)*, the value *v* of *Ai* in t is equal to the value *v′* of *Ai* in *t′*.

- Two tuples are **duplicates** if and only if they are equal.

The Relational Model: (Structures)

Equal???

<u>DNUM: INTEGER</u>	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

NAME: STRING	<u>DNUM: INTEGER</u>	MAXEMP: INTEGER
Data Processing	4132	310

The Relational Model: (Structures)

Equal???

DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

DNUM: INTEGER	NAME: STRING	MINEMP: INTEGER
4132	Data Processing	310

The Relational Model: (Structures)

Relation (formally)

- Definition
  - Let *{H}* be a tuple heading and let *t1, t2, …, tm (m ≥ 0)* be distinct tuples, all with heading *{H}*.
  - Then the combination, *r* say, of *{H}* and the set of tuples *{t1, t2, …, tm}* is a relation value (or just a relation for short) over the *attributes A1, A2, …, An*, where *A1, A2, …, An* are all the attributes in *{H}*.
  - The *heading* of *r* is *{H}*; *r* has the same attributes (and hence the same attribute names and types) and the same *degree* as that heading does.
  - The set of tuples *{t1, t2, …, tm}* is the *body* of *r*.
  - The value *m* is the *cardinality* of *r*.

The Relational Model: (Structures)

Relation

DEPT		
DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

- Degree:
- Cardinality:
- Heading:
- Body:

The Relational Model: (Structures)

Relation

DEPT		
DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

- Degree: 2
- Cardinality:
- Heading:
- Body:

The Relational Model: (Structures)

Relation

DEPT		
DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

- Degree: 2
- Cardinality: 4
- Heading:
- Body:

The Relational Model: (Structures)

Relation

DEPT		
DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

- Degree: 2
- Cardinality: 4
- Heading:

DNUM: INTEGER	MAXEMP: INTEGER	NAME: STRING

- Body:

The Relational Model: (Structures)

Relation

DEPT		
DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12
8842	Payroll	27

- Degree: 2
- Cardinality: 4
- Heading:

MAXEMP: INTEGER	NAME: STRING	DNUM: INTEGER

- Body:

8842	Payroll	27
4132	Data Processing	310
7130	Data Administration	12
5187	Accounting	43

The Relational Model (Structures)

Relations

- Every subset of a body is a body.
  - Or, loosely, every subset of a relation is a relation.
- If relation *r* has *n* attributes, then *each tuple in r represents a point in a certain n-dimensional space* (and the relation overall represents a set of such points).

The Relational Model (Structures)

Candidate Keys (informally)

- A candidate key is a unique identifier.
  - It is a set of attributes such that *every tuple in the relation has a unique value for the set in question.*
  - Every relation has at least one candidate key.

The Relational Model (Structures)

Candidate Keys (informally)

- A candidate key is a unique identifier.
    - It is a set of attributes such that *every tuple in the relation has a unique value for the set in question.*
    - Every relation has at least one candidate key.

Why can we make the claim that ***<u>every</u>*** relation has at least one candidate key?

The Relational Model (Structures)

Candidate Keys (formally)

Definition:

- Let *K* be a subset of the heading of relation *R*. Then *K* is a candidate key for *R* if and only if it possesses both of the following properties:
    a. **Uniqueness**: No valid value for R contains two distinct tuples with the same value for K.
    b. **Irreducibility**: No proper subset of K has the uniqueness property.
- If *K* consists of *n* attributes, then *n* is the degree of *K*.

The Relational Model (Structures)

Candidate Keys

STUDENT				
STUDENT_ID: ID_NUM	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

- Which are candidate keys?
  a. { STUDENT_ID }
  b. { EMAIL, NAME }
  c. { SSN }

The Relational Model (Structures)

Candidate Keys

STUDENT				
STUDENT_ID: ID_NUM	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

- Which are candidate keys?
  a. { STUDENT_ID }
  ~~b. { EMAIL, NAME }~~
  c. { SSN }

The Relational Model (Structures)

Candidate Keys

STUDENT				
STUDENT_ID: ID_NUM	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

- Which are candidate keys?
  a. { STUDENT_ID }
  b. { EMAIL }
  c. { SSN }

The Relational Model (Structures)

Candidate Keys

STUDENT				
STUDENT_ID: ID_NUM	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

- Which are candidate keys?
  a. { STUDENT_ID }
  b. { EMAIL }
  c. ~~{ SSN }~~

The Relational Model (Structures)

Candidate Keys

STUDENT				
STUDENT_ID: ID_NUM	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

- ***Why is key irreducibility important in the Relational Model?***
    a. *Less typing to look up a tuple.*
    b. *Less storage space required to hold the key.*
    c. *A global constraint becomes unenforceable.*
    d. *Easier to remember.*

The Relational Model (Structures)

Candidate Keys

STUDENT				
STUDENT_ID: ID_NUM	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

- ***Why is key irreducibility important in the Relational Model?***
  a. ***Less typing to look up a tuple.***
  b. ***Less storage space required to hold the key.***
  c. ***A global constraint becomes unenforceable.***
  d. ***Easier to remember.***

The Relational Model (Structures)

Primary Key

Definition

- A candidate key that is chosen by the database designer as the primary means of identifying tuples within the relation.

STUDENT				
<u>STUDENT_ID: ID_NUM</u>	EMAIL: EMAIL_ADDR	SSN: SSN	NAME: NAME	DOB: DATE

OR

STUDENT				
STUDENT_ID: ID_NUM	<u>EMAIL: EMAIL_ADDR</u>	SSN: SSN	NAME: NAME	DOB: DATE

The Relational Model (Structures)

Superkey

Definition

- Let *SK* be a subset of the heading of relation *R* that possesses the uniqueness property but not necessarily the irreducibility property.
    - Then *SK* is a ***superkey*** for *R* (and a superkey that is not a candidate key is called a ***proper superkey***.
- Note that the heading for any relation *R* is a superkey for *R*.

The Relational Model (Structures)

Foreign Key (informally)

- A foreign key is a set of attributes in one relation whose values are supposed to correspond to the values of some candidate key--the *target key*--in some other relation (or possibly the same relation).

The Relational Model (Structures)

Foreign Key (informally)

- A foreign key is a set of attributes in one relation whose values are supposed to correspond to the values of some candidate key--the *target key*--in some other relation (or possibly the same relation).

DEPT		
DNUM	NAME	MAXEMP

EMP			
ID	Name	DNUM	MANAGER_ID

The Relational Model (Structures)

Foreign Key (formally)

Definition

- Let *R1* and *R2* be relations, not necessarily distinct, and let *K* be a key for *R1*.
- Let *FK* be a subset of the heading of *R2* such that there exists a possibly empty sequence of attribute renamings on *R1* that maps *K* into *K'* (say), where *K'* and *FK* contain exactly the same attributes.
- Further, let *R2* and *R1* be subject to the constraint that, at all times, every tuple *t2* in *R2* has an *FK* value that's the *K'* value for some *(necessarily unique)* tuple *t1* in *R1* at the same time in question.
- Then, *FK* is the *foreign key*; *K* is the corresponding *target key*; the associated constraint is a *referential constraint*; and *R2* and *R1* are the *referencing relation* and *referenced relation*, respectively, for that constraint.

The Relational Model

Relations as Predicates

- Most people think of relations as if they were just files in the traditional computing sense--rather abstract files, but files none the less.
- There is a different way to look at them that can lead to a much deeper understanding of what’s really going on.
- Consider that relations represent some portion of the real world.

The Relational Model

Relations as Predicates

- To be more precise:
    - The heading of that relation represents a certain ***predicate***, meaning that it is *a generic statement about some portion of the real world.*
    - Consider the DEPT relation:

DEPT		
DNUM: INTEGER	NAME: STRING	MAXEMP: INTEGER
4132	Data Processing	310

- The predicate would look like:
    - *Department DNUM, named NAME, has max employees MAXEMP.*
- ***This predicate is the intended interpretation for the DEPT relation.***

The Relational Model

Relations as Predicates

- To be more precise:
  - The heading of that relation represents a certain ***predicate***, meaning that it is *a generic statement about some portion of the real world.*
  - The statement is generic because it is parameterized.

The Relational Model

Relations as Predicates

- To be more precise:
    - The heading of that relation represents a certain *predicate*, meaning that it is *a generic statement about some portion of the real world.*
    - The statement is generic because it is parameterized.
        - You can think of a predicate as a truth values function.

The Relational Model

Relations as Predicates

- To be more precise:
  - The heading of that relation represents a certain ***predicate***, meaning that it is *a generic statement about some portion of the real world.*
  - The statement is generic because it is parameterized.
    - You can think of a predicate as a truth values function.
    - Every tuple *t* appearing in a relation *R* can thought of as a proposition derived by invoking that truth function with the attribute values of *t* as the arguments.

The Relational Model

Relations as Predicates

- To be more precise:
    - The heading of that relation represents a certain ***predicate***, meaning that it is *a generic statement about some portion of the real world.*
    - The statement is generic because it is parameterized.
        - You can think of a predicate as a truth values function.
        - Every tuple *t* appearing in a relation *R* can thought of as a proposition derived by invoking that truth function with the attribute values of *t* as the arguments.
        - We assume that each proposition so obtained evaluates to TRUE.

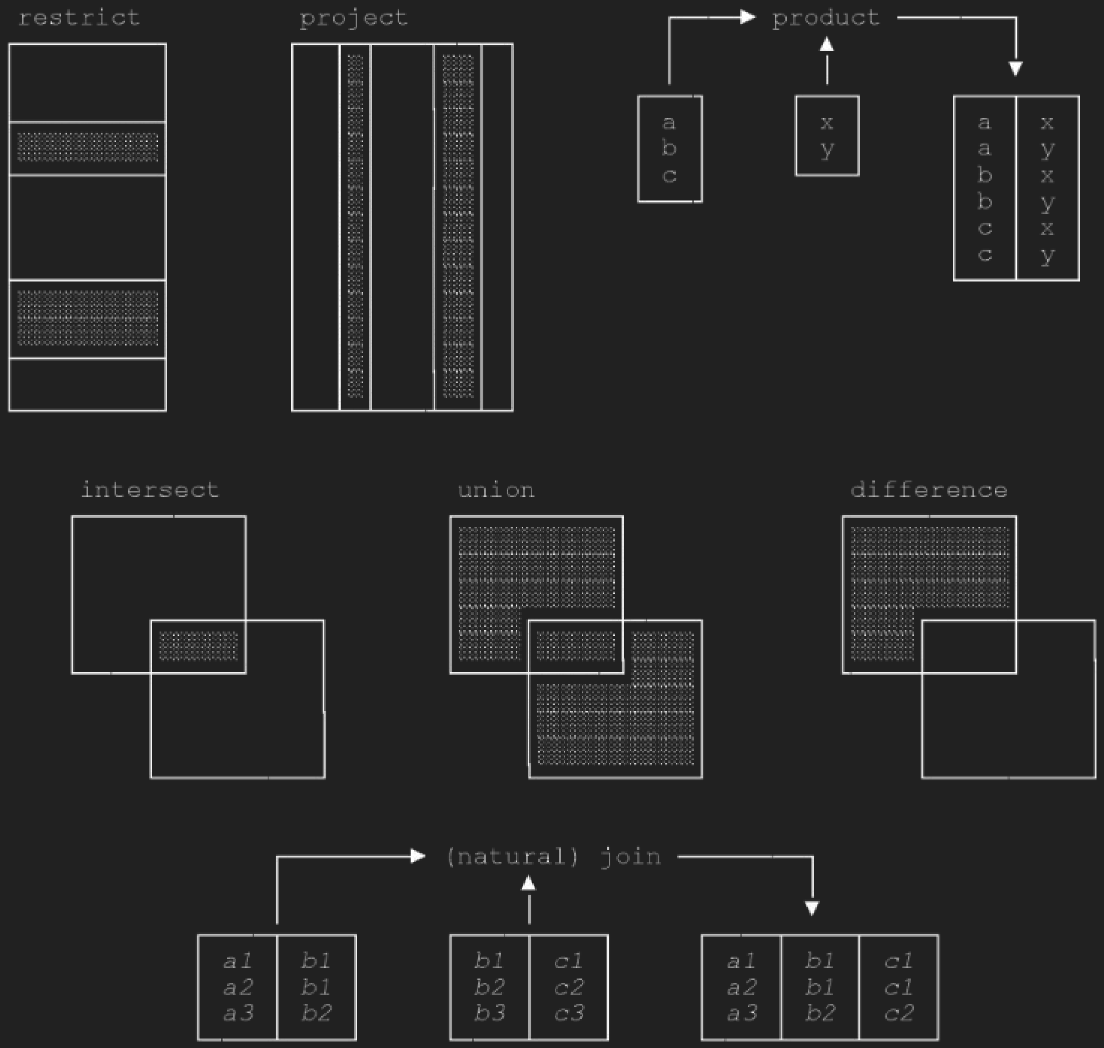
The Relational Model (Structures)

Relations as Predicates

- A database can be thought of as a collection of true propositions.
- A database, together with the operators that apply to the propositions represented in that database, is a *logical system*.
  - It has axioms
  - It has rules of inference
  - We can prove theorems (“derived truths”) from those axioms.
- The inference rules are the rules that tell us how to apply the operators of the relational algebra.

Relational Algebra

- A procedural query language.
- Consists of a set of operations that *take one or more relations as input and produce a new relation as their result*. This is called ***the closure property***.
- The closure property allows us to nest relational expressions.



Relational Algebra: Operations

RENAME:

- Denoted by lowercase Greek letter sigma (ρ)
- Unlike relations in the database, the results of relational-algebra expressions do not have a name that we can use to refer to them.
- It is useful to be able to give them names.

Relational Algebra: Operations

RENAME:

- Example

$$\rho_{BAR(C,D)}(FOO) = ?$$

FOO	
A	B
X	1
Y	2
Z	3

Relational Algebra: Operations

RENAME:

- Example

$$\rho_{BAR(C,D)}(FOO)$$

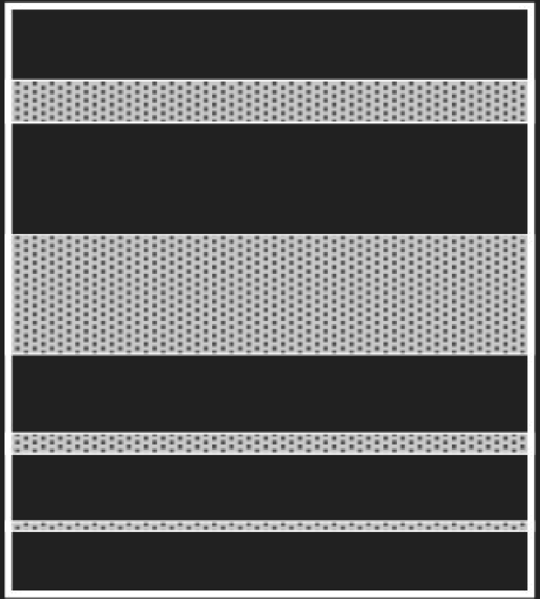
BAR	
C	D
X	1
Y	2
Z	3

Relational Algebra: Operations

SELECT (aka RESTRICT):

- Denoted by lowercase Greek letter sigma (σ)
- “Selects” tuples that satisfy a given predicate.
- The predicate appears as a subscript to σ
- The argument relation is in parentheses after the σ

SELECT



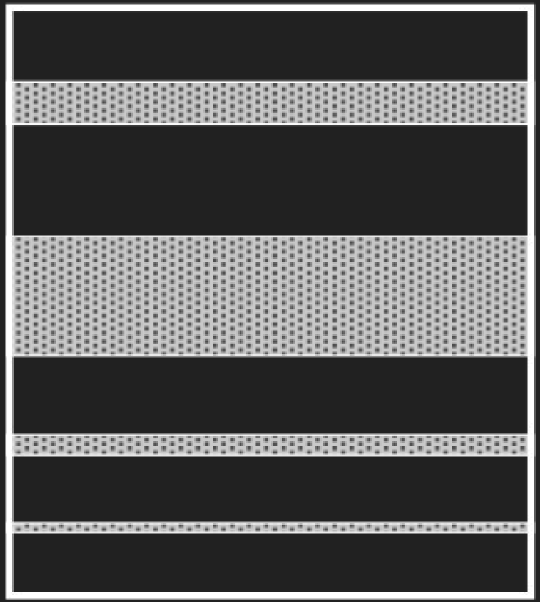
- NOTE: The term “select” in relational algebra has a different meaning than the one used in SQL. In relational algebra, the term “select” corresponds to SQL’s WHERE.

Relational Algebra: Operations

SELECT (aka RESTRICT):

- Definition:
  - Let *r* be a relation and let *bx* be a boolean expression.
  - Then *bx* is a selection condition and the selection of *r* according to *bx*, *r* WHERE *bx*, is a relation with (a) the same heading as that of *r* and (b) a body consisting of all tuples of *r* for which *bx* evaluates to TRUE.

SELECT



Relational Algebra: Operations

SELECT Example

$\sigma_{SALARY > 40000}(EMP) = ?$

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

Relational Algebra: Operations

SELECT Example

$\sigma_{SALARY > 40000}(EMP) = ?$

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

Relational Algebra: Operations

SELECT Example

ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany
22321	Brady Kathy	4132	63410	New York

$$\sigma_{SALARY > 40000}(EMP)$$

Relational Algebra: Operations

SELECT Example

ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany
22321	Brady Kathy	4132	63410	New York

$$\rho_{40K\_CLUB}(\sigma_{SALARY > 40000}(EMP)) = ?$$

Relational Algebra: Operations

SELECT Example

40K_CLUB				
ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany
22321	Brady Kathy	4132	63410	New York

$$\rho_{40K\_CLUB}(\sigma_{SALARY > 40000}(EMP))$$

Relational Algebra: Operations

SELECT Example 2

ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany
22321	Brady Kathy	4132	63410	New York

$\sigma_{\text{LOCATION = "Albany"}}(\sigma_{\text{SALARY > 40000}}(\text{EMP})) = ?$

Relational Algebra: Operations

SELECT Example 2

ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany
22321	Brady Kathy	4132	63410	New York

$\sigma_{\text{LOCATION = “Albany”}}(\sigma_{\text{SALARY > 40000}}(\text{EMP})) = ?$

Relational Algebra: Operations

SELECT Example 2

<u>ID</u>	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany

$$\sigma_{\text{LOCATION = “Albany”}}(\sigma_{\text{SALARY > 40000}}(\text{EMP}))$$

Relational Algebra: Operations

SELECT Example 2

ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany

$\sigma_{\text{LOCATION = “Albany”}}(\sigma_{\text{SALARY > 40000}}(\text{EMP}))$ can be written as $\sigma_{\text{(LOCATION = “Albany”) AND (SALARY > 40000)}}(\text{EMP})$

Relational Algebra: Operations

SELECT Example 2

ID	Name	DNUM	SALARY	LOCATION
12206	Ryan Alfred	7130	48342	Albany

$\sigma_{\text{LOCATION = “Albany”}}(\sigma_{\text{SALARY > 40000}}(\text{EMP}))$ can be written as $\sigma_{\text{(LOCATION = “Albany”) AND (SALARY > 40000)}}(\text{EMP})$

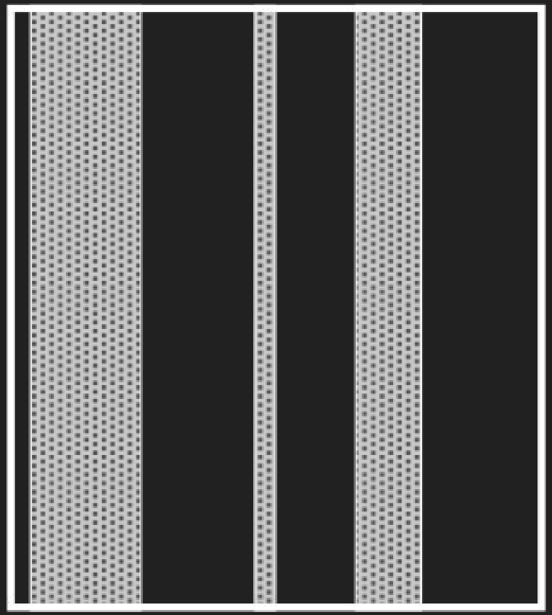
Expression transformation is a key part of query optimization. We will cover this topic later in the course.

Relational Algebra: Operations

PROJECT:

- Denoted by uppercase Greek letter pi (Π)
- Unary operation that returns its argument relation, with certain attributes left out.
- Since a relation (body) is a set, any duplicate rows are eliminated.
- We list those attributes that we wish to appear in the result as a subscript to Π.
- The argument relation is in parentheses after the Π.

PROJECT



Relational Algebra: Operations

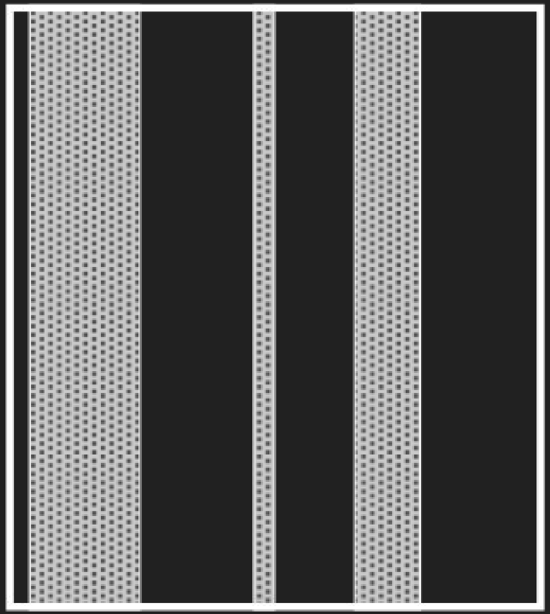
PROJECT:

- Definition:
  - Let *r* be a relation and let *A, B, …, C* be attributes of *r*.
  - Then the projection of *r* on (or over) those attributes,

$$\Pi_{A,\ B,\ \ldots C}(r)$$

is a relation with (a) heading { *A, B, …, C* } and (b) body the set of all tuples *x* such that there exists some tuple *t* in *r* with *A* value equal to the *A* value in *x*, *B* value equal to the *B* value in *x*, …, *C* value equal to the *C* value in *x*.

PROJECT



Relational Algebra: Operations

PROJECT Example

$\Pi_{\text{LOCATION, DNUM}}(\text{EMP}) = ?$

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

Relational Algebra: Operations

PROJECT Example

$\Pi_{\text{LOCATION, DNUM}}(\text{EMP}) = ?$

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

Relational Algebra: Operations

PROJECT Example

$\Pi_{\text{LOCATION, DNUM}}(\text{EMP}) = ?$

DNUM	LOCATION
7130	Albany
7130	Albany
0060	Buffalo
4132	New York
7130	Albany
4132	New York

Relational Algebra: Operations

PROJECT Example

$\Pi_{\text{LOCATION, DNUM}}(\text{EMP}) = ?$

DNUM	LOCATION
7130	Albany
7130	Albany
0060	Buffalo
4132	New York
7130	Albany
4132	New York

Relational Algebra: Operations

PROJECT Example

DNUM	LOCATION
7130	Albany
0060	Buffalo
4132	New York

$\Pi_{\text{LOCATION, DNUM}}(\text{EMP})$

Relational Algebra: Operations

PROJECT Example

DNUM	LOCATION
7130	Albany
0060	Buffalo
4132	New York

$$\sigma_{\text{Location} = \text{“Albany”}}(\Pi_{\text{LOCATION, DNUM}}(\text{EMP})) = ?$$

Relational Algebra: Operations

PROJECT Example

DNUM	LOCATION
7130	Albany
0060	Buffalo
4132	New York

$$\sigma_{\text{Location = “Albany”}}(\Pi_{\text{LOCATION, DNUM}}(\text{EMP})) = ?$$

Relational Algebra: Operations

PROJECT Example

DNUM	LOCATION
7130	Albany

$$\sigma_{\text{Location = "Albany"}}(\Pi_{\text{LOCATION, DNUM}}(\text{EMP}))$$

Relational Algebra: Operations

PROJECT Example

DNUM	LOCATION
7130	Albany

Are these two expressions equal???

1. $\sigma_{\text{Location = “Albany”}}(\Pi_{\text{LOCATION, DNUM}}(\text{EMP}))$

2. $\Pi_{\text{LOCATION, DNUM}}(\sigma_{\text{Location = “Albany”}}(\text{EMP}))$

Relational Algebra: Operations

PROJECT Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo

$$\sigma_{\text{SALARY} < 30000}(\sigma_{\text{LOCATION} = \text{“Albany”}}(\Pi_{\text{LOCATION, DNUM}}(\text{EMP}))) = ?$$

Relational Algebra: Operations

PROJECT Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo

$\cancel{\sigma_{SALARY < 30000}(\sigma_{LOCATION = \text{“Albany”}}(\Pi_{LOCATION,\ DNUM}(EMP))) = ?}$

Relational Algebra: Operations

PROJECT Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo

$\Pi_{\text{LOCATION, DNUM}}(\sigma_{(\text{SALARY} < 30000) \text{ AND } (\text{LOCATION} = \text{“Albany”})}(\text{EMP})) = ?$

Relational Algebra: Operations

PROJECT Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo

$\Pi_{\text{LOCATION, DNUM}}(\sigma_{(\text{SALARY} < 30000) \text{ AND } (\text{LOCATION} = \text{“Albany”})}(\text{EMP})) = ?$

Relational Algebra: Operations

PROJECT Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
21004	Perry Bill	0060	21876	Buffalo

$$\Pi_{\text{LOCATION, DNUM}}(\sigma_{(\text{SALARY} < 30000) \text{ AND } (\text{LOCATION} = \text{“Albany”})}(\text{EMP})) = ?$$

Relational Algebra: Operations

PROJECT Example

DNUM	LOCATION

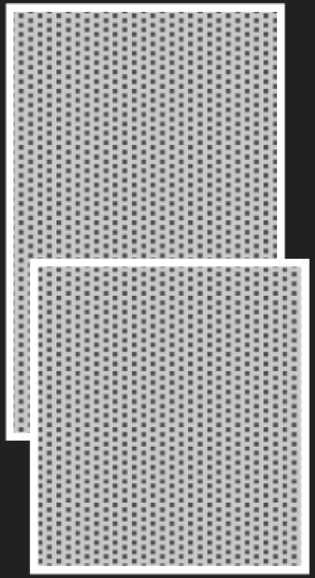
$$\Pi_{\text{LOCATION, DNUM}}(\sigma_{(\text{SALARY} < 30000)\ \text{AND}\ (\text{LOCATION} = \text{“Albany”})}(\text{EMP}))$$

Relational Algebra: Operations

UNION

UNION:

- Denoted, as in set theory, by $\cup$.
- Binary operation that returns the set theory union of the bodies of the two argument relations.
- We must make sure that unions are taken between *compatible* relations. (More on this in a later slide.)

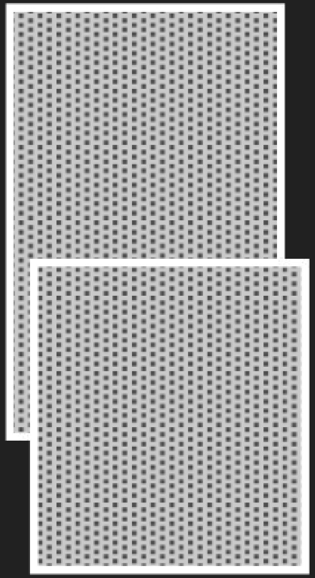


Relational Algebra: Operations

UNION

UNION:

- Definition:
  - Let relations *r1* and *r2* be two relations with the same heading.
  - Then their union, *r1* ∪ *r2*, is a relation with (a) the same heading and (b) a body consisting of all tuples *t* such that *t* appears in *r1* or *r2* or both.



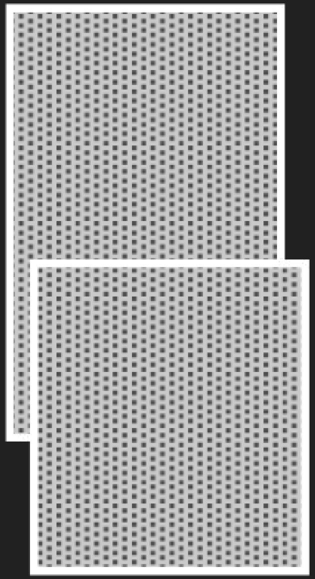
UNION

Relational Algebra: Operations

UNION Compatibility:

- Definition (from Date)
  - The relations must have the same heading.

- Definition (from Silberschatz)
  - The relations *r* and *s* must be of the same arity. That is, they must have the same number of attributes.
  - The domains of the i^{th} attribute of *r* and the i^{th} attribute of *s* must be the same, for all *i*.



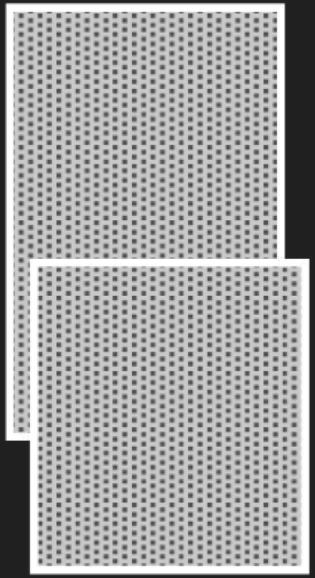
Relational Algebra: Operations

UNION

UNION Compatibility:

- Definition (from Date)
  - The relations must have the same heading.

- Definition (from Silberschatz)
  - The relations *r* and *s* must be of the same arity. That is, they must have the same number of attributes.
  - The domains of the i^{th} attribute of *r* and the i^{th} attribute of *s* must be the same, for all *i*.



Order of attributes matters. Not following the convention of treating the attributes as a set.

Relational Algebra: Operations

UNION Example

DOWNSTATE_EMP ∪ UPSTATE_EMP

UPSTATE_EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
31890	Coulsen Mary	7130	21400	Albany

DOWNSTATE_EMP				
ID	Name	DNUM	SALARY	LOCATION
22321	Brady Kathy	4132	63410	New York
47862	Anders John	4132	33700	New York

Relational Algebra: Operations

UNION Example

DOWNSTATE_EMP ∪ UPSTATE_EMP

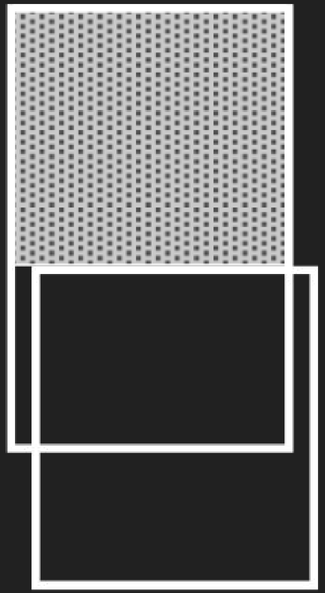
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
12206	Ryan Alfred	7130	48342	Albany
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

DIFFERENCE

Relational Algebra: Operations

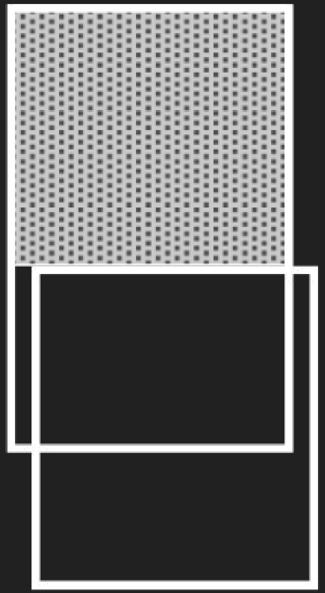
Set-Difference:

- Denoted by $-$
- Binary operation that allows us to find tuples that are in one relation but not another.
- $r - s$ produces a relation containing those tuples that are in *r* but not *s*.
- As with the union operation, we must insure that set differences are taken between compatible relations.



DIFFERENCE

Relational Algebra: Operations



Set-Difference:

- Definition:
  - Let relations r1 and r2 be relations with the same heading.
  - Then their difference, *r1* − *r2*, is a relation of the same type, with a body consisting of all tuples *t* such that *t* appears in *r1* and not in *r2*.

Relational Algebra: Operations

DIFFERENCE Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

$$\Pi_{DNUM}(DEPT) - \Pi_{DNUM}(EMP) = ?$$

DEPT		
DNUM	NAME	MAXEMP
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12

Relational Algebra: Operations

DIFFERENCE Example

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

$$\Pi_{DNUM}(DEPT) - \Pi_{DNUM}(EMP) = ?$$

DEPT		
DNUM	NAME	MAXEMP
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12

Relational Algebra: Operations

DIFFERENCE Example

$$\Pi_{DNUM}(DEPT) \; - \; \Pi_{DNUM}(EMP) = ?$$

DNUM
7130
4132

DNUM
4132
5187
7130

Relational Algebra: Operations

DIFFERENCE Example

$$\Pi_{DNUM}(DEPT) \; - \; \Pi_{DNUM}(EMP)$$

DNUM
5187

Relational Algebra: Operations

DIFFERENCE Example

$$\Pi_{DNUM}(EMP) - \Pi_{DNUM}(DEPT)$$

EMP				
ID	Name	DNUM	SALARY	LOCATION
12058	Borys Ted	7130	39200	Albany
22321	Brady Kathy	4132	63410	New York
31890	Coulsen Mary	7130	21400	Albany
47862	Anders John	4132	33700	New York

DEPT		
DNUM	NAME	MAXEMP
4132	Data Processing	310
5187	Accounting	43
7130	Data Administration	12

Relational Algebra: Operations

DIFFERENCE Example

$$\Pi_{DNUM}(EMP) \; - \; \Pi_{DNUM}(DEPT)$$

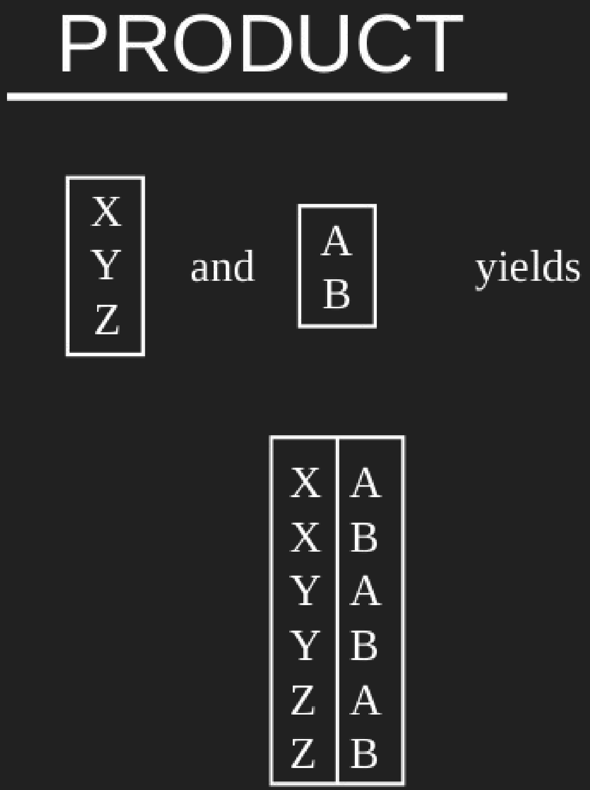
DNUM

Why must this always produce an empty table if our database is a *"faithful model of reality?"*

Relational Algebra: Operations

PRODUCT:

- Denoted by a cross (X)
- Binary operation that allows us to combine information from (any) two relations.
- Recall that a relation is a subset of the Cartesian product of a set of attribute domains.

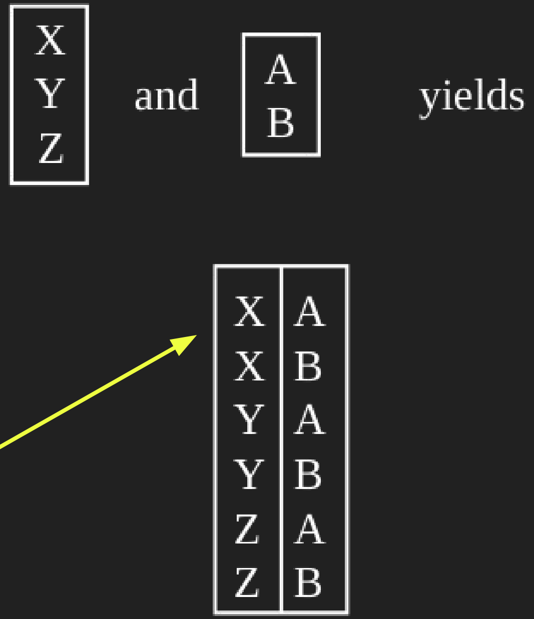


PRODUCT

Relational Algebra: Operations

PRODUCT:

- Denoted by a cross ($\times$)
- Binary operation that allows us to combine information from (any) two relations.
- Recall that a relation is a subset of the Cartesian product of a set of attribute domains.



What about the heading of this derived relation???

Relational Algebra: Operations

FOO
<u>COL</u>
X
Y
Z

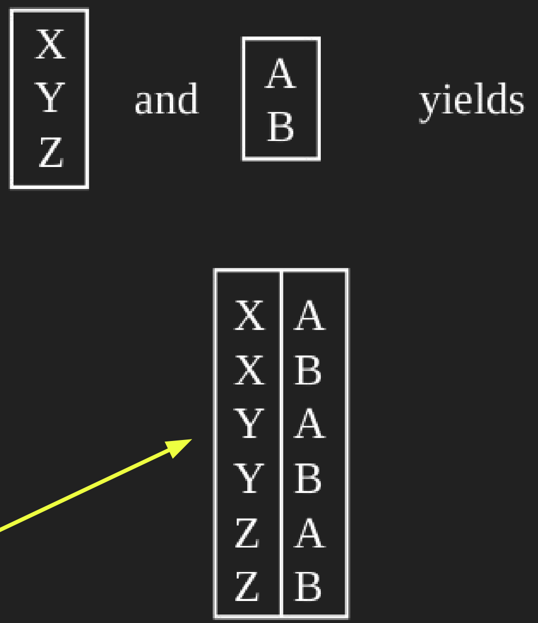
×

BAR
<u>COL</u>
A
B

$\stackrel{?}{=}$

<u>???</u>	<u>???</u>
X	A
Y	B
Z	A
X	B
Y	A
Z	B

PRODUCT



What about the heading of this derived relation???

Relational Algebra: Operations

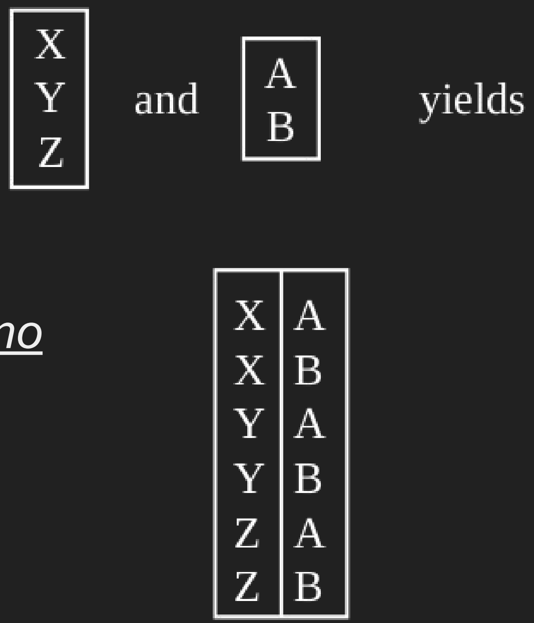
PRODUCT

PRODUCT:

- Definition (by Date)
  - The cartesian product (or just product for short) of relations *r1* and *r2*, *r1* X *r2*, *where r1 and r2 have no common attribute names*, is a relation with

    a. heading the set theory union of the headings of *r1* and *r2*.

    b. body the set of all tuples *t* such that t is the set theory union of a tuple from *r1* and a tuple from *r2*.



Note that under Date's rules, you may need rename attributes before applying PRODUCT.

Relational Algebra: Operations

PRODUCT:

- Example (under Date's rules)

FOO
<u>COL</u>
X
Y
Z

BAR
<u>COL</u>
A
B

$$\rho_{F(FOO)}(FOO) \times \rho_{B(BAR)}(BAR) = ?$$

Relational Algebra: Operations

PRODUCT:

- Example (under Date's rules)

F
FOO
X
Y
Z

B
BAR
A
B

$$\rho_{F(FOO)}(FOO) \times \rho_{B(BAR)}(BAR) = ?$$

Relational Algebra: Operations

PRODUCT:

- Example (under Date's rules)

$$\rho_{F(FOO)}(FOO) \times \rho_{B(BAR)}(BAR)$$

FOO	BAR
X	A
Y	B
Z	A
X	B
Y	A
Z	B

Relational Algebra: Operations

PRODUCT:

- Attribute naming schema (Silbershatz)
  - Since the same attribute name may appear in both r1 and r2 of r1 X r2, we need to devise a naming scheme to distinguish between these attributes.
  - We do so by attaching to an attribute name the name of the relation from which the attribute originally came.

Note that this schema requires that the relations that are the arguments of the Cartesian-product operation have distinct names.

PRODUCT

X
Y
Z

and

A
B

yields

X	A
X	B
Y	A
Y	B
Z	A
Z	B

Relational Algebra: Operations

PRODUCT:

- Example (under the attribute naming schema)

FOO
COL
X
Y
Z

BAR
COL
A
B

$\text{FOO} \times \text{BAR} = ?$

Relational Algebra: Operations

PRODUCT:

- Example (under the attribute naming schema)

FOO $\times$ BAR

FOO.COL	BAR.COL
X	A
Y	B
Z	A
X	B
Y	A
Z	B

Relational Algebra: Operations

The operations that we covered so far allow us to give us a complete definition of an expression in the relational-algebra.

- $E_1 \cup E_2$
- $E_1 - E_2$
- $E_1 \times E_2$
- $\sigma_P(E_1)$
- $\Pi_S(E_1)$
- $\rho_x(E_1)$

Relational Algebra: Operations

The operations that we covered so far allow us to give us a complete definition of an expression in the relational algebra.

- $E_1 \cup E_2$
- $E_1 - E_2$
- $E_1 \times E_2$
- $\sigma_P(E_1)$
- $\pi_S(E_1)$
- $\rho_x(E_1)$

These fundamental operations of the relational algebra are sufficient to express any relational-algebra query.

Relational Algebra: Operations

The operations that we covered so far allow us to give us a complete definition of an expression in the relational algebra.

- $E_1 \cup E_2$
- $E_1 - E_2$
- $E_1 \times E_2$
- $\sigma_P(E_1)$
- $\Pi_S(E_1)$
- $\rho_x(E_1)$

These fundamental operations of the relational algebra are sufficient to express any relational-algebra query.

However, if we restrict ourselves to just these operations, certain common queries are lengthy to express.

Relational Algebra: Operations

The operations that we covered so far allow us to give us a complete definition of an expression in the relational algebra.

- *Set-intersection*
- *Assignment*
- *Natural-Join*
- *Left outer join*
- *Right outer join*
- *Full outer join*

The next set of slides will cover these operations that do not add any power to the algebra, but which simplify common queries.