INFORMATION SYSTEM DEPARTMENT
INFORMATION TECHNOLOGY FACULTY– HCM UNIVERSITY SCIENCE

INTRODUCTION TO DATABASE

Chapter 02
Data Models & Relational Data Model & Relational Algebra

Lecturer- PhD. NGUYEN TRAN MINH THU

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Outline

- ❑ Part 1- Data Models
- ❑ Part 2- Relational Data Model
- ❑ Part 3 - Relational Algebra

Outline

- **PART 1: Data Models**
  - Data Modeling & Data Models
  - The Evolution of Data Models
  - Data Abstraction Levels
  - Business Rules

Outline

- PART 1: Data Models
  - Data Modeling & Data Models
  - The Evolution of Data Models
  - Data Abstraction Levels
  - Business Rules

Data Modeling & Data Models

- **Data modeling**: Iterative and progressive process of creating a specific data model for a determined problem domain
- **Data models**: Simple representations of complex real-world data structures
  - Useful for supporting a specific problem domain
- **Model** - Abstraction of a real-world object or event

Importance of Data Models

- Are a communication tool
- Give an overall view of the database
- Organize data for various users
- Are an abstraction for the creation of good database

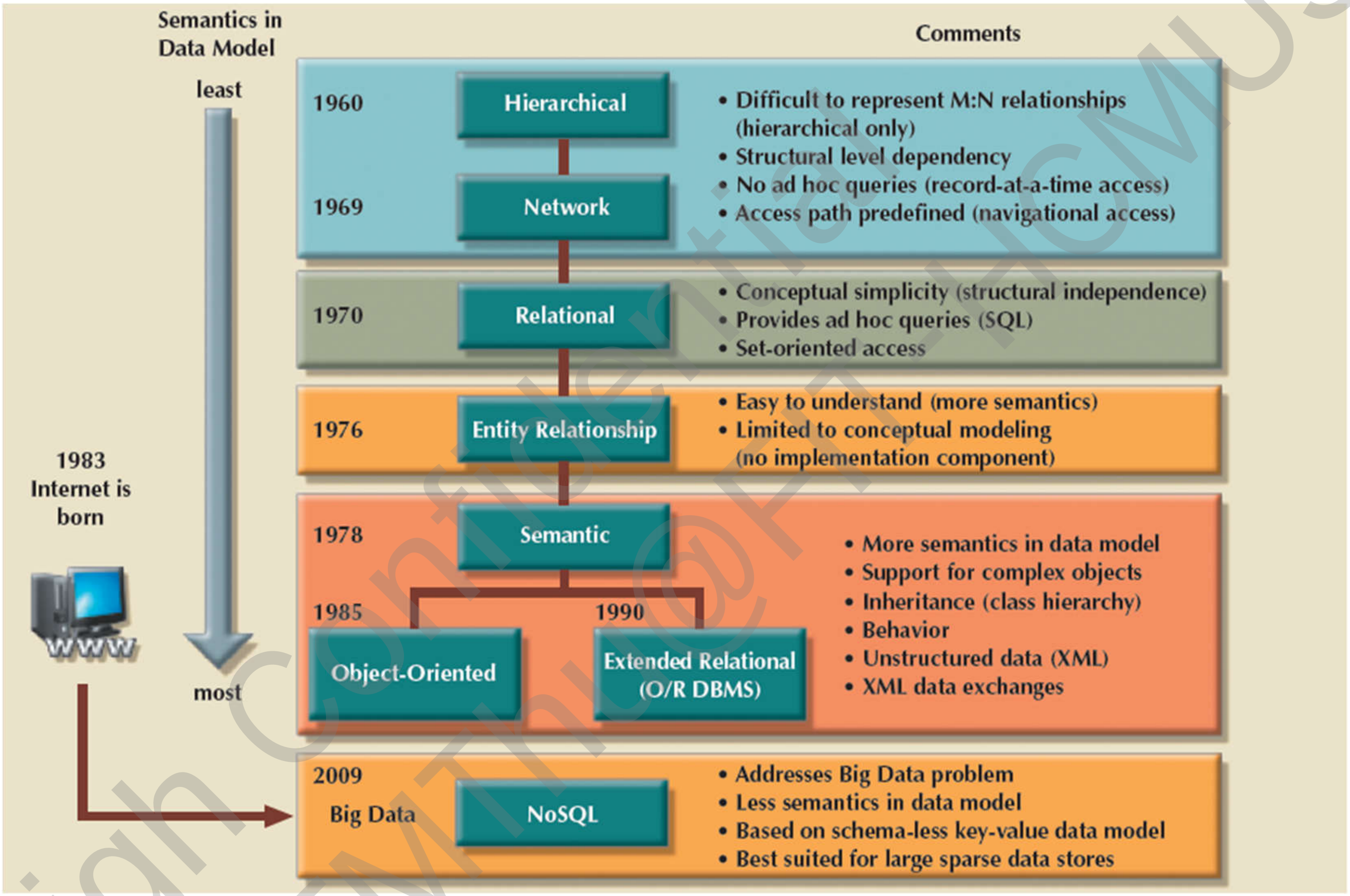
Data Model Basic Building Blocks

- **Entity**: Unique and distinct object used to collect and store data
  - **Attribute**: Characteristic of an entity
- **Relationship**: Describes an association among entities
  - **One-to-many (1:M)**
  - **Many-to-many (M:N or M:M)**
  - **One-to-one (1:1)**
- **Constraint**: Set of rules to ensure data integrity

Outline

- PART 1: Data Models
  - Data Modeling & Data Models
  - The Evolution of Data Models
  - Data Abstraction Levels
  - Business Rules

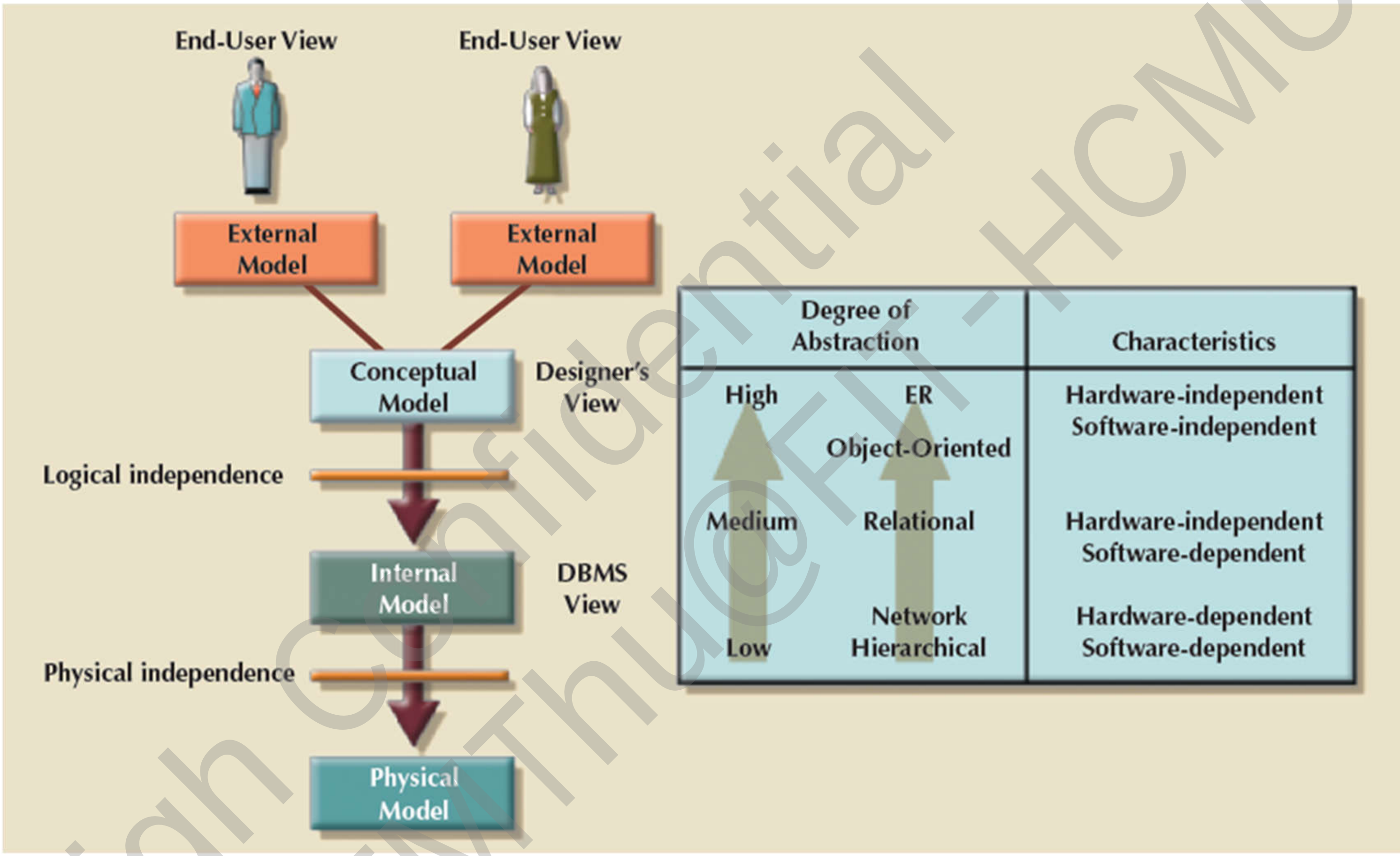
The Evolution of Data Models



Outline

- **PART 1: Data Models**
  - Data Modeling & Data Models
  - The Evolution of Data Models
  - Data Abstraction Levels
  - Business Rules

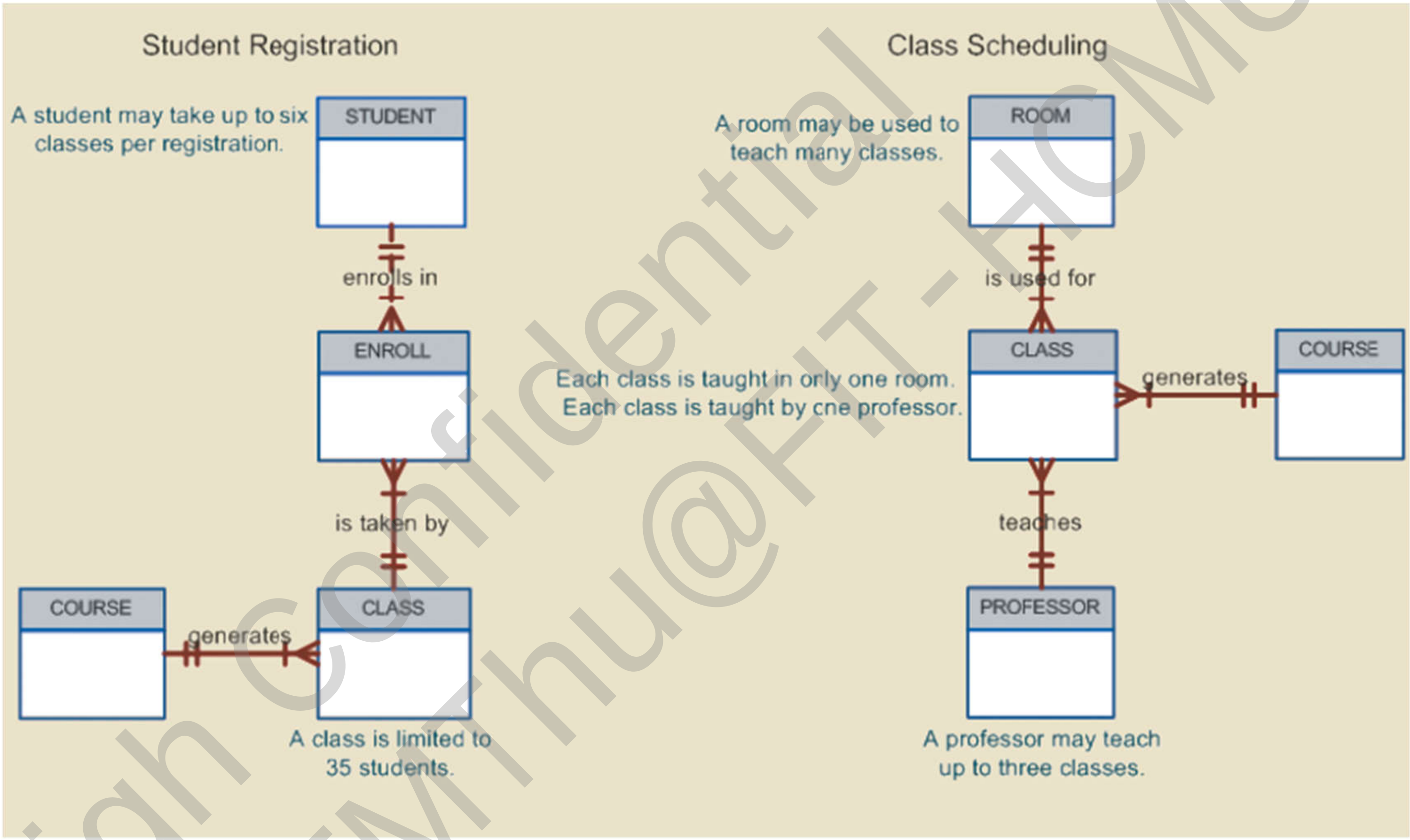
Data Abstraction Levels



The External Model

- End users' view of the data environment
- ER diagrams are used to represent the external views
- **External schema**: Specific representation of an external view

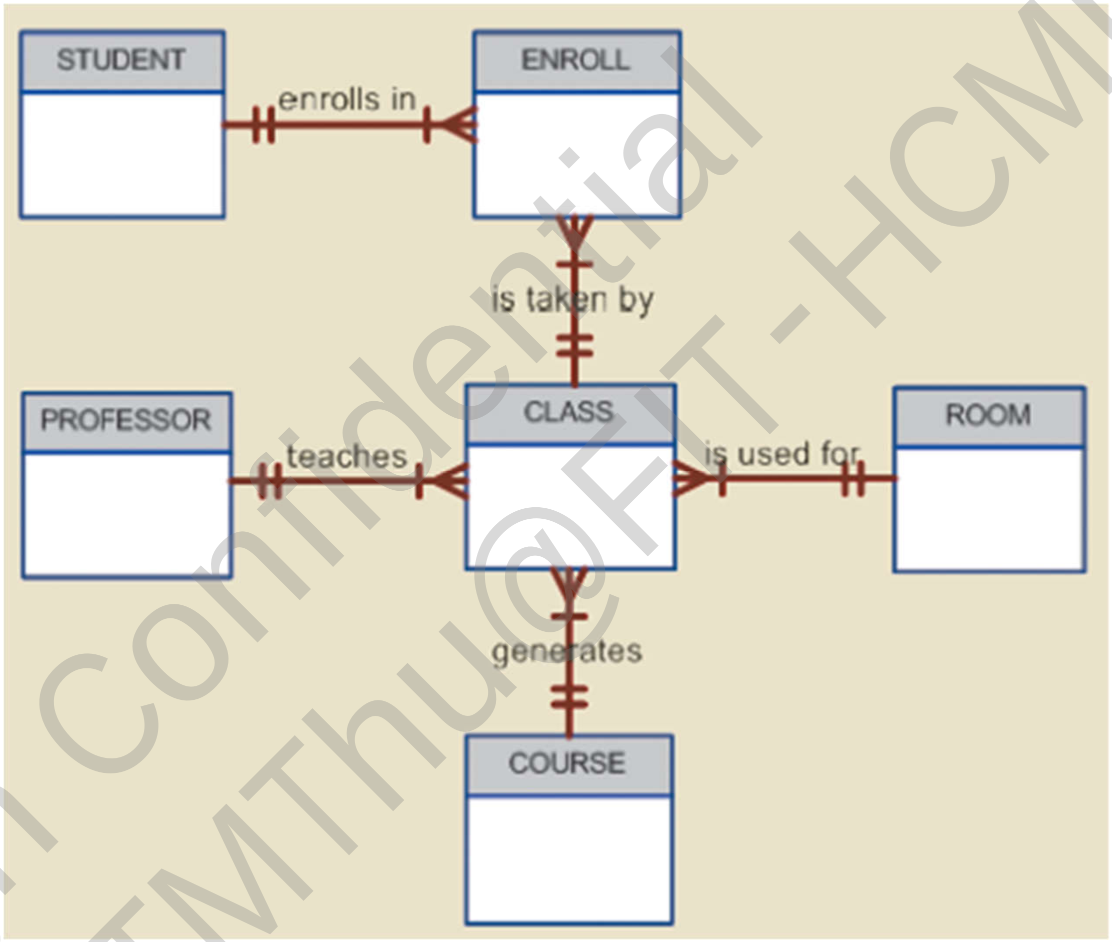
The External Model - Example



The Conceptual Model

- Represents a global view of the entire database by the entire organization
- **Conceptual schema**: Basis for the identification and high-level description of the main data objects
- Has a macro-level view of data environment
- Is software and hardware independent
- **Logical design**: Task of creating a conceptual data model

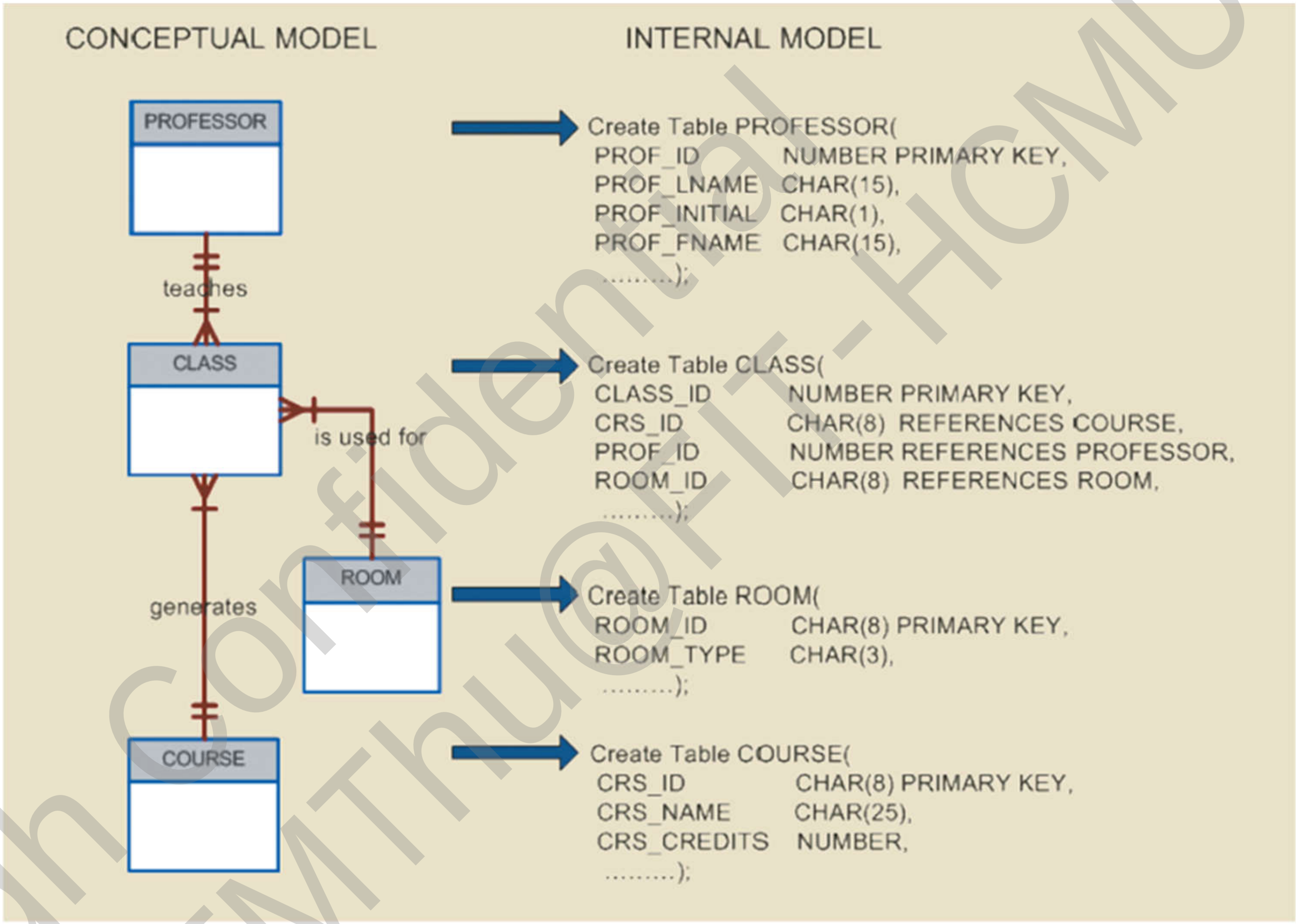
The Conceptual Model - Example



The Internal Model

- Representing database as seen by the DBMS mapping conceptual model to the DBMS
- **Internal schema**: Specific representation of an internal model
  - Uses the database constructs supported by the chosen database
- Is software dependent and hardware independent
- **Logical independence**: Changing internal model without affecting the conceptual model

The Internal Model - Example



The Physical Model

- Operates at lowest level of abstraction
- Describes the way data are saved on storage media such as disks or tapes
- Requires the definition of physical storage and data access methods
- Relational model aimed at logical level
  - Does not require physical-level details
- **Physical independence**: Changes in physical model do not affect internal model

Levels of Data Abstraction

LEVELS OF DATA ABSTRACTION			
MODEL	**DEGREE OF ABSTRACTION**	**FOCUS**	**INDEPENDENT OF**
External	High	End-user views	Hardware and software
Conceptual	↕	Global view of data (database model independent)	Hardware and software
Internal		Specific database model	Hardware
Physical	Low	Storage and access methods	Neither hardware nor software

Outline

- PART 1: Data Models
  - Data Modeling & Data Models
  - The Evolution of Data Models
  - Data Abstraction Levels
  - Business Rules

Business Rules

Brief, precise, and unambiguous description of a policy, procedure, or principle

Enable defining the basic building blocks

Describe main and distinguishing characteristics of the data

Sources of Business Rules

- Company managers
- Policy makers
- Department managers
- Written documentation
- Direct interviews with end users

Reasons for Identifying and Documenting Business Rules

- Help standardize company's view of data
- Communications tool between users and designers
- Allow designer to:
  - Understand the nature, role, scope of data, and business processes
  - Develop appropriate relationship participation rules and constraints
  - Create an accurate data model

Business Rules Example

A customer may make many payments on an account.

Each payment on an account is credited to only one customer

A customer may generate many invoices.

Each invoice is generated by only one customer

Business Rules Example

- *"A student may register for a section of a course only if he or she has successfully completed the prerequisites for that course."*
- *"A preferred customer qualifies for a 10 percent discount, unless he has an overdue account balance."*
- **"Friday is business casual dress day"** (**not business rules**)

Translating Business Rules into Data Model Components

- Nouns translate into entities
- Verbs translate into relationships among entities
- Relationships are bidirectional
- Questions to identify the relationship type
  - How many instances of B are related to one instance of A?
  - How many instances of A are related to one instance of B?

Naming Conventions

- Entity names - Required to:
  - Be descriptive of the objects in the business environment
  - Use terminology that is familiar to the users
- Attribute name - Required to be descriptive of the data represented by the attribute
- Proper naming:
  - Facilitates communication between parties
  - Promotes self-documentation

Outline

- **Part 2 - Relational Data Model**
  - Relational Model Concepts
  - Relational Model Constraints and Relational Database Schemas
  - Update Operations and Dealing with Constraint Violations

Outline

- **Part 2 - Relational Data Model**
  - Relational Model Concepts
  - Relational Model Constraints and Relational Database Schemas
  - Update Operations and Dealing with Constraint Violations

Relational Model Concepts

- The Relational Model of Data is based on the concept of a Relation.
- A Relation is a mathematical concept based on the ideas of sets.
- The strength of the relational approach to data management comes from the formal foundation provided by the theory of relations.
- We review the essentials of the relational approach in this part.

Relational Model Concepts

- The model was first proposed by Dr. E.F. Codd of IBM in 1970 in the following paper:
  "A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970.

  The above paper caused a major revolution in the field of Database management and earned Ted Codd the coveted ACM Turing Award.

Informal Definitions

- **RELATION**: A table of values
  - A relation may be thought of as a **set of rows**.
  - A relation may alternately be though of as a **set of columns**.
  - Each row represents a fact that corresponds to a real-world **entity** or **relationship**.
  - Each row has a value of an item or set of items that uniquely identifies that row in the table.
  - Sometimes row-ids or sequential numbers are assigned to identify the rows in the table.
  - Each column typically is called by its column name or column header or attribute name.

Example

Relation name

Attributes

Tuples

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25

Formal Definitions

- A **Relation** may be defined in multiple ways.
- The **Schema** of a Relation: *R* (A1, A2, ...An)

  Relation schema *R* is defined over **attributes** A1, A2, ... An

  For Example -

  CUSTOMER (Cust-id, Cust-name, Address, Phone#)

  Here, CUSTOMER is a relation defined over the four attributes Cust-id, Cust-name, Address, Phone#, each of which has a **domain** or a set of valid values. For example, the domain of Cust-id is 6 digit numbers.

Formal Definitions

- A **tuple** is an ordered set of values
- Each value is derived from an appropriate domain.
- Each row in the CUSTOMER table may be referred to as a tuple in the table and would consist of four values.
- **<632895, "John Smith", "101 Main St. Atlanta, GA 30332", "(404) 894-2000">** is a tuple belonging to the CUSTOMER relation.
- A relation may be regarded as a ***set of tuples*** (rows).
- Columns in a table are also called attributes of the relation.

Formal Definitions

- A **domain** has **a logical definition**: e.g., "USA_phone_numbers" are the set of 10 digit phone numbers valid in the U.S.
- A domain may have a data-type or a format defined for it. The USA_phone_numbers may have a format: (ddd)-ddd-dddd where each d is a decimal digit. E.g., Dates have various formats such as monthname, date, year or yyyy-mm-dd, or dd mm,yyyy etc.
- An attribute designates the **role** played by the domain. E.g., the domain Date may be used to define attributes "Invoice-date" and "Payment-date".

Formal Definitions

- The relation is formed over the cartesian product of the sets; each set has values from a domain; that domain is used in a specific role which is conveyed by the attribute name.
- For example, attribute Cust-name is defined over the domain of strings of 25 characters. The role these strings play in the CUSTOMER relation is that of the name of customers.
- Formally,

  Given $R(A_1, A_2, ..., A_n)$

  $r(R) \subset dom\ (A_1)\ X\ dom\ (A_2)\ X\ ....X\ dom(A_n)$
- R: schema of the relation
- r of R: a specific "value" or population of R.
- R is also called the **intension** of a relation
- r is also called the **extension** of a relation

Formal Definitions

- Let S1 = {0,1}
- Let S2 = {a,b,c}

- Let R $\subset$ S1 X S2

- Then for example: r(R) = {<0,a> , <0,b> , <1,c> } is one possible “state” or “population” or “extension” r of the relation R, defined over domains S1 and S2. It has three tuples.

Definition Summary

Informal Terms	Formal Terms
Table	Relation
Column	Attribute/Domain
Row	Tuple
Values in a column	Domain
Table Definition	Schema of a Relation
Populated Table	Extension

Characteristics of Relations

- **Ordering of tuples in a relation r(R)**: The tuples are *not* considered to be ordered, even though they appear to be in the tabular form.
- **Ordering of attributes in a relation schema R** (and of values within each tuple): We will consider the attributes in $R(A_1, A_2, ..., A_n)$ and the values in $t=<v_1, v_2, ..., v_n>$ to be *ordered* .

  (However, a more general *alternative definition* of relation does not require this ordering).
- **Values in a tuple**: All values are considered *atomic* (indivisible). A special **null** value is used to represent values that are unknown or inapplicable to certain tuples.

Characteristics of Relations

- <u>Notation:</u>

- We refer to **component values** of a tuple t by $t[A_i] = v_i$ (the value of attribute A_i for tuple t).

  Similarly, $t[A_u, A_v, ..., A_w]$ refers to the sub tuple of t containing the values of attributes $A_u, A_v, ..., A_w$, respectively.

Characteristics of Relations

Relation name

Attributes

Tuples

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Dick Davidson	422-11-2320	null	3452 Elgin Road	749-1253	25	3.53
	Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	null	19	3.25
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Katherine Ashly	381-62-1245	375-4409	125 Kirby Road	null	18	2.89
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21

Outline

- Part 2 - Relational Data Model
  - Relational Model Concepts
  - Relational Model Constraints and Relational Database Schemas
  - Update Operations and Dealing with Constraint Violations

Relational Integrity Constraints

- Constraints are *conditions* that must hold on *all* valid relation instances. There are three main types of constraints:
  1. **Key** constraints
  2. **Entity integrity** constraints
  3. **Referential integrity** constraints

Key Constraints

- **Superkey** of R: A set of attributes SK of R such that no two tuples *in any valid relation instance r(R)* will have the same value for SK. That is, for any distinct tuples t1 and t2 in r(R), t1[SK] ≠ t2[SK].
- **Key** of R: A "minimal" superkey; that is, a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey.

  Example: The CAR relation schema:

  CAR(State, Reg#, SerialNo, Make, Model, Year)

  has two keys Key1 = {State, Reg#}, Key2 = {SerialNo}, which are also superkeys. {SerialNo, Make} is a superkey but *not* a key.
- If a relation has *several* **candidate keys**, one is chosen arbitrarily to be the **primary key**. The primary key attributes are *underlined*.

Key Constraints

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

The CAR relation, with two candidate keys: License_number and Engine_serial_number

Entity Integrity

- **Relational Database Schema**: A set S of relation schemas that belong to the same database. S is the *name* of the **database**.

$$S = \{R_1, R_2, ..., R_n\}$$

- **Entity Integrity**: The *primary key attributes* PK of each relation schema R in S cannot have null values in any tuple of r(R). This is because primary key values are used to *identify* the individual tuples.

$$t[PK] \neq \text{null for any tuple } t \text{ in } r(R)$$

- Note: Other attributes of R may be similarly constrained to disallow null values, even though they are not members of the primary key.

Referential Integrity

- A constraint involving *two* relations (the previous constraints involve a *single* relation).
- Used to specify a *relationship* among tuples in two relations: the **referencing relation** and the **referenced relation**.
- Tuples in the *referencing relation* R_1 have attributes FK (called **foreign key** attributes) that reference the primary key attributes PK of the *referenced relation* R_2. A tuple t_1 in R_1 is said to **reference** a tuple t_2 in R_2 if $t_1[FK] = t_2[PK]$.
- A referential integrity constraint can be displayed in a relational database schema as a directed arc from R_1.FK to R_2.

Referential Integrity Constraint

Statement of the constraint

The value in the foreign key column (or columns) FK of the the **referencing relation** R_1 can be either:

(1) a value of an existing primary key value of the corresponding primary key PK in the **referenced relation** $R_{2,}$, or

(2) a null.

In case (2), the FK in R_1 should not be a part of its own primary key.

Other Types of Constraints

Semantic Integrity Constraints:

- based on application semantics and cannot be expressed by the model per se
- E.g., "the max. no. of hours per employee for all projects he or she works on is 56 hrs per week"
- A *constraint specification language* may have to be used to express these
- SQL-99 allows triggers and ASSERTIONS to allow for some of these

Schema diagram for the COMPANY relational database schema.

EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS_ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

One possible database state for the COMPANY relational database schema

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

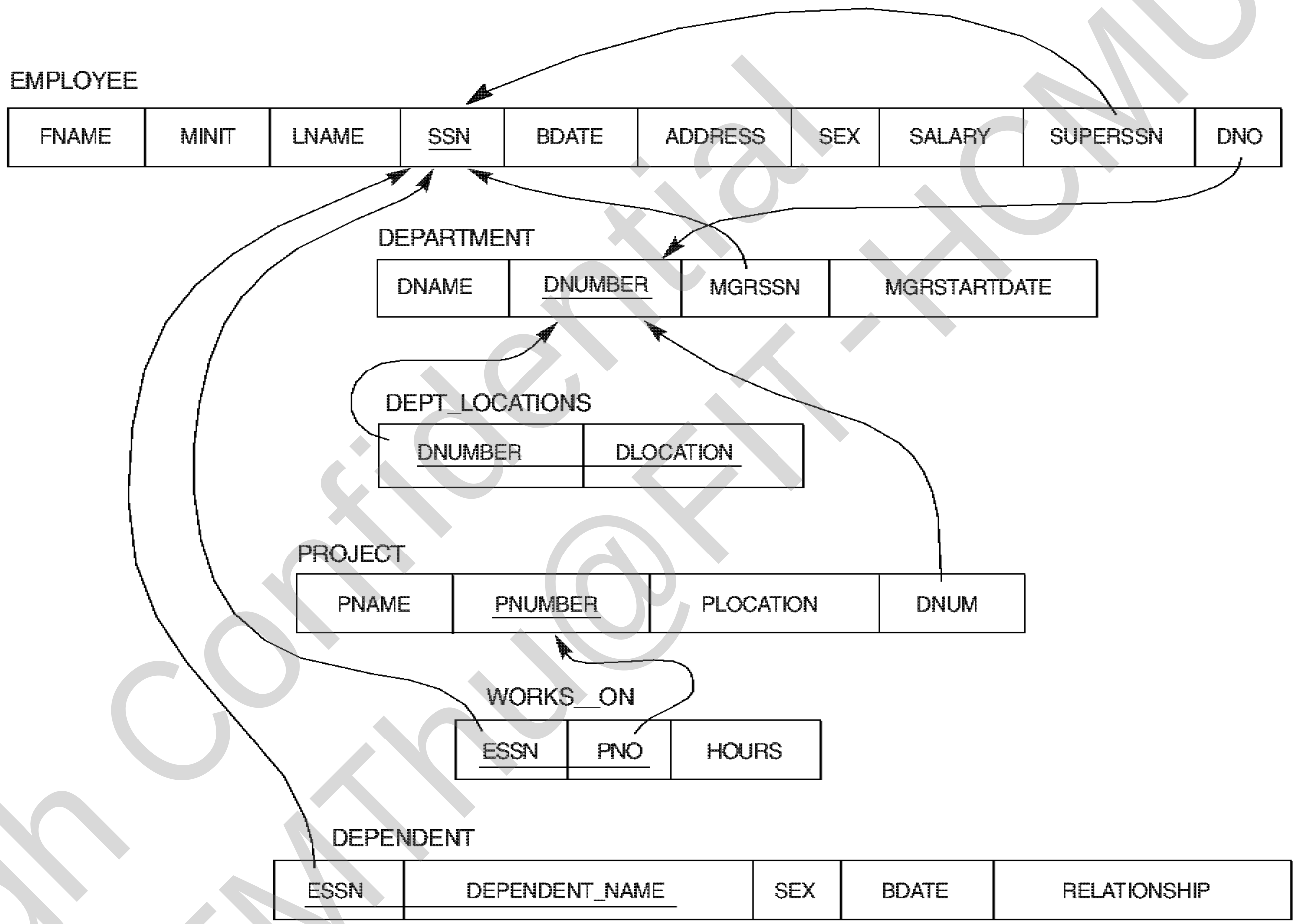
PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Referential integrity constraints displayed on the COMPANY relational database schema.



Outline

- Part 2 - Relational Data Model
  - Relational Model Concepts
  - Relational Model Constraints and Relational Database Schemas
  - Update Operations and Dealing with Constraint Violations

Update Operations on Relations

- INSERT a tuple.
- DELETE a tuple.
- MODIFY a tuple.

- Integrity constraints should not be violated by the update operations.
- Several update operations may have to be grouped together.
- Updates may *propagate* to cause other updates automatically. This may be necessary to maintain integrity constraints.

Update Operations on Relations

- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (REJECT option)
  - Perform the operation but inform the user of the violation
  - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  - Execute a user-specified error-correction routine

In-Class Exercise

Consider the following relations for a database that keeps track of student enrollment in courses and the books adopted for each course:

STUDENT(SSN, Name, Major, Bdate)

COURSE(Course#, Cname, Dept)

ENROLL(SSN, Course#, Quarter, Grade)

BOOK_ADOPTION(Course#, Quarter, Book_ISBN)

TEXT(Book_ISBN, Book_Title, Publisher, Author)

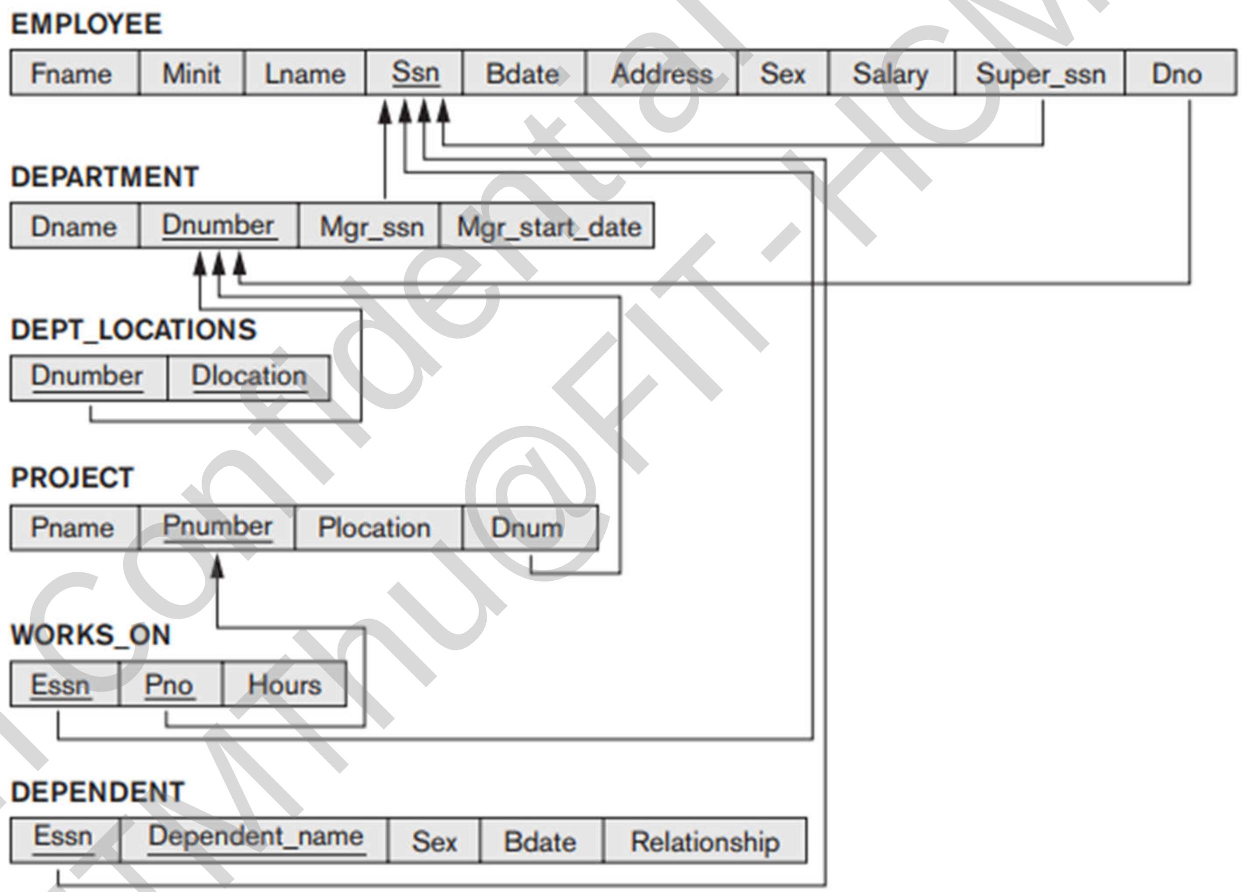
Draw a relational schema diagram specifying the foreign keys for this schema.

Outline

- **Part 3 - Relational Algebra**
  - Unary Relational Operations
  - Relational Algebra Operations From Set Theory
  - Binary Relational Operations
  - Additional Relational Operations

Database State for COMPANY

All examples discussed below refer to the COMPANY database shown here.



Database State for COMPANY

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Relational Algebra

- The basic set of operations for the relational model is known as the relational algebra. These operations enable a user to specify basic retrieval requests.
- The result of a retrieval is a new relation, which may have been formed from one or more relations. The **algebra operations** thus produce new relations, which can be further manipulated using operations of the same algebra.
- A sequence of relational algebra operations forms a **relational algebra expression**, whose result will also be a relation that represents the result of a database query (or retrieval request).

Outline

❑ Part 3 - Relational Algebra

- ❑ Unary Relational Operations
- ❑ Relational Algebra Operations From Set Theory
- ❑ Binary Relational Operations
- ❑ Additional Relational Operations

Unary Relational Operations

- **SELECT Operation (σ)**

SELECT operation is used to select a *subset* of the tuples from a relation that satisfy a **selection condition**. It is a filter that keeps only those tuples that satisfy a qualifying condition – those satisfying the condition are selected while others are discarded.

Example: To select the EMPLOYEE tuples whose department number is four or those whose salary is greater than $30,000 the following notation is used:

$$\sigma_{DNO = 4}\ (\text{EMPLOYEE})$$

$$\sigma_{SALARY > 30,000}\ (\text{EMPLOYEE})$$

In general, the select operation is denoted by $\sigma_{\text{<selection condition>}}(R)$ where the symbol **σ** (sigma) is used to denote the select operator, and the selection condition is a Boolean expression specified on the attributes of relation R

Unary Relational Operations

SELECT Operation Properties

- The SELECT operation $\sigma_{\text{<selection condition>}}(R)$ produces a relation S that has the same schema as R
- The SELECT operation σ is **commutative;** i.e.,
  $\sigma_{\text{<condition1>}}(\sigma_{\text{< condition2>}}(R)) = \sigma_{\text{<condition2>}}(\sigma_{\text{< condition1>}}(R))$
- A cascaded SELECT operation **may be applied in any order;** i.e.,
  $\sigma_{\text{<condition1>}}(\sigma_{\text{< condition2>}}(\sigma_{\text{<condition3>}}(R))$
  $= \sigma_{\text{<condition2>}}(\sigma_{\text{< condition3>}}(\sigma_{\text{< condition1>}}(R)))$
- A cascaded SELECT operation may be replaced by a single selection with a conjunction of all the conditions; i.e.,
  $\sigma_{\text{<condition1>}}(\sigma_{\text{< condition2>}}(\sigma_{\text{<condition3>}}(R))$
  $= \sigma_{\text{<condition1>} \wedge \text{< condition2>} \wedge \text{< condition3>}}(R)))$

Unary Relational Operations (cont.)

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

$\sigma_{(\text{Dno} = 5 \wedge \text{SALARY} > 30000) \vee (\text{Dno} = 4 \wedge \text{SALARY} > 25000}$ **(EMPLOYEE)**

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5

Unary Relational Operations (cont.)

- **PROJECT Operation (π)**

This operation selects certain *columns* from the table and discards the other columns. The PROJECT creates a vertical partitioning – one with the needed columns (attributes) containing results of the operation and other containing the discarded Columns.

Example: To list each employee's first and last name and salary, the following is used:

$$\pi_{\text{LNAME, FNAME,SALARY}}(\text{EMPLOYEE})$$

The general form of the project operation is π<attribute list>(R) where π (pi) is the symbol used to represent the project operation and <attribute list> is the desired list of attributes from the attributes of relation R.

The project operation *removes any duplicate tuples,* so the result of the project operation is a set of tuples and hence a valid relation.

Unary Relational Operations (cont.)

PROJECT Operation Properties

- The number of tuples in the result of projection $\pi_{\text{<list>}}(R)$ is always less or equal to the number of tuples in R.

- If the list of attributes includes a key of R, then the number of tuples is equal to the number of tuples in R.

- $\pi_{\text{<list1>}}(\pi_{\text{<list2>}}(R)) = \pi_{\text{<list1>}}(R)$ as long as <list2> contains the attributes in <list1>

Unary Relational Operations (cont.)

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Lname	Fname	Salary
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

$\pi_{\text{Lname, Fname,Salary}}$(EMPLOYEE)

Sex	Salary
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

$\pi_{\text{Sex, Salary}}$(EMPLOYEE)

Unary Relational Operations (cont.)

- **Rename Operation**

We may want to apply several relational algebra operations one after the other. Either we can write the operations as a single **relational algebra expression** by nesting the operations, or we can apply one operation at a time and create **intermediate result relations**. In the latter case, we must give names to the relations that hold the intermediate results.

Example: To retrieve the first name, last name, and salary of all employees who work in department number 5, we must apply a select and a project operation. We can write a single relational algebra expression as follows:

$\pi_{\text{FNAME, LNAME, SALARY}}(\sigma_{\text{DNO=5}}(\text{EMPLOYEE}))$

OR We can explicitly show the sequence of operations, giving a name to each intermediate relation:

$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{DNO=5}}(\text{EMPLOYEE})$

$\text{RESULT} \leftarrow \pi_{\text{FNAME, LNAME, SALARY}}(\text{DEP5\_EMPS})$

Unary Relational Operations (cont.)

- **Rename Operation (cont.)**

  The rename operator is ρ

  The general Rename operation can be expressed by any of the following forms:

- $\rho_{S(B_1, B_2, \ldots, B_n)}(R)$ is a renamed relation S based on R with column names $B_1, B_1, ..B_n$.
- $\rho_S(R)$ is a renamed relation S based on R (which does not specify column names).
- $\rho_{(B_1, B_2, .., B_n)}(R)$ is a renamed relation with column names $B_1, B_1, \ldots B_n$ which does not specify a new relation name.

Unary Relational Operations (cont.)

Rename Operation (cont.)

$$\text{TEMP} \leftarrow \sigma_{\text{Dno}=5}(\text{EMPLOYEE})$$
$$R(\text{First\_name, Last\_name, Salary}) \leftarrow \pi_{\text{Fname, Lname, Salary}}(\text{TEMP})$$

TEMP

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston,TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston,TX	M	40000	888665555	5
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble,TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

R

First_name	Last_name	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

Outline

- Part 3 - Relational Algebra
  - Unary Relational Operations
  - Relational Algebra Operations From Set Theory
  - Binary Relational Operations
  - Additional Relational Operations

Relational Algebra Operations From Set Theory

- UNION: The result of this operation, denoted by *R* ∪ *S*, is a relation that includes all tuples that are either in *R* or in *S* or in both *R* and *S*. Duplicate tuples are eliminated.
- INTERSECTION: The result of this operation, denoted by R ∩ S, is a relation that includes all tuples that are in both R and S.
- SET DIFFERENCE (or MINUS): The result of this operation, denoted by R – S, is a relation that includes all tuples that are in R but not in S.
- The resulting relation for $R_1 \cup R_2$, $R_1 \cap R_2$, or $R_1 - R_2$ has the same attribute names as the *first* operand relation R1 (by convention).
- The two operands must be "type compatible"

Relational Algebra Operations From Set Theory (cont.)

- **Type Compatibility**
  - The operand relations $R_1(A_1, A_2, ..., A_n)$ and $R_2(B_1, B_2, ..., B_n)$ must have:
    - the same number of attributes, and
    - the domains of corresponding attributes must be compatible; that is, $dom(A_i)=dom(B_i)$ for $i=1, 2, ..., n$.

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

Relational Algebra Operations From Set Theory

- **UNION Operation**

The result of this operation, denoted by R ∪ S, is a relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated. The two operands must be "type compatible"

- **UNION Example**

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

STUDENT ∪ INSTRUCTOR

Relational Algebra Operations From Set Theory (cont.)

Example: To retrieve the social security numbers of all employees who either work in department 5 or directly supervise an employee who works in department 5, we can use the union operation as follows:

DEP5_EMPS $\leftarrow \sigma_{DNO=5}$ **(EMPLOYEE)**

RESULT1 $\leftarrow \pi_{SSN}$**(DEP5_EMPS)**

RESULT2(SSN) $\leftarrow \pi_{SUPERSSN}$**(DEP5_EMPS)**

RESULT $\leftarrow$ **RESULT1** $\cup$ **RESULT2**

The union operation produces the tuples that are in either RESULT1 or RESULT2 or both. The two operands must be “type compatible”.

Relational Algebra Operations From Set Theory (cont.)

- **INTERSECTION OPERATION**

The result of this operation, denoted by R ∩ S, is a relation that includes all tuples that are in both R and S. The two operands must be "type compatible"

Example: The result of the intersection operation (figure below) includes only those who are both students and instructors.

FN	LN
Susan	Yao
Ramesh	Shah

FN	LN
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

STUDENT ∩ INSTRUCTOR

Relational Algebra Operations From Set Theory (cont.)

- **Set Difference (or MINUS) Operation**

The result of this operation, denoted by R - S, is a relation that includes all tuples that are in R but not in S. The two operands must be "type compatible".

Example: The figure shows the names of students who are not instructors, and the names of instructors who are not students.

STUDENT	FN	LN
	Susan	Yao
	Ramesh	Shah
	Johnny	Kohler
	Barbara	Jones
	Amy	Ford
	Jimmy	Wang
	Ernest	Gilbert

FN	LN
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

STUDENT - INSTRUCTOR

FNAME	LNAME
John	Smith
Ricardo	Browne
Francis	Johnson

INSTRUCTOR - STUDENT

Relational Algebra Operations From Set Theory (cont.)

- Notice that both union and intersection are *commutative operations;* that is

  $$\mathbf{R \cup S = S \cup R, \text{ and } R \cap S = S \cap R}$$

- Both union and intersection can be treated as n-ary operations applicable to any number of relations as both are *associative operations;* that is

  $$\mathbf{R \cup (S \cup T) = (R \cup S) \cup T, \text{ and } (R \cap S) \cap T = R \cap (S \cap T)}$$

- The minus operation is *not commutative;* that is, in general

  $$\mathbf{R - S \neq S - R}$$

Relational Algebra Operations From Set Theory (cont.)

- **CARTESIAN (or cross product) Operation**
  - This operation is used to combine tuples from two relations in a combinatorial fashion.
  - In general, the result of $R(A_1, A_2, . . ., A_n) \times S(B_1, B_2, . . ., B_m)$ is a relation Q with degree n + m attributes $Q(A_1, A_2, . . ., A_n, B_1, B_2, . . ., B_m)$, in that order.
  - The resulting relation Q has one tuple for each combination of tuples—one from R and one from S. Hence, if R has n_R tuples (denoted as $|R| = n_R$), and S has n_S tuples, then | R x S | will have $n_R * n_S$ tuples.
  - The two operands do NOT have to be "type compatible”

Relational Algebra Operations From Set Theory (cont.)

- Cartesian Operation

r	A	B
	α	1
	β	2

s	X	C	D
	α	10	+
	β	10	+
	β	20	-
	γ	10	-

$\rho_{(X,C,D)}(s)$

unambiguous

$r \times s$	A	R.B	X	C	D
	α	1	α	10	+
	α	1	β	10	+
	α	1	β	20	-
	α	1	γ	10	-
	β	2	α	10	+
	β	2	β	10	+
	β	2	β	20	-
	β	2	γ	10	-

Relational Algebra Operations From Set Theory (cont.)

For example, suppose that we want to retrieve a list of names of each female employee's dependents

FEMALE_EMPS $\leftarrow \sigma_{Sex='F'}$**(EMPLOYEE)**

EMPNAMES $\leftarrow \pi_{Fname,\ Lname,\ Ssn}$**(FEMALE_EMPS)**

EMP_DEPENDENTS $\leftarrow$ **EMPNAMES × DEPENDENT**

ACTUAL_DEPENDENTS $\leftarrow \sigma_{Ssn=Essn}$**(EMP_DEPENDENTS)**

RESULT $\leftarrow \pi_{Fname,\ Lname,\ Dependent\_name}$**(ACTUAL_DEPENDENTS)**

FEMALE_EMPS

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Alicia	J	Zelaya	999887777	1968-07-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5

EMPNAMES

Fname	Lname	Ssn
Alicia	Zelaya	999887777
Jennifer	Wallace	987654321
Joyce	English	453453453

ACTUAL_DEPENDENTS

Fname	Lname	Ssn	Essn	Dependent_name	Sex	Bdate	...
Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	...

RESULT

Fname	Lname	Dependent_name
Jennifer	Wallace	Abner

EMP_DEPENDENTS

Fname	Lname	Ssn	Essn	Dependent_name	Sex	Bdate	...
Alicia	Zelaya	999887777	333445555	Alice	F	1986-04-05	...
Alicia	Zelaya	999887777	333445555	Theodore	M	1983-10-25	...
Alicia	Zelaya	999887777	333445555	Joy	F	1958-05-03	...
Alicia	Zelaya	999887777	987654321	Abner	M	1942-02-28	...
Alicia	Zelaya	999887777	123456789	Michael	M	1988-01-04	...
Alicia	Zelaya	999887777	123456789	Alice	F	1988-12-30	...
Alicia	Zelaya	999887777	123456789	Elizabeth	F	1967-05-05	...
Jennifer	Wallace	987654321	333445555	Alice	F	1986-04-05	...
Jennifer	Wallace	987654321	333445555	Theodore	M	1983-10-25	...
Jennifer	Wallace	987654321	333445555	Joy	F	1958-05-03	...
Jennifer	Wallace	987654321	987654321	Abner	M	1942-02-28	...
Jennifer	Wallace	987654321	123456789	Michael	M	1988-01-04	...
Jennifer	Wallace	987654321	123456789	Alice	F	1988-12-30	...
Jennifer	Wallace	987654321	123456789	Elizabeth	F	1967-05-05	...
Joyce	English	453453453	333445555	Alice	F	1986-04-05	...
Joyce	English	453453453	333445555	Theodore	M	1983-10-25	...
Joyce	English	453453453	333445555	Joy	F	1958-05-03	...
Joyce	English	453453453	987654321	Abner	M	1942-02-28	...
Joyce	English	453453453	123456789	Michael	M	1988-01-04	...
Joyce	English	453453453	123456789	Alice	F	1988-12-30	...
Joyce	English	453453453	123456789	Elizabeth	F	1967-05-05	...

Outline

- **Part 3 - Relational Algebra**
  - Unary Relational Operations
  - Relational Algebra Operations From Set Theory
  - Binary Relational Operations
  - Additional Relational Operations

Binary Relational Operations

- **JOIN Operation**
  - The sequence of cartesian product followed by select is used quite commonly to identify and select related tuples from two relations, a special operation, called **JOIN**. It is denoted by a $\bowtie$
  - This operation is very important for any relational database with more than a single relation, because it allows us to process relationships among relations.
  - The general form of a join operation on two relations $R(A_1, A_2, \ldots, A_n)$ and $S(B_1, B_2, \ldots, B_m)$ is:

$$R \bowtie_{\text{<join condition>}} S$$

where R and S can be any relations that result from general *relational algebra expressions.*

Binary Relational Operations (cont.)

- **EQUIJOIN Operation**

  The most common use of join involves join conditions with equality comparisons only. Such a join, where the only comparison operator used is =, is called an EQUIJOIN. In the result of an EQUIJOIN we always have one or more pairs of attributes (whose names need not be identical) that have *identical values* in every tuple.

  The JOIN seen in the previous example was EQUIJOIN.

- **NATURAL JOIN Operation**

  Because one of each pair of attributes with identical values is superfluous, a new operation called natural join—denoted by *—was created to get rid of the second (superfluous) attribute in an EQUIJOIN condition.

  The standard definition of natural join requires that the two join attributes, or each pair of corresponding join attributes, have the **same name** in both relations. If this is not the case, a renaming operation is applied first.

Binary Relational Operations

- JOIN Operation(theta join)

r	A	B	C
	1	2	3
	4	5	6
	7	8	9

s	D	E
	3	1
	6	2

$r \bowtie_{B<D} s$

A	B	C	D	E
1	2	3	3	1
1	2	3	6	2
4	5	6	6	2

$$r \bowtie_C s = \sigma_C(r \times s)$$

Binary Relational Operations

- JOIN Operation(equijoin)

r	A	B	C
	1	2	3
	4	5	6
	7	8	9

s	D	E
	3	1
	6	2

$r \bowtie_{C=D} s$

A	B	C	D	E
1	2	3	3	1
4	5	6	6	2

r	A	B	C
	1	2	3
	4	5	6
	7	8	9

S	S.C	D
	3	1
	6	2

$\rho_{(S.C,D)}\ s$

$r \bowtie_{C=s.C} s$

A	B	C	s.C	D
1	2	3	3	1
4	5	6	6	2

Binary Relational Operations

- JOIN Operation (Natural JOIN)

r	A	B	C
	1	2	3
	4	5	6
	7	8	9

s	C	D
	3	1
	6	2

r * s

r ⋈ s

A	B	C	~~S.C~~	D
1	2	3	~~3~~	1
4	5	6	~~6~~	2

A	B	C	D
1	2	3	1
4	5	6	2

Binary Relational Operations (cont.)

Example: Suppose that we want to retrieve the name of the manager of each department. To get the manager's name, we need to combine each DEPARTMENT tuple with the EMPLOYEE tuple whose SSN value matches the MGRSSN value in the department tuple. We do this by using the join $\bowtie$ operation.

$$\textbf{DEPT\_MGR} \leftarrow \textbf{DEPARTMENT} \bowtie_{\textbf{MGRSSN=SSN}} \textbf{EMPLOYEE}$$

DEPT_MGR

Dname	Dnumber	Mgr_ssn	...	Fname	Minit	Lname	Ssn	...
Research	5	333445555	...	Franklin	T	Wong	333445555	...
Administration	4	987654321	...	Jennifer	S	Wallace	987654321	...
Headquarters	1	888665555	...	James	E	Borg	888665555	...

Binary Relational Operations (cont.)

Example: To apply a natural join on the DNUMBER attributes of DEPARTMENT and DEPT_LOCATIONS, it is sufficient to write:

DEPT_LOCS ← DEPARTMENT * DEPT_LOCATIONS

DEPT_LOCS

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Location
Headquarters	1	888665555	1981-06-19	Houston
Administration	4	987654321	1995-01-01	Stafford
Research	5	333445555	1988-05-22	Bellaire
Research	5	333445555	1988-05-22	Sugarland
Research	5	333445555	1988-05-22	Houston

Complete Set of Relational Operations

- The set of operations including **select σ, project π , union $\cup$, set difference - , and cartesian product X** is called a complete set because any other relational algebra expression can be expressed by a combination of these five operations.

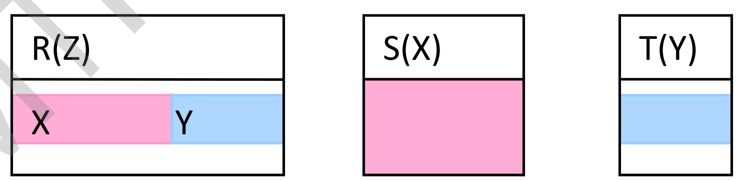
- For example:

  $\mathbf{R \cap S = (R \cup S) - ((R - S) \cup (S - R))}$

  $R \bowtie_{\text{<join condition>}} S = \boldsymbol{\sigma}_{\text{<join condition>}} \mathbf{(R\ X\ S)}$

Binary Relational Operations (cont.)

- **DIVISION Operation**
  - The division operation is applied to two relations $R(Z) \div S(X)$, where X subset Z.
  - Let $Y = Z - X$ (and hence $Z = X \cup Y$); that is, let Y be the set of attributes of R that are not attributes of S.
  - The result of DIVISION is a relation T(Y) that includes a tuple t if tuples t_R appear in R with $t_R [Y] = t$, and with $t_R [X] = t_s$ *for every tuple* t_s in S.
  - For a tuple t to appear in the result T of the DIVISION, the values in t must appear in R in combination with *every* tuple in S.



Binary Relational Operations (cont.)

- **DIVISION Operation**

r	A	B	C	D	E
	α	a	α	a	1
	α	a	γ	a	1
	α	a	γ	b	1
	β	a	γ	a	1
	β	a	γ	b	3
	γ	a	γ	a	1
	γ	a	γ	b	1
	γ	a	β	b	1

s	D	E
	a	1
	b	1

$r \div s$

A	B	C
α	a	γ
γ	a	γ

Binary Relational Operations (cont.)

Example: Retrieve the names of employees who work on all the projects

$R \leftarrow \pi_{Essn,\ Pno}(WORKS\_ON)$

$S \leftarrow \pi_{Pnumber}(PROJECT)$

$Result \leftarrow R \div S$

Binary Relational Operations (cont.)

Example: Retrieve the names of employees who work on all the projects that 'John Smith' works on.

$$SMITH \leftarrow \sigma_{Fname='John' \text{ AND } Lname='Smith'}(EMPLOYEE)$$

$$SMITH\_PNOS \leftarrow \pi_{Pno}(WORKS\_ON \; Essn=SsnSMITH)$$

$$SSN\_PNOS \leftarrow \pi_{Essn,\ Pno}(WORKS\_ON)$$

$$SSNS(Ssn) \leftarrow SSN\_PNOS \div SMITH\_PNOS$$

$$RESULT \leftarrow \pi_{Fname,\ Lname}(SSNS * EMPLOYEE)$$

Binary Relational Operations (cont.)

- The DIVISION operation can be expressed as a sequence of π, $\times$, and – operations as follows:

$$T1 \leftarrow \pi_Y(R)$$
$$T2 \leftarrow \pi_Y((S \times T1) - R)$$
$$T \leftarrow T1 - T2$$

Recap of Relational Algebra Operations

OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{\text{<selection condition>}}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{\text{<attribute list>}}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\text{<join condition>}} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{(\text{<join attributes 1>}),(\text{<join attributes 2>})} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{(\text{<join attributes 1>}),(\text{<join attributes 2>})} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

Outline

- **Part 3 - Relational Algebra**
  - Unary Relational Operations
  - Relational Algebra Operations From Set Theory
  - Binary Relational Operations
  - Additional Relational Operations

Additional Relational Operations

- **Generalized Projection**

$$\pi_{F1,\ F2,\ \ldots,\ Fn}\,(R)$$

where *F*1, *F*2, … , *Fn* are functions over the attributes in relation *R* and may involve arithmetic operations and constant values

As an example, consider the relation

EMPLOYEE (Ssn, Salary, Deduction, Years_service)

A report may be required to show

Net Salary = Salary – Deduction,

*Bonus = 2000 * Years_service, and*

*Tax = 0.25 * Salary*

Then a generalized projection combined with renaming may be used as follows:

$$\textbf{REPORT} \leftarrow \rho_{\text{(Ssn, Net\_salary, Bonus, Tax)}}(\pi_{\text{Ssn, Salary – Deduction, 2000 * Years\_service, 0.25 * Salary}}(\textbf{EMPLOYEE}))$$

Additional Relational Operations

- **Aggregate Functions and Grouping**
  - A type of request that cannot be expressed in the basic relational algebra is to specify mathematical **aggregate functions** on collections of values from the database.
  - Examples of such functions include retrieving the average or total salary of all employees or the total number of employee tuples. These functions are used in simple statistical queries that summarize information from the database tuples.
  - Common functions applied to collections of numeric values include SUM, AVERAGE, MAXIMUM, and MINIMUM. The COUNT function is used for counting tuples or values.

Additional Relational Operations

- Aggregate Functions and Grouping

r	A	B
	1	2
	3	4
	1	2
	1	2

SUM(B) = 10
AVG(A) = 1.5
MIN(A) = 1
MAX(B) = 4
COUNT(A) = 4

Additional Relational Operations

- **Aggregate Functions and Grouping**

$$_{\langle\text{grouping attributes}\rangle}\mathfrak{I}_{\langle\text{function list}\rangle}(R)$$

where *<grouping attributes>* is a list of attributes of the relation specified in *R*, and *<function list>* is a list of (<function> <attribute>) pairs

In each such pair, <function> is one of the allowed functions—such as **SUM, AVERAGE, MAXIMUM, MINIMUM, COUNT**—and <attribute> is an attribute of the relation specified by *R*

r	A	B	C
	α	2	7
	α	4	7
	β	2	3
	γ	2	10

$\mathfrak{I}_{SUM(C)}(r)$

SUM_C
27

$_{A}\mathfrak{I}_{SUM(C)}(r)$

A	SUM_C
α	14
β	3
γ	10

Additional Relational Operations

The aggregate function operation.

a. $\rho_{R(\text{Dno, No\_of\_employees, Average\_sal})}(_{\text{Dno}} \Im_{\text{COUNT Ssn, AVERAGE Salary}} (\text{EMPLOYEE}))$.

b. $_{\text{Dno}} \Im_{\text{COUNT Ssn, AVERAGE Salary}}(\text{EMPLOYEE})$.

c. $\Im_{\text{COUNT Ssn, AVERAGE Salary}}(\text{EMPLOYEE})$.

(a) R

Dno	No_of_employees	Average_sal
5	4	33250
4	3	31000
1	1	55000

(b)

Dno	Count_ssn	Average_salary
5	4	33250
4	3	31000
1	1	55000

(c)

Count_ssn	Average_salary
8	35125