Databases 2022

Darko Bozhinoski,
Ph.D. in Computer Science

Agenda

- Announcements
- Relational Model (Recap)
- Relational Model Operations
- Relational Algebra
- Relational Database Design by ER-to-Relational Mapping

Announcements

Mid-term exam: 08 April;
Office-hours: write to your instructors on telegram to organize a session.
Response time: 3-5 days.

Reading Materials:

Fundamentals of Database Systems. Ramez Elmasri and Shamkant B. Navathe. Pearson.

Lecture 1: Chapter 1, Chapter 2, Chapter 3
Lecture 2: Chapter 3, Chapter 4
Lecture 3: Chapter 4, Chapter 5
Lecture 4: Chapter 5, Chapter 8, Chapter 9

Relational Model Recap

Relational Model Concepts

- In the relational model, all data must be stored in relations (tables)
  - It uses the concept of a mathematical relation

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

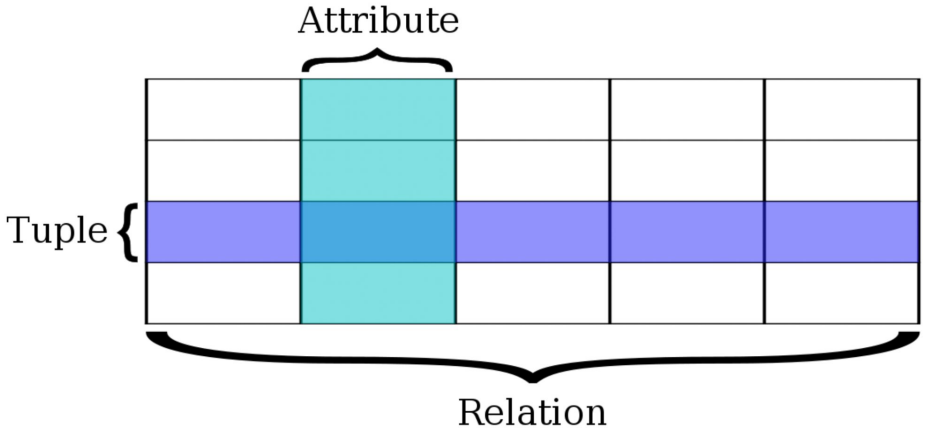
PROJECT

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

Relational Model Concepts (2)

- Each relation consists of rows and columns.
  - ➢ Each row in the table represents a collection of related data values.

- **Relation schema** *R*, denoted by *R*(*A*1, *A*2, ... , *An*), is made up of a **relation name** *R* and a list of attributes, ***A*1, *A*2, ... , *An***. Each **attribute** *Ai* is the name of a role played by some **domain *D*** in the relation schema *R*.



Relational Model Constraints

- Inherent model-based constraints (implicit constraints):
  - constraints that are inherent in the definition/assumptions of a particular data model hold in every database having that data model as its underpinning (e.g., a relation cannot have duplicate tuples)
- Schema-based constraints (explicit constraints)
  - expressed in the schemas of the data-model
- Application-based constraints(business rules):
  - enforced by the application programs

Semantic Integrity Constraints

Semantic Integrity Constraints: application-specific restrictions that are unlikely to be expressible in DDL

- Mechanisms called triggers and assertions can be used in SQL, to specify some of these constraints. It is more common to verify these types of constraints within the application programs.

State vs transition constraints:

- **State constraints:** Define the constraints that a valid state of the database must satisfy.
- **Transition constraints:** deal with state changes in the database

Relational Model Operations

Operations

- Operations on the relational model: **retrievals** and **updates**.
  - Update operations.Three basic operations that can change the states of relations in the database: **Insert**, **Delete**, and **Update**
    - Insert is used to insert one or more new tuples in a relation;
    - Delete is used to delete tuples
    - Update is used to change the values of some attributes in existing tuples.
  - Whenever these operations are applied, the **integrity constraints specified on the relational database schema should not be violated**.

Dealing with Constraint Violations through INSERT

INSERT Constraint violations:

- **domain constraint violation:** some attribute value is not of correct domain
- **entity integrity violation:** some attribute within a key of the new tuple has "value" **null**
- **key constraint violation:** key of new tuple is same as key of already-existing tuple
- **referential integrity violation:** foreign key of new tuple refers to non-existent tuple

Dealing with Constraint Violations through INSERT (2)

INSERT Constraint violations:

- **domain constraint violation:** some attribute value is not of correct domain
- **entity integrity violation:** some attribute within a key of the new tuple has "value" **null**
- **key constraint violation:** key of new tuple is same as key of already-existing tuple
- **referential integrity violation:** foreign key of new tuple refers to non-existent tuple

Solution: Reject the attempt to insert!

Dealing with Constraint Violations through DELETE

DELETE Constraint violations:

- **referential integrity violation:** a tuple referring to the deleted one exists.

Dealing with Constraint Violations through DELETE (2)

DELETE Constraint violations:

- **referential integrity violation:** a tuple referring to the deleted one exists.

SOLUTION:

- Reject the deletion
- Attempt to **cascade** (or **propagate**) by deleting any referencing tuples (plus those that reference them, etc., etc.)
- modify the foreign key attribute values in referencing tuples to **null** or to some valid value referencing a different tuple

Dealing with Constraint Violations through UPDATE

UPDATE Constraint violations:

- **Key constraint violation: primary key is changed so as to become same as another tuple**
- **Referential integrity violation:**
  a. foreign key is changed and new one refers to nonexistent tuple
  b. primary key is changed and now other tuples that had referred to this one violate the constraint

Dealing with Constraint Violations through UPDATE (2)

UPDATE Constraint violations:

- **Key constraint violation: primary key is changed so as to become same as another tuple**
- **Referential integrity violation:**
  a. foreign key is changed and new one refers to nonexistent tuple
  b. primary key is changed and now other tuples that had referred to this one violate the constraint

Solution:

- Reject the update;
- Modify the foreign key attribute values in referencing tuples to **null** or to some valid value referencing a different tuple

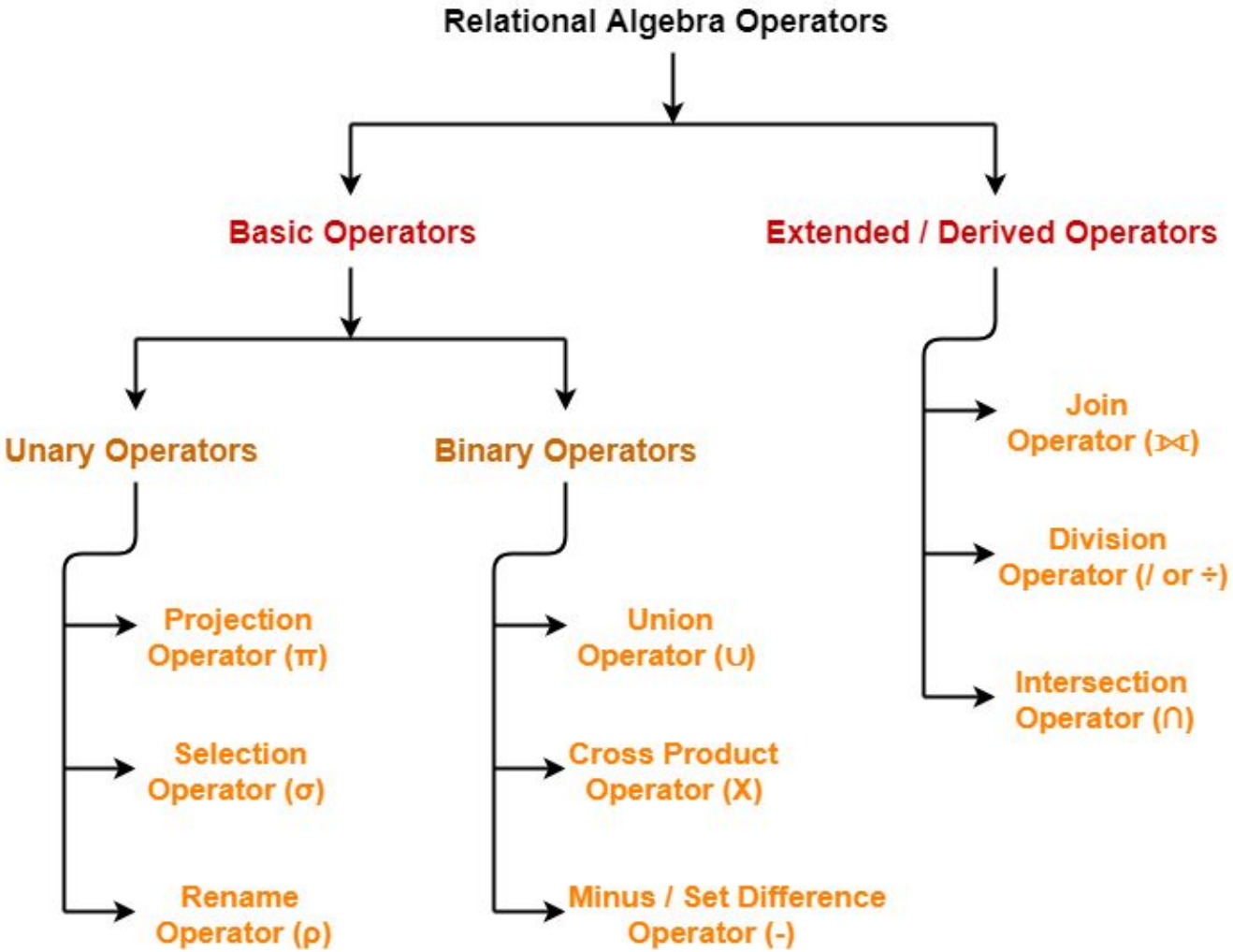
Transactions

- A transaction is an executing program that includes some database operations, such as reading from the database, or applying insertions, deletions, or updates to the database.
- At the end the transaction must leave the database in a valid or consistent state that satisfies all the constraints specified on the database schema.
- These retrievals and updates will together form an atomic unit of work against the database. For example, a transaction to apply a bank withdrawal will typically read the user account record, check if there is a sufficient balance, and then update the record by the withdrawal amount.

Relational Algebra

The Relational Algebra and Relational Calculus

- **Relational algebra**
  - Basic set of operations for the relational model
  - formal foundation for relational model operations
- **Relational algebra expression**
  - Sequence of relational algebra operations
  - Order is important
- **Relational calculus**
  - Higher-level declarative language for specifying relational queries
  - No order of operations to specify how to retrieve the query result—only what information the result should contain.



Two types of operations:

- set operations from set theory

- operations developed specifically for relational databases

Unary Relational Operations: SELECT and PROJECT

- The **SELECT** Operation
  - Subset of the tuples from a relation that satisfies a selection condition:

$$\sigma_{<\text{selection condition}>}(R)$$

    - Boolean expression contains clauses of the form <attribute name> <comparison op> <constant value>

      or

    - <attribute name> <comparison op> <attribute name>

Unary Relational Operations: SELECT and PROJECT (cont.)

- Example:

$$\sigma_{(\text{Dno}=4\ \textbf{AND}\ \text{Salary}>25000)\ \textbf{OR}\ (\text{Dno}=5\ \textbf{AND}\ \text{Salary}>30000)}(\text{EMPLOYEE})$$

- <selection condition> applied independently to each individual tuple *t* in *R*
  - If condition evaluates to TRUE, tuple selected
- Boolean conditions **AND**, **OR**, and **NOT**
- **Unary**
  - Applied to a single relation

Unary Relational Operations: SELECT and PROJECT

- **Selectivity**
  - Fraction of tuples selected by a selection condition
- SELECT operation commutative
- **Cascade** SELECT operations into a single operation with **AND** condition

$$\sigma_{name='Mary' \vee suburb='Bundoora'}(Student)$$

Student

Id	Name	Suburb
1108	Robert	Kew
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

Result

Id	Name	Suburb
3936	Glen	Bundoora
8507	Norman	Bundoora
8452	Mary	Balwyn

The PROJECT Operation

- Selects columns from table and discards the other columns:

$$\pi_{<\text{attribute list}>}(R)$$

- **Degree**
  - Number of attributes in <attribute list>
- **Duplicate elimination**
  - Result of PROJECT operation is a set of distinct tuples

$$\prod_{name,\,dept}(Subject)$$

Subject

Id	Name	Dept
11	Databases	CSE
12	Data Structure	CSE
21	Electrical Eng	Electrical
22	Discrete Math	Math

Result

Name	Dept
Databases	CSE
Data Structure	CSE
Electrical Eng	Electrical
Discrete Math	Math

Sequences of Operations and the RENAME Operation

- **In-line** expression:

$$\pi_{\text{Fname, Lname, Salary}}(\sigma_{\text{Dno}=5}(\text{EMPLOYEE}))$$

- Sequence of operations:

$$\text{DEP5\_EMPS} \leftarrow \sigma_{\text{Dno}=5}(\text{EMPLOYEE})$$
$$\text{RESULT} \leftarrow \pi_{\text{Fname, Lname, Salary}}(\text{DEP5\_EMPS})$$

- **Rename** attributes in intermediate results
  - RENAME operation

$$\rho_{S(B1, B2, ..., Bn)}(R) \quad \text{or} \quad \rho_{S}(R) \quad \text{or} \quad \rho_{(B1, B2, ..., Bn)}(R)$$

Relational Algebra Operations from Set Theory

- **UNION**, **INTERSECTION**, and **MINUS**
  - Merge the elements of two sets in various ways
  - Binary operations
  - Relations must have the same type of tuples (union compatibility)
  - Two relations R(A1, A2, ... , An) and S(B1, B2, ... , Bn) are said to be union compatible (or type compatible) if they have the same degree n and if dom(Ai) = dom(Bi) for $1 \leq i \leq n$.
- **UNION**
  - $R \cup S$
  - Includes all tuples that are either in *R* or in *S* or in both *R* and *S*
  - Duplicate tuples eliminated

Relational Algebra Operations from Set Theory

- **INTERSECTION**
  - $R \cap S$
  - Includes all tuples that are in both *R* and *S*
- **SET DIFFERENCE (or MINUS)**
  - $R - S$
  - Includes all tuples that are in *R* but not in *S*

The set operations UNION, INTERSECTION, and MINUS. (a) Two union-compatible relations. (b) STUDENT ∪ INSTRUCTOR. (c) STUDENT ∩ INSTRUCTOR. (d) STUDENT – INSTRUCTOR. (e) INSTRUCTOR – STUDENT.

(a) **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

(b)

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

(c)

Fn	Ln
Susan	Yao
Ramesh	Shah

(d)

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

The set operations UNION, INTERSECTION, and MINUS. (a) Two union-compatible relations. (b) STUDENT ∪ INSTRUCTOR. (c) STUDENT ∩ INSTRUCTOR. (d) STUDENT – INSTRUCTOR. (e) INSTRUCTOR – STUDENT.

UNION and **INTERSECTION** are commutative operations;

MINUS is not commutative operation;

(a) **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

(b)

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

(c)

Fn	Ln
Susan	Yao
Ramesh	Shah

(d)

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

(e)

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

The CARTESIAN PRODUCT (CROSS PRODUCT) Operation

- **CARTESIAN PRODUCT**
  - **CROSS PRODUCT** or **CROSS JOIN**
  - Denoted by ×
  - Binary set operation
  - Relations do not have to be union compatible
  - Useful when followed by a selection that matches values of attributes

Binary Relational Operations: JOIN and DIVISION

- The **JOIN** Operation
  - Denoted by ⋈
  - Combine related tuples from two relations into single “longer” tuples
  - General join condition of the form <condition> **AND** <condition> **AND**...**AND** <condition>
  - Example:

$$\text{DEPT\_MGR} \leftarrow \text{DEPARTMENT} \bowtie_{\text{Mgr\_ssn=Ssn}} \text{EMPLOYEE}$$
$$\text{RESULT} \leftarrow \pi_{\text{Dname, Lname, Fname}}(\text{DEPT\_MGR})$$

Binary Relational Operations: JOIN and DIVISION

- **THETA JOIN**
  - Each <condition> of the form A_i θ B_j
  - A_i is an attribute of R
  - B_j is an attribute of S
  - A_i and B_j have the same domain
  - θ (theta) is one of the comparison operators:
    - {=, <, ≤, >, ≥, ≠}

Variations of JOIN: The EQUIJOIN and NATURAL JOIN

- **EQUIJOIN**
  - Only = comparison operator used
  - Always have one or more pairs of attributes that have identical values in every tuple
- **NATURAL JOIN**
  - Denoted by *
  - Removes second (superfluous) attribute in an EQUIJOIN condition

Variations of JOIN: The EQUIJOIN and NATURAL JOIN

- **Join selectivity**
  - Expected size of join result divided by the maximum size $n_R * n_S$
- **Inner joins**
  - Type of match and combine operation
  - Defined formally as a combination of CARTESIAN PRODUCT and SELECTION

A Complete Set of Relational Algebra Operations

- Set of relational algebra operations {σ, π, ∪, ρ, –, ×} is a **complete set**
  - Any relational algebra operation can be expressed as a sequence of operations from this set

A Complete Set of Relational Algebra Operations

- Set of relational algebra operations {σ, π, ∪, ρ, –, ×} is a **complete set**
  - Any relational algebra operation can be expressed as a sequence of operations from this set
  - **INTERSECTION** can be expressed by using UNION and MINUS as follows: $R \cap S \equiv (R \cup S) - ((R - S) \cup (S - R))$
  - **JOIN** can be specified as a CARTESIAN PRODUCT followed by a **SELECT** operation: $R \ _{<condition>} S \equiv \sigma_{<condition>}(R \times S)$

How to express NATURAL JOIN?

A Complete Set of Relational Algebra Operations

- Set of relational algebra operations {σ, π, ∪, ρ, –, ×} is a **complete set**
  - Any relational algebra operation can be expressed as a sequence of operations from this set
  - **INTERSECTION** can be expressed by using UNION and MINUS as follows: $R \cap S \equiv (R \cup S) - ((R - S) \cup (S - R))$
  - **JOIN** can be specified as a CARTESIAN PRODUCT followed by a **SELECT** operation: $R \ _{<condition>} S \equiv \sigma_{<condition>}(R \times S)$

How to express NATURAL JOIN?

NATURAL JOIN can be specified as a CARTESIAN PRODUCT preceded by RENAME and followed by SELECT and PROJECT operations

The DIVISION Operation

- Denoted by ÷
- Example: retrieve the names of employees who work on all the projects that ‘John Smith’ works on
- Apply to relations $R(Z) \div S(X)$
  - Attributes of R are a subset of the attributes of S

The DIVISION operation. (a) Dividing SSN_PNOS by SMITH_PNOS. (b) $T \leftarrow R \div S$.

(a)

SSN_PNOS

Essn	Pno
123456789	1
123456789	2
666884444	3
453453453	1
453453453	2
333445555	2
333445555	3
333445555	10
333445555	20
999887777	30
999887777	10
987987987	10
987987987	30
987654321	30
987654321	20
888665555	20

SMITH_PNOS

Pno
1
2

SSNS

Ssn
123456789
453453453

(b)

R

A	B
a1	b1
a2	b1
a3	b1
a4	b1
a1	b2
a3	b2
a2	b3
a3	b3
a4	b3
a1	b4
a2	b4
a3	b4

S

A
a1
a2
a3

T

B
b1
b4

Example Division

Operations of Relational Algebra

Table 6.1 Operations of Relational Algebra

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 $R_1 \bowtie_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$
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 $R_1 *_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$ OR $R_1 * R_2$

Operations of Relational Algebra

Table 6.1 Operations of Relational Algebra

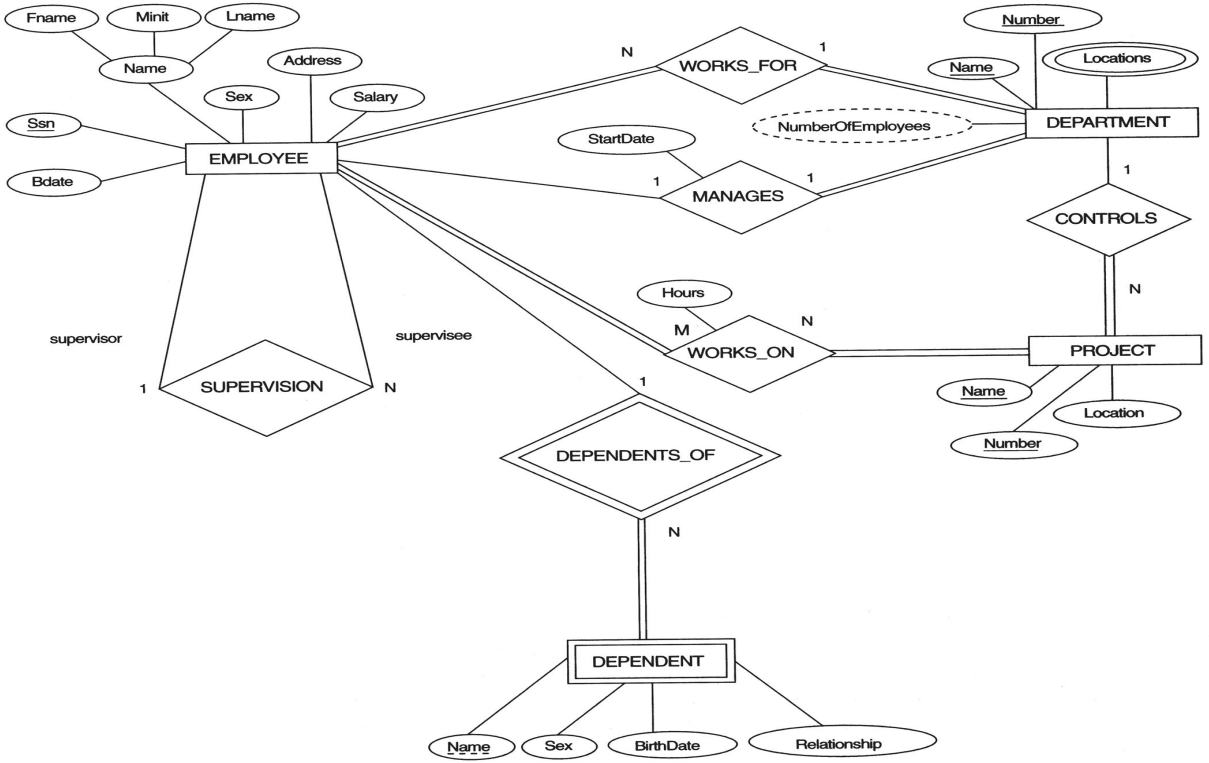
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a 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 $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

Relational Database Design by ER-to-Relational Mapping

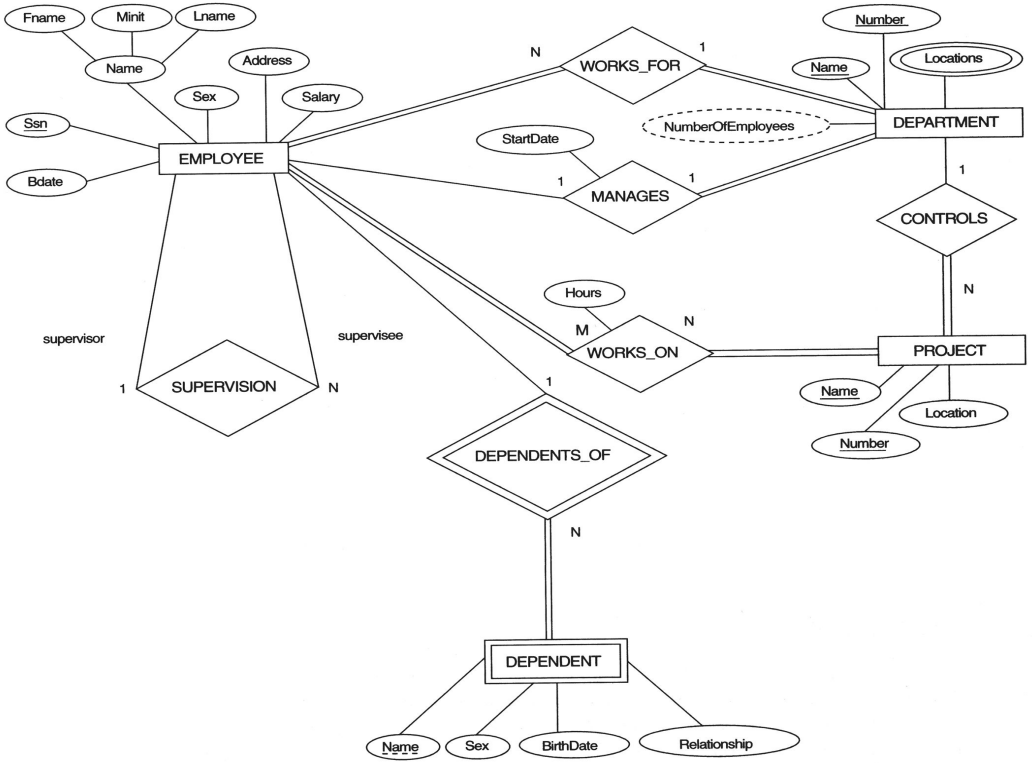
ER-to-Relational Mapping Algorithm

- **Step 1: Mapping of Regular Entity Types.**
  - For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
  - Choose one of the key attributes of E as the primary key for R.
  - If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

The ER conceptual schema diagram for the COMPANY database.



The ER conceptual schema diagram for the COMPANY database.



- **Example:** We create the relations EMPLOYEE, DEPARTMENT, and PROJECT in the relational schema corresponding to the regular entities in the ER diagram.
  - SSN, DNUMBER, and PNUMBER are the primary keys for the relations EMPLOYEE, DEPARTMENT, and PROJECT as shown.

Partial Result of mapping the COMPANY ER schema into a relational schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

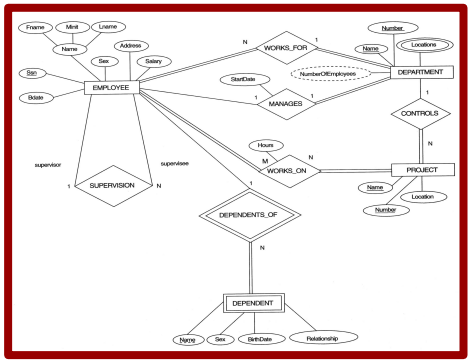
Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours



ER-to-Relational Mapping Algorithm

- **Step 2: Mapping of Weak Entity Types**
  - For each weak entity type W in the ER schema with owner entity type E, create a relation R & include all simple attributes (or simple components of composite attributes) of W as attributes of R.
  - Also, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).
  - The primary key of R is the *combination of* the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.
- **Example:** The relation DEPENDENT in this step corresponds to the weak entity type DEPENDENT.
  - Include the primary key SSN of the EMPLOYEE relation as a foreign key attribute of DEPENDENT (renamed to ESSN).
  - The primary key of the DEPENDENT relation is the combination {ESSN, DEPENDENT_NAME} because DEPENDENT_NAME is the partial key of DEPENDENT.

ER-to-Relational Mapping Algorithm

- **Step 3: Mapping of Binary 1:1 Relation Types**
  - For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R.
- There are three possible approaches:
  1. **Foreign Key approach:** Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S.
     - Example: 1:1 relation MANAGES is mapped by choosing the participating entity type DEPARTMENT to serve in the role of S, because its participation in the MANAGES relationship type is total.
  2. **Merged relation option:** An alternate mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This may be appropriate when both participations are total.
  3. **Cross-reference or relationship relation option:** The third alternative is to set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types.

ER-to-Relational Mapping Algorithm

- **Step 4: Mapping of Binary 1:N Relationship Types.**
  - For each regular binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type.
  - Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.
  - Include any simple attributes of the 1:N relation type as attributes of S.
- Example: 1:N relationship types WORKS_FOR, CONTROLS, and SUPERVISION in the figure.
  - For WORKS_FOR we include the primary key DNUMBER of the DEPARTMENT relation as foreign key in the EMPLOYEE relation and call it DNO.

ER-to-Relational Mapping Algorithm

- **Step 5: Mapping of Binary M:N Relationship Types.**
  - For each regular binary M:N relationship type R, *create a new relation* S to represent R.
  - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; *their combination will form the primary key* of S.
  - Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S.
- **Example:** The M:N relationship type WORKS_ON from the ER diagram is mapped by creating a relation WORKS_ON in the relational database schema.
  - The primary keys of the PROJECT and EMPLOYEE relations are included as foreign keys in WORKS_ON and renamed PNO and ESSN, respectively.
  - Attribute HOURS in WORKS_ON represents the HOURS attribute of the relation type. The primary key of the WORKS_ON relation is the combination of the foreign key attributes {ESSN, PNO}.

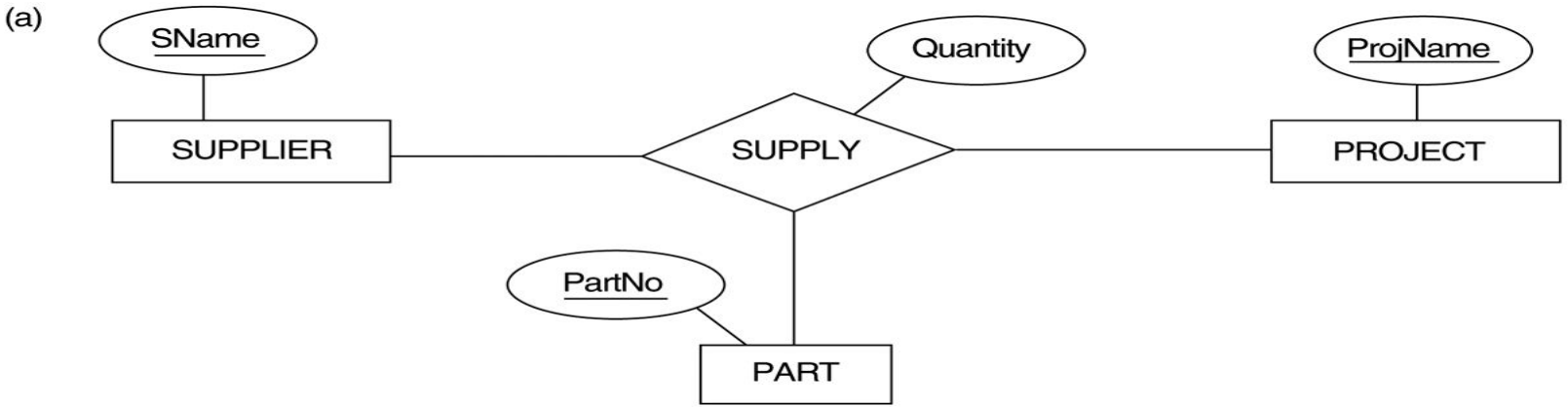
ER-to-Relational Mapping Algorithm

- **Step 6: Mapping of Multivalued attributes.**
  - For each multivalued attribute A, create a new relation R.
  - This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.
  - The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.
- **Example:** The relation DEPT_LOCATIONS is created.
  - The attribute DLOCATION represents the multivalued attribute LOCATIONS of DEPARTMENT, while DNUMBER-as foreign key-represents the primary key of the DEPARTMENT relation.
  - The primary key of R is the combination of {DNUMBER, DLOCATION}.

ER-to-Relational Mapping Algorithm

- **Step 7: Mapping of N-ary Relationship Types.**
  - For each n-ary relationship type R, where n>2, create a new relationship S to represent R.
  - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.
  - Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.
- **Example:** The relationship type SUPPY in the ER on the next slide.
  - This can be mapped to the relation SUPPLY shown in the relational schema, whose primary key is the combination of the three foreign keys {SNAME, PARTNO, PROJNAME}

Exercise.



Map the following ER Diagram to Relational model.

SUPPLIER

SNAME	. . .

PROJECT

PROJNAME	. . .

PART

PARTNO	. . .

SUPPLY

SNAME	PROJNAME	PARTNO	QUANTITY

Summary of Mapping constructs and constraints

Correspondence between ER and Relational Models

ER Model	Relational Model
Entity type	"Entity" relation
1:1 or 1:N relationship type	Foreign key (or "relationship" relation)
M:N relationship type	"Relationship" relation and two foreign keys
n-ary relationship type	"Relationship" relation and n foreign keys
Simple attribute	Attribute
Composite attribute	Set of simple component attributes
Multivalued attribute	Relation and foreign key
Value set	Domain
Key attribute	Primary (or secondary) key


Q & A