DATABASE - 1

3RD CLASS

COMPUTER SCIENCE DEPARTMENT

4th Lecture – Logical Database Design and The Relational Model

Monday 14th of Oct. 2024

LECTURER :

DR. RAYAN YOUSIF ALKHAYAT

LOGICAL DATABASE DESIGN AND THE RELATIONAL MODEL:

Logical database design:

is the process of transforming the conceptual data model (ER-Model) into a logical data model.

- **There are two reasons for emphasizing on the logical model:**

1. It is the most used in contemporary database applications.
2. Some of the principles of logical data base design for the relational model apply to the other logical model as well.

THE RELATIONAL DATA MODEL :

E. F. Codd first introduced the relational data model in 1970. Today RDSBMSs have been the dominate technology for database management.

Basic definitions:

The relational model based on mathematical theory and therefore has a solid theoretical foundation, it consists of the following components:

1. **Data Structure**: Data are organized in the form of tables with rows and columns.
2. **Data manipulation :** Powerful operations (using SQL Language) are used to manipulate data stored in the relation.
3. **Data integrity:** Facilities are included to specify business rules that maintain the integrity of data when they are manipulated.

Relational Data Structure:

A Relation is a named two-dimensional table of data , each relation consists of a set of named columns and an arbitrary number of unnamed rows .

An Attribute (consistent of its previous definition) is a named column of a relation.

EACH ROW OF THE RELATION CORRESPONDS TO A RECORD THAT CONTAINS DATA (ATTRIBUTES) VALUES FOR A SINGLE ENTITY. SEE NEXT FIGURE:

EMPLOYEE1

Emp_ID	Name	Dept_Name	Salary
100	Mohammad zaki	Marketing	530.000
140	Sufyan salim	Accounting	480.000
110	Sinan zohair	Info systems	225.000
190	Muthanna Mohammad	Finance	400.000
150	Mahmmod khalid	Marketing	75.000

We can express the structure of a relation by a shorthand notation in which the name of the relation is followed (in parentheses) by the names of the attributes in that relation. For EMPLOYEE1 Example we would have:

EMPLOYEE1 (Emp_id, Name,Dept_Name,Salary).

Relational Keys;

We must be able to store and retrieve a row of data in a relation based on the data values stored in that row.

- To achieve this goal, every relation must have a primary key .
- A **Primary Key** : is an attribute (or combination of attributes) that uniquely identifies each row in a relation.
- We designate a Primary Key by underlining the attribute name. For example the primary key for the relation EMPLOYEE is Emp_id and we represent it as follows :

EMPLOYEE1 (Emp_id, Name,Dept_Name,Salary).

- The Concept of a primary key is related to the term "identifier" in many cases the same attribute (or attributes) indicates as an entity's identifier in an E R-diagram will be the same attributes that compose the primary key for the relation representing that entity .
- There are exceptions, for example associated entities is not have to had an identifier and the identifier of a weak entity forms only part of a weak entity's primary key..
- In addition there may be several attributes of an entity that may be several attributes of an entity that may serve as the associated relation's primary key.

- **A composite key** : is a primary key that consists of more than one attribute .

For example the primary key for a relation DEPENDENT would likely consists of the combination of Emp_id and Dependent_name.

- **Foreign key** an attribute in a relation of a database that serves as the primary key in another relation in the same database..

Some authors emphasize the fact that an attribute is a foreign key by using dashed underline, such as :

EMPLOYEE1(Emp_id,Name,Dept_Name,Salary).

DEPARTMENT(Dept_Name,Location,Fax)

The attribute Dept_Name is a foreign key in EMPLOYEE . It allows a user to associate any employee with the department to which he or she is assigned.

D→Properties of the Relations:

Relation have several properties; we summarize them in the following:

1. Each relation (or table) in a database has a unique name.
2. Any entry at the intersection of each row and column is atomic (or single valued) there can be no multivalued attributes in a relation.
3. Each row is a unique , no two rows in a relation are identical
4. Each attribute (or column) within a table has a unique name.
5. The sequence of columns (left to right) in insignificant . The columns of a relation can be interchanged without changing the meaning or use of the relation.
6. The sequence of rows (top to bottom) .As with columns .The rows of a relation may be interchanged or stored in any sequence .

Removing Multivalued Attributes from Tables :

- The second property of the relation above states that can be no multivalued attributes in a relation. Thus, a table that contains one or more multivalued attributes is not a relation.

Removing Multivalued Attributes from Tables :

- As an example, the next figure shows the employee data from the EMPLOYEE1 relation extended to include courses that may have been taken by those employees.
- Since a given employee may have taken more than one course, the attributes Course_Title and Date_Completed are multivalued attributes. For example , the employee with Emp_Id 100 has taken two courses .
- If an employee has not taken the any courses, the Course_Title and Date_Completed attributes (see employee with Emp_Id 190 for an example).

- **EMPLOYEE2**

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Mohammad zaki	Marketing	530.000	SPSS	6/19/2006
				Cisco	10/7/2006
140	Sufyan salim	Accounting	480.000	CNC	12/8/2005
110	Sinan zohair	Info systems	225.000	Tax Acc	1/12/2006
				SPSS	4/22/2005
190	Muthanna Mohammad	Finance	400.000		
150	Mahmmod khalid	Marketing	75.000	SPSS	6/16/2005
				Java	8/12/2006

We show how to eliminate the multivalued attributes in the next figure, by filling the relevant data values into the previously vacant cells of previous figure. As a result, the table in figure below has only single-valued attributes and now satisfies the atomic property of relation, but still have some undesirable properties.

Employee2

Emp_ID	Name	Dept_Name	Salary	Course_Title	Date_Completed
100	Mohammad zaki	Marketing	530.000	SPSS	6/19/2006
100	Mohammad zaki	Marketing	530.000	Cisco	10/7/2006
140	Sufyan salim	Accounting	480.000	CNC	12/8/2005
110	Sinan zohair	Info systems	225.000	Tax Acc	1/12/2006
110	Sinan zohair	Info systems	225.000	SPSS	4/22/2005
190	Muthanna Mohammad	Finance	400.000		
150	Mahmmod khalid	Marketing	75.000	SPSS	6/16/2005
150	Mahmmod khalid	Marketing	75.000	Java	8/12/2006

Example Database :

A relational data base consists of any number of relations. The structure of the database is described through the use of a conceptual schema. There are two common methods for expressing a (conceptual) schema:

a. Short text statement , in which each relation is named and the names of its attributes follow in parentheses such as:
EMPLOYEE(Emp_id,Name,Dept_Name,Salary).
b. A graphical representation, in which each relation is represented by a rectangle containing the attributes for the relation such as .

CUSTOMER

Customer_Id	Customer_Name	Adress	City

ORDER

Order_Id	Order_Date	Customer_Id

ORDER LINE

Order_Id	Product_Id	Quantity

PRODUCT

Product_Id	Product_description	Product_Finish	Unit_Price	On_Hand

It is a good idea to create an instance of the relational schema with sample data for three reasons :

1. The sample data provide a conventional way to check the accuracy of the design.
2. The sample data help improve communications with users in discussing the design .
3. Sample data can be used to develop prototype applications and test queries.

INTEGRITY CONSTRAINS :

- The relational data model includes several types of constrains whose purpose is to facilitate maintaining the accuracy and integrity of data in the database . The major types of integrity constrains are domain constrains . entity integrity , referential integrity, and operational constrains.

DOMAIN CONSTRAINS :

All the values that appear in a column of a relation must be taken from the same domain . **A domain** is the set of values that may be assigned to an attribute .

The **domain** consists of the following components :

1. Domain name
2. Data type
3. Size(or length)
4. Allowable values (or allowable range)

The table below shows domain definition for domain associated with attributes for EMPLOYEE1:

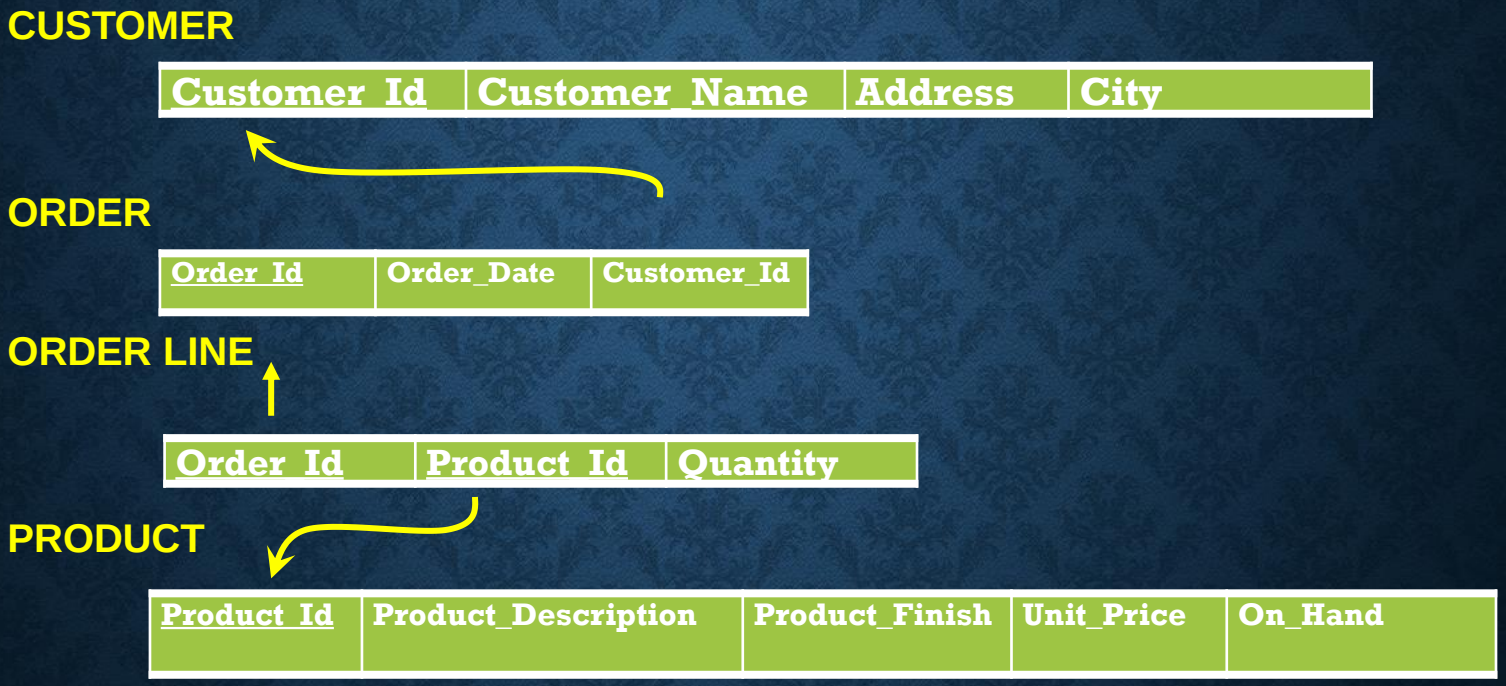
Attribute	Domain name	Description	Domain
Emp_id	Emp_id	Set of all possible employee IDs	Character Size 6
Name	Name	Set of all possible employee names	Character size 30
Dept_Name	Dept_Name	Set of all possible departments	Character size 20
Salary	Salary	Set of all possible e Salaries	Numeric size 8

2.ENTITY INTEGRITY:

- The entity integrity rule is designed to assure that every relation has a primary key, and that the data values for the primary key are all valid. It guarantees that every primary key attribute is non-null.
- **NULL:** a value that may be assigned to an attribute when no other value applies , or when the application value is unknown.

3.REFERENTIAL INTEGRITY:

- In the relational data model the association between tables are defined through the use of foreign keys.
- A **referential** integrity constraint is rule that maintains consistency among the rows of two relations. The rule states that if there is a foreign key in one relation, either each foreign key value must match a primary key value in the other relation or else the foreign key value must be NULL.


CUSTOMER
Customer_Id
Customer_Name
Address
City
ORDER
Order_Id
Order_Date
Customer_Id
ORDER LINE
Order_Id
Product_Id
Quantity
PRODUCT
Product_Id
Product_Description
Product_Finish
Unit_Price
On_Hand

4. OPERATIONAL CONSTRAINTS

- this constraints belongs to business rules we discuss it early before .as an example(A person may purchase a ticket for the all – star game only if that person is a season – ticket holder)

Creating Relational Tables:

At this point, we create table definition for the four tables above . These definitions are created using CREATE TABLE statements from SQL data definition language , actually these table definitions are created during the implementation phase later in the database development process.

The SQL table definition are shown in the relational schema , each attribute for a table is then defined. Notice that the data type and length for each attribute is taken from the domain definition .

```
CREATE TABLE  CUSTOMER
(CUSTOMER_ID            VARCHAR(5)          NOTNULL,
CUSTOMER_NAME       VARCHAR(25)          NOTNULL,
CUSTOMER_ADDRESS  VARCHAR(30)          NOTNULL,
PRIMARY KEY(CUSTOMER_ID));
```

```
CREATE TABLE  ORDER
(ORDER_ID            CHAR (5)              NOTNULL,
ORDER_DATE          DATE                  NOTNULL,
CUSTOMER_ID         VARCHAR (5)           NOTNULL,
PRIMARY KEY (ORDER_ID),
FOREGIN KEY (CUSTOMER_ID) REFERENCES CUSTOMER
   (CUSTOMER_ID));
```

```
CREATE TABLE ORDER LINE
   (ORDER_ID                CHAR (5)                          NOTNULL,
   PRODUCT_ID               CHAR(5)          NOTNULL,
QUANTITY                          INT                          NOTNULL
PRIMARY KEY (ORDER_ID, PRODUCT_ID),
FOREGIN KEY (ORDER_ID) REFERENCES ORDER (ORDER_ID),
FOREGIN KEY (PRODUCT_ID) REFERENCES PRODUCT_ID (PRODUCT_ID));

CREATE TABLE PRODUCT
(PRODUCT_ID                 CHAR (5)                NOTNULL,
  PRODUCT_DESCRIPTION    VARCHAR (25),
  PRODUCT_FINISH             VARCHAR (12,
  UNIT_PRICE                  DECIMAL (8, 2)                NOTNULL,
ON_HAND                           INT                          NOTNULL,
PRIMARY KEY (PRODUCT_ID));
```

WELL STRUCTURED RELATIONS:

A relation that contains minimal redundancies and allows users to insert, modify , and delete the rows in a table without errors or inconsistencies .

- Redundancies in a table may results in errors or inconsistencies called anomalies.
- **Anomalies** : errors or inconsistencies that may result when a user attempts to update a table

THERE ARE THREE TYPES OF ANOMALIES :

1. ***Insertion anomaly***:

Suppose that we need to add anew employee to EMPLOYEE2 .

The primary key for this relation id Emp_Id and Course_Title, therefore, to insert anew row , the user must supply values for both Emp_Id and Course_Title(since primary key values cannot be null or nonexistent)

This is an anomaly ,since the user should be able to enter employee data without supplying course data.

2. Deletion anomaly:

Suppose that data for employee number 140 are deleted from the table, this will result in losing the information that this employee completed a course (CNC) on 12/8/2005.

It results in losing the information that this course had an offering that completed on that date.

3. ***Modification anomaly:*** suppose that employee 100 gates a salary increases . We must record this increase in each of the rows for that employee (two occurrences); otherwise, the data will be inconsistent .

- These anomalies indicate that EMPLOYEE2 is not well structured relation.
- The problem with this relation is that it contains data about two entities: EMPLOYEE and COURSE.
- It should be normalized using normalization theory to divide EMPLOYEE2 into two relations , on of the resulting relation is EMPLOYEE1 , the other will be EMP_COURSE, which appears with sample data in next figure :

EMP_COURSE

Emp_ID	Course_Title	Date_Completed
100	SPSS	6/19/2006
100	Cisco	10/7/2006
140	CNC	12/8/2005
110	Tax Acc	1/12/2006
110	SPSS	4/22/2005
150	SPSS	6/16/2005
150	Java	8/12/2006

The primary key of this relation is the combination of Emp_Id and Course_Title ,and we underline these attribute names to highlight this fact, so it is a well structured relation.

TRANSFORMING ER DIAGRAMS INTO RELATIONS:

In the logical design the E-R (and EER) Diagram transforms to a relational database schema, the input to this process is the ER-Diagram and the output is the relational schema.

It is a straightforward process with a well-defined set of rules

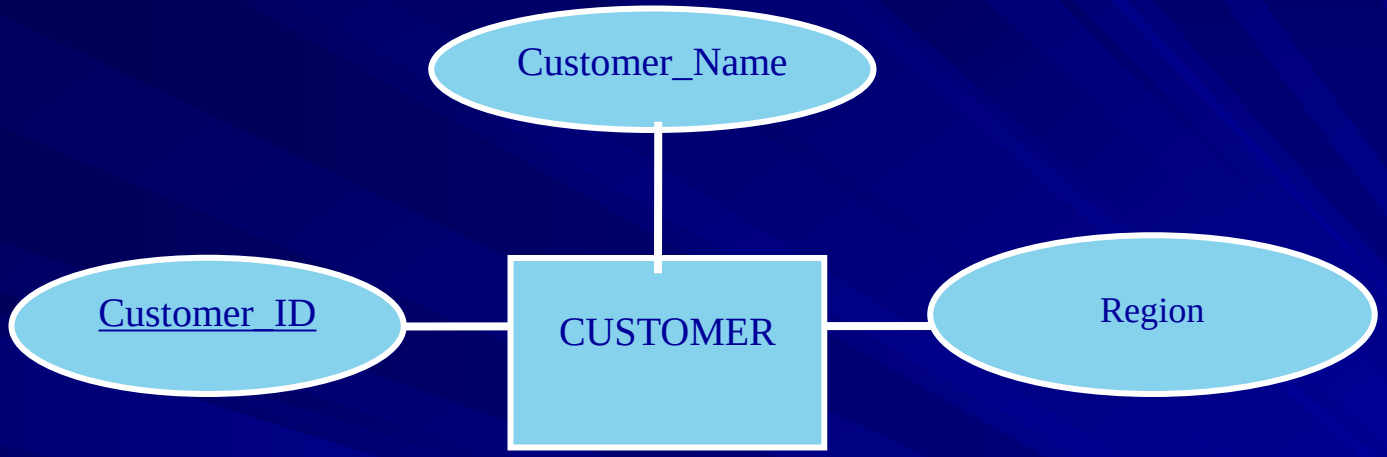
Three Types of Entities Will Discussed:

- ***Regular entities***: are entities that have an independent existence and generally represent real-word objects such as persons and products. Regular entity types are represented by rectangles with single line.
- ***Weak entities***: are entities that cannot exist except with an identifying relationship with an owner (regular)entity type. Weak entities identified by a rectangle with a double line.
- ***Associative entities*** (gerunds): they formed from many-to-many relationships between other entity types .Associative entities represented by a rectangle with single line that encloses the diamond relationship symbol.

STEP1: Map regular Entities:

- Each regular entity type in an ER-diagram is transformed into a relation .The name given to the relation is generally the same as the entity type. Each simple attribute of the entity type becomes an attribute of the relation. The identifier of the entity type becomes the primary key satisfies the desirable properties of identifiers.

- The figure below shows a representation of CUSTOMER entity type, and the corresponding relation shown in graphical form, (we take only few attributes to simplify the figure).

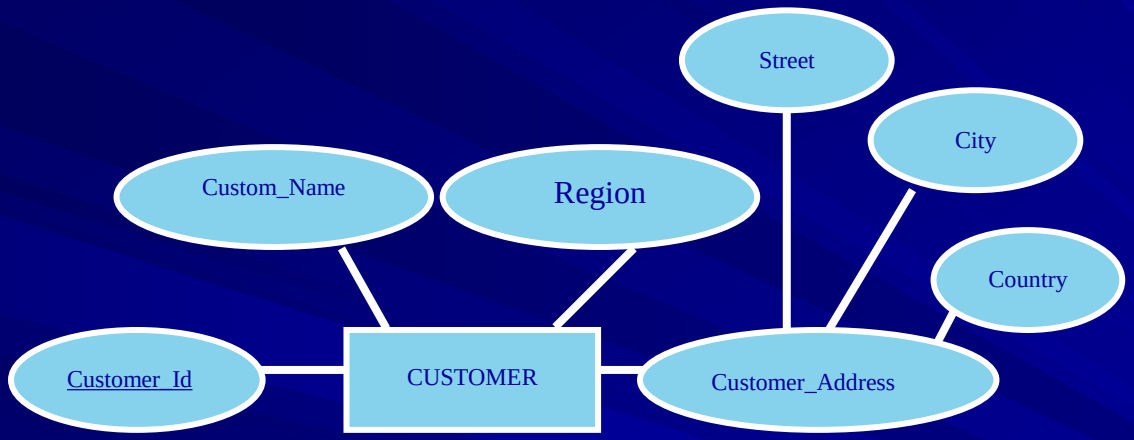


CUSTOMER

Customer_Id	Customer_Name	Region

Composite attribute:

- When the regular entity type has composite attributes, only the simple components attributes of the composite attribute are included in the new relation. Figure below shows a variation of the previous example:



CUSTOMER

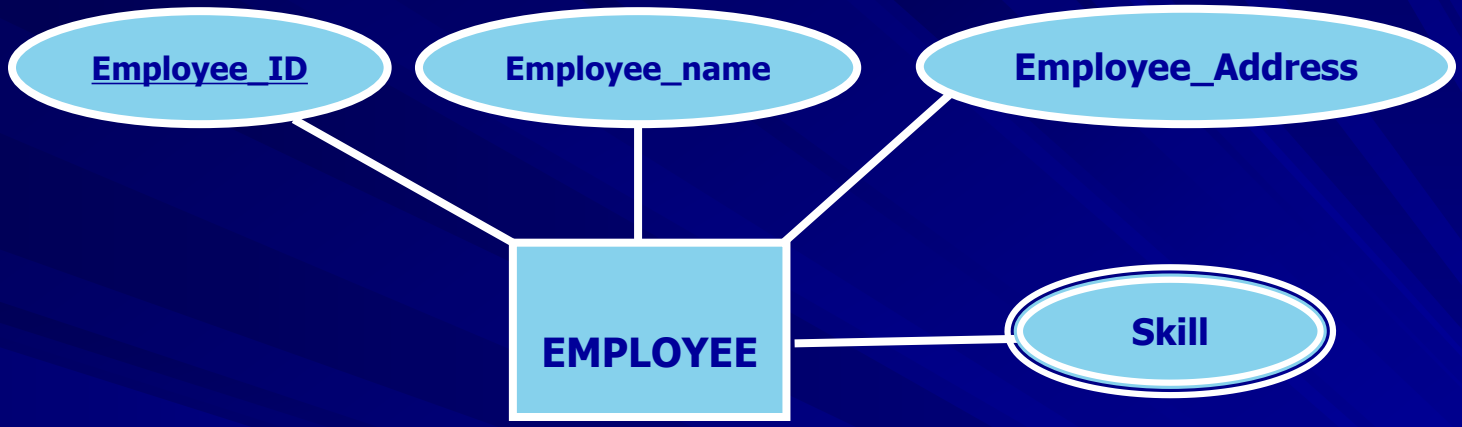
Customer_Id	Customer_Name	Street	City	Country

We see that Customer_Address, which is a composite attribute, with components Street, City, Country, mapped to the relation with its simple attributes only.

Multivalued attributes:

- When regular entity contain multivalued attribute, two new relations (rather than one) are created. The first relation contains all of the attributes of the entity type except the multivalued attribute. The Second relation contains two attributes that form the primary key of the second relation. The first of these attributes is the primary key from the first relation, which becomes a foreign key in the second relation. The second is the multivalued attribute .the name of the second relation should capture the meaning of the multivalued attribute.

- As an example of this procedure, in the figure below, this is the EMPLOYEE entity type for a company. As shown in part (a) EMPLOYEE has Skill as a multivalued attribute. Part (b) shows the two relations that are created. The first (called EMPLOYEE) has the primary key Employee_ID. The second relation (called EMPLOYEE_SKILL) has the two attributes Employee_ID and Skill, which form the primary key. The relationship between foreign and primary keys is indicated by arrow in the figure.



EMPLOYEE

Employee_ID	Employee_Name	Employee_Address

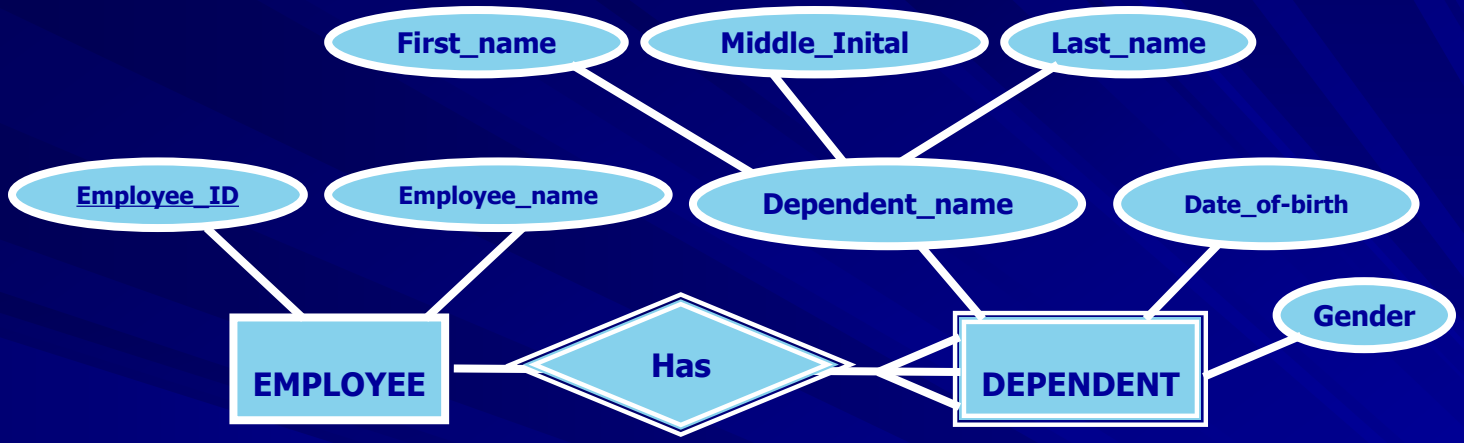
EMPLOYEE_SKILL

Employee_ID	Skill

STEP 2: Map Weak Entities:

1. for each weak entity type, create a new relation and include all of the simple attributes(or the simple components of composite attributes) as attributes of this relation.
2. Include the primary key of the *owner* relation as a foreign key attribute in this new relation. The primary key of the new relation is the combination of this primary key of the owner and the partial identifier of the weak entity type.

- An example of this process is shown in figure bellow; in part(a) we see the weak entity type DEPENDENT and its owner entity type EMPLOYEE, linked by the identifying relationship Has, notice that the attribute Dependent_name ,which is the partial identifier for this relation , is a composite attribute with components First_Name, Middle_Initial, and Last_Name .
- Thus we assume that *for a given employee* these items will uniquely identify a dependent. In Part (b) we see two relations that result from mapping this E-R segment.
- The primary key of DEPENDENT consists of four attributes: Employee_ID, First_Name , Middle_Initial, and Last_Name, Date_of_Birth and GENDER are the nonkey attributes. The foreign key relationship with its primary key is indicated by the arrow in the figure.



EMPLOYEE

Employee_ID	Employee_Name

DEPENDENT

First_Name	Middle_Inital	Last_Name	Date_of_Birth	Employee_ID	Gender

END OF LECTURE 4