Database Design and SQL

Chapter 3: Advanced Data Modelling using ERD

CPCM/FSDM-2023F

Sagara Samarawickrama

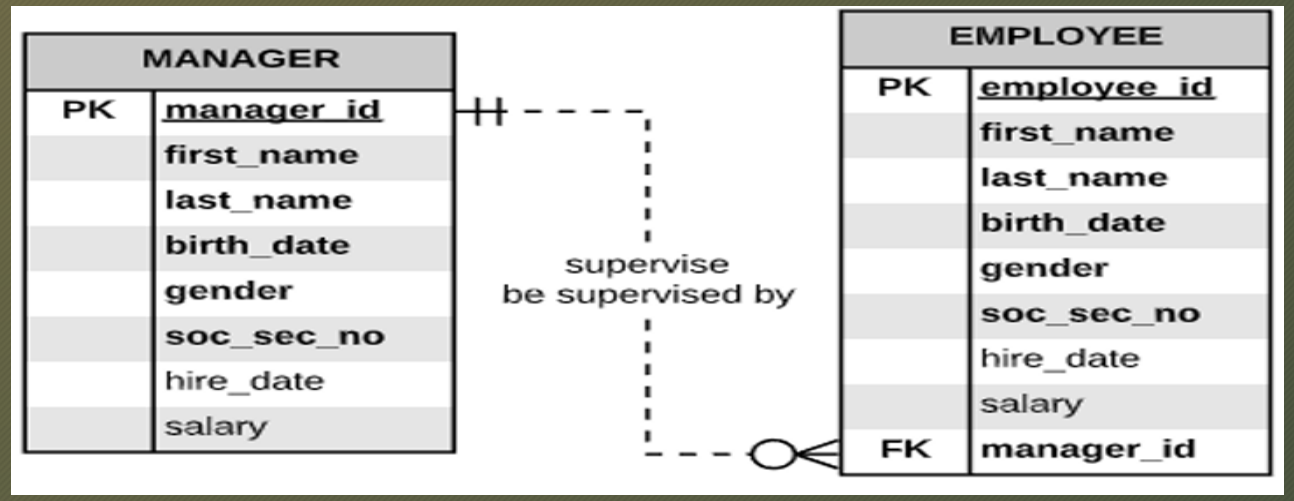
Chapter Objectives

Upon completion of this chapter, you should be able to:

- Explain what entity relationship diagram (ERD) are
- Identify and explain the steps for deveoping ERD
- Develop ER diagrams
- Explain what a schema is
- Create schemas, tables and indexes
- Explain and use different data types
- Explain the purpose of NULL values and how they are used in tables
- Insert data into tables
- Delete tables and indexes

Advanced Data Modelling using ERD

Recursive Relationship represents a self-referencing relationship. That is , when an entity has a relationship with itself. Let us consider a situation where , as in many companies , each employee reports to one manager. This can be represented by the ERD as shown below

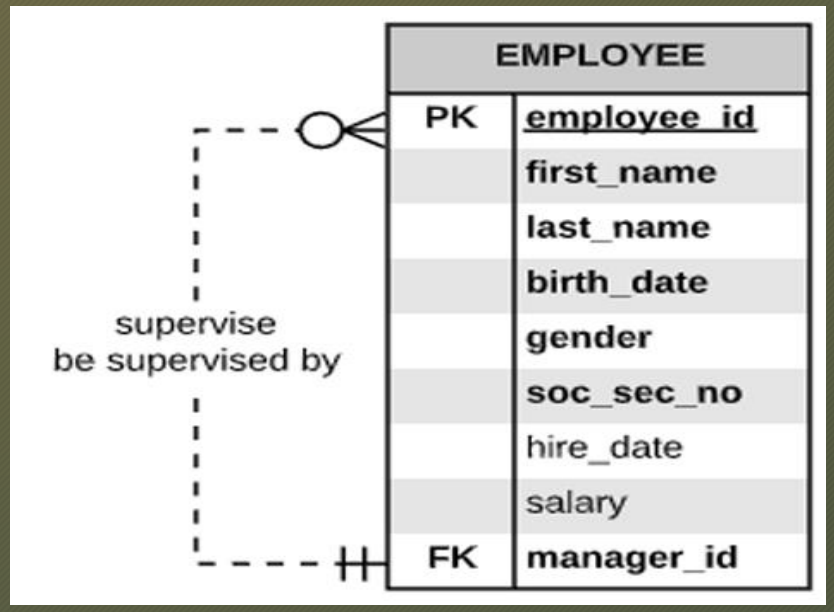


Each MANAGER may *supervise* zero, one or more EMPLOYEEs

Each EMPLOYEE must *be supervised by* one and only one MANAGER

Advanced Data Modelling using ERD

The problem with this model is each manager is also an employee. Creating the second MANAGER entity not only duplicates data from the EMPLOYEE entity, but it virtually guarantees that there will be mistakes and conflicts in the data. This problem can be fixed by eliminating the redundant entity and redrawing the relationship line

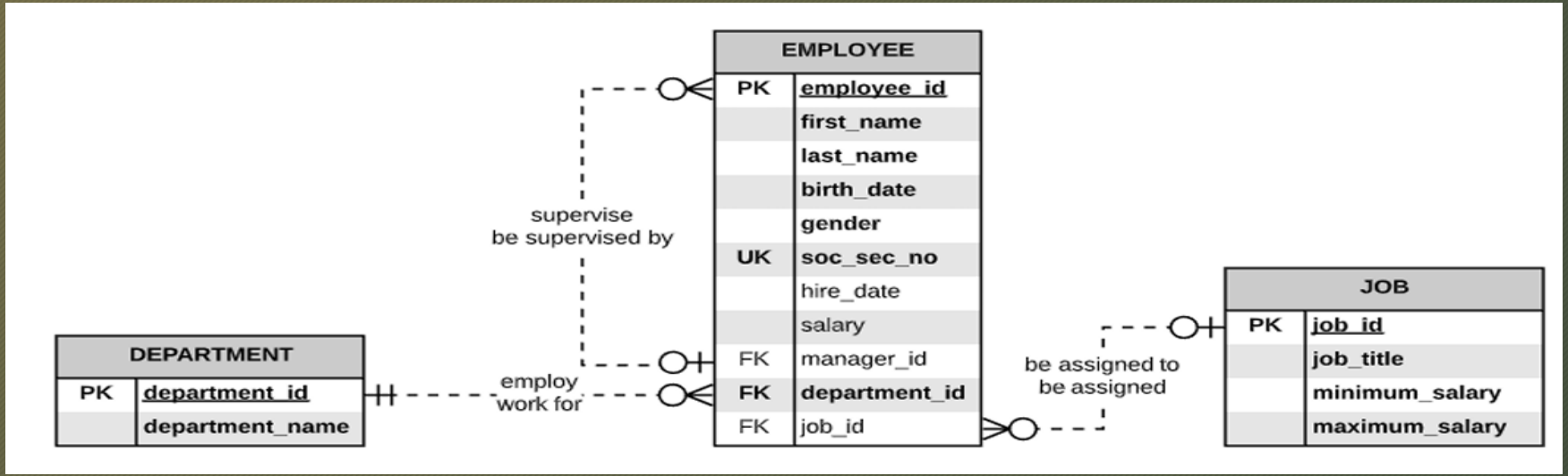


- Every manager is an employee
- Not all employees are managers
- Every employee reports to a manager

Each EMPLOYEE (Manager role) may *supervise* zero, one or more EMPLOYEEs (Employee role)
Each EMPLOYEE (Employee role) must *be supervised by* one and only one EMPLOYEE (Manager role)

Advanced Data Modelling using ERD

ERD with a recursive relationship



Each DEPARTMENT may *employ* zero, one or more EMPLOYEEs
Each EMPLOYEE must *work for* one and only one DEPARTMENT
Each JOB may *be assigned to* zero, one or more EMPLOYEEs
Each EMPLOYEE may *be assigned* zero or one JOB
Each EMPLOYEE (Manager role) may *supervise* zero, one or more EMPLOYEEs (Employee role)
Each EMPLOYEE (Employee role) may *be supervised by* zero or one EMPLOYEE (Manager role)

Advanced Data Modelling using ERD

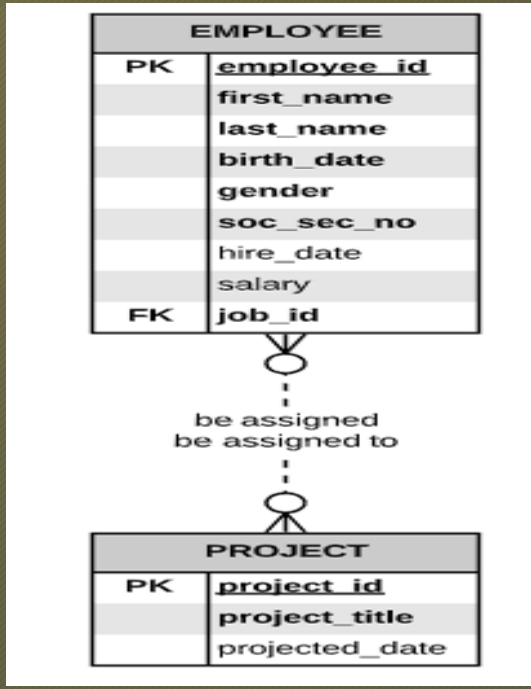
Many-to-Many Relations

A many-to-many relationship is a relationship between two entities where an instance in entity 1 may connect with one or more instances in entity 2 and one instance in entity 2 may connect to with one or more instances in entity 1

In addition ,following requirements were also determined

- Each employee may be assigned to many projects. Sometimes, however, an employee may be off work and is not assigned any projects.
- Each project may be assigned to several employees. However, when a project is first initiated, there are no employees assigned to it.
- The project data is stored in the project entity and includes the project identification, project title, and projected completion date.

Advanced Data Modelling using ERD

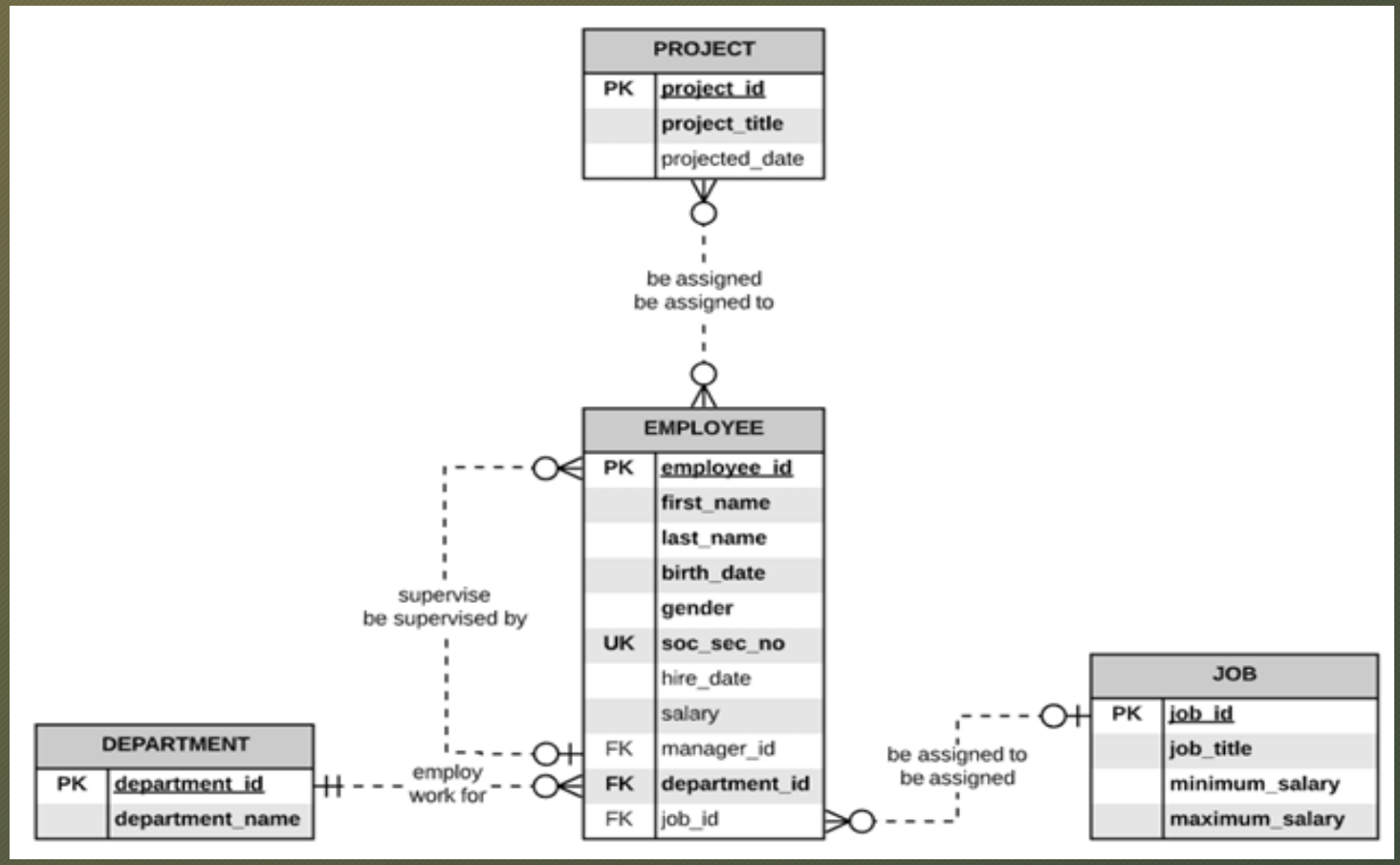


Each EMPLOYEE may *be assigned* zero, one or more PROJECTs
Each PROJECT may *be assigned to* zero, one or more EMPLOYEEs

Now we can create the updated data model . First we will write down the relationships

Each DEPARTMENT may *employ* zero, one or more EMPLOYEEs
Each EMPLOYEE must *work for* one and only one DEPARTMENT
Each JOB may *be assigned to* zero, one or more EMPLOYEEs
Each EMPLOYEE may *be assigned* zero or one JOB
Each EMPLOYEE (Manager role) may *supervise* zero, one or more EMPLOYEEs (Employee role)
Each EMPLOYEE (Employee role) may *be supervised by* zero or one EMPLOYEE (Manager role)
Each EMPLOYEE may *be assigned* zero, one or more PROJECTs
Each PROJECT may *be assigned to* zero, one or more EMPLOYEEs

Advanced Data Modelling using ERD



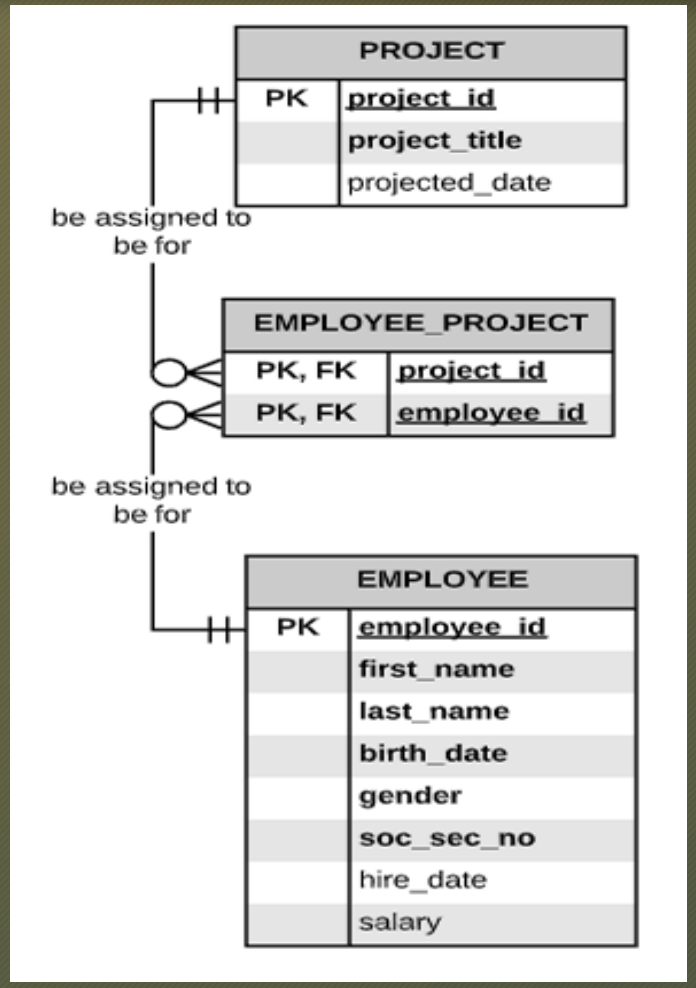
Advanced Data Modelling using ERD

Eliminating Many-To-Many Relationships

Many-to-many relationships add complexity and confusion to the model and to the application development process. Such relationships will be a problem later when the related entities are implemented into database tables because two entities cannot be children of each other. There is no place to put the foreign keys

The many-to-many relationship problem can be solved by introducing a new entity, called intersection, junction or associative entity

Advanced Data Modelling using ERD

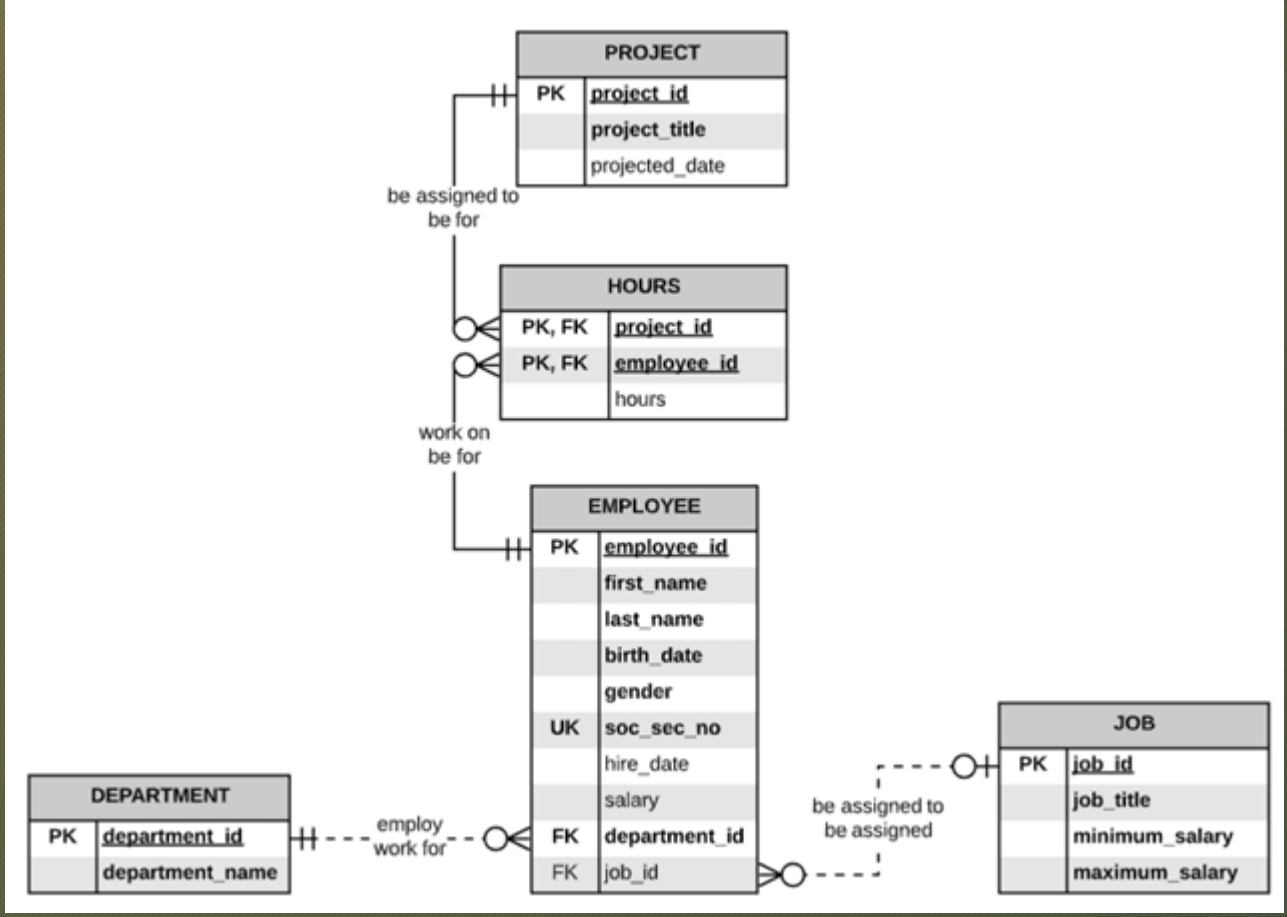


Each EMPLOYEE may *work on* zero, one or more EMPLOYEE_PROJECTs
Each EMPLOYEE_PROJECT must *be for* one and only one EMPLOYEE
Each PROJECT may *be assigned to* zero, one or more EMPLOYEE_PROJECTs
Each EMPLOYEE_PROJECT must *be for* one and only one PROJECT

Store the data in intersection entity

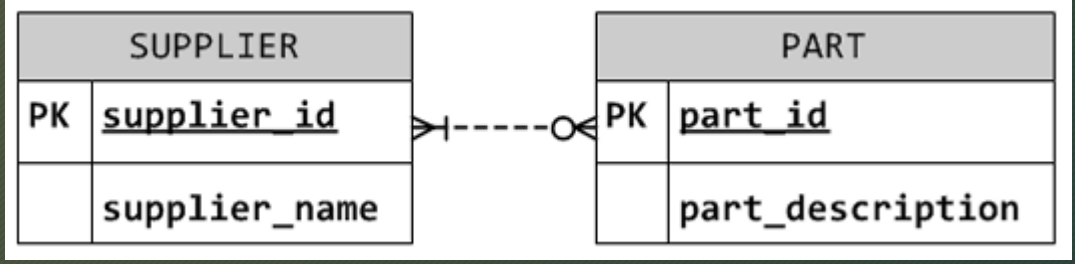
Lets say the company wants to store the hours worked by each employee. Based on the data model so far where would the hours be stored? Since each employee can work on more than one project at a time and each project can have many employees assigned to it, the hours needs to be stored in the intersection entity. It seems logical to change in intersection entity name to HOURS

Advanced Data Modelling using ERD



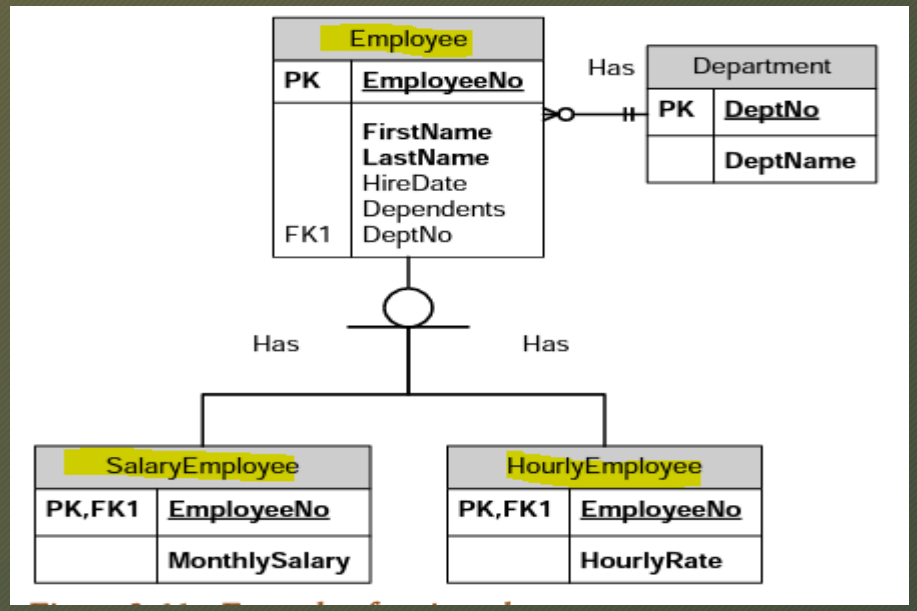
Each DEPARTMENT may *employ* zero, one or more EMPLOYEEs
Each EMPLOYEE must *work for* one and only one DEPARTMENT
Each JOB may *be assigned to* zero, one or more EMPLOYEEs
Each EMPLOYEE may *be assigned* zero or one JOB
Each EMPLOYEE may *be assigned to* zero, one or more EMPLOYEE_PROJECTs
Each EMPLOYEE_PROJECT must *be for* one and only one EMPLOYEE
Each PROJECT may *be assigned to* zero, one or more EMPLOYEE_PROJECTs
Each EMPLOYEE_PROJECT must *be for* one and only one PROJECT

Task :Resolve the following M:N Relationship



Advanced Data Modelling using ERD

Super Type and Sub Type: An entities are identified , also identify entity hierarchies. For example if some parts are produced on-site and others are purchased , both are part of the PART entity. As another example SalaryEmployee and HourlyEmployee are sub types of EMPLOYEE.A subtype has all the attributes and integrity rules of its supertype



Advanced Data Modelling using ERD

Modelling Terminology

Naming Transformation from Logical Data Model to Relational Data Model

- An entity becomes a table
- An instance becomes a row
- An attribute becomes a column
- A primary unique identifier (UID) becomes a primary key
- A secondary unique identifier becomes a unique key
- A relationship becomes a foreign-key column and a foreign key constraint
- Column data types are defined
- A business rule become a constraints

Terminology mapping		
Logical data model	→	**Relational data model**
Entity	→	Table
Instance	→	Row
Attribute	→	Column
Unique Identifier (UID)	→	Primary Key
Secondary Unique Identifier	→	Unique Key
Relationship	→	Foreign Key
Business Rules or Constraints	→	Constraints

Figure 3-14 Technology mapping from logical data model to relational data model

Advanced Data Modelling using ERD

	Conceptual	Logical	Relational
Entity Name	✓	✓	
Entity Relationships	✓	✓	
Attributes		✓	
Unique Identifier		✓	
Primary Key			✓
Foreign Key		✓	✓
Table Name			✓
Column Name			✓
Column Data Type			✓

Figure 3-15 Comparison of data modeling naming convention

Advanced Data Modelling using ERD

Data TypesA base tables, usually referred to as table, is a database object that permanently store data. In SQL terms, the data is stored in columns in database tables. These columns are often refereed to as fields. Every column or field in a base table must include a data type that specifies the type of data that will be stored as well as a size that indicates the maximum length of data

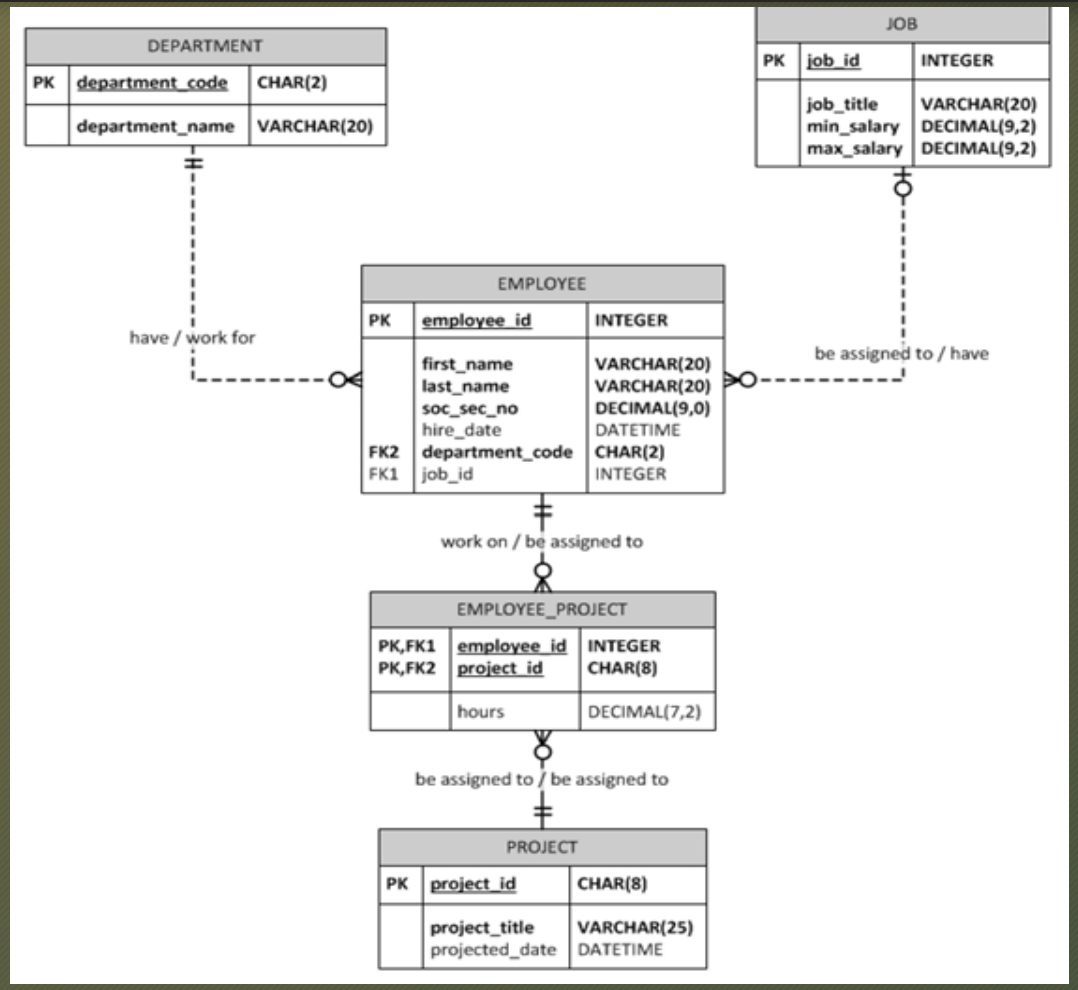
Data type	Description
VARCHAR(n) or CHARACTER VARYING(n)	• Variable-length character string that stores one character in each byte • Length (n) is specified as an integer in parentheses • Stores only the actual character string. For example, if a character string of 65 characters is stored in VARCHAR(225), it occupies only 65 bytes • Stores any printable character (alphabetic characters, special characters and numeric values) • If no length is specified, the default length of 1 is used
CHARACTER(n) or CHAR(n)	• Same as VARCHAR, except stored as a fixed-length character string with a fixed-length of n characters • If a character string of 65 characters is stored in CHAR(225), it occupies 225 characters (65 characters for the actual data plus 160 blank spaces)

Advanced Data Modelling using ERD

`DECIMAL(p,s)` or `NUMERIC(p,s)`	• A numeric value that can have a decimal point • The size argument has two parts: precision (p) and scale (s) • Precision is the total number of digits • Scale is the number of digits to the right of the decimal point, and can range from zero (0) to the value specified for precision • Precision comes first, and a comma must separate from the scale argument • Example: `DECIMAL(9,2)` holds a numeric value that is 9 digits in length with 7 digits before the decimal and 2 digits after the decimal • If scale is not specified, it defaults to zero
`INTEGER`	• A numeric value without a decimal point • Stored in binary format • Stored as a length of 4-bytes binary with a precision of 10 digits
`SMALLINT`	• Same as `INTEGER` except holds a smaller range of values • Stored as a length of 2-bytes binary with a precision of 5 digits

Data type	Description
`BIGINT`	• Same as `INTEGER` except holds a larger range of values • Stored as a length of 8-bytes binary with a precision of 19 digits
`DATE`	• Stores year, month, and day in a 10 character format • Enclosed in single quotation marks and has the form, `YYYY-MM-DD` (for example, '`2020-05-15`' is May 15, 2020)
`Boolean`	• Some DBMSs do not support a Boolean data type. However, a `CHECK` constraint can be used to simulate a `BOOLEAN` data type. This is illustrated in chapter 6.

Advanced Data Modelling using ERD



Each DEPARTMENT may *have* zero, one, or more EMPLOYEEs
Each EMPLOYEE must *work for* one and only one DEPARTMENT
Each JOB may *be assigned to* zero, one, or more EMPLOYEEs
Each EMPLOYEE may *have* zero or one JOB
Each EMPLOYEE may *work on* zero, one, or more EMPLOYEE_PROJECTs
Each EMPLOYEE_PROJECT must *be assigned to* one and only one EMPLOYEE
Each PROJECT may *be assigned to* zero, one, or more EMPLOYEE_PROJECTs
Each EMPLOYEE_PROJECT must *be assigned to* one and only one PROJECT

Conclusion