DATABASE SYSTEM LAB 7

ENTITY RELATIONSHIP DIAGRAM

I. ERD REVIEW

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ERDs are extremely important in database design and projects that require a clear structure of all data — think of it as the standardized way to draw a database diagram. By applying this standard, your team can easily understand the structure of a database or the information you collect within your system.

1.1 Components of ER Diagram

You base an ER Diagram on three basic concepts:

Entities

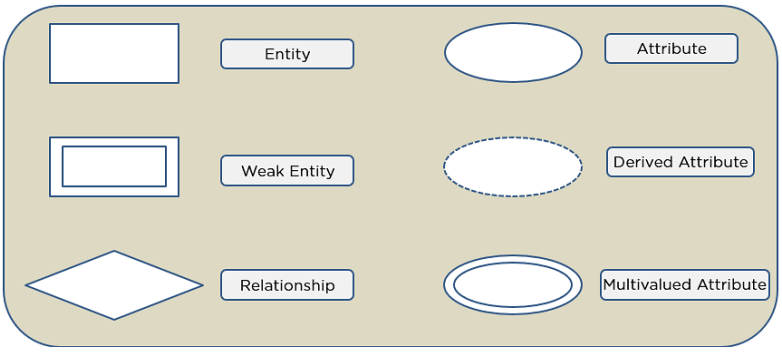
- Weak Entity

Attributes

- Key Attribute
- Composite Attribute
- Multivalued Attribute
- Derived Attribute

Relationships

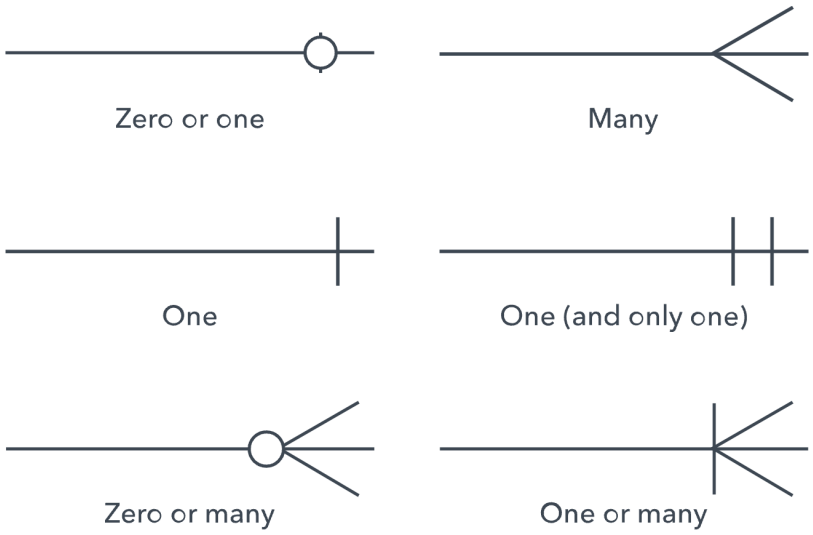
- One-to-One Relationships
- One-to-Many Relationships
- Many-to-One Relationships
- Many-to-Many Relationships



1.2 Cardinality

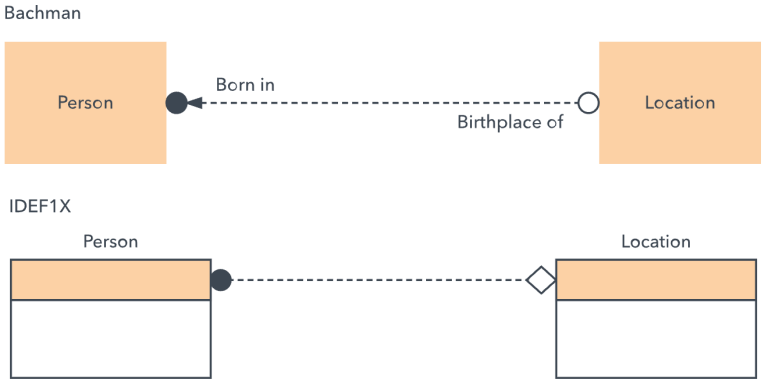
Defines the numerical attributes of the relationship between two entities or entity sets. The three main cardinal relationships are one-to-one, one-to-many, and many-many.

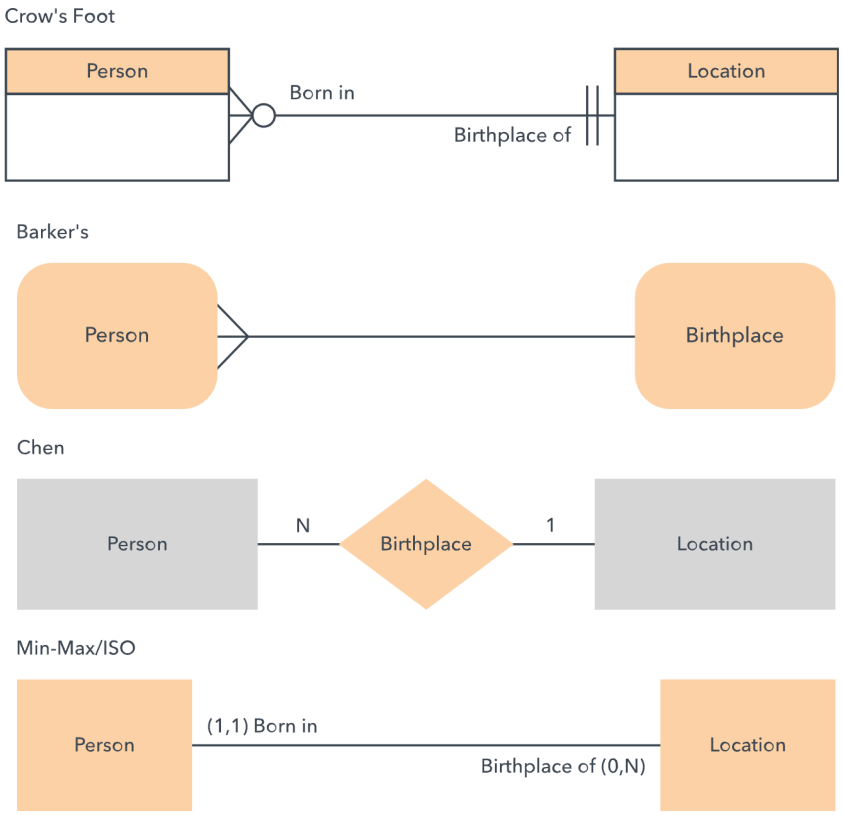
- **A one-to-one** example would be one student associated with one mailing address.
- **A one-to-many** example (or many-to-one, depending on the relationship direction): One student registers for multiple courses, but all those courses have a single line back to that one student.
- **Many-to-many** example: Students as a group are associated with multiple faculty members, and faculty members in turn are associated with multiple students.



1.3 Examples

Following are examples of ERD diagrams made in each system.



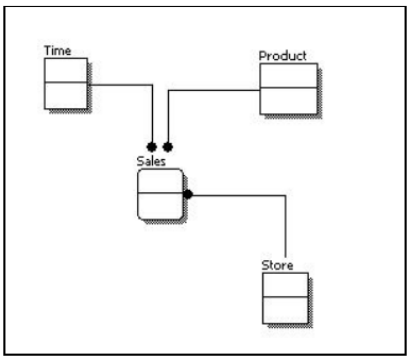


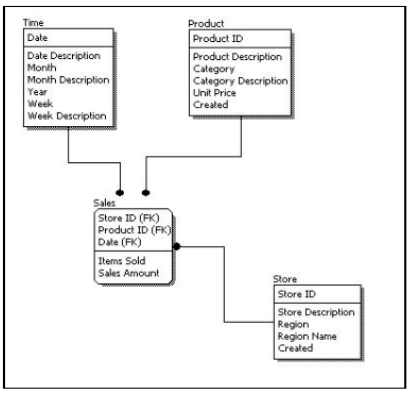
1.4 Conceptual, logical and physical data models

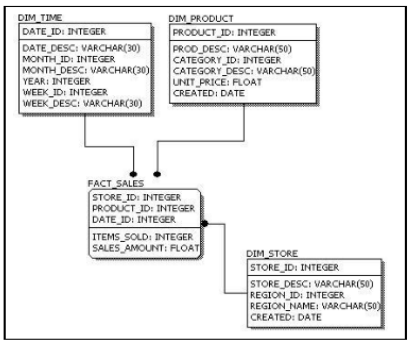
ER models and data models are typically drawn at up to three levels of detail:

- **Conceptual data model**: The highest-level view containing the least detail. Its value is showing overall scope of the model and portraying the system architecture. For a system of smaller scope, it may not be necessary to draw. Instead, start with the logical model.
- **Logical data model**: Contains more detail than a conceptual model. More detailed operational and transactional entities are now defined. The logical model is independent of the technology in which it will be implemented.
- **Physical data model**: One or more physical model may be developed from each logical model. The physical models must show enough technology detail to produce and implement the actual database.

Conceptual Model Design Logical Model Design Physical Model Design







Note that similar detail and scope levels exist in other types of diagrams, such as data flow diagrams, but that it contrasts with software engineering's three schema approach, which divides the information a bit differently. Sometimes, engineers will branch out ER diagrams with additional hierarchies to add necessary information levels for database design. For example, they may add groupings by extend up with superclasses and down with subclasses.

A logical data model may contain one or more many-to-many relationships. Physical data modelling techniques transform a many-to-many many-relationships into one-to many-relationships by adding additional tables. These are referred to as **bridge tables**.

II. EXERCISES

Requirements

1. The **draw the ERD** model for each of the following exercises.
2. **Write sql script** to create a table corresponding to the drawn erd model.
3. Expected output for each exercise: an **image** file and a **sql** file.

Drawing Tools

- https://draw.io/
- StarUML
- https://www.lucidchart.com/
- https://online.visual-paradigm.com/

Exercise 1

A company needs to manage employee information including ID number, full name, year of birth and phone number. Each employee belongs to a specific department, the department information includes the department number, the department name, and the department's location. Each department has only one head of the department, one employee can be the head of at most one department.

The company does many different projects, each of which belongs to a certain department. Information of each project includes project code, project name, deadline and description. A project can have one or more employees involved, then the company needs to keep track of the number of hours each employee works on the projects.

In addition, the company also needs to save information of employees' relatives. Their information includes full name, gender, relationship with the employee.

Exercise 2

A manufacturing company produces products. The following product information is stored: product name, product ID and quantity on hand. These products are made up of many components. Each component can be supplied by one or more suppliers. The following

component information is kept: component ID, name, description, suppliers who supply them, and products in which they are used.

Create an ERD to show how you would track this information. Show entity names, primary keys, attributes for each entity, relationships between the entities and cardinality.

Assumptions

- A supplier can exist without providing components.
- A component does not have to be associated with a supplier.
- A component does not have to be associated with a product. Not all components are used in products.
- A product cannot exist without components.

Extercise 3

A hospital has a large number of registered physicians. Attributes of physicians include Physician_ID (the identifier) and Specialty. Patients are admitted to the hospital by physicians. Attributes of PATIENT include Patient_ID (the identifier) and Patient_Name. Any patient who is admitted must have exactly one admitting physician. A physician may optionally admit any number of patients. Once admitted, a given patient must be treated by at least one physician. A particular physician may treat any number of patients, or may not treat any patients. Whenever a patient is treated by a physician, the hospital wishes to record the details of Treatment_Detail include Date, Time, and Results.

Exercise 4

Create an ERD for a car dealership. The dealership sells both new and used cars, and it operates a service. Base your design on the following business rules:

- A salesperson may sell many cars, but each car is sold by only one salesperson.
- A customer may buy many cars, but each car is bought by only one customer.
- A salesperson writes a single invoice for each car he or she sells.
- A customer gets an invoice for each car he or she buys.

- A customer may come in just to have his or her car serviced; that is, a customer need not buy a car to be classified as a customer.
- When a customer takes one or more cars in for repair or service, one service ticket is written for each car.
- The car dealership maintains a service history for each of the cars serviced. The service records are referenced by the car's serial number.
- A car brought in for service can be worked on by many mechanics, and each mechanic may work on many cars.
- A car that is serviced may or may not need parts (e.g., adjusting a carburetor or cleaning a fuel injector nozzle does not require providing new parts).