Database management systems: Oracle, MYSQL, MSSQL (öğrenicez)

Metadata: Data about data.

Entities: Composed of attributes. Mesela projedeki student ve course objeleri classları birer entity idi. Bunların arasındakine de relationship denir.

Relationships: one-to-many (1:N), many-to-many (M:N), one-to-one (1:1)

**The Database Development Process**

**SDLC (System Development Life Cycle)**

* Detailed process
* Time consuming
* Scrum, waterfall etc..

1. Planning: Understanding of business situtation. Data modeling.
2. Analysis: Functional requirements. Detailed conceptual data modeling.
3. Design: Logical and physical database design.
4. Implementation: Writing programs, testing and documenting.
5. Maintenance: Monitoring, repairing.

**Prototyping (Rapid Application Development (RAD))**

1. Identify problem
2. Initial prototype
3. READ THE FIRST CHAPTER

**E-R Model Constructs (Entities-Relationship)**

* **Entities**: Entity Type: Collection of entities. Entity Instance: A single occurence of an entity type.
* **Relationships**: Connection between entities.
* **Attributes**: Futures, properties etc..

E-R diagramda entityler dikdörtgen içinde yazılır. Aralarında çizgi varsa relationship olur.

Bağlantıların üstündeki çizgiler mandatory, yuvarlaklar optional demektir. Tek çizgi one, çok çizgi many demek.

**Good Business Rules**

* Declarative
* Precise
* Atomic
* Consistent
* Expressible
* Distinct
* Business-oriented

Note: An object should have many instances.

**Strong Entity**

* Exists independently
* Has its own unique identifier
* Studenti employee etc..

**Weak Entity**

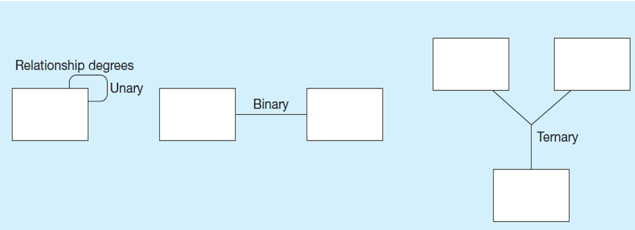
* Dependent on a strong entity
* Does not have a unique identifier
* Entity box and partial identifier have double lines
* Children of strong entity. (Inheritance maybe?)

**Identifying Relationship**

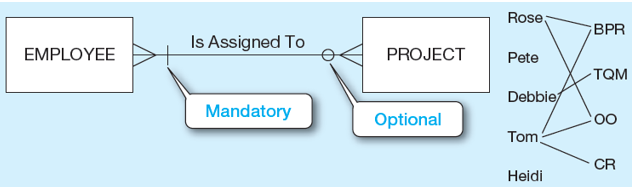
* Links strong entities to weak entities

**Degree of Relationships**

* **Unary Relationship:** Entities of the same entity type related to each other.
* **Binary Relationship**
* **Ternary Relationship**



Note: Bi tane tek çizgi mandatory, “o” işareti optional.



Here every project must have at least one employee assigned to it, but it is possible for an employee not to be assigned to any projects.

**Associative Entities**

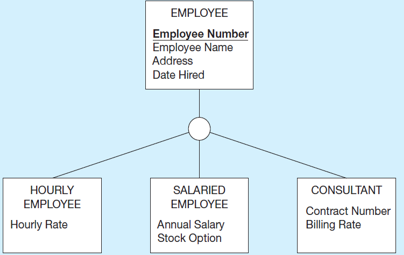
* Relationships should be many
* Could have meaning independently
* Has a unique identifier, should also have other attributes.
* Convert ternary relationships to associative entities. (WOULD BE BETTER)
* THINK ABOUT PROFESSOR, COURSE AND SCHEDULE. (Schedule is an associative entity)

Note: If there are two associative entity, choose the one with the identifier, it would fit better!!!

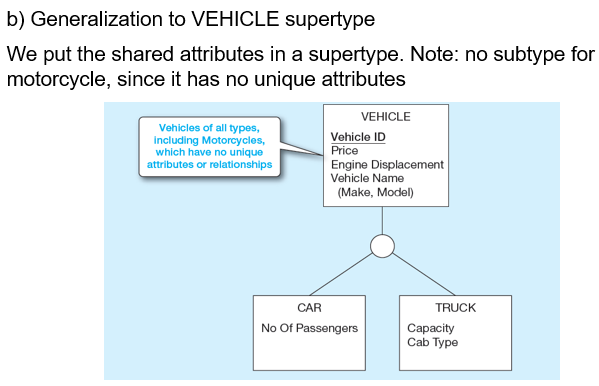
Multivalued ve composite farkına bak !!!

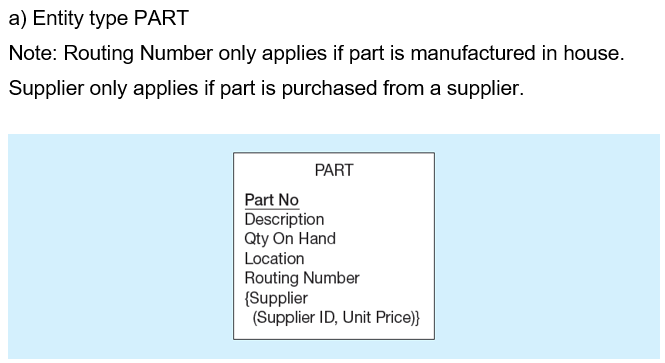
**Supertypes and Subtypes**

* Enhanced E-R (EER) model: Extend original E-R with new modeling constructs
* Subtype: A subgrouping of the entities in an entity type that has attributes. Inherits values of all attributes and relationships of the supertype. (Subclass)
* Supertype: A generic entity type that has a relationship with one or more subtypes. An instance of a subtype is also an instance of the supertype. (Superclass)



**Generalization and Specialization**





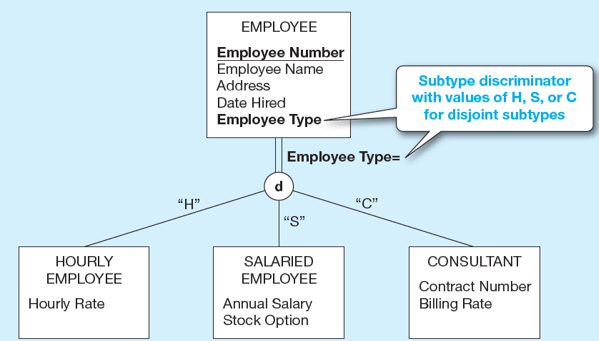
**Total and Partial Specialization**

* **Total Specialization:** An object has to be a subtype. For example a patient has to be either an Outpatient or a Resident Patient. Shown with double line.
* **Partial Specialization:** An object doesn’t have to be either. Shown with single line.

**Disjoint and Overlap**

* **Disjoint Rule:** An instance of the supertype can be only one of the subtypes. Shown with “d” in a circle. A simple attribute with alternative values to indicate the possible subtypes.
* **Overlap Rule:** An instance of the supertype could be more than one of the subtypes. Show with “o” in a circle. A composite attribute whose pertain to different subtypes.

**Subtype Discriminator:** An attribute of the supertype whose values determine the target subtypes.



**Entity Clusters:** Set of one or more entity types and associated relationships grouped into a single abstract entity type.

**Relation**

* A relation is a named, two-dimensional table of data.
* Consists of rows (records) and columns (attribute or field).
* It must have a unique name.
* Every attribute value must be atomic (not multivalued, not composite).
* Every row must be unique with different values.
* Columns must have unique names.
* Order of the rows and the columns must be irrelevant.

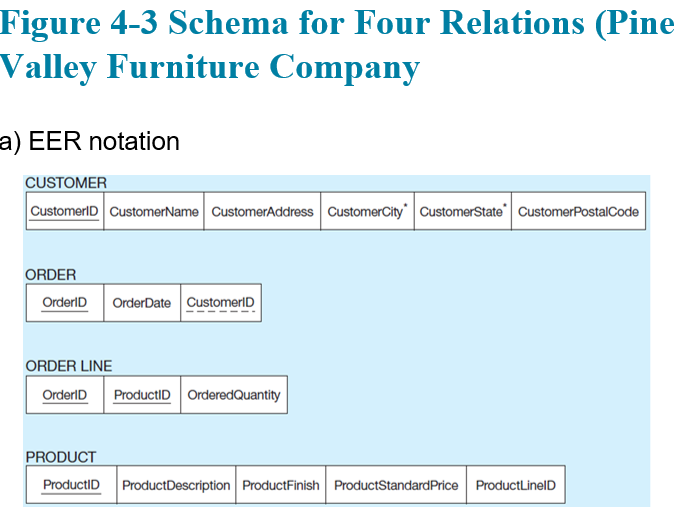
**Note**: All relations are in 1st Normal form.

**Key Fields**

* Keys can be simple or composite.

**Primary Keys:** Unique identifiers of the relation. This guarantees that all rows are unique.

**Foreign Keys:** Identifiers that enable a dependent relation to refer to its parent relation.



**Integrity Constraints**

* **Domain Constraints:** Allowable values for an attribute.
* **Entity Integrity:** No primary key attribute may be null.
* **Referential Integrity:** Rules that maintain consistency between the rows of two related tables. Any foreign key value MUST match a primary key value in the relation of the one side. (Or it can be null). Referential integrity constraints are drawm via arrows from depentent to parent side.
  + **Restrict:** Don’t allow delete of “parent” side if related rows exist in “dependent” side.
  + **Cascade:** Automatically delete “dependent” side rows that correspond with the “parent” side row to be deleted.
  + **Set-to-null:** Set the foreign key in the dependent side to null if deleting from the parent side. (not allowed for weak entities)