**Modelling & SQL**

**Terminology:**

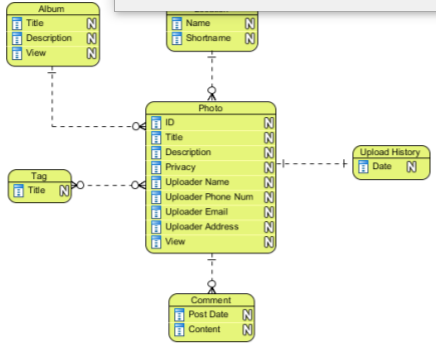
* A student as an **entity** has a number of **attributes** (ID, Name).
* A student as a **relation**: Student (ID, Name) has the attributes/columns (ID, Name).
* A student **has** a Tutor (that’s a **relationship**).

**Data Modelling** is how the “things” of importance to the system relate to each other. This can be represented as UML class diagrams.

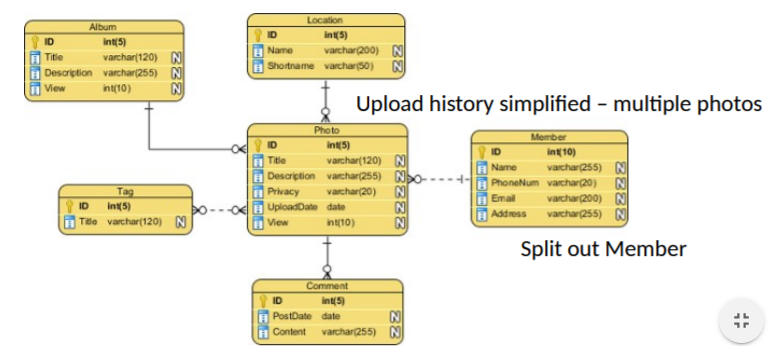
There are **three types of data modelling:**

* **Conceptual** -> Ideas
* **Logical** -> High Level Design
* **Physical** -> Low Level

**Conceptual model:** Can build it directly from the requirements and the domain. At this stage, there is no though of database design. Just create entity names and relationships and sometimes their attributes.

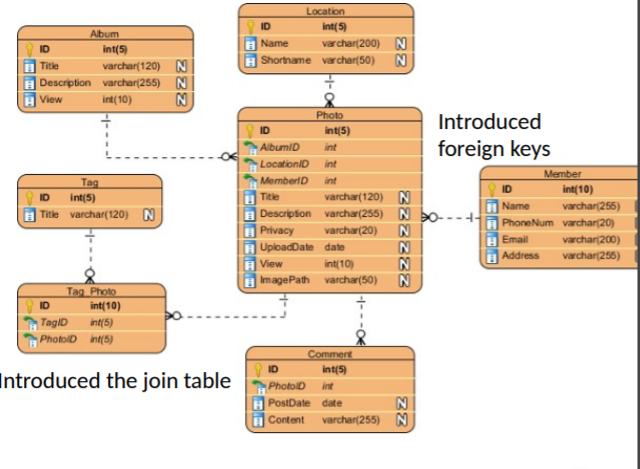


**Logical Model:** Identification of the attributes & identification of attribute types.



**Physical Model**:

* Consideration of database structure.
* Actual tables and fields.
* Implementation of relationships (e.g. keys, join tables), indexes, etc.



**From Logical to Physical design:**

From the high-level design (concept and ideas) to the low-level design (implementation).

Logical modelling is for attributes used in the real world. You shouldn’t be making attributes up in the logical stages. Your natural keys uniquely identify entities in the world.

SIDE NOTE:

**Natural Key verses Surrogate Key**

When you design tables with SQL Server, a table typically has a column or a number of columns that are known as the primary key. The primary key is a unique value that identifies each record.  Sometimes the primary key is made up of real data and these are normally referred to as natural keys, while other times the key is generated when a new record is inserted into a table.   When a primary key is generated at runtime, it is called a surrogate key.   A surrogate key is typically a numeric value.

**Don’t just add ID fields…**

**Surrogate Keys: ID Fields**

* **Automatic ID fields are what we term surrogate keys**: They have no business meaning.
* Surrogate keys are attributes created and maintained **by the system** to aid in uniquely identifying an instance of an entity – they do not occur in the real world.
* They are used because natural keys aren’t generally stable, consistent or efficient to use in a database system.
* They **stand in place of the natural keys for technical purposes only**, and are added as part of the physical design – after the logical modelling and normalization.
* But you don’t need to use them everywhere.

**Primary key:** a key with attributes that are **not** **allowed** to be **null**.

**Foreign key**: an attribute of one relation that references an attribute (primary key) of another relation

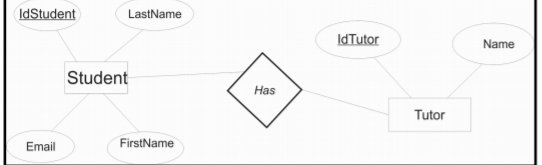
* E.g. Employee (ID, Name, **DeptNo**)
* Department (ID, Name)

**Conceptual data model:**

* Include important entities and the relationship between them.
* Do not specify attributes.
* Do not specify primary keys.
* Used as the foundation for logical data models.

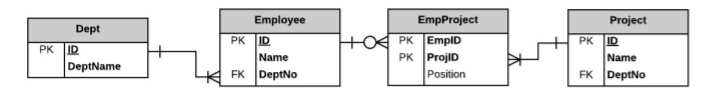
**Logical Data model:**

* Include all entities and relationships between them.
* Specify attributes for each entity.
* Specify primary key for each entity.
* Specify foreign keys, which identify the relationship between different entities.
* Involve normalisation.



**Physical data model:**

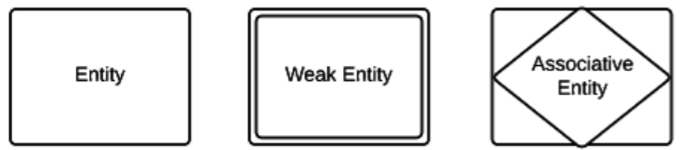
* Specify all tables and columns.
* Include foreign keys to identify relationships between tables.
* May include normalization, depending on user requirements.
* May be significantly different from the logical data model.
* May differ depending on which DBMS is used.



**Entity:**

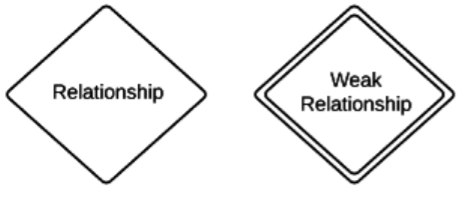
* An object we identify in our system.
* Entities have **attributes** (e.g. Employee has name/dept).
* Some of these attributes may functionally determine others.

**Conceptual model of Entities**



* **Strong entities** exist independently from other entity types. They always possess a key.
* **Weak entities** depend on some other entity type (e.g. Representing Employees in Projects using an EmployeeProject entity).
* **Associative entities** are entities that associate the instances of one or more entity types (e.g. Representing Matches played by Players from a certain Team).

**Relationships:**



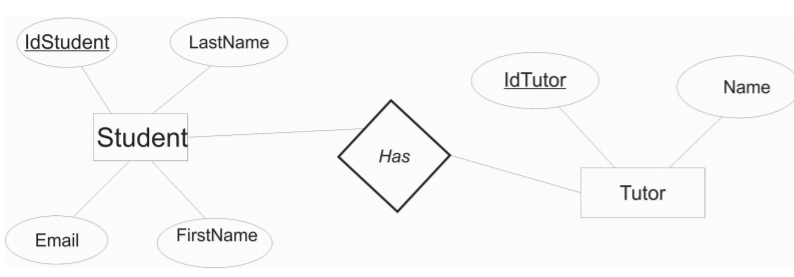
* **Relationships** are meaningful associations between or among entities. A relationship provides useful information that could **not be discerned** with just the entity types.
* **Weak relationships:** connections that exist between a weak entity type and its owner.

**Attributes: (columns)**



* **Attributes:** characteristics of either an entity, a many-to-many relationship, or a one-to-one relationship.
* **Multivalued attributes:** take on more than one value.
* **Derived attributes:** value can be calculated from related attribute values.

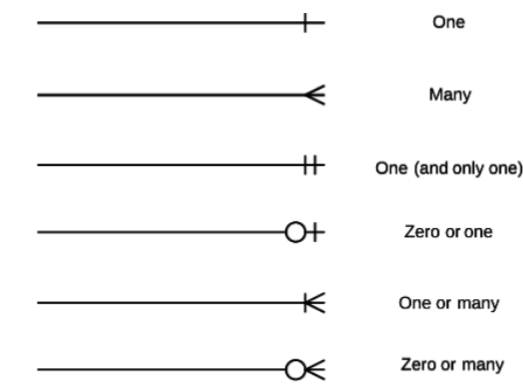
**Example conceptual to logical:**



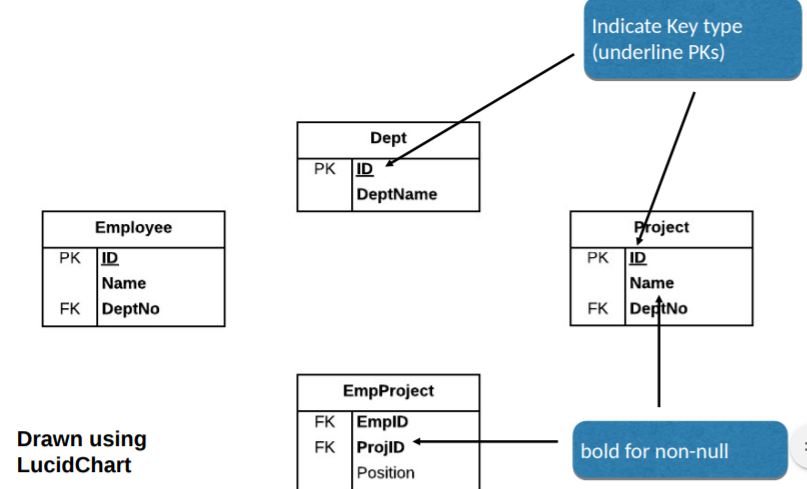
**Cardinality (MAX) and Modality (MIN):**

* **Cardinality** and **Modality** are the indicators of the business rules around a relationship.
* **Cardinality** refers to the **maximum number of times** an instance in one entity can be **associated** with instances in the related entity.
* **Modality** refers to the **minimum number of times** an instance in one entity can be **associated** with an instance in the related entity.

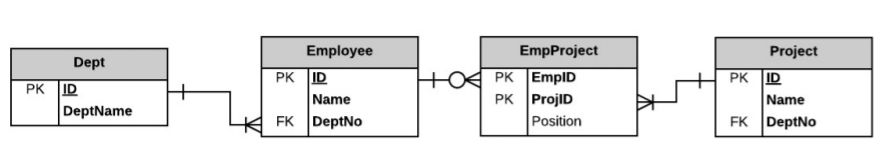
**Relationships:**



**Representing entities:**



**An an ERD (entity relationship diagram) for Research Projects:**



**SQL: Structured Query Language**

* Specifies a DDL (Data Definition Language).
  + Tables and views.
  + Convert a data model to a physical database.
* Specifies a Data Manipulation (DML)
  + Programmatic data manipulation.
  + Declarative.
  + INSERT, DELETE, UPDATE or SELECT data.
* Enforces Data integrity:
  + Referential integrity.
  + Transactions.
  + Checks keys for consistency.
* Access control: security.
* Data sharing: by concurrent users.