By Uraz C Turker

2024

Data Engineering

Nomenclature for Entity-Relationship Diagrams

1. **Entity**: An object or concept that is represented in the database. Entities can be physical (like a person or product) or abstract (like an event).
2. **Entity set**: A set of entities.
3. **Attribute**: A property or characteristic of an entity. For example, a "Customer" entity might have attributes like "CustomerID," "Name," and "Email."
4. **Primary Key**: An attribute (or a combination of attributes) that uniquely identifies each entity. For example, "StudentID" in a "Student" entity.
5. **Foreign Key**: An attribute in one entity set that refers to the primary key of another entity set, establishing a relationship between the two.
6. **Relationship**: A connection between two entities that describes how they interact with each other. Relationships can be one-to-one, one-to-many, or many-to-many.
7. **Relationship set**: A set of relations.
8. **Cardinality**: The specification of the number of instances of one entity that can or must be associated with another entity. It can be classified as:
   - **One-to-One (1:1)**: One entity of an entity set is related to one entity of another entity set.
   - **One-to-Many (1:M)**: One entity of an entity set is related to many entities of another entity set.
   - **Many-to-Many (M:N)**: Many entities of an entity set are related to many entities of another entity set.
9. **Weak Entity**: An entity that cannot be uniquely identified by its attributes alone and relies on a "strong" entity's primary key for its identification.
10. **Composite Attribute**: An attribute that can be divided into smaller sub-parts, which represent more basic attributes. For example, an "Address" attribute can be composed of "Street," "City," and "Zip Code."
11. **Derived Attribute**: An attribute whose value can be calculated or derived from other attributes. For instance, "Age" can be derived from the "Date of Birth."
12. **Multi-Valued Attribute**: An attribute that can hold multiple values for a single entity. For example, a "Phone Numbers" attribute for a "Customer" entity might contain multiple phone numbers.

Nomenclature for Participation Constraints

1. **Total Participation**:
   - **Definition**: Every entity of an entity set must be involved in at least one entity of another entity set.
   - **Notation**: Represented by a double line connecting the entity to the relationship in the ERD.
   - **Example**: In a relationship between "Course" and "Student," if every student must enroll in at least one course, then the participation of the "Student" entity in the relationship is total.
2. **Partial Participation**:
   - **Definition**: Only some entities of the entity set may be involved in the relationship.
   - **Notation**: Represented by a single line connecting the entity to the relationship in the ERD.

   - **Example**: In a relationship between "Employee" and "Project," if some employees may not be assigned to any project, the participation of the "Employee" entity is partial.

1. Referential Integrity Constraint

- **Definition**: Ensures that a foreign key value in one table must match a primary key value in another table or be null. This constraint maintains the consistency and validity of the relationships between entities.
- **Purpose**: To prevent orphaned records and ensure that relationships between entities remain valid.
- **Example**: If a "Student" entity has a foreign key that refers to a "Course" entity's primary key, every student record must reference an existing course or be null if the student is not enrolled in any course.

2. Entity Integrity Constraint

- **Definition**: Ensures that each entity instance has a unique identifier (primary key) and that this identifier cannot be null. This constraint guarantees the uniqueness of each record in a table.
- **Purpose**: To ensure that each instance of an entity can be uniquely identified and accessed.
- **Example**: In a "Customer" entity, the "CustomerID" attribute must be unique for each customer and cannot be null, ensuring that every customer can be distinctly identified.

3. Domain Constraint

- **Definition**: Specifies the permissible values for an attribute in a database. This constraint defines the data type, range, and format of values that can be stored in a particular attribute.
- **Purpose**: To enforce data integrity by restricting the values that can be entered into a database field.
- **Example**: If an "Age" attribute in a "Person" entity is defined to be an integer with a domain constraint of 0 to 120, any attempt to enter a value outside this range would violate the domain constraint.

Summary for constraints

Constraint type	Definition	Purpose	Example
Referential integrity	Foreign key must match a primary key or be null.	Maintain valid relationships between entities.	A "Student" must reference a valid "Course" or be null if not enrolled.
Entity integrity	Each entity must have a unique identifier (primary key) and cannot be null.	Ensure uniqueness of each record in a table.	A "CustomerID" must be unique and cannot be null for the "Customer" entity.
Domain integrity	Specifies permissible values for an attribute (data type, range, format).	Enforce data integrity in database fields.	An "Age" attribute must be an integer between 0 and 120.