DBMS

Week 06

Chapter 6 database design using e-r model

Database design using the Entity-Relationship (E-R) model involves creating a blueprint for how data will be organized, stored, and related in a relational database. The E-R model helps you define the entities (objects), their attributes (properties), and the relationships between them.

Overview of the Design Process

Design Phases

- Characterize User Requirements: In this initial phase, the goal is to fully understand the data needs of the prospective users of the database. This often requires extensive interaction with domain experts and users to gather requirements.
- Choose a Data Model: The next step is to select an appropriate data model that will serve as the basis for the database design. Common models include the entity-relationship model (ER), relational model, and more.
- Conceptual Schema: In this phase, the high-level requirements are translated into a conceptual schema. The schema defines entities, attributes, relationships, and constraints. Entity-relationship diagrams are often used for graphical representation.
- Functional Requirements: Alongside the conceptual schema, the functional requirements of the enterprise are identified. This includes specifying the types of operations (transactions) that will be performed on the data.
- Logical Design: The logical-design phase involves mapping the conceptual schema to the specific data model of the chosen database system. For many systems, this means mapping to the relational model.
- Physical Design: In the final phase, the physical schema of the database is defined. This includes specifying file organization, indexing structures, and other physical storage details.

Design Alternatives

- During the design process, the database designer needs to make choices and decisions to avoid common pitfalls:
  - Redundancy: Redundancy should be minimized to avoid inconsistencies. Data should ideally be stored in one place, and updates should be carefully managed to maintain consistency.
  - Incompleteness: The design should not make it difficult to model certain aspects of the enterprise. Missing entities or relationships can hinder the ability to represent the full scope of data requirements

Challenges

- Database design can be challenging due to the complexity of real-world applications.
- There are often many potential design choices, and the designer must carefully consider how to represent entities, attributes, relationships, and constraints.
- Making choices like whether a relationship should be represented as an entity or as a direct relationship between entities can have significant implications for the database's ability to accurately model the enterprise.

Good Design Principles

- A combination of scientific principles and good design practices is required to create an effective database schema.
- The goal is to represent the data requirements accurately, minimize redundancy, and avoid design flaws that could lead to inconsistencies or limitations in representing the enterprise's data needs.

The Entity-Relationship Model

Entity Sets

- An entity is a distinguishable "thing" or "object" in the real world, like a person, course, or product.
- An entity has attributes that describe its properties, and some attributes uniquely identify each entity.
- An entity set is a collection of entities of the same type, sharing the same attributes. For example, the entity set "instructor" contains all instructors' data.
- Entity sets can overlap; for instance, a "person" entity set may include both students and instructors.

Attributes

- Attributes are properties that describe each entity in an entity set.
- Attributes have values for each entity. For example, an "instructor" entity may have attributes like ID, name, and salary.

Relationship Sets

- A relationship represents an association between entities. For example, an "advisor" relationship associates instructors with students.
- A relationship set is a set of relationships of the same type.
- Relationships can have descriptive attributes that provide additional information about the relationship.
- Relationship sets are represented by diamonds in E-R diagrams, connecting related entity sets with lines.

Role Indicators

- In cases where multiple entity sets participate in a relationship set, role indicators can be used to specify how an entity participates in a relationship.
- Role names clarify the meaning of relationships, especially in recursive relationship sets where the same entity set appears in different roles.

Complex Attributes

- Attributes can be composite and multivalued, but the text focuses on simple attributes.
- Composite attributes consist of subparts (e.g., an address attribute composed of street, city, state, and ZIP).
- Multivalued attributes store sets of values (e.g., multiple phone numbers for a contact).

Degree of Relationship Sets

- The degree of a relationship set is the number of entity sets it connects. Binary relationships involve two entity sets, while ternary relationships involve three.

Ternary Relationship Set

- Ternary relationships involve three entity sets and are represented in E-R diagrams to capture complex associations.
- An example is the "proj guide" relationship involving students, instructors, and research projects.

E-R Diagrams

- E-R diagrams graphically represent the structure of a database, including entity sets, relationship sets, attributes, and their associations.
- Entities are represented by rectangles, and relationships by diamonds. Lines connect them, indicating associations.
- Descriptive attributes are represented in undivided rectangles linked to relationships with dashed lines.

Complex Attributes

Domain of Attributes

- Each attribute in a database has a domain, which is a set of permitted values for that attribute.
- For example, the attribute "course id" might have a domain consisting of all text strings of a certain length, while the attribute "semester" might have a domain of strings like {Fall, Winter, Spring, Summer}.

Attribute Types

- Simple and Composite Attributes: Simple attributes are not divided into subparts, while composite attributes are composed of several sub-attributes. For example, "name" could be a composite attribute with sub-attributes like first name, middle initial, and last name.
- Single-Valued and Multivalued Attributes: Single-valued attributes have one value per entity, while multivalued attributes can have a set of values for a specific entity. For instance, an instructor may have multiple phone numbers, making the "phone number" attribute multivalued.
- Derived Attributes: Derived attributes have values that can be computed from other related attributes or entities. For instance, "age" can be derived from the "date of birth" and the current date.

Composite Attributes

- Composite attributes allow related attributes to be grouped together, making the database schema cleaner.
- Composite attributes can have a hierarchical structure, where a composite attribute itself contains sub-attributes.
- For example, an instructor's "name" could be composed of sub-attributes like first name, middle initial, and last name.

Multivalued Attributes

- Multivalued attributes can have multiple values for a single entity.
- An example is the "phone number" attribute, where an entity like an instructor may have several phone numbers.

Derived Attributes

- Derived attributes are computed based on other attributes or entities.
- An example is the "age" attribute, which can be derived from the "date of birth" and the current date.

Null Values

- Attributes can take null values, indicating that either the value is missing, not applicable, or unknown.
- For example, if an instructor's name attribute is null, it may imply that the value is missing. A null apartment number in an address could mean it's not applicable, missing, or unknown.

Mapping Cardinalities

Mapping Cardinalities in Binary Relationships

- Mapping cardinalities are most used to describe binary relationship sets, which involve two entity sets.
- For a binary relationship set R between entity sets A and B, the mapping cardinality can be one of the following:
  - One-to-One (1:1): An entity in A is associated with at most one entity in B, and vice versa.
  - One-to-Many (1:N): An entity in A can be associated with multiple entities in B, but an entity in B can be associated with at most one entity in A.
  - Many-to-One (N:1): An entity in A can be associated with at most one entity in B, but an entity in B can be associated with multiple entities in A.
  - Many-to-Many (N:N): An entity in A can be associated with multiple entities in B, and vice versa.

E-R Diagram Representation

- In E-R diagrams, mapping cardinalities are represented using lines between relationship sets and entity sets.
- A directed line (→) represents one-way cardinality, indicating the direction of the relationship.
- An undirected line (—) represents a two-way relationship without direction.

Total vs. Partial Participation

- The participation of an entity set in a relationship set can be either total or partial.
- Total participation means that every entity in the entity set must participate in at least one relationship in the relationship set.
- Partial participation means that it's possible for some entities in the entity set not to participate in relationships in the relationship set.

Cardinality Constraints

- Lines in E-R diagrams can have associated minimum and maximum cardinality constraints in the form of l..h, where l is the minimum and h is the maximum.
- A minimum value of 1 indicates total participation, meaning each entity occurs in at least one relationship.
- A maximum value of 1 indicates an entity participates in at most one relationship.
- A maximum value of $*$ (asterisk) indicates no limit on participation.

Nonbinary Relationship Sets

- Mapping cardinalities can also be specified for nonbinary relationship sets (involving more than two entity sets).
- Care must be taken when drawing E-R diagrams for nonbinary relationships to avoid ambiguity.

Alternative Representations

- Cardinality constraints can also be represented using arrows and double lines instead of numeric values.

Primary Key

Primary Keys for Entity Sets

- In the E-R model, entities within an entity set must be uniquely identifiable based on their attributes.
- The primary key for an entity set is a set of attributes that uniquely identifies each entity within that set.
- Just like in relational databases, primary keys in the E-R model help distinguish entities from each other.
- The concepts of superkey, candidate key, and primary key, commonly used in relational databases, also apply to entity sets in the E-R model.

Primary Keys for Relationship Sets

- Relationship sets in the E-R model also need a way to distinguish individual relationships.
- The primary key for a relationship set depends on the attributes associated with that relationship set.
- If a relationship set has no attributes, then the primary key consists of the union of the primary keys of the participating entity sets.
- If a relationship set has attributes, the primary key includes the union of primary keys of participating entity sets along with those attributes.
- The choice of primary key varies based on the mapping cardinality of the relationship set (one-to-one, one-to-many, many-to-one, or many-to-many).

Weak Entity Sets

- Weak entity sets are entity sets that depend on another entity set, known as the identifying entity set, for their existence.
- Weak entities are identified using the primary key of the identifying entity set along with additional attributes called discriminator attributes.
- The identifying relationship connects the weak entity set to the identifying entity set, and it's depicted as a double diamond in E-R diagrams.
- Weak entity sets typically have total participation in the identifying relationship, meaning every weak entity is associated with an identifying entity.
- In E-R diagrams, weak entity sets are represented within a double rectangle with the discriminator attributes underlined by dashed lines.

Additional Considerations

- Weak entity sets can participate in relationships with other entity sets, and they can also participate in identifying relationships with other weak entity sets.
- In some cases, a weak entity set may have more than one identifying entity set, and its primary key is a combination of primary keys from all identifying entity sets along with discriminator attributes.

Removing Redundant Attributes in Entity Sets

Identifying Entity Sets

- Database designers start by identifying the entity sets that should be included in the database. These entity sets represent real-world entities or concepts.
- For example, in the context of a university, entity sets like "student" and "instructor" are identified.

Choosing Attributes

- After identifying entity sets, the next step is to select the appropriate attributes for each entity set.
- Attributes represent the various values and characteristics that need to be captured in the database.
- The choice of attributes is based on the designer's understanding of the enterprise's structure and requirements.
- Attributes are selected to represent information such as names, identifiers, contact details, and other relevant data.
- Attribute selection is a crucial step in the design process.

Removing Redundant Attributes

- When designing entity sets and their attributes, it's important to avoid redundancy.
- Redundant attributes are attributes that appear in multiple entity sets and can lead to data inconsistencies.
- Relationships between entity sets can help eliminate redundant attributes. For example, when two entity sets share a common attribute, the attribute can be moved to a relationship set connecting them.

Relationship Sets

- Relationship sets are established to model connections or associations between entity sets.
- Relationships describe how entities from different entity sets are related to each other.
- For instance, in a university database, there are relationship sets like "inst_dept" to relate instructors to departments and "stud_dept" to relate students to departments.

Use of Weak Entity Sets

- In some cases, an entity set might depend on another entity set for its existence. Such entity sets are called "weak entity sets."
- Weak entities are identified using the primary key of the entity that depends on them, along with discriminator attributes.
- The identifying relationship connects the weak entity set to the strong entity set it depends on.
- Weak entity sets typically have total participation in the identifying relationship, ensuring that each weak entity is associated with the strong entity.
- This approach helps avoid premature assumptions about the uniqueness of attributes.

Constraints and Completeness

- The E-R model also allows for constraints to be added to the design, such as ensuring that each instructor is associated with exactly one department or that each student has at most one advisor.
- Completeness is important in the design to ensure that all required information is captured.

E-R Diagram

- An E-R diagram is a graphical representation of the database design, showing entity sets, attributes, relationships, and constraints.
- It provides a visual overview of the database structure and serves as a blueprint for creating the database schema.

Reducing E-R Diagrams to Relational Schemas

Representation of Strong Entity Sets

- Strong entity sets with simple attributes are represented as relation schemas with distinct attributes.
- The primary key of the entity set becomes the primary key of the schema.

Representation of Strong Entity Sets with Complex Attributes

- When strong entity sets have composite attributes, each component attribute is represented as a separate attribute in the schema.
- Multivalued attributes are handled differently by creating a separate relation schema for them, using a foreign key that references the primary key of the parent entity set.

Representation of Weak Entity Sets

- Weak entity sets are represented as relation schemas.
- The primary key of the strong entity set they depend on is combined with the discriminator of the weak entity set to form the primary key of the schema.
- Foreign key constraints are added to ensure the connection between weak and strong entities.

Representation of Relationship Sets

- Relationship sets are represented as relation schemas.
- The attributes of the schema are derived from the primary keys of the participating entity sets and the descriptive attributes of the relationship.
- The primary key of the relationship set becomes the primary key of the schema.
- Foreign key constraints are added for each participating entity set to maintain referential integrity.

Redundancy of Schemas

- Some relation schemas derived from relationship sets, especially those connecting weak entity sets to strong entity sets, can be redundant and are often eliminated.

Combination of Schemas

- When there is a many-to-one relationship with total participation (all entities in A must participate in the relationship AB), schemas for entity sets A and AB can be combined into a single schema.
- Primary keys and foreign key constraints are adjusted accordingly when schemas are merged.
- Combining schemas can simplify the database design.

Handling Null Values

- If participation in a many-to-one relationship is partial, null values can be used in the combined schema to indicate missing relationships.

Handling Foreign Key Constraints

- When merging schemas, foreign key constraints related to the entity set into which the relationship schema is merged are dropped, while constraints related to other entity sets are retained.