Lesson 5: Advanced Data Modeling

The Extented Entity Relationship Model

- Created to handle more complex data structures and business rules.
- Adds more meaning (semantics) to the original ER model, such as entity supertypes, entity subtypes and entity clustering.

Entity Subtypes and Supertypes

- Sometimes data in an entity needs to be broken down further, particularly if attributes are not shared in all entity occurrences.
    - A good indicator of this is rows that look sparse, that is rows with many null values for many columns.
- **Entity Supertypes** - a generic entity that is related to one or more subtypes. Containt common attributes for all entity occurrences (rows).
- **Entity Subtypes** - contain their own unique attributes.

Specialization Hierarchy

- The depiction of entity subtypes (child) and supertypes (parent).
    - Reflective of the 1:1 relationship between the parent and children.
- In writing these hierarchies out you can use the phrase "is a" when describing the hierarchy.
- These hierarchies are not limited to a set number of levels.

Specialization Hierarchy Continued

- Support attribute inheritance.
- Define a special supertype attribute known as a subtype discriminator.
- Define disjoint/overlapping and complete/partial constraints.
- Example time!

Inheritance

- The ability of an entity subtype to inherit both attributes and relationships from their parent supertype.
    - Notably, they always inherit the supertype primary key!

Subtype Discriminator

- The attribute in the supertype entity that determines to which subtype the supertype occurrence is related.
    - The default comparison type is equality, that is "if the value of the subtype discriminator is equal to X then it is part of the Y subtype"

Disjoint/Overlapping Constraints

- Disjoint (aka non-overlapping) subtypes contain a unique subset of the supertype entity set.
    - That is, one row from the each entity supertype can only occur in one row of one entity subtype.
- Overlapping subtypes are the opposite of this, subtypes contain non-unique subsets of the supertype entity set.
    - That is, one row from the each entity supertype can occur in one row of more than one entity subtype.
- Let's review an example.

Completeness vs Partial Completeness Constraints

- A completeness constraint specifies whether each entity supertype occurrence a member of at least one subtype.
- Partial completeness enables optional membership in subtypes.

Specialization vs. Generalization

- Specializaiton is the top-down approach for identifying subtypes from the supertype(s).
- Generalization is the bottom-up approach of identifying supertypes from the subtypes(s).

Entity Clustering

- A way of simplifying the visual representation of entities and relationships by using an abstract entity type.
    - These abstract or virtual entities are no actually an entity in the final ERD.
    - See the exampoe on page 177 in your book.

More on Primary Keys

- **Natural Keys** are real-world, generally accepted identifies used to distinguish things.
    - Bank accounts
    - Social Security Numbers
    - In Ireland they had Eircode which are unique ways of identifying a geospatial locations.
- **Surrogate Keys** are created by the database designer to simplify the identification of an entity instance (row). These keys have no real-world meaning.
    - Depending on the size of the data you could use autoincrmenting numbers, but in other situations you may use UUIDs (550e8400-e29b-41d4-a716-446655440000)

Primary Keys Considerations

- Unique Values
- Nonintelligent
- No change over time
- As few attributes (or one) as possible
- Preferably numeric
- Security-compliant

Time Variant Data

- Whether and how to capture data that changes over time is dependent on the business needs.
  - Do you need all the attributes?
  - Is current data stored separately from historic data?
- In general terms, capturing changes to data is valuable because otherwise we create data loss.

Fan Traps

- Misidentification of relationships can lead to a poor data model.
- Example time

Data Definition Language (DDL)

- Recall that the internal model is the representation of the database as seen by the DBMS. DDL translates the conceptual model (our ERD) to the internal model (the DBMS).
- You use syntax like that below:

```
CREATE TABLE IF NOT EXISTS db_name.tbl_name(first_name VARCHAR(255)
                                           ,last_name VARCHAR(255)
                                           ,id INT);
```

Important Syntax

- In many DBMSs you either need to indicate your database with the USE statement.
    - If I had a database named BASEBALL and I wanted to query a table I might write the following:

```
--Identify the database to use
USE BASEBALL;

#Query the specific table
SELECT *
FROM PLAYERS;
```

Important Syntax Continued

- Sometimes it's easier to use the fully qualified name.
    - This can vary a bit by DBMS, but below is how we could write it in MariaDB.

```
/*Using the fully qualified name instead <DATABASE>.<TABLE>*/
SELECT *
FROM BASEBALL.PLAYERS;
```