- Concepts of database management systems
    - What is a database?
        - An organized collection of logically related data.
    - What is a database management system?
        - A software system that enables the use of a database approach. The primary purpose of a DBMS is to provide a systematic method of creating, updating, storing, and retrieving the data stored in a database. it enables end users and application programmers to share data, and it enables data to be shared among multiple applications rather than propagated and stored in new files for every new application. Also provides facilities for controlling data access, enforcing data integrity, managing concurrency control, and restoring a database.
    - What are the features of a DMBS?
        - See previous answer.
    - What is data independence?
        - The separation of data descriptions from the application programs that use the data.
    - What is the job of a database administrator?
        - Database administration is a technical function that is responsible for physical database design and for dealing with technical issues, such as security enforcement, database performance, and backup and recovery. A database administrator (DBA) must understand the data models built by database administration and be capable of transforming them into efficient and appropriate logical and physical database designs. The DBA implements the standards and procedures established by the data administrator, including enforcing programming standards, data standards, policies, and procedures.
    - What is the job of a database designer?
        - Make good databases?
- Concepts of the database development process
    - What are the steps in the systems development life cycle?
        - The following process is circular and iterative:
            - Planning
                - Enterprise modeling.
                - Conceptual data modeling.
                - Purpose: Develop a preliminary understanding of a business situation and how information systems might help solve a problem or make an opportunity possible.
            - Analysis
                - Conceptual data modeling (continued).
                - Purpose: To analyze the business situation thoroughly to determine requirements, to structure those requirements, and to select among competing system features.
            - Design
                - Logical database design.
                - Physical database design and definition
                - Purpose: To elicit and structure all information requirements; to develop all technology and organizational specifications.
            - Implementation
                - Database implementation.
                - Purpose: To write programs, build databases, test and install the new system, train users, and finalize documentation.
            - Maintenance
                - Database maintenance.
                - Purpose: To monitor the operation and usefulness of the system, and to repair and enhance the system.
    - What are the steps in database development?
        - (Planning) Enterprise data modeling
            - Analyze current data processing.
            - Analyze the general business functions and their database needs.
            - Justify need for new data and databases in support of business.

        - (Planning) Conceptual data modeling
            - Identify scope of database requirements for proposed information system.
            - Analyze overall data requirements for business function(s) supported by database.
        - (Analysis) Conceptual data modeling (continued)
            - Develop preliminary conceptual data model, including entities and relationships.
            - Compare preliminary conceptual data model with enterprise data model.
            - Develop detailed conceptual data model including all entities, relationships, attributes, and business rules.
            - Make conceptual data model consistent with other models of information system.
            - Populate repository with all conceptual database specifications.
        - (Design) Logical database design
            - Analyze in detail the transactions, form, displays, and inquiries (database views) required by the business functions supported by the database.
            - Integrate database views into conceptual data model.
            - Identify data integrity and security requirements, and populate repository.
        - (Design) Physical database design and definition
            - Define database to DBMS (often generated from repository).
            - Decide on physical organization of data.
            - Design database processing programs.
        - (Implementation) Database implementation
            - Code and test database processing programs.
            - Complete database documentation and training materials.
            - Install database and convert data from prior systems.
        - (Maintenance) Database maintenance
            - Analyze database and database applications to ensure that evolving information requirements are met.
            - Tune database for improved performance.
            - Fix errors in database and database applications and recover database when it is contaminated.
    - What are the deliverables from the database development process?
        - A database?
- Concepts of logical database design
    - What are the goals of database design?
        - Database design is the process of creating the structure or blueprint of stored data for an organization. The goals of database design are to produce a structure that:
        - Stores data without redundancy;
        - Can be changed relatively easily; and
        - Stores all data required for organizational processing and decision making.
    - What is a constraint?
        - A rule that cannot be violated by database users.
    - What is a primary key?
        - A key in a relational database that is unique for each record
    - What is a concatenated primary key?
        - Comprises two or more columns which serve as the primary key
    - What is a natural vs. surrogate primary key?
        - A “**natural**” primary key is composed of attributes that are already part of the application specification.
        - A “**surrogate**” primary key is created by the database designer for the explicit purpose of being a primary key.
        - A surrogate primary key should NEVER be concatenated.
        - For logical database design, try and find a natural primary key.
    - What is a business rule?
        - A statement that defines or constrains some aspect of the business. Intended to assert business structure or to control or influence the behavior of the business
    - What is a foreign key?

        - In a 1:M relationship, the foreign key is placed in the entity on the “many” side of the relationship.
    - What are the guidelines for creating a "good" data name?
        - Relate to business, not technical (hardware or software) characteristics
        - Be meaningful
        - Be unique
        - Be readable
        - Be composed of words taken from an approved list
        - Be repeatable
        - Follow a standard syntax
    - How are business rules displayed on an ERD?
        - Relationships
    - What is an entity?
        - **Entities** are nouns that describe the person, event, place, or thing about which we want to store data.
        - The name of an entity is singular, not plural.
        - An entity usually becomes a table in a database.
        - An entity instance is a single occurrence of an entity (think a row in a table, or a record in a file).
    - attribute?
        - A property or characteristic of an entity or relationship type that is of interest to the organization
    - relationship?
        - Entities do not usually exist in isolation.
        - A connecting line between two entities on an ERD represents a relationship.
        - A relationship can be depicted as a diamond or as a simple line.
        - A relationship is a natural business association existing between two or more entities.
        - A verb phrase describes the relationship.
    - cardinality?
        - **Cardinality** is a constraint on the number of entity instances that participate in a relationship.
        - Cardinality describes the minimum and maximum number of instances that one entity has with another entity in a relationship.
        - Cardinality also describes whether the relationship is mandatory or optional.
        - We use the crow’s foot notation to depict cardinality in our ERD’s in this class.
    - What is a composite attribute?
        - An attribute that has meaningful component parts (attributes). eg. address
    - What is data redundancy?
        - Repeated data. Leads to potential of anomalies (especially for text attributes)
    - How does completing a database design help you understand questions needed to be clarified with a subject matter expert (user/client)?
    - What is the difference between stored and derived data?
        - **Stored data** is info you can store in its most granular form
        - **Derived data** is info not in granular form. We never store this data.
    - What is the difference between a strong and weak entity?
        - A **strong entity** is an entity that exists whether or not there is a relationship between it and another entity.
        - A **weak entity** is an entity that relies on the existence of another entity.
            - Does not exist in “real life” – is most often a data modeling requirement, rather than a real data requirement.
    - What is the difference between a parent and child entity?
        - **Child** entity has only meaning in context of parent (weak entity)
        - **Parent** entity is a strong entity
    - What is the difference between an identifying and non-identifying relationship?
        - **Identifying relationship** is where the foreign key is part of the primary key (solid line)
        - **Non-identifying relationship** a relationship where the foreign key is not part of the primary key (dashed line)
    - What are the “rules” of the relational data model?
    - What is a unary relationship?
        - A relationship between instances of a single entity type

    - binary relationship?
        - A relationship between the instances of two entity types
    - Ternary relationship?
        - A simultaneous relationship among the instances of three entity types
    - N-ary relationship?
        - A relationship between n entities
    - What is a 1:1 relationship vs. 1:m relationship vs. m:n relationship?
        - **1:1 relationships** are usually secondary, rather than primary relationships.
            - A 1:1 relationship might indicate that the designer should simply create one entity, rather than two entities.
            - Not trying to create more entities than necessary.
            - Examine all 1:1 relationships. Determine whether the attributes could all be placed in just one of the two entities eliminating the need for the relationship.
            - In a 1:1 relationship, the foreign key can be placed in either entity. Put the foreign key in the entity that seems most “reasonable” and also will result in the fewest number of null values.
        - 1:m relationship
            - In a 1:M relationship, the primary key of the entity on the “1” side of the relationship is added to the entity on the “M” side of the relationship.
            - Once added to the “M” side of the relationship, it is called a “foreign key” in that entity.
        - m:n relationship
            - M:N relationships always produce data redundancy.
            - Divide M:N relationships with an intersection entity that will create at least two 1:M relationships.
            - A single intersection entity can be used to intersect more than two strong entities.
    - What is a timestamp?
        - A time value that is associated with a data value, often indicating when some event occurred that affected the data value
    - How is time-dependent data modeled?
        - The use of simple time stamping is often adequate for modeling time dependent data
    - What is an intersection entity?
        - An “intersection” entity is a weak entity, because it doesn’t exist unless required to divide a m:n relationship in a database.
        - The designer eliminates many-to-many relationships from the data design by creating an “intersection” entity. This is also called a “bridge” entity.

An **ERD** contains:

- Entities.
- Attributes.
- Identifiers (primary key for each entity).
- Relationships.
- Cardinalities of the relationships.
- Foreign keys to support the relationships.

Avoid redundant data that is composed of long alphanumeric data types.
Examples are names, addresses, comments, notes. Standardize any “descriptive” attributes such as categories or types.
Never add more attributes than are absolutely necessary to create a primary key value unique.
Use the primary key of the related strong entity as part of the primary key for the associative entity, if the relationship with the associative entity is mandatory.
Use the primary keys of the strong entities as part or all of the primary key for the related intersection entity.

- Logical data model
    - (Mid-level) Includes PK and FK
- Conceptual data model
    - (High-level) Doesn’t always have PK, FK, or even cardinalities
- Physical data model
    - (Low level) Absolute detail needed to create the database