



Course MANUAL

INFORMATION MANAGEMENT

This course provides students with the theoretical knowledge and practical skills in the utilization of databases and database management systems in the ICT applications. The logical design, physical design and implementation of relational databases are all covered in this course.



INFORMATION
TECHNOLOGY

JULIE ANNE ANGELES-CRYSTAL
ROGER C. PRIMO JR.
MARK JAMES G. CAYABYAB
JOHN IVAN C. MAURAT

Contents

FOREWORD

CHAPTER 1

Topic 1

02

Topic 2

03

Topic 3

04

Topic 4

05

Topic 5

06

Topic 6

07

Topic 7

- INTRODUCTION RO SQL
- DATA DEFINITION COMMANDS
- DATA MANIPULATION COMMANDS
- SELECT QUERIES



MODULE GOALS

Data Modelling and Data Models

The Importance of Data Models

Data Model Basic Building Blocks

Business Rules

Discovering Business Rules

The Evolution of Data Models

Degrees of Data Abstraction

FLEX Course Material



Database Models

WEEK 2 -3 MODULE

Education that works.



2.1

Data Modelling and Data Models

Data modeling is a process used to define and organize data requirements for a system or application.

It involves creating abstract representations of the data and the relationships between different data elements.

The goal of data modeling is to ensure that data is accurately and efficiently stored, retrieved, and managed.



KEY CONCEPTS

1.Entities:

1. An entity is a real-world object or concept that has data stored about it. For example, in a university database, entities could include students, courses, and professors.

2.Attributes:

1. Attributes are the properties or characteristics of entities. For a "Student" entity, attributes could include "StudentID," "Name," and "DateOfBirth."

3.Relationships:

1. Relationships describe how entities are related to each other. For instance, a "Student" entity may have a "Takes" relationship with a "Course" entity, indicating that a student takes one or more courses.

4.Keys:

1. Keys uniquely identify each record (row) in a table. A primary key is a unique identifier for an entity, and a foreign key is a reference to a primary key in another table.

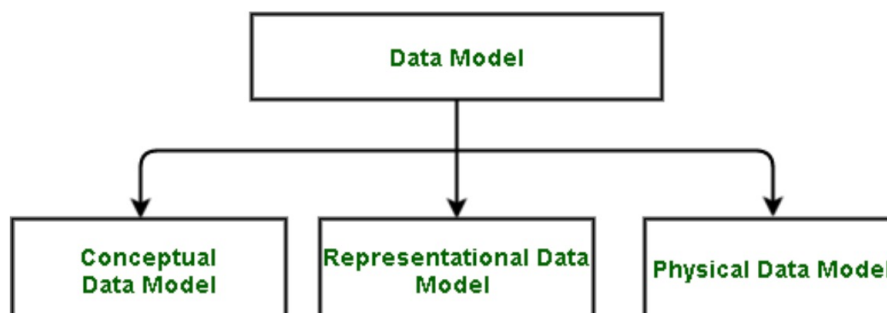


TYPES OF DATA MODELS:

- A data model is a visual or mathematical representation of the structure of a database. There are different types of data models, and each serves a specific purpose.
- A Data Model in Database Management System (DBMS) is the concept of tools that are developed to summarize the description of the database.
- Data Models provide us with a transparent picture of data which helps us in creating an actual database. It shows us from the design of the data to its proper implementation of data.

• Types of Relational Models

- Conceptual Data Model
- Representational Data Model
- Physical Data Model





CONCEPTUAL MODEL

The conceptual data model describes the database at a very high level and is useful to understand the needs or requirements of the database. It is this model, that is used in the requirement-gathering process i.e. before the Database Designers start making a particular database. One such popular model is the entity/relationship model (ER model).

The E/R model specializes in entities, relationships, and even attributes that are used by database designers. In terms of this concept, a discussion can be made even with non-computer science(non-technical) users and stakeholders, and their requirements can be understood.

- **Entity-Relationship Model(ER Model):** It is a high-level data model which is used to define the data and the relationships between them. It is basically a conceptual design of any database which is easy to design the view of data.
- **Components of ER Model:**
 - 1.**Entity:** An entity is referred to as a real-world object. It can be a name, place, object, class, etc. These are represented by a rectangle in an ER Diagram.
 - 2.**Attributes:** An attribute can be defined as the description of the entity. These are represented by Eclipse in an ER Diagram. It can be Age, Roll Number, or Marks for a Student.
 - 3.**Relationship:** Relationships are used to define relations among different entities. Diamonds and Rhombus are used to show Relationships.



CHARACTERISTICS OF CONCEPTUAL MODEL

- Offers Organization-wide coverage of the business concepts.
- This type of Data Models are designed and developed for a business audience.
- The conceptual model is developed independently of hardware specifications like data storage capacity, location or software specifications like DBMS vendor and technology. The focus is to represent data as a user will see it in the “real world.”
- Conceptual data models known as Domain models create a common vocabulary for all stakeholders by establishing basic concepts and scope



REPERESANTATION DATA MODEL

- This type of data model is used to represent only the logical part of the database and does not represent the physical structure of the database. The representational data model allows us to focus primarily, on the design part of the database.
- A popular representational model is a Relational model. The relational Model consists of Relational Algebra and Relational Calculus. In the Relational Model, we basically use tables to represent our data and the relationships between them.
- It is a theoretical concept whose practical implementation is done in Physical Data Model.
- The advantage of using a Representational data model is to provide a foundation to form the base for the Physical mode



PHYSICAL DATA MODEL

- The physical Data Model is used to practically implement Relational Data Model. Ultimately, all data in a database is stored physically on a secondary storage device such as discs and tapes. This is stored in the form of files, records, and certain other data structures.
- It has all the information on the format in which the files are present and the structure of the databases, the presence of external data structures, and their relation to each other. Here, we basically save tables in memory so they can be accessed efficiently. In order to come up with a good physical model, we have to work on the relational model in a better way.
- Structured Query Language (SQL) is used to practically implement Relational Algebra.
- This Data Model describes **HOW** the system will be implemented using a specific DBMS system. This model is typically created by DBA and developers. The purpose is actual implementation of the database.
- **Characteristics of a physical data model:**
 - The physical data model describes data need for a single project or application though it maybe integrated with other physical data models based on project scope.
 - Data Model contains relationships between tables that which addresses cardinality and nullability of the relationships.
 - Developed for a specific version of a DBMS, location, data storage or technology to be used in the project.
 - Columns should have exact datatypes, lengths assigned and default values.
 - Primary and Foreign keys, views, indexes, access profiles, and authorizations, etc. are defined



ADVANTAGES OF DATA MODEL

- Data systems that have gone through data modeling have many advantages, especially for organizations that collect and use a high volume of business data. Possibly the most important advantage is the data modeling process ensures a higher quality of data because the organization's resulting data governance, or a company's policies and procedures to guarantee quality management of its data assets, follow a well-thought-out plan.
- A proper data modeling process also improves system performance while saving money. Without the data modeling exercise, a business could find the systems they use are more extensive than needed (thus costing more than they need) or won't support their data needs (and performing poorly). Another advantage of data modeling is so much becomes known about the data (type and length, for example) that organizations can create applications and reports that use data more rapidly and with fewer errors.
- A good data modeling plan will also enable more rapid onboarding of acquired companies, especially if they also have a data modeling plan. At the very least, the acquired company's data modeling plan can be used to assess how quickly the two data sets can be connected. What's more, the existence of a plan will expedite conversions to the acquiring company's systems.



DISADVANTAGES OF DATA MODEL

1. In the case of a vast database, sometimes it becomes difficult to understand the data model.
 2. You must have the proper knowledge of SQL to use physical models.
 3. Even smaller change made in structure require modification in the entire application.
 4. There is no set data manipulation language in DBMS.
 5. To develop Data model one should know physical data stored characteristics.
- Data modeling is not for every organization. If an organization does not use or plan to collect a substantial amount of data, the exercise of data modeling might be overkill.
 - For organizations that consider themselves data-driven and have — or plan to have — a lot of data, the main disadvantage of data modeling is the time it takes to create the plan. Depending on the complexity of the organization and the spectrum of data being collected, data modeling might take a long time.
 - Another potential disadvantage depends on the willingness of non-technical staff to fully engage in the process. Integral to data modeling is that the business needs and requirements are as fully described as possible. If business stakeholders are not fully engaged in the data modeling process, it's unlikely that data architects will get the input they need for successful data modeling.



EXAMPLES OF DATA MODELLING

- Consider this example of a small hotel chain to illustrate the role of data modeling and its importance to business decision-making. The hotel chain books rooms through three channels: it's a call center, a website and many independent travel sites. The independent sites are problematic because they take a commission and the hotel chain must spend extra to make it to the top of search results.
- Data modeling reveals the importance of tracking independent travel site data in coordination with other data so that the hotel chain can decide where to spend its money. The data modeler notes that sales source information must be tracked by each independent travel site (i.e., channel) so that the business can decide the value of each. Additionally, the data must connect to two other data sets: commissions paid for orders and advertising spend. This way, a report or dashboard can be created showing the cost of sales that came through each channel. The business can use that report to determine which independent travel sites are most profitable and adjust its spending accordingly.
- Without going through the data modeling process, it's likely that the sales and commissions tables would be naturally connected but the advertising spend table might not. Without data modeling, the advertising spend information may not be accessible at all because it is held in department-level spreadsheets. It's the process of data modeling that brings this information together, setting requirements for data entry when needed, so that the business can get actionable data insights for decision-making.

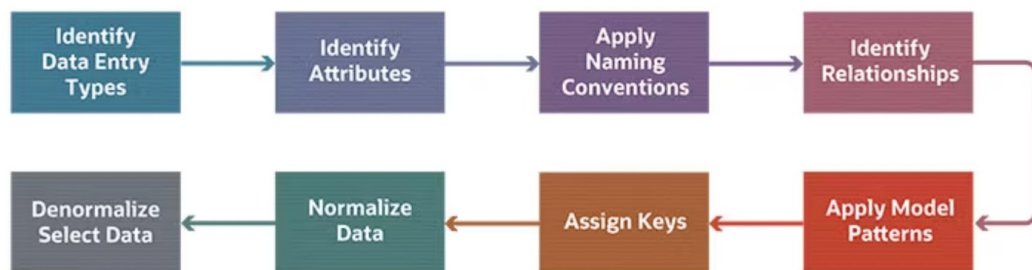


HOW TO MODEL DATA

Once data modelers have obtained a thorough understanding of a business and its data, they're ready to begin the data modeling process itself.

A solid data modeling process typically consists of the following eight steps.

8 Steps to Data Modeling





HOW TO MODEL DATA

- 1. Identify data entity types:** Entity types depend on the specific type of system being constructed. But there are many common entity types, such as sales, customers and employees. A thorough audit and data crunching, or the automated processing of enormous amounts of data, is needed to identify an exhaustive list of entity types. By nature, many entity types (such as customers and sales) might be universally accessible within the organization. But there are also entity types, such as financial data, that need secured access. And there will be some entity types, such as website traffic, that may have little bearing outside of a specific department. All of this must be captured in the entity type definitions.
- 2. Identify attributes:** Each entity type then needs to be defined in terms of its specific attributes. Attributes are the set of fields that go into describing a particular entity. For example, the employee entity type will have attributes such as name, address, phone number, ID badge number and department. It's important to be thorough. It's not that attributes or entities can't be redefined later, but being thorough early in the process avoids later pitfalls.
- 3. Apply naming conventions:** It is also important that the organization set up and use naming conventions along with their definitions. Standard naming conventions allow people to communicate needs more clearly. For example, the hotel chain that pays independent travel sites a "fee" for every sale might want to consider using the word "commission" instead to be more logically compared with the commission paid to the call center sales staff. Also, since the hotel chain pays independent travel sites fees for advertising, it would reduce ambiguity about what each word describes.



HOW TO MODEL DATA

- 1. Identify relationships:** Connected data tables make it possible to use a technique known as data drilling, or the different operations performed on multidimensional and tabular data. The best way to see this is by thinking about an order. The specific phrases used by UML are italicized: Using UML, an order is placed by a customer having potentially multiple addresses and is composed of one or more items that are products or services. By representing relationships in this way, complex ideas in the data model map can be easily communicated and digested at all levels of the organization.
- 2. Apply data model patterns:** Data model patterns are best-practice templates for how to handle different entity types. These patterns follow tested standards that provide solutions for handling many entity types. What's valuable about data model patterns is they can underscore elements that may not have been obvious to data architects in a particular data modeling exercise but are contained in the pattern due to extensive prior experience. For example, the idea of including a "Customer Type" table — which opens up the possibility of doing analysis based on different types of customers — might come from a data model pattern. Data model patterns are available through books and on the web.
- 3. Assign keys:** The concept of "keys" is central to relational databases. Keys are codes that identify each data field in every row (or record) within a table. They're the mechanism by which tables are interconnected — or "joined" — with one another. There are three main types of keys to assign:
 - **Primary keys:** These are unique, per-record identifiers. A data table is allowed only one primary key per record, and it cannot be blank. A customer table, for example, should already have a unique identifier associated with each customer. If that number truly is unique in the database, it would make an excellent primary key for customer records.
 - **Secondary keys:** These also are unique per-record identifiers, but they allow empty (null) entries. They're mainly used to identify other (non-primary) data fields within a record. The email address field in a customer table is a good example of a secondary key since an email address is likely to be unique per customer, yet there might not be an email address for every customer. Secondary keys are indexed for faster lookups.
 - **Foreign keys:** These are used to connect two tables that have a known relationship. The foreign key would be the primary key from a record in the related table. For example, a customer table might have a connection to a customer address table. The primary key from the customer address table would be used as the foreign key in the customer table.



HOW TO MODEL DATA

1. **Normalize data:** Normalization looks for opportunities where data may be more efficiently stored in separate tables. Whenever content is used many times — customer and product names, employees, contracts, departments — it would likely be better and more useful to store it in a separate table. This both reduces redundancy and improves integrity.
 - A good example is customer information in an order table. Each order naturally includes the name and address of the customer, but that information is all redundant — it appears as many times as there are orders in the table. This redundancy causes many problems, not the least of which is that every entry must be spelled exactly like all other entries for searches of that table to work. To reduce this redundancy and improve data integrity, the customer information data is normalized by being placed into a separate “customers” table along with all the relevant customer information. Each customer is assigned a unique customer identifier. The order table is then modified to replace the customer information fields with a single field referencing the customer’s unique identifier. This process of normalization improves the integrity of the data because a single authoritative source — the “customers” table — governs all references to the customer.
 - Another benefit of normalization is it enables faster database searches.
1. **Denormalize (selected data) to improve performance:** Partial denormalization is sometimes needed in specific circumstances. Normalization generally makes for a better, more accurate database with more individual tables that can be quickly and accurately searched. But not always. The cost of more tables is more “joins” connecting tables in the event of complex queries. Individually, joins have a virtually imperceptible performance cost. But they add up for complex queries.
 - Take, for instance, fans interacting with a ticketing system for concerts. A fan selects seats and the system must prevent those seats from being purchased by someone else until the customer makes a decision. If they choose not to buy the seats, or the time limit expires, it’s important that the seats become immediately available for other customers. To support this high-volume transaction system, the data professionals architecting the system might recommend a section of the database be denormalized so that all the core transaction elements — seat numbers, venue, concert date, performing artist, etc. — exist in as few tables as possible, ideally one. So instead of a query that joins the venue information with tables for the artist, date and seat numbers (all likely normalized into separate tables), it will be a faster transaction if the normalization of these tables or these fields are reversed.
 - Not all tables need to be denormalized, and each one that does will have performance as its most critical requirement. It all depends on the need for speed, and it might be that a fully normalized database meets the business’s performance requirements.



CONCLUSION

- Data modeling is the process of developing data model for the data to be stored in a Database.
- Data Models ensure consistency in naming conventions, default values, semantics, security while ensuring quality of the data.
- Data Model structure helps to define the relational tables, primary and foreign keys and stored procedures.
- There are three types of conceptual, logical, and physical.
- The main aim of conceptual model is to establish the entities, their attributes, and their relationships.
- Logical data model defines the structure of the data elements and set the relationships between them.
- A Physical Data Model describes the database specific implementation of the data model.
- The main goal of a designing data model is to make certain that data objects offered by the functional team are represented accurately.
- The biggest drawback is that even smaller change made in structure require modification in the entire application.

2.2

Importance of Data Models

Why is data modelling important?

- A comprehensive and optimized data model helps create a simplified, logical database that eliminates redundancy, reduces storage requirements, and enables efficient retrieval.
- It also equips all systems with a 'single source of truth' – which is essential for effective operations and provable compliance with regulations and regulatory requirements.



SEVERAL KEY ASPECTS (PART 1)

- A data model is a crucial component in the process of designing a database, and it plays a fundamental role in organizing and structuring information.

1. Clarity and Understanding:

- **Communication Tool:** A data model serves as a visual representation of the structure and relationships within a database. It helps communicate complex data concepts in a clear and understandable manner to various stakeholders, including developers, database administrators, and business users.

2. Requirements Analysis:

- **Capture Business Requirements:** Creating a data model involves understanding and capturing the business requirements. It provides a framework for discussing and refining these requirements with stakeholders, ensuring that the database design aligns with the organization's needs.

3. Database Design:

- **Blueprint for Implementation:** A data model acts as a blueprint for the actual implementation of the database. It specifies the entities, attributes, relationships, and constraints that will be used to structure the data. This ensures consistency in the database design process.

4. Data Integrity:

- **Enforce Constraints:** The data model includes constraints, such as primary keys, foreign keys, and unique constraints, which are critical for maintaining data integrity. These constraints help prevent data inconsistencies and inaccuracies.

5. Efficient Data Retrieval:

- **Optimized Querying:** Through proper indexing and organization of data, a well-designed data model facilitates efficient data retrieval. It allows for the execution of queries in a manner that optimizes performance, even when dealing with large dataset



SEVERAL KEY ASPECTS (PART 2)

6. Scalability and Performance:

- **Foundation for Scaling:** The data model provides a foundation for scalability. As the data volume grows, a well-designed data model allows for the implementation of scaling strategies, ensuring that the database can handle increased demands while maintaining performance.

7. Data Maintenance and Management:

- **Simplified Maintenance:** A clear data model makes it easier to understand the structure of the database, simplifying ongoing maintenance tasks. It enables efficient updates, modifications, and enhancements to the database schema without compromising data consistency.

8. Integration with Applications:

- **Support Application Development:** Developers use the data model as a guide for integrating applications with the database. It helps ensure that applications can interact with the database in a consistent and meaningful way.

9. Data Security:

- **Define Security Policies:** The data model can include specifications for access controls, defining who has permission to view, modify, or delete specific data. This is crucial for implementing robust data security measures.

10. Documentation and Knowledge Transfer:

- **Documentation Source:** The data model serves as a valuable source of documentation for the database structure. It aids in knowledge transfer between team members, new hires, and different project phases.

11. Facilitates Change Management:

- **Adaptability:** As business requirements evolve, a well-designed data model provides a foundation for managing changes to the database structure. It supports adaptability and helps minimize disruptions during changes or upgrades.

In essence, the importance of a data model lies in its ability to guide the design and implementation of a database, ensuring that it meets the needs of the organization, adheres to best practices, and supports efficient data management and retrieval.



KEY POINTS TO REMEMBER:

- Data modeling is the process of developing data model for the data to be stored in a Database.
- Data Models ensure consistency in naming conventions, default values, semantics, security while ensuring quality of the data.
- Data Model structure helps to define the relational tables, primary and foreign keys and stored procedures.
- There are three types of conceptual, logical, and physical.
- The main aim of conceptual model is to establish the entities, their attributes, and their relationships.
- Logical data model defines the structure of the data elements and set the relationships between them.
- A Physical Data Model describes the database specific implementation of the data model.
- The main goal of a designing data model is to make certain that data objects offered by the functional team are represented accurately.
- The biggest drawback is that even smaller change made in structure require modification in the entire application.



WHY USED DATA MODEL?


- A database is only as valuable as the faith an organization has in it. Data modeling helps support that faith by ensuring the organization's data, which is often stored in your data warehouse, is accurately represented in a database so it can be meaningfully analyzed.
- On a very tactical level, data modeling helps to define the database — the relationships between data tables, the “keys” used to index and connect tables and the stored procedures that make accessing the data efficient. On a higher level, however, the reason to use data models is to ensure that database designs are aligned with the needs of the business. Without input from business stakeholders, it's very unlikely that subsequent data analysis will reveal useful insights or that data will be organized and stored efficiently.
- Data modeling also provides a standards-based methodology for constructing, managing and growing data assets. By spending the time to create data models and incorporate data model patterns — best-practice templates for common types of data — an organization can make better use of the data it collects and can grow its data and teams more effectively.



INDUTRIES THAT USES DATA MODELING

Industries that tend to benefit from data modeling include any that uses a high volume of data to manage its business. Data modeling increases in importance for industries that use data as a core to their business, especially when privacy concerns and government regulation are added to the mix.

Here are the industries that benefit the most from data modeling:

 Industry	Data Modeling Benefit(s)
 Health Care	Patient data must meet strict HIPAA guidelines; collected data can be used to develop new health care strategies.
 Retail	Customer knowledge and trends; add-on purchase suggestions.
 Financial Services	Reduce fraud; security is top concern.

Importance of Data Models



INDUTRIES THAT USES DATA MODELING

 Insurance	Reduce premiums to better drivers.
 Education	Personalized learning.
 Transportation/ Supply Chain	Stock levels; delivery forecasts; route optimization.
 Agriculture	Tracking equipment performance.
 Energy	Improving fuel exploration efficiency.
 Hospitality	Higher occupancy rates.

Industries That Use Data Modeling

2.3

Data Model Basic Building Blocks

Data models consist of basic building blocks that help organize and represent the structure of data in a systematic and meaningful way.

The primary building blocks in data modeling include entities, attributes, relationships, and constraints



Building Blocks in Data Modelling

1. Entity:

- An entity is a real-world object or concept that is distinguishable from other objects. In data modeling, entities represent the main elements about which data is collected. Examples of entities could include "Customer," "Product," "Employee," etc.

2. Attributes:

- Attributes are the properties or characteristics of entities. They describe the data we want to capture about each entity. For instance, for the "Customer" entity, attributes could include "CustomerID," "Name," "Address," and "Email."

3. Relationships:

- Relationships illustrate how entities are connected or associated with each other. They represent the connections between entities and define how data is shared between them. Relationships can be one-to-one, one-to-many, or many-to-many. For example, a "Student" entity may have a one-to-many relationship with a "Course" entity, indicating that a student can enroll in multiple courses.

4. Key:

- A key is a field or set of fields that uniquely identifies each record in a table. There are two main types of keys:
 1. **Primary Key:** A unique identifier for a record within a table.
 2. **Foreign Key:** A field in one table that refers to the primary key in another table, establishing a relationship between the two tables.



Building Blocks in Data Modelling (cont..)

5. Attributes Types and Data Types:

- Attributes can have different types, specifying the kind of data they can hold. Examples include text, numeric, date, and Boolean types. Data types define the format in which data is stored, ensuring consistency and efficiency.

6. Constraints:

- Constraints are rules or conditions applied to the data to maintain accuracy and integrity. Common constraints include:
 1. **Primary Key Constraint:** Ensures that the primary key values are unique.
 2. **Foreign Key Constraint:** Enforces referential integrity between tables.
 3. **Unique Constraint:** Ensures that values in a column are unique.
 4. **Check Constraint:** Enforces a condition on the values allowed in a column.
 5. **Default Constraint:** Provides a default value for a column if no value is specified.
- These building blocks are used to create visual representations such as Entity-Relationship Diagrams (ERDs) for conceptual models and tables for logical models. The ERD visually represents entities, attributes, relationships, and their connections, helping stakeholders understand the structure of the data.
- In summary, the basic building blocks of a data model—entities, attributes, relationships, keys, attribute types, and constraints—are crucial for designing a database that accurately captures, organizes, and manages information.



3 Primary Types of Data Models

The three primary data model types are relational, dimensional, and entity-relationship (E-R). There are also several others that are not in general use, including hierarchical, network, object-oriented, and multi-value. The model type defines the logical structure – how the data is stored, logically – and therefore how it is stored, organized, and retrieved.

1. Relational:

- Although “older” in approach, the most common database model still in use today is relational, which stores the data in fixed-format records and arranges data in tables with rows and columns.
- The most basic type of data model has two elements: measures and dimensions. Measures are numeric values, such as quantities and revenue, used in mathematical calculations like sum or average.
- Dimensions can be text or numeric. They are not used in calculations and include descriptions or locations.
- The raw data is defined as a measure or a dimension. Other terminology used in relational database design includes “relations” (the table with rows and columns), “attributes” (columns), “tuples” (rows), and “domain” (set of values allowed in a column).
- While there are additional terms and structural requirements that define a relational database, the important factor is the relationships defined within that structure.
- Common data elements (or keys) link tables and data sets together. Tables can also be related explicitly, like parent and child relationships including one-to-one, one-to-many, or many-to-many.



3 Primary Types of Data Models (cont..)

2. Dimensional:

- Less rigid and structured, the dimensional approach favors a contextual data structure that is more related to the business use or context. This database structure is optimised for online queries and data warehousing tools.
- Critical data elements, like a transaction quantity for example, are called “facts” and are accompanied by reference information called “dimensions,” be that product ID, unit price, or transaction date.
- A fact table is a primary table in a dimensional model. Retrieval can be quick and efficient – with data for a specific type of activity stored together – but the lack of relationship links can complicate analytical retrieval and use of the data.
- Since the data structure is tied to the business function that produces and uses the data, combining data produced by dissimilar systems (in a data warehouse, for instance) can be problematic.

2. Entity-Rich (E-R):

- An E-R model represents a business data structure in graphical form containing boxes of various shapes to represent activities, functions, or “entities” and lines to represent associations, dependencies, or “relationships.”
- The E-R model is then used to create a relational database with each row representing an entity and the fields in that row contain attributes. As in all relational databases, “key” data elements are used to link tables together.

2.4

Business Rules

How do you add business rules to your data model?

Adding business rules to a data model is an essential step in ensuring that the database accurately reflects the business requirements and constraints.

Business rules are specifications or guidelines that describe the business logic, policies, and procedures that govern the data and its use. Here's how you can add business rules to your data model:



Business Rules in Data Model

1. Identify Business Rules:

- Work closely with stakeholders, subject matter experts, and end-users to identify and document business rules. These rules may come from regulatory requirements, company policies, industry standards, or specific operational needs.

2. Document Business Rules:

- Clearly document each business rule in a format that is easy to understand. Include details such as the rule's purpose, conditions triggering the rule, and any actions or consequences resulting from the rule.

3. Associate Business Rules with Entities and Attributes:

- Link each business rule to the specific entities and attributes in your data model. Clearly indicate which part of the model each rule pertains to. This association ensures that the rules are applied in the right context.

4. Use Constraints:

- Leverage database constraints to enforce business rules at the database level. Common types of constraints include:
 1. **Unique Constraints:** Ensure that values in a column or combination of columns are unique.
 2. **Check Constraints:** Enforce conditions on the values allowed in a column.
 3. **Default Constraints:** Provide default values for columns.
 4. **Foreign Key Constraints:** Enforce referential integrity between tables.

5. Document Validation Rules:

- Specify any validation rules that need to be applied to the data. For instance, if a certain attribute must be within a specific range or follow a particular format, document these rules explicitly.



Business Rules in Data Model

1. Incorporate Rule Enforcement in Code or Application Logic:

- If certain rules are best enforced at the application level, make sure to incorporate them into the code or application logic. This is particularly relevant for rules that involve complex calculations or interactions between multiple entities.

2. Review and Validate:

- Regularly review and validate the business rules with stakeholders to ensure that they remain accurate and aligned with evolving business needs. As business requirements change, the associated rules may need to be updated.

3. Document Exception Handling:

- Specify how exceptions to the rules should be handled. Clearly document the procedures for addressing data that doesn't conform to the established business rules.

4. Communicate with Stakeholders:

- Maintain open communication with stakeholders to ensure that everyone involved understands the business rules and their implications. This helps prevent misunderstandings and ensures that the data model continues to meet business needs.

5. Version Control:

- Consider implementing version control for your data model documentation. This allows you to track changes to business rules over time and helps maintain a historical record of how the data model has evolved.

2.5

Evolution of Data Model

In a very real sense, data modelling has been around for as long as data processing, data storage, and computer programming, although the term itself probably only came into common use around the time that database management systems began to evolve in the 1960s. There's nothing new or innovative about the concept of planning and architecting a new structure. Data modelling itself has become more structured and formalized as more data, more databases, and more varieties of data have emerged.

Today, data modelling is more essential than ever as technologists struggle with new sources of data (IoT sensors, location-aware devices, clickstreams, social media) along with an onrush of unstructured data (text, audio, video, raw sensor output) – at volumes and velocity that exceed the capabilities of traditional systems. There is now a constant demand for new systems, innovative database structures and techniques, and new data models to tie this new development effort together.



MAJOR STAGES IN EVOLUTION OF DATA MODELS

- The evolution of data models has progressed over several decades, reflecting advancements in technology, changes in data management needs, and improvements in our understanding of how to represent and organize data. The major stages in the evolution of data models include:

1. Hierarchical Model:

- **Timeframe:** Late 1960s
- **Key Characteristics:**
 - Represents data in a tree-like structure with parent-child relationships.
 - Suitable for representing one-to-many relationships.
 - Commonly used in early database systems such as IMS (Information Management System).

2. Network Model:

- **Timeframe:** Late 1960s to 1970s
- **Key Characteristics:**
 - Extends the hierarchical model by allowing many-to-many relationships.
 - Uses a more flexible graph structure.
 - Introduces the concept of pointers to navigate between records.
 - Prominent in database systems like CODASYL DBTG (Conference on Data Systems Languages Database Task Group).

3. Relational Model:

- **Timeframe:** Early 1970s
- **Key Characteristics:**
 - Introduced by Edgar F. Codd.
 - Represents data in tables with rows and columns.
 - Utilizes a set-based approach for data manipulation.
 - Provides a declarative query language (SQL) for accessing and manipulating data.
 - Enables data independence between physical storage and logical representation.
 - Dominates the database landscape and remains highly influential.



MAJOR STAGES IN EVOLUTION OF DATA MODELS

4. Entity-Relationship Model (ER Model):

- **Timeframe:** Mid-1970s
- **Key Characteristics:**
 - Introduced by Peter Chen.
 - Focuses on high-level conceptual modeling using entities, attributes, and relationships.
 - Widely used for designing and visualizing database schemas.
 - Serves as a foundation for creating conceptual data models.

5. Object-Oriented Model:

- **Timeframe:** Late 1980s to 1990s
- **Key Characteristics:**
 - Extends the relational model to include concepts from object-oriented programming.
 - Represents data as objects with attributes and methods.
 - Prominent in object-oriented databases (OODBMS) and certain extensions to relational databases.

6. Object-Relational Model:

- **Timeframe:** 1990s to present
- **Key Characteristics:**
 - Combines features of both the relational and object-oriented models.
 - Supports complex data types, inheritance, and encapsulation.
 - Allows the use of object-oriented programming concepts within a relational database environment.



MAJOR STAGES IN EVOLUTION OF DATA MODELS

7. NoSQL Models:

- **Timeframe:** 2000s to present
- **Key Characteristics:**
 - Responds to the needs of handling large volumes of unstructured or semi-structured data.
 - NoSQL databases include various models, such as document-oriented, key-value, column-family, and graph databases.
 - Emphasizes scalability, flexibility, and horizontal partitioning.

8. Graph Model:

- **Timeframe:** 2000s to present
- **Key Characteristics:**
 - Specifically designed for modeling and querying relationships in data.
 - Represents data as nodes and edges in a graph structure.
 - Effective for scenarios where relationships between entities are a primary focus.

The evolution of data models reflects the continuous effort to address diverse data management needs and challenges. While the relational model remains a cornerstone, newer models, such as NoSQL and graph databases, have emerged to meet the demands of modern applications, big data, and varied data structures.

The choice of a data model often depends on the specific requirements and characteristics of the data being managed.



WHAT IS NEXT FOR DATA MODELLING?

- Information connectivity and large quantities of data from so many different sources –including sensors, voice, video, email, and more – extend the scope of modelling projects for IT professionals. The Internet is, of course, one of the enablers of this evolution.
- The cloud is a major part of the solution as it is the only computing infrastructure big enough, scalable enough, and agile enough to address current and future requirements in the expanding world of connectivity.
- Options for database design are also changing. A decade ago, the dominant database structure was a row-oriented relational database using traditional disk storage technology.
- The data for a typical ERP's general ledger or inventory management was stored in dozens of different tables that need to be updated and modeled.
- Today, modern ERP solutions store active data in memory using a columnar design for a dramatic reduction in tables and increase in speed and efficiency.
- For line of business professionals, the new self-service tools available today will continue to improve. And new tools will be introduced to make data modelling and visualization even easier and more collaborative.

2.6

Degrees of Data Abstraction

What are the three levels of data abstraction?

There are many types of data models, with different types of possible layouts. The data processing community identifies three kinds of modelling to represent levels of thought as the models are developed.



CONCEPTUAL DATA MODEL

- This is the “big picture” model that represents the overall structure and content but not the detail of the data plan.
- It is the typical starting point for data modelling, identifying the various data sets and data flow through the organization.
- The conceptual model is the high-level blueprint for development of the logical and physical models and is an important part of the data architecture documentation.

EXAMPLE

Let's consider a simple example of a conceptual data model for a university system. This system will involve entities related to students, courses, and professors, along with their attributes and relationships.

- **Entities:**

- 1. Student:**

- 1. Attributes: StudentID, Name, DateOfBirth, Address
 - 2. Relationships: Enrolls in (with Course entity)

- 2. Course:**

- 1. Attributes: CourseID, Title, Department, Credits
 - 2. Relationships: Offered by (with Professor entity), Enrolled in (with Student entity)

- 3. Professor:**

- 1. Attributes: ProfessorID, Name, Department
 - 2. Relationships: Teaches (with Course entity)

- **Relationships:**

- Enrolls in:**

- Connects Student entity to Course entity.
 - Represents the fact that a student can enroll in multiple courses, and a course can have multiple enrolled students.



CONCEPTUAL DATA MODEL

Offered by:

- Connects Course entity to Professor entity.
- Represents the fact that a course is offered by a professor.

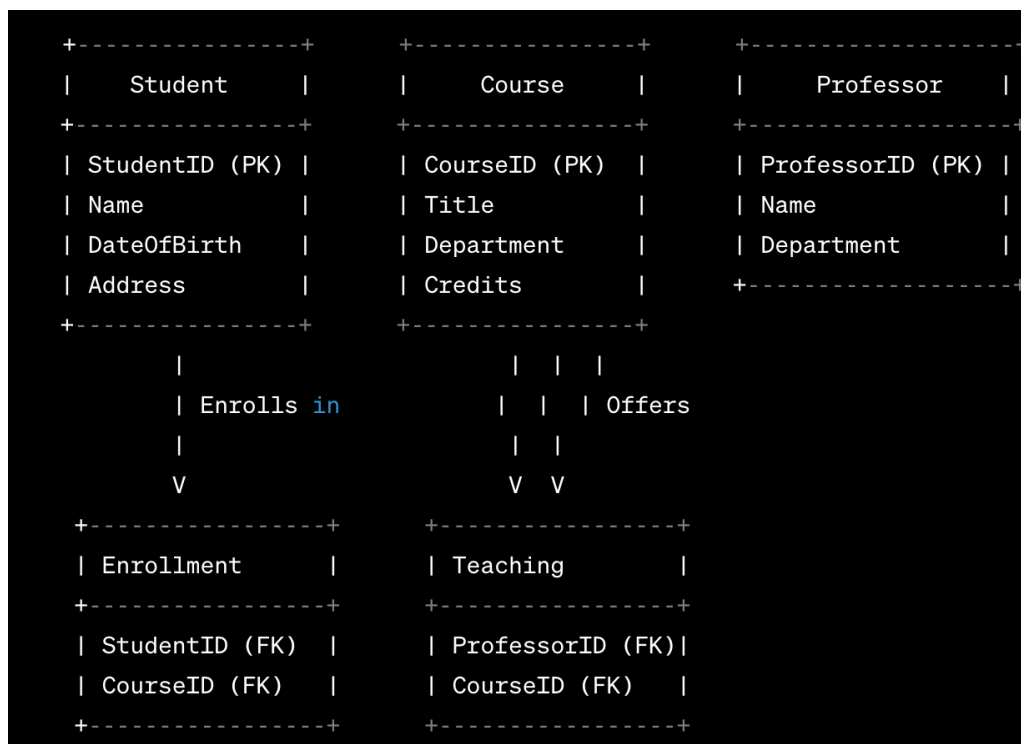
Teaches:

- Connects Professor entity to Course entity.
- Represents the fact that a professor teaches a course.

Example Scenario:

- A student (e.g., StudentID: 123) named Alice is enrolled in two courses (e.g., CourseID: CS101 and CourseID: MATH201).
- The CS101 course is offered by Professor Smith, who teaches multiple courses.
- Professor Smith is part of the Computer Science department.

Visual Representation (Entity-Relationship Diagram - ERD):





DISCUSSION POINTS:

1. Entities and Attributes:

- Discuss the entities (Student, Course, Professor) and their attributes (e.g., StudentID, Name, CourseID, Title).
- Emphasize the importance of choosing relevant attributes based on the requirements.

2. Relationships:

- Explain the relationships between entities (Enrolls in, Offered by, Teaches).
- Discuss the cardinality and participation constraints. For example, a student may be enrolled in zero or more courses, but a course must have at least one enrolled student.

3. Scenario:

- Walk through the example scenario involving Alice, the courses she's enrolled in, and the professor who teaches one of those courses.

4. ERD:

- Explain the visual representation (ERD) and how it captures the entities, attributes, and relationships.
- Highlight the use of primary keys (PK) and foreign keys (FK) to establish relationships between tables.

This example provides a foundation for discussing the conceptual data model, illustrating how entities, attributes, and relationships can be organized to represent the structure of a university database.



LOGICAL DATA MODEL

- The second level of detail is the logical data model. It most closely relates to the general definition of “data model” in that it describes the data flow and database content.
- The logical model adds detail to the overall structure in the conceptual model but does not include specifications for the database itself as the model can be applied to various database technologies and products.

EXAMPLE:

- Let's consider a simplified example of a logical data model for a library management system. This model will represent key entities, attributes, relationships, and constraints for the purpose of organizing and managing library-related data.
- **Entities:**
 - **Book:**
 - Attributes: ISBN (Primary Key), Title, Author, Genre, Publication Year
 - **Author:**
 - Attributes: AuthorID (Primary Key), AuthorName
 - **Publisher:**
 - Attributes: PublisherID (Primary Key), PublisherName
 - **LibraryMember:**
 - Attributes: MemberID (Primary Key), MemberName, Address, Email
 - **LibraryLoan:**
 - Attributes: LoanID (Primary Key), BookISBN (Foreign Key referencing Book), MemberID (Foreign Key referencing LibraryMember), LoanDate, DueDate



LOGICAL DATA MODEL

- **Relationships:**
- **Author-Writes-Book:**
 - Connects Author entity to Book entity.
 - Represents the fact that an author can write multiple books, and a book is written by one or more authors.
- **Book-Published-By-Publisher:**
 - Connects Book entity to Publisher entity.
 - Represents the fact that a book is published by a publisher.
- **LibraryMember-Checks-Out-Book:**
 - Connects LibraryMember entity to LibraryLoan and Book entities.
 - Represents the fact that a library member can check out multiple books, and each book can be checked out by multiple members.

Logical Data Model Diagram:

+-----+	+-----+	+-----+
Author	Book	Publisher
+-----+	+-----+	+-----+
AuthorID (PK)	ISBN (PK)	PublisherID (PK)
AuthorName	Title	PublisherName
+-----+	Author (FK)	+-----+
	Genre	
	PublicationYear	
	+-----+	
+-----+	+-----+	
LibraryMember	LibraryLoan	
+-----+	+-----+	
MemberID (PK)	LoanID (PK)	
MemberName	BookISBN (FK)	
Address	MemberID (FK)	
Email	LoanDate	
+-----+	DueDate	
	+-----+	



DISCUSSION POINTS:

1. Entities and Attributes:

- Discuss the entities (Book, Author, Publisher, LibraryMember, LibraryLoan) and their attributes.
- Emphasize the use of primary keys (PK) to uniquely identify records within each entity.

2. Relationships:

- Explain the relationships between entities (Author-Writes-Book, Book-Published-By-Publisher, LibraryMember-Checks-Out-Book).
- Discuss the cardinality and participation constraints. For example, an author can write multiple books, but a book must have at least one author.

3. Foreign Keys:

- Highlight the use of foreign keys (FK) to establish relationships between tables.
- Discuss how foreign keys create links between related entities.

4. Logical Constraints:

- Mention any logical constraints imposed on the model, such as uniqueness constraints on primary keys or referential integrity constraints.

5. Normalization:

- Discuss how the logical data model adheres to principles of normalization, avoiding redundancy and ensuring data consistency.

This example provides a basis for discussing a logical data model for a library management system. It illustrates how entities, attributes, and relationships are organized to represent the structure of the data at a logical level.



PHYSICAL DATA MODEL

- The physical database model describes the specifics of how the logical model will be realized. It must contain enough detail to enable technologists to create the actual database structure in hardware and software to support the applications that will use it.
- Needless to say, the physical data model is specific to a designated database software system. There can be multiple physical models derived from a single logical model if different database systems will be used.

EXAMPLE:

- Certainly! Let's continue with the example of a library management system and discuss a simplified physical data model. In this model, we'll focus on how the logical data model can be translated into a physical implementation, considering aspects such as data types, indexing, and storage.
- **Entities:**
 - **Book:**
 - Attributes: ISBN (Primary Key), Title, Author, Genre, Publication Year
 - **Author:**
 - Attributes: AuthorID (Primary Key), AuthorName
 - **Publisher:**
 - Attributes: PublisherID (Primary Key), PublisherName
 - **LibraryMember:**
 - Attributes: MemberID (Primary Key), MemberName, Address, Email
 - **LibraryLoan:**
 - Attributes: LoanID (Primary Key), BookISBN (Foreign Key referencing Book), MemberID (Foreign Key referencing LibraryMember), LoanDate, DueDate



PHYSICAL DATA MODEL

- **Data Types:**
 - Choose appropriate data types for each attribute. For example:
 - ISBN may be represented as VARCHAR or CHAR.
 - Title, AuthorName, PublisherName, MemberName, Address, and Email may be represented as VARCHAR.
 - Publication Year may be represented as an integer.
- **Primary and Foreign Keys:**
 - Ensure that primary keys are defined for each table to uniquely identify records.
 - Establish foreign keys to maintain relationships between tables.
- **Indexes:**
 - Consider creating indexes to enhance query performance. For example:
 - An index on ISBN in the Book table can speed up searches for books by ISBN.
 - Indexes on foreign keys (e.g., BookISBN in LibraryLoan) can improve join performance.
- **Normalization:**
 - Verify that the physical data model conforms to the normalization principles. Ensure that each table is in at least the third normal form to minimize redundancy.
- **Constraints:**
 - Enforce constraints such as primary key constraints, foreign key constraints, and check constraints to maintain data integrity.
 - For example, enforce that BookISBN in LibraryLoan references a valid ISBN in the Book table.
- **Storage Considerations:**
 - Consider storage considerations, such as file organization and block size, based on the database management system's requirements.



PHYSICAL DATA MODEL

- **Data Types:**
 - Choose appropriate data types for each attribute. For example:
 - ISBN may be represented as VARCHAR or CHAR.
 - Title, AuthorName, PublisherName, MemberName, Address, and Email may be represented as VARCHAR.
 - Publication Year may be represented as an integer.
- **Primary and Foreign Keys:**
 - Ensure that primary keys are defined for each table to uniquely identify records.
 - Establish foreign keys to maintain relationships between tables.
- **Indexes:**
 - Consider creating indexes to enhance query performance. For example:
 - An index on ISBN in the Book table can speed up searches for books by ISBN.
 - Indexes on foreign keys (e.g., BookISBN in LibraryLoan) can improve join performance.
- **Normalization:**
 - Verify that the physical data model conforms to the normalization principles. Ensure that each table is in at least the third normal form to minimize redundancy.
- **Constraints:**
 - Enforce constraints such as primary key constraints, foreign key constraints, and check constraints to maintain data integrity.
 - For example, enforce that BookISBN in LibraryLoan references a valid ISBN in the Book table.
- **Storage Considerations:**
 - Consider storage considerations, such as file organization and block size, based on the database management system's requirements.



PHYSICAL DATA MODEL

Physical Data Model Diagram:

Book	Author	Publisher
ISBN (PK, CHAR)	AuthorID (PK, INT)	PublisherID (PK, INT)
Title (VARCHAR)	AuthorName (VARCHAR)	PublisherName (VARCHAR)
Author (FK)		
Genre (VARCHAR)		
PubYear (INT)		
LibraryMember	LibraryLoan	
MemberID (PK, INT)	LoanID (PK, INT)	
MemberName (VARCHAR)	BookISBN (FK, CHAR)	
Address (VARCHAR)	MemberID (FK, INT)	
Email (VARCHAR)	LoanDate (DATE)	
	DueDate (DATE)	

- **Data Types:**
 - Discuss the selection of appropriate data types for each attribute based on the nature of the data.
- **Indexes:**
 - Explain the use of indexes to optimize query performance and facilitate faster data retrieval.
- **Constraints:**
 - Emphasize the importance of constraints in maintaining data integrity, such as primary key and foreign key constraints.
- **Normalization:**
 - Discuss how the physical data model adheres to normalization principles, minimizing data redundancy.
- **Storage Considerations:**
 - Mention considerations related to file organization and storage optimization.
- This example demonstrates how the logical data model can be translated into a physical data model, incorporating considerations specific to the database management system and storage environment.