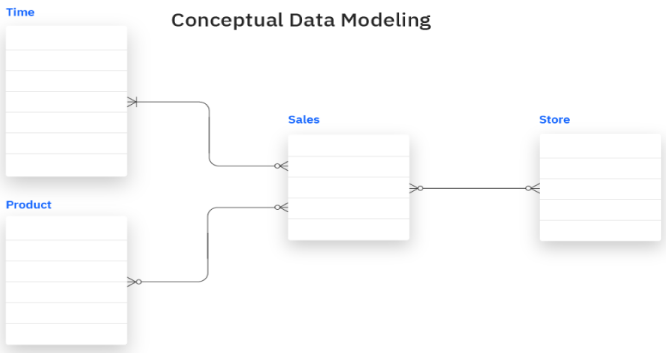
Data modeling is the process of creating a visual representation.

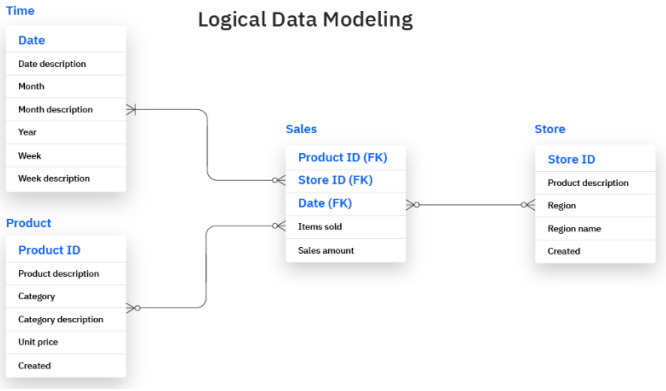
Objective of data model to illustrate the types of data used and stored within the system, the relationships among these data types, the ways the data can be grouped and organized and its formats and attributes.

- Data models are living documents that evolve along with changing business needs.
- Rules and requirements are defined upfront through feedback from business stakeholders.
- Data modeling employs standardized schemas and formal techniques.
- Predictable way of defining and managing data resources across an organization.
- Data models can be shared with vendors, partners, and/or industry peers.

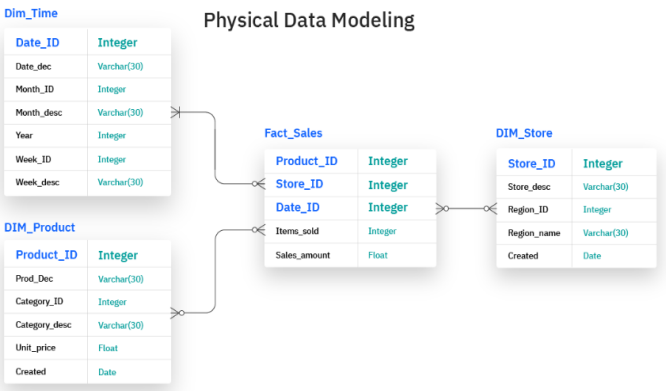
Types of data models:



Conceptual data models include entity classes (defining the types of things that are important for the business to represent in the data model), their characteristics and constraints, the relationships between them and relevant security and data integrity requirements. Any notation is typically simple.



Logical data models provide greater detail about the concepts and relationships in the domain under consideration and indicate data attributes, such as data types and their corresponding lengths, and show the relationships among entities.



Physical data models provide a schema for how the data will be physically stored within a database. Offer a finalized design that can be implemented as a relational database, including associative tables that illustrate the relationships among entities as well as the primary keys and foreign keys that will be used to maintain those relationships. Physical data models can include database management system (DBMS)-specific properties, including performance tuning.

Source: IBM & Databricks

Data Modeling Process

Identify entities: Entities usually become tables. Entities such as sales, store, product, customer.

Identify key properties of each entity. Each entity type can be differentiated from all others because it has one or more unique properties, called attributes. For instance, an entity called “customer” might possess such attributes as a first name, last name, telephone number and salutation.

Map attributes to entities completely. This will ensure the model reflects how the business will use the data.

Identify relationships among entities (cardinality). The earliest draft of a data model will specify the nature of the relationships each entity has with the others. For instance, a record (on the row level) in one table can be associated with multiple records (on the row level) In another table (One - to- Many).

Normalized modeling: Assign keys as needed and decide on a degree of normalization that balances the need to reduce redundancy with performance requirements. Everything is broken out individually. Keys are assigned to groups of data to represent relationships between them without repeating the data. In most cases normalization happens until the 3NF. Could result expensive joins/query.

- **1NF:** Each cell must contain one value; each row must be unique, each column name must be unique, there must be no repeating group and there must be a primary key for each entity/table.
- **2NF:** All non-key attributes or column /data must be dependent on the primary key. There must be a primary key and foreign key for each entity/table.
- **3NF:** The primary key must define all non-key columns/attributes and non-key columns must no depend on any other key. There must be a primary key and foreign key for each entity/table.

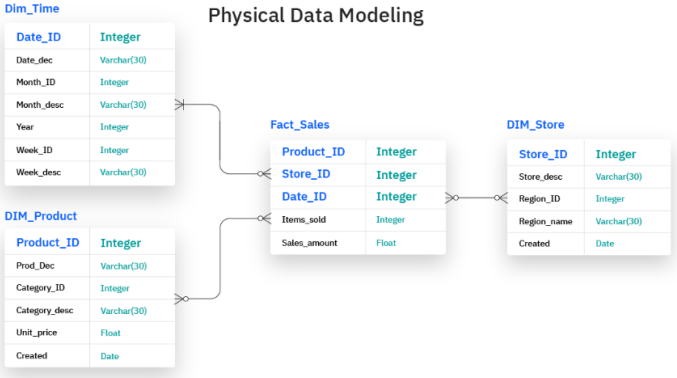
Denormalized modeling: Denormalization modeling refers to the process of intentionally introducing redundancy into a database schema by combining normalized tables into larger, consolidated tables.

- **Insertion anomalies:** Data redundancy and missing null values could lead to inconsistency when adding new data.
- **Deletion anomalies:** To delete one record, need to delete the entire row. This could result in losing essential data from different columns.
- **Update anomalies:** Need to update thousands of rows to update single “price” column. In a normalized database need to update the single “price” column once.

Start schema: A star schema has a single fact table in the center, containing business "facts" (like transaction amounts and quantities). The fact table connects to multiple other dimension tables along "dimensions" like time, or product.

Star schemas de-normalize the data, which means adding redundant columns to some dimension tables to make querying and working with the data faster and easier.

The fact table is normalized but the dimensions tables are not. That is, data from the fact table exists only on the fact table, but dimensional tables may hold redundant data.



Source: IBM & Databricks

Snowflake schema: A snowflake schema is a multi-dimensional data model that is an extension of a star schema, where dimension tables are broken down into subdimensions. Snowflake schemas are commonly used for business intelligence and reporting in OLAP data warehouses, data marts, and relational databases.

Like star schemas, snowflake schemas have a central fact table which is connected to multiple dimension tables via foreign keys. However, the main difference is that they are more normalized than star schemas.

Snowflake schemas offer more storage efficiency, due to their tighter adherence to high normalization standards, but query performance is not as good as with more denormalized data models. Denormalized data models like star schemas have more data redundancy (duplication of data), which makes query performance faster at the cost of duplicated data.

Finalize and validate the data model. Data modeling is an iterative process that should be repeated and refined as business needs change.

Source: IBM & Databricks