Chapter 1

Database Design & Data Modeling

Keys

https://www.guru99.com/dbms-keys.html

Data Modeling

Model is a representation of real data that provides us with characteristics, relations and rules that apply to our data. It doesn't actually contain any data in it. The data model helps us design our database. When building a plane, you don't start with building the engine. You start by creating a blueprint and schematic. Creating a database is just the same, you start with modeling the data.

Terminologies used in Data Modeling

ERD (Entity Relationship Diagram)

https://www.lucidchart.com/pages/er-diagrams
https://creately.com/blog/diagrams/er-diagrams-tutorial/
https://www.guru99.com/er-diagram-tutorial-dbms.html
https://www.youtube.com/playlist?list=PL3R9-um41JsxPg4WAPeEZgH6oAk2oti0Q
The highest level of abstraction for the data model is called the Entity Relationship Diagram (ERD). It is a graphical representation of data requirements for a database

Components of ER Diagram

ER diagram has three main components

1. Entity
2. Attributes
3. Relationship

Entity:

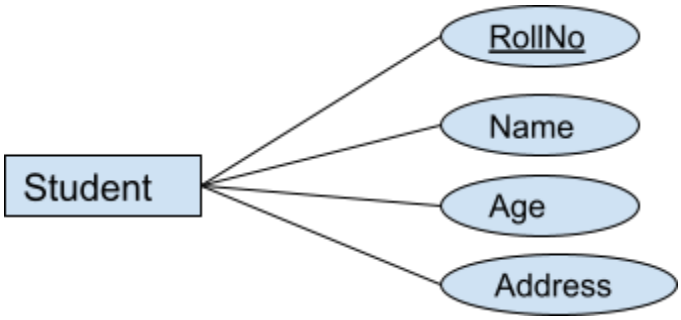
An entity is an object or component of data. An entity is represented as a rectangle in an ER diagram. For example, students, schools, colleges, courses and users are the entities.

Attribute

An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram.

- Key attributes
- Composite attribute
- Multiple attribute
- Derived attribute

Key attributes



A key attribute uniquely identifies an entity from an entity set. In the above figure RollNo attribute uniquely identifies a student from a group of students. Attributes are represented as oval shapes, text of key attribute is underlined.

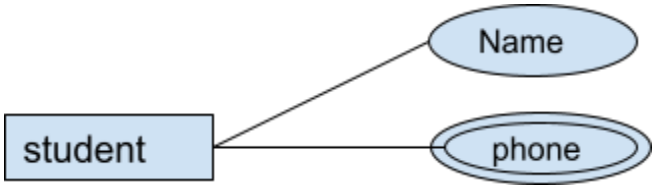
Composite attribute

An attribute that is a combination of other attributes is known as a composite attribute.

In the student entity address is a composite attribute as an address is composed of other attributes such as pin code, state and country.

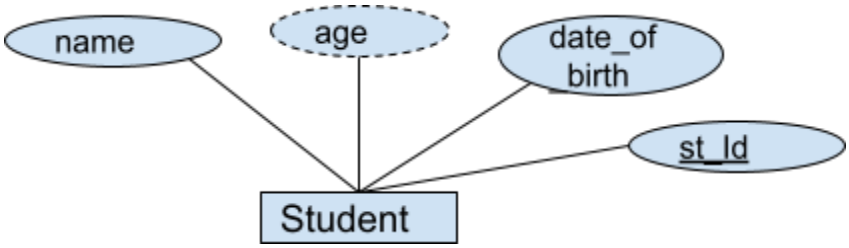
Multivalued attribute

An attribute that can hold multiple values is known as a multivalued attribute.
It is represented with **double ovals** in an ER diagram. For example, a person can have more than one phone number so the phone number is a multivalued attribute.



Derived attribute

A derived attribute is one whose value is dynamic and derived from another attribute. It is represented by a dashed **oval** in an ER diagram. For example, person age is a derived attribute as it changes over time and can be derived from another attribute (Date of Birth).



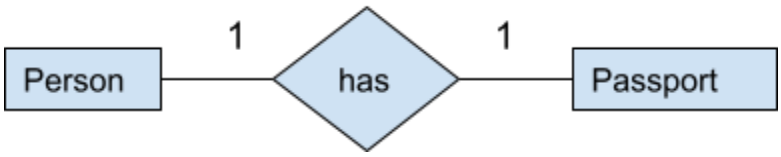
Relationship

A relationship is represented by a **diamond shape** in ER diagram, it shows the relationship among entities. There are four types of relationships:

- One to One
- One to Many
- Many to One
- Many to Many

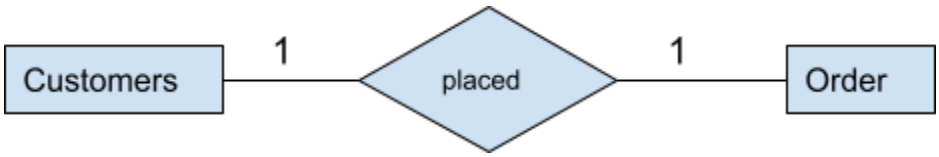
One to One relationship

When a single instance of an entity is associated with a single instance of another entity then it is called one to one relationship. For example, a person has only one passport and a passport is given to one person.



One to Many Relationship

When a single instance of an entity is associated with more than one instance of another entity then it is called one to many relationship. For example a customer can place many orders but an order cannot be placed by many customers.

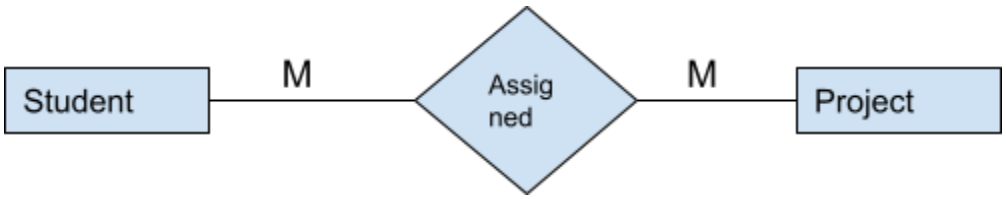


Many to One relationship

When more than one instance of an entity is associated with a single instance of another entity then it is called many to one relationship. For example, many students can study in a single college but a student cannot study in many colleges at the same time.

Many to Many Relationship

When more than one instance of an entity is associated with more than one instance of another entity then it is called many to many relationship. For example, a student can be assigned to many projects and a project can be assigned to many students.



Real Life Example ER Diagram Step by Step

URL : https://tinyurl.com/mprhzdhj

Types of Data Modeling

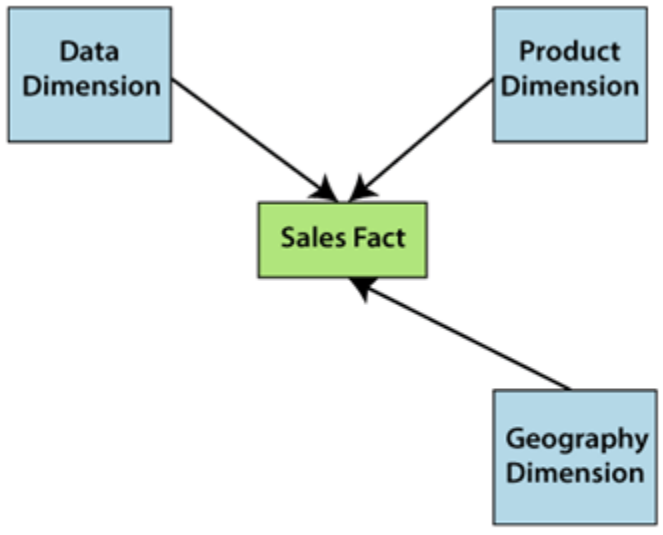
Conceptual Data Model

A conceptual data model recognizes the highest-level relationships between the different entities.

Characteristics of the conceptual data model

- It contains the essential entities and the relationships among them.
- No attribute is specified.
- No primary key is specified.

We can see that the only data shown via the conceptual data model is the entities that define the data and the relationships between those entities. No other data, as shown through the conceptual data model.



Example of Conceptual Data Model

Logical Data Model

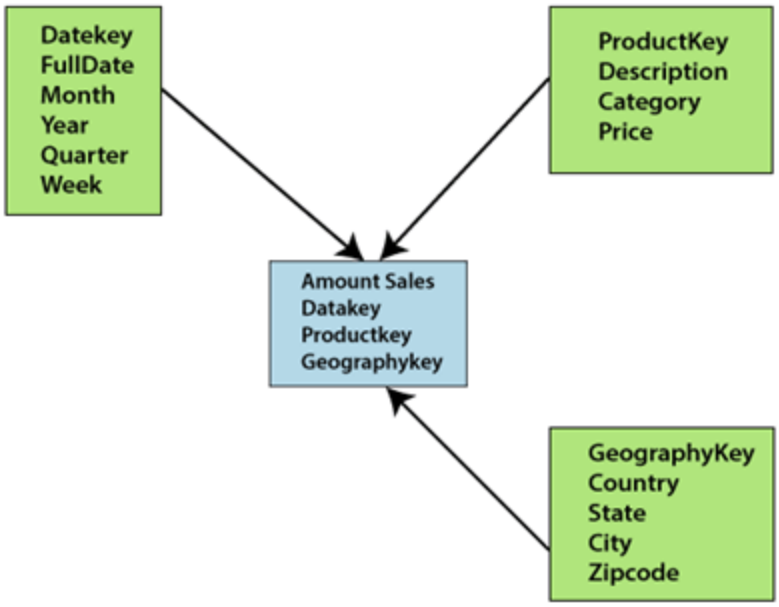
A logical data model defines the information in as much structure as possible, without observing how they will be physically achieved in the database. The primary objective of logical data modeling is to document the business data structures, processes, rules, and relationships by a single view - the logical data model.

Features of a logical data model

- It involves all entities and relationships among them.
- All attributes for each entity are specified.
- The primary key for each entity is stated.
- Referential Integrity is specified (FK Relation)

The phase for designing the logical data model which are as follows:

- Specify primary keys for all entities.
- List the relationships between different entities.
- List all attributes for each entity.
- Normalization.
- No data types are listed



Example of Logical Data Model

Physical Data Model

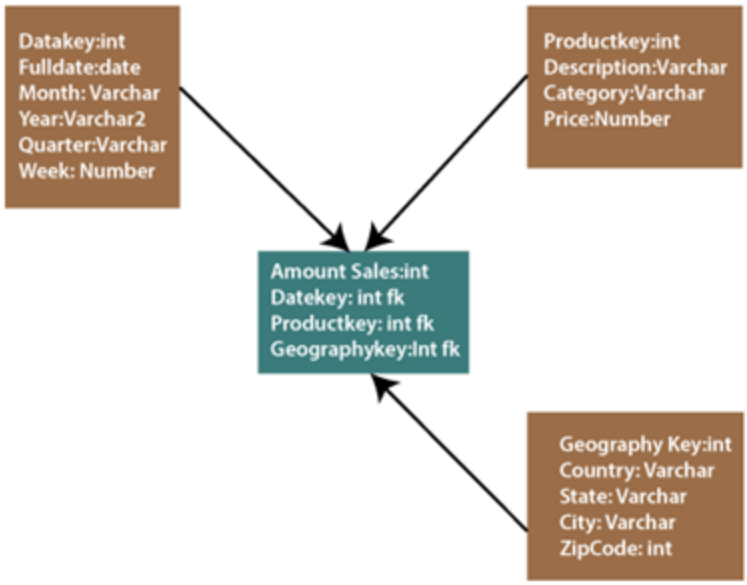
Physical data model describes how the model will be presented in the database. A physical database model demonstrates all table structures, column names, data types, constraints, primary key, foreign key, and relationships between tables. The purpose of physical data modeling is the mapping of the logical data model to the physical structures of the RDBMS system hosting the data warehouse. This contains defining physical RDBMS structures, such as tables and data types to use when storing the information. It may also include the definition of new data structures for enhancing query performance.

Characteristics of a physical data model

- Specification of all tables and columns.
- Foreign keys are used to recognize relationships between tables.

The steps for physical data model design which are as follows:

- Convert entities to tables.
- Convert relationships to foreign keys.
- Convert attributes to columns.



Example of Physical Data Model

Normalization

Identify repeating groups of data and create new tables for them
A more formal definition:
The goals of normalization are to:

- Be able to characterize the level of redundancy in relational schema
- Provide mechanisms for transforming schemas in order to remove redundancy

Normal Forms (NF)

Ordered from least to most normalized:

- First normal form (1NF)
- Second normal form (2NF)
- Third normal form (3NF)

1NF rules:

- Each record must be unique - no duplicate rows
- Each cell must hold one value

Initial data

Student_id	Student_Email	Courses_Completed

235	jim@gmail.com	Introduction to Python, Intermediate Python
455	kelly@yahoo.com	Cleaning Data in R
767	amy@hotmail.com	Machine Learning Toolbox, Deep Learning in Python

Here, a simple table with student_id, student_email and Courses_Completed.
All the rows are unique, but the courses_completed column has more than one course in two records.
To rectify this, we can split the original table as such.

Student_id	Completed
235	Introduction to Python
235	Intermediate Python
455	Cleaning Data in R
767	Machine Learning Toolbox
767	Deep Learning in Python

Now, all the records are unique and each column has one value.

2NF

- Must satisfy 1NF AND
    - If primary key is one column
        - Then automatically satisfies 2NF
- If there is a composite primary key
    - Then each non-key column must be dependent on all the keys

Initial data

Student_id (PK)	Course_id (PK)	Intructor_id	Instructor	Progress
235	2001	560	Nick Carchedi	.55
455	2345	658	Ginger Grant	.10
767	6584	999	Chester Ismay	1.00

In this table, we have the student and course id as composite primary key. We then review the other columns and their dependence on these two keys.

First is the instructor, which isn't dependent on the student_id - only the course_id, meaning an instructor solely depends on the course, not the students who take the course. The same goes for the instructor_id column.
However, the percent completed is dependent on both the student and the course id.

To convert it, we can create two new tables that satisfy the conditions of 2NF.
In 2NF form:

Student_id (PK)	Course_id (PK)	Percent_Completed
235	2001	.55
455	2345	.10
767	6584	1.00

Course_id (PK)	Instructor_id	Instructor
2001	560	Nick Carchedi
2345	658	Ginger Grant
6584	999	Chester Ismay

3NF

- Satisfies 2NF
- No **transitive dependencies**: non-key columns can't depend on other non-key columns.

3NF doesn't allow transitive dependencies. This means that non-primary key columns can't depend on other non-primary key columns. Let's look at an example.

Initial data:

Course_id (PK)	Instructor_id	Instructor	Tech
2001	560	Nick Carchedi	Python
2345	658	Ginger Grant	SQL
6584	999	Chester Ismay	R

Course_id is the primary key so we can ignore this column.
Tech does not depend on the instructor as an instructor can teach different technologies.

We can replace the table from before into these two tables to meet 3NF criteria.

Course_id (PK)	Instructor	Tech
2001	Nick Carchedi	Python
2345	Ginger Grant	SQL
6584	Chester Ismay	R

Instructor_id	Instructor
560	Nick Carchedi
658	Ginger Grant
999	Chester Ismay

These tables have no transitive dependencies and they also meet 2NF as there are no composite primary keys.

Data anomalies

What is risked if we don't normalize enough?

- Update anomaly
- Insertion anomaly
- Deletion anomaly

A database that isn't normalized enough is prone to three types of anomaly errors: update, insertion, and deletion.

Update anomaly

An update anomaly is a data inconsistency that can arise when updating a database with redundancy.

Student_ID	Student_Email	Enrolled_in	Taught_by
230	lisa@gmail.com	Cleaning Data in R	Maggie Matsui
367	bob@gmail.com	Data Visualization in R	Ronald Pearson
520	ken@yahoo.com	Introduction to	Hugo

		Python	Bowne-Anderson
520	ken@yahoo.com	Arima Models in R	David Stoffer

To update student 520's email:

- Need to update more than one record, otherwise, there will be inconsistency
- User updating needs to know about redundancy.

It may sound easy to update multiple records, but it's risky, because it depends on the user updating - if they remember this redundancy.

Insertion anomaly

Unable to add a record due to missing attributes: An insertion anomaly is when you are unable to add a new record due to missing attributes.

Student_ID	Student_Email	Enrolled_in	Taught_by
230	lisa@gmail.com	Cleaning Data in R	Maggie Matsui
367	bob@gmail.com	Data Visualization in R	Ronald Pearson
520	ken@yahoo.com	Introduction to Python	Hugo Bowne-Anderson
520	ken@yahoo.com	Arima Models in R	David Stoffer

Unable to insert a student who has signed up but not enrolled in any courses, they cannot be put into this database.
The only exception is if the enrolled column can accept nulls.
The dependency between columns in the same table unintentionally restricts what can be inserted into the table.

Deletion anomaly

Deletion of record(s) causes unintentional loss of data: A deletion anomaly happens when you delete a record and unintentionally delete other data.

Student_ID	Student_Email	Enrolled_in	Taught_by
230	lisa@gmail.com	Cleaning Data in R	Maggie Matsui
367	bob@hotmail.com	Data Visualization in R	Ronald Pearson
520	ken@yahoo.com	Introduction to	Hugo

		Python	Bowne-Anderson
520	ken@yahoo.com	Arima Models in R	David Stoffer

If we delete Student 230, what happens to the data on Cleaning Data in R?

For example, if you were to delete any of these students, you would lose the course information provided in the columns enrolled_in and taught_by.

This could be resolved if we put that information in another table.
The more normalized the database, the less prone it will be to these anomalies.