# *17* Data Modeling Primer

Learning Objectives

* Know the qualities of good data
* Understand modeling of data for effective organization
* Learn about relational data model
* Understand the Structured Query Language

Data needs to be efficiently structured and stored so that it includes all the information needed for decision making, without duplication and loss of integrity. Here are the top ten qualities of good data.

Data should be

* *Accurate* Data should retain consistent values across data stores, users, and applications. This is the most important aspect of data. Any use of inaccurate or corrupted data to do any analysis is known as the garbage-in-garbage-out (GIGO) condition.
* *Persistent* Data should be available at all times, now and later. It should thus be nonvolatile, stored, and managed for later access.
* *Available* Data should be made available to authorized users, when, where, and how they want to access it, within policy constraints.
* *Accessible* Not only should data be available to the user, but it should also be easy to use. Thus, data should be made available in desired formats with easy tools. MS Excel is a popular medium to access numeric data and then transfer it to other formats.
* *Comprehensive* Data should be gathered from all the relevant sources to provide a complete and holistic view of the situation. New dimensions should be added to data as and when they become available.
* *Analyzable* Data should be available for analysis, for historical and predictive purposes. Thus, data should be organized such that it can be used by analytical tools, such as OLAP, data cube, or data mining.
* *Flexible* Data is growing in a variety of types. Thus, data stores should be able to store a variety of data types – small/large, text/video, and so on.
* *Scalable* Data is growing in volume. Data storage should be organized to meet emergent demands.
* *Secure* Data should be doubly and triply backed up and protected against loss and damage. There is no bigger IT nightmare than corrupted data. Inconsistent data has to be manually sorted out which leads to loss of face, loss of business, downtime, and sometimes the business never recovers.
* *Cost-effective* The cost of collecting data and storing it is coming down rapidly. However, still, the total cost of gathering, organizing, and storing a type of data should be proportional to the estimated value from its use.

### EVOLUTION OF DATA MANAGEMENT SYSTEMS

Data management has evolved from manual filing systems to the most advanced online systems capable of handling millions of data processing and access requests every second.

The first data management systems were called file systems. These mimicked paper files and folders. Everything was stored chronologically. Access to this data was sequential.

The next step in data modeling was to find ways to access any random record quickly. Thus hierarchical database systems appeared. They were able to connect all items for an order, given an order number.

The next step was to traverse the linkages both ways, from the top of the hierarchy to the bottom, and from the bottom to the top. Given an item sold, one should be able to find its order number and list all the other items sold in that order. Thus there were networks of links established in the data to track those relationships.

The major leap came when the relationship between data elements itself became the center of attention. The relationship between data values was the key element of storage. Relationships were established through matching values of common attributes, rather than by location of the record in a file. This led to data modeling using relational algebra. Relations could be joined and subtracted with set operations like union and intersection. Searching the data became an easier task by declaring the values of a variable of interest.

The relational model was enhanced to include variables with non-comparable values like binary objects (such as pictures) which had to be processed differently. Thus, emerged the idea of encapsulating the procedures along with the data elements they worked on. The data and its methods were encapsulated into an *object*. Those objects could be further specialized. For example, a vehicle is an object with certain attributes. A car and a truck are more specialized versions of a vehicle. They inherited the data structure of the vehicle but had their additional attributes. Similarly, the specialized object inherited all the procedures

and programs associated with the more general entity. This became the object-oriented model.

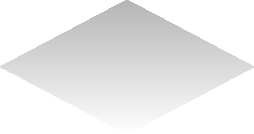
### Relational Data Model

The first mathematical-theory-driven model for data management was designed by Edgar Codd of IBM in 1970.

* A relational database is composed of a set of relations (data tables), which can be joined using shared attributes. A data table is a collection of instances (or records) with a key attribute to uniquely identify each instance.
* Data tables can be joined using the shared key attributes to create larger temporary tables which can be queried to fetch information across tables. Joins can be simple ones as between two tables. Joins can also be complex with AND, OR, UNION or INTERSECTION, and more operations.
* High-level commands in Structured Query Language (SQL) can be used to perform joins, selection, and organizing of records.

Relational data models flow from conceptual models to logical models and physical implementations. Data can be conceived of being entities and relationships among entities. A relationship between entities may be a hierarchy between entities or transactions involving multiple entities. These can be graphically represented as an entity-relationship diagram (ERD).

In Figure 17.1, the rectangle reflects the entities of students and courses. The relationship is enrolment. In the example below, the rectangle reflects the entities students and courses and the diamond shows the enrolment relationship.



STUDENTS

Enroll in



COURSES

FIGURE 17.1 Simple Relationship between Two Entities

Here are some fundamental concepts on ERD

1. An entity is any object or event about which someone chooses to collect data, which may be a person, place, or thing (e.g., salesperson, city, product, vehicle, employee, etc.).
2. Entities have attributes. Attributes are data items that have something in common with the entity. For example, student ID, student name, and student address represent details for a student entity. Attributes can be

single-valued (e.g., student name) or multivalued (list of past addresses for the student). An attribute can be simple (e.g., student name) or composite (e.g., student address, composed of street, city, and state).

1. Every entity must have a key attribute(s) that can be used to identify an instance. For example, a student ID can identify a student. A primary key is the unique attribute value for the instance (e.g., student ID). Any attribute that can serve as a primary key (e.g., student address) is a candidate key. A secondary key—a key that may not be unique, may be used to select a group of records (student city). Some entities will have a composite key—a combination of two or more attributes that together uniquely represents the key (e.g., flight number and flight date). A foreign key is useful in representing a one-to-many relationship. The primary key of the file at the one end of the relationship should be contained as a foreign key on the file at the many end of the relationship.
2. Relationships have many characteristics, e.g., degree, cardinality, and participation.
3. The degree of a relationship depends upon the number of entities participating in a relationship. Relationships can be unary (e.g., employee and manager- as-employee), binary (e.g., student and course), and ternary (e.g., vendor, part, warehouse).
4. Cardinality represents the extent of participation of each entity in a relationship.
   1. One-to-one (e.g., employee and parking space)
   2. One-to-many (e.g., customer and orders)
   3. Many-to-many (e.g., student and course)
5. Participation indicates the optional or mandatory nature of the relationship.
   1. Customer and order (mandatory)
   2. Employee and course (optional)
6. There are also weak entities that are dependent on another entity for its existence (e.g., employees and dependents). If employee data is removed, then the dependent data must also be removed.
7. There are associative entities used to represent a many-to-many relationship (e.g., student-course enrolment). There are two ways to implement a many-to-many relationship. It could be converted into two one-to-many relationships with an associative entity in the middle. Alternatively, the combination of primary keys of the entities participating in the relationship will form the primary key for the associative entity.
8. There are also super subtype entities. This helps represent additional attributes on a subset of the records. For example, a vehicle is a supertype, and a passenger car is its subtype.

### Implementing the Relational Data Model

Once the logical data model has been created, it is easy to translate it into a physical data model which can then be implemented using any public available DBMS. Every entity should be implemented by creating a database table. Every table will be a specific data field (key) that would uniquely identify each relation (or row) in that table. Each master table or database relation should have programs to create, read, update, and delete the records.

The databases should follow 3 integrity constraints.

1. *Entity integrity* ensures that the entity or table is healthy. The primary key cannot have a null value. Every row must have a unique value or else that row should be deleted. As a corollary, if the primary key is composite, none of the fields participating in the key can contain a null value. Every key must be unique.
2. *Domain integrity* is enforced by using rules to validate the data as being of the appropriate range and type.
3. *Referential integrity* governs the nature of records in a one-to-many relationship. This ensures that the value of a foreign key should have a matching value in the primary keys of the table referred to by the foreign key.

### Database Management Systems (DBMS)

These are many database management software systems that help manage the activities related to storing the data model, the data itself, and doing the operations on the data and relations. The data in the DBMS grows and it serves many users of the data concurrently. The DBMS typically runs on a computer called a database server – in the *n*-tier application architecture. Thus in an airline reservation system, millions of transactions might simultaneously try to access the same set of data. The database is constantly monitored and managed to provide data access to all authorized users, securely and speedily, while keeping the database consistent and useful. Content management systems are special-purpose DBMS or just features within standard DBMS that help people manage their data on a website. There are object-oriented and other more complex ways of managing data.

### Structured Query Language

SQL is a very easy and powerful language to access relational databases. There are two essential components of SQL – the Data Definition Language (DDL) and Data Manipulation Language.

DDL provides instructions to create a new database and to create new tables within a database. Further, it provides instructions to delete a database or just a few tables within a database. There are other ancillary commands to define indexes etc., for efficient access to the database.

DML is the heart of SQL. It provides instructions to add, read, modify, and delete data from the database and any of its tables. The data can be selectively accessed and then formatted to answer a specific question. For example, to find the sales of movies by quarter, the SQL query would be

SELECT Product-Name, SUM (Amount)

FROM Movies-Transactions

GROUP BY Product-Name

Conclusion

Data should be modeled to achieve business objectives. Good data should be accurate and accessible so that it can be used for business operations. The relational data model is the most popular way of managing data today.

## Questions

1. Who invented the relational model and when?
2. How does the relational model mark a clear break from previous database models?
3. What is an Entity-Relationship diagram?
4. What kinds of attributes can an entity have?
5. What are the different kinds of relationships?