Database Systems
Theory Assignment - 1

C. Jayitha
20171401

September 23, 2018

1 Data Models

The process of defining the conceptual design of data elements and their inter-relationships is called data modeling. The traditional applications approach to data organization built different models for each data file.

Such a diversity of ways in which different data elements are linked and stored in data files make these files suitable only for the applications that they were originally created for. In fact, the details regarding the exact placement of different data elements in a file have to be documented very carefully.

Any change in the order in which various data elements are placed results in changes in the application programs using the data file. The database approach uses a common data model for the entire database and the user program is not concerned with the placement of a particular data element. The database management system (DBMS) acts as an interface between the database and the user programs.

The DBMS fetches the data from the database and makes it available to the user program. This feature offers the advantage of data independence in the database approach.

A database model shows the logical structure of a database, including the relationships and constraints that determine how data can be stored and accessed. Individual database models are designed based on the rules and concepts of whichever broader data model the designers adopt. Most data models can be represented by an accompanying database diagram.

Most database management systems are built with a particular data model in mind and require their users to adopt that model, although some do support multiple models.

In addition, different models apply to different stages of the database design process. High-level conceptual data models are best for mapping out relationships between data in

ways that people perceive that data. Record-based logical models, on the other hand, more closely reflect ways that the data is stored on the server.

Selecting a data model is also a matter of aligning your priorities for the database with the strengths of a particular model, whether those priorities include speed, cost reduction, usability, or something else.

1.1 Hierarchical Model

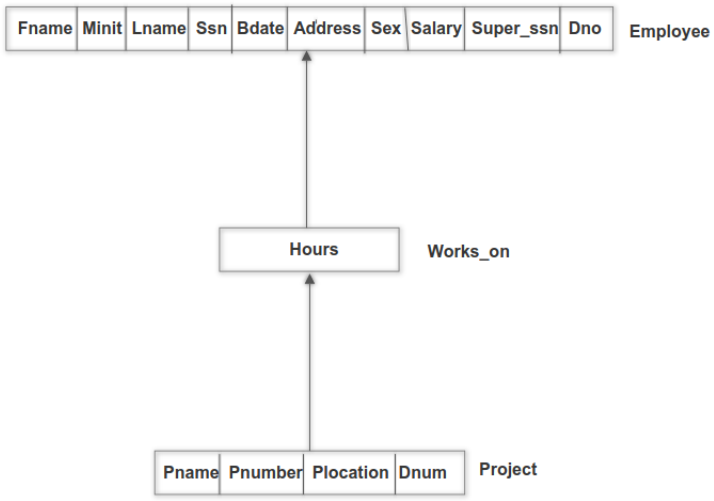

Figure 1: Tree Structure Diagram representing one-to-many mapping from Employee to Project (in actual it's many-to-many)

The hierarchical model organizes data into a tree-like structure, where each record has a single parent or root with one-to-many relationship between two different types of data, for example, one department can have many courses, many professors and of-course many students.A sort field keeps sibling records in a particular order. That order is used as the physical order for storing the database. This model is good for describing many real-world relationships.

This model was primarily used by IBMs Information Management Systems in the 60s and 70s, but they are rarely seen today due to certain operational inefficiencies.

This model presents data to users in a hierarchy of data elements that can be represented in a sort of inverted tree. In a sales order processing system, a customer may have many invoices raised to him and each invoice may have different data elements. Thus, the root level of data is customer, the second level is invoice and the last level is line items such as invoice number, date, product, quantity, etc.

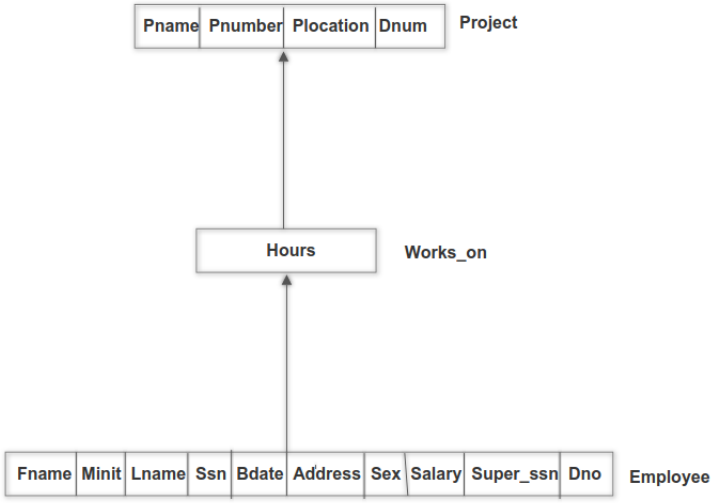

Figure 2: Tree Structure Diagram representing one-to-many mapping from Project to Employee. Both tree structures needed to represent many-to-many mapping

This structure is quite natural when seen from the event point of view. However, the lower levels are owned by higher level data elements, and elements at the same level have no linkage at all. As a result, the query such as what products are purchased by which customer, in the above example, shall be difficult to carry out in the hierarchical structure.

The query as to which customer purchased which product would be convenient. Thus, where there are many-to-many relationships between two entities, this model would not be appropriate.

A hierarchical database consists of a collection of records that are connected to each other through links. A record is a collection of fields, each of which contains only one data value. A link is an association between precisely two records. The hierarchical model is thus similar to the network model in the sense that data and relationships between data are also represented by records and links, respectively. The hierarchical model differs from the network model in that the record types are organized as collections of trees, rather than as arbitrary graphs. A tree-structure diagram is a schema for a hierarchical database. Such a diagram consists of two basic components: boxes, which correspond to record types, and lines, which correspond to links. A tree-structure diagram serves the same purpose as an E-R diagram; it specifies the overall logical structure of the database. A tree-structure diagram is similar to a data-structure diagram in the network model. The main difference is that, in the former, record types are organized in the form of an arbitrary graph, whereas in the latter, record types are organized in the form of a rooted tree. For every E-R diagram, there is a corresponding tree-structure diagram. The database schema is thus represented as a collection of tree-structure diagrams. For each such diagram, there exists a single instance of a database tree. The root of this tree is a dummy node. The children of the dummy node

are instances of the root record type in the tree-structure diagram. Each record instance may, in turn, have several children, which are instances of various record types, as specified in the corresponding tree-structure diagram.

Record access is done by navigating downward through the data structure using pointers combined with sequential accessing. Because of this, the hierarchical structure is inefficient for certain database operations when a full path (as opposed to upward link and sort field) is not also included for each record. Such limitations have been compensated for in later IMS versions by additional logical hierarchies imposed on the base physical hierarchy.

In the case of many-to-many relationships, record replication is necessary to preserve the tree-structure organization of the database. Record replication has two major drawbacks: data inconsistency may result when updating takes place and waste of space is unavoidable. The solution is to introduce the concept of a virtual record. Such a record contains no data value; it does contain a logical pointer to a particular physical record. When a record needs to be replicated, a single copy of the actual record is retained, and all other records are replaced with a virtual record containing a pointer to that physical record. The data-manipulation language for this new configuration remains the same as in the case where record replication is allowed. Thus, a user does not need to be aware of these changes. Only the internal implementation is affected. Implementations of hierarchical databases do not use parent-to-child pointers, since that would require the use of variable-length records. Instead, they use preorder threads. This technique allows each record to contain exactly two pointers. Optionally, a third child-to-parent pointer may be added.

In conclusion, the notion of data independence is not prominent in this data model, but there is still some levy. Query processing is efficient only for particular types of queries else multiple trees are needed. It's easier to model applications because Hierarchy models perfectly model real life mini worlds. But it has some major drawbacks, for one it only processes one-many relationships which is highly inconvenient and the data redundancy can cause inconsistencies .

1.2 Network Model

The network model builds on the hierarchical model by allowing many-to-many relationships between linked records, implying multiple parent records. Based on mathematical set theory, the model is constructed with sets of related records. Each set consists of one owner or parent record and one or more member or child records. A record can be a member or child in multiple sets, allowing this model to convey complex relationships.

It was most popular in the 70s after it was formally defined by the Conference on Data Systems Languages (CODASYL).

In the network model of database, there are no levels and a record can have any number of owners and also can have ownership of several records. Thus, the problem raised above in the sales order processing will not arise in the network model.

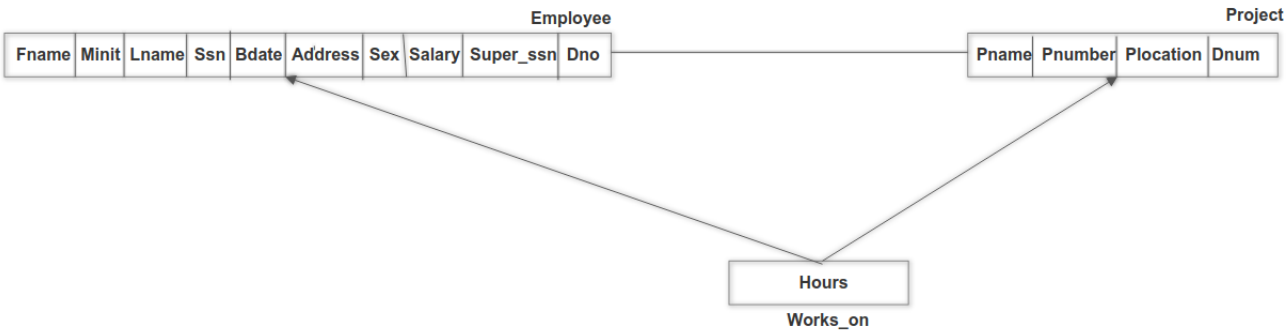

Figure 3: Figure shows a Data-Structure Diagram of the many-to-many relationship between Employee and Project

As there is no definite path defined for retrieval of data, the number of links is very large and thus network databases are complex, slow and difficult to implement. In view of the difficulty in implementation, network model is used only when all other options are closed.

This is an extension of the Hierarchical model. In this model data is organized more like a graph, and are allowed to have more than one parent node.

In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and fast. This database model was used to map many-to-many data relationships.

This was the most widely used database model, before Relational Model was introduced.

The network model organizes data using two fundamental concepts, called records and sets. Records contain fields (which may be organized hierarchically, as in the programming language COBOL). Sets (not to be confused with mathematical sets) define one-to-many relationships between records: one owner, many members. A record may be an owner in any number of sets, and a member in any number of sets.

A set consists of circular linked lists where one record type, the set owner or parent, appears once in each circle, and a second record type, the subordinate or child, may appear multiple times in each circle. In this way a hierarchy may be established between any two record types, e.g., type A is the owner of B. At the same time another set may be defined where B is the owner of A. Thus all the sets comprise a general directed graph (ownership defines a direction), or network construct. Access to records is either sequential (usually in each record type) or by navigation in the circular linked lists.

The network model is able to represent redundancy in data more efficiently than in the hierarchical model, and there can be more than one path from an ancestor node to a descendant. The operations of the network model are navigational in style: a program maintains

a current position, and navigates from one record to another by following the relationships in which the record participates. Records can also be located by supplying key values.

Although it is not an essential feature of the model, network databases generally implement the set relationships by means of pointers that directly address the location of a record on disk. This gives excellent retrieval performance, at the expense of operations such as database loading and reorganization.

Popular DBMS products that utilized it were Cincom Systems' Total and Cullinet's IDMS. IDMS gained a considerable customer base; in the 1980s, it adopted the relational model and SQL in addition to its original tools and languages.

Most object databases (invented in the 1990s) use the navigational concept to provide fast navigation across networks of objects, generally using object identifiers as "smart" pointers to related objects. Objectivity/DB, for instance, implements named one-to-one, one-to-many, many-to-one, and many-to-many named relationships that can cross databases. Many object databases also support SQL, combining the strengths of both models.

A network database consists of a collection of records that are connected to each other through links. A link is an association between precisely two records. Records are organized in the form of an arbitrary graph. A data-structure diagram is a schema for a network database. Such a diagram consists of two basic components: boxes, which correspond to record types, and lines, which correspond to links. A data-structure diagram serves the same purpose as an E-R diagram; namely, it specifies the overall logical structure of the database. For every E-R diagram, there is a corresponding data-structure diagram. In the late 1960s, several commercial database systems based on the network model emerged. These systems were studied extensively by the Database Task Group (DBTG) within the CODASYL group. In the DBTG model, only many-to-one links can be used. Many-to-many links are disallowed to simplify the implementation. One-to-one links are represented as many-to-one links. A data-structure diagram consisting of two record types that are linked together is referred to, in the DBTG model, as a DBTG set. Each DBTG set has one record type designated as the owner of the set, and another record type designated as a member of the set. A DBTG set can have any number of set occurrences.

Implementation techniques for the DBTG model exploit the restrictions of the model to allow the physical representation of DBTG sets without the need for variable-length records. A DBTG set is represented by one ring structure for each occurrence

In conclusion, the network model is comparatively better at controlling data redundancy as it can have many-to-many relationships but it's complexity makes it hard to implement and hence building applications on the network model gets increasingly difficult.

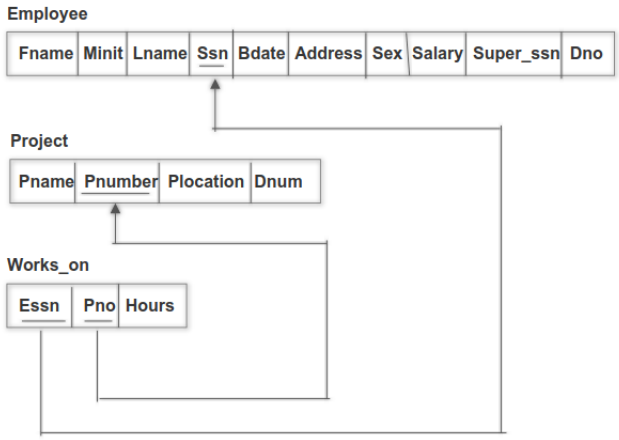

Figure 4: Figure shows the relations Employee, Project and relationship Works on

1.3 Relational Model

The most common model, the relational model sorts data into tables, also known as relations, each of which consists of columns and rows. Each column lists an attribute of the entity in question, such as price, zip code, or birth date. Together, the attributes in a relation are called a domain. A particular attribute or combination of attributes is chosen as a primary key that can be referred to in other tables, when its called a foreign key.

Each row, also called a tuple, includes data about a specific instance of the entity in question, such as a particular employee.

In this model, data is organized in two-dimensional tables and the relationship is maintained by storing a common field.

This model was introduced by E.F Codd in 1970, and since then it has been the most widely used database model, infact, we can say the only database model used around the world.

The basic structure of data in the relational model is tables. All the information related to a particular type is stored in rows of that table.

Hence, tables are also known as relations in relational model.

The model also accounts for the types of relationships between those tables, including one-to-one, one-to-many, and many-to-many relationships.

Within the database, tables can be normalized, or brought to comply with normalization rules that make the database flexible, adaptable, and scalable. When normalized, each piece of data is atomic, or broken into the smallest useful pieces.

The relational model draws greatly on the work of E.F. Codd who identifies features of a good relational database as following:

a) All information is logically represented as tables and the access of data are possible by the names of fields. Thus, the order, position or file linkage is not a matter of concern for users.

b) The data dictionary has information regarding the database structure including the data type; size, etc., definitions, relationships and access permissions. The authorized users can learn about the database environment and change the environment using the data description language (DDL).

c) A data manipulation language (DML) is available to users including programmers for creation, insertion, modification, retrieval, organizing and deletions of any part of the database. These manipulations are possible at the record level as well as for the whole file, giving the flexibility in defining access permissions for various categories of users.

d) Any modification in the structure of database in terms of splitting the table horizontally or vertically should not have any impact on the logic of the program using the database. This data independence is the core advantage of the relational model of database.

e) The distributed independence of data is another feature of a good relational database. The user programs do not require any change when data are first distributed or redistributed. The actual physical location of data does not matter to the user so long as that field appears in the data dictionary as local.

In conclusion, the relational database model is capable of handling all aspects of data independence, ease of application development and query processing (depending on the kind of query) faster and more efficiently.