BCA Part II
Paper-XI: DBMS Using MS-ACCESS
Topic: Introduction

Prepared by: Dr. Kiran Pandey
(School of Computer Science)

Email-id: kiranpandey.nou@gmail.com

Database is a collection of interrelated data and a set of programs used to access those data. Data is the collection of facts and figures that can be processed to produce information. Data is used for producing information, which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks. Some more examples include deposit and/or withdrawal from a bank, airline or railway reservation, accessing a computerized library, order a magazine subscription from a publisher, purchase items from supermarkets etc.

In all the above cases a database is accessed. These may be called **Traditional Database Applications.** In these types of databases, the information stored and accessed is textual or numeric. However, with advances in technology in the past few years, different databases have been developed such as **Multimedia Databases** that store pictures, video clips and sound messages. **Geographical Information Systems (GIS)** that can store maps, weather data and satellite images, etc., and **Real time databases** that can control industrial and manufacturing processes.

The primary goal of a DBMS is to provide a way to store and retrieve database information that is both convenient as well as efficient.

ELEMENTS OF DATABASE

Database Schema

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data. A *schema* is quite simply a group of related objects in a database. Within a schema, objects that are related have relationships with one another.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

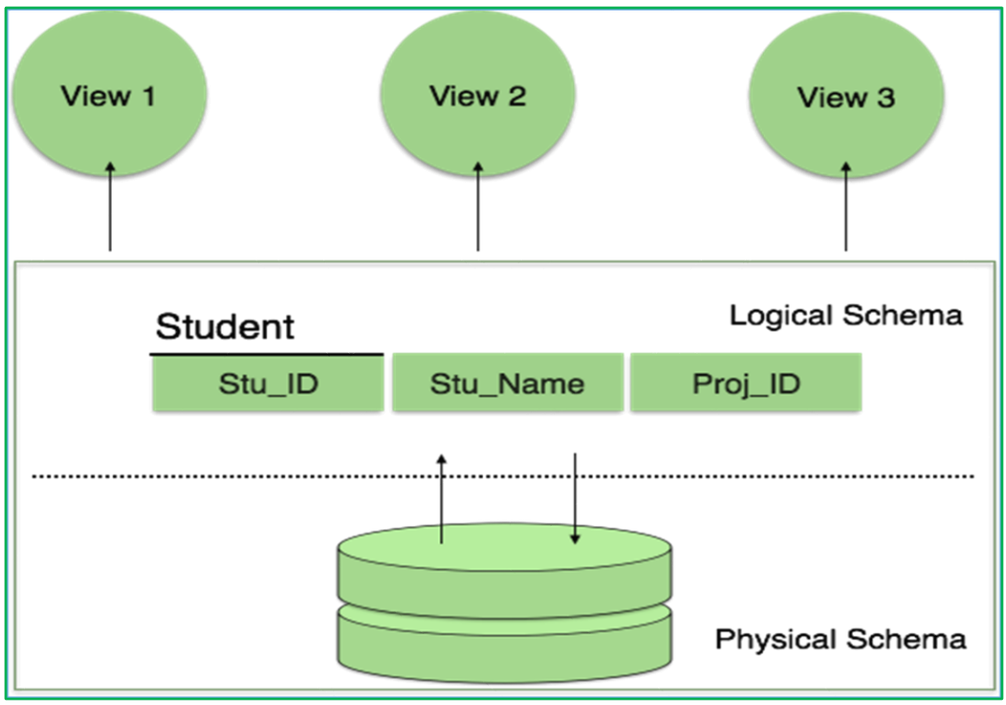


Figure 1: Database Schema

A database schema can be divided broadly into two categories −

- **Physical Database Schema** − this schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
- **Logical Database Schema** − this schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

The three models associated with a schema are as follows:

- The ***conceptual* model** is the basic database model, which deals with organizational structures that are used to define database structures such as tables and constraints. It is also known as the *logical* model
- The ***internal* model** deals with the physical storage of the database, as well as access to the data, such as through data storage in tables and the use of indexes to expedite data access. The internal model separates the physical requirements of the hardware and the operating system from the data model. It is also called known as the *physical* model.
- The ***external* model** deals with methods through which users may access the schema, such as through the use of a data input form. The external model allows relationships to be created between the user application and the data model. This also known as ***application interface.***

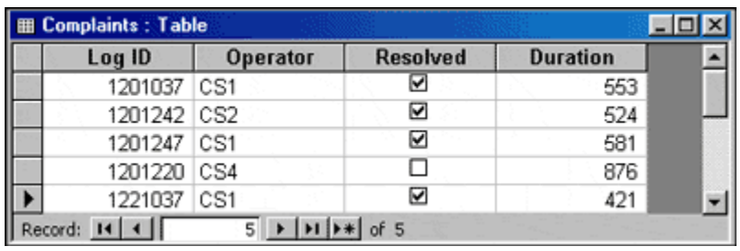
Database Instance

It is important that we distinguish these two terms individually. Database schema is the skeleton of database. It is designed when the database doesn't exist at all. Once the database is operational, it is very difficult to make any changes to it. A database schema does not contain any data or information.

A database instance is a state of operational database with data at any given time. It contains a snapshot of the database. Database instances tend to change with time. A DBMS ensures that its every instance (state) is in a valid state, by diligently following all the validations, constraints, and conditions that the database designers have imposed.

Table

The table is the most fundamental element found in a database schema. Columns and rows are associated with tables. Tables, columns, and rows are discussed below. A table is the primary unit of physical storage for data in a database. When a user accesses the database, a table is usually referenced for the desired data. Multiple tables might comprise a database, therefore a relationship might exist between tables. A database table is composed of records and fields that hold data. Tables are also called datasheets. Each table in a database holds data about a different, but related, subject.



Tables are used to store the data that the user needs to access. Tables might also have constraints attached to them, which control the data allowed to be entered into the table.

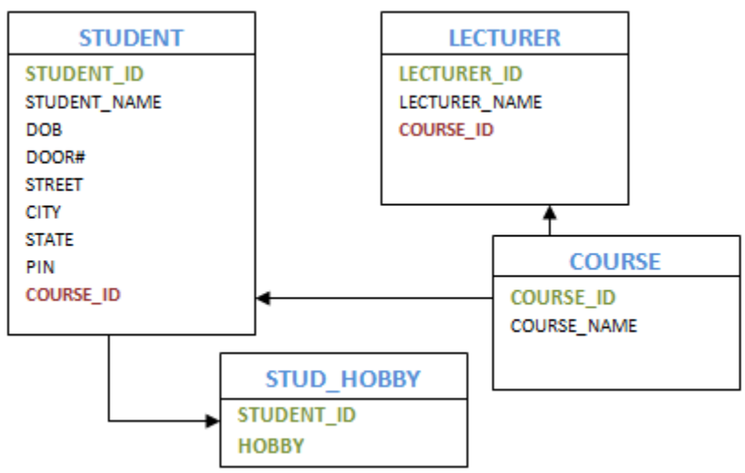
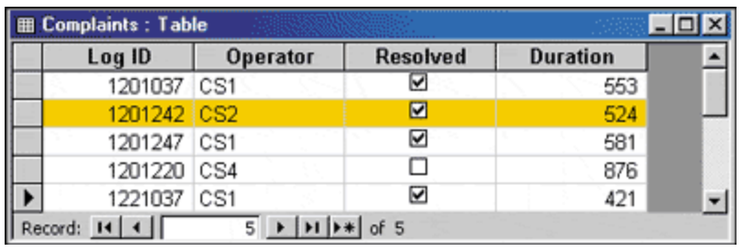


Figure 2 - Database tables and their relationships.

Record

A record or a *row* of data is the collection of all the columns in a table associated with a single occurrence. Simply speaking, a row of data is a single record in a table. Data is stored in records. A record is composed of fields and contains all the data about one particular person, company, or item in a database. In this database, a record contains the data for one customer support incident report. Records appear as rows in the database table. A record for Log ID 1201242 is highlighted in Figure below.

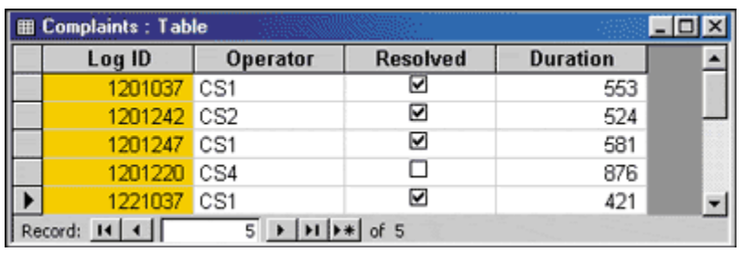


Field

A field is part of a record and contains a single piece of data for the subject of the record. In the database table illustrated in Figure below, each record contains four fields:

Log ID	A number assigned to this customer support incident for identification purposes
Operator	The code for the customer support operator who handled this incident
Resolved	A check box to indicate whether the incident was resolved
Duration	The time in seconds the operator spent on this incident

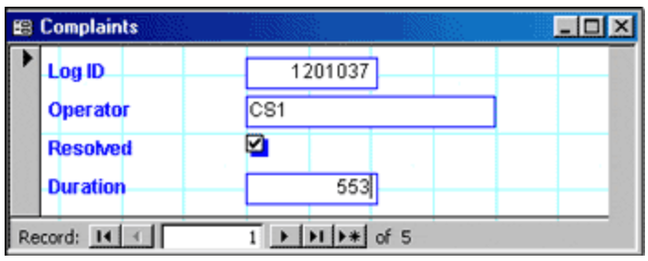
Fields appear as columns in a database table. Data from the Log ID field for five records is highlighted in the Figure below.



In other words, when a business model is converted into a database model, entities become tables and attributes become columns. A column represents one related part of a table and is the smallest logical structure of storage in a database. Each column in a table is assigned a data type. The assigned data type determines what type of values that can populate a column. When visualizing a table, a column is a vertical structure in the table that contains values for every row of data associated with a particular column.

Forms

A database form shows all or selected fields for one record. Forms show field names and data in an attractive and easy-to-read format. Record 1 of the database is shown in Figure below.



You can enter data in fields in a form just as you can in a table. Data has been entered in three fields in the form shown in Figure below. Navigation buttons for moving from record to record are displayed at the bottom of the form.

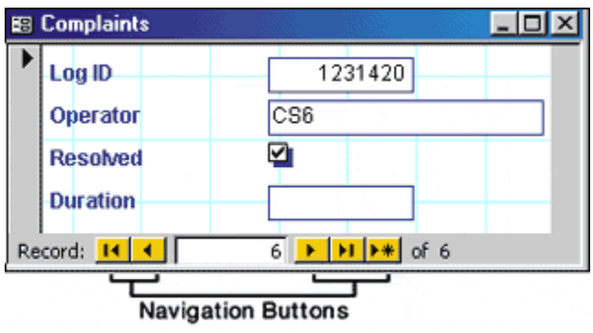


Figure3: Use navigation buttons to move from record to record.

Queries

A query finds records in a database according to criteria you specify. Sometimes working with all the data in a large database is not practical. You might wish to work with just a small part or subset of the data. For example, you might want to find records for unresolved customer support incidents. A query can be used to display these records. You could construct the query to display only selected fields, such as Log ID and Resolved. A query displays only the data you request.

The table in Figure below shows all data that has been entered in the database. An empty checkbox in the **Resolved** field indicates the incident has not been resolved.

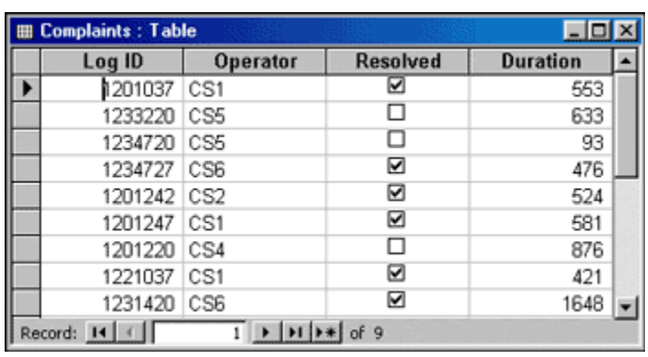


Figure 4: Table Displaying All Data

Figure below shows the results of a query. Only the Log ID and Resolved fields for unresolved incidents are displayed.

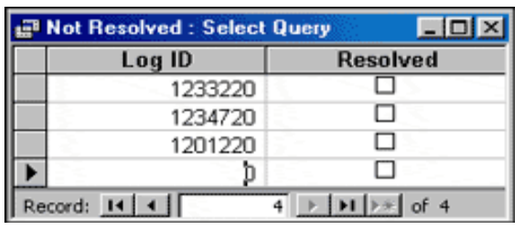


Figure 5: Data from a Query

Reports

A report presents data in an attractive format and is especially suitable for printing. Reports can display data from tables or queries. All or selected fields can be included in a report. Data can be grouped or sorted and arranged in a variety of ways.

Figure below shows a report containing data from the Complaints table. The data is grouped by resolved and unresolved incidents. The Log ID field is sorted in ascending order. Compare the format of the data in this report to the database table shown in Figure.

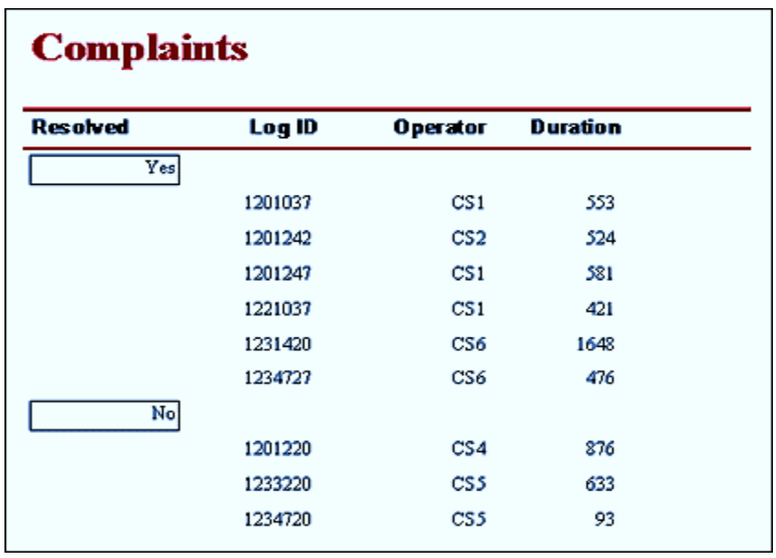


Figure 6: Database Report

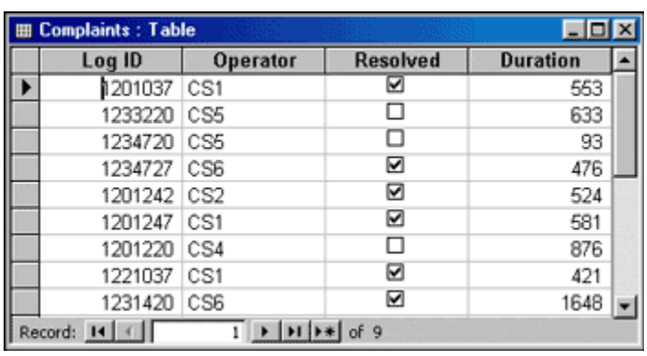


Figure 7: Database Table

Data Types

A data type determines the type of data that can be stored in a database column. Although many data types are available, three of the most commonly used data types are:

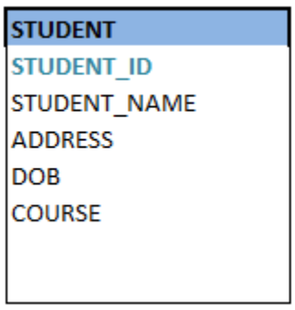
- **Alphanumeric**

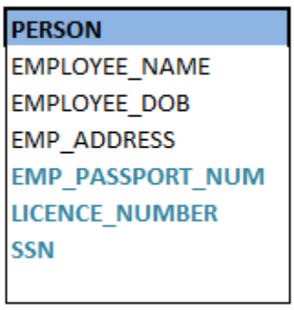
- **Numeric**
- **Date and time**

Alphanumeric data types are used to store characters, numbers, special characters, or nearly any combination. If a numeric value is stored in an alphanumeric field, the value is treated as a character, not a number. In other words, you should not attempt to perform arithmetic functions on numeric values stored in alphanumeric fields. **Numeric data types** are used to store only numeric values. *Date and time* data types are used to store date and time values, which widely vary depending on the relational database management system (RDBMS) being used.

KEYS

The integrity of the information stored in a database is controlled by keys. A *key* is a column value in a table that is used to either uniquely identify a row of data in a table, or establish a relationship with another table. A key is normally correlated with one column in table, although it might be associated with multiple columns. There are two types of keys: primary and foreign. Any attribute in the table which uniquely identifies each record in the table is called key. It can be a single attribute or a combination of attributes. For example, in STUDENT table, STUDENT_ID is a key, since it is unique for each student. In PERSON table, his passport number, driving license number, phone number, SSN, email address is keys since they are unique for each person.

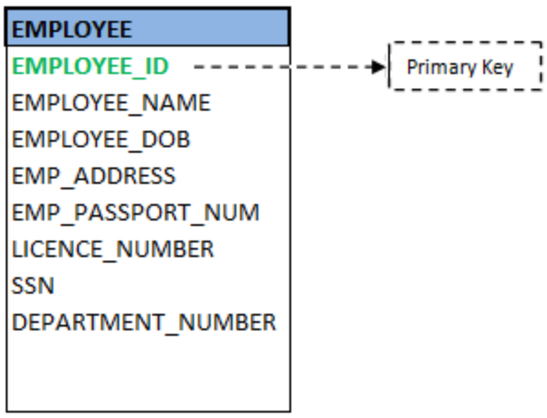




(i) Primary Key

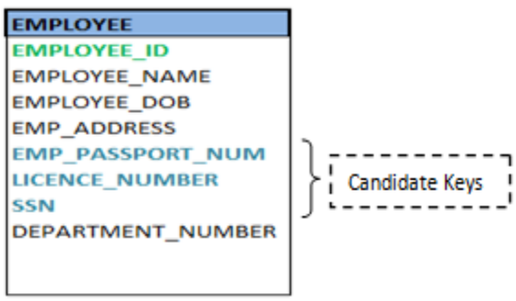
It is the first and foremost key which is used to uniquely identify a record. It can be a single attribute or a combination of attributes. Even if a table has no child table, a primary key can be used to disallow the entry of duplicate records into a table. For an entity, there could be multiple keys as we saw in PERSON table. Most suitable key from those lists becomes a primary key. In the Person table above, we can select SSN as primary key, since it is unique for each person. We can even select Passport Number or license number as primary key as they are also unique for a person. However, selection of primary key for each entity is based on requirement and developer.

For a student, STUDENT_ID is a primary key and for an employee EMPLOYEE_ID is a primary key.



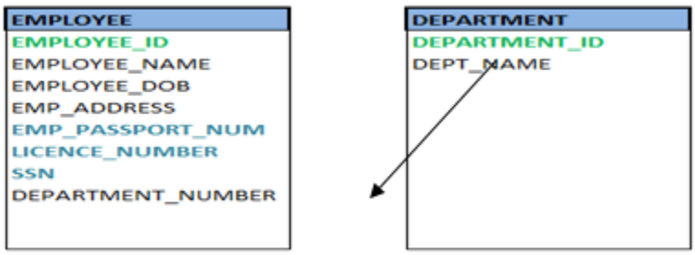
(ii) Candidate Keys

As we discussed above, an employee is identified by his ID in his office. Apart from his ID, does he have any other unique keys, so that he can be identified from others? Yes, he has passport number, PAN number, SSN number (if applicable), driving license number, email address etc. These are also identifies specific person uniquely. But we can choose any one of these unique attribute as primary key in the table. Rest of the attributes, which holds as strong as primary key are considered as Candidate key/secondary key. In our example of employee table, EMPLOYEE_ID is best suited for primary key as its from his own employer. Rest of the attributes like passport number, SSN, license Number etc are considered as candidate key.



(iii) **Foreign Key**

In a company there would be different departments - Accounting, Human Resource (HR), development, Quality, etc. An employee, who works for that company, works in specific department. But we know that employee and department are two different entities. So we cannot store his department information in employee table. Instead what we do is we link these two tables by means of primary key of one of the table i.e.; in this case, we pick the primary key of department table - DEPARTMENT_ID and add it as a new attribute/column in the Employee table. Now DEPARTMENT_ID is a foreign key for Employee table, and both the tables are related!



(iv) Composite Key

A key in a table is formed by combining more than one attributes/columns of the same table. These columns of the table **can or cannot be keys** in the table. The composite key acts as a primary key only when all the columns in the composite keys are together, individually those columns are not keys. In other words, unique record from the table is fetched only if we combine more than one column. If we use them individually, we will not get any unique record.

In the example of M: N relationship - 'Student enrolls for a course', STUDENT_ID and COURSE_ID, when combined together gives the particular course to which he is enrolled for. Only STUDENT_ID or COURSE_ID alone does not inform correct data.

In the table above, STUDENT_ID, 100 alone gives us multiple courses. To know about particular course we need both STUDENT_ID and COURSE_ID. In this case, both the IDs are primary keys from their table, but in STUDENT_COURSE table, they form primary key when they are combined together. Hence they are compound key.

STUDENT_COURSE

STUDENT_ID	COURSE_ID	No_of_Hours
100	401	20
100	402	15
101	401	25
103	403	10