Introduction to Database Management System

Introduction

In today's digital world, organizations generate vast amounts of data every second. This data needs to be saved, processed and organized in a proper manner so that it can be used later to make decisions. Understanding the difference between data and information thus becomes important for effective data management. A database serves as a structured repository allowing for efficient retrieval and manipulation of data. In this chapter, we will learn about the difference between data and information. We will look at the Database Management System (DBMS) and its components. We will also create a template of a sample database and learn about data types that play an important role in databases.

Data vs Information

Sometimes, the users use the terms Data and Information synonymously. Therefore, it is necessary to know the exact meaning of both these terms. Data and information can be defined in a number of ways, so let us start by figuring out what those might be.

Data refers to unorganized facts, figures and details related to people, places, things or events. It is often considered to be raw. Data can be in any format like numbers, text, images or video . It could be written or spoken. Data in its raw state may not be very helpful. Given the significance of data in the decision-making process, several companies view it as a key asset. The basic property of data is that it is irrelevant at times and often unstructured.

Information is processed data. The raw data when passed through a transformation process of some kind gets converted into information. The basic property of information is that it is structured, relevant and meaningful.

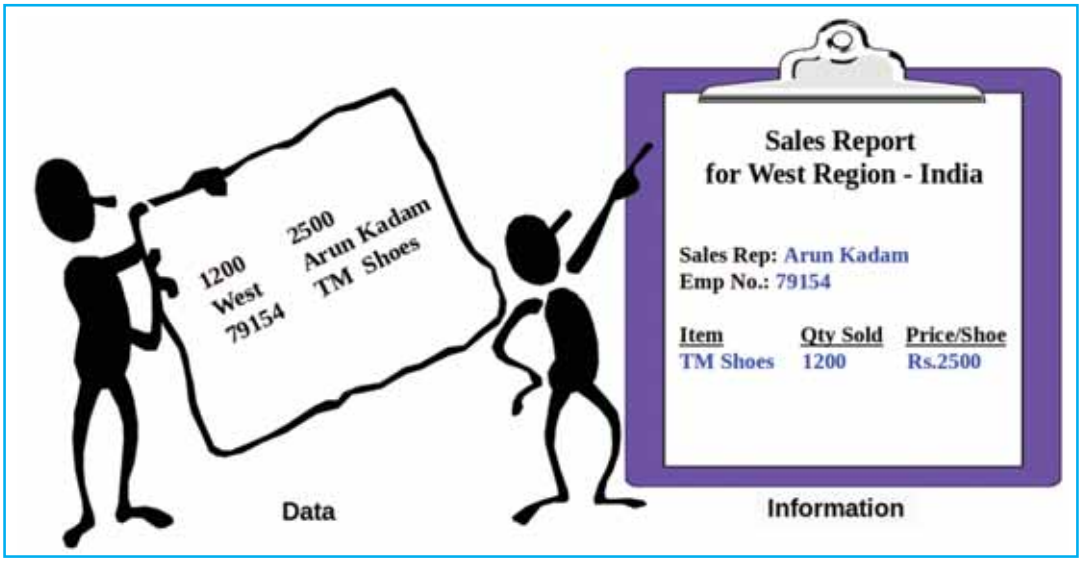


Figure 1.1 : Difference between Data and Information

Let us try to understand the difference between data and information by taking an example. A sequence of numbers and text 1200, 2500, 79154, "West", "Arun Kadam", and "TM Shoes" is just data. Individually each of them does not seem to be relevant. However, if you put it into context: "Mr. Arun Kadam, Emp No.: 79154 has sold 1200 TM Shoes at Rs. 2500 each in the West Region of India", it becomes information because it provides meaning and context. Figure 1.1 shows the difference between data and information.

What is Database?

Today all of us use one or more types of databases in our day-to-day life. A most understandable example of a database would be the contact book on our mobile phone that contains contact numbers and emails of our acquaintances. The details pertaining to an individual acquaintance is known as a record.

Database is a collection of multiple related data items stored in a properly organized manner. According to this definition, there are two key points to be considered when we mention a database. First; it must represent data items that are related to each other and second, it must be an organized collection.

Logical arrangement of things makes searching always easier as and when required. The databases are often designed according to a predefined set of rules and data models. The data model describes a way of storing and retrieving the data. There are different data models in DBMS like hierarchical, network, relational, object-oriented, flat, semi-structured, associative and context. Two of these models, flat and relational, play a significant role. The flat model was the initial model of storing data while the relational model is the most widely used model today.

Flat Data Model

Data is represented in the flat data model as a single, two-dimensional table with no defined connections. The quantity and kind of fields are consistent across all records. Due to its simplicity and ease of comprehension, this format is ideal for small or one-time datasets. This format is frequently used in comma-separated values (CSV), LibreOffice Calc and MS Excel files. A sample dataset of a CSV file and LibreOffice Calc is shown in figure 1.2 and 1.3.

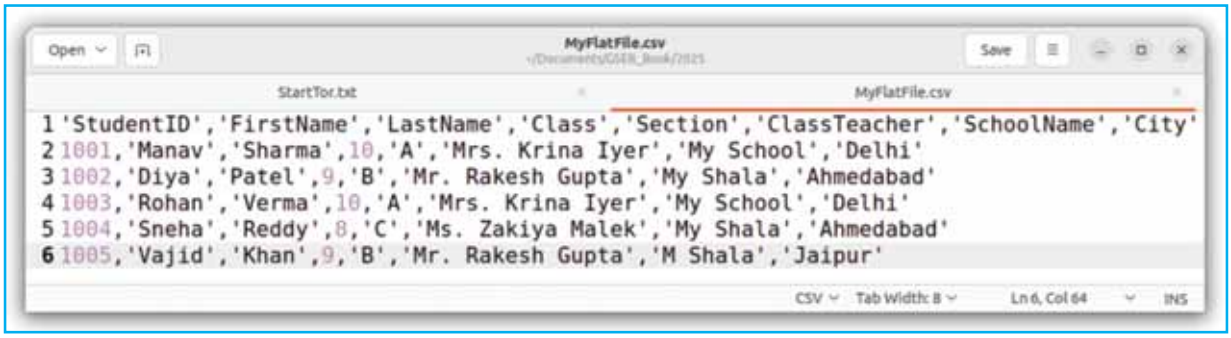
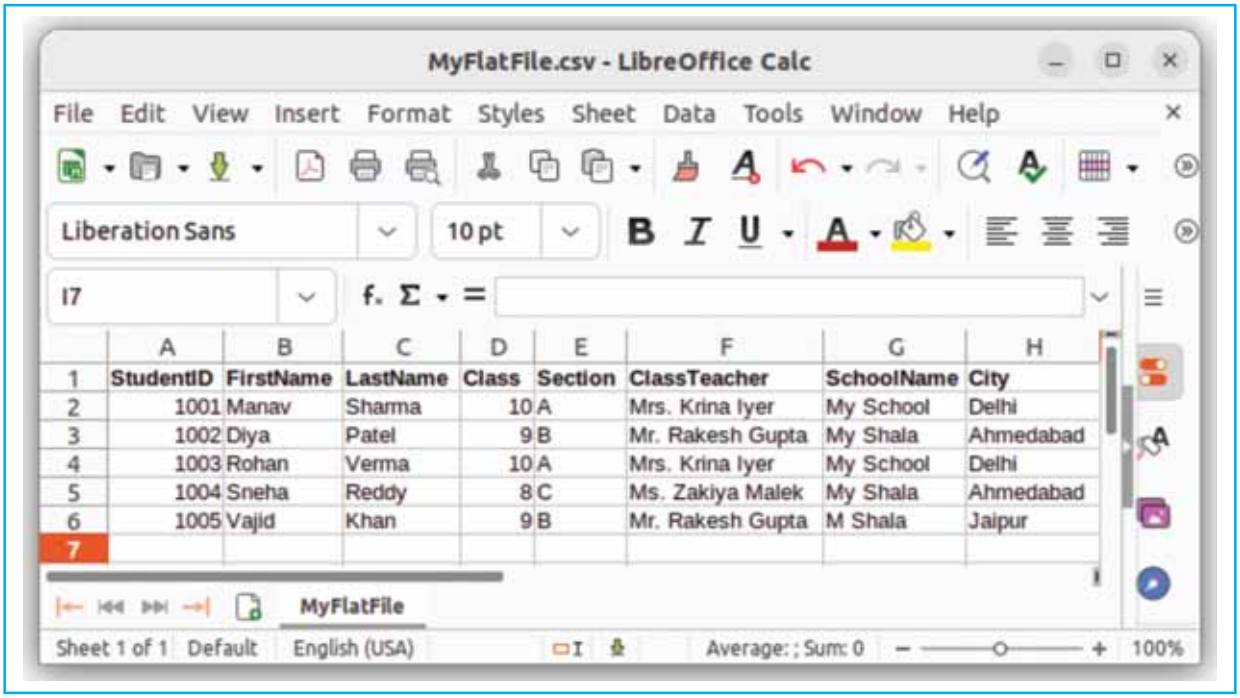


Figure 1.2 : Example of Flat Data Model (CSV file)



	A	B	C	D	E	F	G	H
1	StudentID	FirstName	LastName	Class	Section	ClassTeacher	SchoolName	City
2	1001	Manav	Sharma	10	A	Mrs. Krina Iyer	My School	Delhi
3	1002	Diya	Patel	9	B	Mr. Rakesh Gupta	My Shala	Ahmedabad
4	1003	Rohan	Verma	10	A	Mrs. Krina Iyer	My School	Delhi
5	1004	Sneha	Reddy	8	C	Ms. Zakiya Malek	My Shala	Ahmedabad
6	1005	Vajid	Khan	9	B	Mr. Rakesh Gupta	M Shala	Jaipur
7								

Figure 1.3 : Example of Flat Data Model

The flat database model is very simple and easy to understand. It though cannot be used for complex or large-scale data sets due to several drawbacks. Few of the drawbacks of flat database models are as mentioned:

Data Redundancy: There are very high chances that the same data is repeated multiple times in flat databases. Observe that in figure 1.3 class teacher's name appears multiple times.

Update Anomalies: As the data is repeated multiple times, if the value, for example, My School is to be updated then we need to change data in each and every row that has this data. If the number of rows are too many then it increases the risk of errors.

Limited Modularity: As can be seen in figure 1.3, all data is stored in a single table. This makes it very hard to isolate changes or reuse the data components in different contexts.

Poor Scalability: For small amounts of data say 100 records, flat data model works fine. As the data grows, managing, updating, and querying become inefficient. The lack of structure further leads to performance issues.

No Data Relationships: Representing real-world structures is difficult in a flat data model as it can not define relationships between different entities. For example, in figure 1.3 identifying the relation between students-teachers or classes-subjects is quite difficult.

Data Integrity Issues: Maintaining consistency during operation might become difficult thus creating data integrity issues. For example, while updating a school name a typographical error may go undetected.

Querying Complex Data is Difficult: The database is required to make some decisions in an organization. In flat data models, performing analytics or extracting meaningful relationships becomes very hard due to unavailability of structured links between data.

Relational Data Model

The relational data model solves all the issues that the flat data model faces. The data in the relational data model is arranged in the form of tables (relationships). The tables are made up of rows and columns. A table is used to represent a particular entity, and the relationships between different tables are controlled using keys (fields of tables). The database shown in figure 1.3 thus can be divided into tables that may represent entities like student, teacher, school and others.

The relational data model makes use of Structured Query Language (SQL) (an english like statements) for querying the complex databases. Because of its simplicity, adaptability and solid theoretical basis, the model is used extensively. We will learn how to create tables in the upcoming sections of this chapter.

Need of Database

Businesses today thrive on data, without data it would be impossible for them to come up with new solutions and products. So let us now look at what makes a database so important and what is its need. The following requirements of an organization highlight the need of a database.

Data Organization and Structure

As mentioned earlier databases provide a systematic way or approach to organize the large amounts of data generated by the organization in structured formats. It enables logical grouping of related information thus leading to efficient data storage.

Creation of multiple tables like student, teacher and school thus helps us organize the data that would be required for any school management activities.

Efficient Data Retrieval

Businesses need to make quick decisions based on the data that they have. As data is stored in logical groups it enables fast searching and retrieval of the required information.

Suppose we wanted to find details about a student, it could be easily retrieved by looking into the data stored in the student table. Additional information if required for example, marks of students could be fetched by using data from another related table that stored marks.

Data Integrity and Consistency

No business would like to have incorrect data. Creating a database that can enforce validation rules on data allows the businesses to maintain accurate and consistent data across all their operations.

As data is organized in a logical and structured manner any updates required would be done only at a single place. The effect of this change would then be reflected across the entire dataset. For example, if we changed the name of the school, we would only change it in the school table. The updated value would then be reflected across all tables.

Concurrent Access

Many times a single piece of data needs to be accessed by multiple stakeholders. The database allows multiple users to access and modify data simultaneously without conflicts. In the case of figure 1.3, the class teacher's name may be accessed by the student as well as the school principal.

Access Control

Data needs to be often shared but in a secured and restricted manner. Database provides role-based permissions and user-level security to protect sensitive information. In figure 1.3, if we add a field of total marks in every row then we would need different types of access control.

For example, a teacher would like to know marks of all students of a particular class hence needs to be given access to all the marks of that particular class. A student on the other hand should be able to see only his or her marks.

Data Backup and Recovery

A fallback mechanism for any business is a must. Having a database allows for establishing automated backup solutions to prevent data loss. The backups thus enable point-in-time recovery capabilities.

Assume that we had stored a physical copy of the student results for the past ten years, but the records got destroyed for some reason. Getting this data back would be almost impossible, but if the same records were available in the database, it would be very easy to regenerate the mark sheets.

Reporting and Analytics

Having a database enables a business to generate reports and dashboards for decision-making. It further facilitates complex data analysis and business intelligence operations that can be used to enhance the business capabilities.

Creating and printing mark sheets of hundreds of students would be very easy if the data is part of a database. Also performing analysis like student failure in a subject, best performing teacher and others would be very easy task.

It is possible to think of many more reasons why a database would be needed but for now the above reasons should suffice.

What is a Database Management System?

Database in simple sense is a repository. The repository can be created manually or using computers. A Database Management System (DBMS) is a software application that serves as a bridge between users and databases. It enables efficient creation, management, and manipulation of data using computers.

A DBMS functions as a central system for arranging data in a structured manner. It uses tables, relationships, and schema to maintain data integrity and consistency. The DBMS provides essential functionalities such as data storage, retrieval, modification, and deletion via standard query languages such SQL. It also simultaneously manages multiple user access and maintains security aspects. It manages complicated tasks such as transaction management, backups, and recovery. Additionally, it facilitates scalability by enabling databases to expand in size and handle more users.

Oracle, IBM DB2, Microsoft Access, and Microsoft SQL Server are well-known examples of proprietary DBMS. There are also several outstanding open source database management systems available, such as CouchDB, MariaDB, PostgreSQL, LibreOffice Base and MySQL. Each of these has a unique set of attributes and features that make them ideal for a wide range of uses.

In general, a DBMS is a vital component of most software application since it offers a strong, safe, and effective framework for handling the enormous amounts of data that businesses depend on to support their everyday operations and decision-making procedures.

Components of Database Management System

Most DBMSs provide four basic objects or components known as Tables, Queries, Forms and Reports to work with a required database. These objects become a building block for creating, using and managing any database. Let us have a brief look at each of these objects.

Table

The basic unit for storage in a DBMS is known as a table. A table is a two dimensional structure, it often consists of multiple rows and columns. The column is referred to as a 'Field' and the row referred to as a 'Record'. A table generally belongs to a specific entity like person, place, account, exam or others.

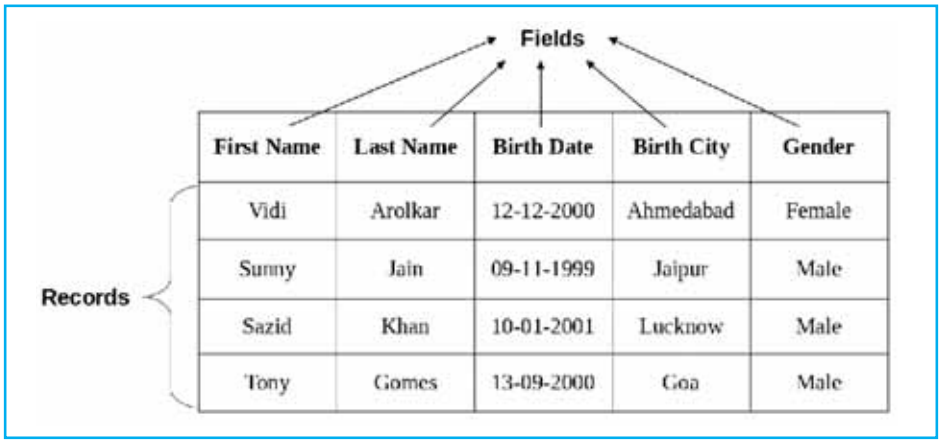


Figure 1.4 : Sample representation of a Table storing data of persons

Let us look at the terms entity, field and record from the perspective of figure 1.4.

Entity: Observe that we have stored the data pertaining to the person in the table. Here a person thus becomes an entity.

Field: It represents the discrete piece of data known as attribute, independently it may not have much of a significance. Here First Name, Last Name, Birth Date, Birth City and Gender are known as fields of the table.

Record: It is a collection of fields that shows details about a single instance of an entity. Observe that combined First Name, Last Name, Birth Date, Birth City and Gender represent a detail of one person.

Looking at the table given in figure 1.4, we can say that we have stored records of four unique persons and each person has five attributes.

Forms

We can insert data in the table as and when needed. Once inserted we may need to edit existing records, delete a record or view the records available within the table. Form provides us a graphical user interface that can be customized to one's personal requirement or liking. The forms may have a specific data entry format, design or layouts that ease the work of the user in case he wants to add, edit or delete a record.

Queries

The data in the table is used to answer the questions asked by the user. A question formed and asked in a database is known as Query. The queries can be created to answer questions like, How many

males are there in the persons table or What is the birth date of Sunny Jain. The query can be used to display a selected set of data contained in different tables within a database.

Reports

The output of a table or a query in a database is often displayed in the form of rows and columns. A user, though, expects more explainable and formatted output for the questions raised so that things can be inferred properly. The reports thus refer to proper presentation of the required information in a properly formatted, organized and readable format.

Macros

One additonal component of Base that is very helpful to users is a Macro. A macro is sequences of commands that automate repetitive tasks within the database. It can perform a series of actions with a single click, such as running a query, opening a report or updating records. We can create user-interface macros to control forms and reports or data macros to automate actions triggered by table events. Additionally macros can be used to implement checks and enforce rules, making data entry more reliable.

Sample Database Creation

Designing a database for any application requires a structured approach. We initially need to understand the requirements of an application, its processes, and the workflow that connects different activities. Based on these we need to identify key entities and their attributes that may be needed.

Let us try and create a structure for a relational data model that can be used to store data pertaining to school management activities. Although the school management includes a wide range of activities, such as managing students and teachers, examination management, payroll, and many more. We will look into only a small part of it where we will look into details of the student, teacher, subject, class and grade.

We have now decided about whom data will be stored? All these terms pertaining to whom data will be stored are known as entities. We will design a separate table for each entity. Thus in this case we will design five tables namely Student, Teacher, Subject, Class and Grade.

The next step is to decide what attributes each of these entities will have and what type of values are stored in them. These attributes will be represented as fields within the table. The sample schema of all these entities and their attributes is as shown in table 1.1.

Entity	Key Attributes
Student	StudentID, FirstName, MiddleName, LastName, BirthDate, Gender, Address, ClassID
Teacher	TeacherID, FirstName, MiddleName, LastName, BirthDate, Gender, Address, SubjectID
Subject	SubjectID, SubjectName, ClassID
Class	ClassID, ClassName, TeacherID
Grade	GradeID, StudentID, SubjectID, GradeDate, Marks

Table 1.1 : Sample Schema for School Database

This schema can be expanded to include any additional features as per the need of the school. For example we may add the details such as payroll, training, or recruitment.

Let us try and understand the details of attributes that will be stored in each of the tables as shown in table 1.2, 1.3, 1.4, 1.5 and 1.6.

STUDENT		
Attributes	**Description**	**Type of value to be stored**
StudentID	Used to store the ID of the student.	Can be a text or numeric value depending on organization policy.
FirstName	Used to store the first name of the student.	A text value.
MiddleName	Used to store the middle name of the student.	A text value
LastName	Used to store the last name of the student.	A text value.
BirthDate	Used to store the birth date of the student.	A date value.
Gender	Used to store the gender of the student.	A text value.
Address	Used to store the address of the student.	An alpha numeric value (May contain text, number as well as special characters).
ClassID	Used to store the ID of the class in which the student studies.	Will depend on the type of data stored in the Class table.

Table 1.2 : Details of Student Table

TEACHER		
Attributes	**Description**	**Type of value to be stored**
TeacherID	Used to store the ID of the teacher.	Can be a text or numeric value depending on organization policy.
FirstName	Used to store the first name of the teacher.	A text value.
MiddleName	Used to store the middle name of the teacher.	A text value
LastName	Used to store the last name of the teacher.	A text value.
BirthDate	Used to store the birth date of the teacher.	A date value.
Gender	Used to store the gender of the teacher.	A text value.
Address	Used to store the address of the teacher.	An alpha numeric value (May contain text, number as well as special characters).
SubjectID	Used to store the ID of the subject that the teacher teaches.	Will depend on the type of data stored in the Subject table.

Table 1.3 : Details of Teacher Table

SUBJECT		
Attributes	**Description**	**Type of value to be stored**
SubjectID	Used to store the ID of the subject.	Can be a text or numeric value depending on organization policy.
SubjectName	Used to store the name of the subject.	A text value.
ClassID	Used to store the ID of the class in which the subject is taught.	Will depend on the type of data stored in the Class table.

Table 1.4 : Details of Subject Table

CLASS		
Attributes	**Description**	**Type of value to be stored**
ClassID	Used to store the ID of the class.	Can be a text or numeric value depending on organization policy.
ClassName	Used to store the name of the class.	A text value.
TeacherID	Used to store the ID of the teacher who manages the class.	Will depend on the type of data stored in the teacher table.

Table 1.5 : Details of Class Table

GRADE		
Attributes	**Description**	**Type of value to be stored**
GradeID	Used to store the ID of records within the Grade table.	An integer number.
StudentID	Used to store the ID of the student who has been given the grade.	Will depend on the type of data stored in the student table.
SubjectID	Used to store the ID of the subject for which the student is graded.	Will depend on the type of data stored in the subject table.
GradeDate	Used to store the date when the student is given the grade.	A date value.
Marks	Used to store the marks.	A numeric value.

Table 1.6 : Details of Grade Table

We will revisit these tables in the next chapter when we create a database using LibreOffice Base. Let us talk about the concept of keys in a relational data model. As mentioned earlier, data in relational data model is arranged into tables (relationships) made up of rows and columns. The keys help us maintain these relationships properly. Four keys namely; primary, foreign, candidate and alternate play an important role.

Primary Key

A field that uniquely identifies a row in a table is called a primary key. The primary key can be made up of one or more columns with distinct values. The primary key cannot have the same value for multiple rows in the table, also the data value for that field cannot be left empty. For example, every student in the Student table has a distinct StudentID that can be considered as a primary key. If, in a table we use more than one field to identify a record, then this combined set of fields is known as a composite key.

Foreign Key

The foreign key is the field in one table that can be used to uniquely identify records in another table, either by itself or in combination with other fields. This foreign key facilitates the establishment of a relationship between two tables. For example, the ClassID field in the Student table acts as a foreign key, it tells us in which class the student studies. It also identifies the unique records in the class table. Thus we can say that the ClassID key relates the Student and Class tables.

Candidate Key

The candidate keys for a table are all the field values that are eligible to be the primary key. There must be no duplicate values in such fields, and they cannot be left empty. Therefore, in the Class table, both the ClassID and ClassName are potential candidate keys.

Alternate Key

One or two of the candidate keys are chosen as a primary key for a table. The candidate keys that are left out are known as alternate keys. Therefore, in the Class table the ClassName is the alternate key if the ClassID is used as the primary key.

Data and Data types

We have already seen in table 1.2 that the nature of data stored in different fields can vary. For example, in our data set the fields FirstName, LastName and Gender have text data while the field BirthDate is a date. We may also use alphanumeric data in case of a field named Address or a numeric data in a field named Age or a decimal number in a field named Percentage.

A data type thus refers to the type of data that will be stored in a specific field. The data type also decides the amount of memory that would be allocated to a particular field. It is possible that the memory size may vary within an individual data type also. Most DBMS support some common data types like text, numeric, currency, date/time, boolean and binary. In this section, we will discuss the data types that are supported by LibreOffice Base.

Text Data Type

The text data type allows us to store a combination of alphabets, numbers or special characters as a data value in a field. We cannot perform any arithmetic calculations on such data. Few examples of the fields that can use text data type are Aadhaar Card Number, First Name, Delivery Address and Email. There are four different text data types, and they differ mostly in how they consume space. Table 1.7 shows the list of different text data types that can be used in LibreOffice Base.

Type	Field Data Type in Base	Storage in memory
Text	VARCHAR	Variable
Text	VARCHAR_IGNORECASE	Variable
Text (fix)	CHAR	Fixed
Memo	LONGVARCHAR	Variable

Table 1.7 : Text Data Types

VARCHAR: The term here represents variable characters. It is possible to define the maximum number of characters that can be entered in a field when using VARCHAR. For example, if a field has data type VARCHAR(30) then we can enter a maximum of 30 characters as data in it. In case the user enters data which is less than 30 characters, only the needed space will be automatically allocated by the DBMS.

VARCHAR_IGNORECASE: It is similar to the VARCHAR, but is a case insensitive version of VARCHAR. The ignore case here plays a critical role at the time of searching. Assume that the user has stored some data "AHMEDABAD" in the field named city, now at the time of search if the user looks for "Ahmedabad" he will still get an output.

CHAR: This data type refers to a fixed size text field. The size of the field is set at the time of definition similar to the VARCHAR data type. If the text entered in the field does not occupy all the characters, the remaining characters are padded with spaces. This data type is best used when we expect a predefined number of characters like in a PAN or a credit card number.

LONGVARCHAR: This data type is designed to store very large blocks of text. It stores up to the maximum length of characters indicated by the user. It generally is used to store data having more than 255 characters. A good example of data for such a field is an article whose text may be up to 64,000 characters.

Numeric Data Type

This data type is used to store data of numerical form. The numeric data can be in the form of integer numbers or decimal numbers (floating point numbers). Both integer and decimal numbers can be signed or unsigned. It is possible to perform arithmetic operations on such data. Few examples of the fields that can use numeric data type are Marks, Salary, Interest and TotalPayment. LibreOffice Base has four integer data type which generally vary in size and four floating-point data types which vary in level of accuracy. Table 1.8 shows the list of different numeric data types along with the number of bits/bytes it uses and its range.

Type	Field Data Type in Base	Range of values	Bytes required in memory
Tiny Integer	TINYINT	0 to 255 if unsigned –128 to +127 if signed	1 Byte
Small Integer	SMALLINT	0 to 65535 if unsigned –32768 to +32767 if signed	2 Bytes
Integer	INTEGER	0 to 4294967295 –2147483648 to +2147483647	4 Bytes
BigInt	BIGINT	0 to 18446744073709551615	8 Bytes
Number	NUMERIC	Unlimited	Variable
Decimal	DECIMAL	Unlimited	Variable
Float	FLOAT	$5 \times 10^{(-234)}$ to $1.79 \times 10^{(308)}$	4 Bytes
Real	REAL	Not exact	4 Bytes
Double	DOUBLE	Adjustable with 15 decimal places maximum	8 Bytes

Table 1.8 : Numeric Data Types

TINYINT: It is the smallest of the integer data types. Its size is one byte hence it occupies 8 bits in memory.

SMALLINT: It is double the size of TINYINT. Its size is 2 bytes hence it occupies 16 bits in memory.

INTEGER or INT: The most commonly used numeric data type is INTEGER. Its size is 4 bytes hence it occupies 32 bits in memory.

BIGINT: Though available most of the simple programs do not require it, hence its rarely used. Its size is 8 bytes hence it occupies 64 bits in memory.

Floating-point numbers are numbers with decimals, or real numbers. They are made of a whole part and a partial part separated by a decimal point.

DECIMAL and NUMERIC: These data types have an unlimited range. While defining them we need to specify the total numbers of digits allowed along with the number of digits that will fall after the decimal point. For example, a definition DECIMAL(10, 2) for a field would mean that the field has a maximum of 10 places and two digits after the decimal point. The accuracy for DECIMAL and NUMERIC is nearly perfect.

DOUBLE or REAL: These data types have a limited range and can have a maximum of 15 decimal places. The accuracy of this data type is not so good.

Currency Data Type

The currency data type allows us to store numerical values that indicate the monetary values. We can store data using currencies of various countries. For example, ₹ 1250.50, $1500 or £ 700. This data type ensures that calculations involving currency are accurate and that the display of monetary values is formatted correctly.

Date/Time Data Type

The Date/Time data types are used to store information like year, month, day, hour, minute, second and fraction of a second. Few examples of the fields that can use date/time data type are Date of Joining, Birth Date, In Time and Out Time. LibreOffice Base has three types under it, they differ in the content they store. Table 1.9 shows the list of different date/time data types.

Type	Field Type in Base	Range of Values	Storage in memory
Date	DATE	–	4 Bytes
Time	TIME	–	4 Bytes
Date/Time	TIMESTAMP	Adjustable (0.5 – 5 with milliseconds)	8 Bytes

Table 1.9 : Date/Time Data Types

DATE: It stores the system date. The format for date entry is YYYY-MM-DD, i.e. 2025-05-30.

TIME: It stores a clock time. The default format for time is 24 hour clock, as HH:MM:SS, i.e. 15:30:36 for 3:30:36 PM.

TIMESTAMP or DATETIME: It is a combination of both the date and the time. The default format of data here is YYYY-MM-DD HH:MM:SS, i.e. 2025-05-30 15:30:36. This data type is used in the audit fields like Time of Transaction.

Boolean Data Type

The boolean data type allows us to store only two values, True or False. These two values can also be represented in multiple formats like Yes/No and On/Off.

Binary Data Type

The Binary data types allow us to store information like digitized images, videos, sounds or may be files. All these entities are stored as a long string of zeros and ones. Few examples of the fields that can use binary data type are Profile Picture and Voice Sample. Table 1.10 shows the list of different binary data types.

Type	Field Type in Base	Range of Values	Storage in memory
Binary field (fix)	BINARY	Similar to Integer	Fixed
Binary field	VARBINARY	Similar to Integer	Variable
Image	LONGVARBINARY	Similar to Integer	Variable used for very large images

Table 1.10 : Binary Data Types

Summary

In this chapter, we learned the difference between data and information. Data consists of raw facts, while information is the organized form of data. We further looked at data model in a database, it shows how data is stored and retrieved. Database management systems use various data models, including flat, hierarchical, and relational models. The relational model connects tables within a database. We also learned the concepts of database and DBMS, a database is an organized collection of data, and DBMSs help add, update, and access this data. We came to know that real world objects are called entities, and their specific details are known as attributes. The tables are representation of entities and contain records that are arranged in form of rows and columns, with the smallest unit being a field that represents an attribute. Data values stored in the fields can be numbers, letters or special characters. A record thus is a complete set of data values for a specific entity. A primary key uniquely identifies a row, and foreign keys connect different tables. Candidate keys are potential main keys for a table. This chapter has laid down a strong foundation for learning further concepts of database in the coming chapters.

EXERCISE

1. Explain the terms Data and Information with proper examples.

2. Differentiate between Database and Database Management systems.

3. Define the terms table, record and field.

4. Explain the need of a database.

5. Write a note on general components of a database.

6. What is Data type? List the data types available in Base.

7. What is the difference between CHAR and VARCHAR?

8. What is the use of currency data type?

9. When should one use boolean data type?

10. Identify any five attributes of an entity STUDENT.

11. State whether true or false.

(1) Database is a software that allows us to manipulate data.

(2) If we store data about a bus then it can be considered as an entity.

(3) Forms allow us to delete data from a table.

(4) It is possible to store a numeric value in Text data type.

(5) An audio clip can be stored in a Time data type.

12. Fill-in the blanks.

(1) Database is a ……... of related data items.

(2) The attributes of an entity are referred to as ……... in a table.

(3) A table consists of a set of ……... that give details on an entity.

(4) The BIGINT uses ……... bytes of memory storage.

(5) Digitized images can be stored in ……... data type.

13. Multi-choice questions. Choose the most correct answer.

(1) Which of the following is the full form of DBMS?

(a) Data Block Management System

(b) Database Management System

(c) Data Byte Management System

(d) Data Bit Management System

(2) Which of the following terms refers to processed data?

(a) Information (b) Raw (c) Facts (d) Field

(3) Which of the following is not a data model?

(a) Flat (b) Relational

(c) Rotational (d) Both (a) and (b)

(4) Which of the following best describes a StudentName in Database?

(a) Relationship (b) Attribute (c) Entity (d) Data

(5) Which of the following is not a database?

(a) MySQL (b) LibreOffice Base

(c) LibreOffice Impress (d) SQL Server

(6) Which of the following is not a general component of the database?

(a) Chart (b) Table (c) Queries (d) Form

(7) Which of the following data types is used to store an image in a Base database?

(a) Text (b) Binary (c) Boolean (d) Byte

(8) In which of the following forms, can a data not be represented?

(a) Text (b) Numeric

(c) Alphanumeric (d) Picture

(9) Which of the following is a feature that allows us to enter data in a table in an easy and user friendly manner.

(a) Report (b) Query (c) Form (d) Field

(10) Which of the following cannot be used to store numbers?

(a) TINYINT (b) DOUBLE (c) DATE (d) CHAR

Laboratory Exercise

1. Given the following entities, identify their attributes and decide what data types will be suitable to store the data.

   a. Book

   b. Customer

   c. Department

   d. Employee

   e. Flight

   f. Magazine

   g. Movie

   h. Product

   i. Salesman

   j. Train

   k. Vehicle