Database Management

1. Database :

A database can be defined as a collection of related data, information in organized manner so that it can easily be accessed, managed and updated.

In addition to the storage and retrieval of data, certain other operations can be performed on database. These operations includes adding data, deleting data and query from stored data etc. All this operation on a database are performed using a database management system.

2. Database Management System (DBMS):

DBMS is a software that allows access to the data contained in database. The objective of DBMS is to provide convenient and effective way of finding and retrieving information contained in database. The DBMS has several advantages that overcomes the shortcomings of traditional file system.

3. Advantages of DBMS:

There are following advantages in DBMS as stated below:

a) **Reduce Data Redundancy :** It refers to the duplication of data.

For Example: In case of school administration, the office of the school keeps a record of the student details in a file. A similar file is also kept with class teacher, exam dept. etc. The same data is duplicated in two separate places. This leads to redundancy.

However, A DBMS maintains a single copy of

data/information related to the students record that can be accessed by all the department associates with it.

b) **Control Data Inconsistency :** It refers to the data not manipulated consistently.
For Example: In the school, if student requests for a change in his postal address the changed addressed is reflected in the file maintained by the school office. However, it is not certain that the same change is made by the exam dept or by the class teacher. This leads to inconsistency. The same can be avoided in DBMS.

c) **Sharing of Data :** In a DBMS, unlike the file system approach, the data can be shared among various programs / applications. The sharing of data allows the existing application to use the existing data in the database. It also helps in developing new application which will use the same shared data.

d) **Enforcement of standards :** In a DBMS, unlike the file system approach, standards are easier to enforce as the access to the database is through the DBMS. Standards may relate to the naming of data, format of data, structure of data etc.

e) **Data Security :** Setting up of a DBMS makes it easier to enforce security restrictions since the data is stored centrally. A DBMS ensures that the only means of access to the database is through an

authorized channel. To ensure security, a DBMS provides security tools such as user codes and passwords.

f) **Data Integrity:** Data integrity refers to ensuring that the data in the database is accurate and consistent. The accuracy of database implies that correct data has been entered or updated.

Database Architecture

To know the database architecture it is required **to familiar with following terms:**

student ⟵ Table

Field / Record	Field 1	Field 2	Field 3	Field 4	Field 5
	admin_no	stu_name	gender	address	class
Record 1	AECS1200	PRATIK	M	MUMBAI	12A
Record 2	AECS1201	GARIMA	F	DELHI	12B
Record 3	AECS1202	KHUSHI	F	PUNA	12A
Record 4	AECS1203	BABU	M	TATA	12C
Record 5	AECS1207	AMIT	M	DELHI	12A
Record 6	AECS1225	SHARMILA	F	MUMBAI	12A

result12A

admin_no	Class	roll_no	marks
AECS1200	12A	1	80
AECS1202	12A	2	55
AECS1207	12A	4	90
AECS1225	12A	8	70

attendence12A

roll_no	attendence
1	98.0
2	82.5
3	35.2
4	55.6
8	78.4

.

Note: By using these tables student, result12A and attendence12A creates a new table **class12A** termed as **relation**.

class12A ⟵ relation

Attribute / Tuple	Attribute 1 admin_no	Attribute 2 stu_name	Attribute3 gender	Attribute 4 address	Attribute 5 class	Attribute 6 roll_no	Attribute 7 marks	Attribute 8 Attendance
Tuple 1	AECS1200	PRATIK	M	MUMBAI	12A	1	80	98.0
Tuple 2	AECS1202	KHUSHI	F	PUNA	12A	2	55	82.5
Tuple 3	AECS1207	AMIT	M	DELHI	12A	4	90	55.6
Tuple 4	AECS1225	SHARMILA	F	MUMBAI	12A	8	70	78.4

Table / Relation : A table or a relation is a collection of logically related records. In a table/ relation, the data is arranged in rows (records) or columns (fields).
Note : ***When more than one table are associate to each other to form a new table are termed as relation.***

Field : A field (column) holds one piece of data about an item of a database.
For Example : The table student in a database school maintaining information about a student, the fields can be stu_name, gender, address etc.

Record : A record(row) is a collection of multiple related fields that can be treated as a unit.
For Example : The table student in a database school, the collection of admin_no, stu_name, gender, address, class for a particular student forms a record of that student.

Attribute : The column of a relation is referred to as attribute. Each attribute holds a piece of information / data about an entity. The collection of attributes for a particular entity forms a tuple. The collection of tuples in turns forms a relation.

Tuple : The rows of a relation is referred to as tuple. Each tuple represents a record of that table and is uniquely identified by distinct value for an attribute.

Degree : The no. of attribute in a relation determines the degree of a relation. If a relation has 5 attributes it is said to be a relation of degree 5.

Cardinality: The number of tuples in a relation is called the cardinality of the relation. If a relation has 4 tuples it is said to be a relation of cardinality 4.

Table / Relation	Degree / No. of column/ attribute	Cardinality / No of rows /tuples
student	5	6
result12A	4	4
attendence12A	2	5
class12A	8	4

Domain : A domain is a set of permissible values of the sane type for a column (attribute). It is defined for every column of all the relations in a database. If a column is common in two relations, then the domain is also the same for both the relations.

For Examples: In a **school** database **student** relation/table, the column **admin_no** must consist of '**AECS**' followed by a four-digit number ranging from **1201 to 1299**. Hence, the domain of admin_no is **AECS1201** to **AECS1299**. In addition, the relation **result12A** also contains the attribute **admin_no**. Hence, this column also has the same domain.

Key: A key is an attribute that ensures that **NO** two rows of the relation are identical. In addition, it also associates two relations i.e establish a relationship with another table. There are four types of keys, namely

i) Primary Key
ii) Candidate Key
iii) Alternate Key
iv) Foreign Key

i) **Primary Key :** It denotes a key chosen by the database designer which uniquely identifies a record within a table. The primary key should be such that its value must not change. A table can have only one primary key and cannot accept a NULL value.
For Example : The attribute **admin_no** in the **student** relation / table can be designated as the primary key because all admin_no are unique and the value once entered is never changed.

ii) **Candidate key :** All the columns in a table that have unique value and can serve as primary key are called candidate key as they are the candidates for primary key position.
For Example : The attribute **roll_no** in the **results12A** relation / table can be designated as the candidate key.

iii) **Alternate key :** A candidate key that is not chosen as a primary key in a relation is known as an alternate key.
For Example : In the relation **results12A,** if the attribute **admin_no** is made the primary key. In such condition the attribute **roll_no** is the alternate key.

iv) **Foreign key :** The combination of a column in a table that reference a primary key in another table is known as foreign key.
For Example : Consider the table **student** and **result12A** made a relation by using primary key **admin_no** (*i.e attribute admin_no is present in both the tables*). In such condition **admin_no** is primary key for table **student** and foreign key for the table **result12A**.

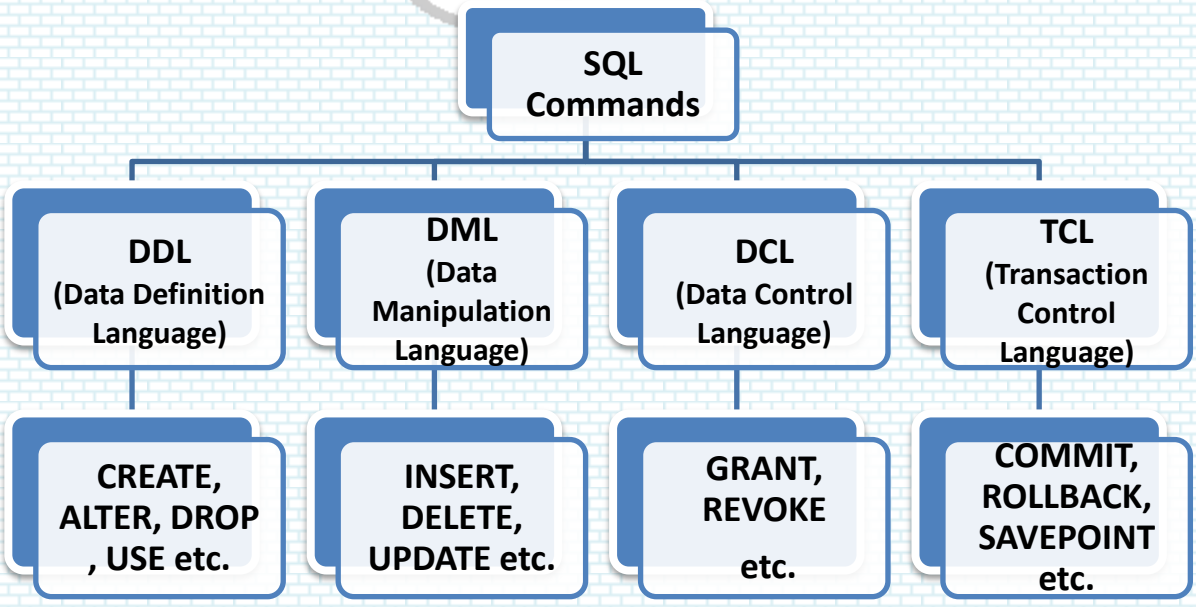
Structured Query Language (SQL)

The SQL is one such standard language which enables the user to retrieve the data without writing a complex program. The SQL is referred to as a declarative and non-procedural language. That is, the user has to specify what data to retrieve and how to retrieve it. Some common relational database management systems that use SQL are Oracle, Microsoft SQL, Sybase, Access, Ingres etc.

Some points must be kept in mind while using SQL commands.

- The SQL command are not case sensitive.
- The key word ate generally entered in upper case.
- The SQL commands can be written on more than one line.
- The keywords cannot be abbreviated or split across lines.
- A semicolon (;) is placed at the end of the last clause.

Classification of SQL commands : The SQL queries are basically the SQL commands that's a user to DBMS to interact with database. Based on that the SQL commands are divided into four categories mentioned below:



- **Data Definition Language (DDL) :** The DDL of SQL provides the commands for creating, alteration and deleting a database tables. It also defines indices(keys), specifies links between tables and imposes constraints on tables.
  Example of DDL commands in SQL are:
  - **CREATE DATABASE** – creates a new database
  - **USE command** – to select and open an already existing database.
  - **CREATE TABLE** – creates a new table.
  - **ALTER TABLE** – modifies a table.
  - **DROP TABLE** – deletes a table.

- **Data Manipulation Language (DML) :** The DML of SQL provides the commands to manipulate the data in the tables. DML commands are executed to retrieve data, insert data, delete data, update data, query operations from the tables.
  Example of DML commands in SQL are:
  - **SELECT statement** – To extract information from the table.
  - **INSERT INTO statement** – To insert new data (record) into a table.
  - **UPDATE statement** – To modify the data in a table. (*not modifying the data type of column*)
  - **DELETE** – To delete data (tuple) from a table. (*not deleting a column*)

- **Data Control Language (DCL) :** The DCL of SQL provides the commands to grant or revoke access rights to the database user. These commands are basically used to control the data stored in the database.
  (*not in the syllabus*)

Example of DCL commands in SQL are:

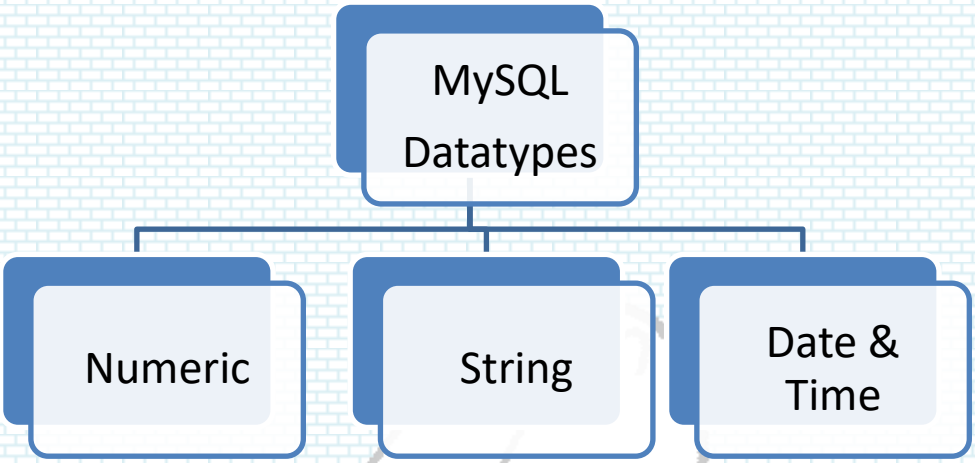
- **GRANT**
- **REVOKE**

➢ **Transaction Control Language (TCL) :** The TCL of SQL provides the commands to transactions in the database. These are used to manage the changes made to the data in a table by DML statements. It also allows statements to be grouped together into logical transactions.
(*not in the syllabus*)
Example of TCL commands in SQL are:

- **COMMIT**
- **ROLLBACK**
- **SAVEPOINT**

SQL Data types

SQL supports the following important data types for the specification of various data items or fields of a table/relation.



S.No.	Syntax	Description
1.	INTEGER or INT(x)	It is a 32-bit signed integer. It stores positive whole numbers up to 11 digit and negative numbers up to 10 digits. The range of integers is from -2147483648 to 2147483647. (i.e $\mathbf{2^{31}}$) '**x**' is the number of digit. Only INTEGER means INT(11) .
2.	SMALLINT	It is a 16-bit signed integer. It stores positive whole numbers up to 5 digit and negative numbers up to 4 digits. The range of integers is from -32768 to 32767. (i.e $\mathbf{2^{15}}$)

<table>
<tr><td>3.</td><td>NUMERIC(x,y)</td><td>It stores the number in the given format, where ‘x’ is the total number of digits (including decimal point and number after decimal point). ‘y’is the number of places to the right of the decimal point. If y is not defined, the number taken as integer type. It holds upto 20 significant digits.

For Example: NUMERIC(8,2)
<table>
<tr><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th></tr>
<tr><td>x</td><td>x</td><td>X</td><td>x</td><td>x</td><td>.</td><td>x</td><td>x</td></tr>
</table>
</td></tr>
<tr><td>4.</td><td>DECIMAL(x,y)</td><td>It stores the number in the given format, where ‘x’ is the precision (total size excluding decimal point and including number after decimal point) of digits and ‘y’ is the scale part i.e the number of places to the left of the decimal point. It holds up to 19 significant digits.

For Example: DECIMAL(7,2)
<table>
<tr><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th></tr>
<tr><td>x</td><td>x</td><td>X</td><td>x</td><td>x</td><td>.</td><td>x</td><td>x</td></tr>
</table>
</td></tr>
<tr><td>5.</td><td>FLOAT(x)</td><td>It stores the number in exponential notation of the base 10. The argument size ‘x’ specifies the minimum precision required.
(Precision means total digits without decimal).</td></tr>
</table>

6.	CHAR(x)	It stores 'x' number of characters in the string which has a fixed length. If you store strings that are not long as the size of 'x' parameter value, the remaining spaces are left unused but memory space is used. A maximum 254 characters can be stored in a string. **For Example:** CHAR(10)
7.	VARCHAR(x) or VARCHAR2(x)	It is used to store variable length alphanumeric data. The advantage of using this data type is that it releases the unused memory spaces. **For Example:** VARCHAR(10)
8.	DATE	It is used to store a date (year, month, day) in **'yyyy/mm/dd'** format. DATE values can be compared with each other only not with any other data type. The date values can be entered with **single quotation** marks or enclosed in **{ }.**
9.	TIME	It is used to store a time (hour, minute, second) in **hh:mm:ss** format.
10.	BOOLEAN	It is used to store a logical values, either **true** or **false.** In both upper case and lower case, **T** or **Y** (**t** or **y**) stands for logical true and **F** or **N** (**f** or **n**) stands for logical false. It can be compared only with other logical columns or constants.
11.	BLOB or	It is used to store data up to

	RAW or LONG RAW	maximum 65535 characters, BLOBs are 'Binary Large Object' can store images, animation, clips or any other types of files. **For Example:** RAW(2000);
12.	MEMO or LONG	It is used to store big contents of characters or remarks up to 2 GB per record.

Thank you