UNIT-I

DATABASE MANAGEMENT SYSTEM

What is Database

The database is a collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema, views, and reports, etc.

For example: The college Database organizes the data about the admin, staff, students and faculty etc.

Using the database, you can easily retrieve, insert, and delete the information.

Database Management System

- Database management system is a software which is used to manage the database. For example: MySQL, Oracle, etc are a very popular commercial database which is used in different applications.
- DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.
- It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

DBMS allows users the following tasks:

- **Data Definition:** It is used for creation, modification, and removal of definition that defines the organization of data in the database.
- **Data Updation:** It is used for the insertion, modification, and deletion of the actual data in the database.
- **Data Retrieval:** It is used to retrieve the data from the database which can be used by applications for various purposes.
- **User Administration:** It is used for registering and monitoring users, maintain data integrity, enforcing data security, dealing with concurrency control, monitoring performance and recovering information corrupted by unexpected failure.

Characteristics of DBMS

- It uses a digital repository established on a server to store and manage the information.
- It can provide a clear and logical view of the process that manipulates data.
- DBMS contains automatic backup and recovery procedures.
- It contains ACID properties which maintain data in a healthy state in case of failure.
- It can reduce the complex relationship between data.
- It is used to support manipulation and processing of data.

- It is used to provide security of data.
- It can view the database from different viewpoints according to the requirements of the user.

Advantages of DBMS

- **Controls database redundancy:** It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
- **Data sharing:** In DBMS, the authorized users of an organization can share the data among multiple users.
- **Easily Maintenance:** It can be easily maintainable due to the centralized nature of the database system.
- **Reduce time:** It reduces development time and maintenance need.
- **Backup:** It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
- **multiple user interface:** It provides different types of user interfaces like graphical user interfaces, application program interfaces

Disadvantages of DBMS

- **Cost of Hardware and Software:** It requires a high speed of data processor and large memory size to run DBMS software.
- **Size:** It occupies a large space of disks and large memory to run them efficiently.
- **Complexity:** Database system creates additional complexity and requirements.
- **Higher impact of failure:** Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

DBMS VS. FILE SYSTEM

There are following differences between DBMS and File system:

DBMS	File System
DBMS is a collection of data. In DBMS, the user is not required to write the procedures.	File system is a collection of data. In this system, the user has to write the procedures for managing the database.

DBMS gives an abstract view of data that hides the details.	File system provides the detail of the data representation and storage of data.
DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from the system failure.	File system doesn't have a crash mechanism, i.e., if the system crashes while entering some data, then the content of the file will lost.
DBMS provides a good protection mechanism.	It is very difficult to protect a file under the file system.
DBMS contains a wide variety of sophisticated techniques to store and retrieve the data.	File system can't efficiently store and retrieve the data.
DBMS takes care of Concurrent access of data using some form of locking.	In the File system, concurrent access has many problems like redirecting the file while other deleting some information or updating some information.

DATABASE USERS and ADMINISTRATORS (DBA)

Database users

Database users are the one who really use and take the benefits of database. There will be different types of users depending on their need and way of accessing the database.

1. **Unsophisticated users/Native Users/Naïve users -** these are the users who use the existing application to interact with the external database. For example, online library system, ticket booking systems, ATMs etc which has existing application and users use them to interact with the database to fulfill their requests.
2. **System analyst/application programmers:** system analysis determines the requirements of naïve users and develops specifications for querying and updating the database using standard type of queries and updates. application programmer

implements these specification to database as application packages to facilitate data access to end user or naïve user

3. **Sophisticated users**: They interact with the internal and external system .This include scientists, engineers who familiarize with the facilities of DBMS as to implement their applications to meet their complex requirements.

4. **Specialized users**: They are also sophisticated users who write specialized database applications that do not fit into the traditional data processing framework. Example: Expert System, Knowledge Based System, etc

5. **Stand-alone Users:** Those who are using database for personal usage. There are many database packages for this type database users

Database Administrators (DBA)

Database Administrators (DBA) coordinate all the activities of the database system. The person who has the central control over a database system is called Database Administrator (DBA). They have all the permissions. In order to perform his entire task, he should have very good command over DBMS.

Tasks of DBA

–Creating the schema
–Changing the schema
–Specifying integrity constraints
–Storage structure and access method definition
–Granting permission to other users.
–Monitoring performance

A DBA has many responsibilities. A good performing database is in the hands of DBA.

- **Installing and upgrading the DBMS Servers: -** DBA is responsible for installing a new DBMS server for the new projects. He is also responsible for upgrading these servers as there are new versions comes in the market or requirement. If there is any failure in upgradation of the existing servers, he should be able revert the new changes back to the older version, thus maintaining the DBMS working. He is also responsible for updating the service packs/ hot fixes/ patches to the DBMS servers.
- **Design and implementation: -** Designing the database and implementing is also DBA's responsibility. He should be able to decide proper memory management, file organizations, error handling, log maintenance etc for the database.
- **Performance tuning: -** Since database is huge and it will have lots of tables, data, constraints and indices, there will be variations in the performance from time to time. Also, because of some designing issues or data growth, the database will not work as expected. It is responsibility of the DBA to tune the database performance. He is responsible to make sure all the queries and programs works in fraction of seconds.

- **Backup and Recovery: -** Proper backup and recovery programs needs to be developed by DBA and has to be maintained him. This is one of the main responsibilities of DBA. Data/objects should be backed up regularly so that if there is any crash, it should be recovered without much effort and data loss.
- **Security: -** DBA is responsible for creating various database users and roles, and giving them different levels of access rights.
- **Documentation: -** DBA should be properly documenting all his activities so that if he quits or any new DBA comes in, he should be able to understand the database without any effort. He should basically maintain all his installation, backup, recovery, security methods. He should keep various reports about database performance.

There are different kinds of DBA depending on the responsibility that he owns.

- **Administrative DBA -** This DBA is mainly concerned with installing, and maintaining DBMS servers. His prime tasks are installing, backups, recovery, security, replications, memory management, configurations and tuning. He is mainly responsible for all administrative tasks of a database.
- **Development DBA -** He is responsible for creating queries and procedure for the requirement. Basically his task is similar to any database developer.
- **Database Architect -** Database architect is responsible for creating and maintaining the users, roles, access rights, tables, views, constraints and indexes. He is mainly responsible for designing the structure of the database depending on the requirement. These structures will be used by developers and development DBA to code.
- **Application DBA -**He acts like a bridge between the application program and the database. He makes sure all the application program is optimized to interact with the database. He ensures all the activities from installing, upgrading, and patching, maintaining, backup, recovery to executing the records works without any issues.

DBMS DATABASE MODELS

A Database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in a database management system. While the **Relational Model** is the most widely used database model, there are other models too:

- Hierarchical Model
- Network Model
- Entity-relationship Model
- Relational Model

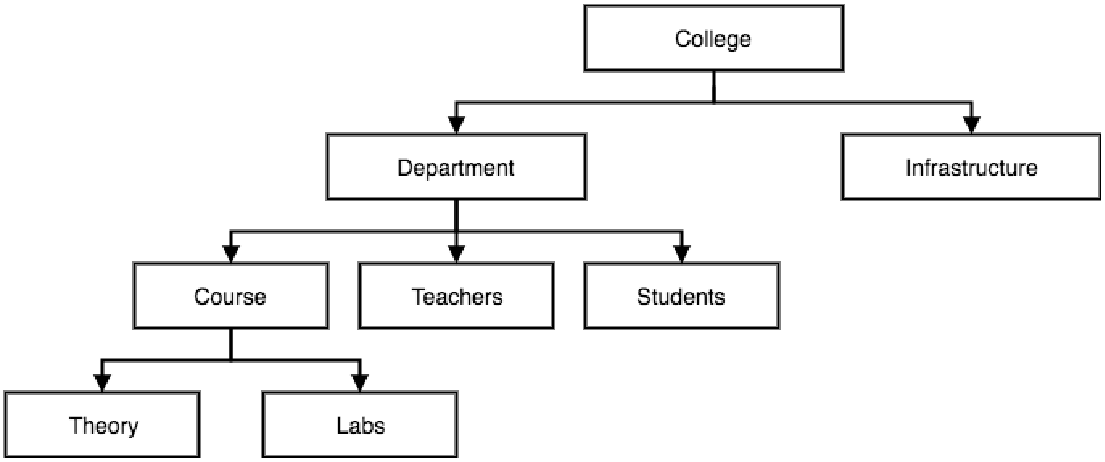
Hierarchical Model

This database model organizes data into a tree-like-structure, with a single root, to which all the other data is linked. The hierarchy starts from the **Root** data, and expands like a tree, adding child nodes to the parent nodes.

In this model, a child node will only have a single parent node.

This model efficiently describes many real-world relationships like index of a book, recipes etc.

In hierarchical model, data is organised into tree-like structure with one 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.

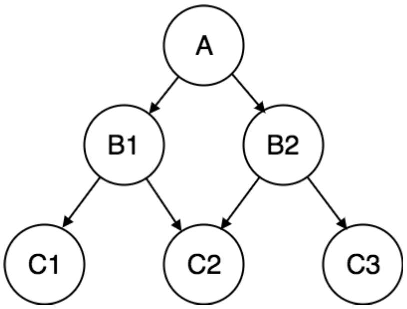


Network Model

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.



Entity-relationship Model

In this database model, relationships are created by dividing object of interest into entity and its characteristics into attributes.

Different entities are related using relationships.

E-R Models are defined to represent the relationships into pictorial form to make it easier for different stakeholders to understand.

- **Entity** − An entity in an ER Model is a real-world entity having properties called **attributes**. Every **attribute** is defined by its set of values called **domain**. For example, in a school database, a student is considered as an entity. Student has various attributes like name, age, class, etc.
- **Relationship** − The logical association among entities is called ***relationship***. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of association between two entities.

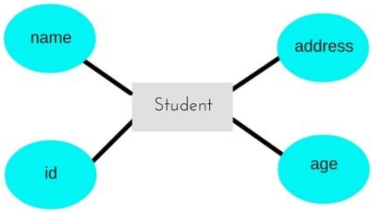
  Mapping cardinalities −

  - one to one
  - one to many
  - many to one
  - many to many

This model is good to design a database, which can then be turned into tables in relational model

Let's take an example, If we have to design a School Database, then **Student** will be an **entity** with **attributes** name, age, address etc. As **Address** is generally complex, it can be another **entity** with **attributes** street name, pincode, city etc, and there will be a relationship between them.

Relationships can also be of different types.



Relational Model

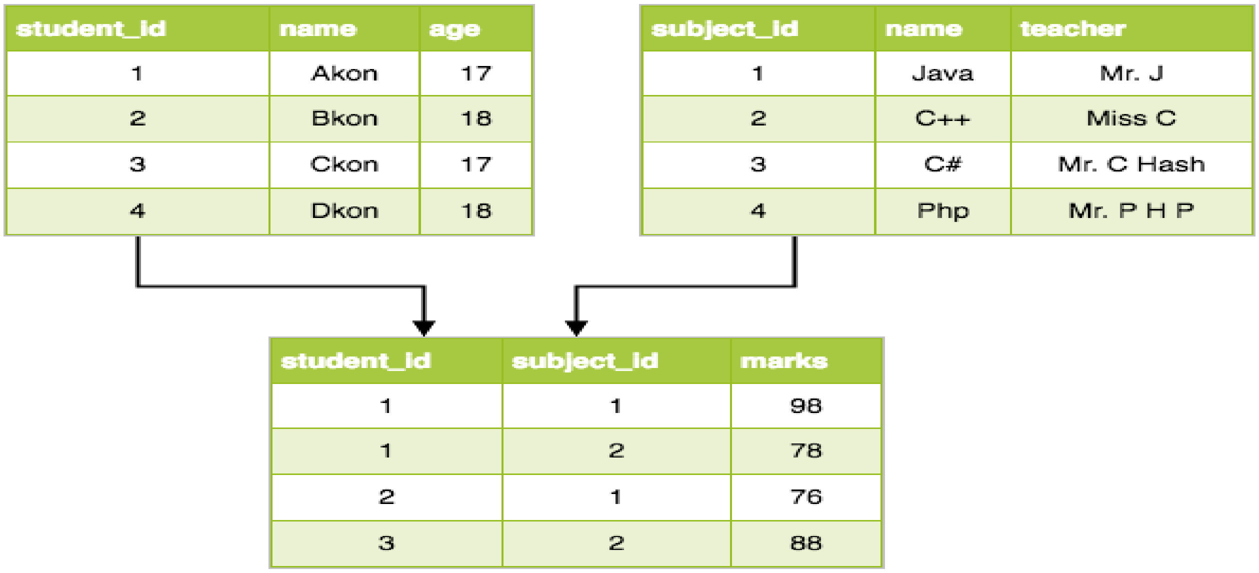
In this model, data is organized in two-dimensional **tables** and the relationship is maintained by storing a common field.

This model was introduced by E.F Codd in 1970, and since then it has been the most widely used database model, in fact, 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.

In the coming tutorials we will learn how to design tables, normalize them to reduce data redundancy and how to use Structured Query language to access data from tables.



student_id	name	age
1	Akon	17
2	Bkon	18
3	Ckon	17
4	Dkon	18

subject_id	name	teacher
1	Java	Mr. J
2	C++	Miss C
3	C#	Mr. C Hash
4	Php	Mr. P H P

student_id	subject_id	marks
1	1	98
1	2	78
2	1	76
3	2	88

DBMS SCHEMA AND DBMS INSTANCE

DBMS schema

Definition of schema: Design of a database is called the schema.

Schema is of three types

1. physical schema/ Internal schema
2. logical schema/ Conceptual schema
3. **3.** View schema/ External schema

The design of a database at physical level is called **physical schema**, how the data stored in blocks of storage is described at this level.

Design of database at logical level is called **logical schema**, programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures, however the internal details such as implementation of data structure is hidden at this level (available at physical level).

Design of database at view level is called **view schema**. This generally describes end user interaction with database systems.

DBMS Instance

Definition of instance: The data stored in database at a particular moment of time is called instance of database. Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

For example, let's say we have a single table student in the database, today the table has 100 records, and so today the instance of the database has 100 records. Let's say we are going to add another 100 records in this table by tomorrow so the instance of database tomorrow will have 200 records in table. In short, at a particular moment the data stored in database is called the instance that changes over time when we add or delete data from the database.

THREE LEVEL ARCHITECTURE OF DBMS

The three level architecture describes how data is viewed or represented by user in database. The three levels of DBMS are explained separately below:

A) External level

B) Conceptual level, and

C) Physical level

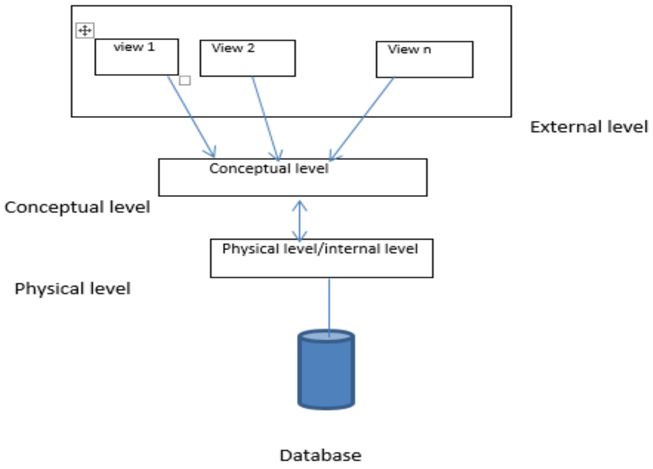


Fig. Three level architecture of dbms

External level/View level:

The external level is also called the view level.

In the external level there are the ‘n’ numbers of external end users who can view the data from the database shown above in the figure.

In this level the required data is provided to the end users because there is no need to know about the all data structure from the database. The users need only that data which is required.

The data can be viewed uniquely or separately by the users or viewers.

The external level is at the top level in architecture of dbms.

Conceptual /logical level:

The conceptual level is also called the logical level.

In this level it ensures what data can be stored and also express the relationship among the given data shown above in the figure.

It contains the constraints that are needed to apply on the stored data. The constraints may be the information related to security etc.

The conceptual level holds all the logical structure of entire database.

The entire database in this level is maintained by the DBA.

The conceptual level is at the middle level in architecture of dbms.

Internal/physical level:

The internal level is also named as physical level. The internal level express how data is stored in the files, indices .etc. on the storage devices i.e. secondary storage.

It also defines the data types of the data.

The most important task of the internal level is to allocate the space. So, the space is allocated for the data and indexes in this level.

This level is at the bottom level of architecture of dbms.

Although the internal level and the physical level is a single level but there is minimal difference between the physical level and internal level that is the physical level is maintained by the operating system under the guidance or instructions of the DBMS whereas the internal level is maintained by the DBMS directly.

Note:-All these three level architecture of dbms are interrelated to the database which is shown in above in the figure.

Schema

Schema is the overall view of database in dbms.in other words, schema is the plan design, scheme of database is known as schema. Schema is actually the frameset in which values of the data items are organized. At each level of architecture there is a separate schema.

Types of schema:

There are three types of schema that are explained individually below:

a. External schema
b. Conceptual schema, and
c. Internal schema

External schema:- External view of database is known as external schema. The external schema is linked with each end user of data.

Conceptual schema:- The conceptual view is defined by the conceptual schema which describes all the attributes, records, relationship etc.

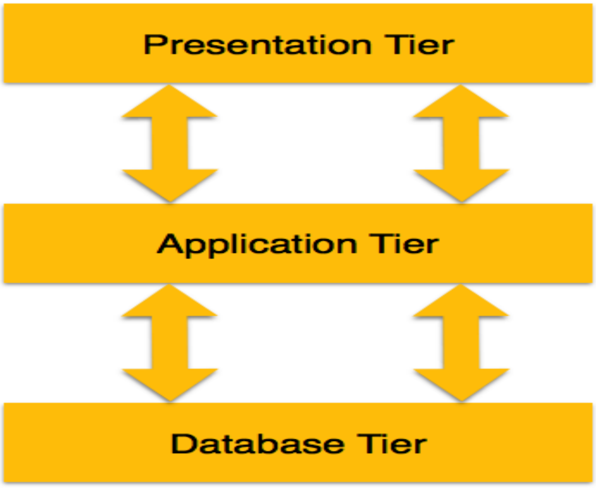
Internal schema: - Internal view is defined by the internal schema which describes how data will be stored physically in the database. It also described how data will be accessed using the resources that are provided by dbms.

DBMS – ARCHITECTURE

The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. An n-tier architecture divides the whole system into related but independent **n** modules, which can be independently modified, altered, changed, or replaced.

In 1-tier architecture, the DBMS is the only entity where the user directly sits on the DBMS and uses it. Any changes done here will directly be done on the DBMS itself. It does not provide handy tools for end-users. Database designers and programmers normally prefer to use single-tier architecture.

If the architecture of DBMS is 2-tier, then it must have an application through which the DBMS can be accessed. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.



3-tier Architecture

A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database. It is the most widely used architecture to design a DBMS.

- **Database (Data) Tier** − At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
- **Application (Middle) Tier** − At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
- **User (Presentation) Tier** − End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.

DATA INDEPENDENCE

A database system normally contains a lot of data in addition to users' data. For example, it stores data about data, known as metadata, to locate and retrieve data easily. It is rather difficult to modify or update a set of metadata once it is stored in the database. But as a DBMS expands, it needs to change over time to satisfy the requirements of the users. If the entire data is dependent, it would become a tedious and highly complex job.

Metadata itself follows a layered architecture, so that when we change data at one layer, it does not affect the data at another level. This data is independent but mapped to each other.

Logical Data Independence

Logical data is data about database, that is, it stores information about how data is managed inside. For example, a table (relation) stored in the database and all its constraints, applied on that relation.

Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk. If we do some changes on table format, it should not change the data residing on the disk.

Physical Data Independence

All the schemas are logical, and the actual data is stored in bit format on the disk. Physical data independence is the power to change the physical data without impacting the schema or logical data.

For example, in case we want to change or upgrade the storage system itself − suppose we want to replace hard-disks with SSD − it should not have any impact on the logical data or schemas.

DBMS LANGUAGES

Database languages are used to read, update and store data in a database. There are several such languages that can be used for this purpose; one of them is SQL (Structured Query Language).

Mainly has two parts:

- Data Definition Language (DDL) and
- Data Manipulation Language (DML).

The Data Definition Language is used for specifying the database schema, and the Data Manipulation Language is used for both reading and updating the database. These languages are called data sub-languages as they do not include constructs for all computational requirements.

Computation purposes include conditional or iterative statements that are supported by the high-level programming languages. Many DBMSs can embed the sublanguage in a high-level programming language such as 'Fortran,' 'C,' C++, Java, or Visual Basic. Here, the high-level language is sometimes referred to as the host language as it is acting as a host for this language. To compile the embedded file, the commands in the data sub-language are first detached from the host-language program and are substituted by function calls. The pre-processed file is then compiled and placed in an object module which gets linked with a DBMS-specific library that is having the replaced functions and executed based on requirement. Most data sub-languages also supply non-embedded or interactive commands which can be input directly using the terminal.

Data Definition Language

Data Definition Language (DDL) statements are used to classify the database structure or schema. It is a type of language that allows the DBA or user to depict and name those entities, attributes, and relationships that are required for the application along with any associated integrity and security constraints. Here are the lists of tasks that come under DDL:

- CREATE - used to create objects in the database
- ALTER - used to alters the structure of the database
- DROP - used to delete objects from the database
- TRUNCATE - used to remove all records from a table, including all spaces allocated for the records are removed
- COMMENT - used to add comments to the data dictionary
- RENAME - used to rename an object

Data Manipulation Language

A language that offers a set of operations to support the fundamental data manipulation operations on the data held in the database. Data Manipulation Language (DML) statements are used to manage data within schema objects. Here are the lists of tasks that come under DML:

- SELECT - It retrieves data from a database
- INSERT - It inserts data into a table
- UPDATE - It updates existing data within a table
- DELETE - It deletes all records from a table, the space for the records remain
- MERGE - UPSERT operation (insert or update)
- CALL - It calls a PL/SQL or Java subprogram
- EXPLAIN PLAN - It explains access path to data
- LOCK TABLE - It controls concurrency

Data Control Language

There are another two forms of database sub-languages. The Data Control Language (DCL) is used to control privilege in Database. To perform any operation in the database, such as for creating tables, sequences or views we need privileges. Privileges are of two types,

- System - creating a session, table, etc. are all types of system privilege.
- Object - any command or query to work on tables comes under object privilege. DCL is used to define two commands. These are:

- Grant - It gives user access privileges to a database.
- Revoke - It takes back permissions from the user.

Transaction Control Language (TCL)

Transaction Control statements are used to run the changes made by DML statements. It allows statements to be grouped into logical transactions.

- COMMIT - It saves the work done
- SAVEPOINT - It identifies a point in a transaction to which you can later roll back
- ROLLBACK - It restores the database to original since the last COMMIT
- SET TRANSACTION - It changes the transaction options like isolation level and what rollback segment to use