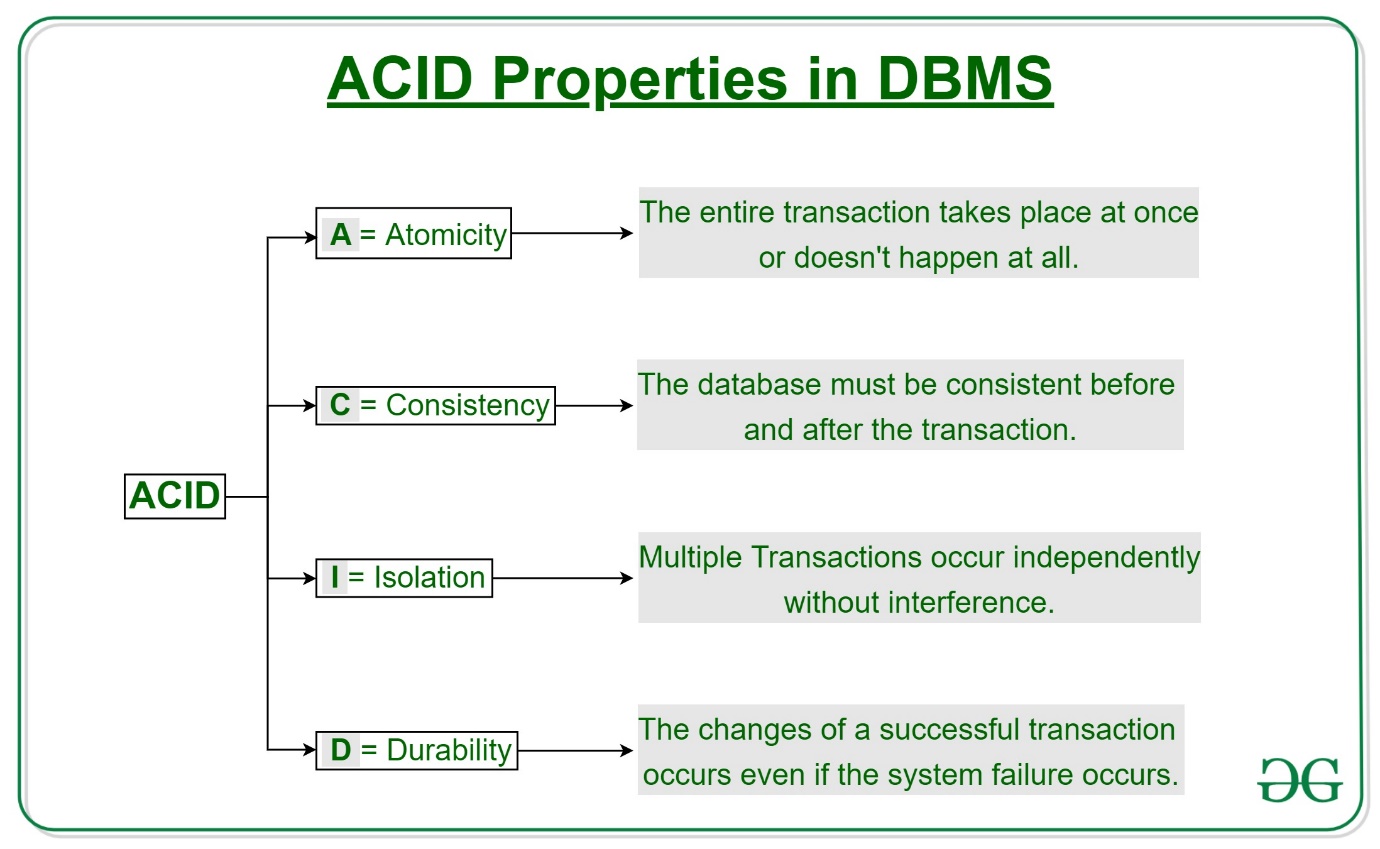
1..There are several advantages of Database management system over file system. Few of them are as follows:

* **No redundant data**: Redundancy removed by data [**normalization**](https://beginnersbook.com/2015/05/normalization-in-dbms/). No data duplication saves storage and improves access time.
* **Data Consistency and Integrity**: As we discussed earlier the root cause of data inconsistency is data redundancy, since data normalization takes care of the data redundancy, data inconsistency also been taken care of as part of it
* **Data Security**: It is easier to apply access constraints in database systems so that only authorized user is able to access the data. Each user has a different set of access thus data is secured from the issues such as identity theft, data leaks and misuse of data.
* **Privacy**: Limited access means privacy of data.
* **Easy access to data** – Database systems manages data in such a way so that the data is easily accessible with fast response times.
* **Easy recovery**: Since database systems keeps the backup of data, it is easier to do a full recovery of data in case of a failure.
* **Flexible**: Database systems are more flexible than file processing systems.

# 2. ACID Properties in DBMS

A [**transaction**](https://www.geeksforgeeks.org/sql-transactions/) is a single logical unit of work which accesses and possibly modifies the contents of a database. Transactions access data using read and write operations.   
In order to maintain consistency in a database, before and after the transaction, certain properties are followed. These are called **ACID** properties.



3. Explain the concept of normalization.

This concept document provides a brief discussion of the topic of data normalization as it applies to the development of the [Artifact: Data Model](http://home.iscte-iul.pt/~hro/RUPSmallProjects/core.base_rup/workproducts/rup_data_model_65B46980.html). It does not provide a full treatment of normalization, because the subject is quite broad and has been documented in many texts on database design. In [[NBG01](http://home.iscte-iul.pt/~hro/RUPSmallProjects/core.base_rup/customcategories/references_56F06DFD.html#NBG01)], normalization is defined as "an analytic technique used to produce a correct relational database design." In practice, normalization is a procedure for eliminating redundancy in the Data Model by means of applying restrictive rules. Elimination of data redundancy in the tables of the Data Model helps enforce referential integrity of the data in the database.

Normalization is usually performed on the Data Model after an initial version of the tables and their relationships has been developed in the model. The exact timing of when to apply normalization depends on the specific project situation and is up to the [Database Designer](http://home.iscte-iul.pt/~hro/RUPSmallProjects/core.base_rup/roles/rup_database_designer_524DC34F.html). The normalization process is applied to the tables in the Data Model in series of steps in which each step applies rules that are stricter than the last.

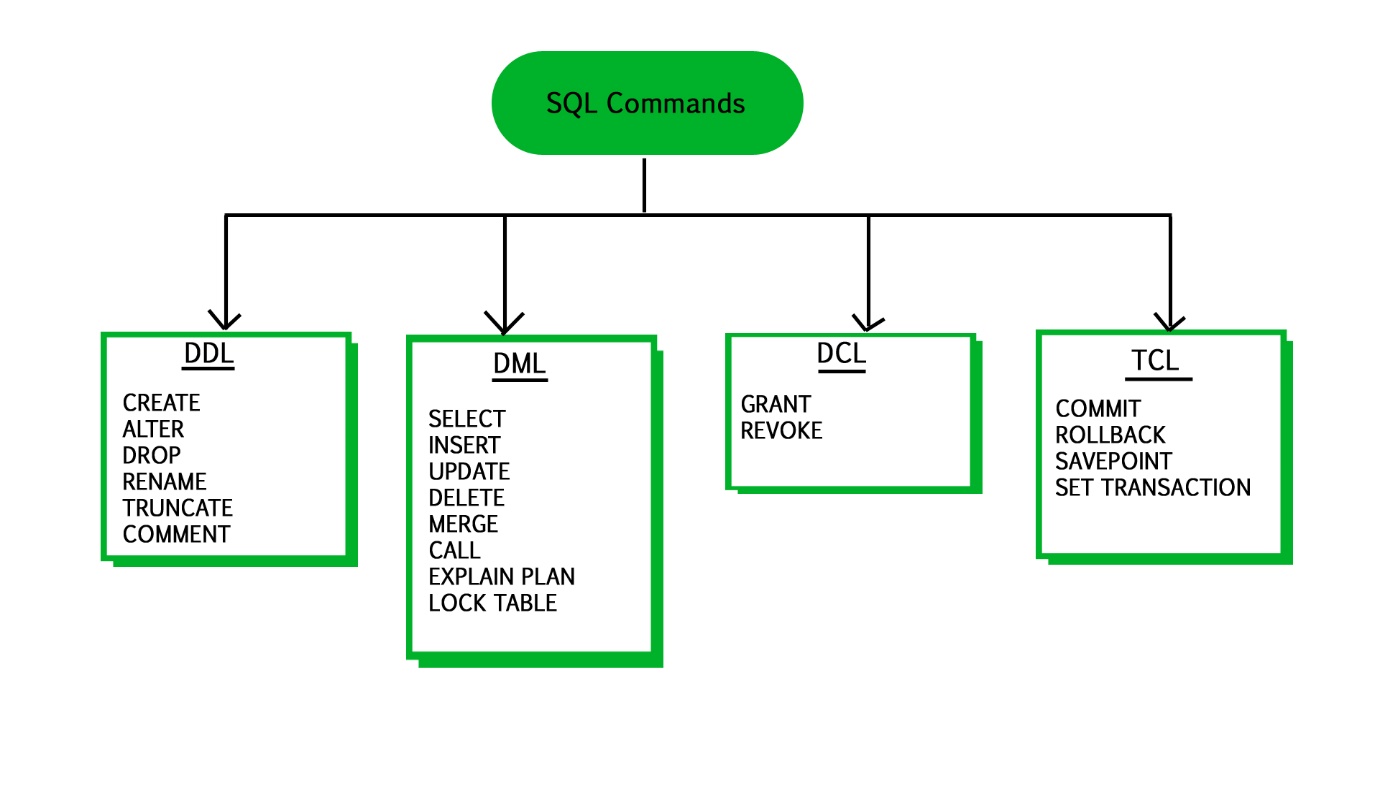
### Levels of Normalization

Normalization is hierarchically classified into numeric forms, with the most common being first, second, and third normal form. Each level of normalization is more restrictive than the previous. The first three hierarchical levels of normalization are:

* **First Normal Form**-Repeating groups of data columns in tables have been eliminated such that data is organized into atomic units.
* **Second Normal Form**-Data is in first normal form, and redundancy on primary key fields has been eliminated such that column values are wholly dependent on the primary key field.
* **Third Normal Form**-Data is in second normal form, and each column is not dependent on any other non-key column.

Other levels of normalization are possible but are not covered in this discussion.  Information on additional levels of normalization can be found in [[DAT99](http://home.iscte-iul.pt/~hro/RUPSmallProjects/core.base_rup/customcategories/references_56F06DFD.html#DAT99)]. The exact level of normalization to apply to the Data Model is a decision that the database designer must make based on the specifics of a situation.

# 4. SQL | DDL, DML, TCL and DCL

In this article, we’ll be discussing Data Definition Language, Data Manipulation Language, Transaction Control Language, and Data Control Language.  


#### DDL (Data Definition Language) :

Data Definition Language is used to define the database structure or schema. DDL is also used to specify additional properties of the data. The storage structure and access methods used by the database system by a set of statements in a special type of DDL called a data storage and definition language. These statements define the implementation details of the database schema, which are usually hidden from the users. The data values stored in the database must satisfy certain consistency constraints.  
For example, suppose the university requires that the account balance of a department must never be negative. The DDL provides facilities to specify such constraints. The database system checks these constraints every time the database is updated. In general, a constraint can be an arbitrary predicate pertaining to the database. However, arbitrary predicates may be costly to the test. Thus, the database system implements integrity constraints that can be tested with minimal overhead.

1. **Domain Constraints :** A domain of possible values must be associated with every attribute (for example, integer types, character types, date/time types). Declaring an attribute to be of a particular domain acts as the constraints on the values that it can take.
2. **Referential Integrity :** There are cases where we wish to ensure that a value appears in one relation for a given set of attributes also appear in a certain set of attributes in another relation i.e. Referential Integrity. For example, the department listed for each course must be one that actually exists.
3. **Assertions :** An assertion is any condition that the database must always satisfy. Domain constraints and Integrity constraints are special form of assertions.
4. **Authorization :** We may want to differentiate among the users as far as the type of access they are permitted on various data values in database. These differentiation are expressed in terms of Authorization. The most common being :  
   read authorization – which allows reading but not modification of data ;  
   insert authorization – which allow insertion of new data but not modification of existing data  
   update authorization – which allows modification, but not deletion.

**Some Commands:**

CREATE : to create objects in database

ALTER : alters the structure of database

DROP : delete objects from database

RENAME : rename an objects

Following SQL DDL-statement defines the department table :

create table department

(dept\_name char(20),

building char(15),

budget numeric(12,2));

Execution of the above DDL statement creates the department table with three columns – dept\_name, building, and budget; each of which has a specific datatype associated with it.

#### DML (Data Manipulation Language) :

DML statements are used for managing data with in schema objects.  
DML are of two types –

1. **Procedural DMLs** : require a user to specify what data are needed and how to get those data.
2. **Declerative DMLs** (also referred as **Non-procedural DMLs**) : require a user to specify what data are needed without specifying how to get those data.

Declarative DMLs are usually easier to learn and use than procedural DMLs. However, since a user does not have to specify how to get the data, the database system has to figure out an efficient means of accessing data.

**Some Commands :**

SELECT: retrieve data from the database

UPDATE: update existing data within a table

DELETE: deletes all records from a table, space for the records remain

Example of SQL query that finds the names of all instructors in the History department :

select instructor.name

from instructor

where instructor.dept\_name = 'History';

The query specifies that those rows from the table instructor where the dept\_name is History must be retrieved and the name attributes of these rows must be displayed.

#### TCL (Transaction Control Language) :

Transaction Control Language commands are used to manage transactions in the database. These are used to manage the changes made by DML-statements. It also allows statements to be grouped together into logical transactions.

Examples of TCL commands –

COMMIT: Commit command is used to permanently save any transaction

into the database.

ROLLBACK: This command restores the database to last committed state.

It is also used with savepoint command to jump to a savepoint

in a transaction.

SAVEPOINT: Savepoint command is used to temporarily save a transaction so

that you can rollback to that point whenever necessary.

#### DCL (Data Control Language) :

A Data Control Language is a syntax similar to a computer programming language used to control access to data stored in a database (Authorization). In particular, it is a component of Structured Query Language (SQL).

Examples of DCL commands :

GRANT: allow specified users to perform specified tasks.

REVOKE: cancel previously granted or denied permissions.

The operations for which privileges may be granted to or revoked from a user or role apply to both the Data definition language (DDL) and the Data manipulation language (DML), and may include CONNECT, SELECT, INSERT, UPDATE, DELETE, EXECUTE and USAGE.

In the Oracle database, executing a DCL command issues an implicit commit. Hence, you cannot roll back the command.

## 5. **Primary Key**

First, a primary key uniquely identifies each record in a database table. Any individual key that does this can be called a candidate key, but only one can be chosen by database engineers as a primary key.

## **Composite Key**

Next, there's the composite key, which is composed of two or more attributes that collectively uniquely identify each record.

An example would be a list of homes on a real estate market. In a well-ordered database, there should be a primary key that uniquely identifies each record.

# 6. Constraints in SQL Server: SQL NOT NULL, UNIQUE and SQL PRIMARY KEY

This article explains the SQL NOT NULL, Unique and SQL Primary Key constraints in SQL Server with examples.

[Constraints in SQL Server](https://www.quest.com/community/blogs/b/database-management/posts/exploring-the-different-constraints-in-sql-server) are predefined rules and restrictions that are enforced in a single column or multiple columns, regarding the values allowed in the columns, to maintain the integrity, accuracy, and reliability of that column’s data. In other words, if the inserted data meets the constraint rule, it will be inserted successfully. If the inserted data violates the defined constraint, the insert operation will be aborted.

Constraints in SQL Server can be defined at the column level, where it is specified as part of the column definition and will be applied to that column only, or declared independently at the table level. In this case, the constraint rules will be applied to more than one column in the specified table. The constraint can be created within the CREATE TABLE T-SQL command while creating the table or added using ALTER TABLE T-SQL command after creating the table. Adding the constraint after creating the table, the existing data will be checked for the constraint rule before creating that constraint. There are six main constraints that are commonly used in SQL Server that we will describe deeply with examples within this article and the next one. These constraints are:

* SQL NOT NULL
* UNIQUE
* PRIMARY KEY
* FOREIGN KEY
* CHECK
* DEFAULT

In this article, we will go through the first three constraints; SQL NOT NULL, UNIQUE and SQL PRIMARY KEY, and we will complete the rest three constraints in the next article. Let us start discussing each one of these SQL Server constraints with a brief description and practical demo

## **NOT NULL Constraint in SQL**

By default, the columns are able to hold NULL values. A NOT NULL constraint in SQL is used to prevent inserting NULL values into the specified column, considering it as a not accepted value for that column. This means that you should provide a valid SQL NOT NULL value to that column in the INSERT or UPDATE statements, as the column will always contain data.

Assume that we have the below simple CREATE TABLE statement that is used to define the ConstraintDemo1 table. This table contains only two columns, ID and Name. In the ID column definition statement, the SQL NOT NULL column-level constraint is enforced, considering the ID column as a mandatory column that should be provided with a valid SQL NOT NULL value. The case is different for the Name column that can be ignored in the INSERT statement, with the ability to provide it with NULL value. If the null-ability is not specified while defining the column, it will accept the NULL value by default:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | USE SQLShackDemo  GO  CREATE TABLE ConstraintDemo1  (         ID INT NOT NULL,     Name VARCHAR(50) NULL  ) |

THE END