**What is Database ?**

Database Management System (DBMS) is a software for storing and retrieving users' data while considering appropriate security measures. It consists of a group of programs which manipulate the database. The DBMS accepts the request for data from an application and instructs the operating system to provide the specific data. In large systems, a DBMS helps users and other third-party software to store and retrieve data.

DBMS allows users to create their own databases as per their requirement. The term “DBMS” includes the user of the database and other application programs. It provides an interfa ce between the data and the software application.

**Characteristics of Database Management System**

• Provides security and removes redundancy

• Self-describing nature of a database system

• Insulation between programs and data abstraction

• Support of multiple views of the data

• Sharing of data and multiuser transaction processing

• DBMS allows entities and relations among them to form tables.

• It follows the ACID concept ( Atomicity, Consistency, Isolation, and Durability).

• DBMS supports multi-user environment that allows users to access and manipulate data in parallel.

**Disadvantage of DBMS**

DBMS may offer plenty of advantages but, it has certain flaws-

• Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization.

• Most database management systems are often complex systems, so the training for users to use the DBMS is required.

• In some organizations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media

• Use of the same program at a time by many users sometimes lead to the loss of some data.

• DBMS can't perform sophisticated calculations

**Application of DBMS**

|  |  |
| --- | --- |
| **Sector** | **Use of DBMS** |
| Banking | For customer information, account activities, payments, deposits, loans, etc. |
| Airlines | For reservations and schedule information. |
| Universities | For student information, course registrations, colleges and grades. |
| Telecommunication | It helps to keep call records, monthly bills, maintaining balances, etc. |
| Finance | For storing information about stock, sales, and purchases of financial instruments like stocks and bonds. |
| Sales | Use for storing customer, product & sales information. |
| Manufacturing | It is used for the management of supply chain and for tracking production of items. Inventories status in warehouses. |
| HR Management | For information about employees, salaries, payroll, deduction, generation of paychecks, etc. |

**What is SQL?**

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

• SQL stands for Structured Query Language.

• It is designed for managing data in a relational database management system (RDBMS).

• It is pronounced as S-Q-L or sometime See-Qwell.

• SQL is a database language, it is used for database creation, deletion, fetching rows, and modifying rows, etc.

• SQL is based on relational algebra and tuple relational calculus.

**What Can SQL do?**

• SQL can execute queries against a database

• SQL can retrieve data from a database

• SQL can insert records in a database

• SQL can update records in a database

• SQL can delete records from a database

• SQL can create new databases

• SQL can create new tables in a database

• SQL can create stored procedures in a database

• SQL can create views in a database

• SQL can set permissions on tables, procedures, and views

**Advantages of DBMS**

• DBMS offers a variety of techniques to store & retrieve data

• DBMS serves as an efficient handler to balance the needs of multiple applications using the same data

• Uniform administration procedures for data

• Application programmers never exposed to details of data representation and storage.

• A DBMS uses various powerful functions to store and retrieve data efficiently.

• Offers Data Integrity and Security

• The DBMS implies integrity constraints to get a high level of protection against prohibited access to data.

• A DBMS schedules concurrent access to the data in such a manner that only one user can access the same data at a time

• Reduced Application Development Time

**Disadvantage of DBMS**

DBMS may offer plenty of advantages but, it has certain flaws-

• Cost of Hardware and Software of a DBMS is quite high which increases the budget of your organization.

• Most database management systems are often complex systems, so the training for users to use the DBMS is required.

• In some organizations, all data is integrated into a single database which can be damaged because of electric failure or database is corrupted on the storage media

• Use of the same program at a time by many users sometimes lead to the loss of some data.

• DBMS can't perform sophisticated calculations

**Basic datatypes**

**char(n):-** A fixed-lengh character string with user –specified length n.

The full form , **characte**r, can be used instead.

**varchar(n):-** A variable-length character string with user-specified maximum length n. The full form, **character varying**, is Equivalent.

**int:** An integer( a finite subset of the integers that is machine dependent). The full form, **integer**, is equivalent.

**smallint:-** A small integer( a machine-dependent subset of ineger domain type).

**numeric(p,d):-** A fixed- point number with user-specified precision. The Number consists of p digit(plus a sign), and d of the p digits are to the right of the decimal point. Thus, numeric(3,1) allows 44.5 to be stored exactly, but neither 444.5 or 0.32 can be stored exactly in a field of this type.

number consists of p digits(Plus a sign), and d digits are to the right of the decimal point. Thus, numeric(3,1) allows 44.5 to be stored exactly, buin a field of this type.

**real,doble precision:-** Floating-point and double-precision floating-point

numbers machine-dependent precision.

**float(n):-** A floating-point number, with precision of at least n digits.

SQL data types are divided into five categories:

1. Character/String Data Types
2. Numeric Data Types
3. Date/Time Data Types
4. Binary Data Types
5. Miscellaneous Data Types

**SQL**

**Data types**

Char/string

Numeric

Date/Time

Binary

Miscellaneous

**Char, Varchar,Nchar,Narchar,Text,Ntext**

**Bit,smallint,int,bigint,decimal,real,numeric,float**

**Datetime, Date,Time, Timestamp,year**

**Binary,Varbinary,**

**Varbinary(max) Image**

**Clob,Blob,**

**Json,XML**

**1. Character/String Data Types**

1. Character/Strings can be used to store strings like names, locations, etc. and even large text files.

|  |  |
| --- | --- |
| 2. Data Type | 1. Description |
| Char | Maximum 8000 character long fixed length string |
| VARCHAR | Maximum 8000 character long variable length string |
| VARCHAR(max) | Maximum max character long variable length string |
| NCHAR | Unicode Maximum 8000 character long fixed length string |
| NVARCHAR | Unicode Maximum 8000 character long variable length string |
| NVARCHAR(max) | Unicode Maximum max character long variable length string |
| TEXT | Maximum 2GB sized variable length string |

**2. Numeric Data Types**

Numeric data types can be used to store exact or decimal numeric data, from very small numbers in a bit or ***tinyint*** to very large numbers in a ***bigint.***

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Minimum** | **Maximum** |
| Bit | 0 | 1 |
| tinyint | 0 | 25 5 |
| smallint | -32,768 | 32,767 |
| int | -2,147,483,648 | 2,147,483,647 |
| bigint | -9,223,372,036,854,775,808 | 9,223,372,036,854,775,807 |
| Decimal | -10^38 - 1 | -10^38 - 1 |
| Real | -3.40E + 38 | 3.40E + 38 |
| numeric | -10^38 +1 | 10^38 -1 |
| float | -1.79E + 308 | 1.79E + 308 |

**3. Date/Time Data Types**

Date/Time data types are used to store data and time objects. Storing a date as a date/type object instead of a string can make it very easy to work with data at a later point.

|  |  |
| --- | --- |
| **Data Type** | **Description** |
| Datetime | Date and Time stored in the YYYY-MM-DD HH:MI:SS format |
| Date | Date stored in the YYYY-MM-DD format |
| Time | Time stored in the HH:MI:SS format |
| Timestamp | Number of seconds passed stored since ‘1970-01-01 00:00:00’ UTC |
| Year | Year stored in 2-digit (from 1970-2069) or 4-digit (from 1901 2155) format |

**4. Binary Data Types**

Binary data types are used to store data in its binary form, instead of storing it as strings, which can be a big factor in saving space.

|  |  |
| --- | --- |
| Data Type | Description |
| BINARY | Maximum length of 8000 bytes fixed binary |
| VARBINARY | Maximum length of 8000 bytes variable binary |
| VARBINARY(max) | Maximum length of max bytes fixed binary |
| NTEXT | Maximum 2GB sized variable length binary |

**5. Miscellaneous Data Types**

These data types are for special types of data as described below:

|  |  |
| --- | --- |
| Data Type | Description |
| CLOB | Upto 2GB large character objects |
| BLOB | For storing binary large objects |
| JSON | For storing JSON objects |
| XML | For storing XML objects |

**SQL COMMANDS**

1. **CREATE**

This statement is used to create a table or a database.

The ‘CREATE DATABASE’ Statement

As the name suggests, this statement is used to create a database.

Syntax

CREATE DATABASE DatabaseName;

Example

CREATE DATABASE College;

The ‘CREATE TABLE’ Statement

This statement is used to create a table.

Syntax

*CREATE TABLE TableName (*

*Column1 datatype,*

*Column2 datatype,*

*Column3 datatype,*

*....*

*ColumnN datatype*

*);*

Ex-

*create table student*

*(Rollno varchar(5) primary key,*

*Name varchar(30) not null,*

*DOB date ,*

*address varchar(50) not null,*

*City varchar(20),*

*Mob varchar(11) unique key ,*

*Fee Double(8,2));*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** |

1. **Insert(insert data into table):-**

Once a table create most natural thing is to load this table with data to be manipulation data.

**Syntax**

*INSERT INTO TableName (Column1, Column2, Column3, ...,ColumnN)*

*VALUES (value1, value2, value3, ...);*

Ex-

*insert into student(rollno,Name,Dob,address,City,Mob,Fee)values('S0101','Raj Sharma' ,'1999/05/09','Suhag Nagar Firozabad','Firozabad',8273555968,45236.15);*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **RollNo** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** |
| S0101 | Raj Gupta | 1999-05-09 | Suhag Nagar Firozabad | Firozabad | 8273555968 | 45236.15 |

3.**SELECT(Viewing Data):-** The SQL SELECT statement is used to fetch the data from a database table which returns this data in the form of a result table.

**Syntax**

The basic syntax of the SELECT statement is as follows −

*SELECT column1, column2, columnN FROM table\_name;*

Here, column1, column2... are the fields of a table whose values you want to fetch. If you want to fetch all the fields available in the field, then you can use the following syntax.

*SELECT \* FROM table\_name;*

Ex-

To view Specific fields of Table-

*SELECT Rollno,Name from student;*

|  |  |
| --- | --- |
| **Rollno** | **Name** |
| S0101 | Raj Sharma |

To View all row of table-

*SELECT \* from Student;*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **RollNo** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** |
| S0101 | Raj Gupta | 1999-05-09 | Suhag Nagar Firozabad | Firozabad | 8273555968 | 45236.15 |

4.**ALTER TABLE(Modify the structure of table)**:-

The SQL ALTER TABLE command is used to add, delete or modify columns in an existing table. You should also use the ALTER TABLE command to add and drop various constraints on an existing table.

**Syntax-**

The basic syntax of an ALTER TABLE command to add a New Column in an existing table is as follows.

*ALTER TABLE table\_name ADD column\_name datatype;*

The basic syntax of an ALTER TABLE command to DROP COLUMN in an existing table is as follows.

*ALTER TABLE table\_name DROP COLUMN column\_name;*

The basic syntax of an ALTER TABLE command to change the DATA TYPE of a column in a table is as follows.

*ALTER TABLE table\_name MODIFY COLUMN column\_name datatype;*

The basic syntax of an ALTER TABLE command to add a NOT NULL constraint to a column in a table is as follows.

*ALTER TABLE table\_name MODIFY column\_name datatype NOT NULL;*

(V) The basic syntax of ALTER TABLE to ADD UNIQUE CONSTRAINT to a table is as follows.

*ALTER TABLE table\_name*

*ADD CONSTRAINT MyUniqueConstraint UNIQUE(column1, column2...);*

The basic syntax of an ALTER TABLE command to ADD CHECK CONSTRAINT to a table is as follows.

*ALTER TABLE table\_name*

*ADD CONSTRAINT MyUniqueConstraint CHECK (CONDITION);*

The basic syntax of an ALTER TABLE command to ADD PRIMARY KEY constraint to a table is as follows.

*ALTER TABLE table\_name*

*ADD CONSTRAINT MyPrimaryKey PRIMARY KEY (column1, column2...);*

The basic syntax of an ALTER TABLE command to DROP CONSTRAINT from a table is as follows.

*ALTER TABLE table\_name*

*DROP CONSTRAINT MyUniqueConstraint;*

Ex-

Add a new Column in a table-

*alter table student add Email Varchar(50) unique key;*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** | **Email** |
| S0101 | Raj Sharma | 1999-05-09 | Suhag Nagar Firozabad | Firozabad | 8273555968 | 45236.15 | Null |

1. **UPDATE:-**

The SQL UPDATE Query is used to modify the existing records in a table. You can use the WHERE clause with the UPDATE query to update the selected rows, otherwise all the rows would be affected.

Syntax

The basic syntax of the UPDATE query with a WHERE clause is as follows −

*UPDATE table\_name*

*SET column1 = value1, column2 = value2...., columnN = valueN*

*WHERE [condition];*

(You can combine N number of conditions using the AND or the OR operators.)

Ex-

*update student set email='rajsharma234@gmail.com'where rollno='s0101';*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** | **Email** |
| S0101 | Raj Sharma | 1999-05-09 | Suhag Nagar Firozabad | Firozabad | 8273555968 | 45236.15 | Rajsharma234@gmail.com |

**Delete:-** The SQL DELETE Query is used to delete the existing records from a table.

You can use the WHERE clause with a DELETE query to delete the selected rows, otherwise all the records would be deleted.

**Syntex-**

The basic syntax of the DELETE query with the WHERE clause is as follows −

*DELETE FROM table\_name*

*WHERE [condition];*

You can combine N number of conditions using AND or OR operators.

**Ex-**

Before Deleting

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** | **Email** |
| S0101 | Raj sharma | 1999-05-09 | Suhag Nagar Firozabad | Firozabad | 8273555968 | 45236.15 | rajsharma234@gmail.com |
| S0102 | Lucky Sharma | 2003-11-09 | Ghatlodia ahmdabad | Ahmdabad | 9673595968 | 48236.18 | luckysharma543@gmail.com |
| S0103 | Anjali saxena | 2000-01-19 | Gandhi nagar Delhi | Delhi | 6397367278 | 15246.18 | Saxenaanjali000@gmail.com |
| S0104 | Sourya Bansal | 2004-01-19 | lohiya nagar Delhi | Delhi | 9657367278 | 56246.18 | Souyabansal000@gmail.com |
| S0105 | Lavi Bansal | 1996-02-12 | lohiya nagar Patana | Patana | 9657852278 | 58246.18 | Lavibansal02@gmail.com |

*delete from student where rollno='S0101';*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** | | | **Email** |
| S0102 | Lucky Sharma | 2003-11-09 | Ghatlodia ahmdabad | Ahmdabad | 9673595968 | | 48236.18 | luckysharma543@gmail.com | |
| S0103 | Anjali saxena | 2000-01-19 | Gandhi nagar Delhi | Delhi | 6397367278 | | 15246.18 | Saxenaanjali000@gmail.com | |
| S0104 | Sourya Bansal | 2004-01-19 | lohiya nagar Delhi | Delhi | 9657367278 | | 56246.18 | Souyabansal000@gmail.com | |
| S0105 | Lavi Bansal | 1996-02-12 | lohiya nagar Patana | Patana | 9657852278 | | 58246.18 | Lavibansal02@gmail.com | |

If you want to DELETE all the records from the CUSTOMERS table, you do not need to use the WHERE clause and the DELETE query would be as follows −

*DELETE from student;*

1. ***DROP:-***
2. The SQL **DROP TABLE** statement is used to remove a table definition and all the data, indexes, triggers, constraints and permission specifications for that table.

**NOTE** − You should be very careful while using this command because once a table is deleted then all the information available in that table will also be lost forever.

**Syntax**

The basic syntax of this **DROP TABLE** statement is as follows −

*DROP TABLE table\_name;*

1. The SQL DROP DATABASE statement is used to drop an existing database in SQL schema.

**Syntax**

The basic syntax of DROP DATABASE statement is as follows −

*DROP DATABASE DatabaseName;*

Always the database name should be unique within the RDBMS.

1. **ORDER BY:-**

The SQL **ORDER BY** clause is used to sort the data in ascending or descending order, based on one or more columns. Some databases sort the query results in an ascending order by default.

**Syntax-**

The basic syntax of the ORDER BY clause is as follows −

*SELECT column-list*

*FROM table\_name*

*[WHERE condition]*

*[ORDER BY column1, column2, .. columnN] [ASC | DESC];*

You can use more than one column in the ORDER BY clause. Make sure whatever column you are using to sort that column should be in the column-list.

Ex-

Consider the **Student table** having the following records –

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** | **Email** |
| S0102 | Lucky Sharma | 2003-11-09 | Ghatlodia ahmdabad | Ahmdabad | 9673595968 | 48236.18 | luckysharma543@gmail.com |
| S0103 | Anjali saxena | 2000-01-19 | Gandhi nagar Delhi | Delhi | 6397367278 | 15246.18 | Saxenaanjali000@gmail.com |
| S0104 | Sourya Bansal | 2004-01-19 | lohiya nagar Delhi | Delhi | 9657367278 | 56246.18 | Souyabansal000@gmail.com |
| S0105 | Lavi Bansal | 1996-02-12 | lohiya nagar Patana | Patana | 9657852278 | 58246.18 | Lavibansal02@gmail.com |

The following table has an example, which would sort the result in an ascending order by the Fee –

*SELECT fee from student order by fee;*

|  |
| --- |
| Fee |
| 15246.18 |
| 48236.18 |
| 56246.18 |
| 58246.18 |

1. **DISTINCT:-**

The SQL DISTINCT keyword is used in conjunction with the SELECT statement to eliminate all the duplicate records and fetching only unique records.

There may be a situation when you have multiple duplicate records in a table. While fetching such records, it makes more sense to fetch only those unique records instead of fetching duplicate records.

**Syntax**

The basic syntax of DISTINCT keyword to eliminate the duplicate records is as follows −

*SELECT DISTINCT column1, column2,.....columnN*

*FROM table\_name*

*WHERE [condition]*

Ex-

*SELECT DISTINCT Fee FROM Student*

*ORDER BY Fee;*

1. **GROUP BY:-**

The SQL GROUP BY clause is used in collaboration with the SELECT statement to arrange identical data into groups. This GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

**Syntax**

The basic syntax of a GROUP BY clause is shown in the following code block. The GROUP BY clause must follow the conditions in the WHERE clause and must precede the ORDER BY clause if one is used.

*SELECT column1, column2*

*FROM table\_name*

*WHERE [ conditions ]*

*GROUP BY column1, column2*

*ORDER BY column1, column2;*

**Ex-**

*SELECT NAME, SUM(Fee) FROM Student*

*GROUP BY NAME;*

|  |  |
| --- | --- |
| Name | Sum(Fee) |
| Anjali saxena | 15246.18 |
| lavi Bansal | 58246.18 |
| Lucky Sharma | 48236.18 |
| Sourya Bansal | 56246.18 |

1. ***LIKE Clause:-***

The SQL **LIKE** clause is used to compare a value to similar values using wildcard operators. There are two wildcards used in conjunction with the **LIKE** operator.

The percent sign (%)

The underscore (\_)

The percent sign represents zero, one or multiple characters. The underscore represents a single number or character. These symbols can be used in combinations.

**Syntax**

The basic syntax of % and \_ is as follows –

*SELECT FROM table\_name*

*WHERE column LIKE 'RRRR%';*

or

*SELECT FROM table\_name*

*WHERE column LIKE '%RRRR%';*

or

*SELECT FROM table\_name*

*WHERE column LIKE 'RRRR\_';*

or

*SELECT FROM table\_name*

*WHERE column LIKE '\_RRRR’;*

or

*SELECT FROM table\_name*

*WHERE column LIKE '\_RRRR\_';*

Here RRRR could be any numeric or string value.

|  |  |
| --- | --- |
| Sr No | Statement & Description |
| 1 | WHERE SALARY LIKE '200%'  Finds any values that start with 200. |
| 2 | **WHERE SALARY LIKE '%200%'**  Finds any values that have 200 in any position. |
| 3 | WHERE SALARY LIKE '\_00%'  Finds any values that have 00 in the second and third positions. |
| 4 | WHERE SALARY LIKE '2\_%\_%'  Finds any values that start with 2 and are at least 3 characters in length. |
| 5 | WHERE SALARY LIKE '%2'  Finds any values that end with 2. |
| 6 | WHERE SALARY LIKE '\_2%3'  Finds any values that have a 2 in the second position and end with a 3. |
| 7 | WHERE SALARY LIKE '2\_\_\_3'  Finds any values in a five-digit number that start with 2 and end with 3. |

**Ex-**

Following is an example, which would display all the records from the Student table, where the Fee starts with 5.

*select \* from Student where fee like '5%';*

*This would produce the following result* −

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Rollno | Name | DOB | Address | City | Mob | Fee | Email |
| S0104 | Sourya  Bansal | 2004-01-19 | Lohiya  nagar  Delhi | Delhi | 9657367278 | 56246.18 | Souyabansal000@gmail.com |
| S0105 | Lavi  Bansal | 1996-02-12 | Lohiya  nagar  Patana | Patana | 9657852278 | 58246.18 | Lavibansal02@gmail.com |

**SQL  Aggregate FunctionS**

***Consider A sTudent Table***

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Rollno** | **Name** | **DOB** | **Address** | **City** | **Mob** | **Fee** | **Email** |
| S0102 | Lucky Sharma | 2003-11-09 | Ghatlodia ahmdabad | Ahmdabad | 9673595968 | 48236.18 | luckysharma543@gmail.com |
| S0103 | Anjali saxena | 2000-01-19 | Gandhi nagar Delhi | Delhi | 6397367278 | 15246.18 | Saxenaanjali000@gmail.com |
| S0104 | Sourya Bansal | 2004-01-19 | lohiya nagar Delhi | Delhi | 9657367278 | 56246.18 | Souyabansal000@gmail.com |
| S0105 | Lavi Bansal | 1996-02-12 | lohiya nagar Patana | Patana | 9657852278 | 58246.18 | Lavibansal02@gmail.com |

1. ***SUM:-***

SQL SUM function is used to find out the sum of a field in various records.

Now suppose based on the above table you want to calculate total of all the Fee, then you can do so by using the following command –

*select Sum(fee) from student;*

|  |
| --- |
| *Sum(fee)* |
| *177974.72* |

1. ***Min:-***SQL MIN function is used to find out the record with minimum value among a record set.

Now suppose based on the above table you want to fetch minimum value of min, then you can do so simply using the following command –

SELECT MIN(fee) from student;

|  |
| --- |
| *MIN(fee)* |
| *15246.18* |

1. ***MAX:-*** SQL **MAX** function is used to find out the record with maximum value among a record set.

Now suppose based on the above table you want to fetch maximum value of Fee, then you can do so simply using the following command −

*SELECT MAX(fee) from student;*

|  |
| --- |
| *MAX(fee)* |
| *58246.18* |

1. ***AVG:-*** SQL AVG function is used to find out the average of a field in various records.

Now suppose based on the above table you want to calculate average of all the dialy\_typing\_pages, then you can do so by using the following command –

*SELECT AVG(fee) from student;*

|  |
| --- |
| *AVG(fee)* |
| *48086.314000* |

1. ***COUNT:-***

SQL **COUNT** function is the simplest function and very useful in counting the number of records, which are expected to be returned by a SELECT statement.

Now suppose based on the above table you want to count total number of rows in this table, then you can do it as follows –

*SELECT COUNT(\*) from Student;*

|  |
| --- |
| *COUNT(fee)* |
| *44493.68* |

***Thank You***