**update** -First, specify the name of the table that you want to update data after the UPDATE keyword.

Second, specify columns and their new values after SET keyword. The columns that do not appear in the SET clause retain their original values.

Third, determine which rows to update in the condition of the WHERE clause.

**update taazaatraining set name ='rohan'**

**day9-# where id = 2;**

* **delete -** First, specify the name of the table from which you want to delete data after the DELETE FROM keywords.
* Second, use a condition in the [WHERE](https://www.postgresqltutorial.com/postgresql-where/) clause to specify which rows from the table to delete.

**delete from taazaatraining**

**day9-# where id = 3;**

**where keyword :** The WHERE clause appears right after the FROM clause of the SELECT statement.  The WHERE clause uses the condition to filter the rows returned from the SELECT clause.

The condition must evaluate to true, false, or unknown. It can be a boolean expression or a combination of boolean expressions using the AND and OR operators.

The query returns only rows that satisfy the condition in the WHERE clause. In other words, only rows that cause the condition evaluates to true will be included in the result set.

**select name**

**day9-# from taazaatraining**

**day9-# where**

**day9-# name ='rohan';**

**name**

**-------**

**rohan**

**(1 row)**

**Order by :** In this section, we are going to learn the PostgreSQL **ORDER BY condition**, which is used for sorting data in **ascending or descending order**. And the records are fetched on the basis of one or more columns.

When fetching the records from a table, the SELECT command returns rows in an undetermined order. For this, we will use the ORDER BY clause in the SELECT statement for sorting the rows of the result set.

The **ORDER BY** clause permits us to sort rows returned by a [SELECT](https://www.javatpoint.com/postgresql-select) condition in ascending or descending order depends on the sort expression.

**day9=# select name**

**day9-# from taazaatraining**

**day9-# order by name asc;**

**name**

**-------------**

**lucky singh**

**rohan**

**(2 rows)**

**Group by:** Most importantly, this clause is used to split rows into groups where the GROUP BY condition collects the data across several records and sets the result by one or more columns.

And every group can apply an aggregate function like **COUNT() function** is used to get the number of items in the groups, and the **SUM() function** is used to analyze the sum of items.

**day9=# select id**

**day9-# from taazaatraining**

**day9-# group by id;**

**id**

**----**

**2**

**1**

**(2 rows)**

**Having :** The having clause is used to specify a search condition for a group or an aggregate. And it is regularly used with the GROUP BY clause to filter groups or aggregates based on a detailed condition.

**select name,count(id)**

**day9-# from taazaatraining**

**day9-# group by name**

**day9-# having count(id)<1**

**day9-# ;**

**name | count**

**------+-------**

**(0 rows)**

**Distinct : is** used to delete the matching rows or data from a table and get only the unique records.

**day9=# select distinct name**

**day9-# from taazaatraining**

**day9-# order by name;**

**name**

**-------------**

**lucky singh**

**rohan**

**(2 rows)**

**Limit :** LIMIT clause, which is used to get a subset of rows produced by a command.

**select id,name,age**

**day9-# from taazaatraining**

**day9-# order by id**

**day9-# limit 5;**

**id | name | age**

**----+-------------+-----**

**1 | lucky singh | 22**

**2 | rohan | 23**

**(2 rows)**

**Fetch : FETCH clause, which is used to recover various rows returned by a command. And**the**FETCH clause**was launched in 2008 by SQL.

**day9=# select id,name,age**

**day9-# from taazaatraining**

**day9-# order by name**

**day9-# fetch first row only;**

**id | name | age**

**----+-------------+-----**

**1 | lucky singh | 22**

**(1 row)**

**IN :** The **PostgreSQL IN condition** is used within the [WHERE clause](https://www.javatpoint.com/postgresql-where-clause) to get those data that matches any data in a list. In other words, we can say that the **IN condition** is used to reduce multiple [OR conditions](https://www.javatpoint.com/postgresql-or-condition)

**day9=# select \* from taazaatraining**

**day9-# where name in('rohan')**

**day9-# order by id desc;**

**id | name | address | age**

**----+-------+---------+-----**

**2 | rohan | aligarh | 23**

**(1 row)**

**Like :** The PostgreSQL Like condition is used to fetch data from a table where the defined condition satisfies the LIKE condition. The result contains strings, which are **case-sensitive** and follow the specified pattern.

In other words, we can also say that the **Like condition** is used to perform pattern matching for identifying the exact outcome.

**day9=# select name**

**day9-# from Taazaatraining**

**day9-# where name like'rohan'**

**day9-# ;**

**name**

**-------**

**rohan**

**(1 row)**

**And & or : AND & OR** Condition can be combined with the [INSERT](https://www.javatpoint.com/postgresql-insert), [SELECT](https://www.javatpoint.com/postgresql-select), [UPDATE](https://www.javatpoint.com/postgresql-update) and [DELETE](https://www.javatpoint.com/postgresql-delete) commands. While combining these conditions, we must be alert of where to use the parentheses so that the database knows the order to assess every condition.

**day9=# select id,name,age**

**day9-# from taazaatraining**

**day9-# where(name = 'rohan' and address = 'aligarh')**

**day9-# or(age >=23);**

**id | name | age**

**----+-------+-----**

**2 | rohan | 23**

**(1 row)**

**Between :** The **PostgreSQL Between condition** is used to define how to retrieve values from an expression within a specific range. In other words, we can say that the **Between condition** is used to match a value against a range of values.

**day9=# select id,name,age**

**day9-# from taazaatraining**

**day9-# where age between 22 and 25**

**day9-# order by age desc;**

**id | name | age**

**----+-------------+-----**

**2 | rohan | 23**

**1 | lucky singh | 22**

**(2 rows)**

**Exist** the EXISTS condition can combine with the [SELECT](https://www.javatpoint.com/postgresql-select), [INSERT](https://www.javatpoint.com/postgresql-insert), [UPDATE](https://www.javatpoint.com/postgresql-update), and [DELETE](https://www.javatpoint.com/postgresql-delete) commands. In other words, we can say that the EXISTS condition is used to check for the presence of any data in a subquery, and returns true if the subquery returns several records.

It is used to group with a subquery and test the existence of records in a subquery. The PostgreSQL EXISTS condition is a type of Boolean operator which returns the true or false result. In other words, we can say that:

* If a subquery returns any record, the Exists condition will return a TRUE value, or else, it will return a FALSE
* Here the TRUE value is always illustrated with numeric value 1, and the FALSE value is denoted as numeric value 0.

**Date:** n [PostgreSQL](https://www.javatpoint.com/postgresql-tutorial), the **Date data type** format is **YYYY-MM-DD**, which helps us to store and **insert the date records into the date column**.

The Date data type involve **4 bytes** of storage size. And the minimum and maximum ranges of date data type start with **4713 BC to 5874897 AD** or **1000-01-01 to 9999-12-31**.

day9=# alter table taazaatraining add sdate date default current\_date;

ALTER TABLE

day9=# insert into taazaatraining(name,address,age) values('sohan','agra',22);

INSERT 0 1

day9=# select \* from taazaatraining;

id | name | address | age | sdate

----+-------------+---------+-----+------------

1 | lucky singh | agra | 22 | 2021-08-25

2 | rohan | aligarh | 23 | 2021-08-25

4 | sohan | agra | 22 | 2021-08-25

(3 rows)

**Time** : In [PostgreSQL](https://www.javatpoint.com/postgresql-tutorial), the next data type is **TIME**, which stores the **Time of days values**.

The PostgreSQL Time Data type involves **8 bytes of storage** and up to **6 digits of precision**, and the range starts from 00:00:00 to 24:00:00 for Time data type.

day9=# alter table taazaatraining add stime time default current\_time;

ALTER TABLE

day9=# insert into taazaatraining(name,address,age) values('ravi','merrut',24);

INSERT 0 1

day9=# select \* from taazaatraining;

id | name | address | age | sdate | stime

----+-------------+---------+-----+------------+-----------------

1 | lucky singh | agra | 22 | 2021-08-25 | 17:11:37.606642

2 | rohan | aligarh | 23 | 2021-08-25 | 17:11:37.606642

4 | sohan | agra | 22 | 2021-08-25 | 17:11:37.606642

5 | ravi | merrut | 24 | 2021-08-25 | 17:12:03.523646

(4 rows)

**Numeric :** In [PostgreSQL](https://www.javatpoint.com/postgresql-tutorial), the **Numeric data type** is used to **store the numbers with various significant numbers of digits**. In other words, we can say that the PostgreSQL Numeric [data type](https://www.javatpoint.com/postgresql-datatypes) is used to specify the numeric data into the table, which needs quantities or monetary amounts where the precision is required.

**day9=# alter table taazaatraining alter age type numeric;**

**ALTER TABLE**

**day9=# insert into taazaataining(age) values(221210266226);**

**ERROR: relation "taazaataining" does not exist**

**LINE 1: insert into taazaataining(age) values(221210266226);**

**^**

**day9=# insert into taazaatraining(age) values(221210266226);**

**INSERT 0 1**

**day9=# select \* from taazaatraining;**

**id | name | address | age | sdate | stime**

**----+-------------+---------+--------------+------------+-----------------**

**1 | lucky singh | agra | 22 | 2021-08-25 | 17:11:37.606642**

**2 | rohan | aligarh | 23 | 2021-08-25 | 17:11:37.606642**

**4 | sohan | agra | 22 | 2021-08-25 | 17:11:37.606642**

**5 | ravi | merrut | 24 | 2021-08-25 | 17:12:03.523646**

**6 | | | 221210266226 | 2021-08-25 | 17:44:26.326404**

**(5 rows)**

**Char :** In **PostgreSQL**, the character data represent the character type values and also known as **CHAR**. In other words, we can say that the PostgreSQL character data type is used to store a character of limited length.

**day9=# alter table taazaatraining add gender varchar(1) default 'm';**

**ALTER TABLE**

**day9=# select \* from taazaatraining;**

**id | name | address | age | sdate | stime | gender**

**----+-------------+---------+--------------+------------+-----------------+--------**

**1 | lucky singh | agra | 22 | 2021-08-25 | 17:11:37.606642 | m**

**2 | rohan | aligarh | 23 | 2021-08-25 | 17:11:37.606642 | m**

**4 | sohan | agra | 22 | 2021-08-25 | 17:11:37.606642 | m**

**5 | ravi | merrut | 24 | 2021-08-25 | 17:12:03.523646 | m**

**6 | | | 221210266226 | 2021-08-25 | 17:44:26.326404 | m**

**(5 rows)**

**day9=# alter table taazaatraining alter gender type char(1);**

**ALTER TABLE**

**day9=# select \* from taazaatraining;**

**id | name | address | age | sdate | stime | gender**

**----+-------------+---------+--------------+------------+-----------------+--------**

**1 | lucky singh | agra | 22 | 2021-08-25 | 17:11:37.606642 | m**

**2 | rohan | aligarh | 23 | 2021-08-25 | 17:11:37.606642 | m**

**4 | sohan | agra | 22 | 2021-08-25 | 17:11:37.606642 | m**

**5 | ravi | merrut | 24 | 2021-08-25 | 17:12:03.523646 | m**

**6 | | | 221210266226 | 2021-08-25 | 17:44:26.326404 | m**

**(5 rows)**