## LINUX - HOW TO USE POSTGRES

Reference: <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-postgresql-on-ubuntu-18-04>

* Select user postgres

sudo -i -u postgres

* Open psql
* Do the database operation in psql
* List command:
  + \? list all the commands
  + \l list databases
  + \conninfo display information about current connection
  + \c [DBNAME] connect to new database, e.g., \c template1
  + \dt list tables
  + Then you can run SQL statements, e.g., SELECT \* FROM my\_table;(Note: a statement must be terminated with semicolon ;)
  + \dS [TABLENAME] to see schema of the table
  + \q quit psql

## SQL COMMAND ORDER

**SELECT** column\_name(s)

**FROM** table\_name

**WHERE** condition

**GROUP BY** column\_name(s)

**HAVING** condition

**ORDER BY** column\_name(s);

## CONCAT STRING

## *Concate string in result*

*Use* || *- ex*:

select name || ‘ ‘ || from sometable;

## CASE

## *If-else conditional*

*Usage in select, ex:*

select m.firstname || ' ' || m.surname as member, f.name,

**case**

**when m.memid=0 then**

**f.guestcost**

**else**

**f.membercost**

**end as cost**

from members as m

… ;

## SUB QUERY

## *Query inside query*

*Usage in select, ex:*

select distinct mems.firstname || ' ' || mems.surname as member,  
 **(select recs.firstname || ' ' || recs.surname as recommender   
 from cd.members recs   
 where recs.memid = mems.recommendedby  
 )**  
 from   
 cd.members mems  
order by member;

select name from **(**

**select m.firstname as name**

**from member m**

**where m.id > 0**

**) as listmember**

where … ;

## IN OPERATOR

## *Select condition of multiple values, shorthand of multiple OR condition*

*Usage ex:*

select \* from listmember

where id **in** (1,3,5,7,9);

select \* from listmember as lm

where lm.id **in** (

select lu.id from listuser

);

## UNIQUE RESULT (DISTINCT)

## *Get unique / distinct result of the column*

*Example:*

*Use* ‘distinct’, *ex*:

select **distinct** name from list;

## UNION

## *Combining results from multiple queries*

*Example:*

select surname   
 from cd.members  
**union**  
select name  
 from cd.facilities;

## LEFT JOIN

ex:

SELECT

c.customerNumber,

c.customerName,

orderNumber,

o.status

FROM

customers c

**LEFT JOIN** orders o **ON** c.customerNumber = o.customerNumber;

## RIGHT JOIN

ex:

SELECT

concat(e.firstName,' ', e.lastName) salesman,

e.jobTitle,

customerName

FROM

employees e

**RIGHT JOIN**

customers c **ON** e.employeeNumber = c.salesRepEmployeeNumber

AND e.jobTitle = 'Sales Rep'

ORDER BY customerName;

## INNER JOIN

ex:

SELECT

productCode,

productName,

textDescription

FROM

products t1

**INNER JOIN**

productlines t2 **ON** t1.productline = t2.productline;

## HAVING

<http://www.mysqltutorial.org/mysql-having.aspx>

*The HAVING clause is often used with the* [*GROUP BY*](http://www.mysqltutorial.org/mysql-group-by.aspx) *clause to filter groups based on a specified condition. If the GROUP BY clause is omitted, the HAVING clause behaves like the* [*WHERE*](http://www.mysqltutorial.org/mysql-where/) *clause.*

ex:

SELECT

ordernumber,

SUM(quantityOrdered) AS itemsCount,

SUM(priceeach\*quantityOrdered) AS total

FROM

orderdetails

**GROUP BY** ordernumber

**HAVING** total > 1000 AND itemsCount > 600;

## GROUP BY

<http://www.mysqltutorial.org/mysql-group-by.aspx>

*The GROUP BY clause groups a set of rows into a set of summary rows by values of columns or expressions. The GROUP BY clause returns one row for each group. In other words, it reduces the number of rows in the result set.*

*You often use the GROUP BY clause with* [*aggregate functions*](http://www.mysqltutorial.org/mysql-aggregate-functions.aspx) *such as* [*SUM*](http://www.mysqltutorial.org/mysql-sum/)*,* [*AVG*](http://www.mysqltutorial.org/mysql-avg/)*,* [*MAX*](http://www.mysqltutorial.org/mysql-max-function/)*,* [*MIN*](http://www.mysqltutorial.org/mysql-min/)*, and* [*COUNT*](http://www.mysqltutorial.org/mysql-count/)*. The aggregate function that appears in the SELECT clause provides the information about each group.*

*Without aggregate, GROUP BY work like distinct.*

ex:

SELECT

orderNumber,

**SUM**(quantityOrdered \* priceEach) AS total

FROM

orderdetails

**GROUP BY** orderNumber;

## LIKE

*search for specified pattern in field, used in WHERE*

Wildcard list:

% : The percent sign represents zero, one, or multiple characters   
\_ : The underscore represents a single character

ex:  
SELECT \* FROM Customers  
WHERE ContactName **LIKE** 'a%o';

## AGGREGATE

COUNT(), AVG() and SUM(), MIN(), MAX() Functions