

# PostGres Cheat Sheet

## POSTGRES NOTES

1. Always put “ ” around column names and table names. Put ‘ ‘ around values (like when inserting).
2. Using the command line Login command line with psql -d databasename or psql -d database name -U:username
3. Remember at the command line you need a semi colon at the end of all statements.
4. SQL Terms Not supported in Postgres
  - a. mid
  - b. len
  - c. round()
  - d. format()
  - e. first()
  - f. top ( ) (but you can use limit instead!)
  - g. ifnull()

## SQL NOTES

### Select Statement

SELECT column\_name(s) FROM table\_name

Or to get everything in that table:

SELECT \* FROM table\_name

Example

SELECT LastName,FirstName FROM Employees

### The Where Clause

SELECT column\_name(s) FROM table\_name WHERE column\_name operator value

SELECT \* FROM Employees WHERE City=

'Richardson'

### Operators

= Equal

Not equal

> Greater than

< Less than

>= Greater than or equal <

= Less than or equal BETWEEN

Between an inclusive range

LIKE Search for a pattern

IN If you know the exact value you want to return for at least one of the columns

### **Modifiers**

You may have more than one criteria. In which case you will want to use a modifier with your SELECT statement.

- ▶ The AND operator displays a record if both the first condition and the second condition is true.
- ▶ The OR operator displays a record if either the first condition or the second condition is true.
- ▶ `SELECT * FROM Employees WHERE City='Plano' AND LastName='Johnson'`
- ▶ `SELECT * FROM Employees WHERE City='Plano' OR City='Frisco'`

### **Distinct**

`SELECT DISTINCT column_name(s) FROM table_name`

Example

`SELECT DISTINCT City FROM Employees`

### **Order By**

`SELECT column_name(s) FROM table_name ORDER BY column_name(s) ASC|DESC`

### Example

```
SELECT * FROM Employees ORDER BY LastName
```

### Between

The BETWEEN operator selects a range of data between two values.

```
SELECT column_name(s)  
FROM table_name  
WHERE column_name  
BETWEEN value1 AND value2
```

```
SELECT * FROM Employees  
WHERE LastName  
BETWEEN 'Jones' AND 'McMurray'
```

### SQL Functions

```
SQL AVG()  
SQL COUNT()  
SQL FIRST()  
SQL LAST()  
SQL MAX()  
SQL MIN()  
SQL SUM()
```

#### AVG Function

```
SELECT AVG(column_name) FROM table_name
```

#### Example

```
SELECT AVG(Salary) FROM Employees
```

#### Count Function

```
SELECT COUNT(column_name) FROM table_name
```

#### Example

```
SELECT COUNT(LastName) FROM Employees
```

#### First Function

```
SELECT FIRST(column_name) FROM table_name
```

```
SELECT FIRST(Last Name) FROM Employees
```

### **Max Function**

```
SELECT MAX(column_name) FROM table_name
```

```
SELECT MAX(Salary) FROM Employer
```

### **Sum Function**

```
SELECT SUM(column_name) FROM table_name
```

```
SELECT SUM(Salary) FROM Employees
```

### **Inner Join**

```
SELECT column_name(s) FROM table_name1 INNER JOIN table_name2 ON  
table_name1.column_name=table_name2.column_name
```

```
SELECT Employees.LastName, Employees.FirstName, Orders.OrderNo FROM  
Employees INNER JOIN Orders ON Employees.P_Id=Orders.P_Id ORDER BY  
Employees.LastName
```

### **Outer Join**

```
SELECT column_name(s) FROM table_name1 FULL OUTER JOIN table_name2 ON  
table_name1.column_name=table_name2.column_name
```

```
SELECT Employees.LastName, Employees.FirstName, Orders.OrderNo FROM  
Employees FULL OUTER JOIN Orders ON Employees.P_Id=Orders.P_Id ORDER  
BY Employees.LastName
```

### **LIMIT**

```
Select * from mytable LIMIT 5
```

### **IS NULL**

```
SELECT Description, ISNULL(Qty, 0.00) AS 'Max Quantity' FROM Sales;
```

Works but only with numeric data types

### **IFNULL**

```
IFNULL (expression-1,expression-2,expression-3)
```

expression-1 The expression to be evaluated to determine if it is NULL or not.

expression-2 An expression that is returned if expression-1 is NULL.

expression-3 Optional – An expression that is returned if expression-1 is not NULL. If expression-3 is not specified, a NULL value is returned when expression-1 is not NULL.

```
SELECT Name, IFNULL(FavoriteColors,'No Preference') AS ColorPref FROM Sample.Person
```

### **CREATE DB**

```
CREATE DATABASE database_name
```

Example

```
CREATE DATABASE my_db
```

### **CREATE TABLE**

```
CREATE TABLE table_name ( column_name1 data_type, column_name2 data_type, column_name3 data_type, .... )
```

Example:

```
CREATE TABLE Employees  
  
( P_Id int,  
  
LastName varchar(255),  
  
FirstName varchar(255),  
  
Address varchar(255),  
  
City varchar(255) )
```

### **ALTER**

The ALTER TABLE statement is used to add, delete or modify columns in an existing table.

```
ALTER TABLE table_name ADD column_name datatype
```

```
ALTER TABLE table_name DROP COLUMN column_name
```

```
ALTER TABLE table_name ALTER COLUMN column_name datatype
```

Examples:

```
ALTER TABLE Employees ADD DateOfBirth date
```

```
ALTER TABLE Employees DROP COLUMN DateOfBirth
```

```
ALTER TABLE Persons ADD PRIMARY KEY (P_Id)
```

### **DATE DIFF**

DATEDIFF(d,date1,date2)

### **TIMEOFDAY**

timeofday() works just like NOW() only it returns the time of day rather than current date.

### **INTERSECTS**

The INTERSECT query allows you to return the results of 2 or more "select" queries. However, it only returns the rows selected by all queries. If a record exists in one query and not in the other, it will be omitted from the INTERSECT results.

```
select field1, field2, . field_n
from tables
INTERSECT
select field1, field2, . field_n
from tables;
```

```
select supplier_id, supplier_name
from suppliers
where supplier_id > 2000
INTERSECT
select company_id, company_name
from companies
where company_id > 1000
```

### **UNION**

The SQL UNION operator combines the result-set of two or more SELECT statements.

```
SELECT column_name(s) FROM table_name1 UNION SELECT column_name(s)
FROM table_name2
```

EXAMPLE

```
SELECT LastName FROM Employees
```

UNION

```
SELECT LastName FROM Customers
```

### **SELECT INTO**

The SELECT INTO statement selects data from one table and inserts it into a different table.

```
SELECT * INTO new_table_name [IN externaldatabase] FROM old_tablename
SELECT * INTO Employees_Backup FROM Employees
```

## **Mathematical Operators**

<b>Operator</b>	<b>Description</b>	<b>Example</b>	<b>Result</b>
+	addition	2 + 3	5
-	subtraction	2 - 3	-1
*	multiplication	2 * 3	6
/	division (integer division truncates results)	4 / 2	2
%	modulo (remainder)	5 % 4	1
^	exponentiation	2.0 ^ 3.0	8
/	square root	/ 25.0	5
/	cube root	/ 27.0	3
!	factorial	5 !	120
!!	factorial (prefix operator)	!! 5	120
@	absolute value	@ -5.0	5
&	bitwise AND	91 & 15	11
	bitwise OR	32   3	35
#	bitwise XOR	17 # 5	20
~	bitwise NOT	~1	-2
<<	bitwise shift left	1 << 4	16
>>	bitwise shift right	8 >> 2	2