

***PostgreSQL***

***¿QUE ES POSTGRESQL?***

Es un motor de base de datos relacional pero también me permite trabajar de manera no relacional con una extensión llamada ***hstore.***

Es el motor de base de datos más avanzada Open Source *(Es libre para todos).*

Es una base de datos que tiene tiempo nació en California.

Actualmente se encuentra en la versión 9, es preferible trabajar con la versión estable.

***¿POR QUÉ USAR POSTGRESQL?***

Por que soporta:

***Stored Procedures:*** Son fragmentos de código, que escribimos, específicos y exclusivos para la base de datos, que nosotros guardamos dentro de nuestra base de datos, para luego utilizarlos, tienen la característica que se puede escribir mediante diferentes LP’S*(extensiones)*.

Los leguajes activos SQL, PL/pgSQL, PL/Perl, PL/Python, PL/V8

Podemos activar o instalar la extensión de Python para que podamos escribir los códigos de los Stored Procedures en Python.

Podemos instalar la extensión V8 de Google para escribir nuestros Stored Procedure en JavaScript.

***Funciones de Agregación:*** AVG, Count, podemos agregar funciones de agregación personalizadas.

***Datos Complejos:*** Fracciones, Json, hstore, PostGIS.

Podemos extender El PostgreSQL para pedirle que guarde este tipo de datos, para decirle quiero guardar un documento de Json, Fracciones.

Guardar información no estructurada con la extensión ***hstore***. g

Con PostGIS podemos guardar Ubicaciones.

***Herramientas Administrativas:***

***PSQL:*** *Línea de comandos* que ejecutamos en consola, con la cual nos podemos conectar a un motor de base de datos de PostGreSQL.

***PgAdmin:*** Es el IDE de PostgreSQL.

***ENTENDIENDO EL CORE DE POSTGRESQL:***  
***Organización artefactos u objetos lógicos como físicos:***

La base de datos PostgreSQL se compone

Elementos físicos: Librerías, archivos, si esta visible dentro de nuestro File System.

Elementos Lógicos: No esta visible en nuestro File System.

***SERVIDOR FÍSICO:***

Un servidor físico puede tener varios servicios.

***Servicio:*** Es una instancia de PostgreSQL, responde a una dirección IP.

Dentro del ***service*** encontraremos uno o más *database*.

Un service puede contener 1 o más *databases*.

***Database*** -> Es una organización lógica de datos y código. Tiene una relación de archivos físicos, visible en el File System.

***Schema:*** Es una organización lógica de datos y código, un database puede contener 1 o más esquemas.

***Tablespaces:*** Es un archivo físico.

***Catalog:*** PostgreSQL, contiene información de los objetos que tenemos en nuestra database.

***Archivos de Configuración:***

***Postgresql.conf***: Contiene información general sobre postgresql, cuanta memoria va utilizar en el servidor.

***Pg\_hba.conf:*** Controla el acceso de seguridad a nuestro servicio o instancia de postgresql.

***Pg\_ident.conf:*** Trabaja en conjunto con el archivo pg\_hba.conf, nos da el mapeo de usuarios.

***PREGUNTAS:***

***¿Comparación de PostgreSQL y ventajas y desventajas?***

El único punto de comparación seria Oracle que sería más avanzado que PostgreSQL, muchas de las características que tiene PostgreSQL Oracle ya las tuvo antes.

***ROLES:***

Hay dos formas de conectarse a una base de datos:

*1***->***Teniendo un Usuario*

*2****->****Creando un Role*

Los Roles toman dos funcionalidades

***Roles de Inicio de Sesión:*** Podemos asignar un password.

***Roles de grupo:*** Cuentan con permisos, agrupan otros dentro de un rol de grupos.

***ATRIBUTOS DE LOS CAMPOS:***

* ***Unsigned*:** Significa que no guarda los signos.
* ***Primary\_key****:* Llave primaria es Unique, entorno a esta columna se ordena la tabla.
* ***Not null*:** significa que siempre tiene q ir un valor.
* ***Unique****:* que es único, garantiza que no haya elementos repetidos que compartan ese dato.
* ***Auto\_increment***: se autoincrementa.
* ***Default:*** Valor por defecto sino pongo nada.

# ***TIPOS DE DATOS:***

## ***DATOS NUMÉRICOS:***

***NOTA:*** int (1), int (2),..Int (5) se refiere al formato de despliegue no al espacio en bytes utilizados.

* ***NUMÉRICOS ENTEROS:***

***Int (1) == Tyniint****: tamaño o capacidad de almacenamiento=1 byte.*

***Int (2) == Smallint:*** *tamaño o capacidad de almacenamiento=2 bytes.*

***Int (3) == Mediumint:*** *tamaño o capacidad de almacenamiento=3 bytes.*

***Int (4) == Int o integer:*** *tamaño o capacidad de almacenamiento=4 bytes.*

***Int (5) == Bigint****: tamaño o capacidad de almacenamiento=8 bytes.*

***Bit (bool o Boolean):*** *toma 0 o 1.*

* ***NUMÉRICOS DECIMALES***

***Float(x):*** *tamaño o capacidad de almacenamiento=4 u 8 bytes.*

***Float:*** *tamaño o capacidad de almacenamiento=4 bytes.*

***Double****: tamaño o capacidad de almacenamiento=8 bytes.*

***Real:*** *tamaño o capacidad de almacenamiento=8 bytes.*

***Decimal (m, d):*** *m+2 bytes si d>0 y m+1byte si d=0.*

***Numeric (m, d):*** *m+2 bytes si d>0 y m+1byte si d=0.*

## ***DATOS ALFANUMÉRICOS:***

***Char(n):*** *almacena una cadena de longitud fija que pude contener desde 0 hasta 255 caracteres.*

***Varchar (n):*** *Almacena una cadena de longitud variable puede contener de 0 hasta 255 caracteres.*

***Tinytext y tinyblob:***

***Text y blob:*** *Un texto con un máximo de 65535 caracteres.*

***Longtext y longblob:***

***Enum:*** Es una lista de valores que puede tomar el campo, sino pongo ninguno de los valores de la lista MySQL me escupirá un error.

***Set:*** Es una lista de valores que puede tomar el campo.

## ***DATOS DE FECHA Y HORA:***

***NOTA:***

***current\_date () ==curdate ()*** --->solo pone la fecha del sistema pero no la hora.

Sirve para: datetime, date, time, year, timestamp.

**current\_time () ==current\_timestamp ()** ---->pone fecha y hora del sistema.

Sirve para: datetime, date, time, year, timestamp.

**Now ()** -->pone fecha y hora del sistema----->pone tanto fecha como hora del sistema.

Sirve para: datetime, date, time, year, timestamp.

***Date:*** Almacena fecha con su formato: AAAA-MM-DD.

***Datetime:*** Almacena fecha y hora con su formato: AAAA-MM-DD HH-MM-SS.

***Timestamp:*** AL igual que Datetime almacena fecha y hora pero con la diferencia que este tiene 6 formatos disponibles, Para poner fecha y hora automática current\_timestamp.

* AAAA-MM-DD HH-MM-SS
* AA-MM-DD HH-MM-SS
* AAAA-MM-DD
* AA-MM-DD
* AA-MM
* AA

***Year:*** Almacena Año con su formato de año.

***Time:*** Almacena Hora con su formato de hora HH-MM-SS.

# COMANDOS BASICOS:

***Nota:*** El servidor corre de manera local en mi PC no hay necesidad de especificar un host

***Versión:*** *psql --version*

**Ayuda:** psql --help

***Para entrar a línea de comandos:***

***1)****psql –U postgres –h localhost*

***2)****root*

# Usuarios:

***Mostrar User***:

SELECT USER FROM mysql.user;

***Crear User***:

*CREATE USER* Jhordan *WITH PASSWORD ‘*123’;

***Agregar Privilegios:***

GRANT SELECT, UPDATE, DELETE, INSERT ON tienda.\* TO ‘jhordan’@´localhost´;

***Eliminar User:***

DROP USER ‘jhordan’@’localhost’;

# BASE DE DATOS (DB):

***Crear DB***:

CREATE DATABASE tienda;

***Mostrar DB***:

\l

***Eliminar DB:***

Drop database untels;

***Usar DB***:

USE universidad;

***Exportar DB:***

Mysqldump –u root –p librería > C:\backup\_libreria.sql;

Mysqldump –opt --user=root librería > backup\_libreria.sql;

***Importar DB:***

Mysql –u root –p dbcreada < backup\_libreria.sql

Mysql --password=rootroot --user=root libreria2 < backup\_libreria.sql

***ROLES:***

***Crear role:***

CREATE ROLE cursos LOGIN PASSWORD ‘1234’;

***Listar Roles:***

SELECT \* FROM pg\_roles;

***Eliminar Role:***

DROP ROLE cursos;

***Crear role con md5:***

CREATE ROLE cursos LOGIN ENCRYPTED PASSWORD ‘1234’;

***Crear ROLE Permanente***: Acceso a la DB de manera indefinida

CREATE ROLE cursos LOGIN ENCRYPTED PASSWORD ‘1234’ VALID UNTIL ‘infinity’;

***Crear ROLE Temporal:*** Acceso a la DB a partir de hoy hasta el 2018-05-1.

CREATE ROLE cursos LOGIN ENCRYPTED PASSWORD ‘1234’ VALID UNTIL ‘2018-05-1 00:00’;

# SCHEMA:

***Crear DB***:

*CREATE SCHEMA cursos;*

# TABLAS:

***Nota: a la hora de crear tablas lo más recomendable es comenzar con tables que no tengan foreign key, luego se crean las demás.***

***Foreign key solo sirve para mantener la integridad referencial de las tablas.***

* ***Mostrar Tablas:***

SHOW TABLES;

* ***Mostrar Estructura:***

DESC (describe) alumno;

* ***Mostrar La Creación De La Tabla*:**

SHOW CREATE TABLE alumnos;

* ***Eliminar Tabla:***

DROP TABLE universidad;

* ***Alterar tabla agregando claves foráneas:***

ALTER TABLE libros ADD FOREIGN KEY libros (id\_editorial) REFERENCES editoriales (id\_editorial);

* ***Renombrar Tablas:***

RENAME TABLE libros TO libritos;

ALTER TABLE libros RENAME libritos;

* ***Cambiar el motor de almacenamiento:***

ALTER TABLE libros ENGINE=INNODB;

# CAMPOS:

* ***Alterar Tabla Modificando La variable del Campo***:

ALTER TABLE libros MODIFY COLUMN precio (6,2);

* ***Alterar Tabla Modificando el nombre del campo:***

ALTER TABLE libros CHANGE apellidos apellido\_paterno VARCHAR (45);

* ***Alterar tabla agregando una columna:***

ALTER TABLE libros ADD COLUMN descripción VARCHAR (45);

* ***Eliminar Campo De Una Tabla***

ALTER TABLE libros DROP comentarios;

# REGISTROS:

* ***Insertar Registros:***

INSERT INTO alumno (idalumno, nombre, apepat, edad) VALUES (‘1’,’Jhordan’,’Escobar’,’20’);

* ***Mostrar Registros Y Mostrar Registros Mixtos Entre Dos Tablas:***
* SELECT \* FROM alumno;
* Select \* from publishers \G;
* SELECT alumno.nombre, alumno.idcurso, curso.nombre FROM alumno, curso WHERE alumno.idcurso=curso.idcurso;
* SELECT alumno.nombre, alumno.idcurso, curso.nombre FROM alumno INNER JOIN curso ON alumno.idcurso=curso.idcurso;
* ***Eliminar Registro***:

DELETE FROM alumno WHERE idalumno=’1’;

* ***Truncar o vaciar(contenido) tablas:***

TRUNCATE TABLE libros;

* ***Editar Registro***:

UPDATE alumno SET nombre=’Manuel’ WHERE idalumno=’1’;

# TRANSACCIONES:

* ***Desactivar El Commit Que Se Ejecuta Al Principio o poner begin antes de una consulta***:

SET AUTOCOMMIT=0;

* ***Ejecutar Los Cambios en La DB***:

COMMIT;

* ***Retroceder Los Cambios***:

ROLLBACK;