Tipos de datos

Los tipos de datos especifican que tipo de valores son permitidos en cada una de las columnas que conforman la estructura de la fila. Por ejemplo, si desea almacenar precios de productos en una columna debería especificar que el tipo de datos sea **money**, si desea almacenar nombres debe escoger un tipo de dato que permita almacenar información de tipo carácter.

Además, PostgreSQL nos ofrece un conjunto de tipos de datos predefinidos, pero también existe la posibilidad de definir tipos de datos de usuario.

Al asignar un tipo de datos a un objeto se definen cuatro atributos del objeto:

* + La clase de datos que contiene el objeto, por ejemplo, carácter, entero o binario.
  + La longitud del valor almacenado o su tamaño.
  + La precisión del número (sólo tipos de datos numéricos). La precisión es el número de dígitos que puede contener el número. Por ejemplo, un objeto smallint puede contener hasta 5 dígitos, con lo que tiene una precisión de 5.
  + La escala del número (sólo tipos de datos numéricos). La escala es el máximo número de dígitos a la derecha del separador decimal. Por ejemplo, un objeto int no puede aceptar un separador decimal y tiene una escala de 0. Un objeto money puede tener hasta 4 dígitos a la derecha del separador decimal y tiene una escala de 4. Si un objeto se define como money, puede contener hasta 19 dígitos y 4 de ellos pueden estar a la derecha del decimal. El objeto usa 8 bytes para almacenar los datos. Por tanto, el tipo de datos money tiene una precisión de 19, una escala de 4 y una longitud de 8.

La siguiente es una tabla que describe los tipos de datos provistos por PostgreSQL:

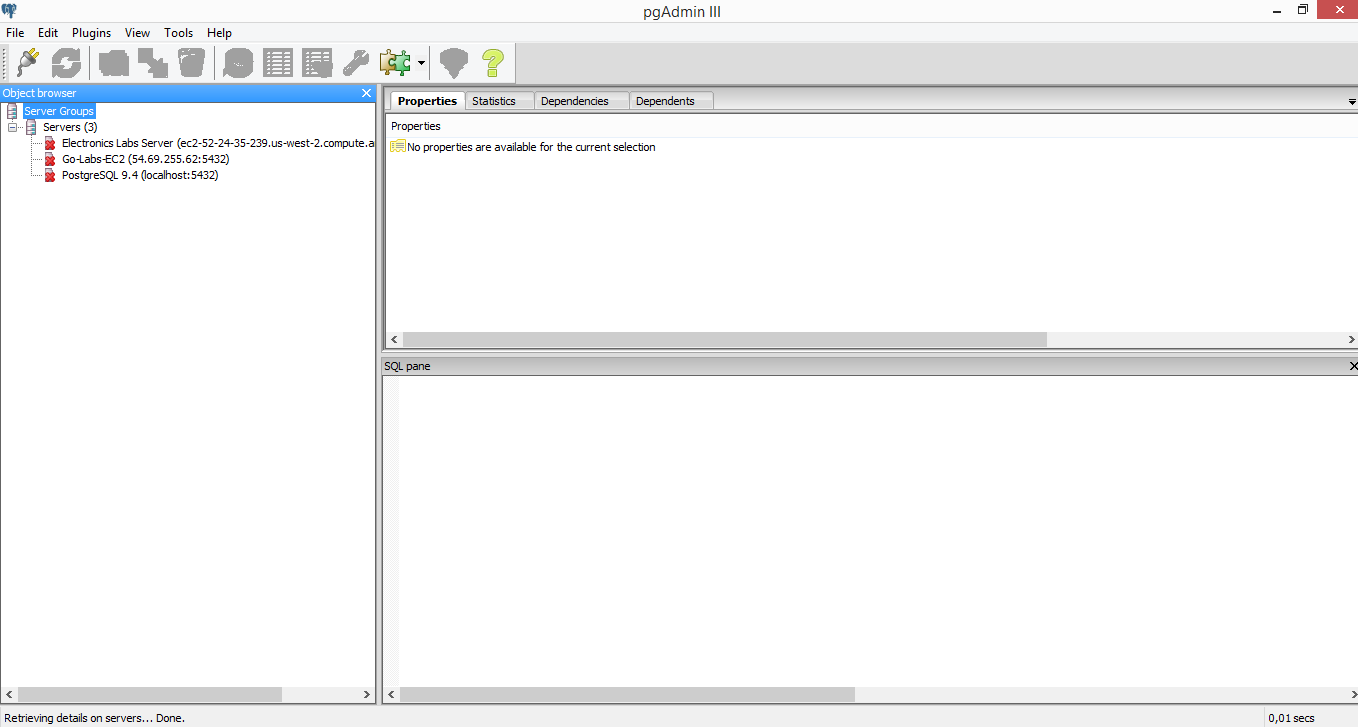
|  |  |  |
| --- | --- | --- |
| Nombre | Alias | Descripción |
| bigint | int8 | entero con signo de ocho bytes |
| bigserial | serial8 | entero autoincremental de ocho bytes |
| bit [ (n) ] |  | cadena de bits de longitud fija |
| bit varying [ (n) ] | varbit | cadena de bits de longitud variable |
| boolean | bool | Booleano lógico (verdadero/falso) |
| box |  | rectángulo en un plano |
| bytea |  | datos binarios ("arreglo de bytes") |
| character varying [ (n) ] | varchar [ (n) ] | cadena de caracteres de longitud variable |
| character [ (n) ] | char [ (n) ] | cadena de caracteres de longitud fija |
| cidr |  | dirección de red IPv4 o IPv6 |
| circle |  | circulo en un plano |
| date |  | fecha de calendario (año, mes, día) |
| double precision | float8 | número de punto flotante de precisión doble (8 bytes) |
| inet |  | dirección de equipo de IPv4 o IPv6 |
| integer | int, int4 | entero con signo de cuatro bytes |
| interval [ fields ] [ (p) ] |  | lapso de tiempo |
| line |  | Linea infinita en un plano |
| lseg |  | segmento de linea en un plano |
| macaddr |  | Dirección MAC (Media Access Control) |
| money |  | importe monetario |
| numeric [ (p, s) ] | decimal [ (p, s) ] | numérico exacto de precisión seleccionable |
| path |  | camino geométrico en un plano |
| point |  | punto geométrico en un plano |
| polygon |  | camino cerrado geométrico en un plano |
| real | float4 | número de punto flotante de precisión simple (4 bytes) |
| smallint | int2 | entero con signo de dos bytes |
| serial | serial4 | entero autoincremental de cuatro bytes |
| text |  | cadena de caracteres de longitud variable |
| time [ (p) ] [ without time zone ] |  | hora del día (sin zona horaria) |
| time [ (p) ] with time zone | timetz | gora del día, incluyendo zona horaria |
| timestamp [ (p) ] [ without time zone ] |  | fecha y hora (sin zona horaria) |
| timestamp [ (p) ] with time zone | timestamptz | fecha y hora, incluyendo zona horaria |
| tsquery |  | consulta de búsqueda de texto |
| tsvector |  | documento de búsqueda de texto |
| txid\_snapshot |  | instantánea de ID de transacción a nivel de usuario |
| uuid |  | identificador universalmente único |
| xml |  | datos XML |

Los tipos de datos definidos por el usuario están basados en los tipos de datos disponibles a través de PostgreSQL. Los tipos de datos definidos por el usuario se pueden emplear para asegurar que un dato tenga las mismas características sobre múltiples tablas.

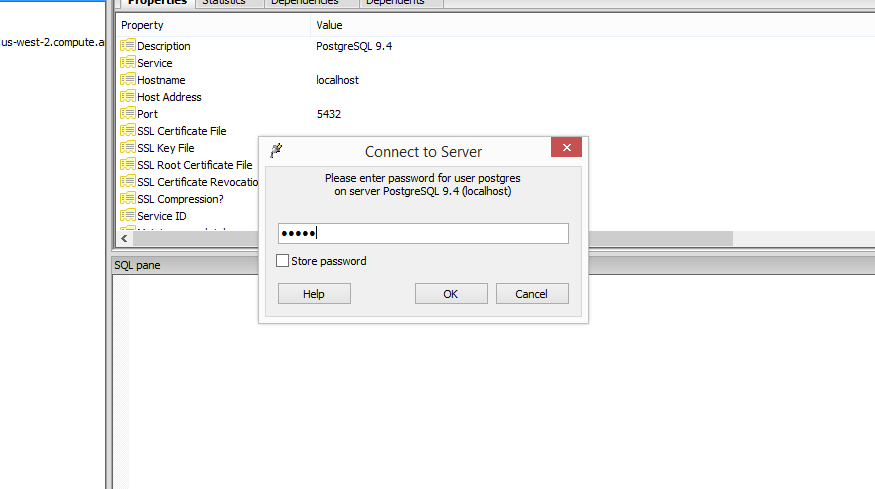
Crear, usar y modificar una BD en Postgresql

A continuación se van a explicar los pasos a realizar para construir y utilizar una base de datos sencilla.

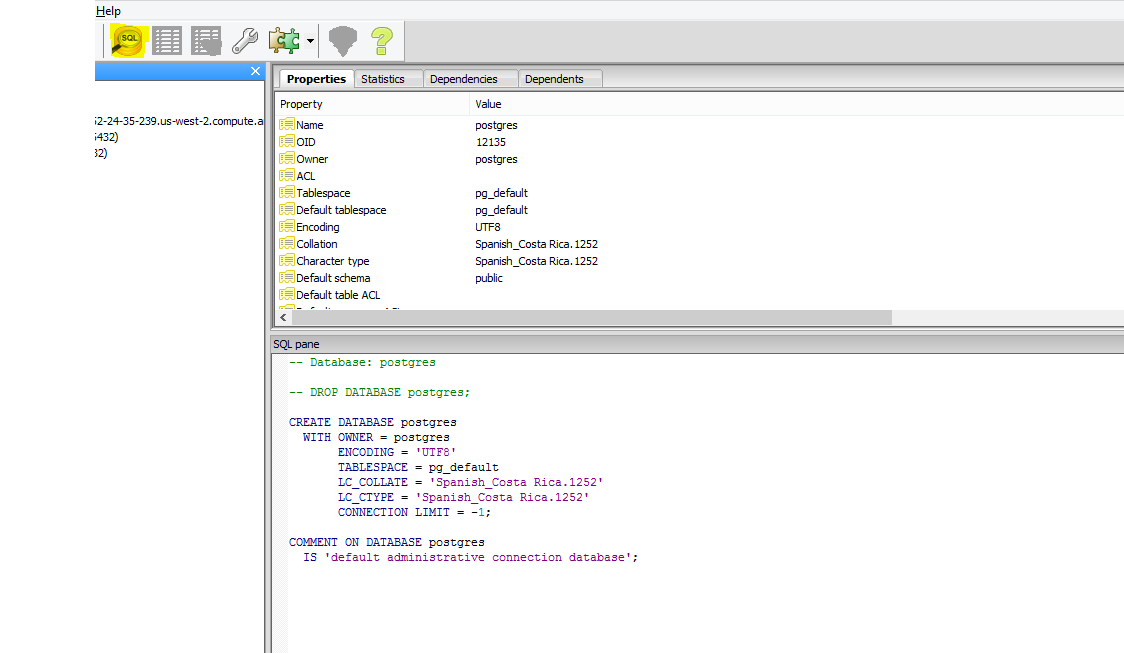
* **Paso 1.** Abrir el pgAdmin y conectarse como administrador



* **Paso 2.** Una vez conectado en la parte izquierda se muestran las BD alojadas en el servidor, por defecto las del sistema (PostgreSQL 9.X.) y otras si se han creado previamente.



* **Paso 3.** Abrir una consulta en la conexión actual



* **Paso 4.**Ejecutarlas siguientes instrucciones. Recordar que en el estándar la instrucción termina con punto y coma

--Instrucción para **crear** la base de datos

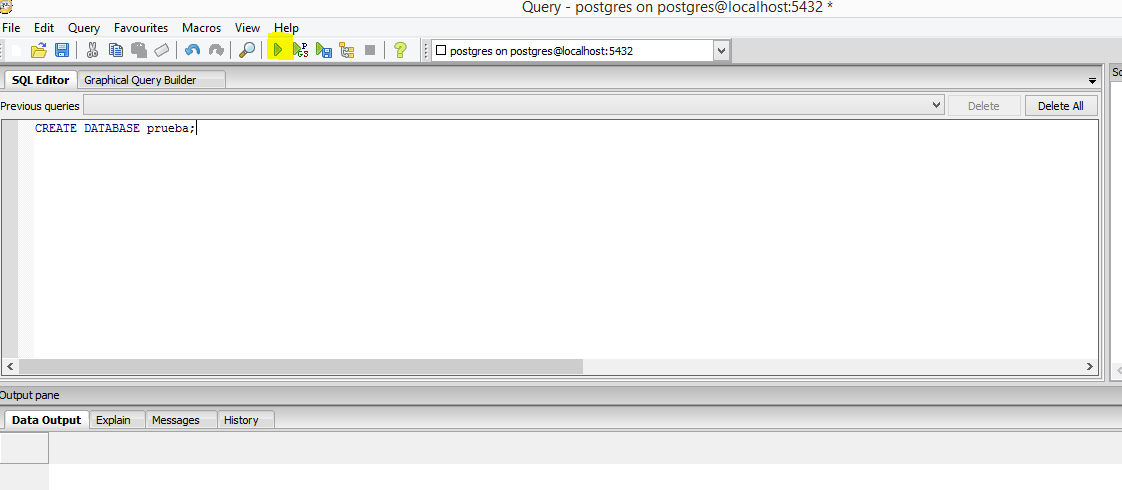
CREATE DATABASE prueba;

--Instrucción para **seleccionar** la base de datos a utilizar

SET SCHEMA 'prueba'; -- no olvidarse nunca de seleccionar la BD de trabajo

--Instrucción para **eliminar** una base de datos

DROP DATABASE prueba;



Crear, modificar y borrar una tabla en Postgresql

A continuación se van a explicar los pasos a realizar para construir, modificar y eliminar tablas en una base de datos.

* **Paso 1.** Crear una base de datos, o utilizar una creada previamente
* **Paso 2.** Abrir una consulta en la conexión actual
* **Paso 3.**Para crear una tabla ejecutar las siguientes instrucciones

CREATE TABLE persona

(

*--Si la llave es compuesta se indica así Primary Key (atributo1, atributo 2)*

cedula CHAR (11) NOT NULL PRIMARY KEY,

nombre VARCHAR (50) NOT NULL,

apellido1 VARCHAR (50) NOT NULL,

apellido2 VARCHAR (50) NOT NULL,

correo VARCHAR (100) NOT NULL,

telefono CHAR (9) NULL,

direccion VARCHAR (200) NOT NULL

) ;

* **Paso 4.** Para modificar una tabla se ejecutan las siguientes instrucciones

--Agregar una columna como llave primaria

ALTER TABLE persona ADD CONSTRAINT pk\_cedula\_persona PRIMARY KEY (cedula);

--Agregar una nueva columna a la tabla y que no pueda ser nula

ALTER TABLE persona ADD fecha\_nacimiento timestamp without time zone NOT NULL;

--Borrar una columna de una tabla

ALTER TABLE persona DROP COLUMN fecha\_nacimiento;

--Agregar una nueva columna a la tabla

ALTER TABLE persona ADD prueba CHAR (10) NOT NULL;

--Modificar el tamaño de la columna que acabamos de agregar

ALTER TABLE persona ALTER COLUMN prueba TYPE CHAR (12);

--Borrar la columna

ALTER TABLE persona DROP COLUMN prueba;

--Borrar la tabla

DROP TABLE persona;

* **Paso 5.**Cree las siguientes tablas en la base de datos

CREATE TABLE estudiante

(

cedula CHAR (11) NOT NULL,

carne INT NOT NULL,

nombre VARCHAR (50)

);

ALTER TABLE estudiante

ADD CONSTRAINT pk\_cedula\_estudiante PRIMARY KEY (cedula);

ALTER TABLE estudiante

ADD CONSTRAINT fk\_cedula\_estudiante FOREIGN KEY (cedula) REFERENCES persona;

ALTER TABLE estudiante

ADD CONSTRAINT uk\_carne\_estudiante UNIQUE (carne);

CREATE TABLE profesor

(

cedula CHAR (11) NOT NULL,

especialidad VARCHAR (50) NOT NULL

);

ALTER TABLE profesor

ADD CONSTRAINT pk\_cedula\_profesor PRIMARY KEY (cedula);

CREATE TABLE carrera

(

--**identity**(1,1) inicia el identificador en 1 y aumenta automáticamente el valor en 1

id\_carrera serial NOT NULL,

nombre VARCHAR (50) NOT NULL,

CONSTRAINT pk\_id\_carrera primary key (id\_carrera)

);

ALTER TABLE carrera

ADD CONSTRAINT pk\_id\_carrera PRIMARY KEY (id\_carrera);

* **Paso 6.**Complete la creación de las siguientes tablas. Agregue las llaves primarias y foráneas que sean necesarias

CREATE TABLE Programa (

id\_programa

fecha

estado

);

CREATE TABLE Curso (

codigo

cedula

id\_programa

nombre

creditos

tipo

periodo

fecha

estado

);

CREATE TABLE Contenido (

id\_contenido

id\_curso

contenido

);

CREATE TABLE Grupo (

id\_grupo

id\_curso

numero\_grupo

);

CREATE TABLE Grupo\_Estudiante (

id\_grupo

carne

nota

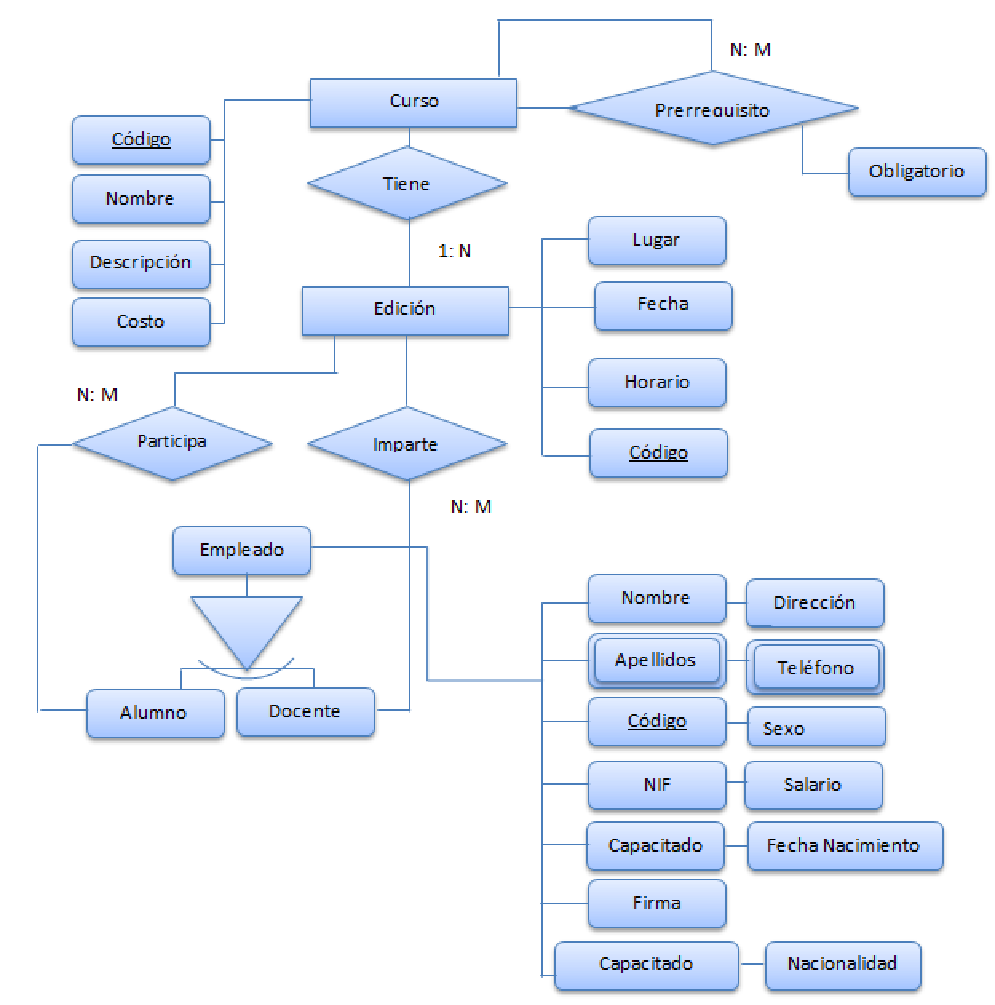
estado

);

* **Paso 7.**Realice los siguientes ejercicios.
  + Agregue a la tabla Programas un atributo descripción
  + Modifique el atributo especialidad de la tabla Profesores para que sea un varchar de 200 caracteres.
  + Agregue un atributo id\_carrera a la tabla Programas. Luego, convierta este atributo en una llave foránea a la tabla Carrera.
  + Agregue un atributo cupo a la tabla Grupo.
  + Elimine el atributo estado de la tabla grupo\_estudiante.
  + Cambie el atributo nombre de la tabla Estudiantes para que sea not null
  + Borre la tabla Contenidos

Modelo Relacional – Práctica

Recuerde que en el transcurso de las clases hemos trabajado con el siguiente modelo E-R.



A partir del modelo anterior generamos el siguiente modelo relacional. Con dicho modelo relacional, elabore la base de datos en Postgresql. Recuerde que debe definir las llaves primarias y foráneas necesarias. Así como, definir el tipo de datos adecuado para cada atributo.

**curso** (código, nombre, descripción, costo)

**prerrequisito** (id, codigo1, codigo2, obligatorio)

**edicion** (código, lugar, fecha, horario, codigo\_curso)

**empleado** (código, nombre, dirección, apellido1, apellido2, sexo, NIF, salario, capacitado, nacionalidad, fecha\_nacimiento, es\_alumno, es\_profesor)

**telefono** (id, código, teléfono)

**participa** (codigo\_edicion, codigo\_empleado)

**imparte** (codigo\_edicion, codigo\_empleado)