

# Planificación UF1469 – SGBD e instalación

Dr. Juan Pedro Cerro

- Estructura interna de una base de datos de Access así como de la interfaz de la aplicación. Partes de una BBDD de Access (\*.accdb)
- Definir una base de datos sencilla con los siguientes campos:

Tabla: Empleados		
Nombre campo	Formato	Regla de Validación
<i>Código empleado</i>		
<i>Nombre</i>	>@	
<i>Apellidos</i>		
<i>Dirección</i>		
<i>Teléfono</i>	000 000 000	
<i>Edad</i>	0 “Años”	<100 “Demasiado longevo”
<i>Casado</i>		
<i>Fecha nacimiento</i>	Fecha corta	
<i>Salario</i>	0,00” €”	>=0 “Ha de ser positivo”

- Dar de alta de 15 a 20 empleados.
- Consultas Básicas: Crear una consulta que sólo presente unos campos y ver su transcripción en SQL. Luego hacer...
  - **Nombre, Apellidos, Edad y Fecha de nacimiento** de un empleado en concreto dado su nombre.
  - **Nombre, Apellidos, Edad y Fecha de nacimiento** de un empleado cuyo nombre empieza por una letra.
  - **Nombre, Apellidos, Edad y Fecha de nacimiento** de un empleado cuya edad es inferior a 25 años.
  - **Nombre, Apellidos, Edad y Fecha de nacimiento** de un empleado que haya nacido antes del año 75.
  - **Nombre, Apellidos, Edad y Fecha de nacimiento** de un empleado que no esté casado.
- Consultas con más de un criterio:
  - **Nombre, Apellidos, Edad y Casado** de aquellos empleados que tengan más de 30 años y / o no estén casados.
  - **Nombre, Apellidos, Edad y Casado** de aquellos empleados que tengan entre 40 y 50 años.
  - **Nombre, Apellidos, Edad y Casado** de aquellos empleados que el campo observaciones esté vacío.
- Parametrización de campos tipo Texto, Lógico y Fecha/Hora.
- Campos calculados (Con los campos Nombre y Salario):
  - Listado de todos los empleados que dentro de 10 años habrán cumplido la edad que introducimos por parámetro.
- Consultas con Agrupaciones / Totales:
  - Suma de los salarios de todos los empleados mayores de 35 años.
  - Suma de los salarios de todos los empleados mayores de 35 años no-casados.
  - Suma todas las edades de aquellos empleados menores de 30 años agrupando por el campo casado.

- Salario más alto de los casados y no-casados respectivamente.
- Nombre del casado que menos cobra.
- Nombre y Apellidos del casado que cobra más que la diferencia máxima de salarios entre empleados (diferencia entre el que menos cobra y el que cobra más)
- Consultas de Acción:
  - Crear una tabla con Nombre, Apellidos, Salario y Casado de los que estén casados.
  - Añadir los registros de los que no estén casados a la tabla de antes.
  - Parametrizar ésta última consulta.
  - Actualizar la Edad multiplicándola por 2.
  - Efectuar una consulta de eliminación.
- Presentar la BBDD de Neptuno y explorar sus relaciones y tablas.
- Realizar el **ejercicio de evaluación continua 1**.
- Explicar el funcionamiento de la aplicación basada en tecnologías web phpmyadmin (MySQL)
  - Instalar XAMPP dentro de la máquina virtual Linux e intentar activarlo para acceder al phpmyadmin (/opt/lamp)
  - Crear una tabla de prueba (tabla1) dentro de la BBDD de prueba “test” con algunos campos, desde la propia interfaz gráfica.
  - Acceder a la estructura de la tabla para cambiar el tipo de los datos.
  - Luego insertar manualmente un registro.
  - Mostrar las opciones de exportación y exportarlo en formato JSON.
  - A continuación, ver el menú “Operaciones” estando dentro de la “tabla1”.
  - Luego ver el menú “Operaciones” estando dentro de la BBDD “test”.
- Instalar Wordpress y explicar le procedimiento para declarar una BBDD dónde almacenar los datos del sistema gestor de contenidos.
- Descargar el SGBDR MySQL de la web oficial e instalarlo. (Explicar todas las opciones de configuración – usar la *configuración avanzada*)
- Descargar e instalar también el MySQL Workbench.
- Ejecutar el MySQL Workbench y ver la configuración de la conexión que nos aparece por defecto: *Database > Manage Connections...*
- Volver a *Database > Manage Connections...* para editar la conexión si hiciera falta.
- Ver cómo da un error al acceder al “Server Status”, para solucionarlo ir a: Panel de Control > Reloj y Región > Región > Administrativo > Cambiar

☒ **Versión beta: Use UTF-8 Unicode para la compatibilidad de idioma en todo el mundo**



configuración regional del sistema >

- En el panel *Navigator* explicar genéricamente todas las opciones disponibles.
- En la parte inferior del *Navigator*, ir a los esquemas **Administration** **Schemas** y mostrar la BBDD junto con sus elementos de tipo organizativo:



- **Tablas:** Son los principales objetos en los que se almacenan los datos en filas y columnas.
- **Vistas:** Son consultas predefinidas que actúan como tablas virtuales.
- **Índices:** Mejoran la velocidad de consulta.
- **Procedimientos Almacenados y Funciones:** Código SQL almacenado que puede ser reutilizado y ejecutado para realizar tareas específicas.

- **Triggers (Disparadores):** Son procedimientos que se ejecutan automáticamente en respuesta a ciertos eventos en una tabla.
- **Eventos:** Tareas programadas que se ejecutan en momentos específicos.
- Crear un esquema de ejemplo llamado “bbdd” con una tabla de contactos:


Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
 idcontactos	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
 nombre	VARCHAR(60)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

- Abrir una nueva pestaña con una Query y usar los siguientes comandos:
 

```
use bdd;
insert into contactos (nombre) values ('JP1');
insert into contactos (nombre) values ('JP1'), ('JP2'), ('JP3');
select * from contactos;
delete from contactos where nombre like '%3';
```
- Ir a la sección de importación del panel de navegación e importar **neptuno.sql**. Luego examinar la importación y ver el contenido de algunas tablas.
- Ejecutar algunas de las sentencias SQL de la *evaluación continua 1* en esta BBDD importada.
- Intentar ahora acceder a la BBDD del servidor de un compañero/a del aula.
- Ahora explicar cómo conectar el lenguaje de programación JAVA con MySQL viendo la importancia de descargar el conector MySQL para Java de la web oficial e incorporarlo a BlueJ.
- Luego hacer un SCRIPT en Java que acceda a la tabla “pedidos” de **neptuno**. Descargar del campus la clase **SQL.java** para analizar su código explicando su contenido y hacer algunas pruebas.
- Modificar la clase anterior para que realice un bucle y ataque al servidor MySQL y poder así monitorizarlo. Usar este bucle:
- Ahora leer un fichero CSV para añadir el contenido del fichero “**contactos.csv**” a la tabla “**contactos**” de la base de datos **bbdd**.
- Hacer un script que añada los registros y luego los borre en bucle para monitorizar el rendimiento.
- Hacer el **ejercicio de evaluación continua 2**: Preguntar al usuario en java y añadir un registro. Usar como referencia el código que viene en el enunciado.
- Explicar como crear una BBDD MySQL en un servidor de computación en la nube:
  - En primer lugar, loguearnos en la web RAILWAY y crear un proyecto nuevo con una BBDD MySQL
  - Luego copiar todos los valores de la sección *variables* para crear una nueva conexión con el Workbench, y luego conectar.
  - Ejecutar el Script para crear la bdd **neptuno** remotamente.
  - Descargar el cliente SQL “**HeidiSQL**” y probar de acceder a la BBDD creada en RAILWAY.
  - Luego en Colab explicar cómo conectar MySQL con Python:
 

```
En Colab -> !pip install mysql-connector-python
En local -> pip install mysql-connector-python
```
  - Luego crear un ejemplo de conector para verificar que realmente conectamos con el servicio en la nube.
- Ahora hacer una consulta SQL y recuperar los datos para mostrarlos por consola.
- Dar de alta un registro usando Python.
- Finalmente, pasar el resultado a un dataframe para hacer alguna consulta con PANDAS.

- Ahora crear un Script en un fichero local (\*.py) y ejecutarlo desde la consola.
- Diferencia entre MySQL y PostgreSQL: <https://aws.amazon.com/es/compare/the-difference-between-mysql-vs-postgresql>
- Descargar PostgreSQL e instalarlo.

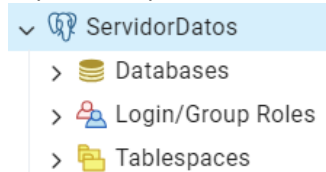
- Luego abrirlo y crear un servidor llamado “ServidorDatos” desde el :




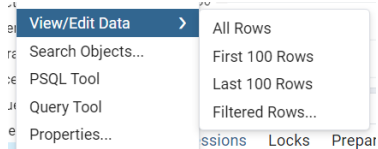
[Add New Server](#)

Hay que indicarle el nombre de la BBDD que se usará en el servidor para gestionarlo (por defecto: **postgres**)

- Explicar las partes del servidor en el árbol de la izquierda:

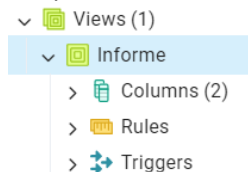


- Crear un usuario llamado “usuario” en **Login/Group Roles**
- Crear un Tablespaces llamado “destinoDatos” que apunte a `c:\datos`
- Crear una base de datos llamada **neptuno** usando el tablespaces “destinoDatos” y comprobar una vez creada cómo almacena los valores allí.
- Ir luego a la bbdd **neptuno** y con el botón derecho pulsar sobre “Grant Wizard” para asignar permisos de acceso al usuario.
- Abrir la “Query tool” para hacer consultas y abrir el fichero **categorias – productos.sql** para ejecutarlo.
- Luego ver el dashboard para observar la actividad: 
- Acceder dentro del árbol de la BBDD neptuno a Schema>public>Tables y ver las tablas. Abrir una de ellas pulsando el botón derecho del ratón:



- Abrir el editor SQL y hacer la siguiente consulta de ejemplo para demostrar que funciona:  

```
select nombreproducto, preciounidad from productos
  join categorias on productos.idcategoria=categorias.idcategoria
 where categorias.nombredecategoria like 'C%'
```
- Después, ir a la sección “Views” y crear un informe con la consulta de antes:



- Ir a la sección de “Sequences” y en las propiedades explicar la serie incremental de dicho campo:

✓ 1..3 Sequences (1)

1..3 categorias\_idcategoria\_seq