

Modelo de datos

Persistencia, Archivos, Bases de Datos, SQL

Resumen de hoy

- Concepto de Persistencia
 - Estructurada
 - No-estructurada
- Modelo de Datos
 - Diagrama E/R
- Métodos de persistencia
 - Archivos (JSON, XML)
 - Crear Tablas
- Primer vistazo a SQL
- LTs

Persistencia

Recordemos las actividades de la organización

Persistencia “es la acción de **preservar** la **información** de un objeto de forma permanente (por ej, un disco rigido), y a su vez poder **recuperar** la misma (leerlo) para que pueda ser nuevamente utilizado”

- Una de las actividades de una organización

Estructura general de los datos

En general, es usual representar los datos con Tablas/Grillas

10:13 AM 100 %

[< Back](#) Editing Change Mode

Id	Priority	From	Sent	Size	Hours Active	
20	Normal	Laura Callahan	5/8/2019	294 B	<div><div></div></div>	<input type="checkbox"/>
21	BelowNormal	Andrew Fuller	4/18/2019	1147 B	<div><div></div></div>	<input type="checkbox"/>
22	AboveNormal	Margaret Peacock	4/29/2019	1280 B	<div><div></div></div>	<input checked="" type="checkbox"/>
23	High	Andrew Fuller	5/7/2019	2477 B	<div><div></div></div>	<input checked="" type="checkbox"/>
24	Low	Robert King	5/20/2019	2721 B	<div><div></div></div>	<input type="checkbox"/>

En el front se usan distintos estilos para **visualizar**
(los charlaremos mas adelante)

Mecanismos de hacer persistencia

- Archivos
 - binarios / Texto
 - JSON / XML
- Bases de datos
 - Estructuradas
 - No-estructuradas
- Ubicación
 - Local
 - En la nube

Representación

Los datos representan partes del **problema**

Cada sistema, tendrá diferentes datos a ser usados, aunque muchos veces sean similares

Sistema de Reserva tipo AirBnB-

Usuarios, Habitaciones, locatario, Reserva

Sistema de Alquiler de Autos-

Usuarios, Vehiculos, empresa, Alquiler

Acciones sobre los datos

Cualquier sistema, independiente del tipo, va a tener que realizar operaciones sobre esos datos

- **A**ltas o *creacion*
- **B**ajas o *eliminar*
- **M**odificaciones o *actualizaciones*
- **C**onsultas

An isometric illustration on a green background depicting a cloud computing environment. In the center is a stack of three dark grey server racks with a white cloud icon on top. Dashed white lines radiate from this central stack to various elements: a laptop on the left with a woman in an orange shirt standing next to it; a smartphone at the top left; a bar chart with three bars (blue, light blue, grey) at the top center; a man in a light blue sweater standing near the top center; a laptop on the right with a man in a purple sweater standing next to it; a woman in an orange shirt standing near the bottom right; a large monitor at the bottom center; and a smartphone at the bottom left. A magnifying glass is positioned near the bottom center. A small white cloud is on the ground near the bottom right. The overall theme is digital data storage and access.

Representación de los datos

Pensemos en un caso

Un cliente quiere que desarrollemos un sistema tipo “UBER” para Tandil.

El sistema debe permitir <- *requerimientos funcionales* :

- Que los usuarios pidan hacer un viaje de un origen a un destino
- Que los usuarios puedan elegir entre una lista de los conductores disponibles
- Que los usuarios puedan registrarse y pedir un viaje

POR DONDE ARRANCAMOS??

Implementando funcionalidad (I)

- Que los usuarios pidan hacer un viaje de un origen a un destino
 - Que los usuarios puedan elegir entre una lista de los conductores disponibles
 - Que los usuarios puedan registrarse y pedir un viaje

Pongamos datos

CONDUCTOR	ORIGEN	DESTINO	USUARIO	VEHICULO	FECHA
JUAN	Colon 1000	Rodriguez 200	MARIA	FORD KA	11:05 - 10/Ag
LUISA	Alsina 200	San Martin 1500	SANTIAGO	VW GOL	11:15 - 10/Ag

Implementando funcionalidad (II)

- Que los usuarios pidan hacer un viaje de un origen a un destino
- Que los usuarios puedan elegir entre una lista de los conductores disponibles
- Que los usuarios puedan registrarse y pedir un viaje

Pongamos datos

CONDUCTOR	ORIGEN	DESTINO	USUARIO	VEHICULO	FECHA	ESTADO
JUAN	Colon 1000	Rodriguez 200	MARIA	FORD KA	11:05 - 10/Ag	Ocupado
LUISA	Alsina 200	San Martin 1500	SANTIAGO	VW GOL	11:15 - 10/Ag	Ocupado

Implementando funcionalidad (III)

- Que los usuarios pidan hacer un viaje de un origen a un destino
- Que los usuarios puedan elegir entre una lista de los conductores disponibles
- Que los usuarios puedan registrarse y pedir un viaje

Pongamos datos

Nuevo usuario : Santiago

CONDUCTOR	ORIGEN	DESTINO	USUARIO	VEHICULO	FECHA	ESTADO
JUAN	Colon 1000	Rodriguez 200	MARIA	FORD KA	11:05 - 10/Ag	Ocupado
JUAN	Colon 1000	Rodriguez 200	BELEN	FORD KA	11:05 - 10/Ag	Ocupado
LUISA	Alsina 200	San Martin 1500	SANTIAGO	VW GOL	11:15 - 10/Ag	Ocupado
			Santiago..			

Que problemas detectamos??

- Posibles datos repetidos
- Una sola estructura difícil de leer

SEPAREMOS EN TABLAS

Revisamos el caso


Un cliente quiere que desarrollemos un sistema tipo “UBER” para Tandil.

El sistema debe permitir :


- Que los **usuarios** pidan hacer un **viaje** de un origen a un destino
- Que los **usuarios** puedan elegir entre una lista de los **conductores** disponibles
- Que los usuarios puedan registrarse y pedir un **viaje**

Modelado de datos


Las etapas para **representar** la información en el mundo computacional:



1.identificación de los datos de la realidad: actores, recursos, objetos, etc. del mundo real de los cuales interesa guardar *información*.



2.identificación de relaciones entre datos: detección de los vínculos significativos que se dan entre los elementos.



3. abstracción de datos y relaciones: representación simbólica de los elementos detectados.

APLIQUEMOS EN EL CASO!!!

Tablas para el Sistema

- Viaje
 - Origen
 - Destino
 - Hora
 - Precio
 - Id_usuario / id_conductor
- Usuario
 - Nombre
 - direccion
- Conductor
 - nombre
 - vehiculo

Datos

¿Qué es un dato?

Es el estado que toma un atributo. .

- Se llama **CAMPO** a cada atributo de la tabla. Cantidad y precio son campos.
- Se llama **REGISTRO** al conjunto de CAMPOS que definen un elemento de la tabla. La categoría 1 (ID_Categoría) llamada libro (Nombre) de libros impresos en color (Descripción) es un registro donde cada CAMPO tiene su valor correspondiente

Tipos de Datos

¿Qué tipos de datos hay?

Enteros, flotantes, textos, caracteres, fecha, booleano, bit, y más...

¿Cuál es la diferencia?

Depende de lo que se quiera representar...

Cada tipo de datos ocupa cierto espacio en memoria (primaria y secundaria)

- INT: 4 bytes (2^{32})
- TINYINT: 1 byte (2^8)
- BIGINT: 8 bytes (2^{64})
- DATE: 3 bytes
- TEXT: longitud + 2 bytes

Registros

Un archivo en general es una **colección de registros** con una cantidad de campos definida.

	Dominio String (6)	Marca String (20)	Modelo int	
	AEX123	Chevrolet	1995	
	FDS256	Ford	2006	
	

Los archivos se pueden ordenar por algún campo o por varios.

Cada registro debe tener un **IDENTIFICADOR**

Modelado de datos - MER

MER (Modelo de Entidad-Relación) es un modelo semántico que describe los requerimientos de datos de un sistema.

Elementos del MER:

- **Entidad:** objeto real o abstracto de la vida real del cual se quiere almacenar información
- **Relación:** asociación entre entidades
- **Atributos:** características que describen las entidades y relaciones

Tablas para el Sistema

- Viaje
 - id_viaje
 - Origen
 - Destino
 - Hora
 - Precio
 - Id_usuario / id_conductor
- Usuario
 - id_usuario
 - nombre
- Conductor
 - id_conductor
 - vehiculo

Ejemplo con datos

Tabla Viaje

ID_VIAJE	ORIGEN	DESTINO	PRECIO	ID_CONDUCTOR
1	Salta 300	Pinto 399	70	1
2	Campus	San Martin 1080	150	2
3		

Tabla Conductor

ID_CONDUCTOR	Nombre	Vehiculo	Puntuaciones positivas
1	Juan Perez	Corsa Blanco	20
2	Diego Garcia	Peugeot 304	3



Almacenamiento en Archivos

Archivos (I)

- Miles de formatos y estructuras
 - binarios (imagenes, video, sonido, otros)
 - Texto (jSon, CSV, TXT)
- Se almacena cualquier tipo de información, en general desorganizada
- Fácil de usar (nativo del SO)

Archivos (II)

¿Cómo abrir y leer un archivo en PHP?

`fopen(ruta,modo)` permite abrir un archivo local o remoto (URL).

Modos:

r -> Sólo lectura

r+ o w+ -> Lectura y escritura

w -> Sólo escritura

a -> Sólo escritura (añadiendo al final)

a+ -> Lectura y escritura (añadiendo al final)

```
<?php
```

```
$path = "/home/user/datos.txt";
```

```
$file = fopen($path, "r");
```

```
while (!feof($file))
```

```
    echo (fgets($file));
```

```
fclose($file);
```

```
?>
```

Ejemplo de Archivo

```
{
  "travels":
    {
      "travel":
        [
          {
            "id": "0001",
            "posLat": -53.4,
            "posLong" : -37.4
            "client":
              {
                "id": "001" , "name" : "Ana Garcia"
              }
            "price": 10.55,
            "time" : 0,
            "driver":
              {
                "id": "001" , "name" : "Juan Perez"
              }
          }
        ]
    }
}
```

Que sucede cuantos tenemos millones de registros???

Archivos (III)

Problemas al usar archivos JSON...

- Implementación ineficiente
- Archivos poco estructurados
- Complejidad al leer y escribir datos
- Uso de tipo de datos ineficiente



TIP:

Usemos archivos JSON/XML/TXT cuando tenemos pocos accesos de **lectura**(por ej:logging).



Bases de datos

Bases de Datos

¿Qué es una Base de Datos?

- una herramienta para recopilar y organizar información
- un contenedor de objetos para almacenar tablas que guardan datos interrelacionados

Ejemplo: sistema de stock de artículos va a tener al menos las tablas artículo y categoría.

¿Cuándo usar una Base de Datos?

- La cantidad de datos es excesiva.
- Redundancia: gran duplicidad en los archivos. Una sola referencia evitando que se dupliquen. Esto evita:
 1. se debe actualizar cada una de las fuentes donde se encuentre la info.
 2. gasto innecesario de almacenamiento.
 3. por seguridad, se crean esquemas de acceso a los datos.

Independencia entre datos y aplicaciones

en sistemas de archivos cada aplicación tiene su propio archivo de datos

- control de restricciones del DBMS
- compartir datos para lectura/escritura
- consistencia en transacciones, cuando no se realiza en su totalidad, queda todo como estaba.

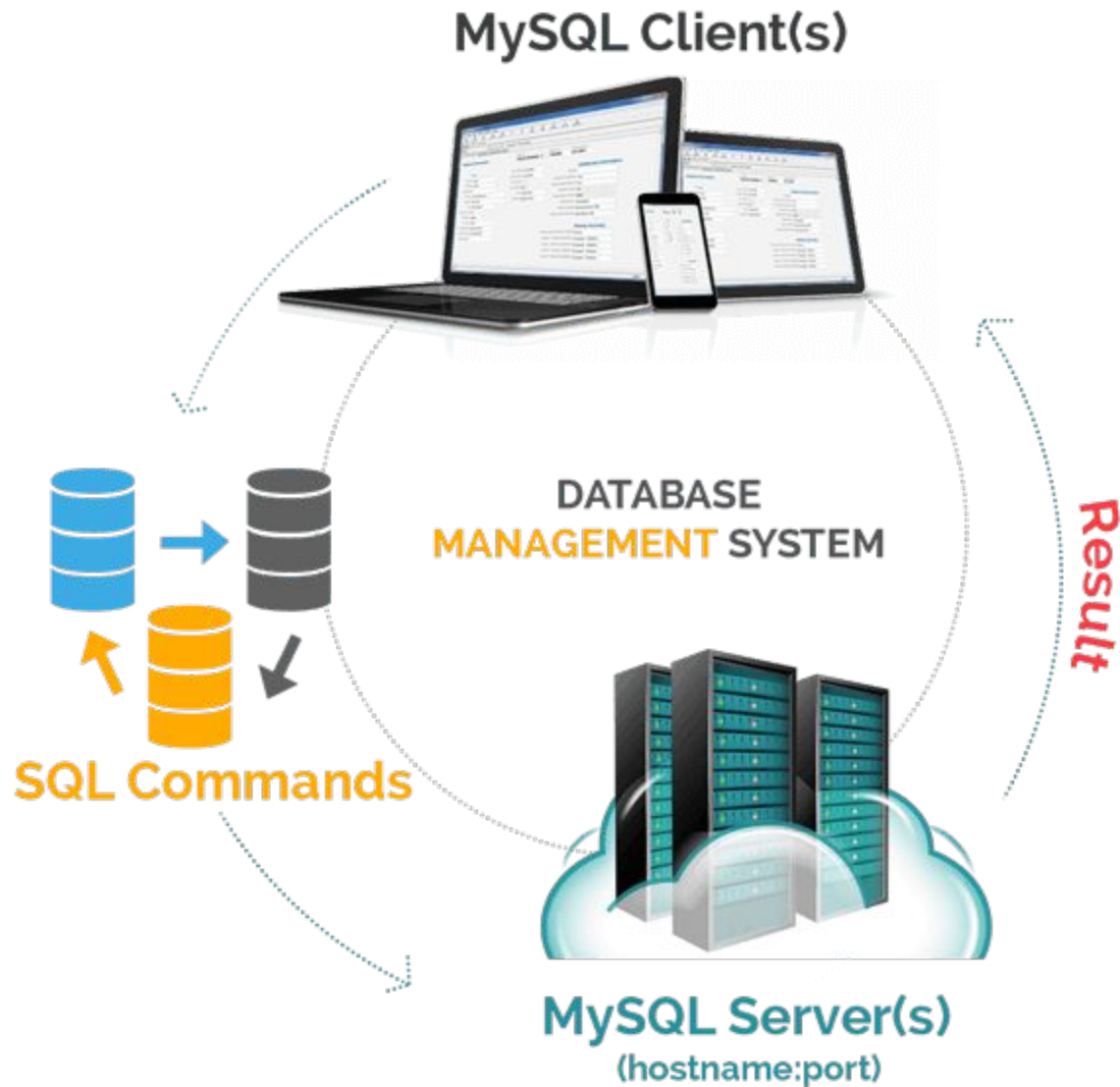
DBMS (Gestores de bases de datos)

- software que permite al usuario definir, crear, configurar y mantener la base de datos.
- manejo de persistencia de los datos
- control de acceso
- evita inconsistencias
- recuperación antes fallas

Ventajas de los DBMS

- Independencia de los datos: oculta detalles de cómo se almacenan
- control de redundancia

Secuencia de Uso



Bases de Datos

- restricción de accesos a usuarios autorizados
- mejora la integridad de los datos
- aumento de la concurrencia
- servicio de backup y recuperación ante fallas

Desventajas de los DBMS

- Complejidad
- Tamaño
- Vulnerabilidad a fallas al ser centralizados
- Costo y equipamiento necesario

Bases de Datos

Algunos gestores de bases de datos:

MySQL

Oracle

SQL Server

Access

PostgreSQL

...



Uls para parametrizar las BBDD

phpMyAdmin

 *Adminer*



QueryPie

Cómo usamos las BASES de DATOS en WEB 2

Utilizando **SQL (Structured Query language)**.

Funciona en todos los gestores de BBDD.

Nos permite:

- dar de alta, borrar y modificar datos (ABM)
- consultar información ya guardada



SQL - Structured Query Language

En un lenguaje de **consulta estructurado**

- Permite crear y modificar esquemas, tablas e índices
- Consultar facilmente datos
- Definir tablas
- Insertar, borrar y actualizar muchos datos
- Buscar muchos datos en poco tiempo

“Quiero saber todos los viajes que Juan hizo ayer”

Como seria en SQL??

```
Select * from VIAJES where CONDUCTOR = “JUAN”  
and DATE = 30/08/2020
```

“Quiero saber el conductor que más viajes hizo”

Como seria en SQL??

..... Los vemos la semana que viene

Sentencias SQL

Seleccionando algunos registros de una tabla, ordenados por algún campo:

SELECT <atributo/s> **FROM** <tabla> **WHERE** <cond.>
ORDER BY <atributo/s>;

```
SELECT Descripcion, Cantidad FROM `sistema`.`articulo`  
WHERE Precio > 97.5 ORDER BY ID_Articulo ASC LIMIT 5;
```

```
SELECT Precio, StockMinimo FROM `sistema`.`articulo`  
WHERE Descripcion LIKE '_o%o';
```

```
CREATE TABLE `Viaje` (  
  `origen-destino` <text>,  
  `id_usuario` <int>,  
  `id_conductor` <int>,  
  `fecha-hora` <date>,  
  `precio` <float>  
);
```

Sentencias SQL

Insertando registros en una tabla sin y con condiciones:

INSERT INTO <tabla> (campo/s) **VALUES** (valor/es);

INSERT INTO `sistema`.`articulo` (`ID_Categoria`, `Descripcion`, `Cantidad`,
`Precio`, `StockMinimo`) **VALUES** (`2`, 'Pronto', 500, `13.56`, `10`);

INSERT INTO `sistema`.`articulo` (`ID_Categoria`)
SELECT ID_Categoria **FROM** `sistema`.`categoria`
WHERE Nombre = 'Libros';

INSERT INTO `sistema`.`categoria` (`ID_Categoria`, `Nombre`,
`Descripcion`) **VALUES** ('1', 'Libro', 'Libros impresos en color'), ('2', 'Revista',
'Variedad de revistas');

Sentencias SQL

Seleccionando registros de una tabla aplicando funciones y agrupándolos

```
SELECT COUNT (*) AS "Cantidad de articulos"  
FROM `sistema`.`articulo` GROUP BY ID_Categoria;
```

```
SELECT COUNT (*) FROM `sistema`.`articulo` WHERE ID_Categoria = 2
```

```
;
```

```
SELECT Descripcion SUM/MIN/MAX/STDDEV (Precio) AS  
"Suma/min/max/DesvSTD de precio"  
FROM `sistema`.`articulo` GROUP BY ID_Categoria;
```

Sentencias SQL

Actualizando valores de campos en una tabla:

UPDATE <tabla> **SET** (valor/es) **WHERE** (cond.);

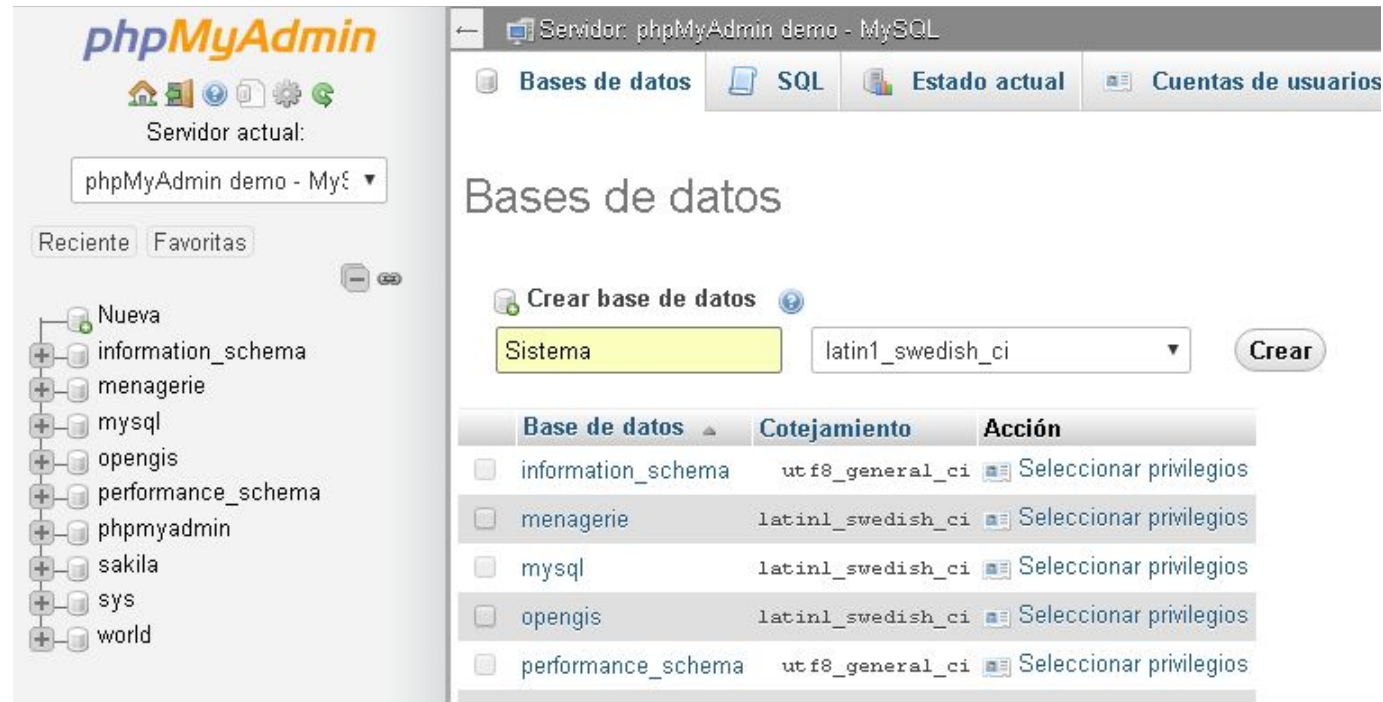
```
UPDATE `sistema`.`articulo` SET Precio = Precio * 1.1  
WHERE ID_Categoria = 2;
```

Borrando registros de una tabla:

DELETE FROM <tabla> **WHERE** (cond.);

```
DELETE FROM `sistema`.`articulo`  
WHERE ID_Categoria = 2 AND Cantidad < 7;
```


Ejemplo en la Practica



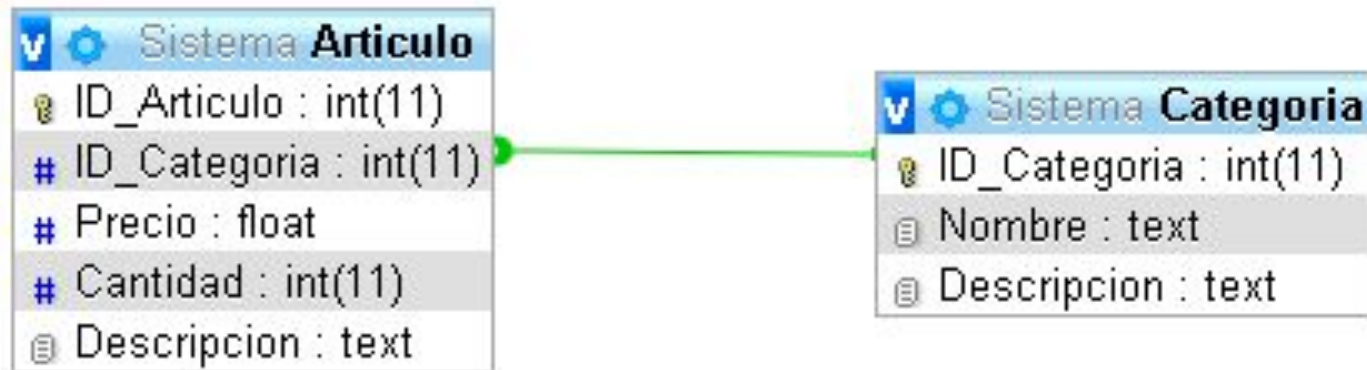
Usemos PhpMyAdmin...

<http://localhost/phpmyadmin/>

Modelado de datos - claves

Claves...

- **Primaria (PK):** identifica unívocamente a la instancia de la entidad, por ejemplo el CUIL o DNI
 - **Autoincremental (AI):** genera numeros automáticamente, asegura la no repetición, se llama clave surrogate (ficticia)
- **Foránea (FK):** clave que coincide con la clave primaria de otra entidad.



Ejemplo

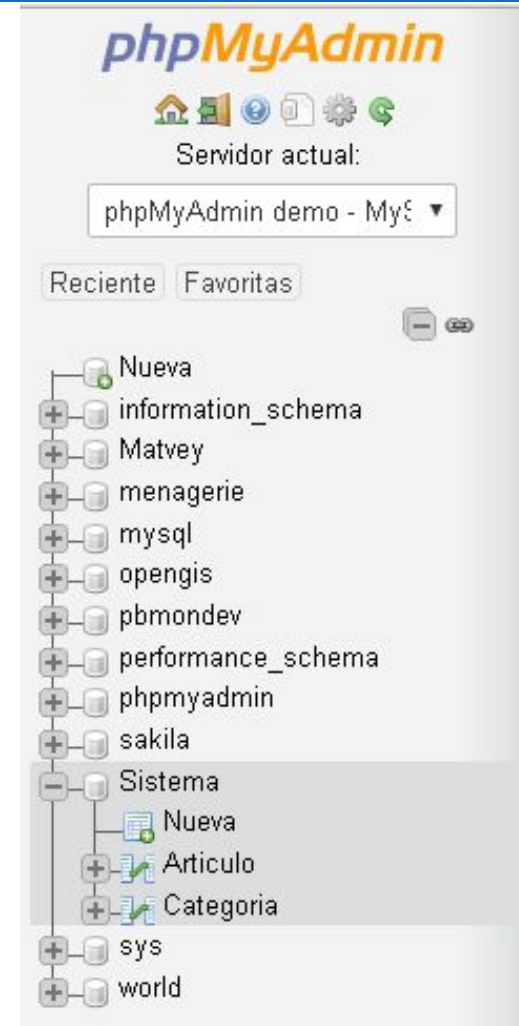
Como crear las tablas

SQL:

```
CREATE TABLE `Sistema`.`Articulo` ( `ID_Articulo` INT NOT NULL , `ID_Categoria` INT NOT NULL , `Precio` FLOAT NOT NULL , `Cantidad` INT NOT NULL , `Descripcion` TEXT NOT NULL ) ENGINE = InnoDB;
```

```
CREATE TABLE `Sistema`.`Categoria` ( `ID_Categoria` INT NOT NULL , `Nombre` TEXT NOT NULL , `Descripcion` TEXT NOT NULL ) ENGINE = InnoDB;
```

```
ALTER TABLE `Articulo` ADD PRIMARY KEY(`ID_Articulo`);
```



Ejemplo

Como crear las claves foráneas

phpMyAdmin

Servidor actual: phpMyAdmin demo - MySQL

Reciente Favoritas

Nueva
a0126837_moiblog
erdill
information_schema
menagerie
mysql
opengis
performance_schema
phpmyadmin
rtd
sakila
Sistema
Nueva
Articulo
Categoria
sys
world
Operaciones con base de datos a0126

Servidor: phpMyAdmin demo - MySQL > Base de datos: Sistema > Tabla: Articulo

Examinar Estructura SQL Buscar Insertar Exportar Importar Privilegios Operaciones Más

[Editar en línea] [Editar] [Crear código PHP]

Estructura de tabla Vista de relaciones

Restricciones de clave foránea

Acciones	Propiedades de la restricción	Columna	Restricción de clave foránea (INNODB)		
			Base de datos	Tabla	Columna
Eliminar	FK_ID_Categoria	ID_Categoria	Sistema	Categoria	ID_Categoria
	ON DELETE RESTRICT	+ Añadir columna			
	ON UPDATE RESTRICT				
	Nombre de la restricción		Sistema		
	ON DELETE RESTRICT	+ Añadir columna			

Consola

Favoritos Opciones Historial Limpiar

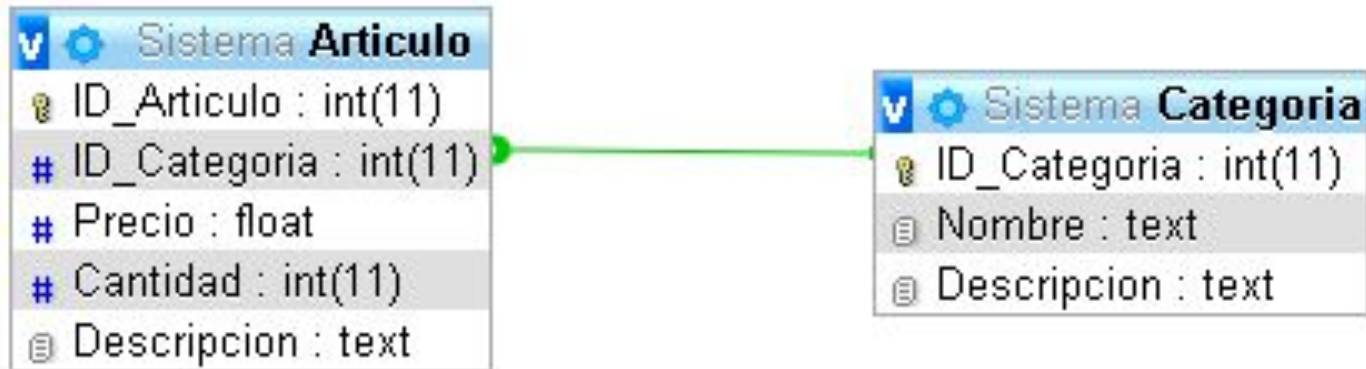
Presione Ctrl+Enter para ejecutar la consulta

```
> ALTER TABLE `Articulo` DROP INDEX `ID_Categoria_2`;  
> ALTER TABLE `Articulo` DROP INDEX `ID_Categoria`;  
> CHECK TABLE `Articulo`  
> SELECT * FROM `Articulo`  
> SELECT * FROM `Articulo`
```

SQL:

```
ALTER TABLE `Articulo` ADD CONSTRAINT `FK_ID_Categoria` FOREIGN KEY  
(`ID_Categoria`) REFERENCES `Categoria`(`ID_Categoria`) ON DELETE RESTRICT  
ON UPDATE RESTRICT;
```

Ejemplo



SQL:

```
ALTER TABLE `Articulo` ADD `StockMinimo` INT NOT NULL ;
```

Conclusiones I

Los **archivos** son una manera clásica y **poco costosa** (al inicio) de almacenar información, pero hay una serie de **inconvenientes** que tiene que ver con:

- Redundancia e inconsistencia de los datos,
- Aislamiento de los datos,
- Problemas de acceso,
- Atomicidad, cuando una operación no se pudo realizar se debe poder volver al estado anterior.
- Integridad
- Acceso concurrente y
- Problemas de seguridad

Conclusiones II

Las bases de datos aparecieron como una solución a estas cuestiones, pero...

- La instalación puede ser costosa
- Necesitan personal capacitado
- La implementación y puesta en marcha puede ser larga
- Poco rentable a corto plazo

SQL nos permite trabajar con una base de datos de manera estructurada siendo un lenguaje de alto nivel muy eficiente.

Por naturaleza, hay cosas que van en Bases de Datos y otras en archivos, por ejemplo la conexión a la base.

Referencias

1. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (Learning Php, Mysql, Javascript, Css & Html5), Robin Nixon, O'Reilly Media, 2014
2. Database Management Systems, P.S. Gill, Publisher I. K. International Pvt Ltd, 2010.
3. Entity-Relationship Modeling: Foundations of Database Technology, Bernhard Thalheim, Springer Science & Business Media, 2013

Info adicional

Archivos - Acceso

Acceso al disco rígido

Costo = tiempo de búsqueda + tiempo de transferencia de bloque + retardo rotacional

Entonces...

Leer bloques consecutivos, es más eficiente!!!

Es prioritario minimizar el número de accesos a disco cuando se trabaja con volúmenes grandes de información.

Método de Acceso: modo de alojar/localizar los registros en un archivo

1. Secuencial o lineal: acceso a un archivo según el orden de almacenamiento de los registros. El acceso a un registro implica haber pasado por los que lo preceden en el orden físico.
2. Asociativo o directo: acceso a un determinado registro por la dirección obtenida a partir del valor de una clave de búsqueda. No implica el acceso a los registros precedentes

Archivos - Niveles de memoria

Por razones económicas, se suelen tener **niveles** de memoria.

- Registros y caches (más rápidos y caros).
- Discos rígidos y cintas (más baratos y lentos)
- Un registro de carga puede tomar una fracción de un **nanosegundo (picosegundo)** y un acceso a la memoria interna toma varios **nanosegundos**.
- La latencia de acceso a los datos en un disco es de varios **milisegundos** (un millón de veces más lento).

Archivos - Organización

Desordenados: el costo está asociado en el instante de buscar información. La inserción es rápida (al final), el acceso es asociativo (campo de búsqueda) o secuencial en cierto orden. Para borrar hay que buscar el registro linealmente.



Ordenados: costo asociado de catalogar los datos al momento de ingresarlos, pero en la búsqueda son más eficientes. Para borrar: primero buscar, marcar y luego reorganizar.

Archivos - Organización

Dispersos: el direccionamiento es el resultado de una relación entre un campo y su ubicación. (Hashing).

Son eficientes para buscar elementos concretos.



Arborescentes: para almacenar grandes cantidades de datos, especialmente ordenados. Permiten búsquedas eficientes por rangos.

Índices

- Estructura auxiliar para localizar registros en un archivo por caminos alternativos.
- Es un archivo especial que contiene un campo de indexación y un puntero.
- Permiten un acceso rápido y eficiente a los registros de un archivo.
- Existen de clave única y multiclaves
- Pueden haber varios índices por diferentes campos.



¿Cuándo usar índices?

al aumentar el volumen de datos

pero...

- agrega complejidad el mantenimiento
- requiere almacenamiento auxiliar
- puede ser muy grande para entrar en memoria principal.