

Persistencia, Bases de Datos & SQL



Que vamos a aprender HOY?

- Persistencia y representación
- Modelo de Datos
- Consultar una BBDD

Recordemos

Se crean sistemas para cubrir una **necesidad** o un **objetivo** de una organización o individuo.

Los sistemas se utilizan para:

- ▶ tomar decisiones,
- ▶ controlar operaciones,
- ▶ analizar problemas y facilitar actividades,
- ▶ brindar productos o servicios

► Sistema de Información

Recopilación: captura o recolecta datos.

Almacenamiento/Persistencia: guardar de forma estructurada.

Procesamiento: convierte esos en una forma más significativa.

Distribución: transferir o mostrar la información procesada a las personas que la usarán.

Persistencia

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

Estructura general de los datos

En general, es usual mostrar 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>	<input type="checkbox"/>
21	BelowNormal	Andrew Fuller	4/18/2019	1147 B	<div></div>	<input type="checkbox"/>
22	AboveNormal	Margaret Peacock	4/29/2019	1280 B	<div></div>	<input checked="" type="checkbox"/>
23	High	Andrew Fuller	5/7/2019	2477 B	<div></div>	<input checked="" type="checkbox"/>
24	Low	Robert King	5/20/2019	2721 B	<div></div>	<input type="checkbox"/>

Cómo representamos datos?

```
[
  {
    "Number": "SO43659",
    "Date": "2011-05-31T00:00:00",
    "AccountNumber": "AW29825",
    "Price": 59.99,
    "Quantity": 1
  },
  {
    "Number": "SO43661",
    "Date": "2011-06-01T00:00:00",
    "AccountNumber": "AW73565",
    "Price": 24.99,
    "Quantity": 3
  }
]
```

JSON

Registros

NUMBER	DATE	ACCOUNT_ NUMBER	PRICE	QUANTITY
SO43659	"2011-05-31"	AW29825	59.99	1
SO43661	"2011-06-01"	AW73565	24.99	3

JSON

- ▶ Fácil de leer
- ▶ Puede tener información no estructurada

REGISTROS

- ▶ Fácil de manipular
- ▶ Ordenar/buscar eficiente
- ▶ Ocupan menos espacio

Donde manejamos los datos?



```
[
  {
    "Number": "SO43659",
    "Date": "2011-05-31T00:00:00",
    "AccountNumber": "AW29825",
    "Price": 59.99,
    "Quantity": 1
  },
  {
    "Number": "SO43661",
    "Date": "2011-06-01T00:00:00",
    "AccountNumber": "AW73565",
    "Price": 24.99,
    "Quantity": 3
  }
]
```

NUMBER	DATE	ACCOUNT_NUMBER	PRICE	QUANTITY
SO43659	"2011-05-31"	AW29825	59.99	1
SO43661	"2011-06-01"	AW73565	24.99	3



MODELAR DATOS DE UN SISTEMA

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

El sistema debe permitir: *(requerimientos funcionales)*

- *Registrar un viaje de un origen a un destino con un conductor.*

Información adicional:

* Cada conductor es dueño de su vehículo

POR DONDE ARRANCAMOS??

Implementando funcionalidad (I)


- ▶ Registrar un viaje de un origen a un destino con un conductor

Implementando funcionalidad (I)

- ▶ Registrar un **viaje** de un **origen** a un **destino** con un **conductor**

Pongamos algunos datos...

CONDUCTOR	ORIGEN	DESTINO	VEHICULO	FECHA	Precio
Juan	Colon 1000	Rodriguez 200	FORD KA	11:05 - 10/Ag	70
Diego	Alsina 200	San Martin 1500	VW GOL	11:15 - 10/Ag	150



CONDUCTOR	ORIGEN	DESTINO	VEHICULO	FECHA	Precio	Medio de pago
Juan	Colon 1000	Rodriguez 200	FORD KA	11:05 - 10/Ag	70	..
Diego	Alsina 200	San Martin 1500	VW GOL	11:15 - 10/Ag	150

Qué problemas vimos?

- Datos repetidos
- Una sola estructura difícil de leer

SEPAREMOS EN DOS TABLAS

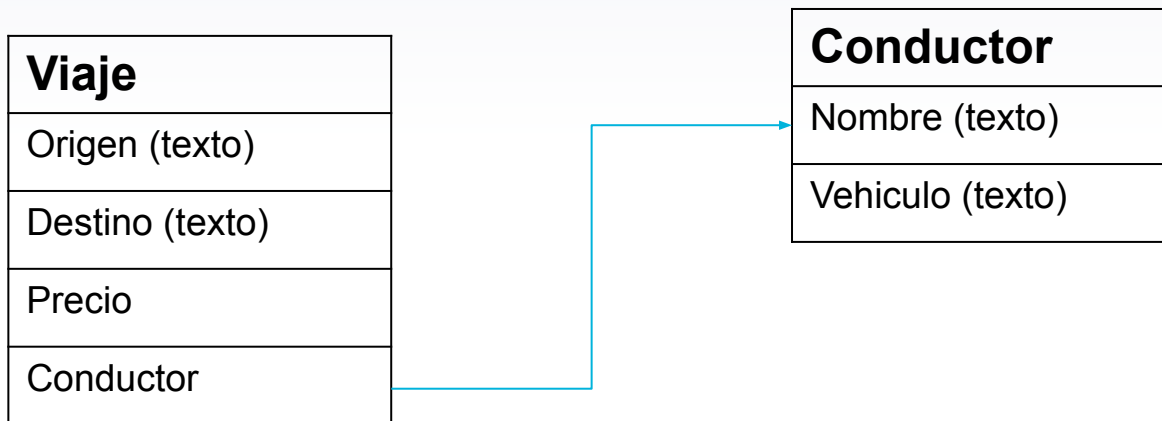
Tabla Viaje

ORIGEN	DESTINO	PRECIO	Conductor
Salta 300	Pinto 399	70	Juan
Campus	San Martin 1080	150	Diego
..	..		

Tabla Conductor

Nombre	Vehiculo
Juan	Corsa Blanco
Diego	Peugeot 304

► Entidad - Relación



Tablas

- VIAJE
 - origen
 - destino
 - hora
 - precio
 - conductor
- CONDUCTOR
 - nombre
 - vehiculo

Podemos tener datos repetidos!

- Juan Perez

Identificadores únicos

- VIAJE
 - **id**
 - origen
 - destino
 - hora
 - precio
 - *id_conductor*
- CONDUCTOR
 - **id**
 - nombre
 - vehiculo

Agregamos claves primarias ID (o *id_tabla*) para identificar de manera única cada registro.




Tabla Viaje

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

Tabla Conductor

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

Ejemplo Extendido, planilla: <https://n9.cl/clpux>

Conductor	Origen	Destino	Usuario	Vehiculo	Modelo	Fecha	Estado	Distancia
JUAN	Colon 1000	Rodriguez 200	MARIA	FORD KA	2012	11:05 - 10:Ag	Ocupado	5
JUAN	Rodriguez 200	Fugl 500	BELEN	FORD KA	2012	12:05 - 10:Ag	Ocupado	1
Juan	Peron 500	Avellaneda 100	PEDRO	FORD KA	2012	13:00 - 11:Ag	Libre	5
JUAN	Colon 1000	Fugl 500	JOSE	Ford KA	2012	15:00 - 12:Ag	Libre	8

Proceso de modelar

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

1. identificar los datos de la realidad (personas, cosas, servicios) y los atributos que nos interesa guardar.

2. identificar las relaciones entre datos.

3. Abstraer y representar

Bases de datos & SQL

► Hasta acá...

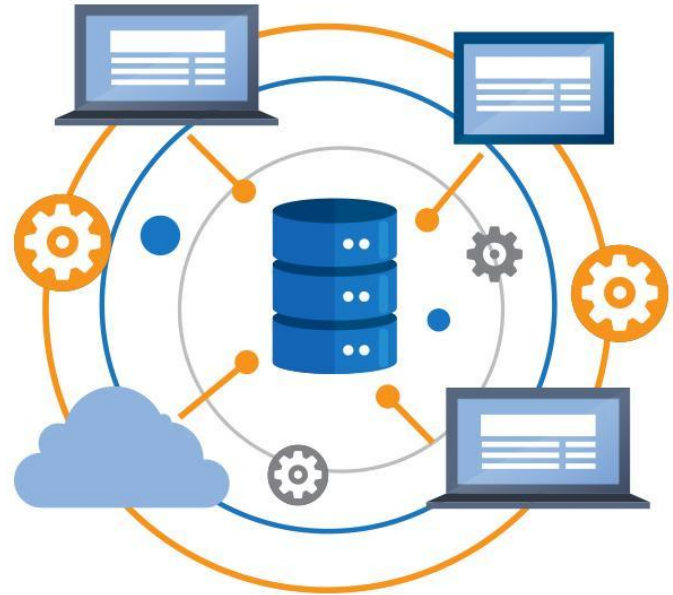
- ▶ Los datos los representamos en tablas
- ▶ Cada tabla es una **entidad** y se **relaciona** con otras entidades
- ▶ Cada tabla tiene **atributos** y un **identificador único**

¿Pero como usamos estos datos?

Un gestor de **base de datos** es una herramienta de software para recopilar y organizar información

Brinda las funciones para *crear tablas y relaciones* y gestionar los *datos*

Pueden ser RELACIONALES o no...



Algunos gestores de bases de datos:

MySQL

Oracle

SQL Server

Access

PostgreSQL

...



Como se usan las BBDD?

Gestionamos usando SQL (Structured Query Language)

Usamos un cliente UI o por código.

Para empezar a manipular nuestra base de datos vamos a usar **phpMyAdmin** que ya viene instalado con XAMPP. Nos permite manejar múltiples BBDD y usuarios en un Servidor!



SQL

Structured Query Language

- ▶ Crear y modificar *esquemas*, tablas y más
- ▶ Consultar por registros de múltiples tablas
- ▶ Realizar **altas, bajas y modificaciones** de datos (ABM)



Sentencias SQL

Crear Tabla

```
CREATE TABLE `viaje` (  
  `id` int,  
  `origen-destino` text,  
  `fecha-hora` date,  
  `precio` float,  
  `id_conductor` int  
);
```



INSERTAR

Insertando registros en una tabla

INSERT INTO <tabla> (campos) **VALUES** (valores);

EJEMPLOS

Inserta un libro en la base de datos

```
INSERT INTO `sistema`.`articulo` (`id_categoria`, `descripcion`,  
`cantidad`, precio`, `stockMinimo`) VALUES (`2`, 'Pronto', 500,  
`13.56`, `10`);
```

Inserta una categoría en la base de datos

```
INSERT INTO `sistema`.`categoria` (`nombre`, `descripcion`)  
VALUES ( 'Libro', 'Libros impresos en color'), ('Revista',  
'Variedad de revistas');
```

SELECCIONAR

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

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

```
SELECT * FROM `sistema`.`articulo`  
WHERE id = 10;
```

```
SELECT precio, stockMinimo FROM `sistema`.`articulo`  
WHERE precio > 97.5 ORDER BY id ASC LIMIT 5;
```


ACTUALIZAR/BORRAR

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 = 2;
```

► Como seguimos?

- ▶ En la práctica vamos a invocar nuestra base de datos desde PHP
- ▶ Para eso usaremos una librería llamada PDO
- ▶ Vamos a empezar a comenzar un proyecto nuevo que va a ser el hilo conductor a lo largo de la materia.
- ▶ Veremos cómo se devuelven los valores y cómo organizar nuestro sistema (Modelo-Vista-Controlador)