Introducción a bases de datos en Python.

Indice del contenido:

Bienvenido a esta **introducción a bases de datos** para aquellos que no tienen aún claros los conceptos de almacenamiento o tienen una vaga noción. Vamos a hablar sobre el manejo de la información en informática, porque son importantes las bases de datos y cómo se organizan las mismas internamente.

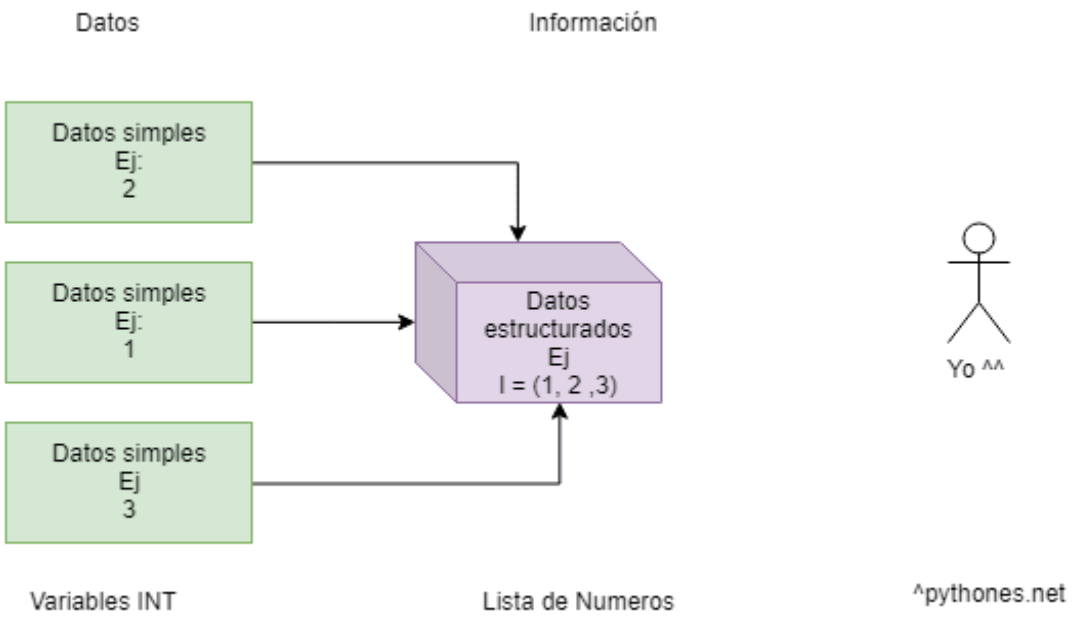
Introducción a las bases de datos.

– ¿Qué son los datos y la información?.

Creo haber explicado anteriormente que **existen diferentes tipos de datos y conjuntos de ellos, si tienen una relación o están ordenados a un propósito pueden considerarse información**.

Nosotros cuando programamos trabajamos con datos, como por ejemplo, unidades de un mismo tipo las cuales almacenamos en variables (**datos simples**), o bien conjuntos de datos individuales que pueden ser del mismo tipo o no (**datos estructurados**). Esto lo explique aquí por si te quedan dudas ->Fundamentos de programación – Python – Estructura de datos y también en Tipos de datos.

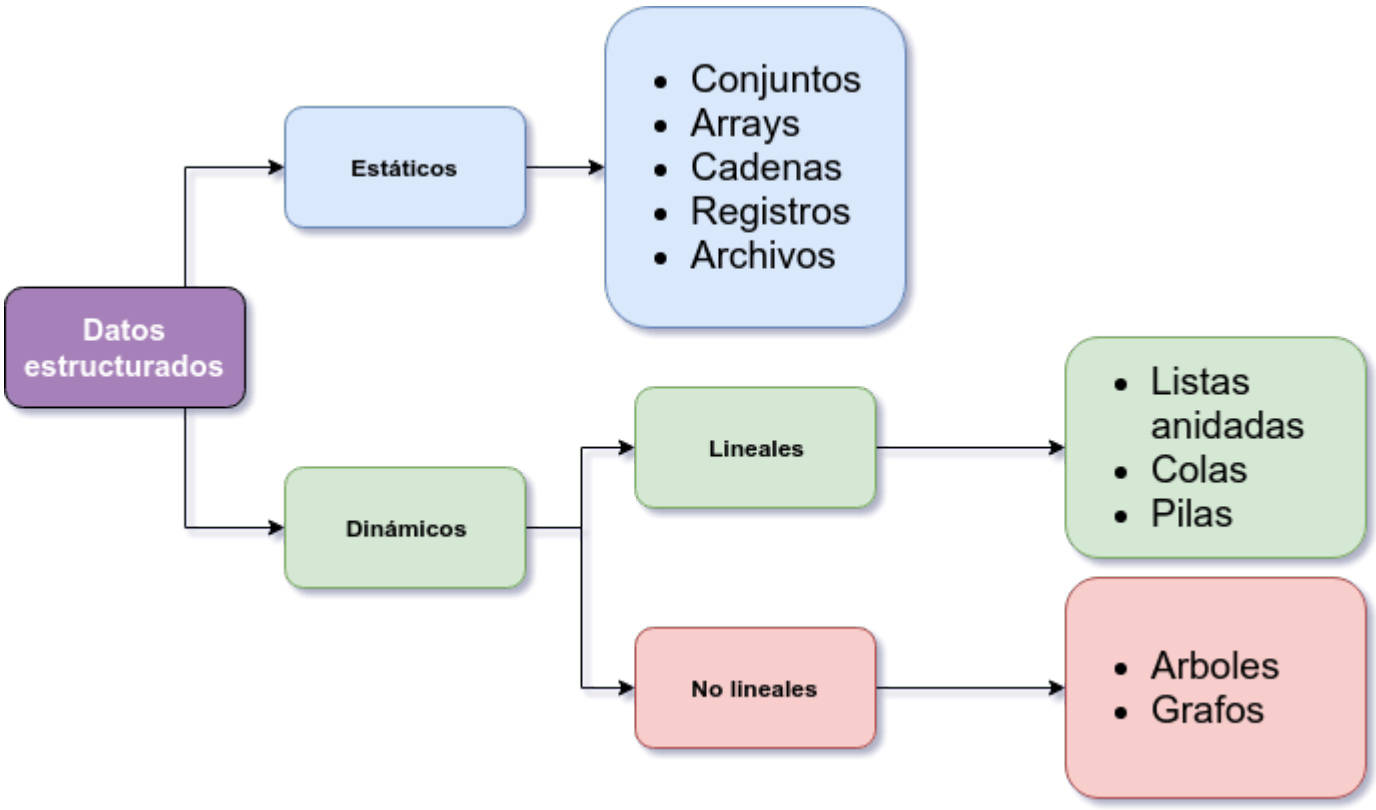
Pero aún así y porque no quiero que te vayas a perder te traigo de nuevo los diagramas:



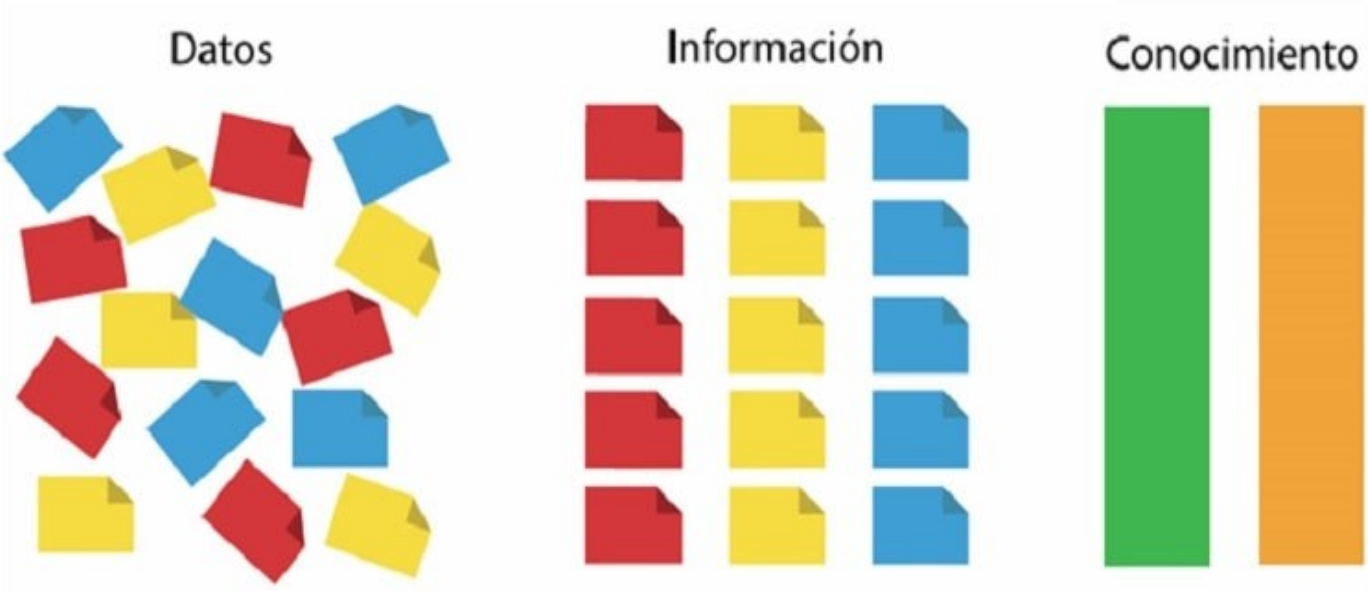
Discernimos un dato simple de un dato estructurado. Sabemos muy bien que ni un dato simple ni un dato estructurado son información si no son pertinentes para quien los **consulta**. ¿Me explico?..**Puede venir alguien y decirnos una sarta de números que a nosotros no nos importan ni tienen relación entre sí**...**¿Podríamos considerar eso información?**. Pues no, deberíamos llamar al centro psiquiátrico más próximo.

Pero ahora, si nosotros identificamos que ese es un **número telefónico**, pues si, esta persona no está loca y nos está dando información (tal vez una que no pedimos, pero es **información**).

Ahora cuando leímos sobre datos estructurados y como ordenar conjuntos de datos simples, vimos algo como lo siguiente:



Estos **datos estructurados** normalmente nos permiten alojar una serie de **datos simples** de diferentes tipos (o no), **que son pertinentes como información, la información ordenada con propósito se convierte en conocimiento** 😉**.**



Así en los ejemplos de programación aprendimos a ordenar datos del tipo:

Maria es mujer, tiene 26 años, trabaja en Administración y tiene un salario de $26000.

Estos datos podríamos almacenarlos en un diccionario en Python, así:

Persona = {
*'**Nombre**' : 'Maria',*
*'**Género**' : 'Femenino',*
*'**Edad**' : 26,*
*'**Empleo**' : 'Administración',*
*'**Salario**' :26000,*
}

Y claro, tenemos un **diccionario con los datos de Maria**, donde podemos consultar toda su información o algún dato simple importante para nosotros en ese momento (también información porque es pertinente). Todos esos datos organizados en un diccionario mediante el par "**Clave – Valor**". Excelente!.

¿Pero, y si te dijera que tienes que añadir los datos de un millón de personas como Maria, ¿Cuánto crees que va a pesar tu archivo ".py", dónde alojarías estos datos?. ¿Y si además te dijera que quiero poder consultar esos datos de forma fácil?. Ahhh, y además quiero poder modificarlos, listarlos según algunos criterios como el salario, también quiero que sea rápido y poder relacionar a María con Carlos y Matías porque viven juntos!. Tu me vas a mandar al diablo y Python ni te cuento!.

Y tú podrías pensar ¿y si guardo todos esos datos en un archivo aparte y luego lo consulto desde un solo software o script ".py"?.

Pues has dado en el clavo.. Eso, es una ***base de datos***!.

- Si pero es un archivo, podría ser txt…
- **Pues es una base de datos igual!**

La importancia de las bases de datos

Esos datos almacenados aparte del código funcional se convierten en una base de datos. Y basándonos en el ejemplo anterior está

claro que no podemos añadir esos millones de datos en el mismo script de Python porque estaríamos mezclando perros, gatos y ratones en un mismo corral!. Y luego vaya que tengas que modificar algo desde la pc de la abuela..

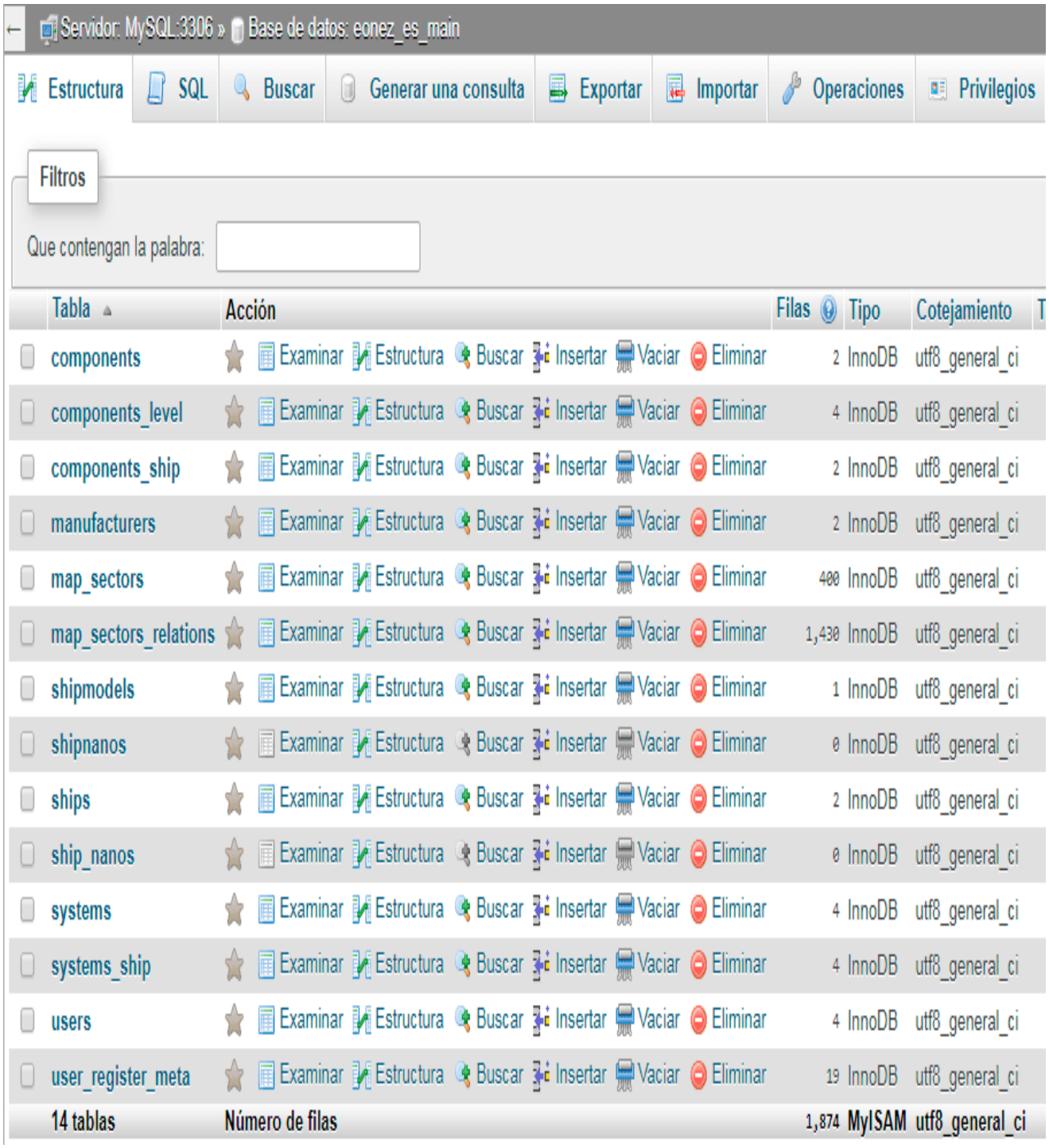
Por ello utilizamos bases de datos dónde almacenamos estos datos aparte del código de Python, entonces en Python solo programamos la parte "funcional" o "el software" y aparte tenemos una **base de datos** a la cual **consultar** cuando necesitamos información. Y te preguntarás... ¿Es esto más rápido que tener todos los datos en el mismo archivo?. **Pues si, es más rápido, más cómodo, más simple y más organizado.**

> *Podríamos verlo así también, palabras son datos, un libro es información porque tiene datos (estas palabras) ordenadas con un propósito (contar una historia) y una biblioteca es una base de datos porque almacena esa información de forma ordenada!. También con su propósito que es servirla al lector rápidamente, porque venga, si viene alguien a pedirte "el principito" y tú tienes una montaña de libros del alto de una casa, no vas a encontrarlo rápido.. Pero si tienes una estantería organizada por autores, tardarás poco tiempo..*

-Y ¿cómo es esa estantería?.

Pues, en una biblioteca así:

Y en informática así:



O así:


```
{
    "_id": "a26ab886a7d200a93a32ae192200416d",
    "_rev": "4-c97d88dfa8c79468bcbe03fcf0e5921d",
    "another_attribute": 17,
    "yet_another_attribute": "a_string",
    "_attachments": {
        "Screenshot.png": {
            "content_type": "image/png",
            "revpos": 3,
            "length": 22666,
            "stub": true
        },
        "trace.nam": {
            "content_type": "application/octet-stream",
            "revpos": 2,
            "length": 7830,
            "stub": true
        }
    }
}
```


Recuerda que esto es una introducción a las bases de datos y por ende no pretendo que comprendas el "código" (entre comillas) de la imagen, ni el manejo de las tablas. Solo que veas como **se puede organizar la información en un fichero externo**.

Claro, una puede parecerse a un excel y la otra a un diccionario en un archivo. **Ambas pueden ser bases de datos, lo que cambia es el modelo!!**. Recordarás que en programación orientada a objetos (clases) hemos hablado de modelo como "plantilla". Pues existen diferentes "plantillas", "modelos" o formas de almacenar los datos en bases de datos. Y estos "modelos" o "plantillas" **los elige el programador según diversos criterios los cuales veremos más adelante. Aunque la mayoría tienen algo en común, sus componentes tradicionales.**

Al igual que en un diccionario en python tú almacenas los datos en un par "**clave : valor**" las bases de datos también tienen una especial forma de almacenarlos. Cada dato irá dentro de lo que se conoce normalmente como "**campo**". Un campo podría verse como una celda, un espacio, un cubículo o en el caso de los libros como ejemplificamos antes un espacio para cada palabra. **Así cada "campo" almacena un dato "del mismo tipo"**. También existen los **registros** que almacenan el "conjunto de campos" (de diferente tipo cada campo o del mismo, pero varios campos). Y estos se organizan en una **tabla (dependiendo del *modelo,* que a su vez se organizan en filas y columnas)**, así:

		Tabla					
			Edad (entero)	Empleo (String)	Salario (Entero)	Columna	
Registro 1	Fila	campo	campo	campo	campo	campo	campo
Registro 2		Maria	26	Administración	26000	NULL	NULL
Registro 3		Carlos	27	Administración	24000	NULL	NULL
Registro 4		campo	campo	campo	campo	campo	campo

Si prestas atención a la imagen (es un excel cutre) la información está ordenada dentro de una tabla en **filas** (de izquierda a derecha) y **columnas** (de arriba a abajo). Esta contiene **campos** de diferentes tipos entre sí pero del mismo tipo determinado por la columna, por ejemplo todos los campos pertenecientes a edad almacenarán datos numéricos o un valor **Null** (nada).

Cada conjunto de campos viéndolos de izquierda a derecha son un **registro** (**que bien podría llamarse (Tupla) a cada registro**) por ende en este ejemplo es una persona con sus datos. Podríamos decir que un registro válido sería:

- **(“Carlos”, 27, “Administración”, 24000, Null, Null)**

Dentro de ese registro podemos ver que Carlos pertenece al tipo **string**, luego su edad al tipo de datos “**int**“, su empleo **string**… etc. **Y al ser almacenado en la tabla estos tipos de datos deben coincidir, puesto que cada columna determina el tipo de dato y no permite ingresar otro tipo, solo datos de ese tipo o bien el valor “null”**.

Null.

El valor null es una referencia para **determinar que aún no se especificó esa información o bien no se conoce**. Normalmente null nos permite evitar dejar un campo vacío indicando que aún no existe ese valor.

Introducción a bases de datos – Organización de los datos

Como programadores o informáticos tenemos que pensar lógicamente siempre en la importancia de los datos, los tipos y el manejo de los mismos.

Simplemente a medida que aprendíamos programación almacenavamos los datos dentro del mismo código de nuestro software normalmente en tipos de datos estructurados, lo cual está bien para empezar, pero es momento de integrar las bases de datos y comenzar a trabajarlas en cada proyecto. Para

ello te invito a leer la siguiente entrada que considero también una introducción a las bases de datos pero ya un poquito más haciendo hincapié en la historia del desarrollo y mantenimiento de software así como el nacimiento del modelo relacional de bases de datos.