Tema 3.

Organización, consulta y tratamiento de la información.

# 1.- Organización y consulta de la información.

Básicamente por poseer un módulo base un tanto más sencillo formado tan sólo por **Empresas** y **Administración**, nos centraremos en Odoo.

La base de datos de un sistema ERP es de gran envergadura. Almacena las tablas con los datos de la aplicación, vistas de las diferentes tablas y otros elementos como funciones o [disparadores](https://ead.murciaeduca.es/pluginfile.php/1433024/mod_resource/content/1/1_organizacin_y_consulta_de_la_informacin.html#t7b529be0-e9d0-b1b3-7747-ede56e6ffe66) que realizan operaciones sobre los datos

Lo que se hace es establecer una serie de normativas o nomenclatura para organizar la información, que los desarrolladores deben seguir a la hora de modificar el código fuente o el esquema de la base de datos. Por ejemplo:

* incluir un prefijo en los componentes de la base de datos, para saber a qué módulo pertenecen.
* o establecer una serie de campos dentro de una tabla como obligatorios, para poder asegurar el funcionamiento correcto de la aplicación.

En los ERP desarrollados en un lenguaje OO, **cualquier dato es accesible a través de objetos**.

**Por ejemplo**, en Odoo tenemos un objeto **res.partner**, para acceder a los datos concernientes a los colaboradores o socios; el objeto **account.invoice** para los datos de las facturas, etc. Como ves, ambos van precedidos de un prefijo que indica el módulo al cual pertenecen.

**Cualquier método que quiera actuar sobre un objeto deberá** tener un parámetro que **indique** **sobre qué recurso o registro** dentro del objeto se quiere actuar

**Por ejemplo**: si queremos enviar un correo electrónico a los colaboradores identificados como 1 y 5 dentro de la tabla ***res.partner***, utilizaremos la siguiente instrucción:

res.partner.send\_email(..., [1,5], ...)

En resumen, un objeto es, en un concepto amplio, todo elemento que forma parte de la aplicación y que permite acceder a los datos de la misma.

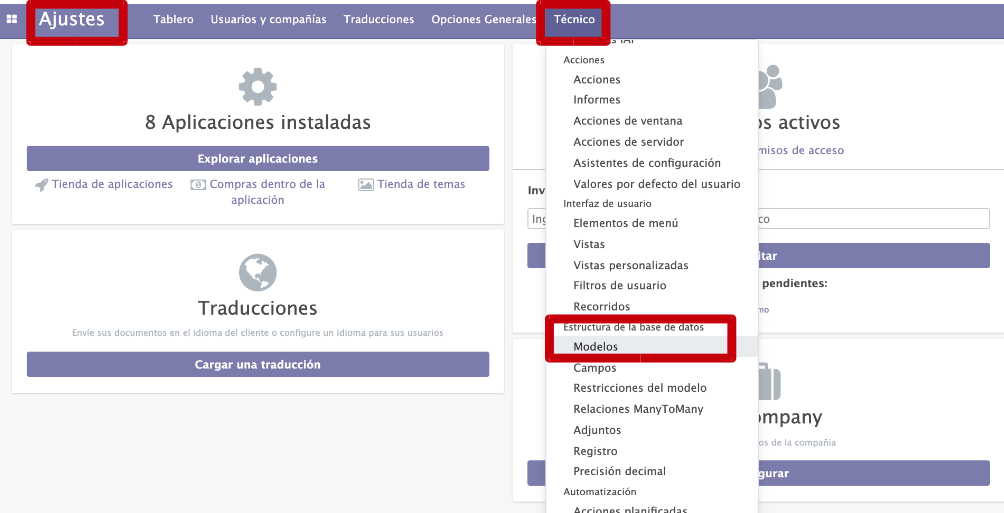
**Para saber más**: en este apartado vamos a encontrarnos con cierta terminología de base de datos al acceder a los datos del ERP. Además de la documentación oficial de Postgres, te recomendamos estos tutoriales: [PostgreSQL Ya (tutorialesprogramacionya.com)](https://www.tutorialesprogramacionya.com/postgresqlya/)

También puedes repasar **conceptos sobre el lenguaje** SQL (lenguaje de consulta estructurado) en el siguiente enlace: [SQL Tutorial (w3schools.com)](https://www.w3schools.com/sql/)

## 1.1.- Tablas y vistas de la base de datos.

**Un objeto guarda todo elemento que forma parte de la aplicación**. Como elementos de la aplicación tenemos la propia base de datos (tablas, disparadores o acciones sobre esas tablas, etc), los formularios, los informes, las ventanas o los asistentes, por ejemplo.

**Cómo acceder a los objetos**: en Odoo > Ajustes > Técnico > Estructura de la base de datos > Modelos



**En Openbravo**: todos los objetos, llamados **metadatos**, están disponibles y accesibles en el módulo **Diccionario de Datos**, donde podremos administrarlos **de una manera centralizada**. En este módulo se dan de alta los **formularios, informes y procesos** que se utilizan en la aplicación, a los que se accederán mediante menús o botones en ventanas, así como las **tablas** y campos (o columnas) que las forman.

Recuerda que para entrar en Odoo, debemos utilizar la IP del servidor seguido del puerto de comunicación. Si trabajamos en local, la dirección será **localhost:8069**.

**Objetos principales en Odoo**:

* **Modelos**: son las tablas de la base de datos.
* **Vistas**: son los distintos tipos de formularios con los que muestra los datos de los modelos.
* **Menús**: son con los que interaccionamos para llamar a las acciones para manejar los datos de los modelos.
* **Acciones**: Los métodos desde los que vamos a abrir nuestras vistas para trabajar en la aplicación.

**Dentro de los objetos, tenemos**:

* **Tablas**: es una **estructura de datos organizada en filas y columnas**, de manera que cada columna es un campo (o atributo) y cada fila un registro.
* La base de datos está formada por tantas tablas y objetos que se vuelve compleja y difícil de manejar. En esos casos, interesa que **algunos grupos o perfiles de usuarios tengan una vista parcial de esos datos**. Para estos casos se utilizan las vistas.
* **Vistas**: una vista es una “tabla virtual”, a la que **se puede acceder como si fuera una tabla del esquema, pero que realmente no lo es**. Tienen la misma estructura que las tablas: filas y columnas.

Para administrar la base de datos podemos conectarnos directamente al gestor de la base de datos , aunque lo normal es utilizar herramientas gráficas que nos faciliten el trabajo (como PGAdmin 4).

**Debes conocer**: [consultar cómo se instala la herramienta gráfica PgAdmin 4 para administrar bases de datos PostgreSQL](https://www.digitalocean.com/community/tutorials/how-to-install-configure-pgadmin4-server-mode-es).

Una vez instalado PgAdmin 4, **nos conectaremos a la base de datos introduciendo los siguientes datos**:

* **Nombre**: nombre que queramos darle a la conexión.
* **Servidor**: Dirección IP o nombre del servidor.
* **Puerto**: Número de puerto para la conexión, normalmente el 5432.
* **Base de datos de Mantenimiento**: base de datos inicial con la que nos conectamos, normalmente llamada "postgres".
* **Nombre de usuario**: usuario con el que queramos conectarnos, en nuestro caso, debemos introducir el usuario que creamos en la segunda unidad, concretamente, el usuario "userbd".
* **Contraseña**: clave del usuario.

Texto

Descripción generada automáticamente

## 1.2.- Consultas de acceso a datos.

* Nos permiten acceder a la información que guardan las tablas y vistas de la base de datos.
* Sirven para indicar al SGBD que devuelva un extracto de la información en forma de un conjunto de registros.

Los **pasos para crear una consulta son**:

* **Seleccionar las tablas o vistas** sobres las que va a actuar la consulta.
* **Establecer la relación entre las tablas y vistas**, en caso de que la aplicación no la proporcione.
* **Seleccionar los campos a mostrar** en la consulta.
* **Ejecutar la consulta**.

Las **consultas** pueden **actuar sobre una o varias tablas o vistas**, y se pueden **guardar** para ser utilizadas posteriormente.

Las consultas se pueden realizar **desde la propia aplicación, o bien** conectándonos directamente al **SGBD**.

Las consultas de acceso a datos se pueden **construir escribiendo el código** en el lenguaje de consulta utilizado, como por ejemplo SQL**, o bien mediante asistentes y constructores gráficos** si se trata de consultas poco complejas.

**Debes conocer**: [cómo visualizar gráficamente una consulta de acceso a datos con PgAdmin 4](https://www.comunidadism.es/gestion-de-bases-de-datos-postgresql-nuevo-pgadmin4/), ahora bien no se puede crear con PgAdmin 4 una consulta gráficamente como sí era posible en su versión anterior pgAdmin III. Puedes ver en esta página web cómo se hace.

# 2.- Visualización de la información.

**Tanto la información de las tablas como la de cualquier otro objeto de la aplicación se muestra a través de interfaces**.

Cada objeto tiene su propia interfaz, por ejemplo, no se muestran de la misma manera los datos de las Empresas que los datos de una Factura. Las interfaces pueden ser:

* **Estáticas**: se crean dentro del código de la aplicación y no pueden ser modificadas.
* **Dinámicas**: pueden ser modificadas por parte del usuario, para lo cual se almacena la descripción de la vista en un lenguaje de descripción de datos que permita su modificación, como por ejemplo **XML**.

Por tanto, las interfaces dinámicas son construidas de forma dinámica por la descripción XML de la pantalla del cliente.

Para ello no es necesario ser unos expertos en ese lenguaje, **podemos hacer objetos sencillos simplemente tomando como ejemplo otros objetos que haya** creados en la aplicación. No obstante, muchas aplicaciones proveen la forma de crear las descripciones de manera gráfica sin necesidad de introducir código manual.

El siguiente código es un ejemplo de descripción de un objeto en Odoo:

<?xml version="1.0" encoding="utf-8"?>

<form string="Partner">

<field name="name"/>

<field name="title"/>

<field name="ref"/>

<field name="lang"/>

<field name="customer"/>

<field name="supplier"/>

</form>

**Con el código anterior estamos creando una interface para la introducción y consulta de datos**. En concreto, se muestran seis campos correspondientes al nombre, título o tratamiento en caso de ser una persona física, código, idioma, y por último dos campos booleanos que guardan si se trata de un cliente o de un proveedor.

**Ejemplo del código que puse yo en mi tarea**:

<form string="Formulario formSARA">

<group>

<field name="name" string="Nombre de la compañia"/>

<field name="company\_type" string="Tipo de compañia"/>

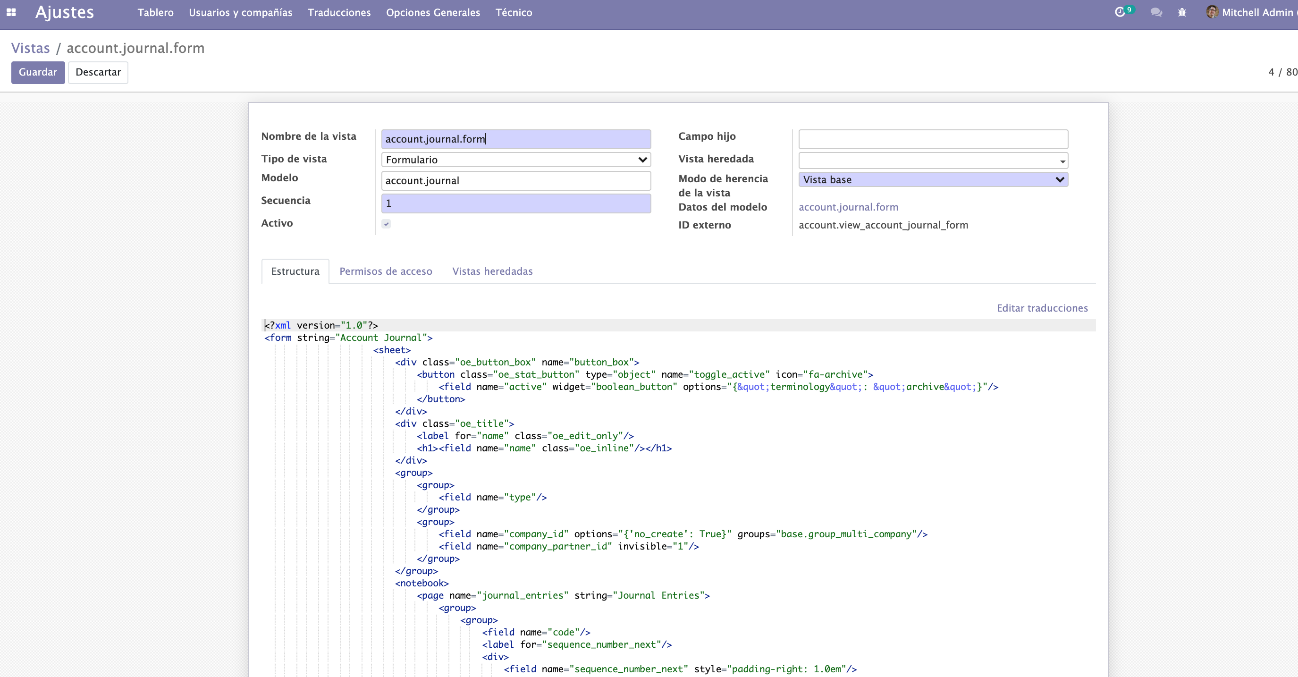
<field name="country\_id" string="Pais"/>

<field name="create\_date" string="Fecha de creacion"/>

<field name="email" string="Email"/>

</group>

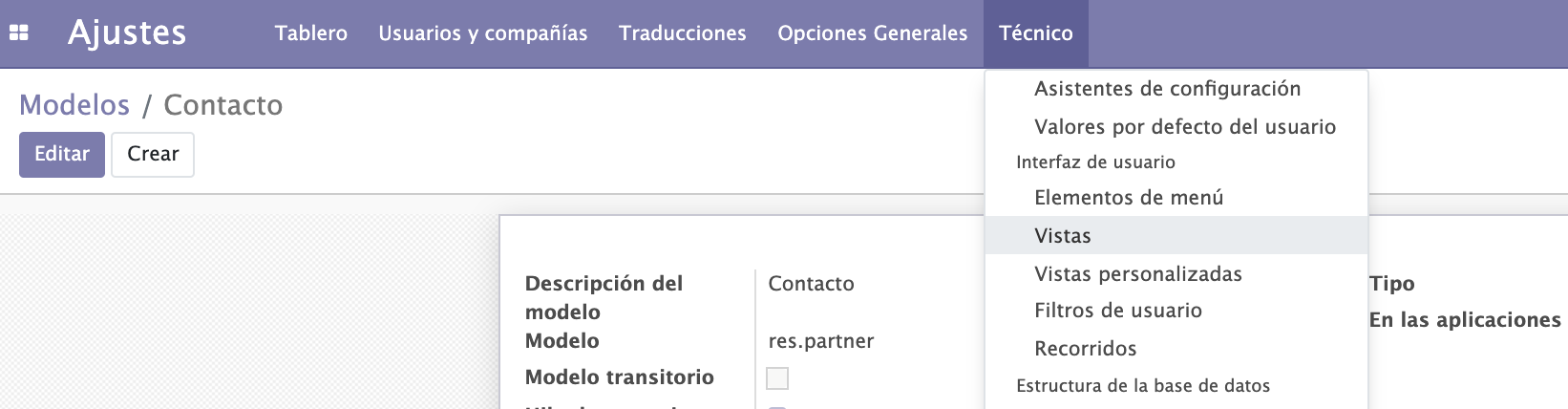
</form>



## 2.1.- Interfaces de entrada de datos y de procesos. Formularios y Gráficos.

**Podemos definir una o varias interfaces para cada objeto** para describir qué campos se muestran y cómo. Dependiendo de **cómo se distribuyan los campos** nos encontramos, entre otras, con los siguientes **tipos de interfaces**:

* **Formularios**: en este tipo de interfaces se muestra un solo registro, cuyos campos se distribuyen en la pantalla siguiendo el mismo criterio: normalmente de izquierda a derecha y de arriba abajo, de acuerdo con el orden en el que son descritos en la vista.



* **Árboles**: mostrar un conjunto de registros en modo lista, y es útil para mostrar varios registros a la vez y realizar búsquedas sobre ellos.
* **Gráficos**: son otra forma de ver los datos de un formulario. Pueden mostrarse en varios formatos y tipos para una mejor visualización de la información.

Odoo llama a las interfaces **Vistas**, y desde la aplicación las podemos personalizar de dos maneras diferentes:

* **Escribiendo código XML**. Editamos la vista y escribimos directamente el código. Esto lo hacemos a desde el menú Ajustes/Técnico/Interfaz de usuario/Vistas. (Recuerda que tienes que ser superusuario)
* **Accediendo al módulo y modificando el fichero XML correspondiente**. La ruta donde se encuentran las aplicaciones es diferente de linux a Windows.
  + En Linux la ruta es /usr/lib/python3/dist-packages/odoo/addons
  + En Windows se encuentra en C:/archivos de programa/odoo/addons

**Una vez modificada la vista, es recomendable actualizar la aplicación** desde el menú **Aplicaciones**, buscas tu aplicación y en los tres puntos de su parte superior derecha pulsas y seleccionas Actualizar.

**Para saber más**: [consultar información sobre los distintos elementos que se pueden utilizar dentro de la definición de una vista](https://www.odoo.com/documentation/16.0/developer/reference/backend/views.html). Comprueba la versión, puede cambiar.

Texto, Sitio web

Descripción generada automáticamente

## 2.2.- Definición de campos.

Por ejemplo, una **vista tipo formulario con dos campos**, nombre y título, la definiríamos así:

<?xml version="1.0"?>

<form string=”Empresa”>

<field name=”name”/>

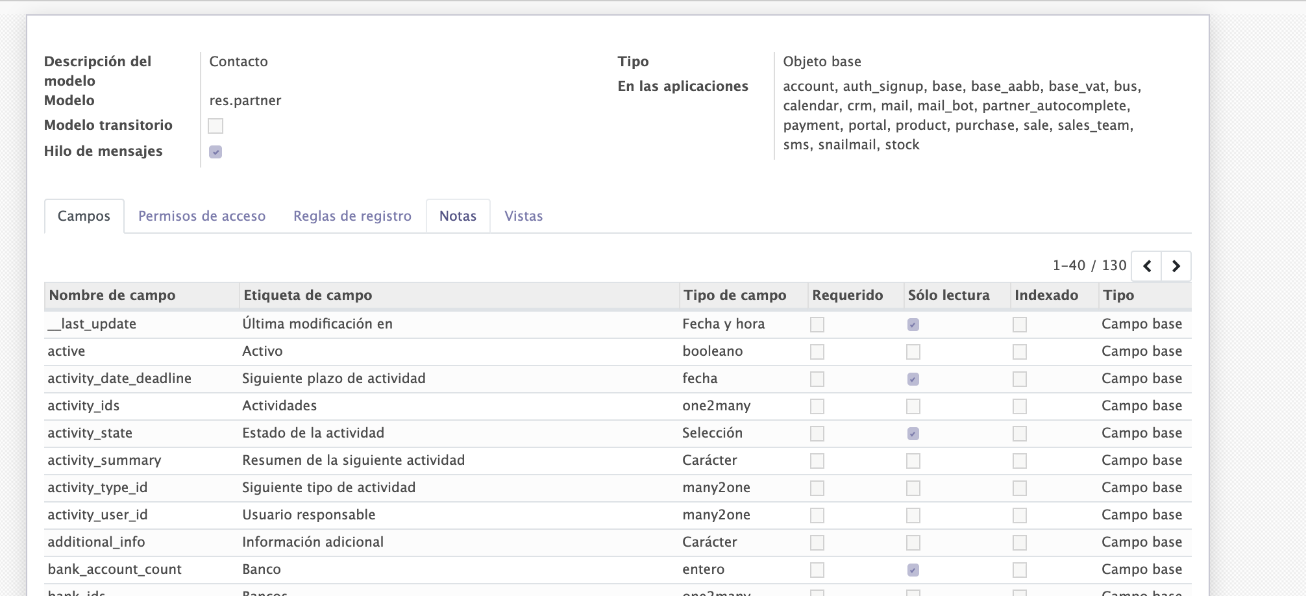
<field name=”title”/>

</form>

**Explicación**:

* **Etiqueta form**: indica que vamos a describir un formulario
* **Etiqueta field**: campo.

En las vistas de tipo formulario, los campos van precedidos por una etiqueta con su nombre y se colocan de izquierda a derecha, en el orden con que son declarados en el archivo XML.



Hay un **tipo especial de campo** que son los campos de relación **one2many**. Estos campos se utilizan para reflejar una relación de uno-a-muchos entre dos objetos.

**Por ejemplo**, el campo **address**, del objeto **res.partner** es un campo one2many, ya que ese campo está enlazado con otro objeto, en este caso, **res.partner.address**. Esto es: que un registro de **res.partner** puede tener muchos registros relacionados en **res.partner.address**. En otras palabras: que una empresa puede tener muchas direcciones también llamadas contactos de la empresa, y esas direcciones se guardan en la tabla **res.partner.address**.

**A la hora de describir este tipo de campos en el archivo XML, utilizaremos la sintaxis**:

<field mode=”form,tree” name=”address” >

<dentro pondremos los campos que forman parte de res.partner.address>

</field>

Además de los campos, existen varios **elementos de diseño** que nos permiten **personalizar las vistas *form* y *tree***. Algunos de ellos son:

* **mode**: tipo de vista que va a permitir el objeto.
* **name**: nombre del campo tal y como aparece en el objeto.
* **separtor**: agrega una línea de separación en el formato.. Ejemplo:

<separator string="link" colspan="4"/>

* **notebook**: permite distribuir los campos de la vista en diferentes pestañas o tabs que van definidos por páginas, ejemplo:

<notebook colspan="4">....</notebook>

Por otra parte, a la etiqueta **field** se le pueden añadir **atributos**. Algunos de ellos son:

* **colspan**="4":El numero de columnas por las que se puede extender un campo.
* **readonly**="1": Establece un campo como solo lectura .
* **invisible**="True": Oculta el campo y su etiqueta .
* **password**="True": Reemplaza la entrada de un campo con un símbolo "•" .
* **string**="": La etiqueta que va a aparecer junto al campo.

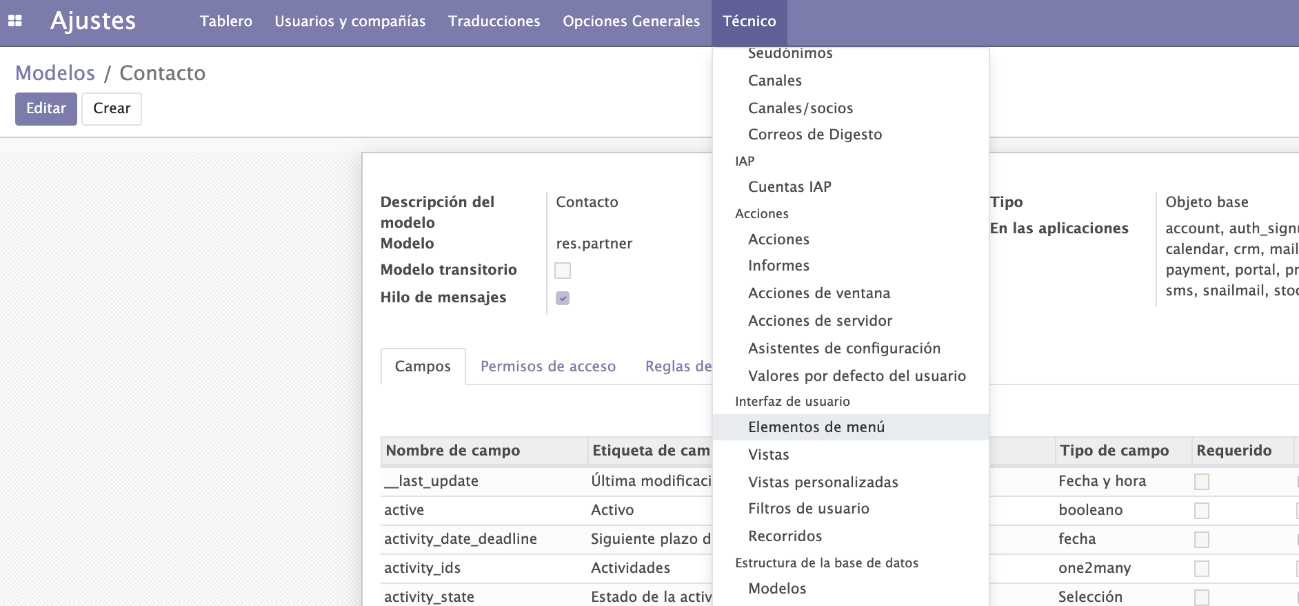
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

## 2.3.- Menús.

Para acceder a los objetos que creamos en la aplicación, deben definirse previamente los menús, así como las acciones asociadas a los mismos.

**Los menús se pueden editar desde**: **Ajustes/Técnico/Interface** **de** **usuario/Elementos de Menú**. Desde ahí podemos modificar el menú seleccionado o crear nuevos menús. **Para crear menú secundario** seguiremos el mismo proceso, tan sólo tendremos que **indicar cuál es el menú padre** del cual depende.



Los **menús tienen asociada la acción que deben realizar**. Hay diferentes tipos de acciones, las más importantes:

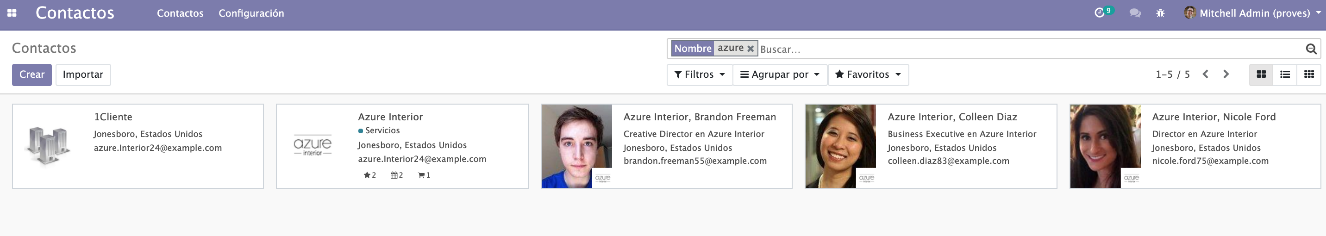
* **ir.actions.act\_window**: Abre una vista en una nueva ventana.
* **ir.actions.report**: Imprime un informe.

Si la acción no está creada, debemos utilizar el enlace **Crear…**. Los campos a introducir variarán según la acción de que se trate. Por ejemplo, **para** **crear una nueva acción que abra una vista**, debemos introducir al menos los siguientes datos:

* **Nombre de la acción**.
* **Tipo de la acción**, en este caso, ir.actions.acc\_window.
* **Datos de la vista**: tipo (árbol, formulario, etc.) y nombre.

## 2.4.- Búsqueda de información.

Normalmente se incluyen búsquedas básicas sobre los campos principales, y búsquedas avanzadas para buscar registros por campos más específicos.



La opción de búsqueda está disponible en cada ventana de la aplicación. En el caso de Odoo, **los criterios de búsqueda se introducen en la parte superior de la ventana** y debajo se sitúa una vista de tipo árbol con el listado de los registros del objeto. El listado se reducirá a los registros coincidentes con los criterios de búsqueda.

También es posible **aplicar filtros a la selección**. Los **registros** pueden ser **seleccionados** y cambiando al modo formulario podremos **ver** todos sus datos, así como **editar** su contenido.

**Debes conocer**: [explicación sobre los menús personalizados de Odoo](https://www.youtube.com/watch?v=OB5hwe1o6cY).

## 2.5.- Informes y listados de la aplicación.

Los informes y listados de la aplicación son una forma de mostrar los datos para mejorar su visualización. Muchos informes y listados están creados, y son accesibles desde los distintos menús de la aplicación.

Algunos de estos documentos pueden ser:

* **Informes contables**.
* **Albaranes**.
* **Pedidos**.
* **Recibos**.
* **Reclamaciones** a proveedores y/o clientes.
* **Informes estadísticos**.
* **Generación de etiquetas** para un conjunto de registros.
* **Informes de agrupamiento**: permiten mostrar los registros existentes para un mismo valor de un campo.
* **Impresión de documentos del sistema**.

Los **informes y listados** que incorpora la aplicación **se pueden utilizar tal cual se proporcionan**, **o bien adaptar su diseño a la imagen corporativa de la empresa**.

También se pueden **añadir nuevos informes** instalándolos como módulos independientes. En el caso de Odoo, los módulos extra que contienen exclusivamente informes de algún tipo llevan en su nombre la etiqueta **"report"**.

Texto

Descripción generada automáticamente

# 3.- Tratamiento de la información.

El tratamiento de la información se lleva a cabo a través de distintos procesos. Antes de poder realizar ningún proceso es necesario introducir la información propia de la compañía.

**Por ejemplo**: si vamos a llevar la contabilidad de la empresa habrán de configurarse las cuentas contables, se tendrán que definir las cuentas bancarias y los diarios de compras, ventas y caja o banco. También habrán de definirse los impuestos.

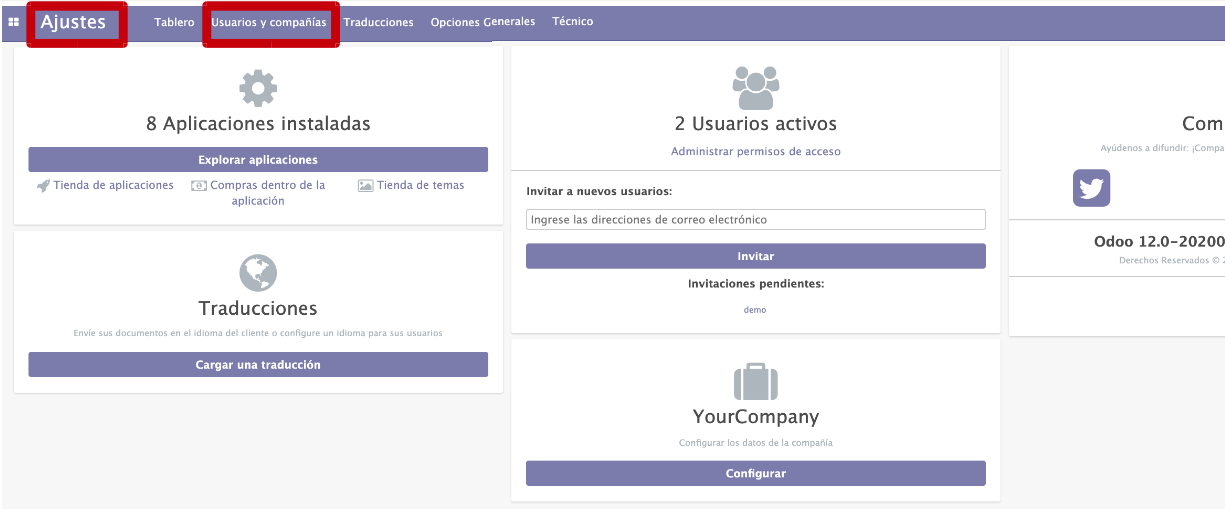
**Por ejemplo**: si tenemos instalado el módulo de contabilidad, cuando creemos una empresa cliente o proveedor, deberemos asociarle una serie de cuentas contables, como las de cobro al cliente y pago al proveedor, por lo que deberemos tener creadas dichas cuentas antes de poder dar de alta a nuestros clientes y proveedores, e igual con los productos.

Dentro de la información propia de la compañía **deberemos crear los clientes, proveedores y productos**. En cualquiera de los casos, existe una **información mínima que tendremos que introducir antes de poder empezar a trabajar**.

Al instalar el perfil mínimo de Odoo deberemos comprobar si existen datos, y si no, introducirlos, de al menos los siguientes objetos o tablas:

* **Título**: Hace referencia al **tipo de empresa** o al tratamiento que se le debe otorgar a una persona de **contacto**. Si es uno u otro se distingue a través del campo **Dominio** que hay en la tabla, que será igual a **"Empresa"** en el primero de los casos o a **"Contacto"** en el segundo caso.
* **Función**: Listado de cargos o funciones de las personas de contacto.
* **Provincias**: Listado de provincias por país.

Esta información es necesaria para poder introducir los datos de las empresas, y podemos acceder a ella desde el menú **Ajustes/Usuarios y Compañías/Compañías**.



## 3.1.- Cálculos: pedidos, albaranes, facturas, asientos predefinidos, trazabilidad y producción.

Entre los procesos a realizar por un sistema de planificación empresarial destacan los siguientes:

* **Contabilidad**: procesos donde se reflejan las **operaciones económicas** realizadas por la empresa, la determinación de los costes de la empresa y los presupuestos del ejercicio fiscal.
  + Se proporcionan asientos predefinidos para la introducción rápida de asientos sin necesidad de tener conocimientos de contabilidad.
* **Operaciones de compra**:
  + Crear una orden de compra o pedido de compra.
  + Recibir los bienes.
  + Controlar la factura de compra.
  + Registrar el pago al proveedor.
* **Operaciones de venta**:
  + Crear una orden de venta o pedido de venta y recibir la conformidad del cliente.
  + Preparar los bienes a enviar al cliente y realizar el albarán y la entrega.
  + Realizar la factura de venta.
  + Registrar el cobro al cliente o pago del cliente.
* **Trazabilidad**: se llama así al proceso de la entrada del producto hasta la salida del mismo.

**Nota**: Es posible que por las necesidades de la empresa no sea necesario utilizar todos los procesos del ERP, por ejemplo, puede ser usado sólo como un CRM, o sólo como un programa de contabilidad, pero su verdadera potencia se alcanza con la integración de todas sus funciones.

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

## 3.2.- Procedimientos almacenados de servidor.

Dentro del tratamiento que le damos a la información, a veces es necesario que se lleven a cabo determinadas tareas de forma automática, en respuesta a algo.

**Por ejemplo**: podemos querer enviar un email al cliente automáticamente cuando nos hace un pedido, o bien registrar en la ficha del cliente el pedido realizado. Ambas acciones, son desencadenadas automáticamente por el sistema, y la acción que las desencadena es la realización de un pedido por parte de un cliente.

**Para hacer esto en la mayoría de los sistemas podemos utilizar**:

* **Procedimientos (o funciones) almacenados de servidor**: se almacena en una base de datos. Pueden ser ejecutados dir4ectamente por el usuario, o bien cuando se cumpla una determinada condición, a través de disparadores.
* **Eventos de servidor**: consiste en detectar que pasa algo en la aplicación y hacer que el sistema responda a este suceso de forma automática. Los eventos de servidor son creados a nivel de objetos, y no de base de datos.

Un procedimiento almacenado en PostgreSQL **se puede escribir en múltiples lenguajes de programación**. Para definir un procedimiento en PgAdmin4 utilizamos la siguiente sintaxis:

CREATE [ OR REPLACE ] FUNCTION

nombre\_funcion([ argumentos ])

RETURNS tipo AS

$BODY$

codigo

$BODY$

LANGUAGE lenguaje;

En el código podemos escribir cualquier instrucción del lenguaje SQL para el manejo de base de datos. Los elementos entre corchetes son opcionales, no es necesario ponerlos si no los vamos a utilizar. **El procedimiento se ejecuta con una instrucción** **SELECT**:

**SELECT nombre\_funcion();**

**Ejercicio resuelto**: Crear un procedimiento almacenado en Odoo que muestre las empresas que sean sociedades limitadas.

CREATE OR REPLACE FUNCTION PRUEBA()

RETURNS **setof** res\_partner AS

$BODY$

select \* from res\_partner where title='Ltd'

$BODY$

LANGUAGE sql;

Captura de pantalla con la imagen de una pantalla

Descripción generada automáticamente

# 4.- Extracción de datos en sistemas de ERP-CRM y almacenes de datos.

El proceso de extracción de datos podemos definirlo como la **operación de sacar datos de una aplicación para ser tratados en otra aplicación**. La extracción de datos puede realizarse utilizando diferentes sistemas.

Un **uso muy común** es utilizar herramientas ofimáticas que se conectan a la aplicación ERP, para **obtener información de la base de datos y volcarla en la aplicación ofimática** como un procesador de textos, una hoja de cálculo, etc.

Existen **procesos más complicados y potentes** **de extraer información**. Son los llamados procesos de **Business Intelligence**. Este tipo de soluciones deben realizar tres tareas:

* Transformar y **combinar los datos para extraer la información**.
* **Convertirla en potentes indicadores** y
* Mostrarla en distintos formatos gráficos.

**Según el origen de los datos y el tipo de información** que queramos obtener, se pueden utilizar:

* **Consultas e Informes**: se usan cuando todos los datos están en una sola BB.DD. Se extraen a partir de una consulta SQL.
  + La aplicación facilita informes y consultas predefinidos, aunque también se pueden generar consultas e informes personalizados desde el propio Odoo, o con herramientas externas como Jasper Reports.
* **Almacenes de datos**: La extracción de datos se hace desde diferentes sistemas y distintas bases de datos, creando almacenes de datos con el objetivo de homogeneizar e integrar la información.
* **Cubos multidimensionales**: Un cubo n-dimensional es un conjunto de datos multidimensionales organizados en ejes y celdas, que maneja la información de una base de datos relacional.
  + **También existen bases de datos multidimensionales**, como contraposición a la operación de guardar los datos en BB.DD relacionales (formadas por filas y columnas) y luego manejarlos con cubos.

En ocasiones, el proceso de **extracción y manipulación de la información no se realiza en tiempo real** debido al gran volumen de información que hay que manejar, para evitar una disminución en la velocidad de respuesta a la hora de presentar los datos.

Esto quiere decir que primero se extrae la información y luego es manipulada, lo cual significa que puede haber una leve diferencia entre la información manipulada y el verdadero contenido de la base de datos.

## 4.1.- Importar y exportar datos.

Una forma de extraer información de la aplicación es exportando los datos. También es posible introducir información de manera masiva mediante la importación de datos.

La aplicación provee **mecanismos estándar para realizar estas dos operaciones**. **El formato usual de importación y exportación de datos es el CSV**, que es un formato de texto utilizado para representar datos en forma de tabla, en las que las columnas se separan por comas (o punto y coma en donde la coma es el separador decimal: España, Francia, Italia...) y las filas por saltos de línea.

**Se pueden importar datos hacia una sola tabla o hacia varias tablas**. En el caso de varias tablas se utiliza un separador para indicar a qué tabla pertenece cada uno de los datos existentes en el archivo a importar.

El proceso de exportación es igual de sencillo: desde el objeto que queramos exportar elegimos la opción de Exportar que aparecerá en el formulario correspondiente.

**Debes conocer**: [cómo se pueden importar datos a Odoo](https://www.odoo.com/documentation/13.0/es/applications/general/export_import_data.html#import-data-into-odoo).

# 5.- Evaluación del rendimiento y auditorías de acceso a los datos.

En este apartado vamos a ver **cómo consultar y tratar la información a nivel de administración de sistemas**. Cuando somos los encargados de administrar un sistema, debemos tener herramientas que nos permitan hacer un seguimiento de los datos que arroja el equipo servidor donde se encuentran las aplicaciones.

**El rendimiento del servidor puede disminuir o ser inexistente debido a diversos motivos**. Para investigar qué ocurre en cada caso, es necesario buscar información en los ficheros de registro o mensajes del sistema llamados logs, o ejecutar herramientas que permitan realizar un análisis y monitorización del rendimiento.

**Podemos obtener datos instantáneos del rendimiento del sistema** relativo al funcionamiento de los procesadores, de la memoria, de los dispositivos de entrada y salida, etc., pero también podemos recoger datos periódicamente y almacenarlos en ficheros históricos para consultarlos posteriormente. Estos datos nos proporcionan información muy importante sobre las posibles carencias y cuellos de botella de nuestro sistema.

**Existen diversas utilidades para recopilar y hacer un histórico del rendimiento y la actividad de los datos**. Estas utilidades recopilan la información del sistema, la almacenan por un periodo de tiempo y calculan los valores medios. **En cualquier momento se pueden tomar lecturas de los parámetros del servidor que se determinen**, para la resolución de problemas o bien simplemente para consultar el estado de nuestro servidor.

## 5.1.- Monitorización y evaluación del rendimiento.

En Linux, existen diversas herramientas de monitorización de rendimiento para servidores.

Por ejemplo, la herramienta ***sar*** (incluida en el paquete ***sysstat***).

sudo apt install sysstat

A partir de ahí ya podemos ver datos instantáneos, tan sólo tenemos que tener en cuenta que todos los comandos de la herramienta aceptan como parámetros:

* el **número de valores que queremos obtener** (intervalo)
* y **cada cuánto tiempo queremos que los capture** (segundos).

**Por ejemplo**: para obtener información de 3 valores sobre el uso del procesador, cada segundo, utilizamos la siguiente orden:

sar 1 3

Tabla

Descripción generada automáticamente

Los parámetros "1 3" indican que sar se ejecutará cada segundo un total de 3 veces, los valores de ejecución se muestran en filas separadas, y la última fila es la media aritmética de todos los valores.

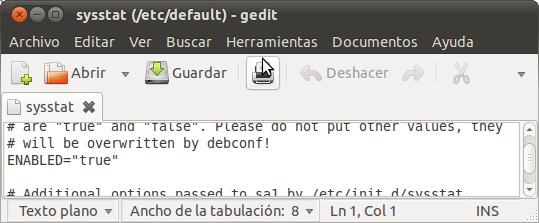
Para conocer la **lista completa de los parámetros que podemos utilizar con sar**, podemos consultar la ayuda del comando (**man sar**). Algunos ejemplos son los siguientes:

* **Procesador**: **sar -P ALL**: este comando muestra el uso del procesador por cada núcleo.
  + -P ALL indica que se quiere ver el detalle de todos los núcleos.
  + Si se quiere ver solo uno, se puede especificar el nº, por ejemplo, -P 0 para el primer núcleo.
  + El resultado tiene las mismas columnas que el comando sar -u[[1]](#footnote-1), pero para cada núcleo por separado.
* **Memoria**: **sar -r**: Este comando muestra el uso de la memoria y el espacio de intercambio (swap) del sistema.
  + El resultado tiene varias columnas que indican la cantidad de memoria disponible, usada, libre, en búfer, en caché y el espacio de intercambio usado y libre. Todos los valores están en kilobytes.
* **Interfaces de red**: **sar -n DEV**: Este comando muestra el uso de las interfaces de red del sistema.
  + El parámetro -n DEV indica que se quiere ver el detalle de cada dispositivo de red, como eth0, lo, etc.
  + El resultado tiene **varias columnas** que indican el número de paquetes recibidos y transmitidos por segundo, el tamaño medio de los paquetes, el porcentaje de errores y el porcentaje de paquetes descartados.
* **Discos**: **sar -d**: Este comando muestra el uso de los discos del sistema.
  + El resultado tiene varias columnas que indican el número de operaciones de lectura y escritura por segundo, el tamaño medio de las operaciones, el tiempo medio de espera y el porcentaje de utilización de los discos.

Texto

Descripción generada automáticamente

## 5.2.- Auditorías de control de acceso a los datos. Trazas del sistema (logs).



La actividad de los programas, en especial si son programas de entorno servidor, queda registrada en ficheros **logs**. Estos ficheros de trazas van **almacenando toda la actividad y eventos que ocurren en el equipo**: quién entra, qué comandos ejecuta, qué errores muestran las aplicaciones, etc.

En la mayoría de las distribuciones Linux, estos ficheros se guardan en el directorio **/var/log**. Para visualizar su contenido necesitamos permisos de **root**, o pertenecer a un grupo de usuarios con permisos para ver esos ficheros.

En este directorio **por ejemplo** se guarda el fichero ***syslog***, que guarda mensajes de trazas de demonios[[2]](#footnote-2) y otros programas como **cron**, **init**, **dhclient**, y algunos mensajes relacionados con el núcleo del sistema operativo.

[Cómo consultar los sucesos del sistema en la interfaz gráfica de Ubuntu 18.04 LTS](http://somebooks.es/consultar-los-sucesos-del-sistema-en-la-interfaz-grafica-de-ubuntu-18-04-lts/).

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

para hacer que **sar** capture información periódica debemos editar el fichero **/etc/default/sysstat** y la línea que empieza por **ENABLED** ponerla con el valor **ENABLED="true"**. Una vez hecho, reiniciamos el servidor de **sysstat** con la orden:

/etc/init.d/sysstat start

A partir de ahí el **script** **/usr/lib/sysstat/sa1**, que se ejecuta por defecto **cada 10 minutos** **empezará a recoger datos de rendimiento** (procesador, memoria, disco, red, etc) que se guardarán en el fichero de sistema **/var/log/sysstat/saXX**, donde **XX** indica el día del mes en el que nos encontramos. Por defecto, se guardan los datos de la última semana.

Para consultar de manera gráfica las trazas del sistema de **sar** podemos utilizar la herramienta Isag, disponible a través de Synaptic.

## 5.3.- Incidencias. Identificación y resolución.

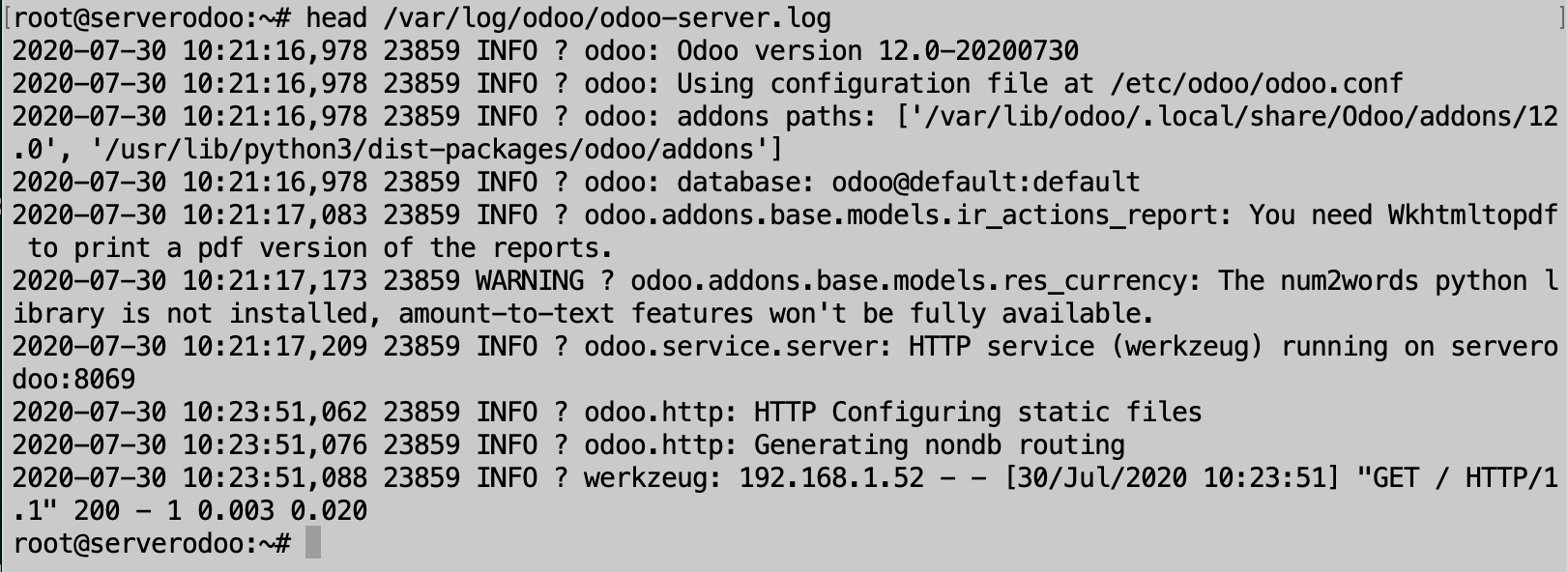
Lo interesante del registro de trazas del sistema es consultarlas para resolver posibles problemas, ya que generalmente, si una aplicación no funciona o no puede inicializarse, lo que hace es imprimir una traza de error, que puede poner sobre aviso de lo que está ocurriendo.

Dentro del directorio **/var/log** podemos encontrar las trazas del sistema de una aplicación en Ubuntu.

**Por ejemplo**: podemos querer visualizar la actividad del servidor de Odoo con el siguiente comando:

**$ sudo head /var/log/odoo/odoo-server.log**

El resultado de la instrucción serán las diez primeras líneas del fichero de log:



Donde podemos interpretar que el servidor de **Odoo** se encuentra ejecutándose y esperando conexiones en el puerto **8069**. Cualquier incidencia quedará reflejada en estos archivos, y será cuestión nuestra identificar la causa y una posible solución.

Texto

Descripción generada automáticamente

1. Sar -u: la diferencia entre poner -u y no ponerlo es que el parámetro -u indica que se quiere ver el uso del procesador en porcentaje, mientras que sin él, se muestra el uso del procesador en ticks o unidades de tiempo. [↑](#footnote-ref-1)
2. Demonio: proceso informático que se ejecuta en segundo plano para dar algún tipo de servicio. [↑](#footnote-ref-2)