Postgresql query

for get into db (postgres is default user)
sudo -i -u postgres psql

```
connect to database by a user in remote and localhost
psql -h host -U user -d database
```

create a user
CREATE USER odoo WITH PASSWORD 'odoo';

list user
SELECT usename FROM pg_user;

drop user
DROP USER tom;

drop database
DROP DATABASE newone;

to give passwd newly for postgres or changing passwd for users
ALTER USER postgres PASSWORD 'postgres';

to give privilage for create bd
ALTER USER odoo WITH CREATEDB;

to grant privilages
GRANT ALL PRIVILEGES ON DATABASE jerry to tom;

for selection of database
```
SELECT datname FROM pg_database WHERE datistemplate = false;
```
or \l
or \list

To switch databases:

```
\connect database_name
```

This lists tables in the current database
```
SELECT table_schema,table_name FROM information_schema.tables ORDER BY table_schema,table_name;
```

First, we need to drop template1. Templates can’t be dropped, so we first modify it so t’s an ordinary database:

```
UPDATE pg_database SET datistemplate = FALSE WHERE datname = 'template1';
```

1. Now we can drop it:

   ```
   DROP DATABASE template1;
   ```

2. Now its time to create database from template0, with a new default encoding:

   ```
   CREATE DATABASE template1 WITH TEMPLATE = template0 ENCODING = 'UNICODE';
   ```

3. Now modify template1 so it's actually a template:

   ```
   UPDATE pg_database SET datistemplate = TRUE WHERE datname = 'template1';
   ```

4. Now switch to template1 and VACUUM FREEZE the template:

   ```
   \c template1
   ```

   ```
   VACUUM FREEZE;
   ```

Problem should be resolved.

Error: permission denied to create database ??

You should do following steps:

1. ssh to your server where odoo is installed

2. sudo su postgres (the idea is to switch to postgresql user)

3. psql

4. ALTER USER odoo WITH CREATEDB;

5. Go back to web page and try create db once again

to retrieve all the rows of table `weather`, type:

```
SELECT * FROM weather;
```

Install Odoo 9 ERP on Ubuntu 14.04

Odoo (formerly known as OpenERP) is an open-source suite of business applications including: Customer Relationship Management, Sales Pipeline, Project Management, Manufacturing, Invoicing , Accounting, eCommerce and Inventory just to name a few. There are 31 main applications created by Odoo team and over 4,500+ developed by community members covering a wide range of business needs.

Once deployed, Odoo's flexibility allows the administrator to install any module combination and configure/customize them at will to satisfy business needs ranging from a small shop to an Enterprise Level Corporation.

This guide covers how to install and configure Odoo in just 35 minutes using Git source, so it will be easy to upgrade, maintain and customize.

Before You Begin

1. Complete the Getting Started guide.
2. Follow the Securing Your Server guide to create a standard user account, harden SSH access and remove unnecessary network services; this guide will use `sudo` wherever possible. Do **not** follow the *Configuring a Firewall* section–this guide has instructions specifically for an Odoo production server.
3. Log in to your Linode via SSH and check for updates using `apt-get` package manager.

```
sudo apt-get update && sudo apt-get upgrade
```

Open Corresponding Firewall Ports (no need mostly)

In this case we're using Odoo's default port 8069, but this could be any port you specify later in the configuration file.

```
sudo ufw allow ssh
sudo ufw allow 8069/tcp
sudo ufw enable
```

Install Database and Server Dependencies

Now we're going to install the PostgreSQL database and other necessary server

```
sudo apt-get install subversion git bzr bzrtools python-pip postgresql postgresql-server-dev-9.3 python-all-dev python-dev python-setuptools libxml2-dev libxslt1-dev libevent-dev libsasl2-dev libldap2-dev pkg-config libtiff5-dev libjpeg8-dev libjpeg-dev zlib1g-dev libfreetype6-dev liblcms2-dev liblcms2-utils libwebp-dev tcl8.6-dev tk8.6-dev python-tk libyaml-dev fontconfig
```

Create Odoo User and Log Directory

1. Create the Odoo system user:

```
sudo adduser --system --home=/opt/odoo --group odoo
```

2.Create the log directory:

```
sudo mkdir /var/log/odoo
```

note:::
In the scenario of running multiple Odoo versions on the same Linode you may want to use different users and directories for each instance.

Install Odoo Server Files from Source

1. Change to the Odoo directory, in our case:

```
cd /opt/odoo/
```

Clone the Odoo files on your server:

```
sudo git clone https://www.github.com/odoo/odoo --depth 1 --branch 9.0 --single-branch .
```

Note:;
Using Git allows great flexibility because any time a new upgrade ,is available you only need to pull that branch, You can even install a different one alongside the production version; just change the destination directory and the `--branch X.x` flag. Before performing any operation, remember to make a full backup of your database and custom files.

Specific Dependencies for Odoo Applications

Using `pip` instead of `apt-get` will guarantee that your installation has the correct versions needed. We'll also abstain of using Ubuntu's packaged versions of Wkhtmltopdf and node-less.

Install Python Dependencies

Install Python libraries using the following commands:

```
sudo pip install -r /opt/odoo/doc/requirements.txt
sudo pip install -r /opt/odoo/requirements.txt
```

Install Less CSS via nodejs and npm

1. Download the `nodejs` installation script from nodesource:

```
wget -qO- https://deb.nodesource.com/setup | sudo bash -
```

Now that our repository list is updated install `nodejs` using `apt-get`:

```
sudo apt-get install nodejs
```

Time to install a newer version of Less via `npm`:

```
sudo npm install -g less less-plugin-clean-css
```

Install Updated Wkhtmltopdf Version

1. Switch to the `/tmp/` directory:

```
cd /tmp/
```

Download the recommended version of wkhtmltopdf for Odoo server, currently **0.12.1**:

```
sudo wget http://download.gna.org/wkhtmltopdf/0.12/0.12.1/wkhtmltox-0.12.1_linux-trusty-amd64.deb
```

Install the package using `dpkg`:

```
sudo dpkg -i wkhtmltox-0.12.1_linux-trusty-amd64.deb
```

To function properly we'll need to copy the binaries to an adequate location:

```
sudo cp /usr/local/bin/wkhtmltopdf /usr/bin
sudo cp /usr/local/bin/wkhtmltoimage /usr/bin
```

Odoo Server Configuration

1. Copy the included configuration file to a more convenient location, changing its name to `odoo-server.conf`:

```
sudo cp /opt/odoo/debian/openerp-server.conf /etc/odoo-server.conf
```

Next we need to modify the configuration file. The finished file should look similar to this depending on your deploying needs:

/etc/odoo-server.conf

> dont use echo in dockerfile.....copy the files into /etc and use sed command for replace this entry...dont provide any extra path with error (addons...etc)

```
[options]
admin_passwd = admin
db_host = False
db_port = False
db_user = odoo
db_password = <PostgreSQL_user_password>
```

`admin_passwd = admin` This is the password that allows database operations.

- `db_host = False` Unless you plan to connect to a different database server address, leave this line untouched.
- `db_port = False` Odoo uses PostgreSQL default port 5432, change only if necessary.
- `db_user = odoo` Database user, in this case we used the default name.
- `db_password =` The previously created PostgreSQL user password.
- `addons_path =` We need to modify this line to read: `addons_path = /opt/odoo/addons`. Add `</path/to/custom/modules>` if needed.
- We need to include the path to log files adding a new line: `logfile = /var/log/odoo/odoo-server.log`.
- Optionally we could include a new line specifying the Odoo Frontend port used for connection: `xmlrpc_port = 8069`. This only makes sense if you're planning to run multiple Odoo instances (or versions) on the same server. For normal installation you could skip this line and Odoo will connect by default to port 8069.

Odoo Boot Script

Next step is creating a boot script called `odoo-server` to gain control over Odoo's behavior and use it at server startup and shutdown.

/etc/init.d/odoo-server

```
#!/bin/sh
### BEGIN INIT INFO
# Provides: odoo-server
# Required-Start: $remote_fs $syslog
# Required-Stop: $remote_fs $syslog
# Should-Start: $network
# Should-Stop: $network
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: Odoo ERP
# Description: Odoo is a complete ERP business solution.
### END INIT INFO

PATH=/bin:/sbin:/usr/bin
# Change the Odoo source files location according your needs.
DAEMON=/opt/odoo/openerp-server
# Use the name convention of your choice
NAME=odoo-server
DESC=odoo-server

# Specify the user name (Default: odoo).
USER=odoo

# Specify an alternate config file (Default: /etc/odoo-server.conf).
CONFIGFILE="/etc/odoo-server.conf"

# pidfile
PIDFILE=/var/run/$NAME.pid

# Additional options that are passed to the Daemon.
DAEMON_OPTS="-c $CONFIGFILE"

[ -x $DAEMON ] || exit 0
[ -f $CONFIGFILE ] || exit 0

checkpid() {
[ -f $PIDFILE ] || return 1
pid=`cat $PIDFILE`
[ -d /proc/$pid ] && return 0
return 1
}

case "${1}" in
start)
echo -n "Starting ${DESC}: "

start-stop-daemon --start --quiet --pidfile ${PIDFILE} \
--chuid ${USER} --background --make-pidfile \
--exec ${DAEMON} -- ${DAEMON_OPTS}

echo "${NAME}."
;;

stop)
echo -n "Stopping ${DESC}: "
```

```
start-stop-daemon --stop --quiet --pidfile ${PIDFILE} \
--oknodo

echo "${NAME}."
;;

restart|force-reload)
echo -n "Restarting ${DESC}: "

start-stop-daemon --stop --quiet --pidfile ${PIDFILE} \
--oknodo

sleep 1

start-stop-daemon --start --quiet --pidfile ${PIDFILE} \
--chuid ${USER} --background --make-pidfile \
--exec ${DAEMON} -- ${DAEMON_OPTS}

echo "${NAME}."
;;

*)
N=/etc/init.d/${NAME}
echo "Usage: ${NAME} {start|stop|restart|force-reload}" >&2
exit 1
;;
esac

exit 0
```

Odoo File Ownership and Permissions

1. Change the `odoo-server` file permissions and ownership so only **root** can write to it, while the **odoo** user will only be able to read and execute it.

```
sudo chmod 755 /etc/init.d/odoo-server
sudo chown root: /etc/init.d/odoo-server
```

Since the **odoo** user will run the application, change its ownership accordingly:

```
sudo chown -R odoo: /opt/odoo/
```

We should set the **odoo** user as the owner of log directory as well:

```
sudo chown odoo:root /var/log/odoo
```

Finally, we should protect the server configuration file changing its ownership and permissions so no other non-root user can access it:

```
sudo chown odoo: /etc/odoo-server.conf
sudo chmod 640 /etc/odoo-server.conf
```

Testing the Server

1. It’s time to check that everything is working as expected; let’s start the Odoo server:

```
sudo /etc/init.d/odoo-server start
```

Let's take a look at log file to verify that no errors occurred:

```
cat /var/log/odoo/odoo-server.log
```

Now we can check if the server stops properly too:

```
sudo /etc/init.d/odoo-server stop
```

Enter the same command as you did in Step 2:

```
cat /var/log/odoo/odoo-server.log
```

Running Boot Script at Server Startup and Shutdown

1. If the Odoo server log doesn't indicate any problems, we can continue and make the boot script start and stop with the server:

```
sudo update-rc.d odoo-server defaults
```

It's a good idea to restart our Linode to see if everything is working:

```
sudo shutdown -r now
```

Once restarted, verify the log file again:

```
cat /var/log/odoo/odoo-server.log
```

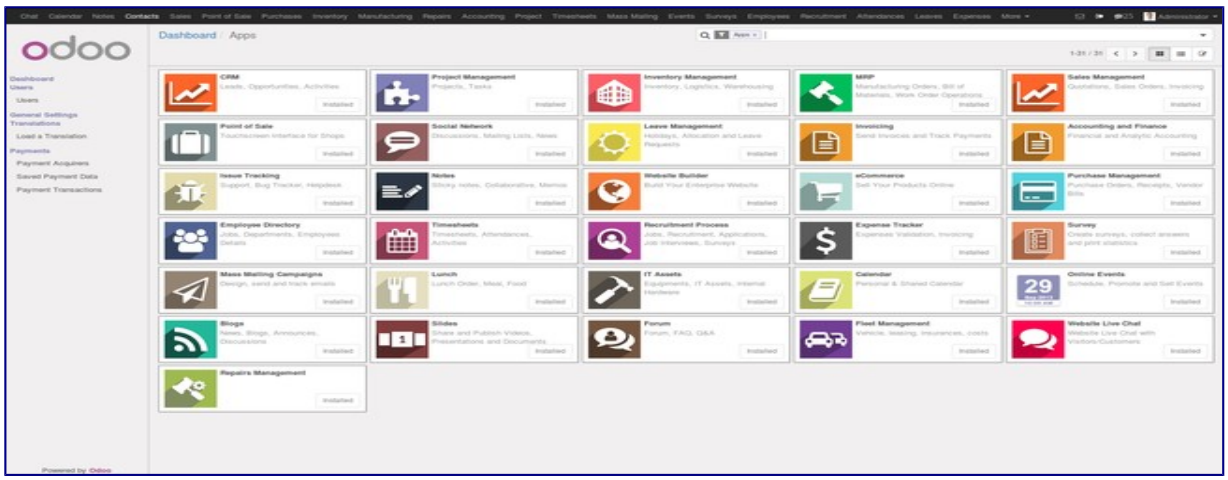
Testing Odoo Frontend

1. Open a new browser window and enter your IP address, followed by `:8069` (to indicate port 8069) in the address bar:

```
http://example_ip:8069
```

A screen similar to this will show:

1. Congratulations, now you can create your first database and start using Odoo!



More Information

You may wish to consult the following resources for additional information on this topic. While these are provided in the hope that they will be useful, please note that we cannot vouch for the accuracy or timeliness of externally hosted materials.

- Odoo User Documentation