How To

Install and Configure ESP on Ubuntu

**Revision History**

| Version Number | Modification Date | | By | | Description of Changes |
| --- | --- | --- | --- | --- | --- |
| 1.0 | 16 Sep 2012 | R. Schaaf | | Initial version | |
| 1.1 | 18 Sep 2012 | R. Schaaf | | Create the /srv/esp30/data/fake directory | |
| 1.2 | | 01 Feb 2013 | B. Zambarano | | Minor updates | |
| 1.3 | | 7/1/2014 | C. Chacin | | Updates for 3.1 and merging all installation of prerequisites. | |
| 1.4 | | 7/3/2014 | C. Chacin | | Adding apache and sql backup | |
| 1.5 | | 7/8/2014 | C. Chacin | | Finalizing all apache configuration and formatting doc | |
| 1.6 | | 8/21/2014 | C. Chacin | | Adding folder structure configuration and settings in applications.ini | |
| 1.7 | | 8/28/2014 | C. Chacin | | Added how to clean data and reload | |
| 1.8 | | 05 Oct 2014 | K. Eberhardt | | Added notes for dev install and modified some commands in sections 4-5 | |
| 1.9 | | 16 Jan 2015 | K. Eberhardt | | Updated some commands in section 8 | |
| 1.10 | 17 Nov 2016 | B. Zambarano | | Updates to repository (git) | |
| 1.11 | 15 Feb 2017 | K. Eberhardt | | Modifications to installation steps and commands | |
| 1.12 | 13 Jul 2017 | B. Zambarano | | Additions of steps describing data ETL, configuration and validation | |
| 1.13 | 29 Nov 2017 | K. Eberhardt | | Rewrite of Apache section. Reformatting. Rewrite of db backups section. Rewrite of daily batch process. Additional minor changes throughout. | |

Contents

[1 Overview 1](#_Toc499733511)

[2 Prerequisites 1](#_Toc499733512)

[3 Install ESP 1](#_Toc499733513)

[4 Create the ESP Database and ESP Database User 2](#_Toc499733514)

[5 Create the Directories and Files Expected by ESP 3](#_Toc499733515)

[6 Configure the ESP Application Settings 3](#_Toc499733516)

[7 Setting up the data feed 5](#_Toc499733517)

[8 Using ESP 5](#_Toc499733518)

[9 Set up the Apache Web Server 7](#_Toc499733519)

[10 Configure Database Backups 9](#_Toc499733520)

[11 Configure ESP Daily Status Reports 10](#_Toc499733521)

[12 Configure ESP Software Tools Daily Run 11](#_Toc499733522)

[13 Setting Up Basic Disease Detection 12](#_Toc499733523)

[14 How to Clean Up and Reload ESP Data 12](#_Toc499733524)

[Appendix A PostgreSQL database backup 15](#_Toc499733525)

# Overview

This document describes the procedure for configuring ESP 3.3 on Ubuntu Linux. The versions of software used to prepare these instructions are:

Linux: Ubuntu 16.04 Server Edition (64-bit)  
PostgreSQL: 9.5  
ESP: 3.3, running in a virtual environment using  
 Python 2.7  
 Django 1.4  
  
The installation steps are to:

* Install ESP
* Create an esp database and database user
* Create directories and files expected by ESP
* Configure the application settings

This document may be used a guide for installing ESP on other Linux systems, but keep in mind that there are differences between Linux systems, particularly in package management and user creation and permissions.

Unless otherwise specified, all commands are executed from the Linux bash shell prompt. It is assumed the installer has sudo privileges.

# Prerequisites

This installation is for an Ubuntu Linux 16.04 server with SSH for remote logins. The bash shell should be the default.

This installation expects that the git software version control system is installed.

The ESP installation will require either a local SMTP service, or an email account on an accessible SMTP server.

# Install ESP

1. As the system administrator, create the esp user. Standard ESP installation will be to /srv/esp/prod or /srv/esp/test. Use /srv/esp as the esp home directory.  
     
   sudo useradd -d /srv/esp/ -m -p *PASSWORD* esp
2. Add the user to the 'sudo' group  
     
   sudo adduser esp sudo
3. Switch to the esp user  
     
   sudo su esp
4. Clone the esp repository in /srv/esp/[prod or test]  
     
   cd ~  
   git clone https://bitbucket.org/Commoninf/esp.git [prod or test]  
     
   as in  
     
   git clone https://bitbucket.org/Commoninf/esp.git prod  
   or   
   git clone https://bitbucket.org/Commoninf/esp.git test
5. Checkout the desired branch and give it a “local” branch name. Unless you specifically require an earlier release, use the latest tagged release branch  
     
   cd /srv/esp/[prod or test]

git checkout *v3.3.[x]* -b *version3.3.[x]*

1. Setup ESP Dependencies by installing the packages that ESP depends on  
     
   sudo ./install-system-dependencies.sh  
     
   If errors are encountered, manually edit the file and make modifications to the versions as required. (For non-Ubuntu Linux systems, inspect this script to determine what dependencies to install manually).
2. Install ESP  
     
   ./install.sh
3. Install any disease detection plugins you require:  
     
   ./setupPlugins.sh

# Create the ESP Database and ESP Database User

1. As the system administrator, create an “esp” role in the PostgreSQL database by entering the following at the command prompt:

sudo -u postgres createuser -P -s esp  
  
Enter your sudo password, and in response to the “Enter password for new role” and “Enter it again” prompts, enter the password for the new esp user.

The options passed into the createuser command control the following:

**-P**: assign a password to the new role (you should be prompted)  
**-s**: role will be superuser

If the postgres database will contain databases other than those managed by ESP, you may choose for security reasons to not provide superuser. Some distributed ESP SQL reports that use the COPY command will not work.

1. Create an “esp” database by entering the following at the command prompt:  
     
   sudo -u postgres createdb -O esp esp  
     
     
   The options passed into the createdb command control the following:

**-O esp**: database user to own the new database

# Create the Directories and Files Expected by ESP

1. As the ESP user, create the directories and files expected by ESP by entering:  
     
   mkdir -p /srv/esp/data/{case\_reports,epic,load\_reports}  
   mkdir -p /srv/esp/data/epic/{archive,incoming,error}  
   mkdir –p /srv/esp/data/fake  
   mkdir -p /srv/esp/scripts  
   mkdir -p /srv/esp/logs

Use of the folder name “epic” is vestigial. These folders are used for text data from any source.

1. Create the ESP Log File  
     
   sudo touch /var/log/esp.log  
   sudo chown esp:esp /var/log/esp.log  
   sudo chmod 666 /var/log/esp.log

# Configure the ESP Application Settings

1. Create initial versions of ESP’s **application.ini** and **secrets.ini** configuration files:

cd /srv/esp/[prod or test]

./bin/esp makeini

These files will be created in the /srv/esp/[prod or test]/etc/ folder.

1. Edit the secrets.ini file:  
     
   vi ./etc/secrets.ini

For the **database\_password**, enter the password for the “esp” database role.  
  
For the **secret\_key**, enter a random string of at least 32 characters.

1. Edit the **application.ini** file:  
     
   vi ./etc/application.ini  
     
   In the **[General]** section, edit the following settings:  
    data\_folder = /srv/esp/data  
    icd10\_support = True  
     
   In the **[Database]** section, edit the following settings:  
    db\_name = esp  
    username = esp  
     
   In the **[Web**] section, edit the following settings:  
    static\_url = /media/static**/**  
     
   In the **[ETL**] section, edit the following settings:  
    load\_report\_dir = /srv/esp/data/load\_reports/

In the **[Logging]** section, edit the following setting:  
 log\_file = /var/log/esp.log

In the **[HEF]** section, edit the following setting:  
 thread\_count = 0  
  
Other settings may be modified as necessary.

1. As the ESP user, from the ESP installation directory, populate the media/static folder:  
     
   ./bin/esp collectstatic
2. As the ESP user, from the ESP installation directory, initialize the ESP database by entering:  
     
   ./bin/esp syncdb  
     
   When prompted to define a superuser, respond as follows:  
   Would you like to create one now? (yes/no): yes  
   Username (Leave blank to use 'superuser'): esp  
   E-mail address: <your email address>  
   Password: <password for the esp superuser>  
   Password (again): <password for the esp superuser>  
     
   Ignore the “IntegrityError: insert or update on table "conf\_reportabledx\_code" violates foreign key constraint "conf\_reportabledx\_code\_dx\_code\_id\_fkey" if received.
3. Load the static data for the ESP application by entering:  
     
   ./bin/esp loaddata ESP/static/fixtures/dx\_code.json  
   ./bin/esp loaddata ESP/static/fixtures/loinc.json

./bin/esp loaddata ESP/static/fixtures/ndc.json

1. Run syndb again to install the objects from the loaded static data. No error should be received.  
     
   ./bin/esp syncdb
2. Test that basic web services are working by running the web server built into Django by entering:  
     
   ./bin/esp runserver  
     
   Browse to http://localhost:8000 and verify that you are able to log into the ESP application using the superuser account created earlier.

# Setting up the data feed

Each site will need a data feed that provides data to ESP in the standard text file format. Please see the document ESP\_Filespec.xlsx for details. For Epic sites, Commonwealth informatics can share SQL-based extract scripts for Epic Clarity, or MUMPS scripts for Epic Cache. These must be modified to conform to site-specific Epic configuration.

There are two data feed steps:

1. Set up the historic data extract and load. The ESP database should have at least two years of data going back from the present in order to make good determinations of disease states for conditions that may be chronic.
2. Set up the nightly data extract and load. This is for ongoing data extraction and loading

The ESP command “load\_epic” is used to load the data file. ESP was developed around Epic systems, and this name is vestigial.

These are highly site-specific activities and must be designed, developed and tested at each site. Using Commonwealth-provided scripts will greatly reduce the time for this task, but for systems starting from the ESP filespec Excel spreadsheet, expect at least 4 weeks of person-effort.

# Using ESP

**ESP Commands**

All ESP commands take this form:

$ $ESP\_HOME/bin/esp command [--argument\_one {optional parameters}] [--argument\_two]

(Most commands have single character versions of arguments, used with a single dash)

For a full listing of available commands:

$ $ESP\_HOME/bin/esp help

For help with a specific command:

$ $ESP\_HOME/bin/esp *command* --help

**Load EMR data**

A typical command to load data might look like this:

$ $ESP\_HOME/bin/esp load\_epic

The use of Epic in the command name is vestigial. This command loads text data from a set of files in the ESP defined input format. See the document “ESP\_Filespec.xlsx”.

This command will take files from the configured data directory ./epic/incoming and load them to the ESP database tables. Details are written to the esp.log file, and table specific load results are written to reports in load\_reports directory.

**Code mapping**

Each individual EMR system uses a unique set of codes to designate tests. We refer to these as abstract labs. Once EMR lab data is loaded to ESP, the native codes of relevant lab tests must be mapped to the named abstract labs used by ESP's disease detection logic. Mapping is stored in the the table conf\_labtestmap.

To perform lab mapping, run:

$ $ESP\_HOME/bin/esp concordance

The concordance command populates a table emr\_labtestconcordnace, which lists all unique native lab test names and code currently loaded in the ESP emr\_labresults table. These are available for efficient searches for mapping. A tool is provided in the Administrative web interface, (setup described in next section), In the menu bar, look under *Administration --> Unmapped Lab Tests Report*. This form page uses the set of search strings provided for each disease detection plugin to detect potentially unmapped lab tests. In the Unmapped Lab Tests Report interface, the user can either map a lab, or mark it to be ignored. Mapped and ignored labs do not appear in the Report again.

**Detecting cases**

Once all labs are mapped using the Admin UI interface, ESP divides the task of detecting disease cases into two distinct parts. First, a uniform table of events is generated from raw medical record data by *HEF*, the Heuristic Events Framework.

$ $ESP\_HOME/bin/esp hef

Second, a table of cases is generated by searching the events table for patterns indicating disease. This function is carried out by *Nodis*, the NOtifiable DISeases framework.

$ $ESP\_HOME/bin/esp nodis

Additional options for both commands are available by passing the '--help' flag.

**Reporting cases**

The process for electronically reporting cases is heavily dependent upon the requirements, both technical and functional, of the intended recipient. ESP provides two report generation commands: case\_report and lab\_report. The case\_report command generated hl7 2.3.1 version lab report message, the lab\_report command generated hl7 2.5.1 version lab report messages. Both require extensive coding for specific recipient requirements. Unfortunately, this mapping process is not currently well documented. We are working on developing a document set for this purpose.

# Set up the Apache Web Server

Set up the Apache web server following the steps specified below. These steps should be executed from an interactive shell that is running as root. These steps are specific to Ubuntu Linux distributions.

1. If not already installed, install the apache package including mod wsgi and start the service.

sudo apt-get install apache2 libapache2-mod-wsgi

1. Make a copy of django.wsgi.sample located in the $ESP\_HOME/share folder to the $ESP\_HOME/etc directory and then edit it as described below:

sudo su esp  
  
cp /srv/esp/prod/share/django.wsgi.sample /srv/esp/prod/etc/django.wsgi  
  
vi /srv/esp/prod/etc/django.wsgi

Update the contents of the file to match the following. Replace directory paths to match your environment where necessary.

import os

import sys

os.environ['DJANGO\_SETTINGS\_MODULE'] = 'ESP.settings'

# Enable this to prepend your ESP src folder to the beginning of PYTHONPATH, in

# case an older version of Django is installed system-wide.

sys.path.insert(0, '/srv/esp/prod/')

execfile('/srv/esp/prod/bin/activate\_this.py')

import django.core.handlers.wsgi

application = django.core.handlers.wsgi.WSGIHandler()

1. Create the Apache configuration file. The name should match the name of your server  
     
   sudo vi /etc/apache2/sites-available/yourservername.conf  
     
   For example:  
     
   sudo vi /etc/apache2/sites-available/esp-server.conf  
     
     
   Add the following contents and update highlighted sections as necessary to match your environment:

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

#

# Sample Apache configuration file for ESP Health

#

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

WSGIRestrictStdout Off

WSGIDaemonProcess esp

WSGIScriptAlias / /srv/esp/prod/etc/django.wsgi

# Change "yourdomain.com" and “youremail” to match your environment

<VirtualHost yourdomain.com:80>

ServerName yourdomain.com

ServerAdmin youremail@yourdomain.com

Alias /static/ /srv/esp/prod/ESP/media/static/

Alias /media /srv/esp/prod/ESP/media

#

# WSGI

#

<Location "/media/">

SetHandler None

Require all granted

</Location>

<LocationMatch "\.(jpg|gif|png)$">

SetHandler None

Require all granted

</LocationMatch>

### Block all trace requests (all vHosts need this!!)

### The following code should be included every VirtualHost.

### It returns a Forbidden on any TRACE or TRACK request.

### TRACE requests are used for debugging, they tell the server

### to return a text version of the REQUEST. This is not

### needed and can be used for abuse/cross-site-scripting

###

### See: http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html

### for more about TRACE

###

RewriteEngine On

RewriteCond %{REQUEST\_METHOD} ^(TRACE|TRACK|CONNECT)

RewriteRule .\* - [F]

<Directory proxy:\*>

Require all denied

</Directory>

<IfModule mod\_proxy.c>

ProxyRequests Off

</IfModule>

</VirtualHost>

1. Add required directory configurations to apache.conf  
     
   sudo vi /etc/apache2/apache2.conf  
     
   Beneath the following default section:  
     
   <Directory /var/www/>

Options Indexes FollowSymLinks

AllowOverride None

Require all granted

</Directory>  
  
Add the following:  
  
<Directory /srv/esp/prod/etc/>

Options Indexes FollowSymLinks

AllowOverride All

Require all granted

</Directory>

<Directory /srv/esp/prod/ESP/media/>

Options Indexes FollowSymLinks

AllowOverride All

Require all granted

</Directory>

1. Enable the required Apache modules.   
     
   sudo a2enmod wsgi   
   sudo a2enmod rewrite  
   sudo a2ensite yourservername (this should match the server name defined above)
2. Restart Apache   
     
   sudo service apache2 restart

# Configure Database Backups

1. Create the logrotate configuration file  
     
   sudo su   
   mdkir -p /srv/esp/backup  
   vi /srv/esp/backup/postgresql-esp.logrotate  
     
   Populate the file by copying the configuration shown in Appendix A
2. Modify the permissions and ownership on the file/directory  
     
   chown -R postgres.postgres /srv/esp/backup   
   chmod 644 /srv/esp/backup/postgresql-esp.logrotate
3. Create initial versions of the “log” files  
     
   sudo su postgres  
   cd /srv/esp/backup

touch esp.daily.dump  
touch esp.weekly.dump  
touch esp.monthly.dump

1. While still the potgres users, add the following entry to the postgres user crontab to run the database backups. Choose a time that does not conflict with daily ESP processing. The example below is set to run at 1pm each day. The script referenced will be created in the next step.  
     
   crontab -e

*# m h dom mon dow command*

0 13 \* \* \* /srv/esp/scripts/db\_backup.sh

1. Create the **/srv/esp/scripts/db\_backup.sh** file   
     
   sudo su esp  
     
   vi /srv/esp/scripts/db\_backup.sh  
     
   Populate it with the following:

#!/bin/bash

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

# @author: Rich Schaaf

# @organization: Commonwealth Informatics, Inc

# @copyright: (c) 2013 Commonwealth Informatics, Inc.

# @license: LGPL

#

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

/usr/sbin/logrotate /srv/esp/backup/postgresql-esp.logrotate

1. Set the permissions on script as follows:

chmod 755 /srv/esp/scripts/db\_backup.sh

# Configure ESP Daily Status Reports

ESP will send a daily email showing information about the cases created and transmitted. Additionally, it will show unmapped lab tests and other information related to ESP.

* + - 1. Configure the application.ini file   
           
         sudo su esp  
           
         sudo vi /srv/esp/prod/etc/application.ini

In the **[General]** section at the top of the file, configure the managers to match your desired recipients.

# Managers are emailed a copy of the daily status report, if it is enabled in the Batch section  
managers = [recip1@youremail.com](mailto:recip1@youremail.com), [recip2@youremail.com](mailto:recip2@youremail.com),  
  
\*\*Note the ending comma\*\*

In the **[Email]**section, configure your parameters. Set the host, port, server\_email, and default\_from\_email to match your configuration.  
  
[Email]  
host = your\_mail\_server\_or\_ip\_address  
port = your\_port\_number (i.e. 25)  
username = ""  
use\_tls = False  
server\_email = [esp-no-reply@youremail.com](mailto:esp-no-reply@youremail.com)  
default\_from\_email = [youremail@youremail.com](mailto:youremail@youremail.com)  
subject\_prefix = "[ESP] "  
  
\*\*Note: If your server requires authentication, you will want to enter the username here and the password in the secrets.ini file  
  
In the **[Batch]** section, set:  
  
mail\_status\_report = True

2. To test if it is working, run the following command from the command line. It just sends the email and doesn't change any data, etc.

/srv/esp/prod/bin/esp status\_report --send-mail

# Configure ESP Software Tools Daily Run

A sample daily\_batch.sh script can be found within the core system repository. If using the default paths, this file will be found in the /srv/esp/prod/share folder. This is the script that is utilized to perform the daily processes for ESP (data load, event creation, case creation, etc).

This is just a sample script and should be modified/customized for your environment.

To enable daily data loading and processing, please the customized daily\_batch.sh file in the /srv/esp/scripts folder.

As the ESP user, add a cron entry in your system:

sudo su esp  
  
crontab – e

Add a similar text to the crontab with your desired time to run:

# ESP Production Daily Run

#  
30      22      \*       \*       \*     /srv/esp/scripts/daily\_batch.sh

Save and quit

To verify the new crontab entry type:

crontab –l

# Setting Up Basic Disease Detection

ESP is distributed with a number of Python plugins for disease detection. Each disease has a separate plugin. These plugins are installed via an interactive text-based screen started from the bash shell prompt:

$ /srv/esp/prod/setupPlugins.sh

This is also described in section 3 above. The plugins are installed into ~/esp/src/[disease name directories]

Configuring the disease plugins is a four-step process:

1. Run the esp concordance command

$ $ESP\_HOME/bin/esp concordance

1. The concordance command populates a table emr\_labtestconcordance, which lists all unique native lab test names and codes currently loaded in the ESP emr\_labresults table. These are available for efficient searches for mapping. A tool is provided in the Administrative web interface. From the menu bar, browse to *Administration --> Unmapped Lab Tests Report*. This form page uses the set of search strings provided for each disease detection plugin to detect potentially unmapped lab tests. In the Unmapped Lab Tests Report interface, the user can either map a lab or mark it to be ignored. Mapped and ignored labs do not appear in the report again.

The lab concepts required by the plugin must be mapped to the appropriate local labs. The required lab concepts are determined by inspection of the plugin file [disease\_name].py. This step requires site support from staff familiar with local lab naming and the corresponding lab test.

1. Run the esp commands hef and nodis:

$ $ESP\_HOME/bin/esp hef

$ $ESP\_HOME/bin/esp nodis

1. Review the resulting cases and perform appropriate clinical validation to confirm that data configuration is correct and working. Cases are available for review in the ESP administrative web interface. Case validation requires site support from staff with the ability to do case review in the EHR.

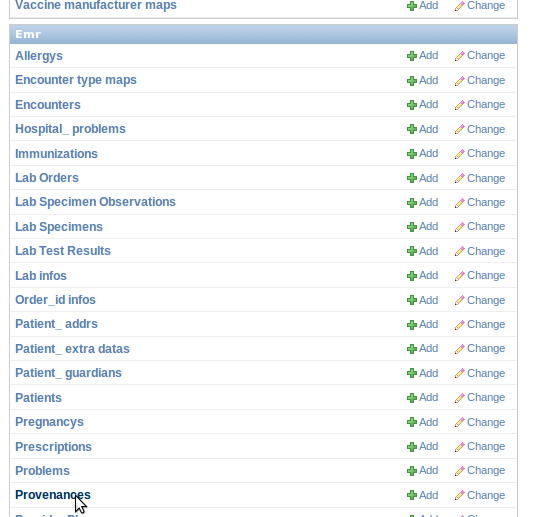
# How to Clean Up and Reload ESP Data

Sometimes it is helpful to know how to clean all the data loaded and start over again. There are several procedures that can be run to accomplish this task but the simplest is to delete all the provenance entries from the administration UI except the first two. This method will automatically delete all of the related records in the db.

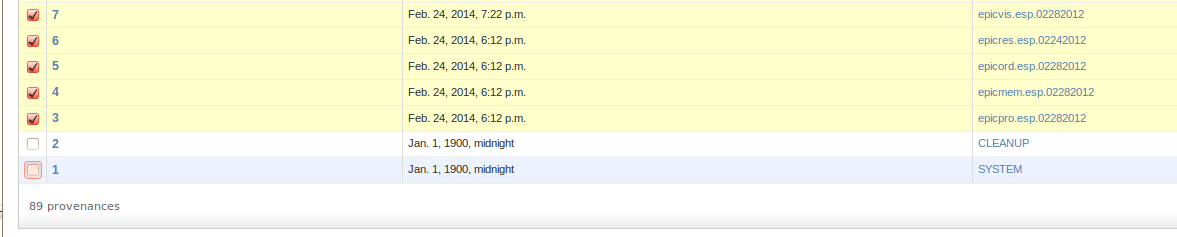
Below are the steps and screen shots to perform this task:

1. From the ESP UI Select Administration -> Site Administration from the menu

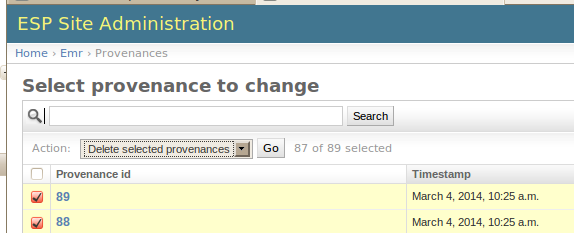
1. Click on “Provenances” from the “Emr” section



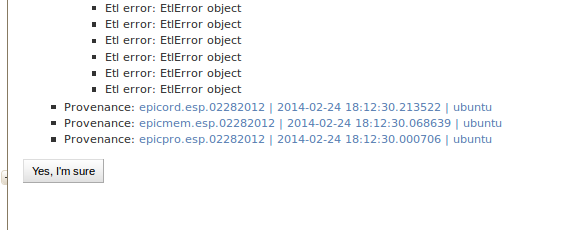
1. Select all records by clicking on the top left check box next to the Provenance id label and then **unselect** the two checkboxes for the record 1 and 2 corresponding to CLEANUP and SYSTEM



1. From the same screen, select “Delete selected provenances” from the “Action” drop down. Next, click “Go” via the button to the right of the dropdown.



1. This will calculate all the related records and elements in the database related to those loaded files and will display a summary of all the records that will be deleted.  
     
   A confirmation window with the prompt “Are you sure you want to delete the selected provenances? All of the following objects and their related items will be deleted.” will appear.   
     
   To continue, scroll down to the bottom and click on the “Yes, I am sure” button:

**

Appendix A PostgreSQL database backup

The following configuration should be saved as **/srv/esp/backup/postgresql-esp.logrotate**.

#

# Logrotate configuration for automatic daily backups of the PostgreSQL

# database 'esp' to '/srv/esp/backup/esp.daily.dump'.

#

# Please note, this configuration does not make any attempt to divine your

# database name from ESP's config files. Edit it if you are using something

# different than 'esp'. You may wish to run logrotate in the context of

# user 'postgres' to avoid dealing with DB authentication issues. The user

# running logrotate will need write permission in /srv/esp/backup.

#

/srv/esp/backup/esp.daily.dump {

daily

rotate 7

dateext

nomissingok

create

nocompress

nocopy

prerotate

test -x /usr/bin/pg\_dump || exit 0

sudo -u postgres /usr/bin/pg\_dump esp -F c > /srv/esp/backup/esp.daily.dump

endscript

}

/srv/esp/backup/esp.weekly.dump {

weekly

rotate 4

dateext

nomissingok

create

nocompress

nocopy

prerotate

test -x /usr/bin/pg\_dump || exit 0

sudo -u postgres /usr/bin/pg\_dump esp -F c > /srv/esp/backup/esp.weekly.dump

endscript

}

/srv/esp/backup/esp.monthly.dump {

monthly

rotate 12

dateext

nomissingok

create

nocompress

nocopy

prerotate

test -x /usr/bin/pg\_dump || exit 0

sudo -u postgres /usr/bin/pg\_dump esp -F c > /srv/esp/backup/esp.monthly.dump

endscript

}