# Django apogee Documentation

*Version 0.1*

**Paul Guichon, Georges Ciobotaru**

20 December 2015

Table des matières

This is a french application for french university. It's why the documentaion is in French.

Cette application a pour but de fournir les modèles et méthodes d'accès à certaines tables d'Apogée.

Elle fait partie d'une suite d'application Django nommée Duck-Duck mais peut être utilisée seule.

---

**Note :** Django ne supporte pas les clés primaires composites. C'est pourquoi certaines Modèle possèdent un équivalent NomInitialeCopy qui correspondent à la table initiale + un id.

L'id correspond à la concaténation des clés primaires.

Les Modèles initiaux ne doit être utilisés que dans certains cas de lecture (c'est à dire pour la copie des données).

Voir la documentation des tables concernées

---

Table des matières :

CHAPITRE 1

# Installation

L'installation décrite ci-dessous vaut pour une ubuntu server 13.10. Dans le cas d'un autre système d'exploitation veuillez adapter les commandes ci-dessous.

## 1.1 Prérequis

Il faut avoir un accès en lecture pour apogée pour la liste des tables suivantes : .....

Et un accès en écriture sur les tables OPI.

Toutes les commandes ci-dessous se font en tant qu'utilisateur normale. Il faut donc tenir compte des sudo pour les commandes en temsp que superutilisateur, et sans le sudo pour les commandes en tant qu'utilisateur normal.

**Avertissement :** L'installation du virtualenv ne doit pas se faire en tant que root pour des raisons de sécurité. De plus pour tous les paquets pythons, il faut passer par le gestionnaire pip et non par le gestionnaire de paquet du système.

## 1.2 Commandes

1. Installation de pip :

   ```
   sudo apt-get install python-setuptools
   sudo easy_install pip
   ```

2. Installation de virtualenv

   ```
   sudo pip install virtualenv
   ```

3. Création du virtualenv

   ```
   mkdir .Envs
   cd .Envs
   virtualenv django_projet
   ```

   Pour rappel : VirtualEnv permet de faire une genre de sandbox pour les applications pythons. Pour activer le virtualenv

   ```
   source ~/.Envs/django_projet/bin/activate
   ```

4. Installation de postgresql

```
sudo apt-get install postgresql libpq-dev
```

Configuration de postgresql

```
sudo -i -u postgres
createuser -P mon_utilisateur
createdb -O mon_utilisateur -E UTF8 ma_database
exit
```

mon_utilisateur et ma_database serviront pour la connexion à la base postgresql

installation du connecteur python

```
sudo apt-get install python-dev
source ~/.Envs/django_projet/bin/activate
pip install psycopg2
```

Modifier votre fichier /etc/postgresql/x.x/main/pg_hba.conf et modifier la ligne local en remplacant peer par indent

```
sudo service postgresql restart
```

5. Installation d'instant client d'oracle :

   Télécharger sur http://www.oracle.com/technetwork/database/features/instant-client/index-097480.html la version 11.2 de instant client (nécessite la création d'un compte) Deux zip à télécharger : instantclient-basic-linux.ARCH.Version.zip et instantclient-sdk-linux.ARCH.Version.zip Il faut d'abord créer un compte. http://download.oracle.com/otn/linux/instantclient/11204/instantclient-basic-linux.x64-11.2.0.4.0.zip http://download.oracle.com/otn/linux/instantclient/11204/instantclient-sdk-linux.x64-11.2.0.4.0.zip une fois les zips téléchargés et uploader sur le serveur :

```
sudo mv instantclient-basic-linux.x64-11.2.0.4.0.zip /opt
sudo mv instantclient-sdk-linux.x64-11.2.0.4.0.zip /opt
sudo apt-get install unzip
cd /opt
sudo unzip instantclient-basic-linux.x64-11.2.0.4.0.zip
sudo unzip instantclient-sdk-linux.x64-11.2.0.4.0.zip
sudo rm instantclient-basic-linux.x64-11.2.0.4.0.zip
sudo rm instantclient-sdk-linux.x64-11.2.0.4.0.zip
cd instantclient_11_2/
sudo ln -s libclntsh.so.11.1 libclntsh.so
sudo ln -s libocci.so.11.1 libocci.so
```

   ensuite rajouté dans le .bashrc de son user

```
export ORACLE_HOME=/opt/instantclient_11_2
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME
```

   puis

```
source ~/.bashrc
sudo apt-get install libaio1
```

6. installation de cx_oracle

```
source ~/.Envs/django_projet/bin/activate
pip install cx_oracle
```

   si besoin (des fois pypi ne fonctionne pas très bien)

```
pip install cx_oracle  --allow-external cx-oracle  --allow-unverified cx-oracle
```

7. ajout de django_apogee

```
sudo apt-get install git
source ~/.Envs/django_projet/bin/activate
pip install git+https://github.com/fsx999/django_apogee.git
```

8. test de l'installation

```
cd ~
mkdir projet
cd projet
django-admin.py startproject test_projet
cd test_projet
chmod +x manage.py
```

on peut installer django-extensions et ipython pour améliorer la console de django

```
pip install django-extensions
pip install ipython
```

configuration du settings.py

```
# après INSTALLED_APPS
INSTALLED_APPS += (
'django_extensions',
'django_apogee',
)

# La connexion aux bases de donnée
DATABASES = {
  'default': {
      'ENGINE': 'django.db.backends.postgresql_psycopg2',
      'NAME': 'ma_base',
      'USER': 'mon_utilisateur',
      'PASSWORD': 'mon_password!',
      'HOST': 'localhost',  # Important pour postgresql
      'PORT': '',

  },
  'oracle': {
      'ENGINE': 'django.db.backends.oracle',
      'NAME': 'nom_de_la_base_oracle_apogee',
      'USER': 'utlisateur',
      'PASSWORD': 'password',
      'HOST': 'url_serveur_oracle',
      'PORT': 'port_oracle',
  },
}

./manage.py syncdb
./manage.py migrate
```

## 1.3 Test connexion

```
./manage.py test_connexion_apogee
```

si le test de connexion fonctionne, passez à la suite : *Initialisation de la base*

CHAPITRE 2

# Table Lecture Apogée

**Note :** Il faut avoir les droits de lecture sur les tables suivantes.

- ANNEE_UNI
- PAYS
- DEPARTEMENT
- SIT_FAM
- TYP_HANDICAP
- SIT_MIL
- COM_BDI
- TYP_HEBERGEMENT
- BAC_OUX_EQU
- MENTION_NIV_BAC
- TYP_ETB
- ETABLISSEMENT
- CAT_SOC_PFL
- QUOTITE_TRA
- DOMAINE_ACT_PFL
- SITUATION_SISE
- TYP_DIPLOME_EXT
- REGIME_PARENT
- MTF_NON_AFL_SSO
- SIT_SOCIALE
- BOURSE
- COMPOSANTE
- CENTRE_GESTION
- ETAPE
- ETP_GERER_CGE
- ELEMENT_PEDAGOGI
- ELP_LIBELLE
- DIPLOME
- CMP_HABILITER_VDI
- VERSION_DIPLOME
- VERSION_ETAPE
- VDI_FRACTIONNER_VET
- INDIVIDU
- ADRESSE
- INS_ADM_ETP

**Note :** Il faut avoir les droits de lecture et d'écriture sur les tables suivantes

— IND_OPI
— OPI_BAC
— ADRESSE_OPI

CHAPITRE 3

# Initialisation de la base

## 3.1 Problèmatique

Il est nécessaire de faire une copie des donées d'apogée.

Il y a plusieurs stratégies de copie et de synchronisation, la meilleurs étant d'implémenter ça du coté d'oracle.

Mais ici nous partons du principe que ce n'est pas possible et donc nous devons faire une copie de certaines données et une modification pour d'autres.

Une copie mais pour quoi faire ?

Premièrement les clés composites. Il faut copier les données et rajoutés un id qui est la concaténation des clés primaires.

Deuxièmement les données d'apogée sont instables, trop de règles effacent des données nécessaire au fonctionnement d'autres applications de la suite.

## 3.2 Initialisation

**Avertissement :** C'est une opération longue, environ 30 minutes

```
./manage.py initialisation_base
```

CHAPITRE 4

# APIS

## 4.1 factories

## 4.2 lookups

## 4.3 managers

**class** `django_apogee.managers.`**`DiplomeManager`**

**`diplome_composante_active`**()

**class** `django_apogee.managers.`**`EtapeCondiValideManager`**

**`get_query_set`**(**args*, ***kwargs*)
**`get_queryset`**()

**class** `django_apogee.managers.`**`EtapeCondiValideManagerOracle`**

**`get_query_set`**(**args*, ***kwargs*)
**`get_queryset`**()

**class** `django_apogee.managers.`**`EtapeManager`**

**`by_centre_gestion`**(*code_centre_gestion*)
**`by_composante`**(*code_composante*)

**class** `django_apogee.managers.`**`EtapeNonCondiValideManager`**

**`get_query_set`**(**args*, ***kwargs*)
**`get_queryset`**()
**`impayes`**()

**class** `django_apogee.managers.`**`EtapeNonCondiValideManagerOracle`**

**`get_query_set`**(**args*, ***kwargs*)
**`get_queryset`**()
**`impayes`**()

## 4.4 Models

### 4.4.1 Apogee

### 4.4.2 Info Individu

### 4.4.3 O.P.I

CHAPITRE 5

# Indices and tables

- *genindex*
- *modindex*
- *search*

Index des modules Python

## d

## B



## D



## E



## G



## I