Household objects database is used by Tabletop object recognition.

To install household objects database server, you can refer to this page. But I can give you steps.

To create server

1.Open a terminal, type

```
sudo apt-get install postgresql
```

2.After the installation, type the command below to become user postgres

```
sudo su - postgres
```

For example, it's like

```
rosfuerte@rosfuerte-K53SM:~$ sudo su - postgres
[sudo] password for rosfuerte:
postgres@rosfuerte-K53SM:~$
```

3.Next, type psql to enter postgre sql interpreter, it's like

```
postgres@rosfuerte-K53SM:~$ psql
psql (9.1.6)
Type "help" for help.

postgres=#
```

4.Then, add a new user willow by entering CREATE ROLE ...(Don't forget the semicolon)

```
postgres=# CREATE ROLE willow LOGIN CREATEDB CREATEROLE PASSWORD 'willow';
```

5.Leave the interpreter

```
postgres=# \q
postgres@rosfuerte-K53SM:~$
```

To make connection over TCP/IP works

1.Get the paths of the files(pg_hba.conf & postgresql.conf) you need to modify

```
postgres@rosfuerte-K53SM:~$ ps auxw | grep postgresql
postgres  1170  0.0  0.1 132084 10400 ?        S    09:53   0:00
/usr/lib/postgresql/9.1/bin/postgres -D /var/lib/postgresql/9.1/main -c
config_file=/etc/postgresql/9.1/main/postgresql.conf
postgres  4095  0.0  0.0  13612   916 pts/1    S+   11:32   0:00 grep postgresql
```

2.Modify pg_hba.conf(usually is /etc/postgresql/9.1/main/pg_hba.conf), You need to add these two lines in the bottom of this file

```
# Anybody through TCP/IP with password
host    all         all         0.0.0.0   0.0.0.0     md5
```

3.Modify postgresql.conf(usually is /etc/postgresql/9.1/main/postgresql.conf), add this line to the file

```
listen_addresses = '*'
```

4.Log out postgres user

```
exit
```

5.Reboot your computer

To add object database to server

1.Install PGAdmin3

```
sudo apt-get install pgadmin3
```

2.Launch PGAdmin3 by enter

```
pgadmin3
```

3.Press the plugin button on the upper-left of PGAdmin GUI

4.Enter the name(whatever name you want)

5.Set the host, enter "localhost"

6.Enter password, which is "willow"

7.When your GUI looks like this, press OK

8.Now you can see your server on Object Browser on the left, press right button of the mouse on "Databases" under the server

9.Choose "New Database..."

10.Set the name and the owner, like(you should follow the name in the picture below) 

11.Press right button of the mouse on the newly added database

12.Press "Restore..."

13.Download the database backup file by

```
svn export https://code.ros.org/svn/data/trunk/household_objects/household_objects-0.6_fuerte_prerelease_1.backup
```

14.Restore the file

15.Press "Restore"(Restore process need some time to complete), then you finished installing the household objects database

---

To use this database, you first need to create a yaml file, for example, I should create my_server.yaml

```
household_objects_database:
  database_host: localhost
  database_port: 5432
  database_user: willow
  database_pass: willow
  database_name: household_objects
```

To integrate the database with tabletop object detector, you should create a launch file(in this example, I created a package named ricky_tabletop_object_detector, I put my_server.yaml in ricky_tabletop_object_detector/yaml/, and put this launch file in ricky_tabletop_object_detector/launch/)

```
<launch>
  <!-- set stereo to true for narrow stereo, false for kinect -->
  <arg name="stereo" default="true" />
  <arg name="use_slip_controllers" default="false"/>
  <arg name="use_right_arm" default="true"/>
  <arg name="use_left_arm" default="true"/>
  <arg name="use_task_cartesian" default="false"/>
  <arg name="log_to_warehouse" default="false"/>
  <arg name="flatten_table" default="false"/>
  <arg name="kinect_frame_prefix" default="/head_mount_kinect" />
  <arg name="kinect_camera_name" default="head_mount_kinect" />
  <arg name="sim" default="false"/>

  <!-- client for object database running on remote server at Willow Garage -->
  <!-- DOES NOT WORK IN TRUNK RIGHT NOW -->
  <!--
  <include file="$(find
household_objects_database)/launch/objects_database_remote_client.launch"/>
  -->

  <!-- alternative option: database server running on a local machine -->
  <rosparam command="load" file="$(find
ricky_tabletop_object_detector)/yaml/my_server.yaml"/>
  <node pkg="household_objects_database" name="objects_database_node"
type="objects_database_node"
    respawn="true" output="screen"/>

  <!-- manipulation prerequisites -->
  <include file="$(find
pr2_object_manipulation_launch)/launch/pr2_manipulation_prerequisites.launch">
    <arg name="stereo" value="$(arg stereo)"/>
    <arg name="use_left_arm" value="$(arg use_left_arm)"/>
    <arg name="use_right_arm" value="$(arg use_right_arm)"/>
    <arg name="log_to_warehouse" value="$(arg log_to_warehouse)"/>
    <arg name="kinect_frame_prefix" value="$(arg kinect_frame_prefix)"/>
    <arg name="sim" value="$(arg sim)"/>
  </include>

  <!-- manipulation -->
  <include file="$(find
pr2_object_manipulation_launch)/launch/pr2_manipulation.launch">
    <arg name="use_slip_controllers" value="$(arg use_slip_controllers)"/>
    <arg name="use_left_arm" value="$(arg use_left_arm)"/>
    <arg name="use_right_arm" value="$(arg use_right_arm)"/>
    <arg name="use_task_cartesian" value="$(arg use_task_cartesian)"/>
    <arg name="sim" value="$(arg sim)"/>
  </include>

  <!-- tabletop collision map processing -->
  <node pkg="tabletop_collision_map_processing"
name="tabletop_collision_map_processing"
        type="tabletop_collision_map_processing_node" respawn="false"
```

```
output="screen"/>
  <param name="tabletop_collision_map_processing/get_model_mesh_srv"
         value="/objects_database_node/get_model_mesh" />
  <param name="tabletop_collision_map_processing/static_map_cloud_name"
value="full_cloud_filtered" />

  <!-- tabletop segmentation and object recognition -->
  <include file="$(find tabletop_object_detector)/launch/tabletop_complete.launch">
      <arg unless="$(arg stereo)" name="tabletop_segmentation_points_input"
value="$(arg kinect_camera_name)/depth_registered/points"/>
      <arg if="$(arg stereo)" name="tabletop_segmentation_points_input"
value="narrow_stereo_textured/points2"/>
      <arg name="flatten_table" value="$(arg flatten_table)"/>
  </include>

</launch>
```