

openCRX Installation Guide for PostgreSQL 8

Version 2.1



www.opencrx.org

License

The contents of this file are subject to a BSD license (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.opencrx.org/license.htm>

Copyright 2008 © CRIXP Corp. All rights reserved.



Table of Contents

1	About this Book.....	4
1.1	Who this book is for.....	4
1.2	What do you need to understand this book.....	4
1.3	Tips, Warnings, etc.....	4
2	Prerequisites.....	5
2.1	Issue(s) specific to PostgreSQL.....	5
2.1.1	Puzzling behavior of PostgreSQL.....	5
2.1.2	Implications for openCRX.....	6
2.1.3	Enhancing performance.....	6
3	Upgrading from previous versions.....	7
3.1	The SQL Script upgrade-from-.....	7
3.2	The SQL Script migrate-from-.....	7
3.3	The SQL Script drop-from-.....	7
3.4	The SQL Script dbcreate-views.sql.....	8
3.5	The SQL Script dbcreate-indexes.sql.....	8
3.6	Populate Preferences.....	8
4	Create the database.....	9
4.1	Create the database with psql terminal.....	9
4.2	Create the database with pgAdmin III.....	11
5	Install the openCRX Database Schema Objects.....	15
5.1	Install database schema objects with psql terminal.....	15
5.2	Install database schema objects with pgAdmin III.....	16
6	Next Steps.....	18

List of Figures

Figure 1: Start pgAdmin III and connect to the database.....	11
Figure 2: Create a new Login Role system – Step 1.....	11
Figure 3: Create a new Login Role system – Step 2.....	12
Figure 4: Create a new Database – Step 1.....	13
Figure 5: Create a new Database – Step 2.....	13
Figure 6: Create a new Database – Step 3.....	14
Figure 7: Start pgAdmin III Query.....	16
Figure 8: Load and execute dbcreate-tables.sql in pgAdmin III Query.....	16

List of Listings

Listing 1: Configure access to DB.....	9
Listing 2: Remove ident sameuser.....	9
Listing 3: Stop PostgreSQL server.....	9
Listing 4: Start PostgreSQL server.....	9
Listing 5: Create user system with psql terminal.....	10
Listing 6: Create the database CRX_2_1 with psql terminal.....	10
Listing 7: Install database schema objects with psql terminal.....	15

1 About this Book

This book describes how to setup an openCRX database instance for PostgreSQL.

1.1 Who this book is for

The intended audience are openCRX database administrators.

1.2 What do you need to understand this book

This book describes the installation of openCRX for PostgreSQL. The book assumes that you are familiar with PostgreSQL installation and configuration.

1.3 Tips, Warnings, etc.

We make use the following pictograms:



Information provided as a “Tip” might be helpful for various reasons: time savings, risk reduction, etc.



You should carefully read information marked with “Important”. Ignoring such information is typically not a good idea.



Warnings should not be ignored (risk of data loss, etc.)

2 Prerequisites

As a first step you must download the following software packages:

- Download and install **openCRX SDK Installer**. It is available from <http://www.opencrx.org/sdk.htm>
The SDK contains the DB scripts required to install an *openCRX* database in the directory `<SDK_Install_Dir>\opencrx-x.x.x\core\src\sql`
- Download **PostgreSQL Database Server** from <http://www.postgresql.org/download/>
- Download **pgAdmin III** from <http://www.postgresql.org/download/>
- Download the **PostgreSQL JDBC driver** from <http://jdbc.postgresql.org/download.html>
The JDBC driver is required for the application server installation.



Please ensure that you install the **correct JDBC driver** (i.e. matching JDK, PostgreSQL version, etc.) and **one JDBC driver** only! Ignoring this wisdom leads to problems as the connection to the database will fail.

As a next step you must install **PostgreSQL** and **pgAdmin III** (please refer to the PostgreSQL documentation for installation details).

2.1 Issue(s) specific to PostgreSQL

Like any other DBMS, PostgreSQL has some issues and we want you to be aware of them.

2.1.1 Puzzling behavior of PostgreSQL

Based on our analysis, it seems PostgreSQL is not behaving consistently across platforms. The replies to the following (trivial) select statements are absolutely irritating:

Select Statement	expected reply	pg 8.1 on Windows	pg 8.1 on Kubuntu 7.0.4	pg 8.2 on Kubuntu 7.0.4
<code>select '0' > '/'</code>	true	true	true	true
<code>select '0' > '/a'</code>	true	true	false	false
<code>select '0' > '/aa'</code>	true	true	false	false

The replies of pg 8.1 on Windows are correct, **some of the replies of pg on Linux are (in our opinion) not correct**. This puzzling behavior is the reason why you might need some **special settings to improve performance of openCRX v2.1.0 with PostgreSQL**.

A note to the PostgreSQL community:

We are aware of locale-specific sorting, but the results of the above 3 select statements **should be true for any locale** as

`'0' > '/'` implies `'0' > '/(.)+'`

(unless '/' is treated as some kind of special escape character in a particular locale so that `'/a' < '0'` is true for such a locale; to our best knowledge, such a locale does not exist).

2.1.2 Implications for openCRX

As “object ID matching” (OID matching) is a frequent operation it is absolutely crucial that it can be done in an efficient way, otherwise openCRX will suffer from a heavy performance hit. The openCRX database plugin does OID matching with SQL statements containing comparisons like

```
(object_id > id_pattern_0) and (object_id < id_pattern_1)
```

However, as the above does not always work reliably with PostgreSQL the default configuration of the openCRX database plugin resorts to a comparison based on LIKE for PostgreSQL. We are aware of the implications – a severe performance hit – as prepared statements with LIKE comparisons typically don't use indexes. The next chapter contains information on how to improve the performance.

2.1.3 Enhancing performance

If any of the following conditions is satisfied you can override the default setting of the openCRX database plugin:

- PostgreSQL runs on Windows
- the locale of the openCRX database is equal to C
- the locale of the openCRX database is equal to POSIX

With PostgreSQL, the system property

```
org.openmdx.persistence.jdbc.useLikeForOidMatching
```

is by default set to true; this ensures that OID matching works as expected, but the price is a severe performance hit. If any of the above conditions is satisfied you can safely set this system property to false resulting in much improved performance. You can override the default setting by providing the following startup option to openCRX:

```
-Dorg.openmdx.persistence.jdbc.useLikeForOidMatching=false
```

3 Upgrading from previous versions

If you already have PostgreSQL for openCRX installed, upgrade the database as explained in this chapter. You can then skip the rest of the document.



Warning

Backup your database **before** you run any DB scripts!



Warning

Please consult <http://www.opencrx.org/faq.htm#upgrade> and find out whether there exist specific instructions for your openCRX version. Instructions below are generic and might not cover all steps required to successfully upgrade your openCRX version.



Warning

Please note that you cannot skip versions when upgrading.

3.1 The SQL Script upgrade-from-...

In a first step you must upgrade your database. openCRX distributions provide an SQL script of the form

upgrade-from-<version from>-to-<version to>.sql

If you have installed openCRX 2.0.0, for example, and you want to upgrade to version 2.1.0 you have to run the script `upgrade-from-2.0.0-to-2.1.0.sql` on your database instance.

3.2 The SQL Script migrate-from-...

In a second step you must migrate your database. openCRX distributions often times provide an SQL script of the form

migrate-from-<version from>-to-<version to>.sql

If you have installed openCRX 2.0.0, for example, and you want to upgrade to version 2.1.0 you have to run the script `upgrade-from-2.0.0-to-2.1.0.sql` on your database instance.

3.3 The SQL Script drop-from-...

Next you can drop unused tables from your database. openCRX distributions often times provide an SQL script of the form

drop-from-<version from>-to-<version to>.sql

If you have installed openCRX 2.0.0, for example, and you want to drop tables not used by openCRX 2.1.0 you can run the script `drop-from-2.0.0-to-2.1.0.sql` on your database instance. Alternatively, you can also rename such tables,

e.g. from `transition_type` to `unused_transition_type`. Also, it goes without saying that you should never drop a table before you made a backup!

3.4 The SQL Script `dbcreate-views.sql`

Most new openCRX versions make use of new/changed views, i.e. if an openCRX distribution includes an SQL script of the form

`dbcreate-views.sql`

then you should run that script. If you have installed openCRX 2.0.0, for example, and you want to upgrade to openCRX 2.1.0 you should run the script `dbcreate-views.sql` on your database instance. Make sure that old views are indeed dropped and new views properly created.

3.5 The SQL Script `dbcreate-indexes.sql`

Most new openCRX versions make use of new/changed indexes, i.e. if an openCRX distribution includes an SQL script of the form

`dbcreate-indexes.sql`

then you should run that script. If you have installed openCRX 2.0.0, for example, and you want to upgrade to openCRX 2.1.0 you should run the script `dbcreate-indexes.sql` on your database instance.

3.6 Populate Preferences

The last step involves deleting old preferences and populating the table with new ones. Run the SQL script `populate-preferences.sql` to do this.



Make sure that old preferences are indeed removed and news ones loaded. This table contains the configuration of the openMDX database plugin, i.e. openCRX persistency will not work properly if the loaded preferences do not match the version of openCRX.

4 Create the database

You can either create the database with the psql terminal or with pgAdmin III.

4.1 Create the database with psql terminal

In order to connect to a PostgreSQL server from a remote pgAdmin III instance you have to properly configure the PostgreSQL server.

Let the PostgreSQL server accept connections from inside your network (suppose 192.168.1.0/24) by postgres and system users. We suppose the connections will be clear (not ssl) and protected by a password authentication mechanism. To do this, add the following lines at the end of the file `<postgres home dir>/data/pg_hba.conf`:

Listing 1: Configure access to DB

#TYPE	DATABASE	USER	IP-ADDRESS	IP-MASK	METHOD
local	all	all			password
host	CRX_2_1	system	192.168.1.0	255.255.255.0	password
host	all	postgres	192.168.1.0	255.255.255.0	password



Remove the line with ident sameuser if it exists in `pg_hba.conf`:

Listing 2: Remove ident sameuser

```
host all all 127.0.0.1 255.255.255.255 ident sameuser
```

If the PostgreSQL server is running, stop it before you continue:

Listing 3: Stop PostgreSQL server

```
postgres$pg_ctl stop
```

Next, the PostgreSQL server has to be configured to accept more than one connection through a socket. To do this, log-in to the system with the PostgreSQL server user account (usually `postgres`) and start the PostgreSQL server with the following options:

Listing 4: Start PostgreSQL server

```
postgres$pg_ctl -o -i -l postmaster.log start
```

Now you can connect to the default PostgreSQL server database (template1) with the `postgres` user (DB Administrator).

Create a PostgreSQL *user* named `system`. Set the password of the user to `manager` (for the purpose of this guide). Allow the user to create databases.

Listing 5: Create user system with psql terminal

```
su - postgres          # change to your postgresql Account.
createuser -P system   # Create the user system and ask for a password
```

Create the database `CRX_2_1` with the following commands:

Listing 6: Create the database CRX_2_1 with psql terminal

```
su - postgres
createdb -h localhost -E utf8 -U system CRX_2_1
```



Set the Encoding to UTF-8 if you intend to make use of the openCRX UTF-8 support.



Please note that you are free to choose any database name you like, i.e. there is no requirement to name the database `CRX_2_1`. However, you must ensure that the database name you choose here is also used in the respective **database definition file** (e.g. `ra.xml` for Tomcat, `jdbc-opencrx-CRX-postgresql-ds.xml` for JBoss, etc.)

Furthermore, the name of the openCRX database is also used for DB-based authentication. Hence, verify (and adapt if required) the respective configuration files (e.g. `opencrx-core-CRX.xml` and `opencrx-ical-CRX.xml` for Tomcat)

You have completed creating the openCRX database.

4.2 Create the database with pgAdmin III

Start pgAdmin III and connect to the database:

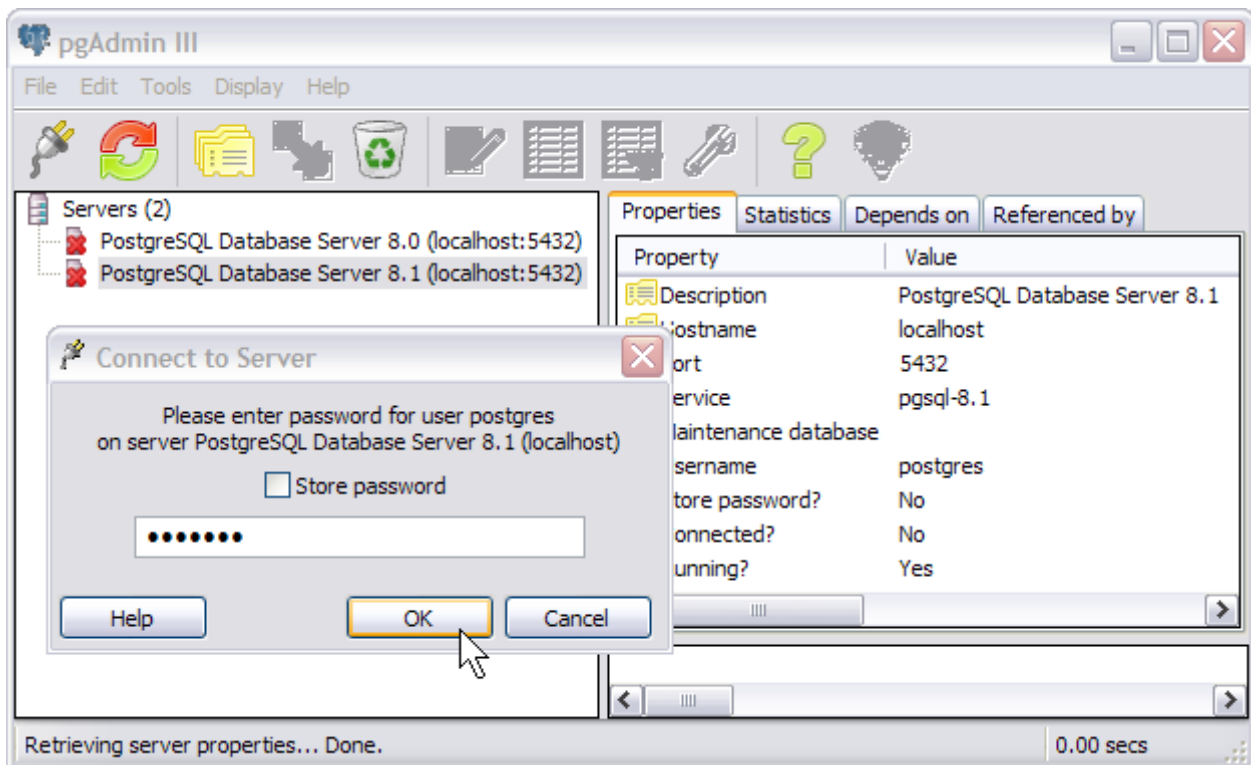


Figure 1: Start pgAdmin III and connect to the database

Next you create a new *login role* **system** as follows:

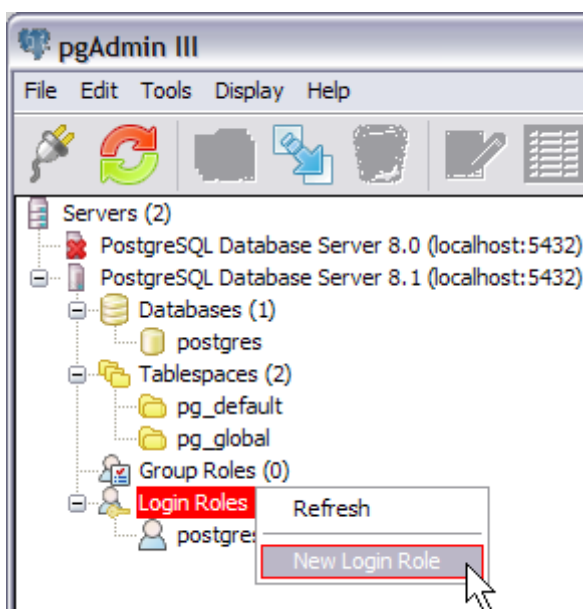


Figure 2: Create a new Login Role system – Step 1

You will get a new window **New Login Role** where you can enter the *Role name* **system** and a *password* (we use *password manager* for the purpose of this guide) – verify that you check options as shown below:

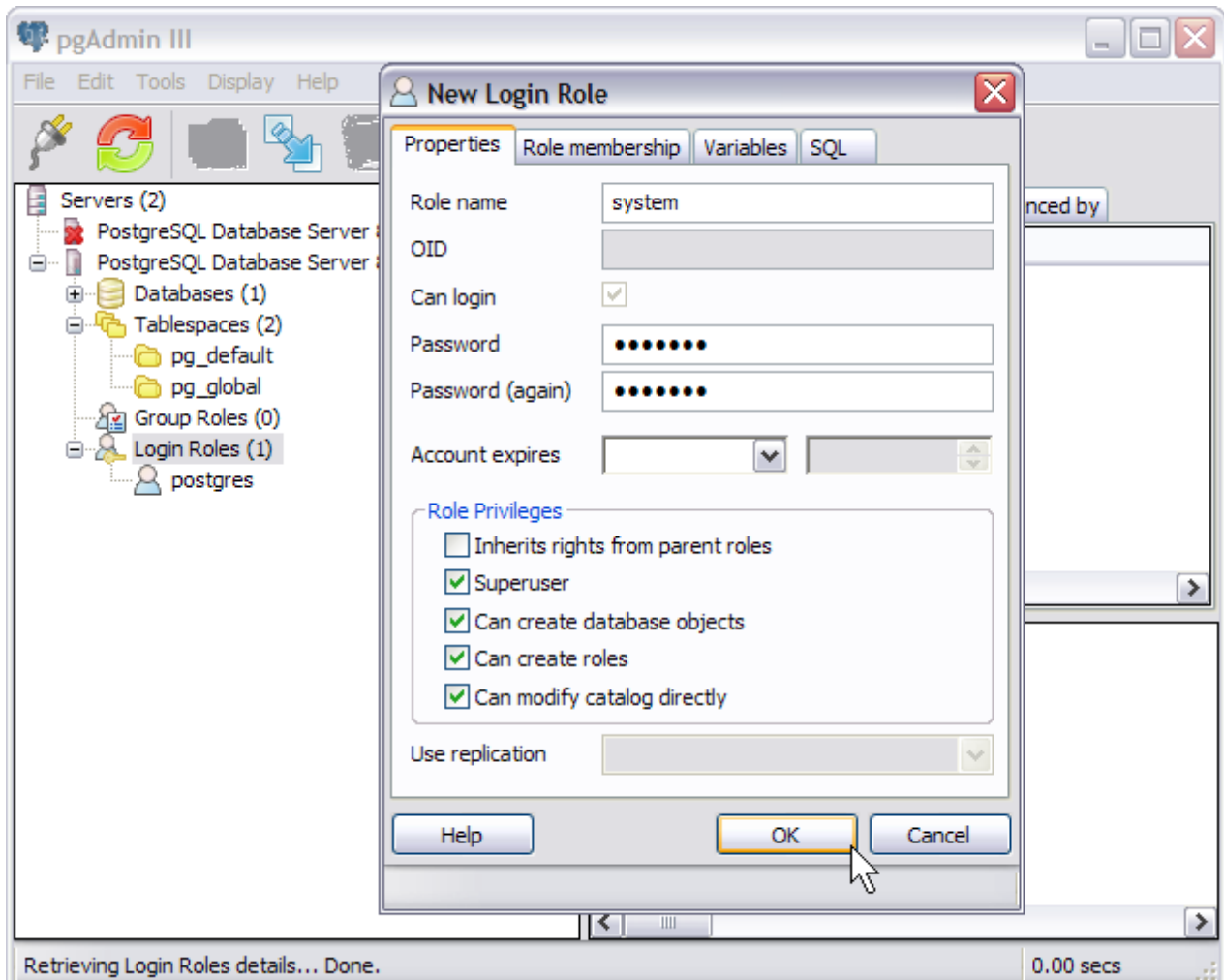


Figure 3: Create a new Login Role system – Step 2



Please note that system must be a **Superuser**

pgAdmin III allows you to create and manage databases. Before you can install the openCRX database schema objects you must create a new database for openCRX. Right-click on the tree item **Databases** and then select the pop-up menu entry **New Database** as shown below:

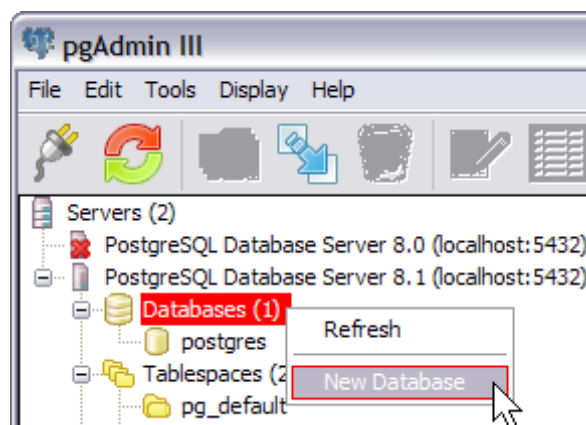


Figure 4: Create a new Database – Step 1

You will get a new window New Database – populate it as shown below to create the database `CRX_2_1` owned by `system`:

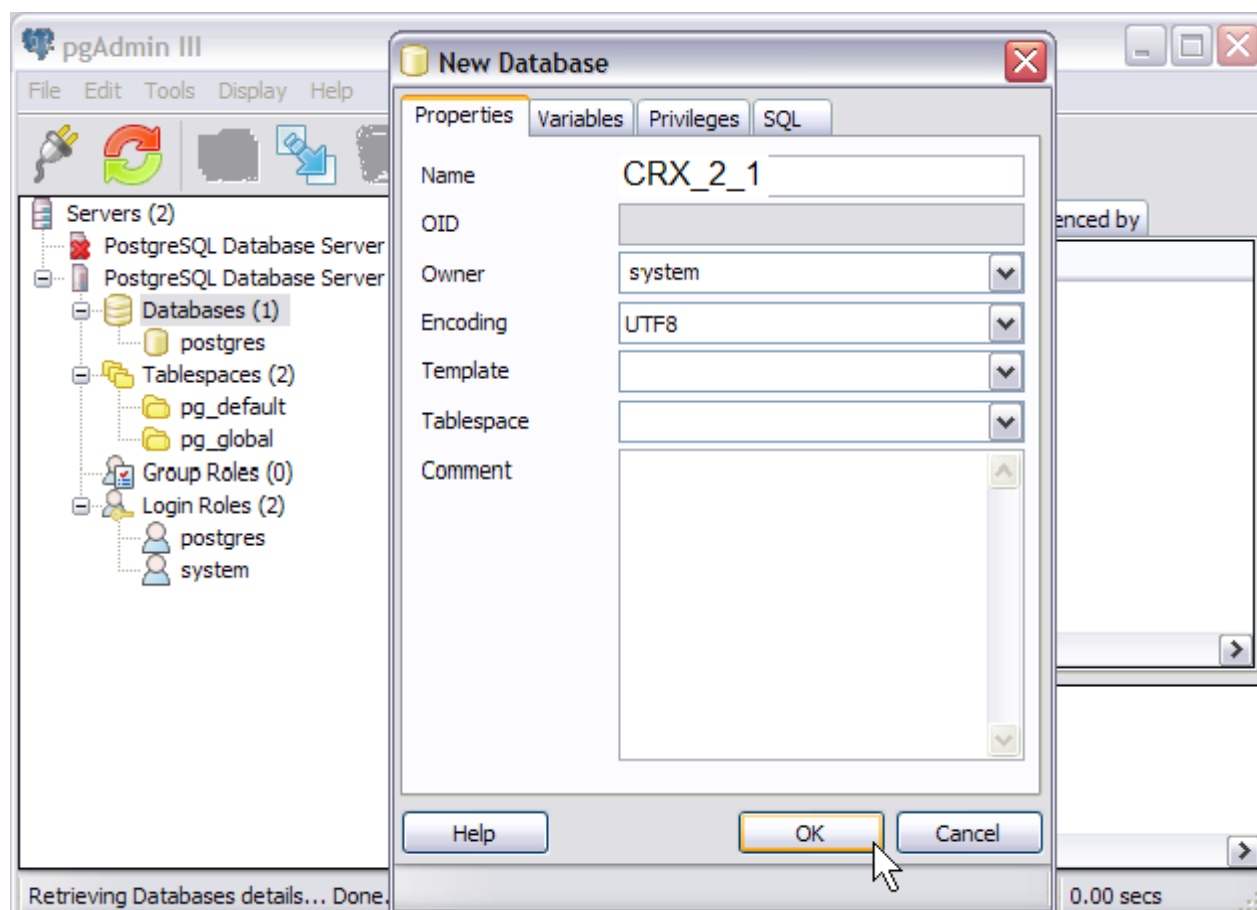


Figure 5: Create a new Database – Step 2



Set the Encoding to UTF-8 if you intend to make use of the openCRX UTF-8 support.

After creating the new Database CRX_2_1 your pgAdmin III window should look similar to the following figure:

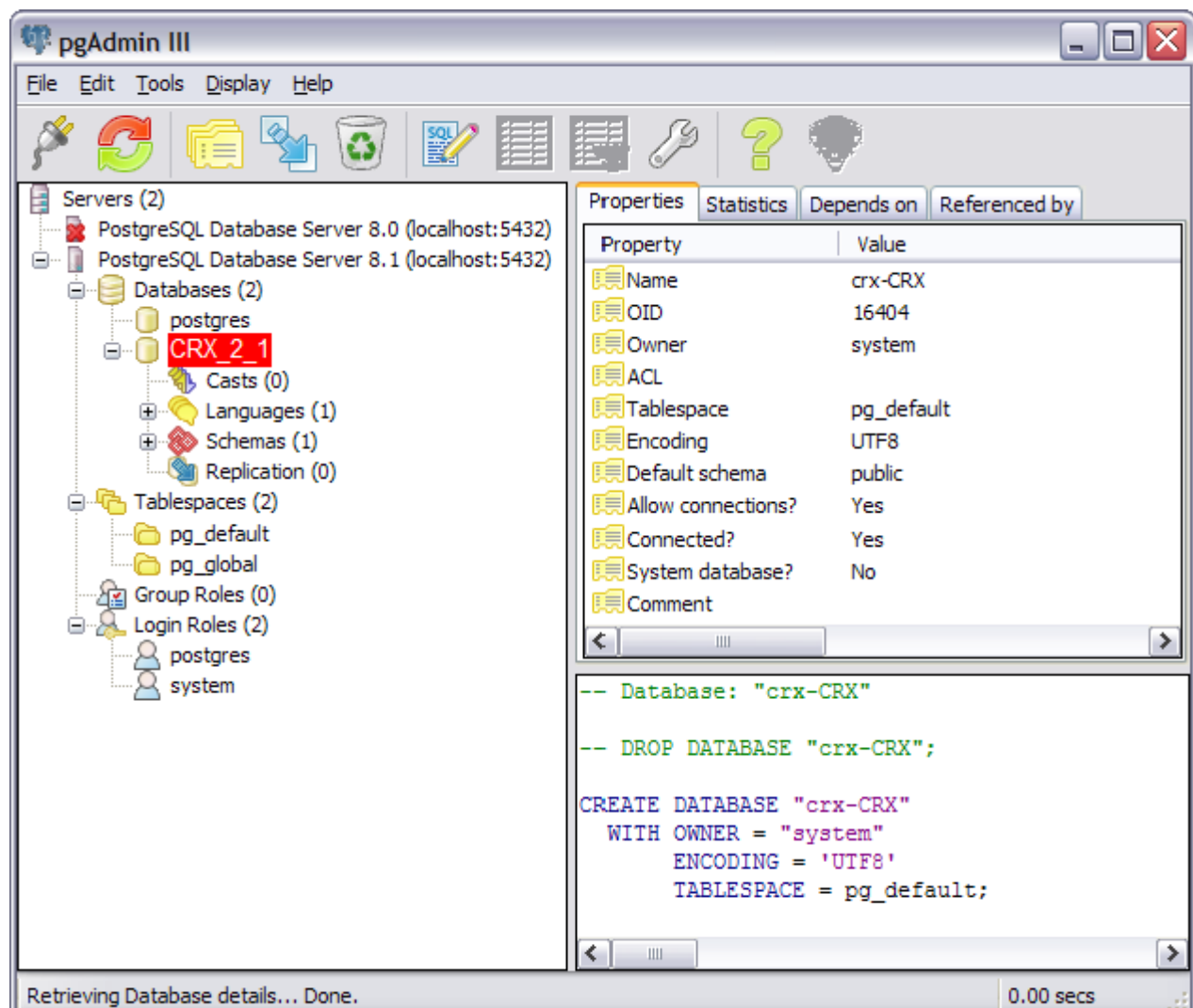


Figure 6: Create a new Database – Step 3

You have completed creating the database **CRX_2_1**.

5 Install the openCRX Database Schema Objects

After creating the schema you are now ready to install the openCRX database schema objects. The following scripts must be executed in the order given below:

- dbcreate-tables.sql
- dbcreate-views.sql
- dbcreate-indexes.sql
- populate-preferences.sql



Do not execute any other scripts included in the distribution.

Again, depending on your preferences you can either use the psql terminal or pgAdmin III to install the openCRX database schema objects.

5.1 Install database schema objects with psql terminal

Execute the scripts in the given order.



If any of the scripts does not run without errors, correct the errors **before** you continue with the next script.

Listing 7: Install database schema objects with psql terminal

```
psql -U system CRX_2_1 < dbcreate-tables.sql
psql -U system CRX_2_1 < dbcreate-views.sql
psql -U system CRX_2_1 < dbcreate-indexes.sql
psql -U system CRX_2_1 < populate-preferences.sql
```



The script **dbcreate-views.sql** tries to drop old views before it creates the new ones. In case there are no existing views you need to comment out the drop statement to successfully create new views.

This completes the installation of the openCRX database schema objects.

5.2 Install database schema objects with pgAdmin III

After creating the database instance and the user `system` you are ready to install the openCRX database schema objects. Connect to the database `CRX_2_1` as user `system` and then start **pgAdmin III Query** as shown below:

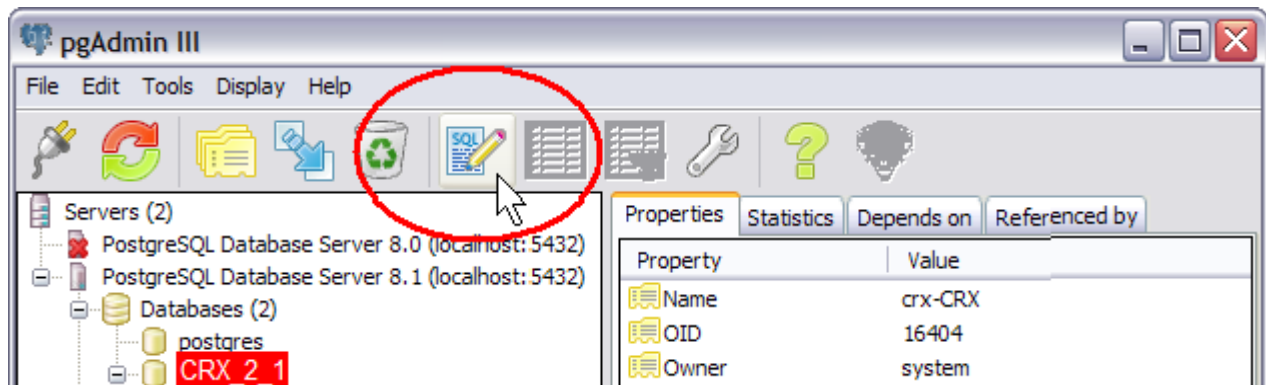



Figure 7: Start pgAdmin III Query

Next you load the script `dbcreate-tables.sql` into the query window and execute it by clicking on the play button :

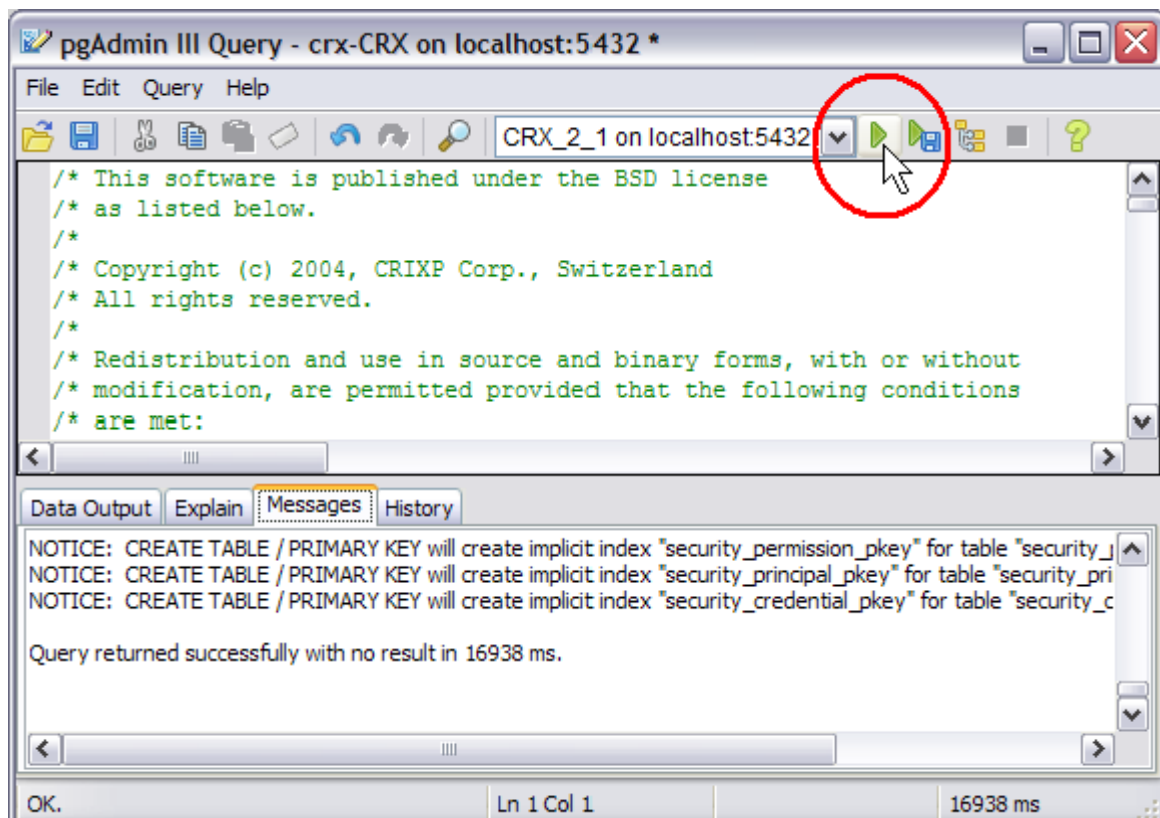


Figure 8: Load and execute `dbcreate-tables.sql` in pgAdmin III Query



If any of the scripts does not run without errors, correct the errors **before** you continue with the next script.

Similarly, execute the remaining scripts in the following order:

- **dbcreate-views.sql**
- **dbcreate-indexes.sql**
- **populate-preferences.sql**



The script **dbcreate-views.sql** tries to drop old views before it creates the new ones. In case there are no existing views you need to comment out the drop statement to successfully create new views.

The scripts should run without errors and after execution you might want to verify with pgAdmin III that everything (i.e. tables, views, indexes, and a populated table `prefs_preference`) was properly created.

This completes the installation of the openCRX database schema objects.

6 Next Steps

If you have completed successfully the database installation you are ready to use the openCRX database **CRX_2_1**. The application server installation guides explain how to connect the application server to the openCRX database instance.