Managing PostgreSQL on Windows

Outline

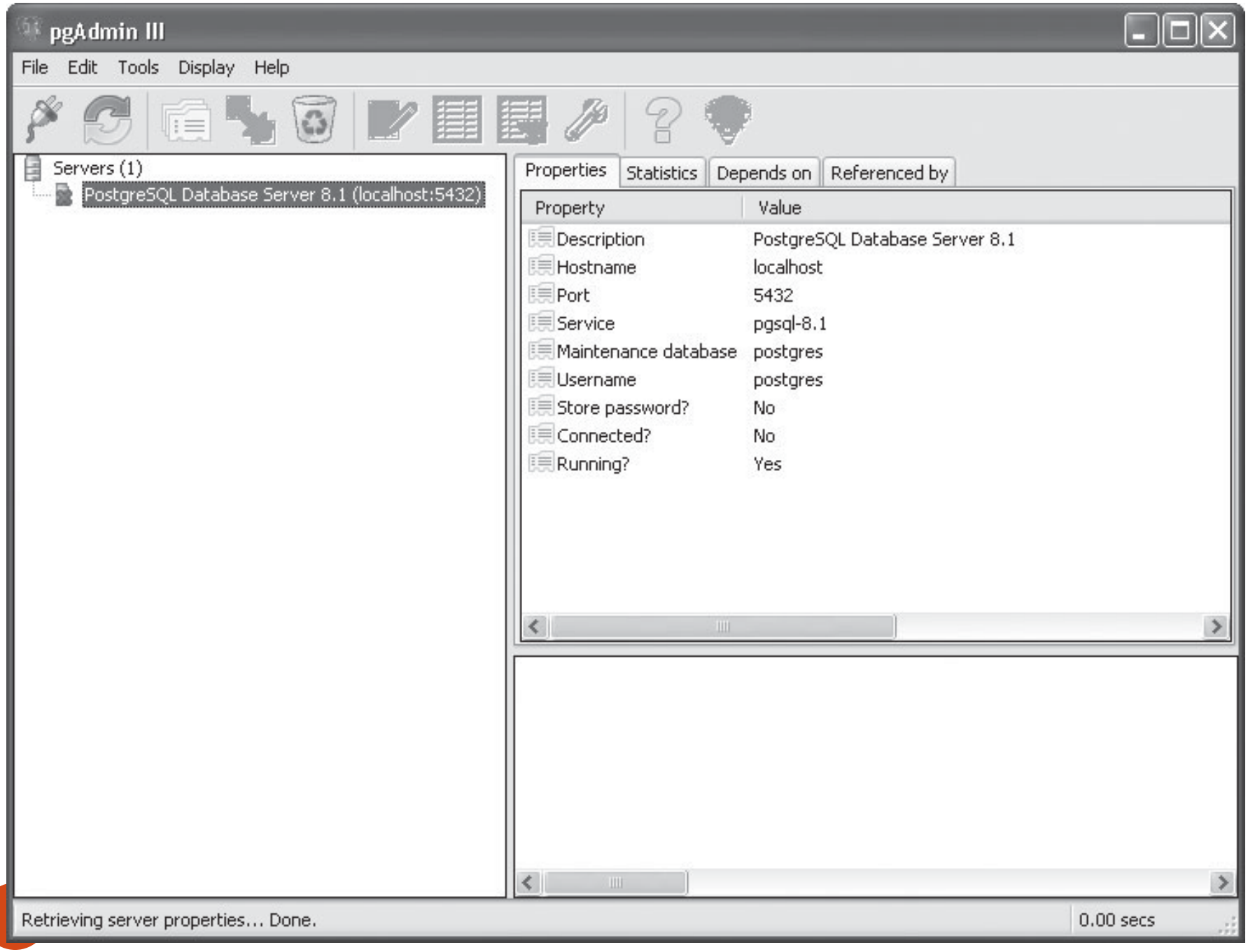
1. The pgAdmin III
2. Parts of the PostgreSQL system
3. Practices – Creat a new application

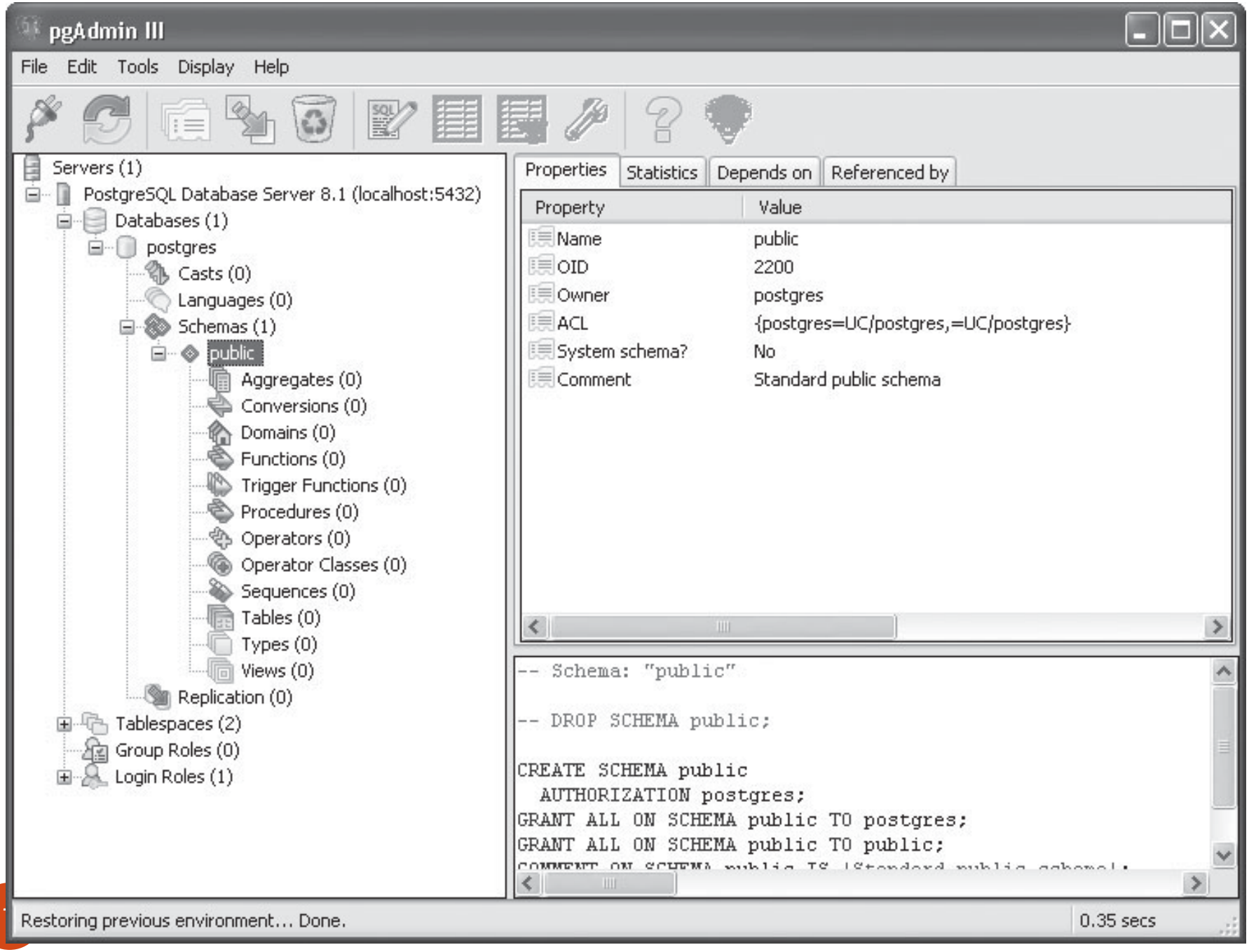
- localhost
- Port: 5432
- Account: postgres
- Password: 12345678

1. The pgAdmin III

The pgAmin III

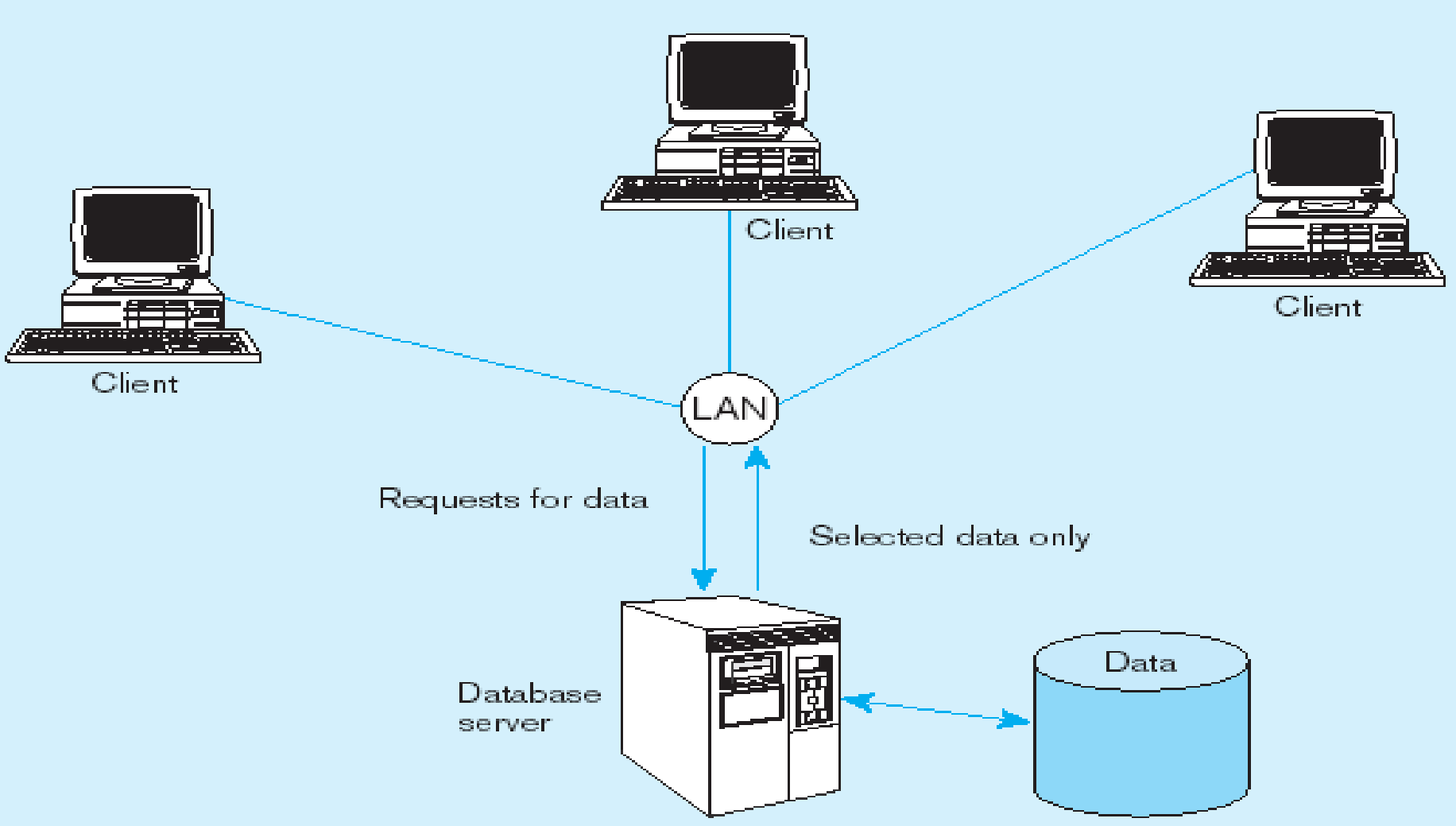
- Any function you need to perform on your PostgreSQL system you can do from within the pgAdmin III graphical interface
- Location: ~bin\pgadmin3.exe
- Default:
  - localhost
  - port: 5432
- Add new connect: File→Add server
- Connect server: right click → Connect


pgAdmin III
File
Edit
Tools
Display
Help
Servers (1)
PostgreSQL Database Server 8.1 (localhost:5432)
Properties
Statistics
Depends on
Referenced by
Property
Value
Description
PostgreSQL Database Server 8.1
Hostname
localhost
Port
5432
Service
pgsql-8.1
Maintenance database
postgres
Username
postgres
Store password?
No
Connected?
No
Running?
Yes
Retrieving server properties... Done.
0.00 secs

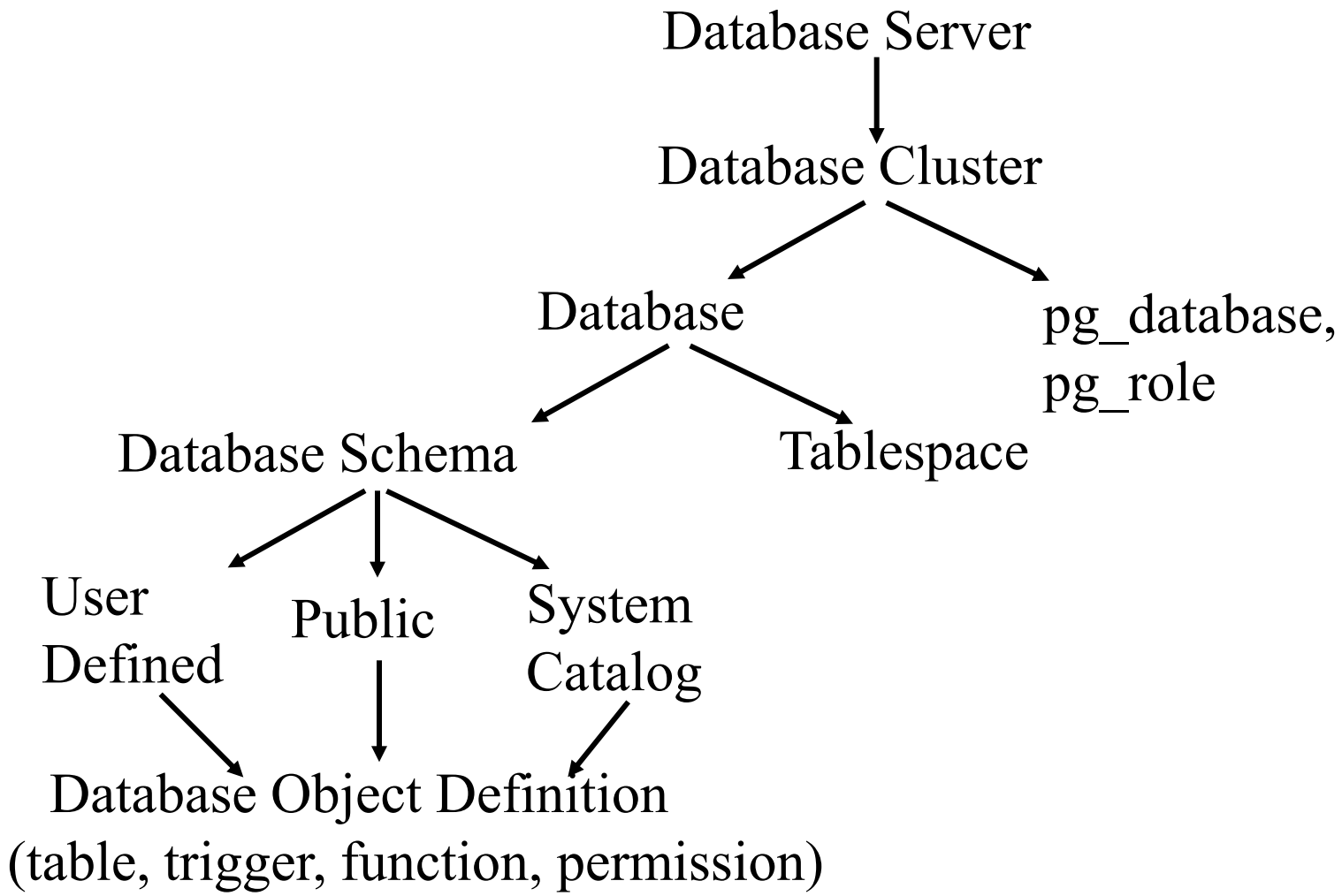

pgAdmin III
File Edit Tools Display Help
Servers (1)
PostgreSQL Database Server 8.1 (localhost:5432)
Databases (1)
postgres
Casts (0)
Languages (0)
Schemas (1)
public
Aggregates (0)
Conversions (0)
Domains (0)
Functions (0)
Trigger Functions (0)
Procedures (0)
Operators (0)
Operator Classes (0)
Sequences (0)
Tables (0)
Types (0)
Views (0)
Replication (0)
Tablespaces (2)
Group Roles (0)
Login Roles (1)
Properties
Statistics
Depends on
Referenced by
Property
Value
Name
public
OID
2200
Owner
postgres
ACL
{postgres=UC/postgres,=UC/postgres}
System schema?
No
Comment
Standard public schema
-- Schema: "public"
-- DROP SCHEMA public;
CREATE SCHEMA public
AUTHORIZATION postgres;
GRANT ALL ON SCHEMA public TO postgres;
GRANT ALL ON SCHEMA public TO public;
Restoring previous environment... Done.
0.35 secs

2. Parts of the PostgreSQL system

Two-tier Database Server Architecture



Database System Structure



Database Server

- Database server
  - A computer program that provides database services to other programs (client applications) that access the server via a network
- A database server divides a client application onto:
  - A front end that runs on a user's computer and typically performs simple tasks as displaying results and
  - A back end that runs on the server computer and performs:
    - User authentication,
    - Transaction control,
    - Query optimization,
    - Database access

Database Cluster

- A database cluster is a directory on disk where all database data will be stored (data area)
- Data in a cluster is stored as a collection of databases
- Cluster databases are managed by a single server instance
- After initialization, a database cluster contains:
  - A database named `postgres`,
  - A database named `template1`
  - A database named `template0`,
  - …

5 basic components

- Tablespaces
- Databases
- Schemas (listed under each individual database)
- Group Roles
- Login Roles

Tablespaces (1/2)

- Table spaces allow a DBA **to define locations in the file system** where the files representing database objects can be stored
- Advantages of using table spaces:
  - If the partition or volume on which the cluster was initialized runs out of space and cannot be extended, a table space can be created on a different partition and used until the system has been reconfigured
  - Table spaces allow a DBA to optimize performance by placing mission critical database objects (like indexes) on highly reliable and fast devices

Tablespaces (2/2)

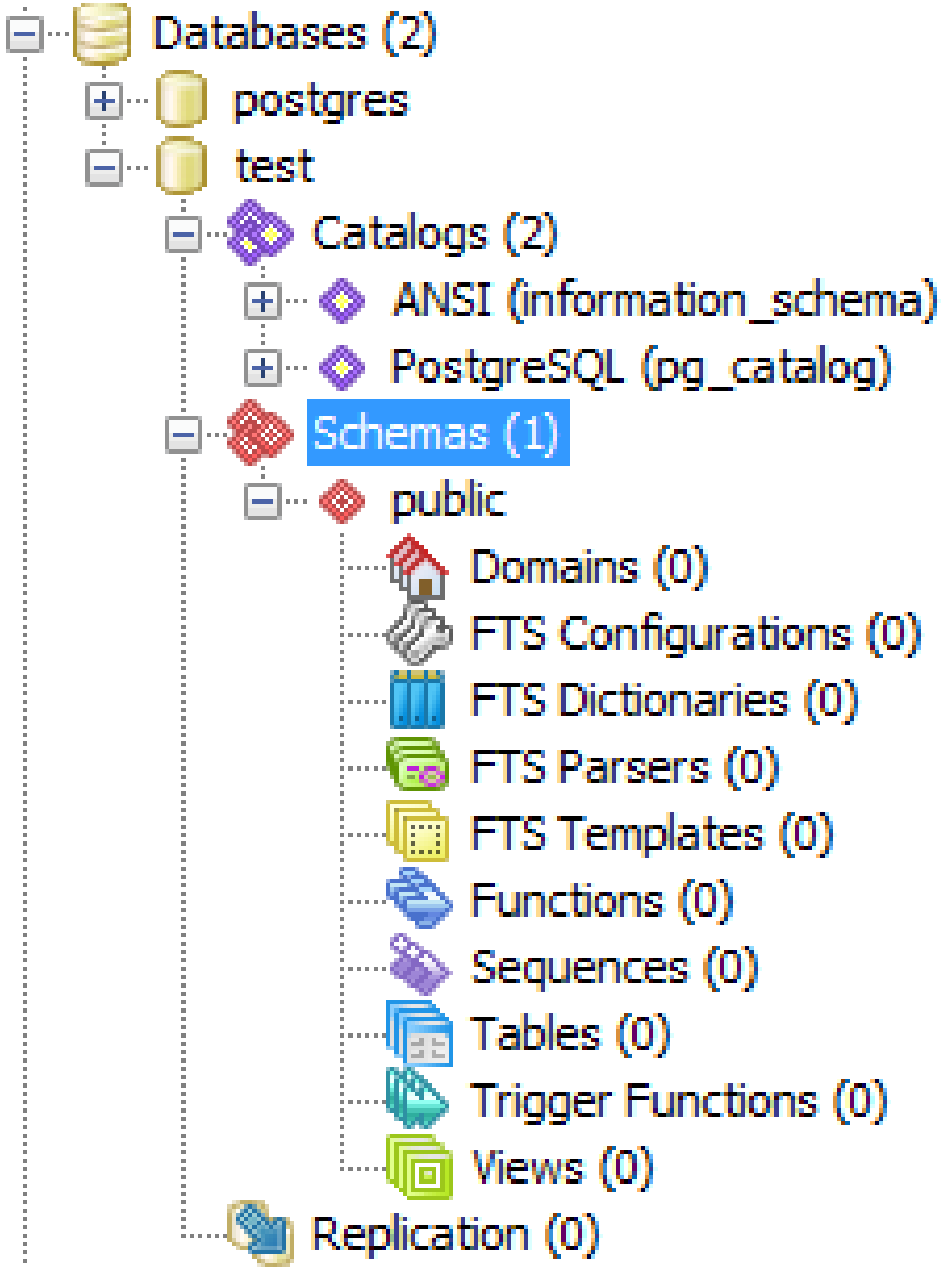
- After initialization, two default tablespaces created:
  - pg_default: the default location for all database objects
  - pg_global: hold PostgreSQL system catalogs, containing internal Data Dictionary information
- When new database objects are created, you must specify which tablespace are they are stored in
- Creating a new tablespace:
  - must point to an empty directory on the system → create directory first
  - *postgres* must have permission to write to the directory
  - default: postgres is a normal account →grant permission to directories

Databases (1/2)

- The core objects in PostgreSQL
- Each client connection to the server can access the data in only one database
- To access data in more than one database a client must make more connections
- The default database created during the PostgreSQL installation is *postgres:*
  - contains the default system tables for handling the internal PostgreSQL Data Dictionary
- *template0* and *template1* (NOT shown in pgAdmin III) are used to create new databases
  - template1 can be modified

Databases (2/2)

- Each database object contains 4 types of objects:
  - Casts: control how Postgres casts from one datatype to another (NOT view in pgAmin III)
  - Languages: these are the languages you can define stored functions, aggregates and triggers in (NOT view in pgAmin III)
  - Schemas: **the most important objects within the database**, containing the tables, triggers, functions, views, and other objects for handling data
  - Replications: define copies (or replicas) of the PostgreSQL database in a fault-tolerant operation


Databases (2)
postgres
test
Catalogs (2)
ANSI (information_schema)
PostgreSQL (pg_catalog)
Schemas (1)
public
Domains (0)
FTS Configurations (0)
FTS Dictionaries (0)
FTS Parsers (0)
FTS Templates (0)
Functions (0)
Sequences (0)
Tables (0)
Trigger Functions (0)
Views (0)
Replication (0)

Template

- CREATE DATABASE actually works by copying an existing database
- Default, it copies the standard system database *template1*
- There is a second standard system database named *template0*
  - the same data as the initial contents of *template1*
  - *never be changed*

Schemas (1/2)

- The **most important objects** within the database
- A database **contains one or more schemas**, which contain **database object** (table, data type, domain, function, trigger) **definitions**
- While users can only access objects **within one database at a time**, they can access **all of the schemas within that database**, *if it has permissions*
- Unlike databases, schemas are not rigidly separated

Schemas (2/2)

Schema Object	Description
Aggregates	Defines functions that produce results based on processing input values from multiple records in a table (such as a sum or average)
Conversions	Defines conversions between character set encodings
Domains	User-defined data types
Functions	User-defined functions
Trigger Functions	User-defined table triggers
Procedures	User-defined functions that manipulate data but do not return a value
Operators	User-defined operators used to compare data
Operator Classes	Defines how a data type can be used within an index
Sequences	Defines a sequenced number generator
Tables	User-created data repositories
Types	User-defined data types used in the database
Views	User-created queries combining data from multiple tables

Catalogs

- A pgAdmin catalog is a schema
- Hold meta data information and built-in Postgres objects
- 2 types:
  - system catalog: pg_catalog
  - information catalog: information_schema

The System Catalog

- Every database system must have a **meta-database of information on the schema** which it contains.
  - The names of the relations in the schemas
  - The names of the columns of each relation.
  - The data type of each column.
  - The integrity constraints on the relations.
  - Information about indices on the relations.
  - The *access privileges* for the elements of the schema.
- This database is often called the *system catalog.*

The System Catalog Schema

- Each PostgreSQL database contains a `pg_catalog` schema
- Normaly, it is copied from the `template1` database
- The `pg_catalog` schema contains tables with information about database objects like:
  - Schemas (`pg_namespace`)
  - Tables, indexes, sequences, and views (`pg_class`),
  - Data types (`pg_type`),
  - Functions and procedures (`pg_proc`),
  - Table columns (`pg_attribute`),
  - Check, unique, primary key, and foreign key constraints (`pg_constraint`),
  - Aggregate functions (`pg_aggregate`),
  - Triggers (`pg_trigger`),
  - Planner (optimizer) statistics (`pg_statistics`), and

Information_schema and pg_catalog

- The *pg_catalog* schema is the standard PostgreSQL meta data and core schema.
- The *information_schema* is part of the ANSI standard, but is not quite so standard. Oracle and DB2 evidentally still don't support
- A lot of this ***information overlaps*** with information found in the information_schema and pg_catalog, but the information_schema is much easier to query
- Although not named explicitly in the **search_path**, `pg_catalog` is the first schema to be searched
  - SELECT * FROM pg_tables
  - SELECT * FROM pg_catalog.pg_tables

Group Roles

- Create access **permissions for groups of users**
- While you can grant an individual user account access directly to a database object, the preferred method is to use Group Roles
- pgAdmin III only allows you to grant Group Roles access to database objects
- Default, **public** group role:
  - applies to all users on the PostgreSQL system
  - NOT able to remove any user account from the public Group Role
  - does not appear in the pgAdmin III Group Roles listing

Login Roles (or user accounts)

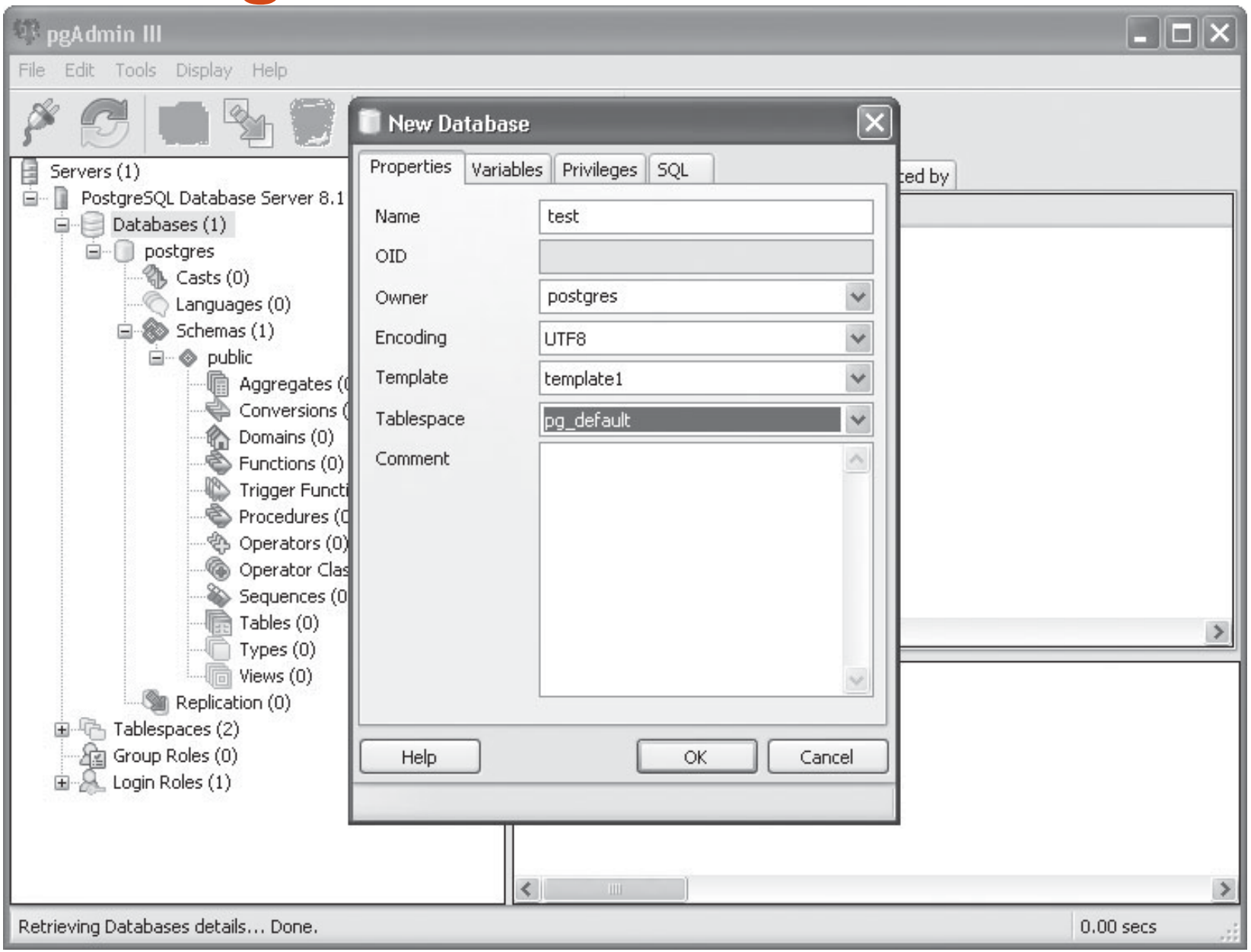
- Are roles that are allowed to log into the PostgreSQL server
- **Each database user** should have **an individual account** for logging into the PostgreSQL system
- That account is then **assigned as a member** of the appropriate **Group Roles that grant privileges** to the database objects required
- Allows you to **easily change access for database objects** without having to touch hundreds (or even thousands) of individual user Login Roles

3. Practices – Create a new application

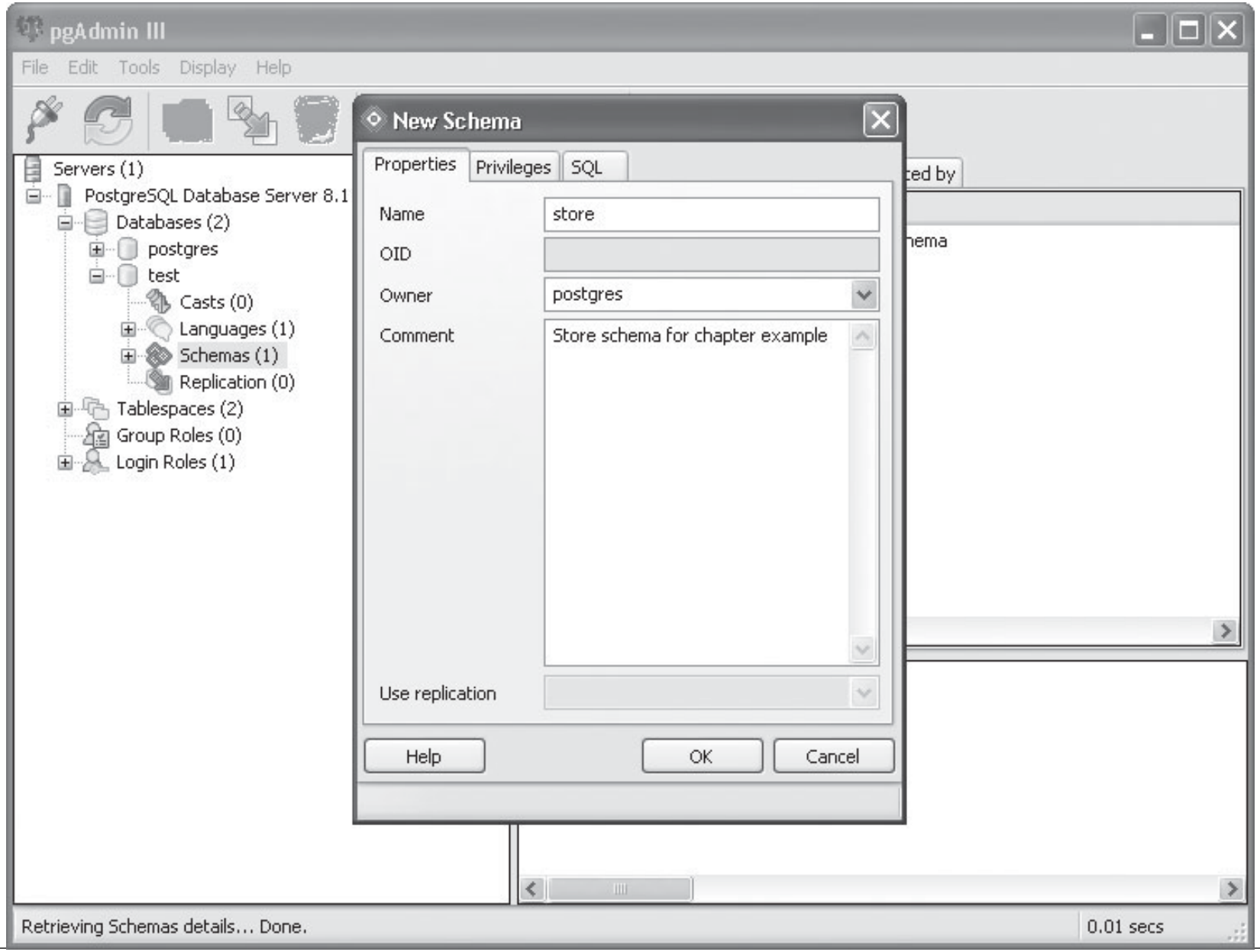
Practices – Create a new application

- Create a database ***test***
  - Customer
  - Product
  - Order
- Create two Group Roles
  - *Salesman* Group Role: write permission on the Customer and Order, only read permission on the Product
  - *Accountant* Group Role: write permission on the Product and Order, read permission on the Customer
- Create two Login Roles
  - salesman - Barney
  - accountant - Fred

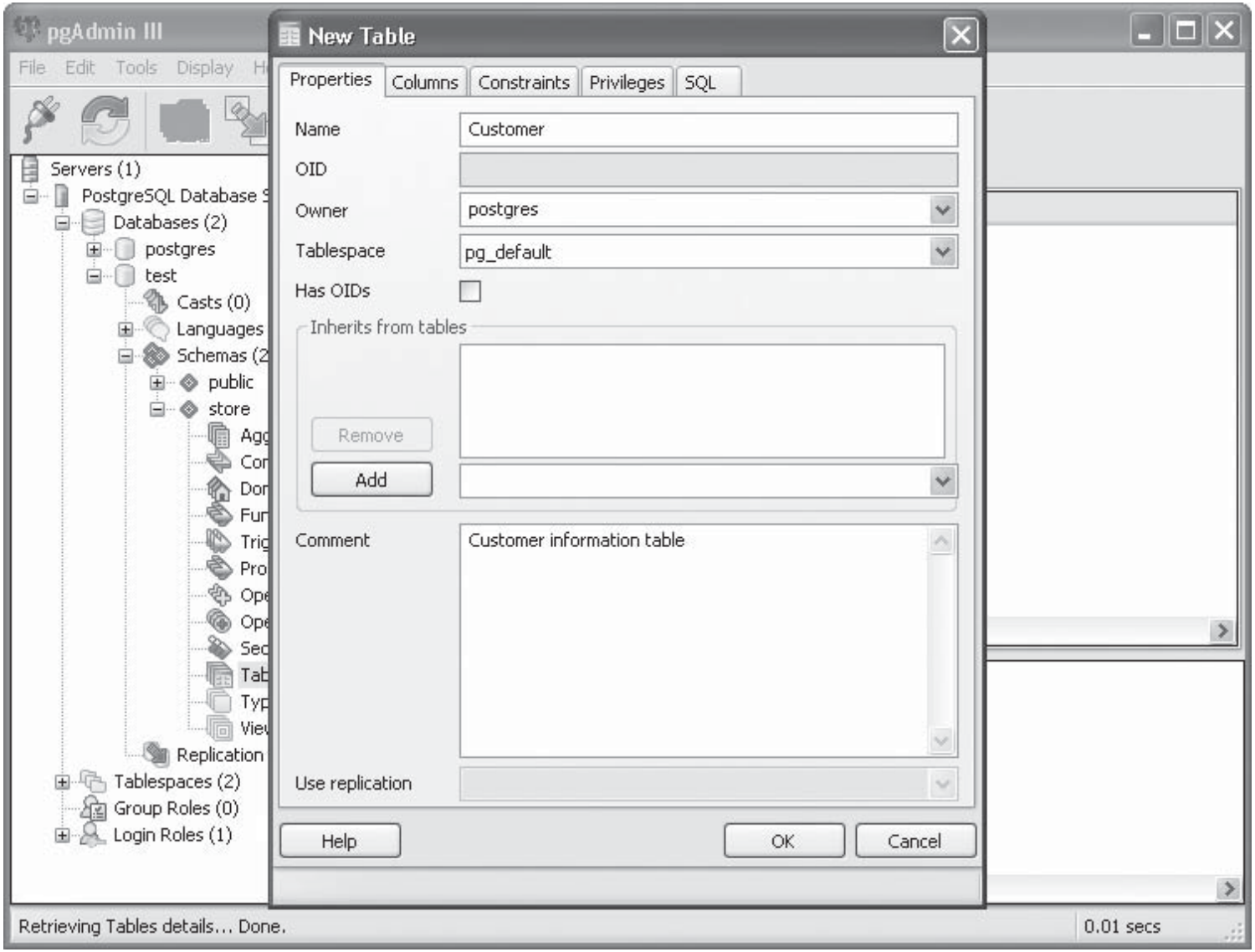
Creating a New Database



Creating a New Schema

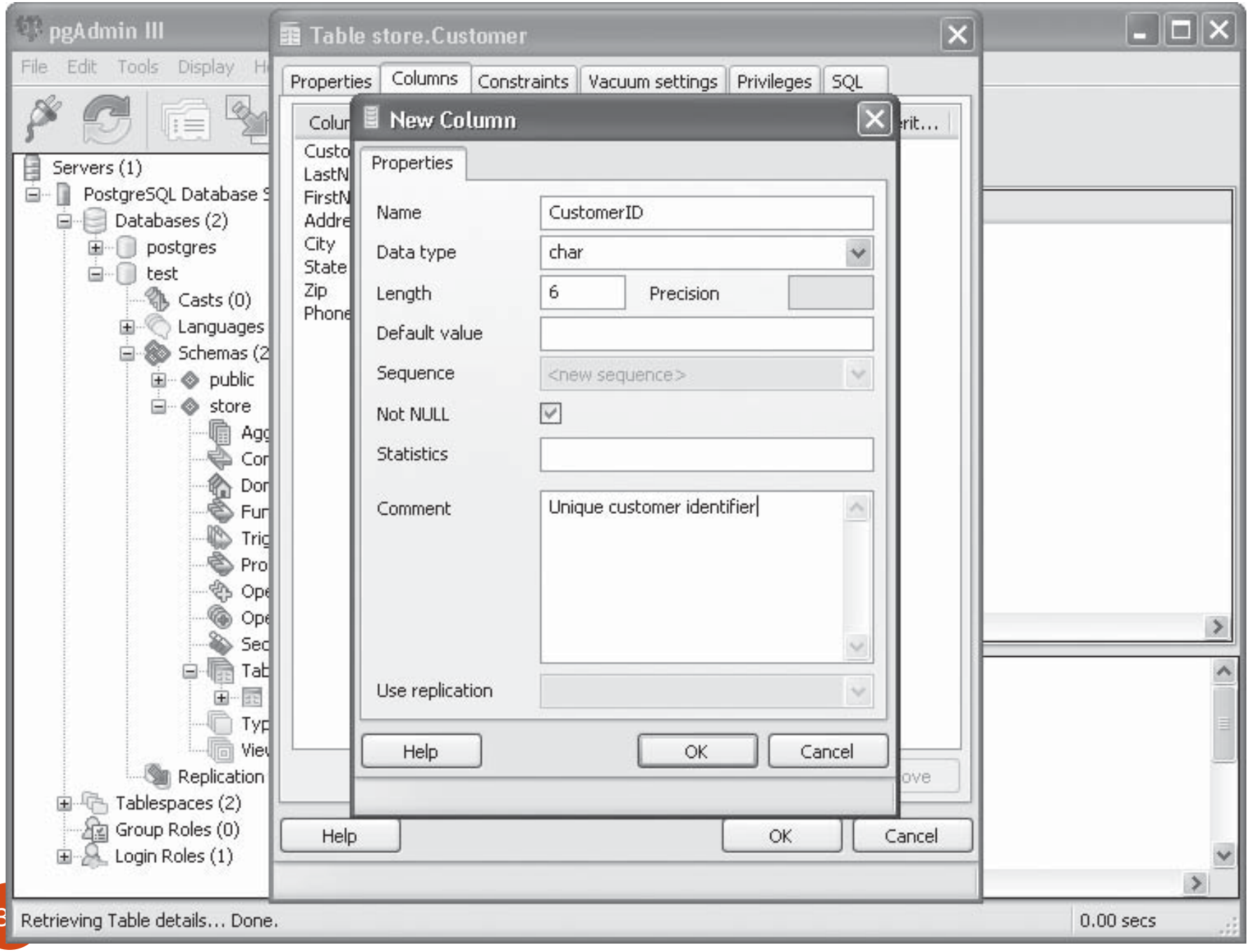


Creating the Tables



Customer Table Columns

Column	Data Type	Description
CustomerID	char—six characters	Unique identifier for each customer
LastName	varchar	Last name of customer
FirstName	varchar	First name of customer
Address	varchar	Street address of customer
City	varchar	City of customer
State	char—two characters	State of customer
Zip	char—five characters	Postal ZIP code of customer
Phone	varchar	Phone number of customer


pgAdmin III
File Edit Tools Display
Servers (1)
PostgreSQL Database
Databases (2)
postgres
test
Casts (0)
Languages
Schemas (2
public
store
Replication
Tablespaces (2)
Group Roles (0)
Login Roles (1)
Table store.Customer
Properties
Columns
Constraints
Vacuum settings
Privileges
SQL
New Column
Properties
Name
CustomerID
Data type
char
Length
6
Precision
Default value
Sequence
<new sequence>
Not NULL
Statistics
Comment
Unique customer identifier
Use replication
Help
OK
Cancel
Help
OK
Cancel
Retrieving Table details... Done.
0.00 secs

Common PostgreSQL Data Types

Name	Aliases	Description
bigint	int8	signed eight-byte integer
bigserial	serial8	autoincrementing eight-byte integer
bit [(*n*)]		fixed-length bit string
bit varying [(*n*)]	varbit	variable-length bit string
boolean	bool	logical Boolean (true/false)
box		rectangular box on a plane
bytea		binary data ("byte array")
character [(*n*)]	char [(*n*)]	fixed-length character string
character varying [(*n*)]	varchar [(*n*)]	variable-length character string
cidr		IPv4 or IPv6 network address
circle		circle on a plane
date		calendar date (year, month, day)
double precision	float8	double precision floating-point number (8 bytes)
inet		IPv4 or IPv6 host address
integer	int, int4	signed four-byte integer
interval [*fields*] [(*p*)]		time span
json		JSON data

file:///C:/Program%20Files/PostgreSQL/9.4/doc/postgresql/html/datatype.html

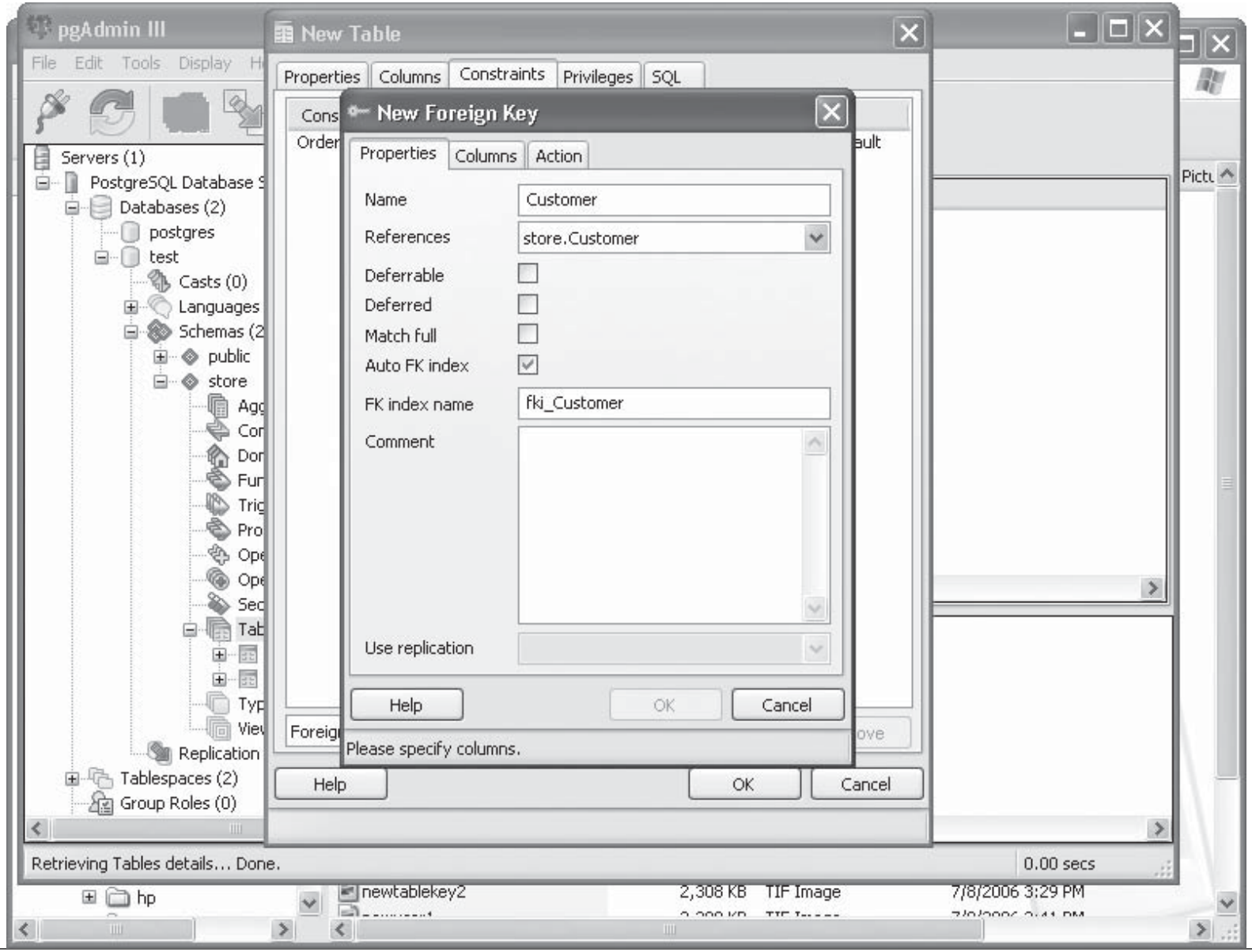
The Product Table Columns

Column Name	Data Type	Description
ProductID	char—six characters	Unique primary key identifier that is not NULL
ProductName	varchar	Name of the product
Model	varchar	Product model number
Manufacturer	varchar	Name of the manufacturer
UnitPrice	money	Current price of product
Inventory	int4	Number of units in inventory

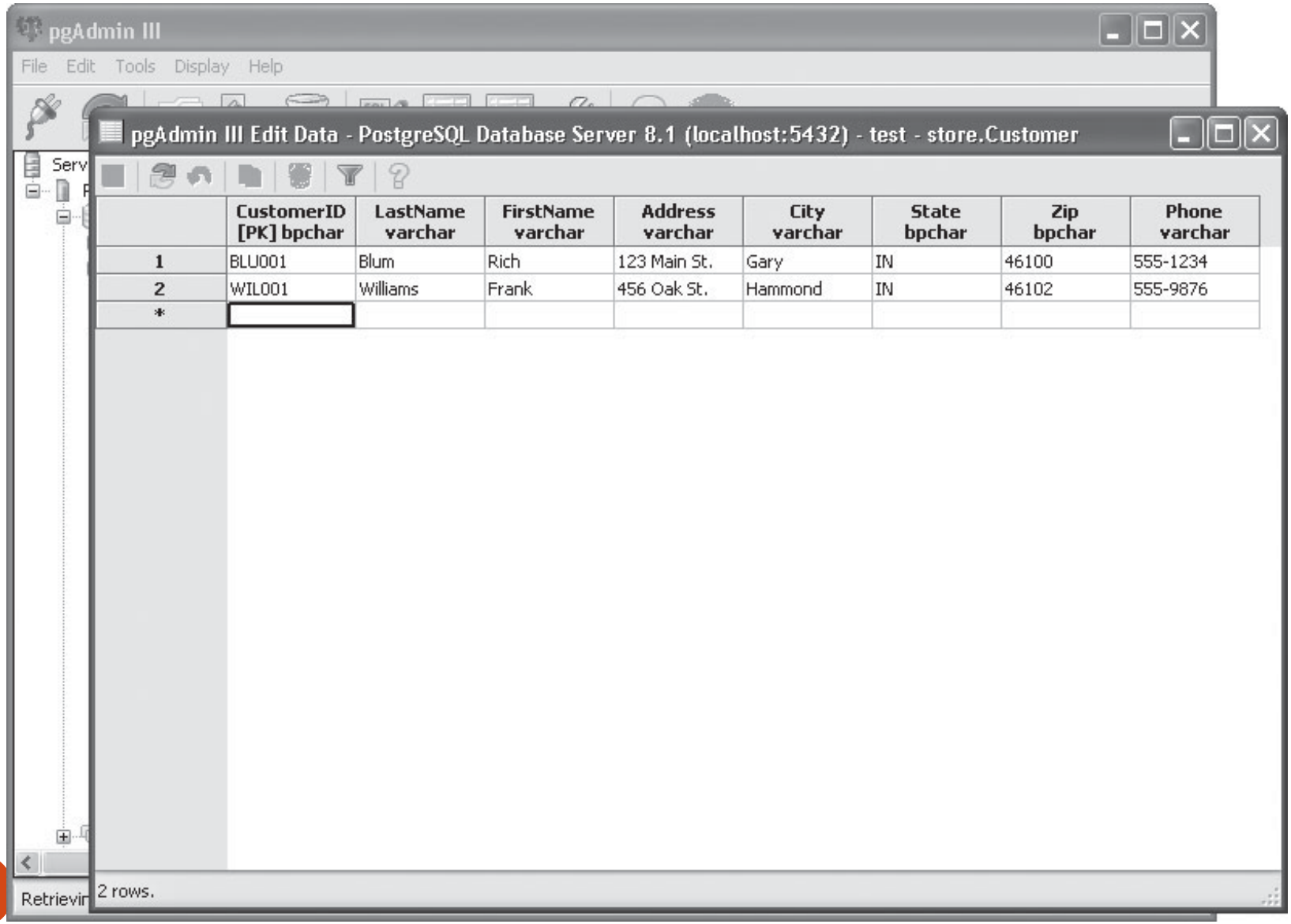
The Columns for the Order Table

Column Name	Data Type	Description
OrderID	char—six characters	Unique primary key identifier that is not NULL
CustomerID	char—six characters	The CustomerID from the Customer table (not NULL)
ProductID	char—six characters	The ProductID from the Product table (not NULL)
PurchaseDate	date	Date of purchase
Quantity	int4	The number of items purchased
TotalCost	money	The total cost of the purchase

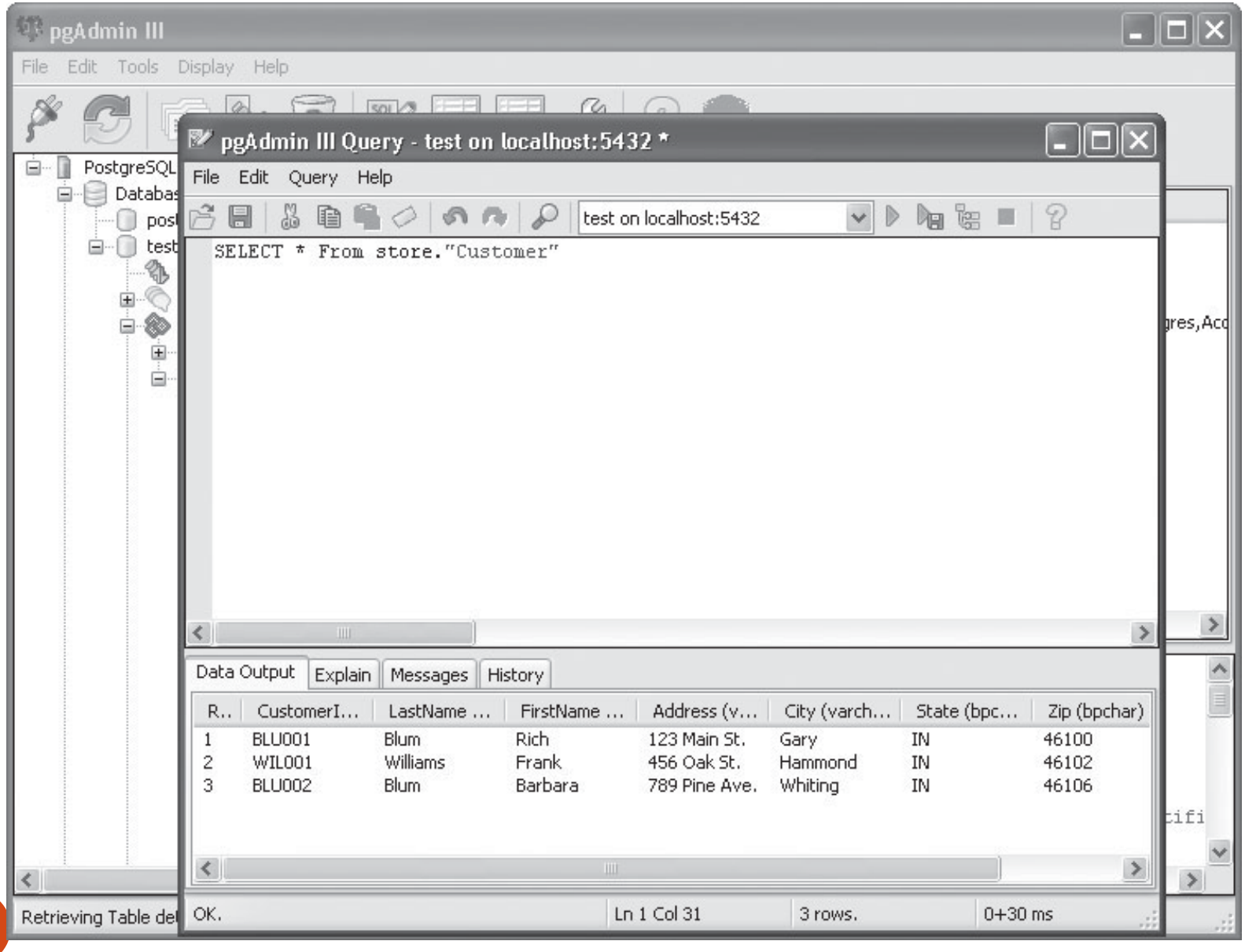
New Foreign Key window for the Order table



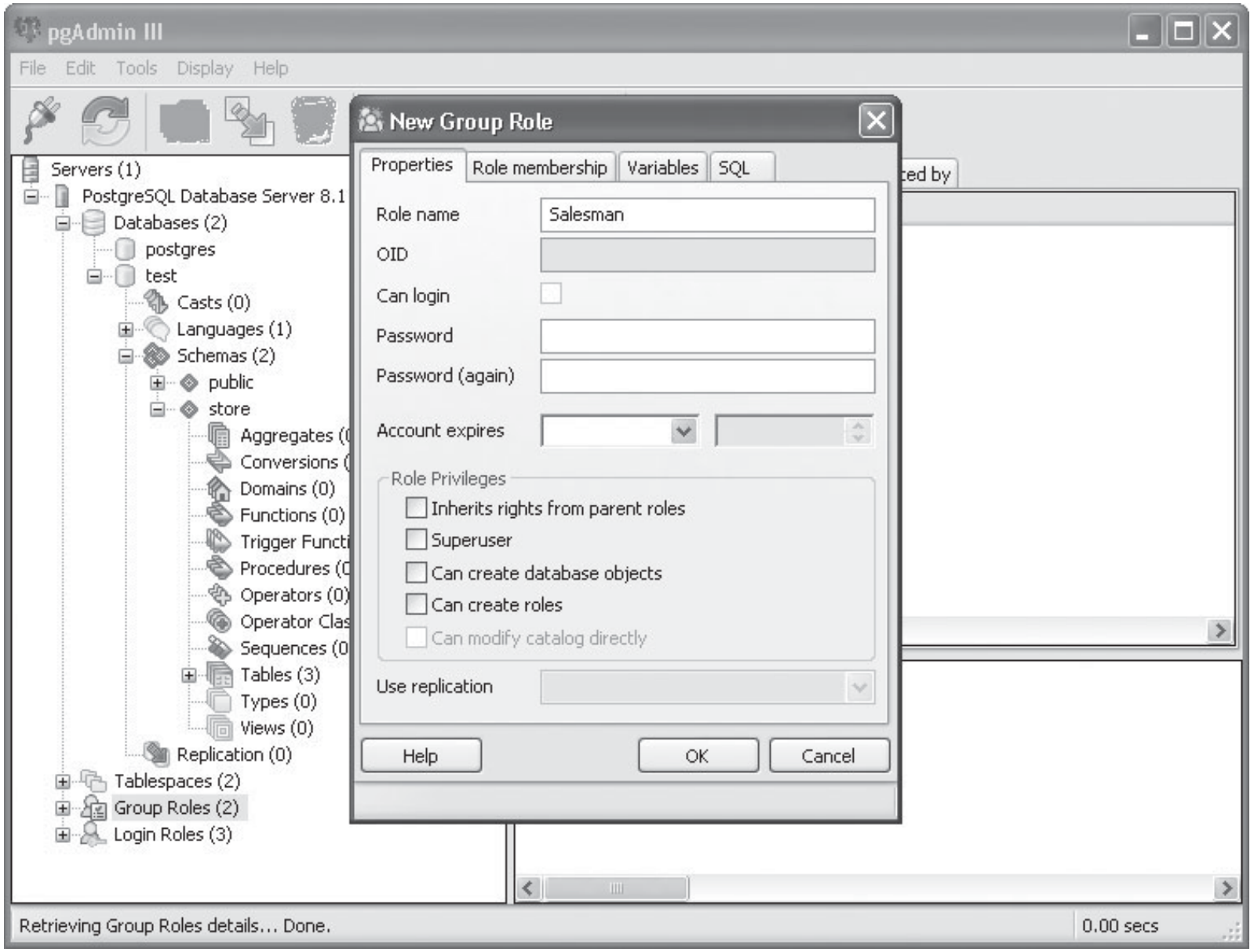
Entering and Viewing Data

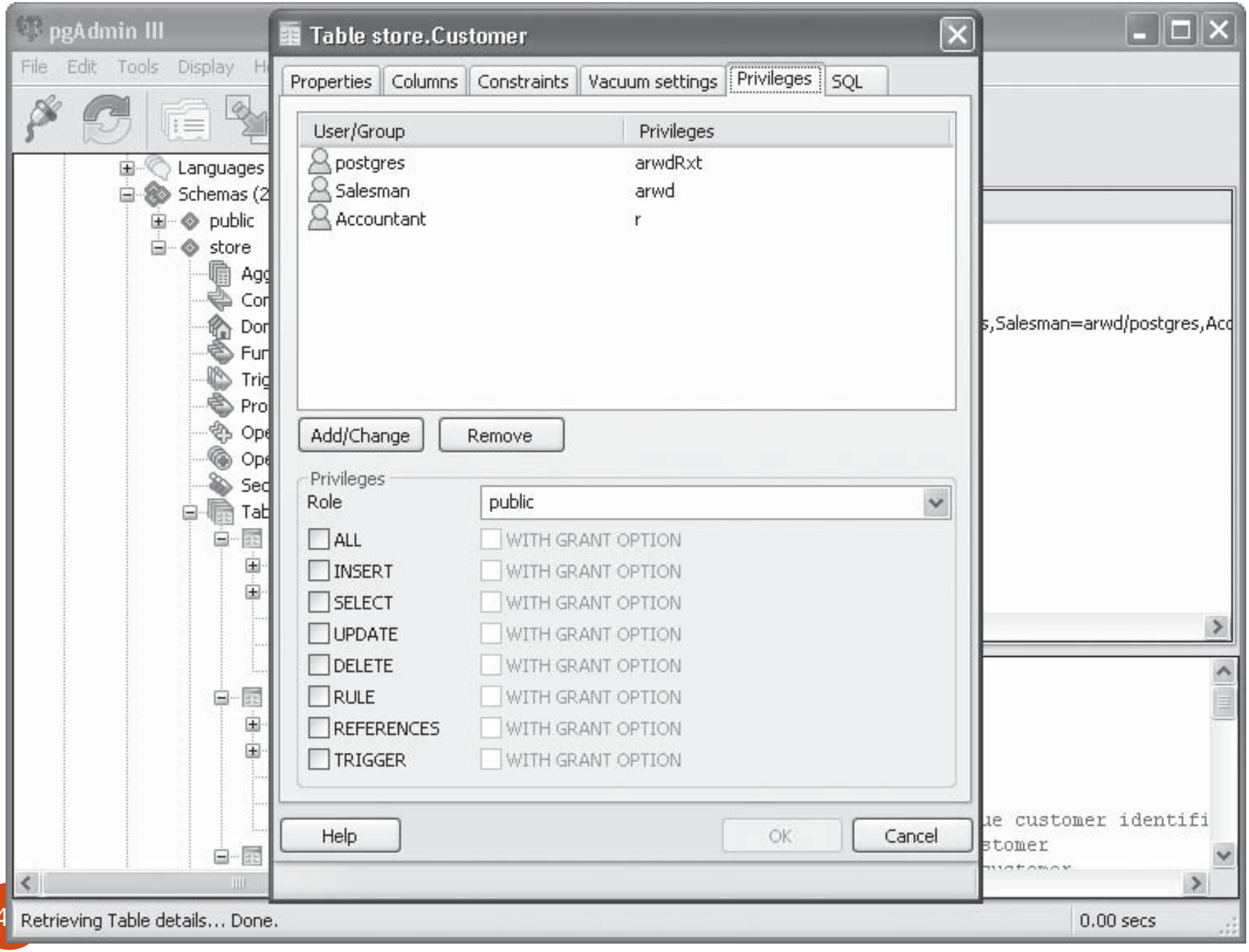


THE pgADMIN III QUERY TOOL



WORKING WITH USER ACCOUNTS

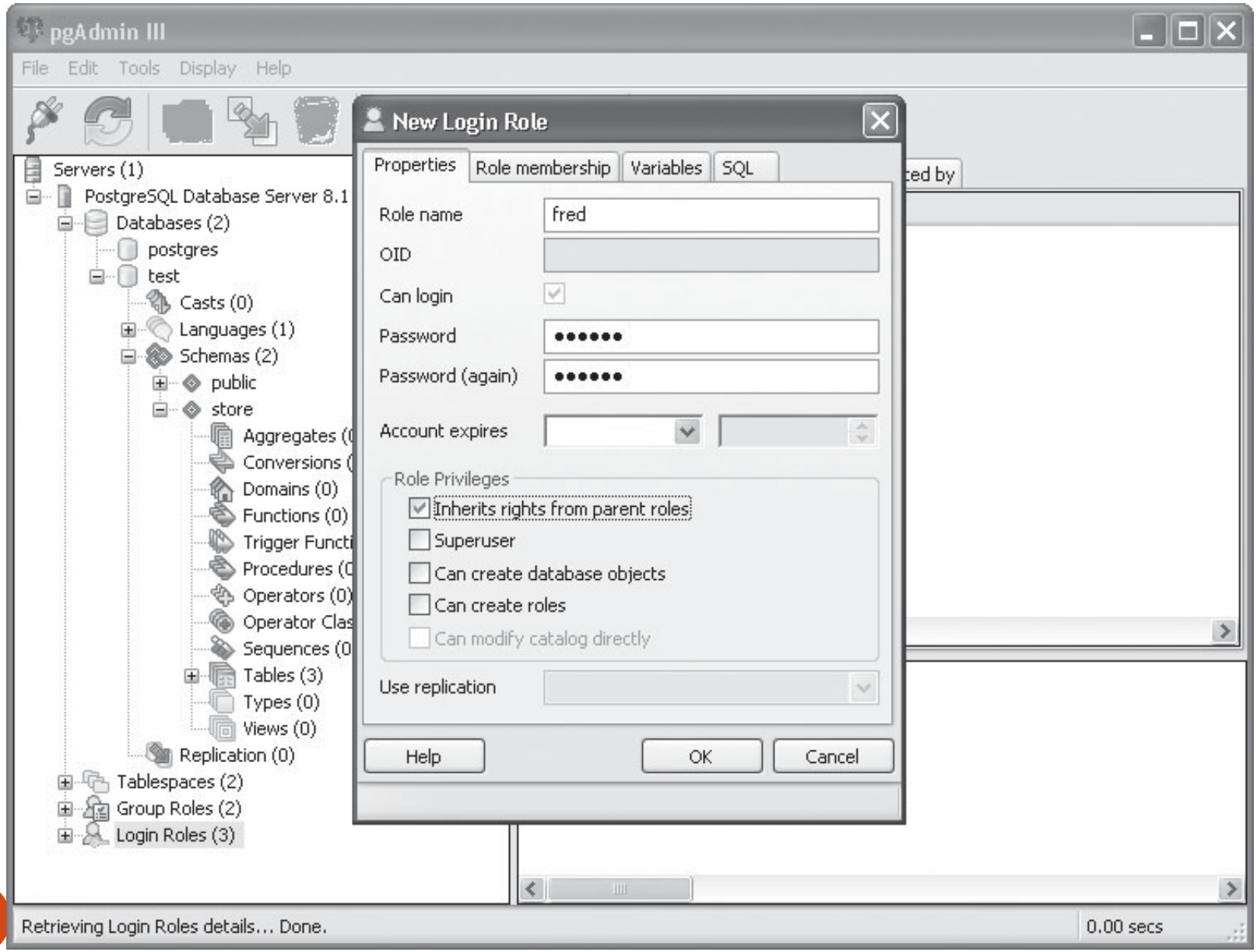



pgAdmin III
File Edit Tools Display
Languages
Schemas (2
public
store
Table store.Customer
Properties Columns Constraints Vacuum settings Privileges SQL
User/Group Privileges
postgres arwdRxt
Salesman arwd
Accountant r
Add/Change Remove
Privileges
Role public
ALL WITH GRANT OPTION
INSERT WITH GRANT OPTION
SELECT WITH GRANT OPTION
UPDATE WITH GRANT OPTION
DELETE WITH GRANT OPTION
RULE WITH GRANT OPTION
REFERENCES WITH GRANT OPTION
TRIGGER WITH GRANT OPTION
Help OK Cancel
s,Salesman=arwd/postgres,Acc
ue customer identifi
stomer
Retrieving Table details... Done.
0.00 secs

pgAdmin Object Privilege Codes

Code	Privilege
a	INSERT (append)
r	SELECT (read)
w	UPDATE (write)
d	DELETE
R	RULE
x	REFERENCES
t	TRIGGER
X	EXECUTE
U	USAGE
C	CREATE
T	TEMPORARY

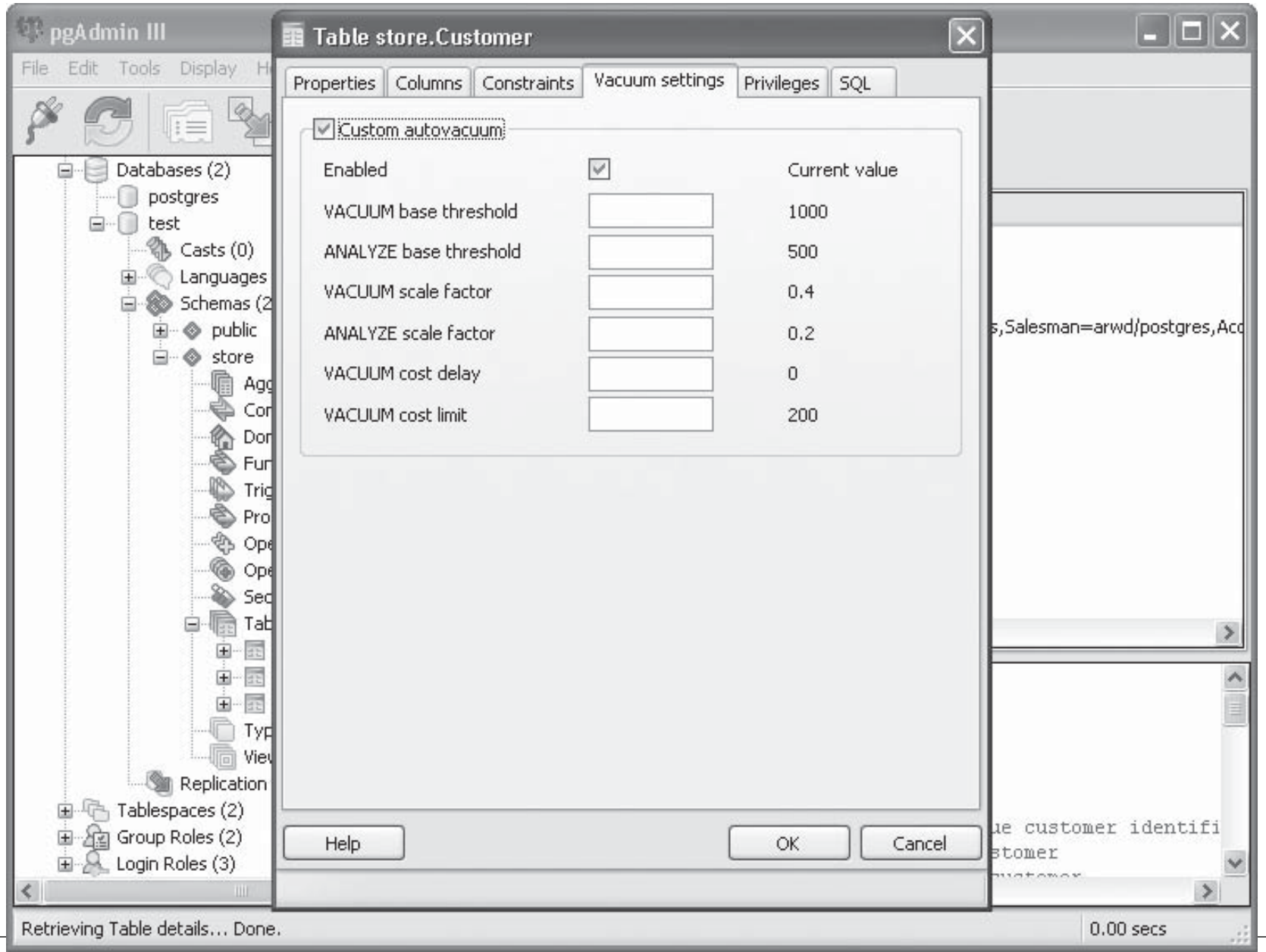
Creating Login Roles

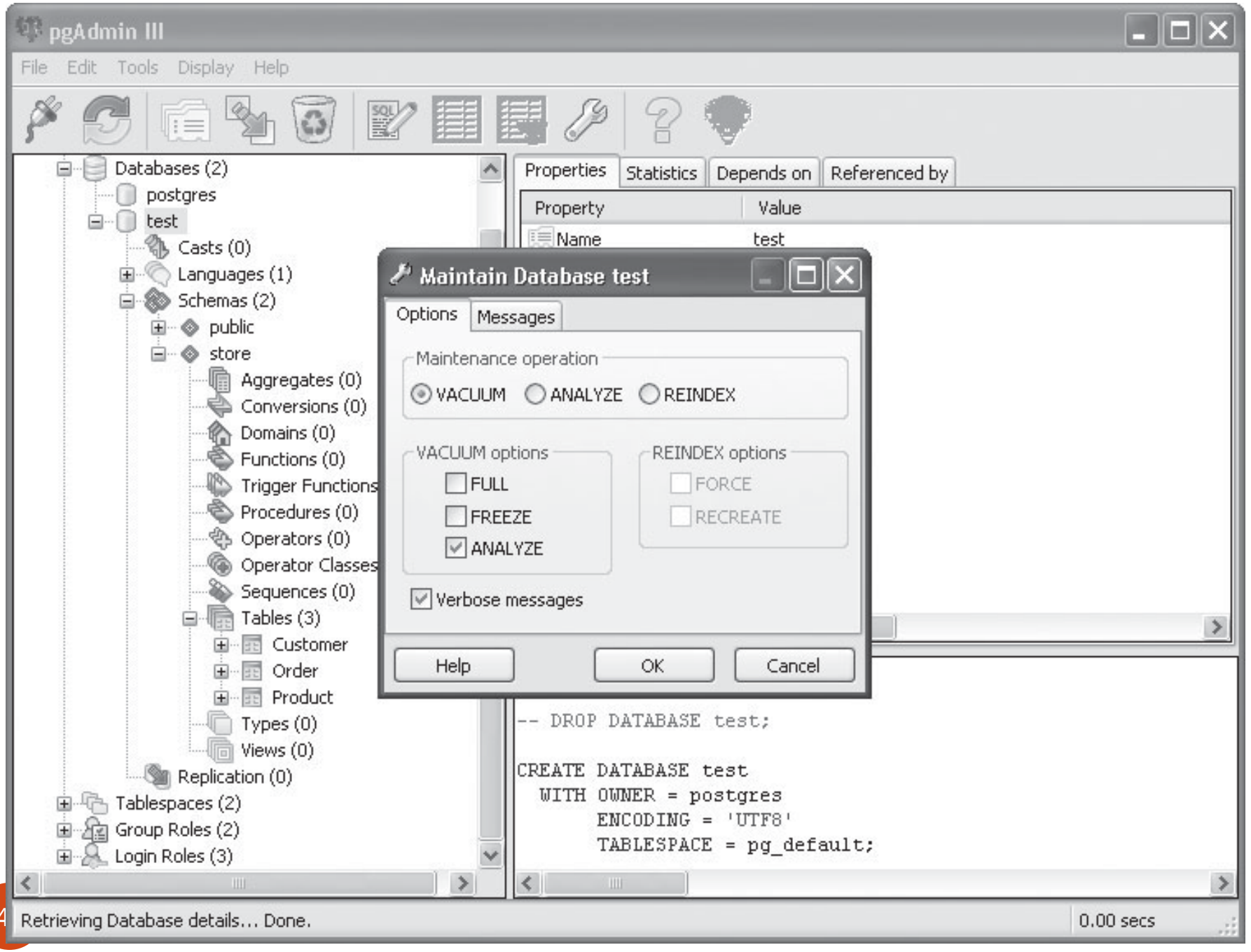


Testing

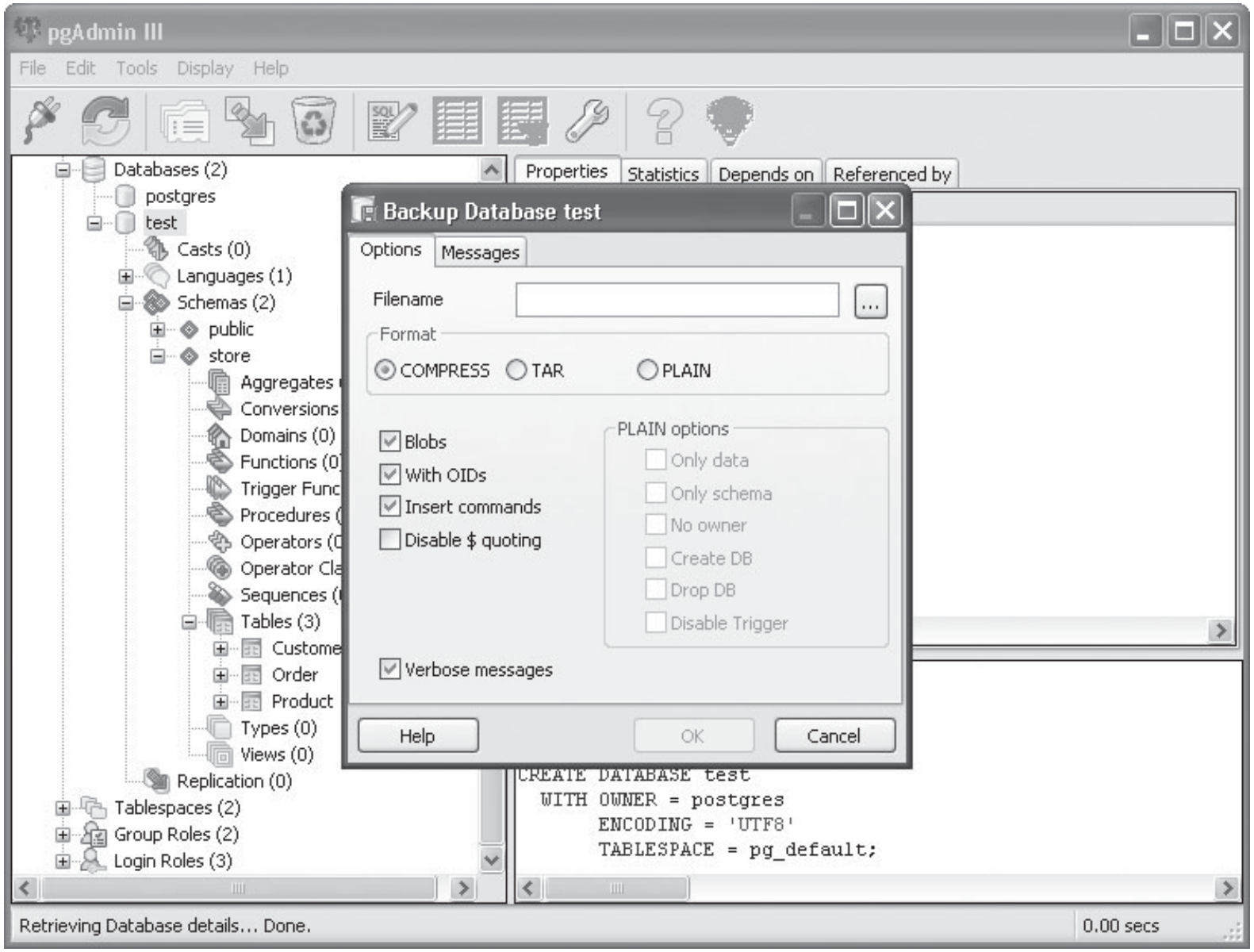
- log in using the test database and the fred Login Role
  - psql test fred
- test=>INSERT into store."Product" VALUES ('LAP001', 'Laptop', 'TakeAlong', 'Acme', '500.00', 100);
- test=>
- test =>INSERT into store. "Customer"("CustomerID", "LastName", "FirstName")VALUES ('Cus001', 'Thi Oanh', 'Nguyen');

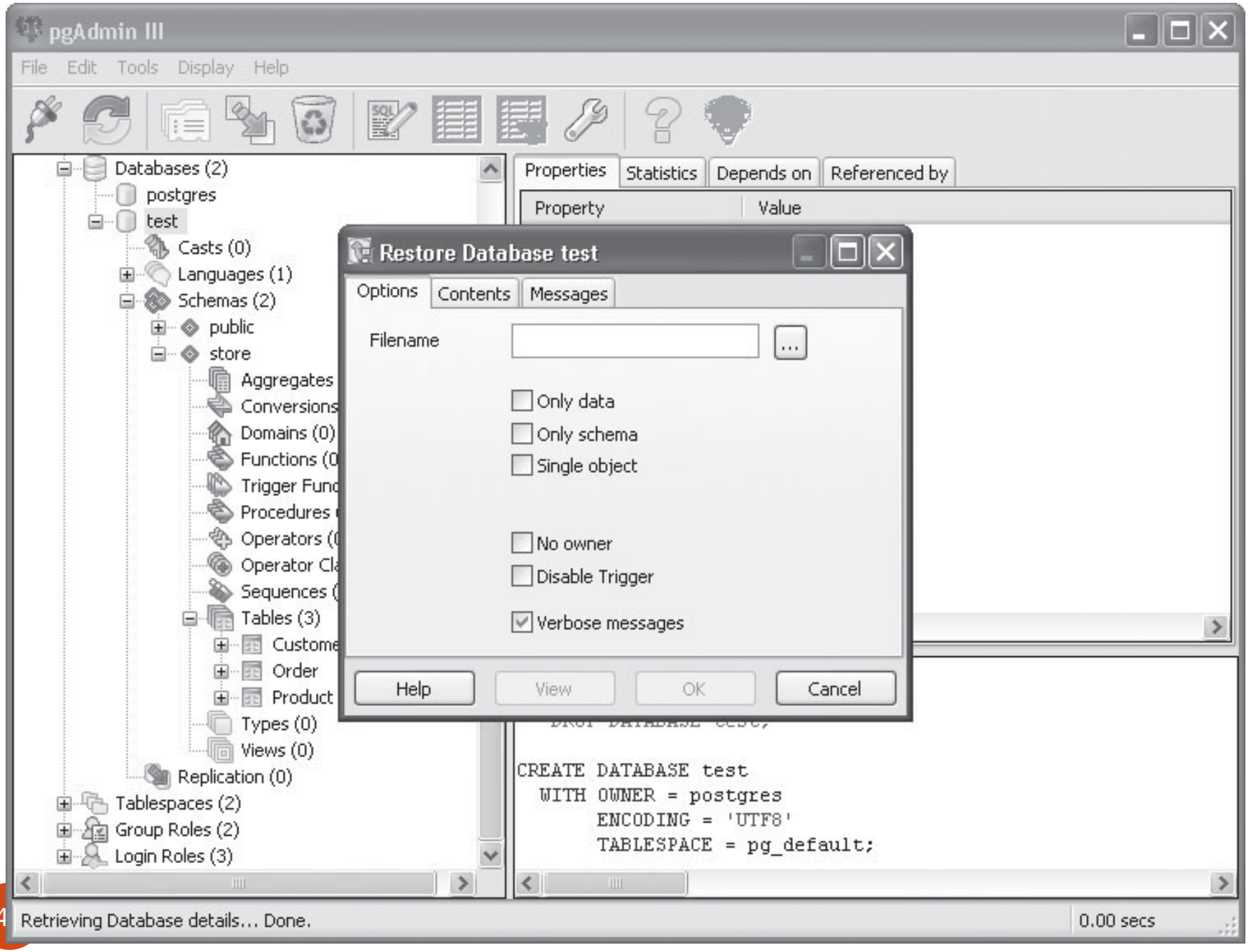
DATABASE MAINTENANCE




pgAdmin III
File Edit Tools Display Help
Databases (2)
postgres
test
Casts (0)
Languages (1)
Schemas (2)
public
store
Aggregates (0)
Conversions (0)
Domains (0)
Functions (0)
Trigger Functions
Procedures (0)
Operators (0)
Operator Classes
Sequences (0)
Tables (3)
Customer
Order
Product
Types (0)
Views (0)
Replication (0)
Tablespaces (2)
Group Roles (2)
Login Roles (3)
Properties
Statistics
Depends on
Referenced by
Property
Value
Name
test
Maintain Database test
Options
Messages
Maintenance operation
VACUUM
ANALYZE
REINDEX
VACUUM options
FULL
FREEZE
ANALYZE
REINDEX options
FORCE
RECREATE
Verbose messages
Help
OK
Cancel
-- DROP DATABASE test;
CREATE DATABASE test
WITH OWNER = postgres
ENCODING = 'UTF8'
TABLESPACE = pg_default;
Retrieving Database details... Done.
0.00 secs

BACKUPS AND RESTORES




pgAdmin III
File Edit Tools Display Help
Databases (2)
postgres
test
Casts (0)
Languages (1)
Schemas (2)
public
store
Aggregates
Conversions
Domains (0)
Functions (0
Trigger Func
Procedures
Operators (
Operator Cl
Sequences (
Tables (3)
Custome
Order
Product
Types (0)
Views (0)
Replication (0)
Tablespaces (2)
Group Roles (2)
Login Roles (3)
Properties
Statistics
Depends on
Referenced by
Property
Value
Restore Database test
Options
Contents
Messages
Filename
...
Only data
Only schema
Single object
No owner
Disable Trigger
Verbose messages
Help
View
OK
Cancel
CREATE DATABASE test
WITH OWNER = postgres
ENCODING = 'UTF8'
TABLESPACE = pg_default;
Retrieving Database details... Done.
0.00 secs