

Contents

1	Introdução	2
2	Migração do .NET / Mono para o Python	4
3	Instalação	4
3.1	OmniDB Application	5
3.2	OmniDB Server	6
4	Criando Usuários e Conexões	7
4.1	Efetuando login como usuário <i>admin</i>	7
4.2	Criando outro Usuário	8
4.3	Logando como novo usuário	10
4.4	Criando conexões	11
5	Workspace	13
5.1	Seções da janela <i>Workspace</i>	13
5.2	Aba Externa de Conexão	14
5.3	Trabalhando com banco de dados	16
5.4	Trabalhando com abas múltiplas dentro da mesma conexão . .	22
6	Gerenciamento de tabela	25
6.1	Criando tabelas	25
6.2	Editando tabelas	32
6.3	Removendo tabelas	33
7	Gerenciamento de dados	34
8	Editor de SQL	40
9	Gráfico com Tabelas e Relações	44
9.1	Gráfico simples	45
9.2	Gráfico Completo	46
10	Modelos SQL	48
11	Funcionalidades Adicionais	53
11.1	Histórico SQL	53
11.2	Configurações do usuário	53

12 Ferramenta de Configuração do OmniDB	55
12.1 Criar Super Usuário	56
12.2 Vacuum	56
12.3 Redefinir o Banco de Dados	57

1 Introdução

DBMS significa *database management system* (sistema de gerenciamento de banco de dados - SGBD). Pode ser uma função simples, biblioteca ou mesmo um sistema maior composto por vários programas e processos executando separadamente e em paralelo, cuja função principal é gerenciar um ou vários bancos de dados hospedados em um servidor. Tem a responsabilidade de manipular e manter a consistência dos dados, permitindo que os desenvolvedores de software se concentrem em suas funcionalidades. Assim, praticamente qualquer sistema moderno que gerencia dados utiliza algum tipo de SGBD, independentemente da quantidade de informações armazenadas.

Os criadores do **OmniDB**, Rafael Thofehrn Castro e William Ivanski, trabalharam em uma empresa onde precisavam lidar com vários bancos de dados de diferentes clientes diariamente. Esses bancos de dados eram de diferentes tecnologias de SGBD, e então eles precisavam continuar alternando entre as ferramentas de gerenciamento de banco de dados (normalmente uma para cada SGBD). Como eles não tinham uma ferramenta de gerenciamento de banco de dados unificadas (que podem gerenciar diferentes SGBD), surgiram com a ideia principal do **OmniDB**.

A primeira versão da **OmniDB** foi apresentada como um projeto final de graduação no Curso de Ciência da Computação da Universidade Federal do Paraná, no Brasil. O objetivo era traçar uma linha comum entre os SGBD populares e como tratavam seus metadados. O resultado foi uma ferramenta escrita em ASP.NET/C# capaz de conectar e identificar as estruturas principais (tabelas, chaves, índices e restrições), de forma genérica, de vários SGBDs:

- Firebird
- MariaDB / MySQL
- Oracle

- PostgreSQL
- SQLite
- Microsoft SQL Server

A primeira versão do OmniDB também permitiu a conversão entre todos os SGBDs suportados pela ferramenta. Este recurso foi desenvolvido para ser amigável, exigindo apenas algumas etapas: o usuário precisa selecionar uma conexão de origem, as estruturas que seriam convertidas (apenas tabelas e todas as suas estruturas, junto com seus dados) e a conexão de destino.

Desde o início do desenvolvimento, o OmniDB foi projetado como um aplicativo Web. Consequentemente, ele é executado em qualquer navegador, a partir de qualquer sistema operacional. Pode ser acessado por vários computadores e vários usuários, cada um deles com o seu próprio grupo de conexões. Também pode ser hospedado em qualquer sistema operacional, sem a necessidade de instalar quaisquer dependências. Veremos mais detalhes sobre a instalação nos próximos capítulos.

O objetivo principal do OmniDB é oferecer um espaço de trabalho unificado com todas as funcionalidades necessárias para manipular diferentes SGBDs. As ferramentas específicas do SGBD não são necessárias: no OmniDB, alternar entre diferentes SGBDs é feito com um simples interruptor de conexão, sem sair da mesma página. O usuário final tem a sensação de que não há diferença quando ele / ela manipula diferentes SGBDs, ele simplesmente se sente trabalhando com diferentes conexões.

Apesar disso, o OmniDB foi construído com a simplicidade em mente, projetado para ser uma aplicação web leve e veloz. OmniDB também é alimentado pelo WebSocket Technology, permitindo ao usuário executar múltiplas consultas e procedimentos em vários bancos de dados, em vários hosts, em segundo plano.

OmniDB também é seguro. Todos os dados do usuário OmniDB são armazenados criptografados e não salva a senha onde o banco de dados está armazenado. Quando o usuário se conecta pela primeira vez a um banco de dados, OmniDB solicita a senha. Esta senha é criptografada e armazenada em memória por um período específico de tempo. Quando esse prazo expirar, OmniDB pergunta novamente a senha. Isso garante a máxima segurança para o banco de dados que o OmniDB está se conectando.

2 Migração do .NET / Mono para o Python

OmniDB foi reescrito para Python usando o framework Django. Iniciando a partir da versão 2.0, o OmniDB versão Python receberá novos recursos e será ativamente mantido.

O código-fonte para a versão ASP.NET/C# está no branch `csharp`. O próximo lançamento do OmniDB versão C# é 1.7, e ela somente receberá correções de bugs.

O código fonte OmniDB está hospedado no GitHub e existem 3 branches principais: - **master**: Contém a versão beta atual do OmniDB versão Python - **dev**: Contém a versão de desenvolvimento atual do OmniDB versão Python - **csharp**: Contém a versão .NET / Mono do OmniDB

Além de ser escrito em Python, a versão inicial do OmniDB 2.0 contém as principais diferenças a partir da versão C#:

- Suporte para HTTPS;
- Permite a execução da consulta em segundo plano e cancelamento através do uso de websockets;
- Existe um novo recurso de Snippet;
- As capacidades de log e um conjunto de testes estão quase concluídos;
- Inicialmente, apenas um suporte aprimorado do PostgreSQL é implementado. Suporte de outros SGBDs em breve;
- O recurso de conversão de banco de dados foi desativado até agora, mas será reativado em breve;
- Você não precisa mais instalar dependências e servidores web. Tudo o que o OmniDB precisa agora está empacotado.

3 Instalação

OmniDB fornece 2 tipos de pacotes para atender a todas as necessidades dos usuários:

- **OmniDB Application**: Executa um servidor web em uma porta aleatória para trás, e fornece uma janela simplificada do servidor web

para usar a interface OmniDB sem qualquer configuração adicional. Funciona como uma aplicação de desktop.

- **OmniDB Server:** executa um servidor web em uma porta aleatória. O usuário necessita de um navegador Web para se conectar. Fornece gerenciamento de usuários e é ideal para ser hospedado em um servidor na rede dos usuários.

Tanto o OmniDB Application quanto o OmniDB Server podem ser instalados na mesma máquina.

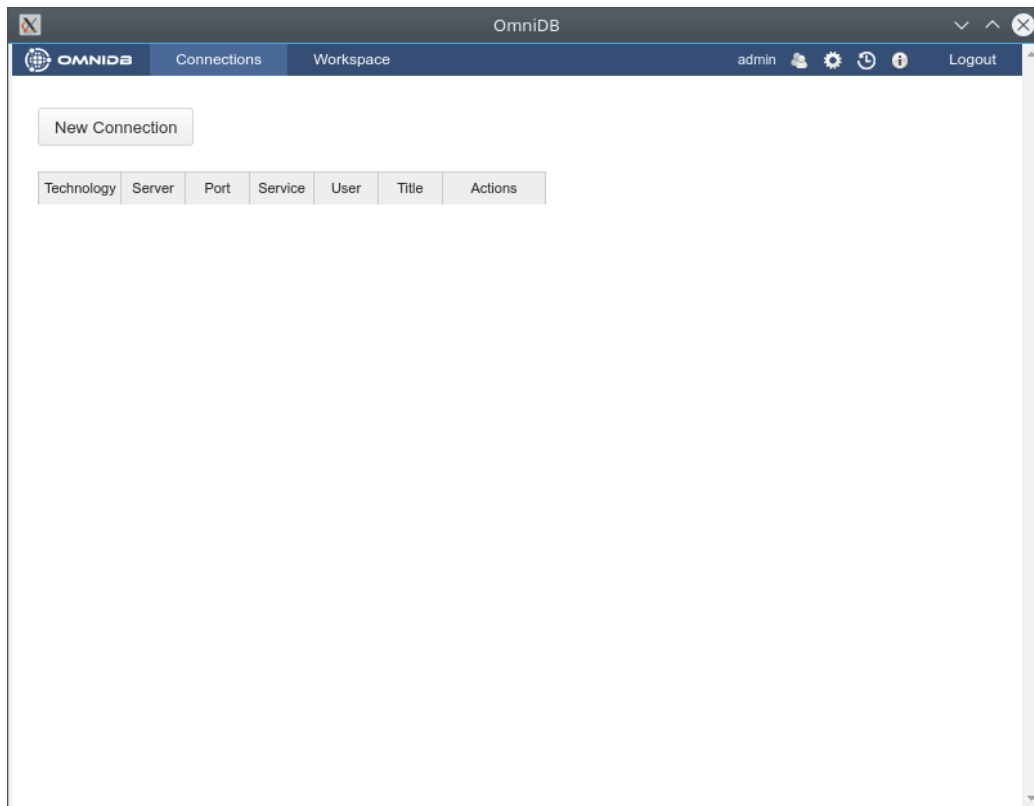
3.1 OmniDB Application

Para executar o aplicativo OmniDB, você não precisa instalar nenhuma peça adicional de software. Basta dirigir-se a omnidb.org e baixar o pacote mais recente para seu sistema operacional específico e arquitetura:

- Linux 32 bits / 64 bits
 - instalador DEB
 - instalador RPM
 - Tarball
- Windows 32 bits / 64 bits
 - instalador EXE
 - pacote ZIP
- Mac OS X
 - Instalador DMG
 - pacote ZIP

Se você escolhe pacotes tarball ou zip, basta extraí-lo em algum lugar do seu computador. Entre na pasta criada e abra o executável do `omnidb-app`. Ele abrirá OmniDB dentro de sua própria janela.

Com o instalador, você pode instalar o OmniDB no seu sistema, e estará disponível através do menu do seu aplicativo de ambiente de trabalho. Quando você abri-lo o OmniDB abrirá sua própria janela.



3.2 OmniDB Server

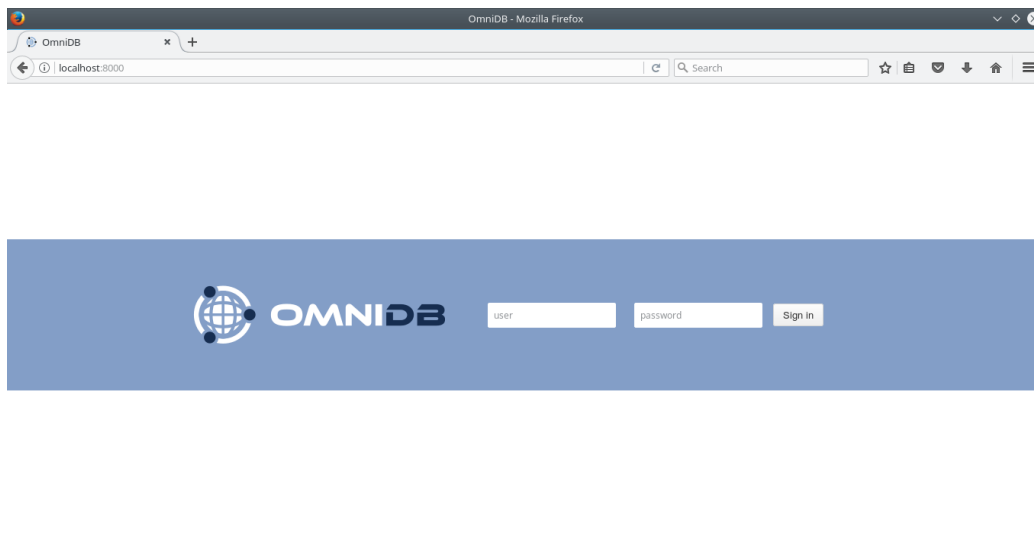
Como o OmniDB app, o OmniDB Server não requer nenhuma peça adicional de software e as mesmas opções para sistema operacional e arquitetura são fornecidas. Se você escolher o tarball ou o pacote zip, extraia-o em algum lugar do seu computador. Entre na pasta descompactada e abra o executável `omniDB-server`.

```
User@machine:~$ cd omnidb-server
User@machine:~ / omnidb-server$ ./omnidb
Iniciando o OmniDB 2.0.2 em http://localhost: 8000
Abra OmniDB no seu navegador favorito
Pressione Ctrl + C para sair
```

Para instalar o OmniDB Server, você precisará de privilégios de administrador:

```
User@machine:~ $ sudo omnidb-server
Iniciando o OmniDB 2.0.2 em http: // localhost: 8000
Abra OmniDB no seu navegador favorito
Pressione Ctrl + C para sair
```

Agora que o servidor web está em execução, você pode acessar o aplicativo da web OmniDB em seu navegador favorito. Digite na barra de endereços: `localhost: 8000` e tecla **Enter**. Se tudo correu bem, você verá uma página como esta:

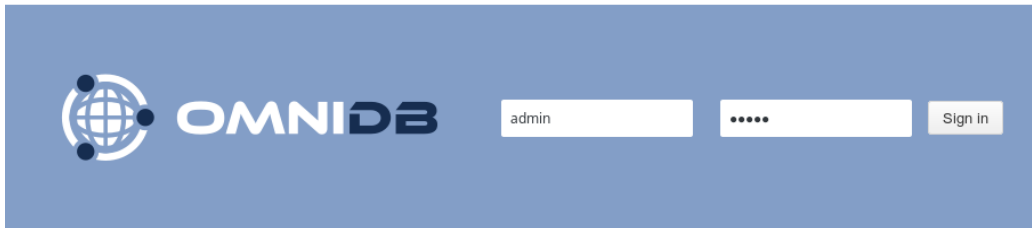


Agora você sabe que o OmniDB está funcionando corretamente. Nos próximos capítulos, nós veremos como fazer o login pela primeira vez, como criar um usuário e utilizar OmniDB.

4 Criando Usuários e Conexões

4.1 Efetuando login como usuário *admin*

O OmniDB vem apenas com o usuário *admin*. Se você estiver usando a versão do servidor, a primeira coisa é fazer login como *admin*, a senha padrão é *admin*. Você não precisa fazer login na versão do aplicativo.



A próxima janela é a janela **Connections**. Vamos falar sobre isso mais tarde.

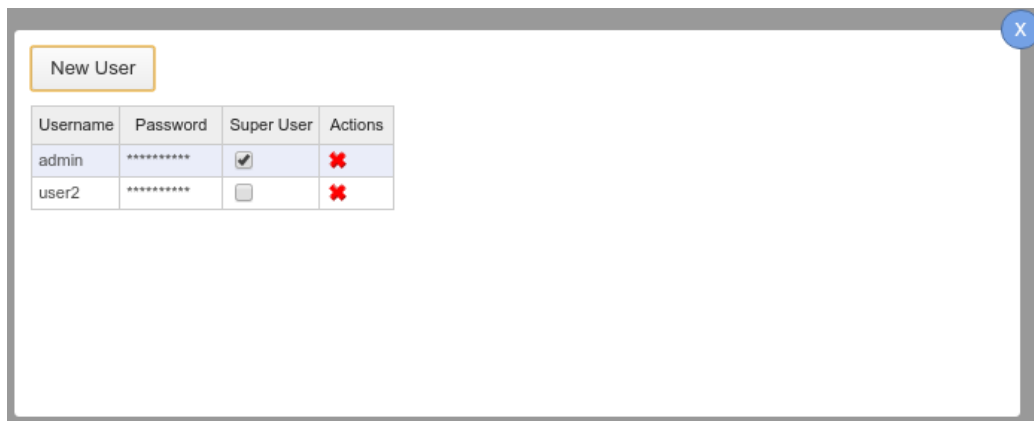


4.2 Criando outro Usuário

Clique no ícone *Users* no canto superior direito. Ele abrirá um pop-up que permite que o super usuário OmniDB atual crie um novo usuário OmniDB.



Depois de clicar no ícone *Users*, a ferramenta insere um novo usuário chamado *user2* (se esse é o primeiro usuário após o *admin*).



Você terá que mudar *username* e *password*. Marque se você deseja que o novo usuário seja um super usuário. Esta janela de gerenciamento de usuários só é acessada por super usuários. Quando terminar, clique no botão *Save Data* dentro do pop-up.

Username	Password	Super User	Actions
admin	*****	<input checked="" type="checkbox"/>	✖
test	****	<input type="checkbox"/>	✖

Você pode criar tantos usuários quanto quiser, editar usuários existentes e também excluir usuários clicando na cruz vermelha na coluna Actions. Agora você pode sair.

4.3 Logando como novo usuário

Vamos logar como o usuário que acabamos de criar.

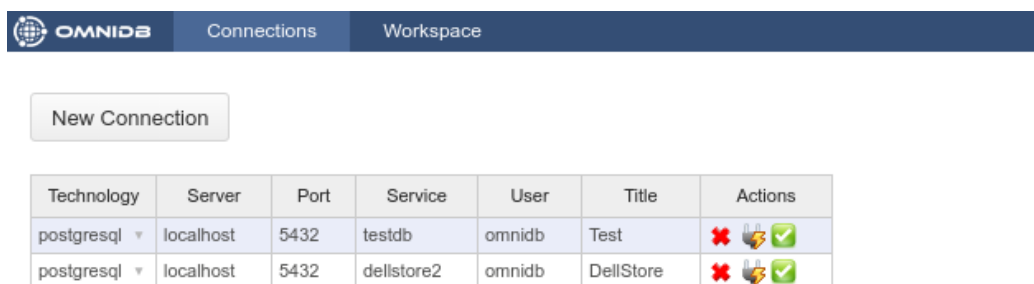
E podemos ver a janela **Connections** novamente. Observe que agora não há o ícone *Users*, porque o usuário de teste não é um super usuário.

Technology	Server	Port	Service	User	Title	Actions
------------	--------	------	---------	------	-------	---------

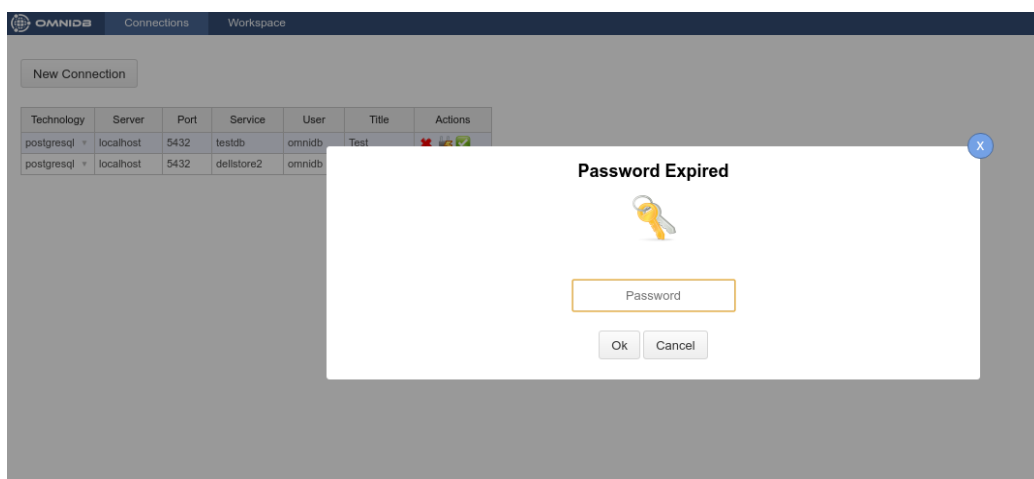
4.4 Criando conexões

O OmniDB versão C# suporta vários SGBDs. No momento, o OmniDB versão Python, ou OmniDB 2.0, suporta apenas o PostgreSQL. O suporte a mais SGBDs está sendo adicionado enquanto você lê isso.

Agora vamos criar duas conexões aos bancos de dados PostgreSQL. Para criar as conexões você deve clicar no botão *New Connection* e, em seguida, escolher a conexão e preencher os outros campos. Depois de preencher todos os campos para ambas as conexões, clique no botão *Save Data*.

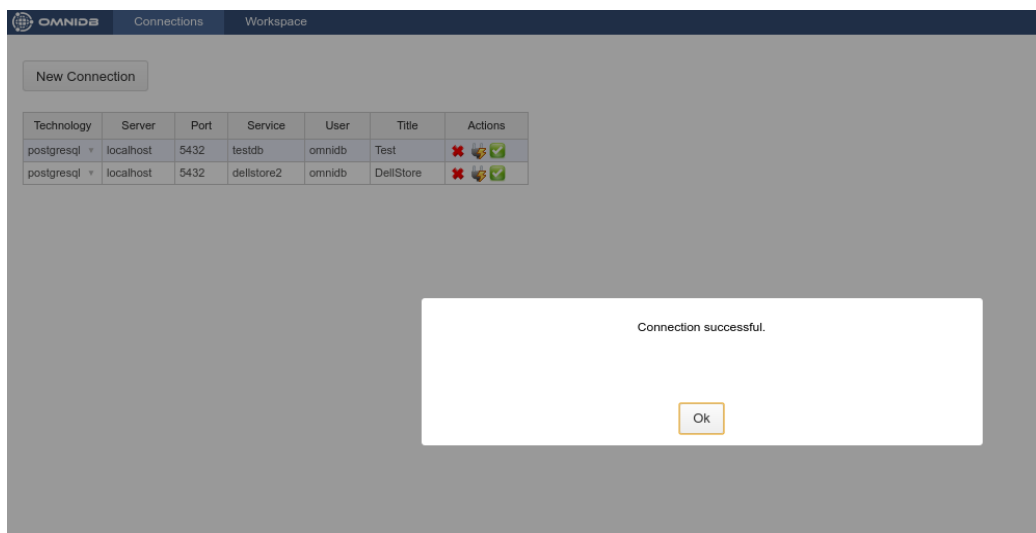


Para cada conexão há uma coluna *Actions* onde você pode excluir, testar e selecioná-la. Vá em frente e teste uma das conexões.

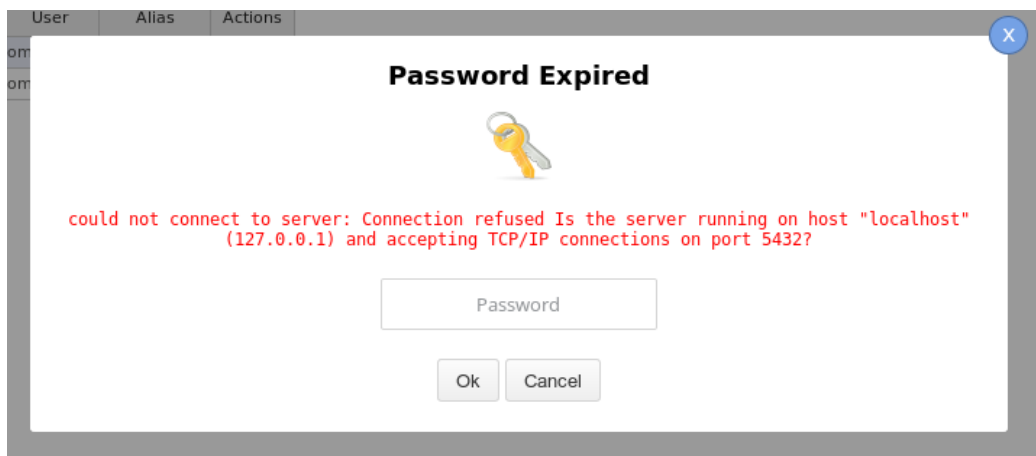


Observe o menu pop-up *Password Expired*. Isso está acontecendo porque OmniDB não armazena a senha do usuário do banco de dados no disco. Quando o usuário digita uma senha neste pop-up, a senha é criptografada e armazenada na memória.

Após digitar a senha e pressionar *Enter*, se a conexão ao banco de dados for bem sucedida você verá um pop-up de confirmação.



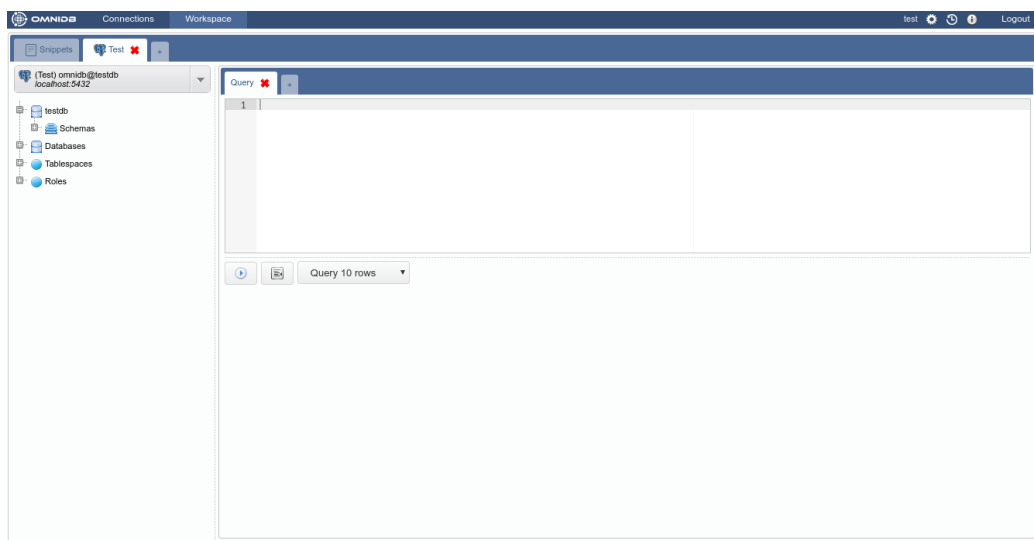
Mas, se você tiver problemas de qualquer tipo na conexão ao seu banco de dados PostgreSQL, o pop-up *Password Expired* continuará mostrando o erro que o OmniDB obteve.



Além disso, na grade de conexões, se você clicar na ação *Select Connection*, o OmniDB irá abrir na janela *Workspace*.

5 Workspace

Depois de criar pelo menos uma conexão, o usuário pode entrar no *Workspace*, seja clicando na guia *Workspace* ou clicando na ação *Select Connection* na grade de conexões.



5.1 Seções da janela *Workspace*

Esta interface possui vários elementos:

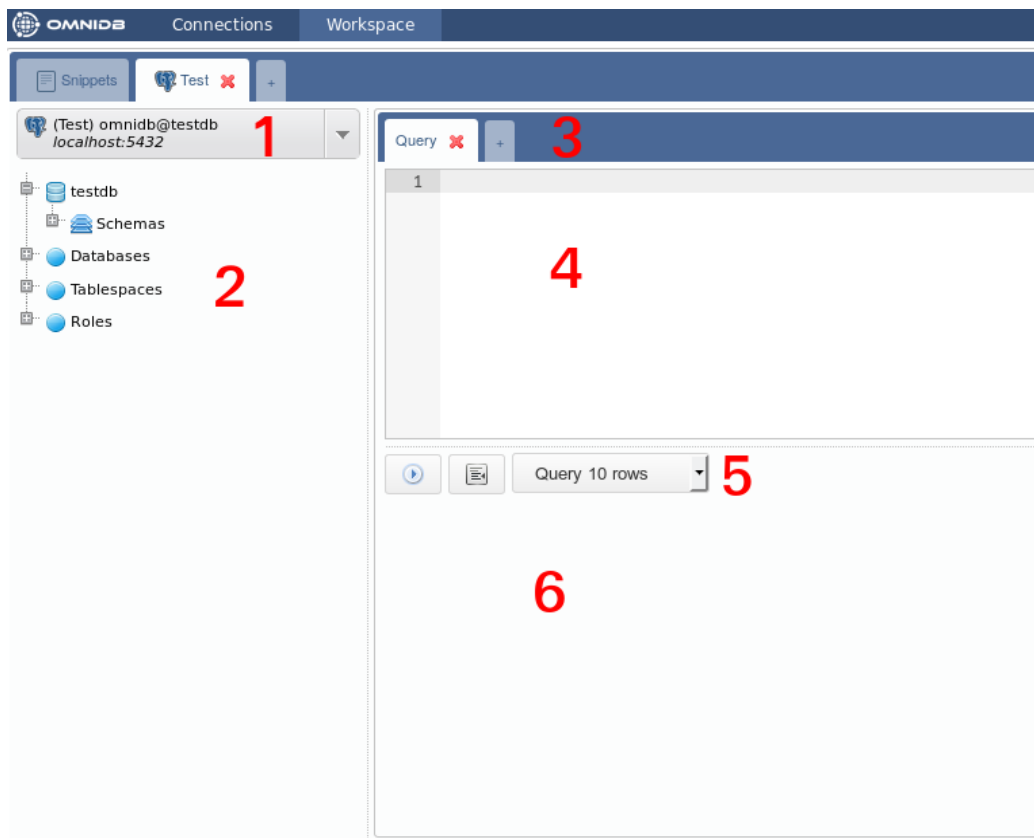


- 1) **Links:** permite ao usuário navegar entre as janelas do OmniDB

- **2) Abas Externas:** OmniDB permite que você trabalhe com vários bancos de dados ao mesmo tempo. Cada banco de dados será acessível através de uma aba externa. As abas externas também podem conter recursos diversos, como o recurso *Snippets*
- **3) Opções:** mostra o usuário atual logado, e também links para *user settings*, *query history*, *information* e *logout*.

5.2 Aba Externa de Conexão

Então, a aba externa chamada *Test* tem esse nome por causa do alias que colocamos na conexão ao `testdb`. Esta aba é uma *aba de conexão externa*. Observe a pequena aba com uma cruz além da aba *Test*. Ela permite que você crie uma nova aba externa que será automaticamente uma *aba externa de conexão*. No entanto, a *aba externa Snippet* é fixa e sempre será a primeira. Uma nova *aba externa de conexão* sempre apontará automaticamente para a primeira conexão em sua lista de conexões de banco de dados. Ou, se você clicar no *Selection Connection*, ela apontará para a conexão selecionada. Observe os elementos dentro desta guia:



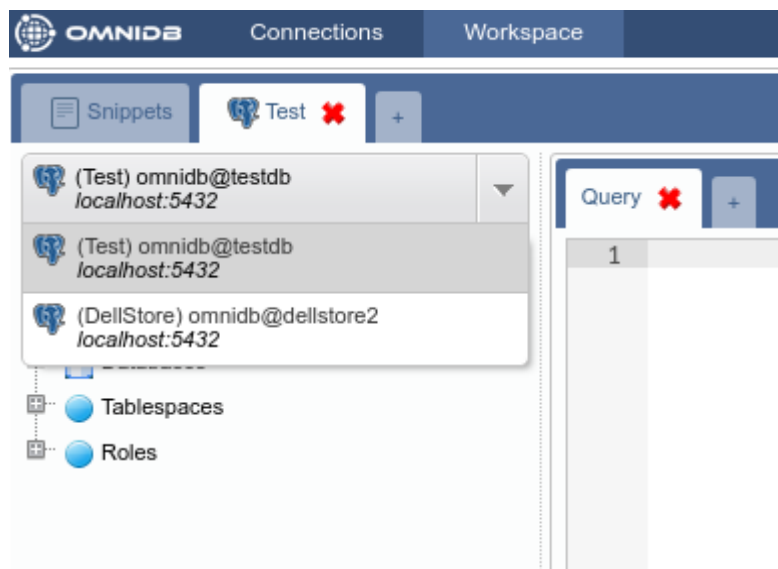
- **1) Seletor de conexão:** mostra todas as conexões e permite ao usuário selecionar a atual
- **2) Árvore de estruturas:** exibe uma árvore hierárquica onde você pode navegar pelos elementos do banco de dados
- **3) Abas Internas:** Permite que o usuário execute ações na corrente base de dados. Existem vários tipos de abas internas para o banco de dados atual. Ao clicar na última aba com uma cruz, você pode adicionar uma nova aba. Uma nova aba sempre será uma *aba de consulta*, onde você pode escrever qualquer tipo de declaração SQL
- **4) Conteúdo da aba interna:** pode variar de acordo com o tipo de aba interna. A figura mostra uma *aba de consulta* e, neste caso, o conteúdo será um *Editor de SQL*, com destaque de sintaxe e preenchimento automático
- **5) Ações da aba interna:** pode variar dependendo do tipo de aba interna. Para uma *aba de consulta*, eles são: *Execute Button*, *Format*

Button e Editor Mode (script, executar ou consulta)

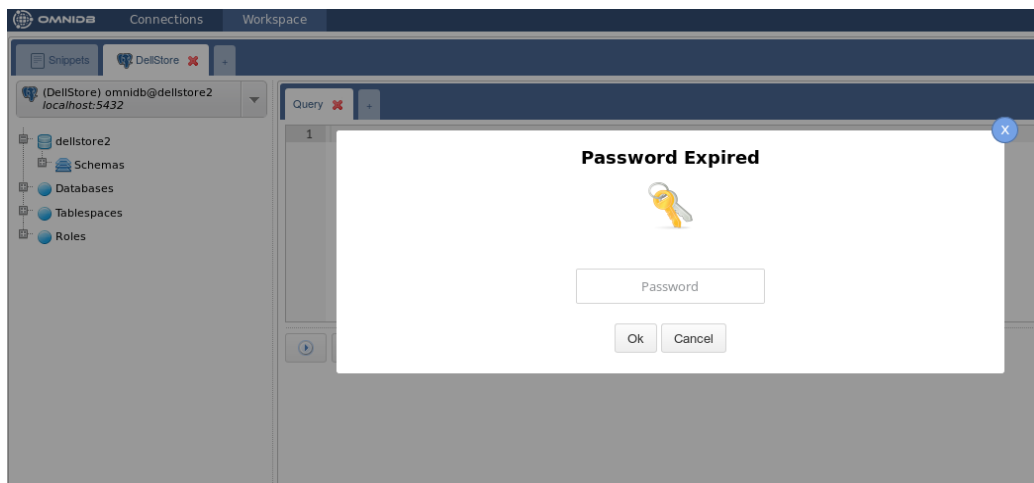
- **6) Resultados da aba interna:** uma *aba de consulta* no modo de consulta, depois de clicar no botão *Execute* ou digitar o atalho de execução (**Alt-Q**), mostrará uma grade com os resultados da consulta. Todos os modos mostrarão mensagens de erro, se houver.

5.3 Trabalhando com banco de dados

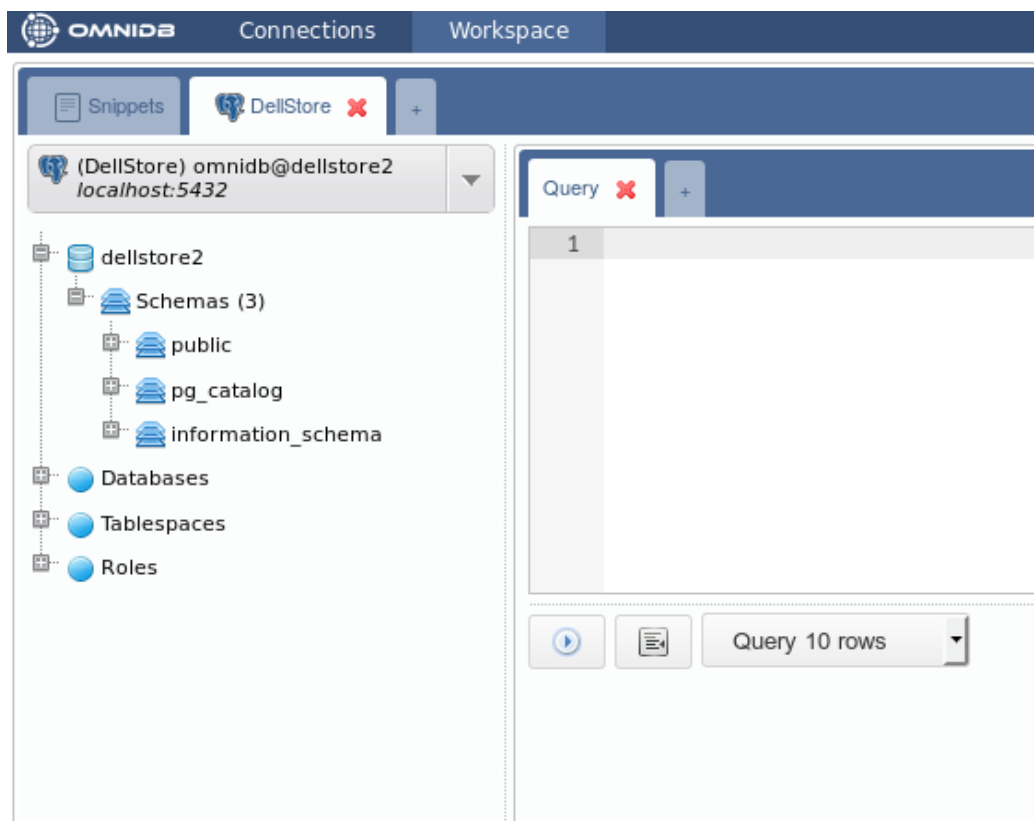
Dê uma olhada em seu seletor de conexões. OmniDB sempre aponta para a primeira conexão disponível, mas você pode alterá-lo clicando no seletor.



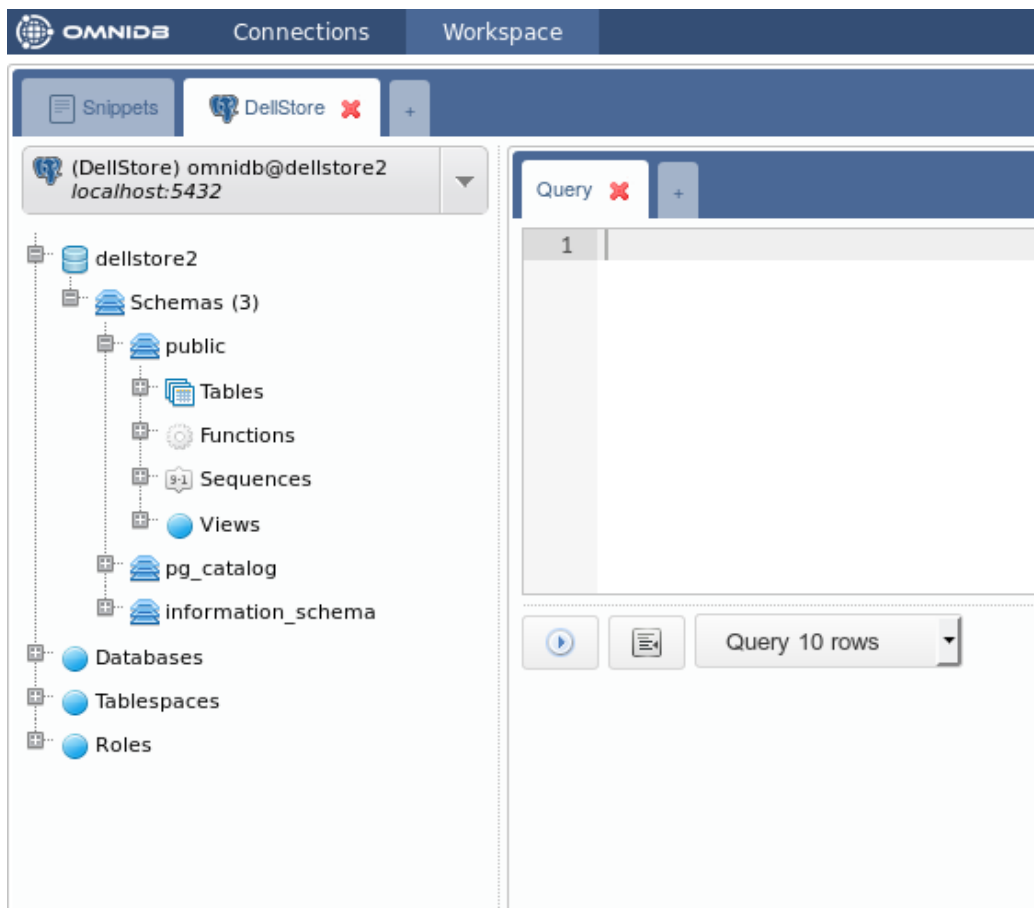
Selecione a conexão *DellStore*. Agora vá para a árvore logo abaixo do seletor e clique para expandir o nó *Schemas*.



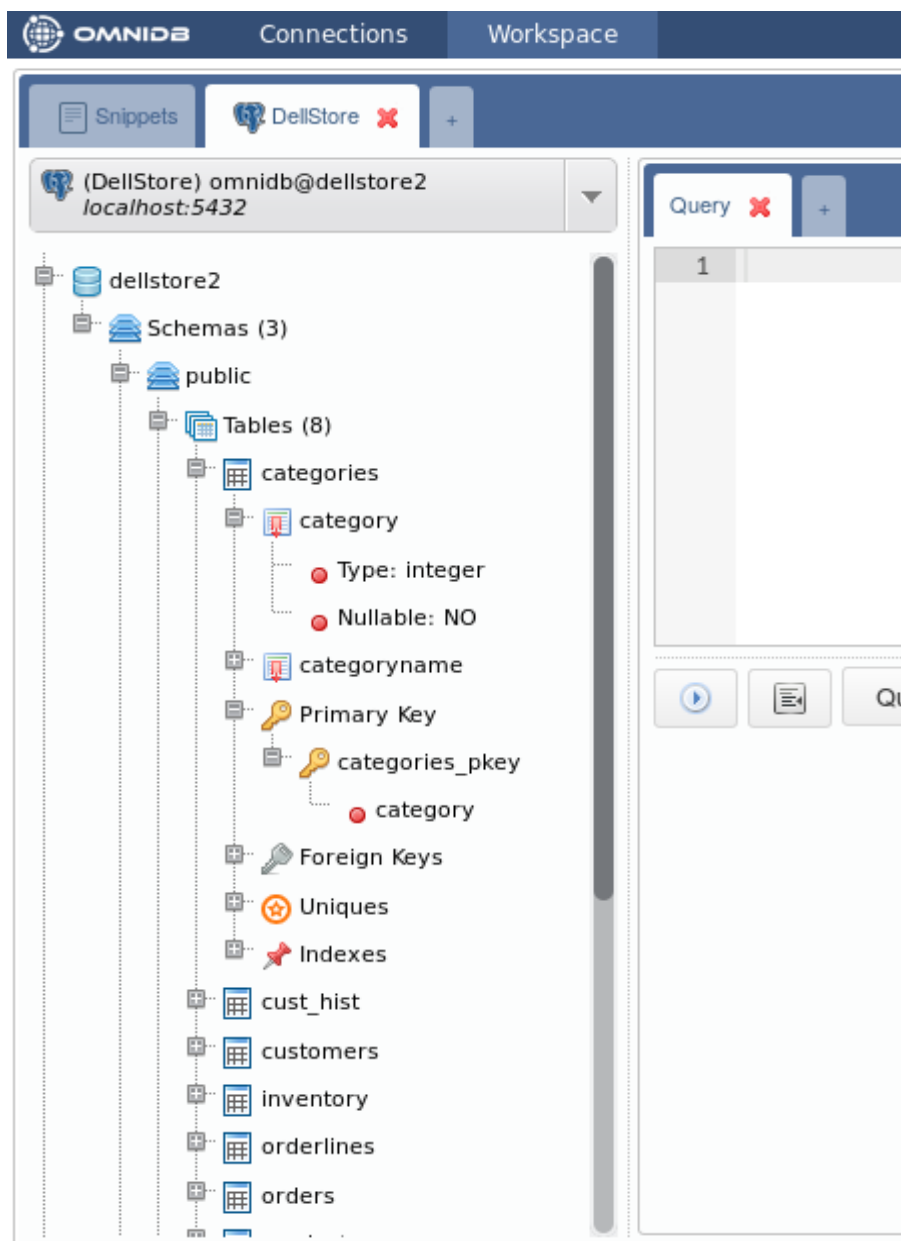
Tenha em mente que a cada 10 minutos sem executar ações no banco de dados, irá ativar um pop-up *Password Expired*. Como explicado anteriormente, isso é importante para a segurança do seu banco de dados. Depois de digitar a senha correta, você verá todos os schemas no seu banco de dados (no caso do PostgreSQL, TOAST e temp schemas não são mostrados).



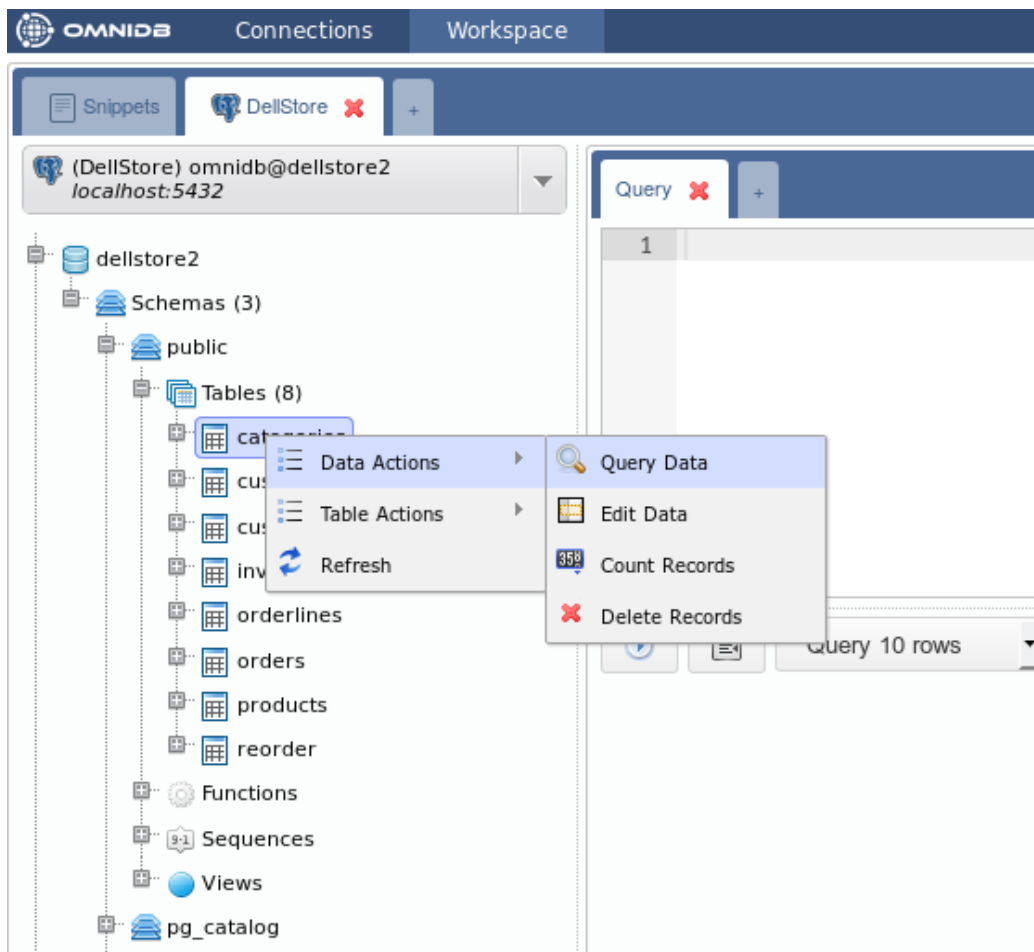
Agora, clique para expandir o schema `public`. Você verá diferentes tipos de elementos contidos neste schema.



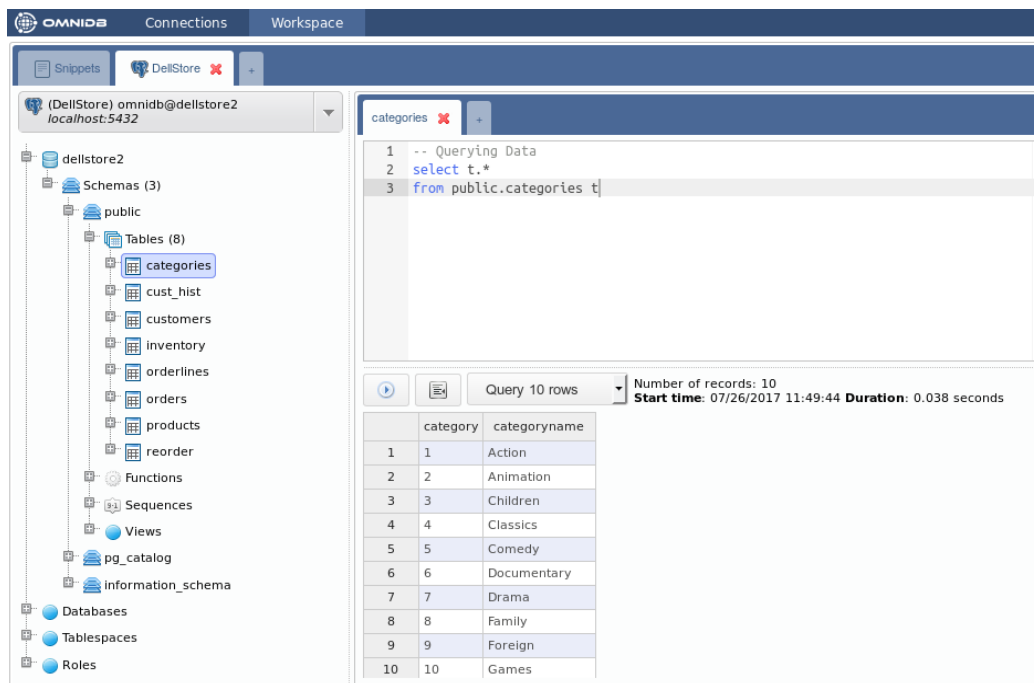
Agora clique para expandir o nó *Tables*, e você verá todas as tabelas contidas no schema *public*. Expanda qualquer tabela e você verá suas colunas, chave primária, chaves estrangeiras, restrições e índices únicos. Cada coluna é também expansível, apresentando tipo de dados e restrição anulável.



Para ver registros dentro de uma tabela, clique com o botão direito e escolha *Data Actions > Query Data*.



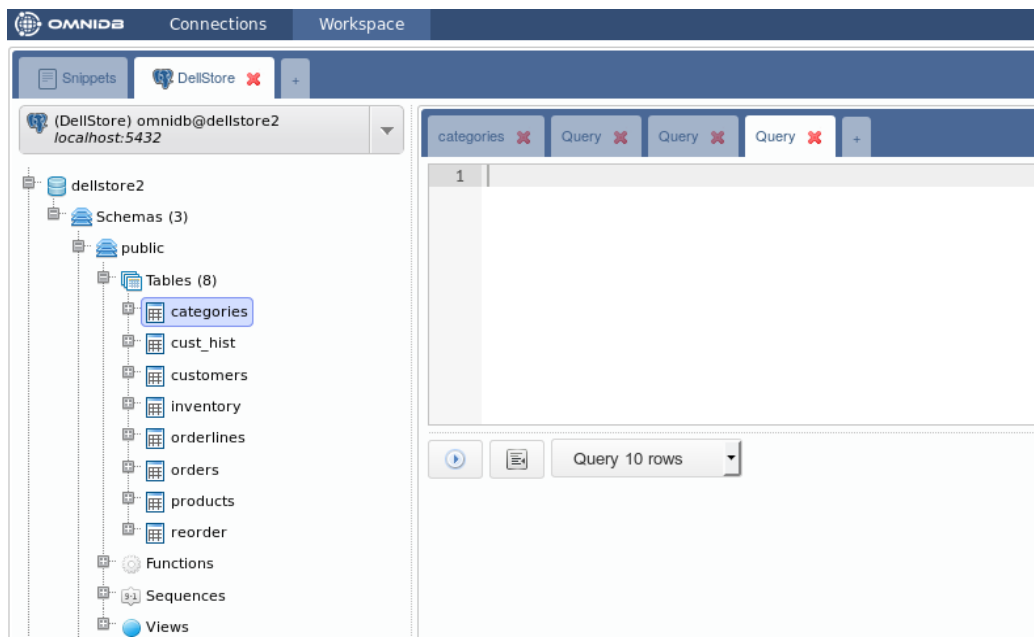
Observe que o OmniDB preenche o editor atual do SQL com uma consulta simples para listar os registros da tabela. Os registros são exibidos em uma grade logo abaixo do editor. Esta grade pode ser controlada com o teclado como se estivesse usando uma planilha. Você também pode copiar dados de células únicas ou blocos de células (que pode ser selecionado com o teclado ou mouse) e colar em qualquer planilha.



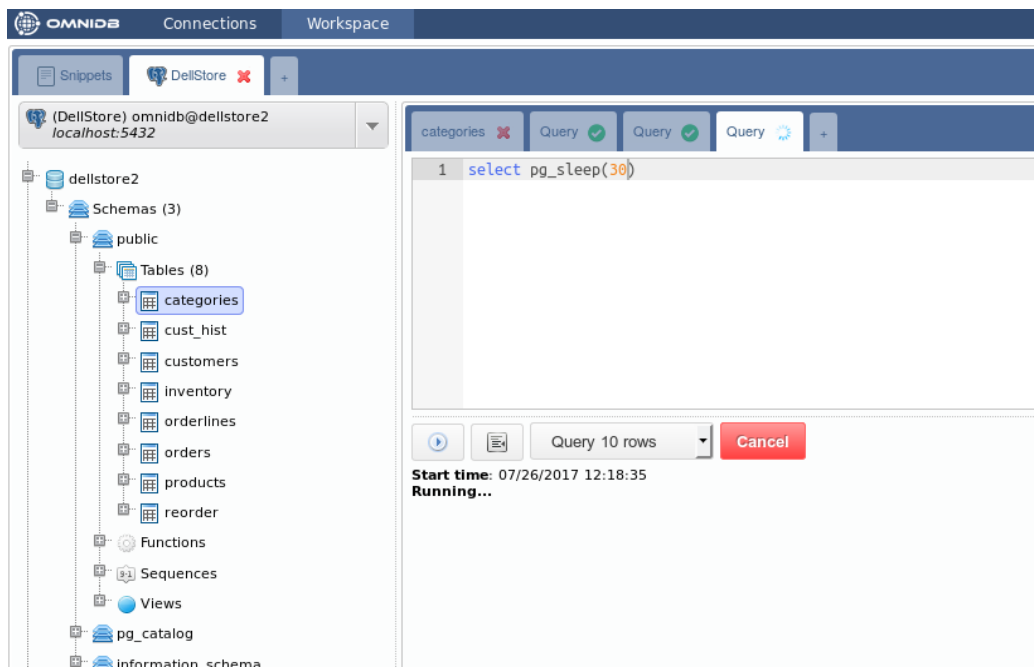
Você pode editar a consulta no editor do SQL, escrevendo simples ou complexas consultas e executá-las clicando no botão action. Você pode controlar quantos registros devem ser exibidos (10, 100, 1000 ou todas as linhas). Mais detalhes nos próximos capítulos.

5.4 Trabalhando com abas múltiplas dentro da mesma conexão

Dentro de uma única conexão, você pode criar várias abas internas clicando na última aba com uma cruz. Cada nova aba interna será uma *aba de consulta*.



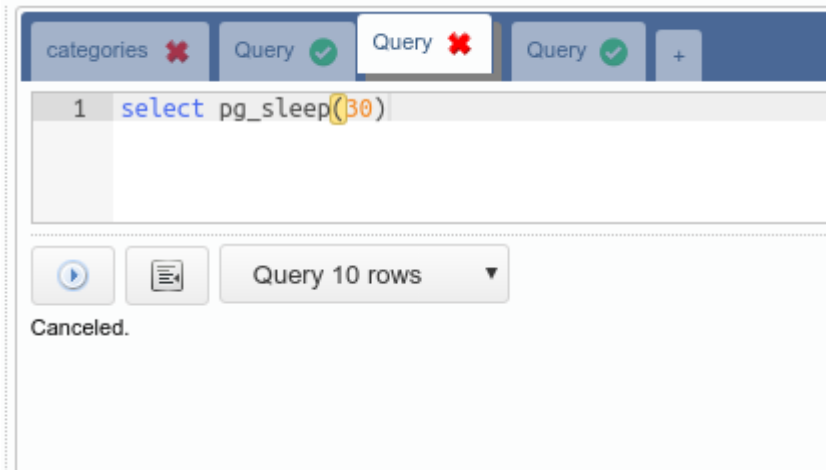
No OmniDB, você pode executar várias instruções SQL e procedimentos em paralelo. Quando está sendo executado, um ícone será mostrado na aba para indicar o estado atual. Se algum processo estiver concluído e não estiver na aba atual, essa aba mostrará um ícone verde para indicar que a rotina que está sendo executada agora terminou.



Ao clicar no botão *Cancel*, você pode cancelar um processo que está sendo executado dentro do banco de dados.



Você também pode arrastar e soltar uma guia para alterar sua ordem. Isso funciona com as abas internas e externas.



Além disso, você pode usar atalhos de teclado para gerenciar abas internas (Consulta SQL) e abas externas (Conexão): - **Ctrl-Insert**: Insere uma nova aba interna - **Ctrl-Delete**: remove uma aba interna - **Ctrl- <**: Muda o foco para a aba interna à esquerda - **Ctrl- >**: Muda o foco para a aba interna à direita - **Ctrl-Shift-Insert**: Insere uma nova aba externa - **Ctrl-Shift-Delete**: remove uma aba externa - **Ctrl-Shift- <**: Muda o foco para a aba externa à esquerda - **Ctrl-Shift- >**: Muda o foco para a aba externa à direita

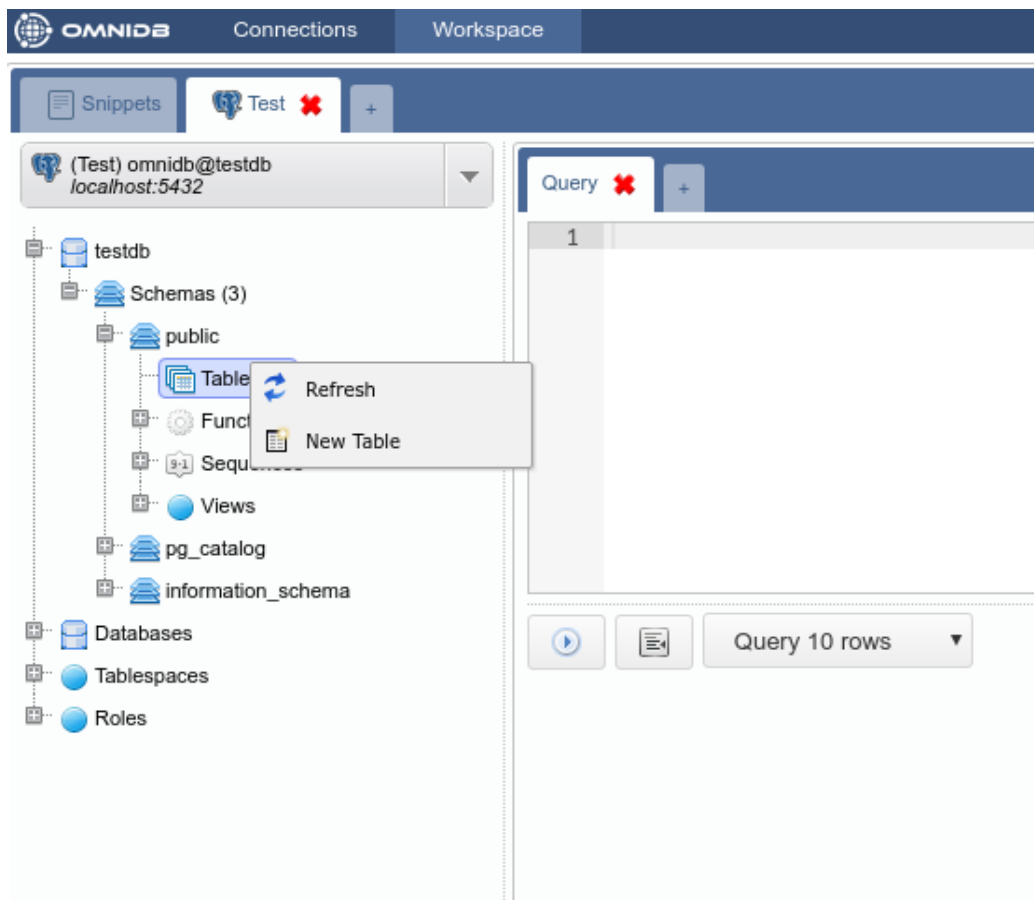
6 Gerenciamento de tabela

6.1 Criando tabelas

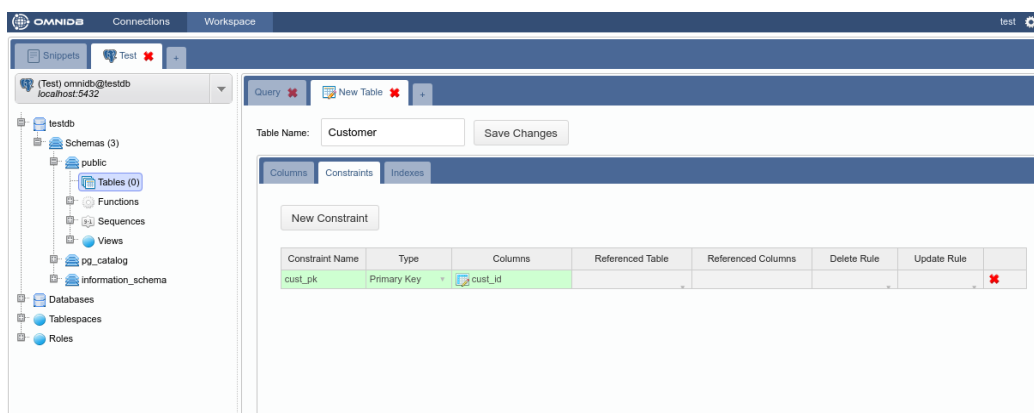
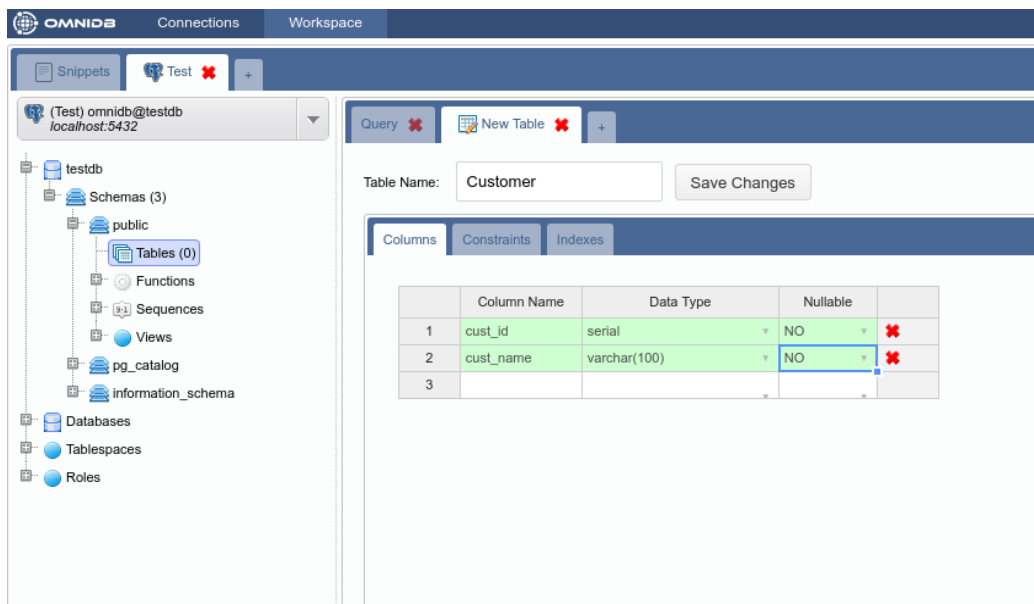
OmniDB possui uma interface de criação de tabelas que permite configurar colunas, restrições (constraints) e índices. Algumas observações devem ser mencionadas: - A maioria dos SGBDs cria automaticamente índices quando as chaves primárias e exclusivas são criadas. Por isso, a guia de índices só está disponível depois de criar a tabela. - Cada SGBD tem suas características e limitações únicas na criação de tabelas e a interface OmniDB reflete essas limitações. Por exemplo, o SQLite não nos permite alterar as colunas existentes e restrições (constraints). Por isso, a interface nos permite mudar apenas o nome da tabela e adicionar novas colunas ao lidar com bancos de dados

SQLite (este ainda não é o caso da versão OmniDB Python, pois atualmente suporta apenas banco de dados PostgreSQL).

Vamos criar tabelas de exemplo (*Customer* e *Address*) no banco de dados **testdb** que conectamos anteriormente. Clique com o botão direito do mouse no nó **Tables** e selecione **New Table**:



Vamos criar a tabela *Customer* com uma chave primária que será referenciada pela Tabela *Address*:



Observe como a tabela aparece no nó *Tables* da árvore:

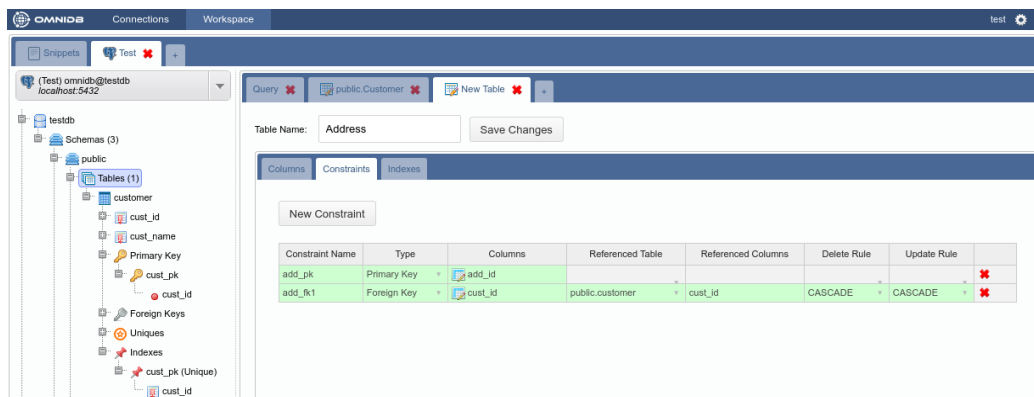
Table Name:

	Column Name	Data Type	Nullable	
1	cust_id	integer	NO	✗
2	cust_name	character varying(100)	NO	✗
3				

Agora, crie a tabela *Address* com uma chave primária e uma chave estrangeira.

Table Name:

	Column Name	Data Type	Nullable	
1	add_id	serial	NO	✗
2	add_street	varchar(200)	NO	✗
3	add_number	integer	YES	✗
4	cust_id	integer	NO	✗
5				



Neste ponto, temos duas tabelas no schema **public**. A estrutura do schema pode ser vista com o recurso de gráfico clicando com o botão direito no schema **public** da árvore e selecionando *Render Graph > Simple Graph*: At this point we have two tables in schema **public**. The schema structure can be seen with the graph feature by right clicking on the schema **public** node of the tree and selecting *Render Graph > Simple Graph*:

OMNIDB

Connections

Workspace

Snippets

Test

+

(Test) omnidb@testdb
localhost:5432

testdb

Schemas (3)

Functions

Sequences

Views

pg_catalog

information_schema

Databases

Tablespaces

Roles

Query

public.Customer

Table Name: Address

Constraints

Indexes

Render Graph

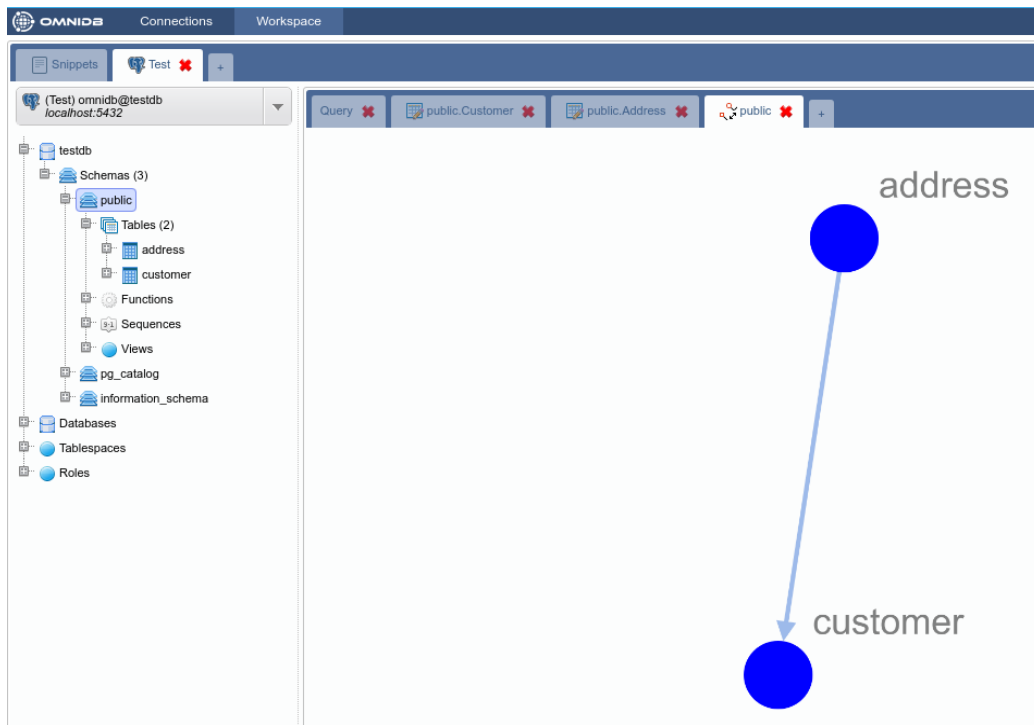
Alter Schema

Drop Schema

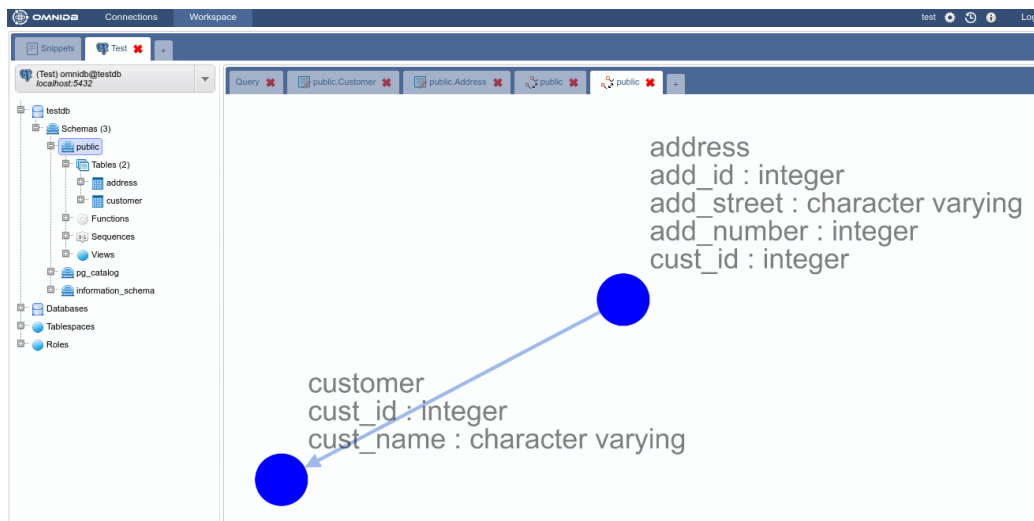
Simple Graph

Complete Graph

	Column Name	
1	add_id	integer
2	add_street	character
3	add_number	integer
4	cust_id	integer
5		

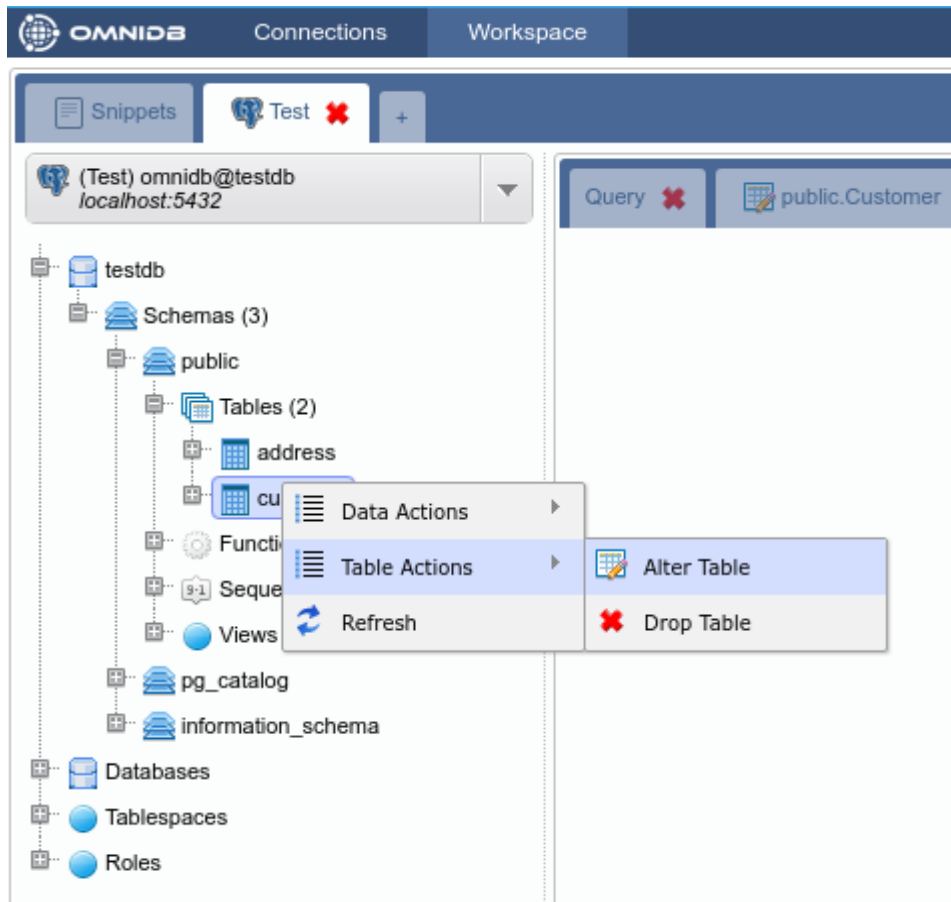


E é assim que o *Complete Graph* exhibe:

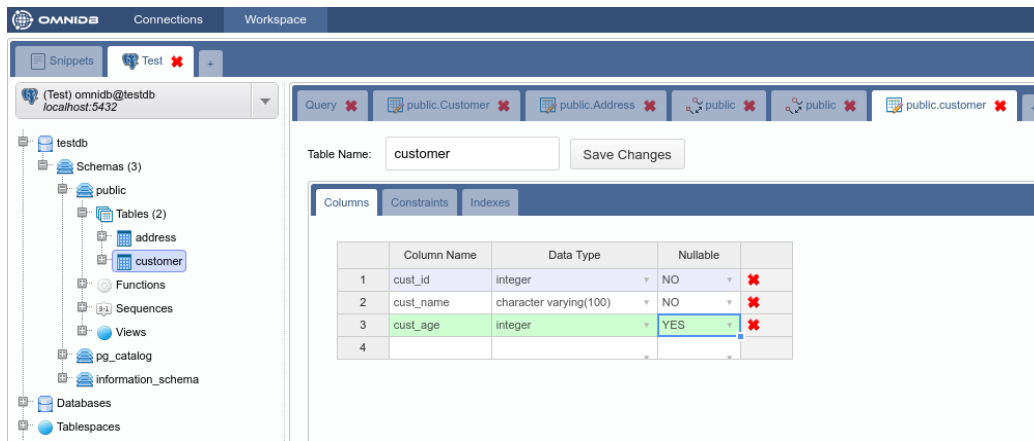


6.2 Editando tabelas

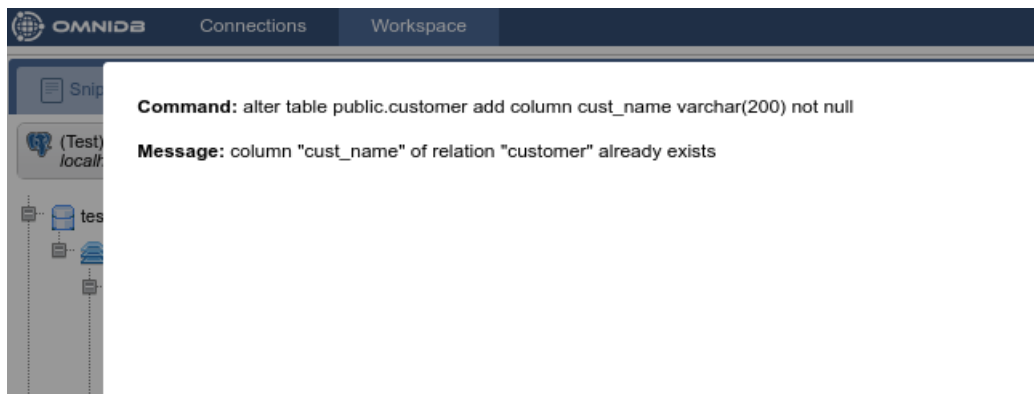
O OmniDB também permite que você edite tabelas existentes (sempre seguindo as limitações do SGBD). Para testar este recurso, adicionaremos uma nova coluna à tabela *Customer*. Para acessar a interface alter table apenas clique com o botão direito do mouse no nó da tabela e selecione *Table Actions > Alter Table*:



Adicione a coluna *cust_age* e salve:

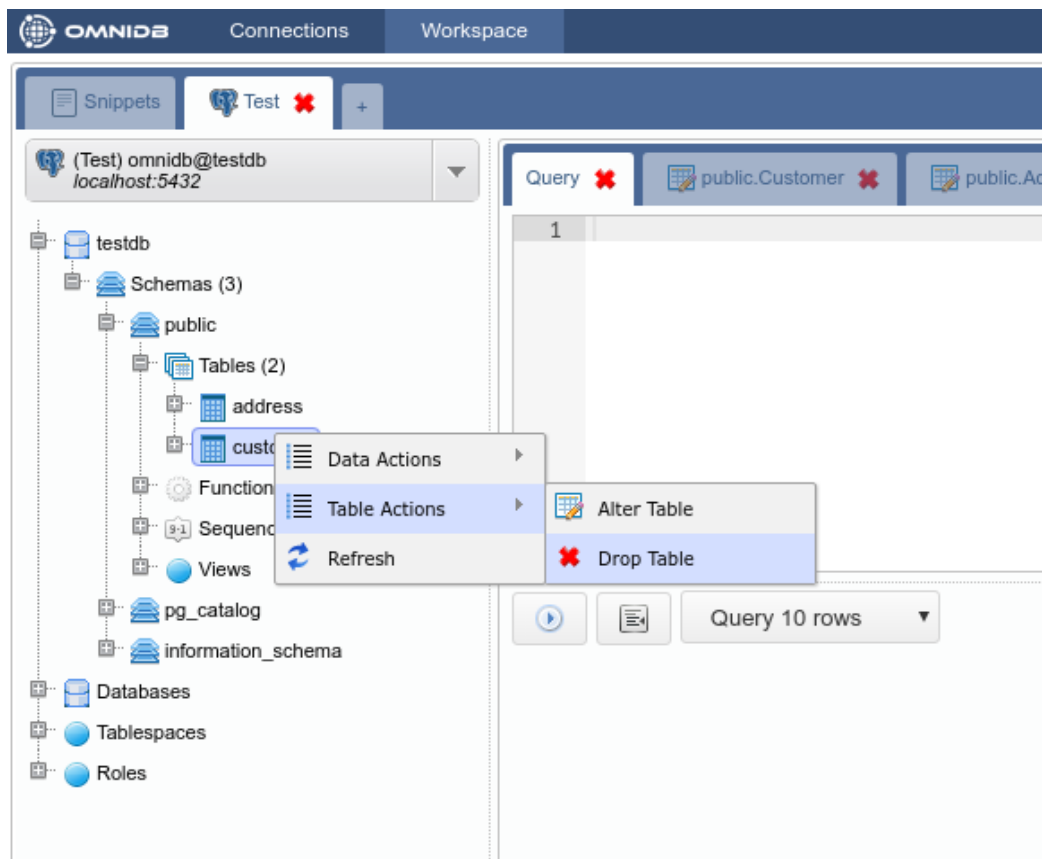


A interface é capaz de detectar erros que podem ocorrer durante as operações Alter Table, mostrando o comando e o erro que ocorreu. Para demonstrar tentaremos adicionar a coluna *cust_name*, que já pertence a esta tabela:



6.3 Removendo tabelas

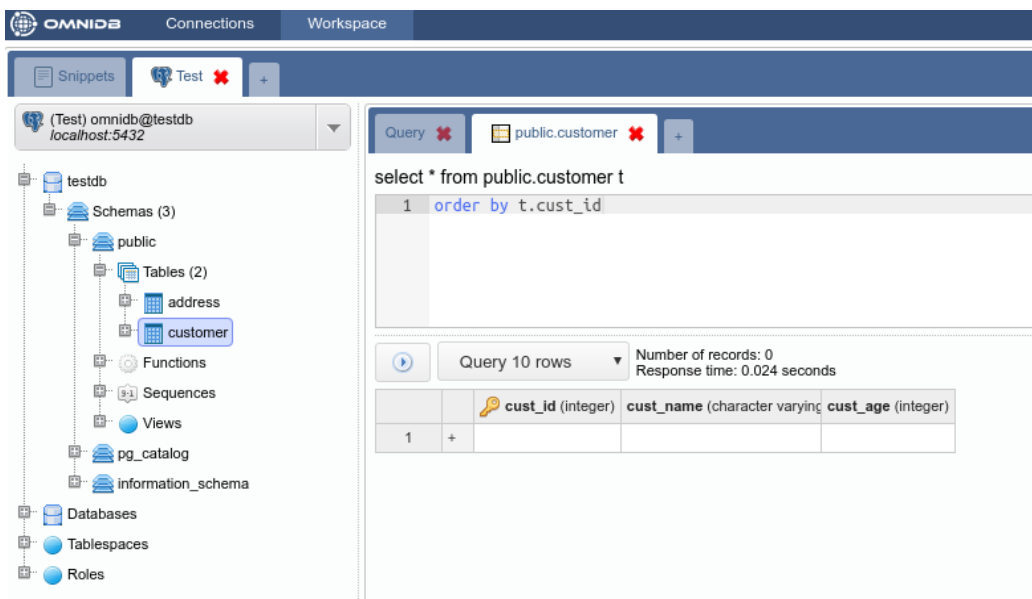
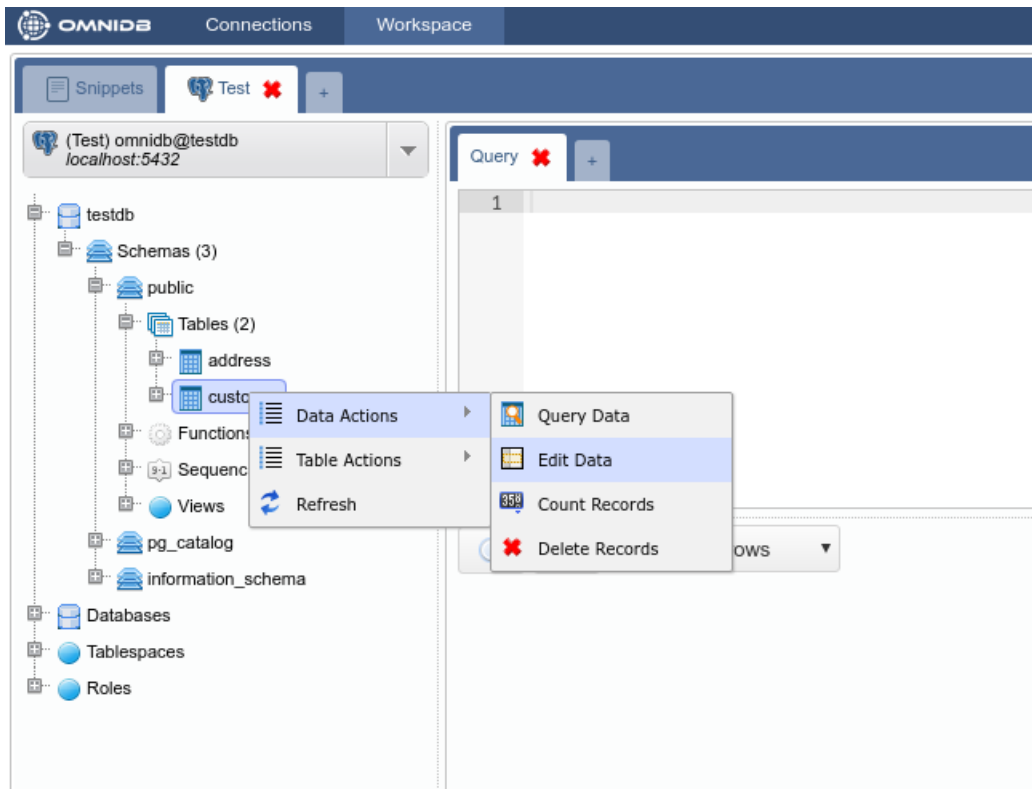
Para remover uma tabela, clique com o botão direito do mouse no nó da tabela e selecione *Table Actions > Drop Table*:



7 Gerenciamento de dados

A ferramenta nos permite editar os registros contidos nas tabelas através de uma simples e intuitiva interface. Como apenas alguns SGBDs têm identificadores únicos para registros de tabelas, optamos por permitir a edição e remoção de dados apenas para tabelas que possuem chaves primárias. Tabelas que não o possuem podem receber apenas novas registros.

Para acessar a interface de edição de registro, clique com o botão direito do mouse no nó da tabela e selecione *Data Actions > Edit Data*:



A interface possui um editor SQL onde você pode filtrar e ordenar registros. Para evitar que a interface solicite muitos registros, há um campo que limita o número de registros a serem exibidos. A grade de registros tem as colunas nomes e tipos de dados. Colunas que pertencem à chave primária têm um ícone de uma chave ao lado de seus nomes.

A linha da grade que tem o símbolo * é a linha para adicionar novos registros. Vamos inserir alguns registros na tabela **Customer**:

The screenshot shows a database management interface. At the top, there's a tab labeled 'Query' and another labeled 'public.customer'. Below this, a SQL query is entered: `select * from public.customer t`. Below the query editor, there's a button to execute the query. Below that, a status bar shows 'Query 10 rows', 'Number of records: 0', and 'Response time: 0.024 seconds'. To the right of the status bar is a 'Save Changes' button. Below the status bar is a table with 5 columns: a line number, a checkbox, a primary key column 'cust_id (integer)', a 'cust_name (character varying)' column, and a 'cust_age (integer)' column. The table contains 9 rows of data, each with a red 'X' in the checkbox column. The 10th row is a new record line with a '+' in the checkbox column. The 'cust_age' value '21' in the 9th row is highlighted with a blue border.

		cust_id (integer)	cust_name (character varying)	cust_age (integer)
1	<input checked="" type="checkbox"/>	0	Pedro	22
2	<input checked="" type="checkbox"/>	1	Ryan	18
3	<input checked="" type="checkbox"/>	2	William	23
4	<input checked="" type="checkbox"/>	3	Susan	31
5	<input checked="" type="checkbox"/>	4	Nicole	19
6	<input checked="" type="checkbox"/>	5	Ricardo	45
7	<input checked="" type="checkbox"/>	6	Ademar	60
8	<input checked="" type="checkbox"/>	7	Felipe	29
9	<input checked="" type="checkbox"/>	8	Rafael	21
10	<input type="checkbox"/>	+		

Após salvar, os registros serão inseridos e podem ser editados (somente porque esta tabela possui uma chave primária). Vamos alterar o `cust_name` de alguns dos registros existentes:

Query
public.customer


```
select * from public.customer t
1 order by t.cust_id
```

▶
Query 10 rows
Save time: 0.141 seconds
Save Changes

		cust_id (integer)	cust_name (character varying)	cust_age (integer)
1	✖	0	Pedro	22
2	✖	1	Ryan	18
3	✖	2	William Changed	23
4	✖	3	Susan	31
5	✖	4	Nicole	19
6	✖	5	Ricardo	45
7	✖	6	Ademar Changed	60
8	✖	7	Felipe	29
9	✖	8	Rafael	21
10	+			

As tabelas podem ter campos com valores representados por strings muito longas. Para editar estes campos, o OmniDB possui uma interface que pode ser acessada clicando na célula específica:


Query ✕

 public.customer ✕

+


select * from public.customer t


1 order by t.cust_id

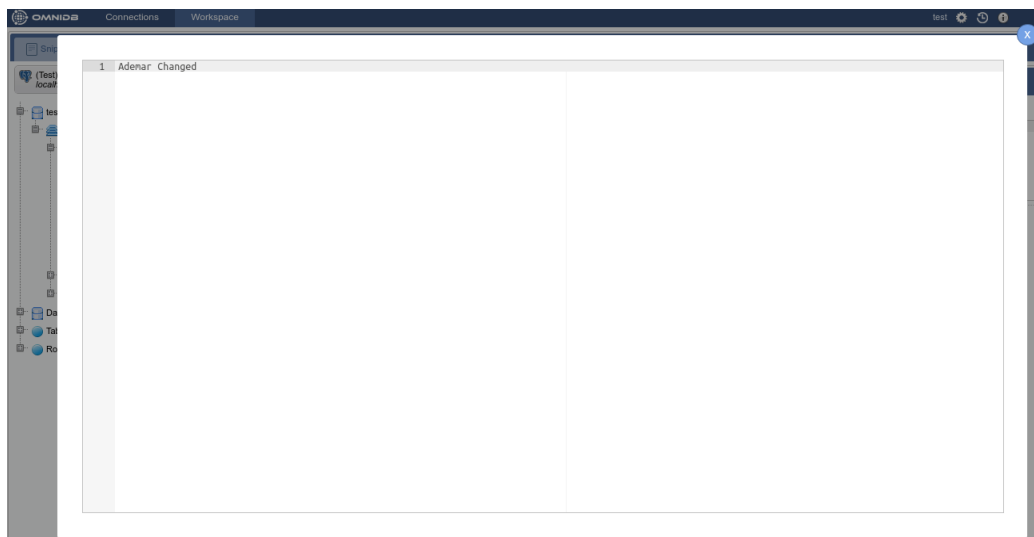


Query 10 rows ▾

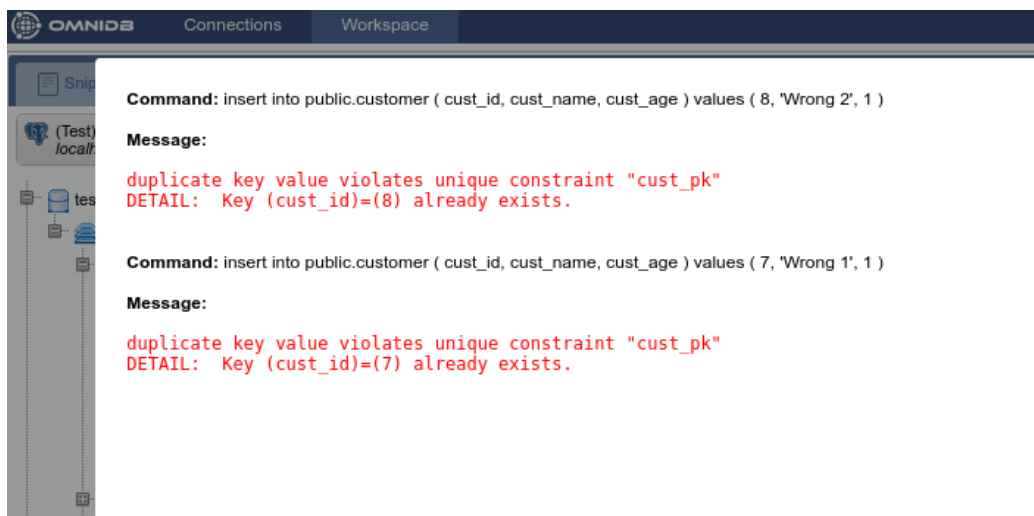
Save time: 0.031 seconds

		 cust_id (integer)	cust_name (character varying)	cust_age (integer)
1	✕	0	Pedro	22
2	✕	1	Ryan	18
3	✕	2	William Changed	23
4	✕	3	Susan	31
5	✕	4	Nicole	19
6	✕	5	Ricardo	45
7	✕	6	Ademar Changed	
8	✕	7	Felipe	28
9	✕	8	Rafael	21
10	+			

 Edit Content



A interface detecta erros que podem ocorrer durante as operações relacionadas a registros. Para demonstrar, insira dois registros com `cust_id` (chave primária) existente:



Mostra quais comandos tentaram ser executados e os respectivos erros. Para completar este capítulo, vamos adicionar alguns registros à tabela *Address*:

Query
public.customer
public.address
+

```
select * from public.address t
```

1
order by t.add_id

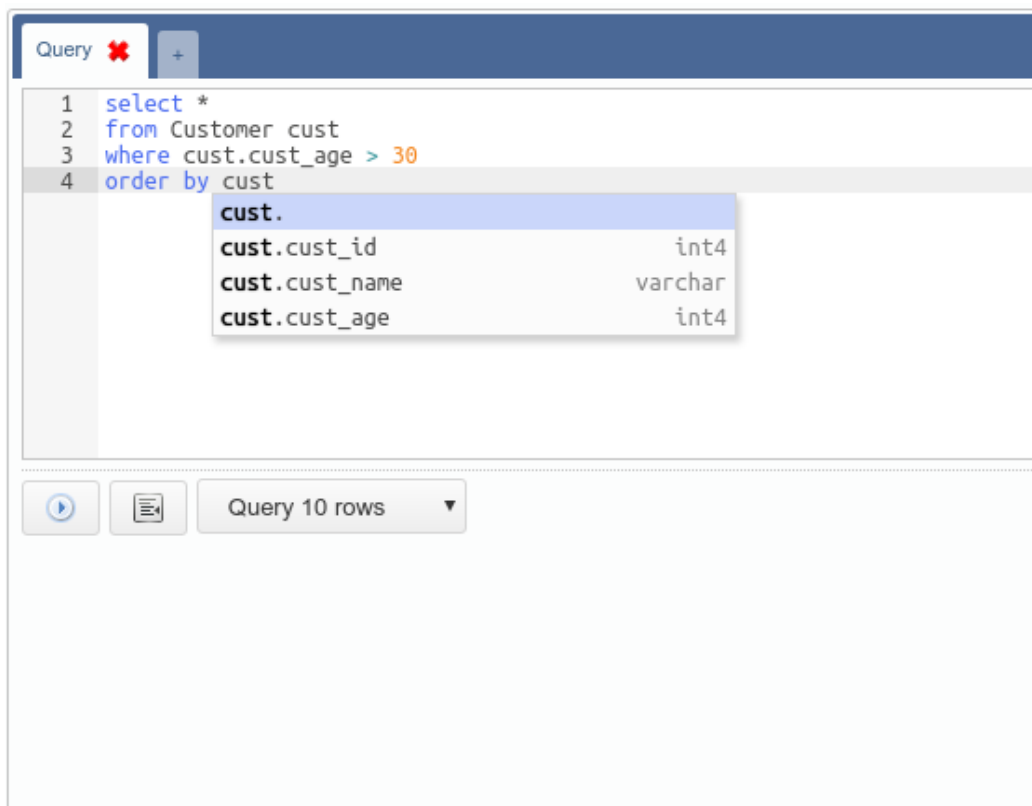
▶
Query 10 rows
Number of records: 0
Response time: 0.019 seconds
Save Changes

		add_id (integer)	add_street (character varying)	add_number (integer)	cust_id (integer)
1	✖	0	Blue Street	114	0
2	✖	1	Red Street	471	1
3	✖	2	Black Street	355	2
4	✖	3	White Street	1002	3
5	✖	4	Green Street	1056	4
6	✖	5	Purple Street	19	5
7	✖	6	Orange Street	47	6
8	✖	7	Yellow Street	33	7
9	✖	8	Brown Street	29	8
10	+				

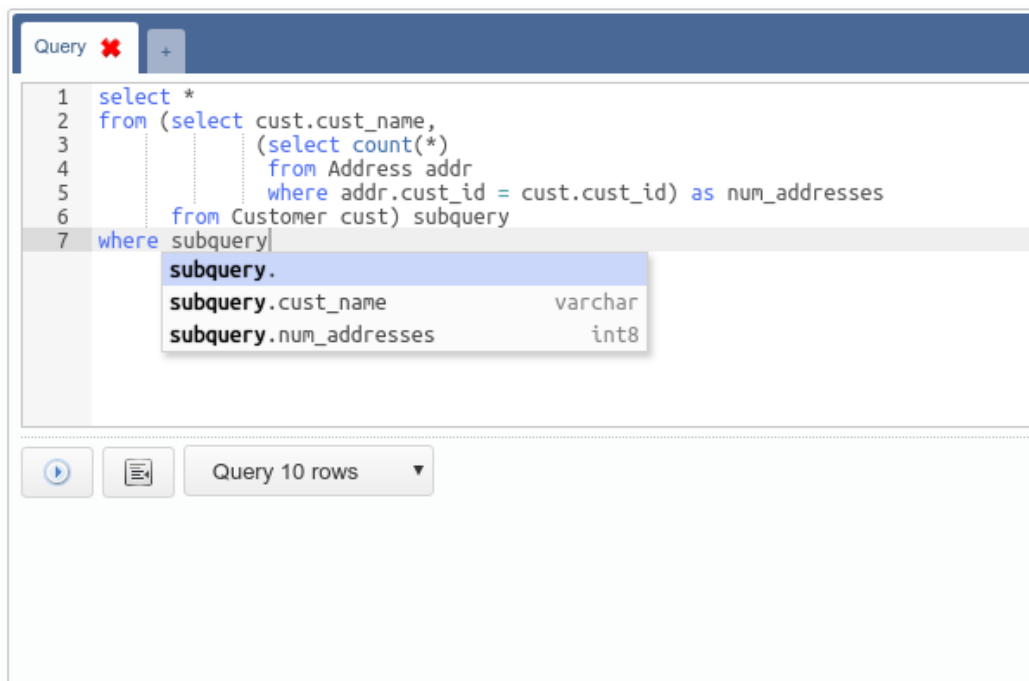
8 Editor de SQL

A ferramenta vem com um sistema de tabulação onde cada guia contém um editor SQL, um botão de ação, um botão de recuo, um campo para selecionar o tipo de comando e um espaço para exibir o resultado.

O editor SQL possui um recurso que ajuda muito ao criar novas consultas: *SQL code completion*. Com esta funcionalidade, é possível preencher automaticamente colunas contidas em uma tabela referenciada por um Alias (apelido). Para abrir a interface de preenchimento automático você só precisa digitar o Alias (apelido) e, em seguida, o caractere de ponto (.):



Além de preencher automaticamente as colunas da tabela, o editor também procura colunas contidas em subquery (sub-consultas):



O campo para selecionar o tipo de comando possui as seguintes opções: -
Script: execução de script, que é uma sequência de comandos separados por ponto e vírgula (;):

Query
+

```

1 insert into Customer (cust_id, cust_name, cust_age) values (9, 'John', 24);
2 insert into Customer (cust_id, cust_name, cust_age) values (4, 'Harry', 21);
3 insert into Customer (cust_id, cust_name, cust_age) values (9, 'Marjorie', 36);

```

Script

Start time: 08/07/2017 12:30:45
Duration: 0.03 seconds

Successful commands: 1
Errors: 2

Errors details:

Command: insert into Customer (cust_id, cust_name, cust_age) values (4, 'Harry', 21)
Message:
duplicate key value violates unique constraint "cust_pk"
DETAIL: Key (cust_id)=(4) already exists.

Command: insert into Customer (cust_id, cust_name, cust_age) values (9, 'Marjorie', 36)
Message:
current transaction is aborted, commands ignored until end of transaction block

O retorno mostra o tempo de resposta, o número de comandos que foram executados com sucesso, o número de comandos que geraram erros e um lista mostrando cada erro. - **Execute**: execução de um único comando. O retorno mostra o tempo de resposta ou um erro. - **Query (10, 100, 1000, all) rows**: execução de uma consulta que retorna um conjunto de registros, que são exibidos em uma grade. Assim como na interface de edição de registro onde cada célula pode ser visualizada separadamente clicando com o botão direito do mouse:

The screenshot shows a SQL query editor with a query window at the top and a results window at the bottom. The query window contains the following SQL code:

```
1 select *
2 from (select cust.cust_name,
3          (select count(*)
4           from Address addr
5           where addr.cust_id = cust.cust_id) as num_addresses
6       from Customer cust) subquery
```

The results window shows the execution of the query. It includes a toolbar with a play button, a refresh button, and a dropdown menu set to "Query 10 rows". To the right of the toolbar, it displays "Number of records: 9", "Start time: 08/07/2017 12:35:16", and "Duration: 0.022 seconds". Below the toolbar is a table with the following data:

	cust_name	num_addresses
1	Pedro	1
2	Ryan	1
3	Susan	1
4	Nicole	1
5	Ricardo	1
6	Felipe	1
7	Rafael	1
8	William Changed	1
9	Ademar Changed	1

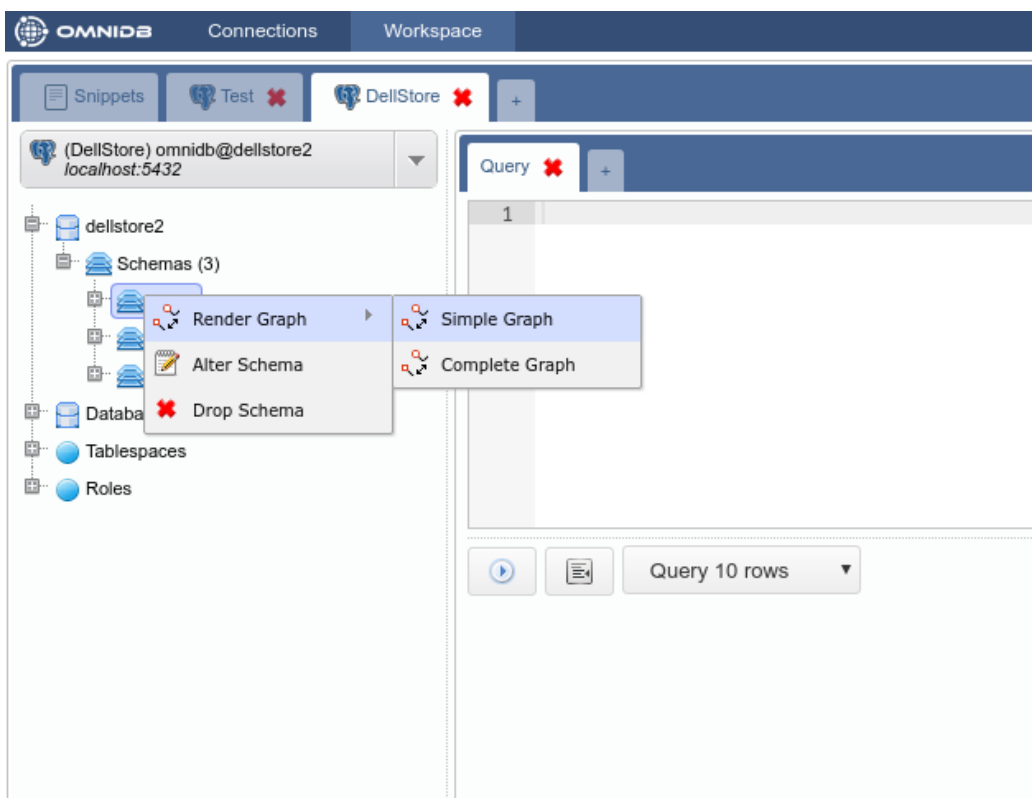
9 Gráfico com Tabelas e Relações

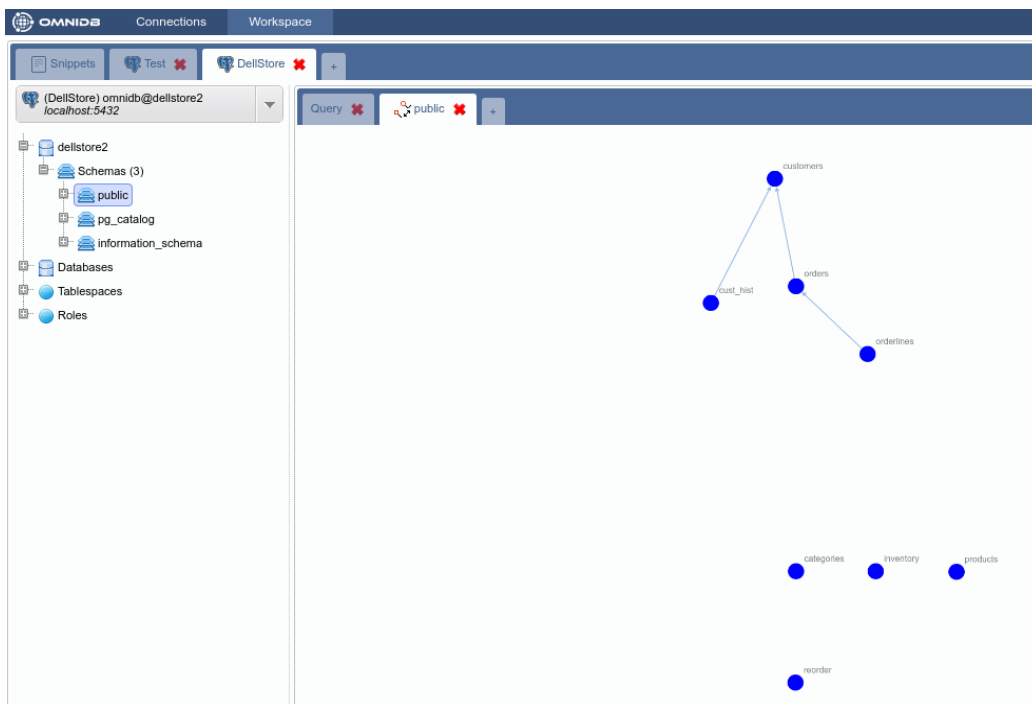
Este recurso exibe um gráfico com nós que representam tabelas e arestas representando os relacionamentos de tabelas com chaves estrangeiras. Usando o mouse, o usuário é capaz de aumentar o zoom, diminuir o zoom e arrastar e soltar nós para mudar sua posição.

Existem dois tipos de gráficos: *Simple Graph* (Gráfico simples) e *Complete Graph* (Gráfico completo).

9.1 Gráfico simples

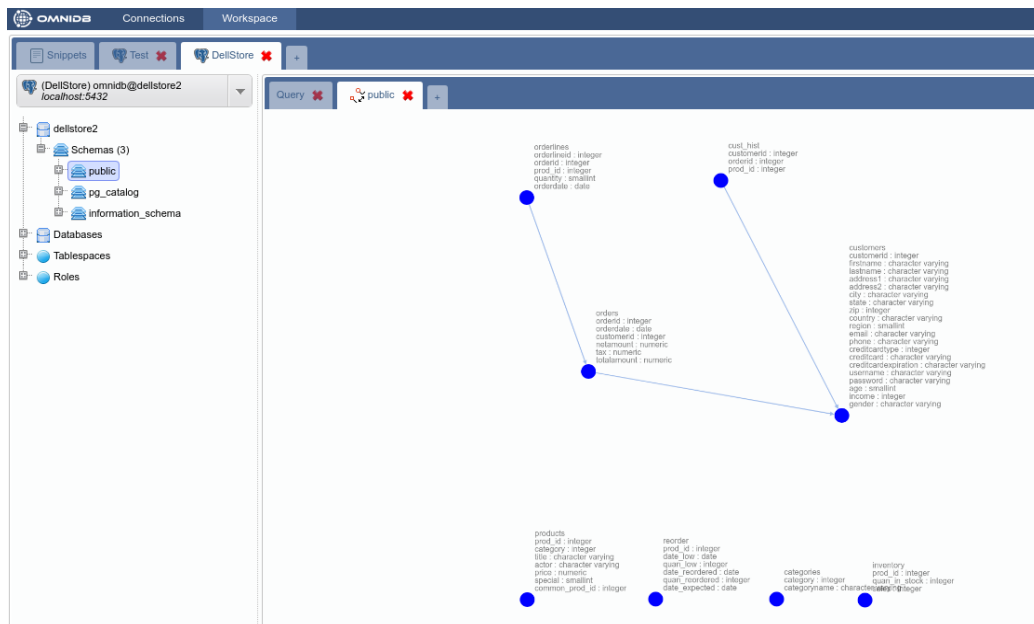
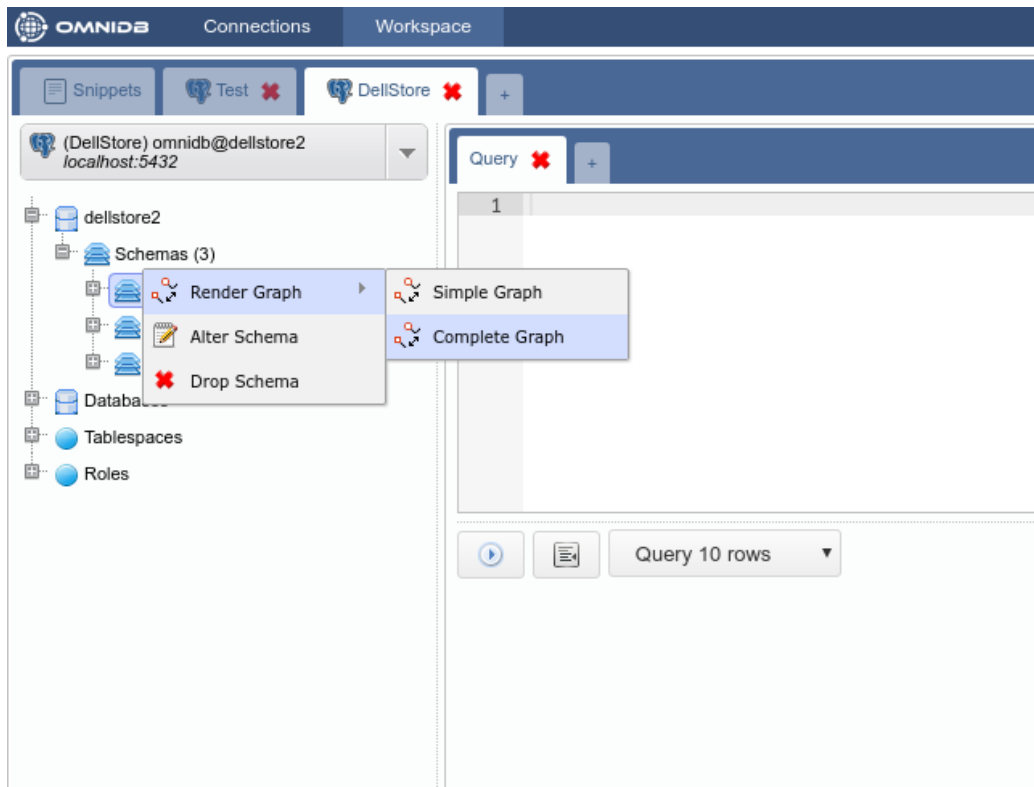
Para acessá-lo, clique com o botão direito do mouse no nó da raiz da árvore e selecione *Render Graph > Simple Graph*:





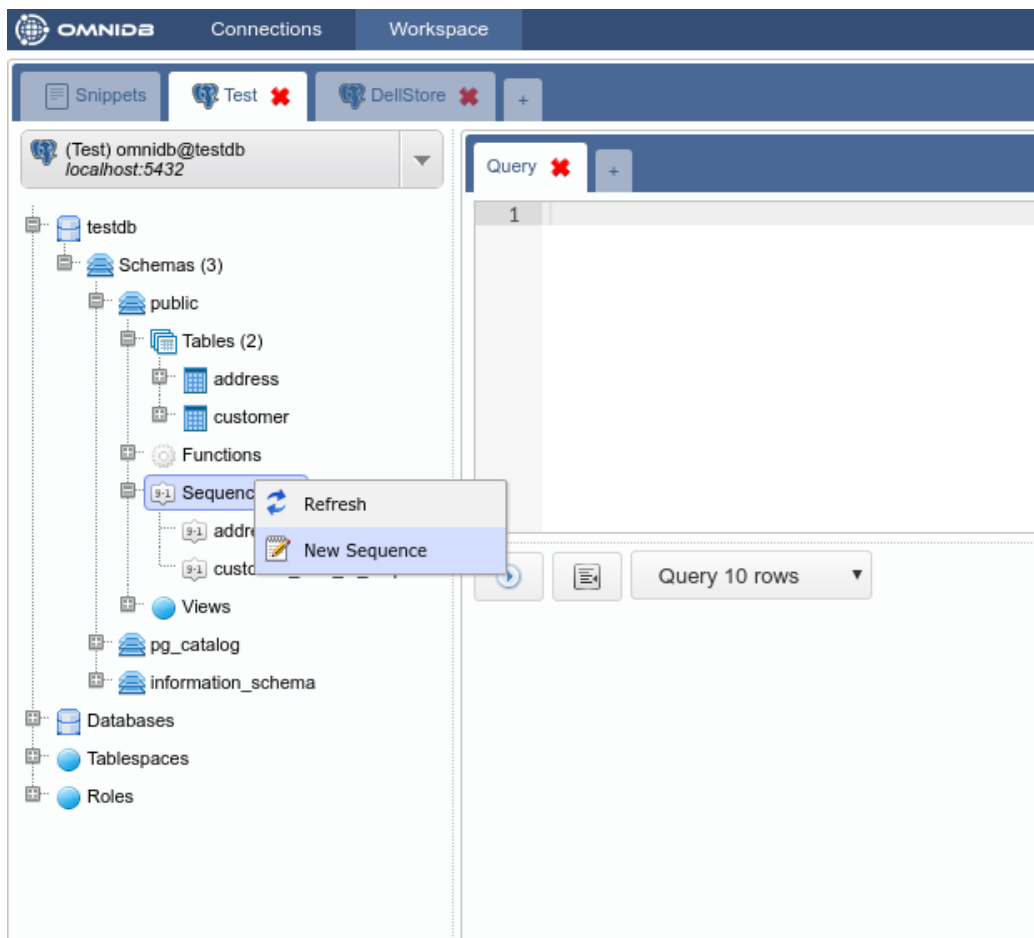
9.2 Gráfico Completo

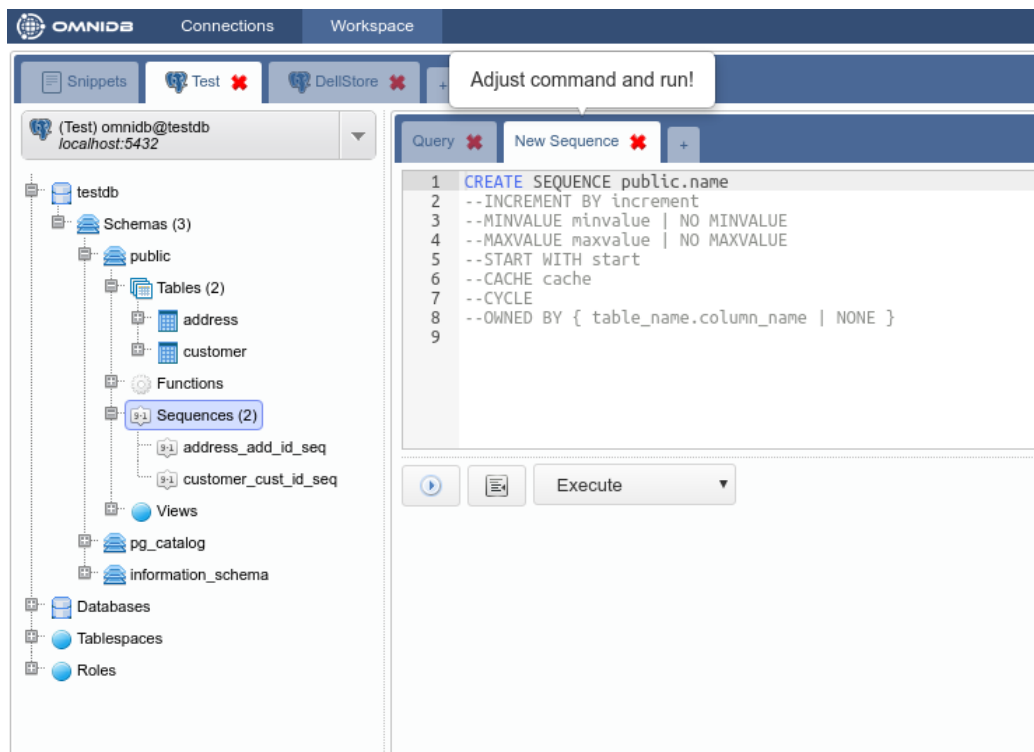
Este gráfico exibe tabelas com todas as suas colunas e respectivos tipos de dados. Além disso, as arestas agora são rotuladas com informações sobre a chave estrangeira específica. Para acessá-lo, clique com o botão direito do mouse no nó da raiz da árvore e selecione *Render Graph > Complete Graph*:



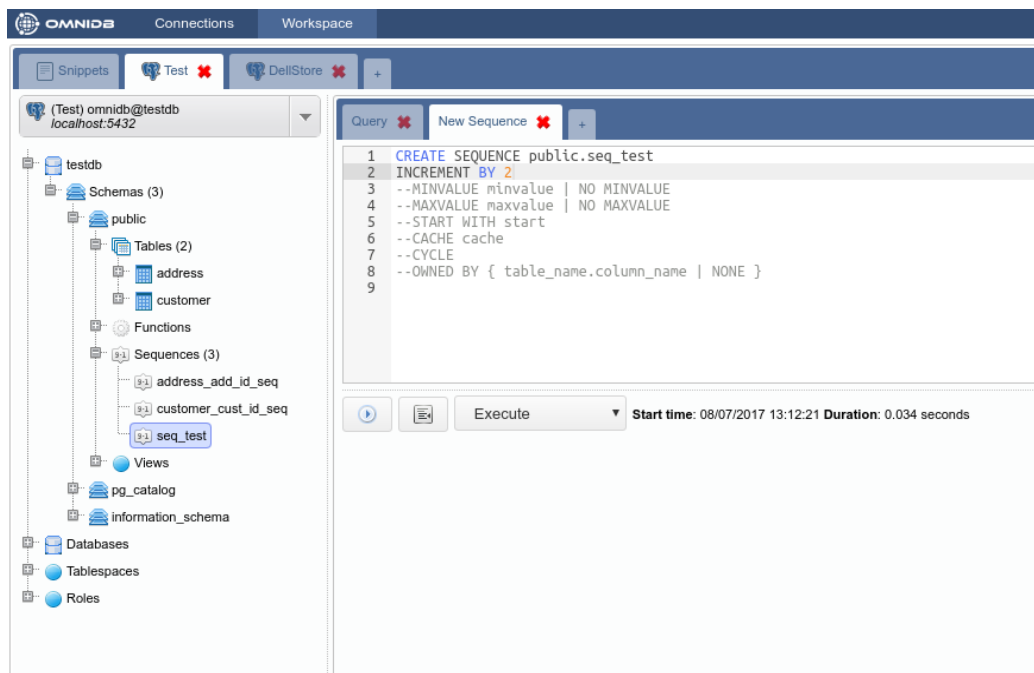
10 Modelos SQL

Com exceção das tabelas, todas as estruturas do PostgreSQL podem ser gerenciadas com o uso de *modelos SQL*. Isso dá ao usuário mais poder do que usando formas gráficas para manipular estruturas. Por exemplo, vamos considerar as seqüências dentro do schema **public** do banco de dados **testdb**. Para criar uma nova seqüência, clique com o botão direito do mouse no nó *Sequences* e escolha *New Sequence*.





Depois de alterar o nome da sequência, você pode descomentar outras opções de comando e configura-las de acordo com suas necessidades. Quando todos os comandos estiverem corretos, você pode clicar no botão *Execute* e uma nova sequência será criada:



Com o clique direito em uma sequência existente, você pode alterá-la ou soltar. Será feito da mesma maneira que a criação, usando um modelo SQL para o usuário efetuar a mudança.

OMNIDB

Connections

Workspace

Snippets

Test ✖

DellStore ✖

+

(Test) omnidb@testdb
localhost:5432

testdb

Schemas (3)

public

Tables (2)

address

customer

Functions

Sequences (3)

address_add_id_seq

customer_cust_id_seq

seq

Views

pg_catalog

information_schema

Databases

Tablespaces

Roles

Query ✖

New

1 CREATE S

2 INCREMEN

3 -- MINVAL

4 -- MAXVAL

5 -- START

6 -- CACHE

7 -- CYCLE

8 -- OWNED

9

▶

⌵

Alter Sequence

Drop Sequence ✖

Query ✖ New Sequence ✖ Alter Sequence ✖ +

```

1 ALTER SEQUENCE public.seq_test
2 INCREMENT BY 1
3 --MINVALUE minvalue | NO MINVALUE
4 --MAXVALUE maxvalue | NO MAXVALUE
5 --START WITH start
6 --RESTART
7 --RESTART WITH restart
8 --CACHE cache
9 --CYCLE
10 --NO CYCLE
11 --OWNED BY { table_name.column_name | NONE }
12 --OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
13 --RENAME TO new_name
14 --SET SCHEMA new_schema
15

```

▶

📄

Execute ▼

Start time: 08/07/2017 13:15:24 Duration: 0.019 seconds

OMNIDB Connections Workspace

Snippets

Test ✖

DelStore ✖

+

(Test) omnidb@testdb

localhost:5432

testdb

Schemas (3)

public

Tables (2)

address

customer

Functions

Sequences (2)

address_add_id_seq

customer_cust_id_seq

Views

pg_catalog

information_schema

Databases

Query ✖ New Sequence ✖ Alter Sequence ✖ Drop Sequence ✖ +

```

1 DROP SEQUENCE public.seq_test
2 -- CASCADE
3

```

▶

📄

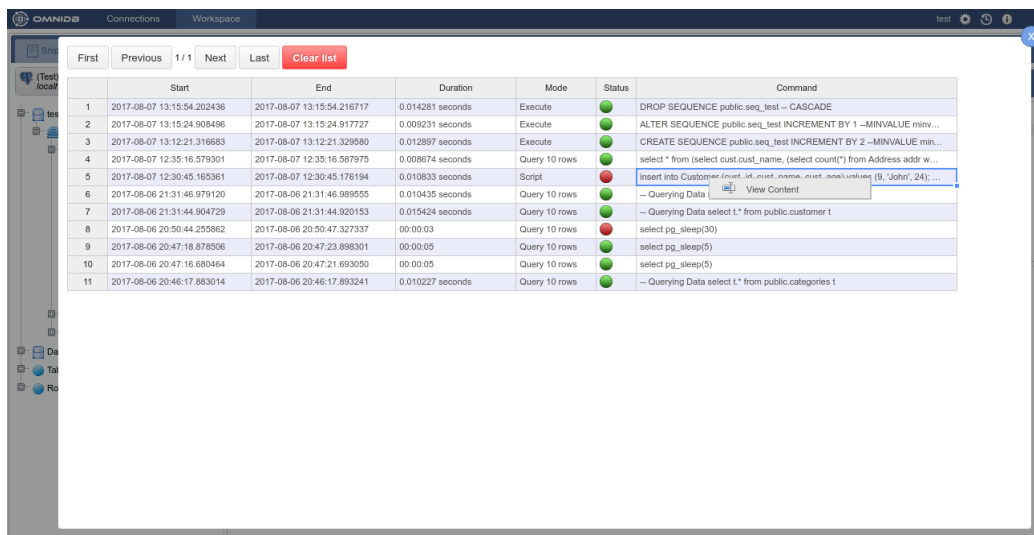
Execute ▼

Start time: 08/07/2017 13:15:54 Duration: 0.024 seconds

11 Funcionalidades Adicionais

11.1 Histórico SQL

Toda interação que o usuário faz com cada banco de dados é registrada no *Histórico SQL* do OmniDB. Para acessá-lo, você precisa clicar no ícone do relógio no canto superior direito. OmniDB mostrará um pop-up com todas as ações em um página de grade.



The screenshot shows the OmniDB SQL History window. It features a table with columns: Start, End, Duration, Mode, Status, and Command. The table lists 11 operations, including DROP SEQUENCE, ALTER SEQUENCE, CREATE SEQUENCE, and various SELECT and INSERT queries. A context menu is open over the 5th row, showing options like 'View Content'.

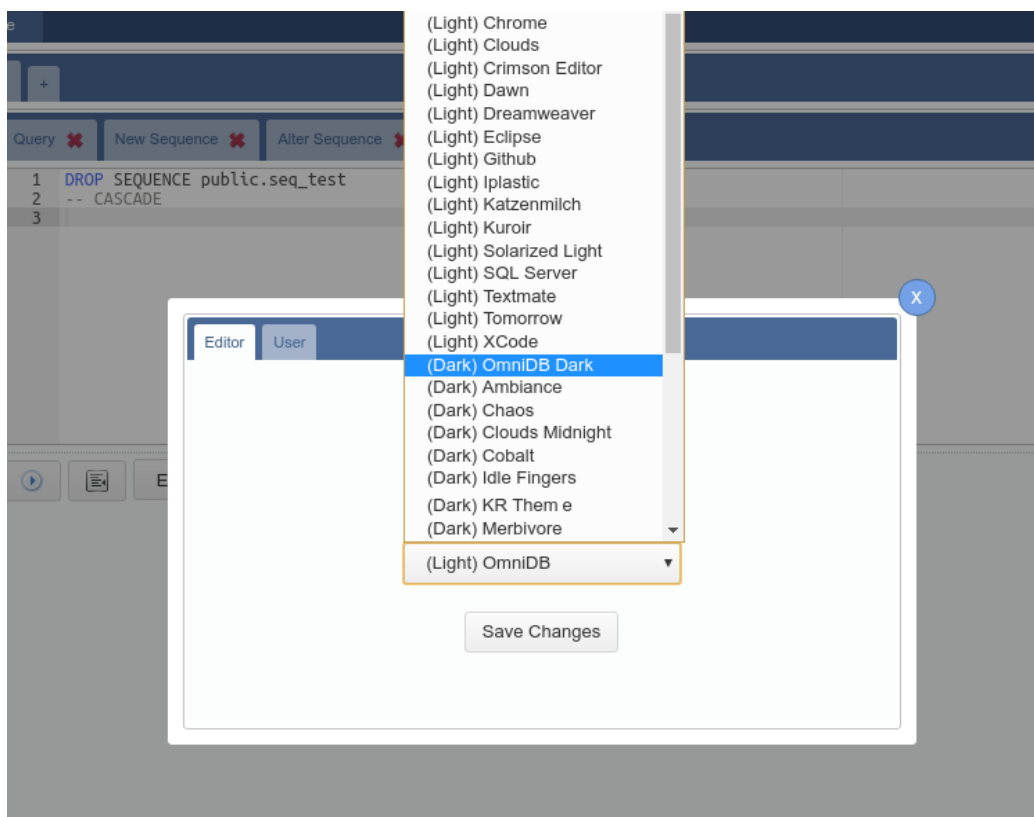
	Start	End	Duration	Mode	Status	Command
1	2017-08-07 13:15:54.202436	2017-08-07 13:15:54.216717	0.014281 seconds	Execute	●	DROP SEQUENCE public.seq_test -- CASCADE
2	2017-08-07 13:15:24.908496	2017-08-07 13:15:24.917727	0.009231 seconds	Execute	●	ALTER SEQUENCE public.seq_test INCREMENT BY 1 -- MINVALUE minv...
3	2017-08-07 13:12:21.316683	2017-08-07 13:12:21.329580	0.012897 seconds	Execute	●	CREATE SEQUENCE public.seq_test INCREMENT BY 2 -- MINVALUE minv...
4	2017-08-07 12:35:16.579301	2017-08-07 12:35:16.587975	0.008674 seconds	Query 10 rows	●	select * from (select cust_name, (select count(*) from Address addr w...
5	2017-08-07 12:30:45.165361	2017-08-07 12:30:45.176194	0.010833 seconds	Script	●	insert into Customer (cust_id, cust_name, cust_email, cust_phone) values (8, 'John', 24); ...
6	2017-08-06 21:31:46.979120	2017-08-06 21:31:46.989555	0.010435 seconds	Query 10 rows	●	-- Querying Data [View Content]
7	2017-08-06 21:31:44.904729	2017-08-06 21:31:44.920153	0.015424 seconds	Query 10 rows	●	-- Querying Data select t.* from public.customer t
8	2017-08-06 20:50:44.255862	2017-08-06 20:50:47.327337	00:00:03	Query 10 rows	●	select pg_sleep(30)
9	2017-08-06 20:47:18.878506	2017-08-06 20:47:23.898301	00:00:05	Query 10 rows	●	select pg_sleep(5)
10	2017-08-06 20:47:16.680464	2017-08-06 20:47:21.693050	00:00:05	Query 10 rows	●	select pg_sleep(5)
11	2017-08-06 20:46:17.883014	2017-08-06 20:46:17.893241	0.010227 seconds	Query 10 rows	●	-- Querying Data select t.* from public.categories t

Cada ação mostra a hora e a data em que começou, o tempo que terminou, a duração, o modo, o status e o comando. Como toda grade no OmniDB, você pode clicar com o botão direito no comando e clique em *View Content* (Exibir Conteúdo), onde outro pop-up irá abrir mostrando o conteúdo em um editor de texto maior.

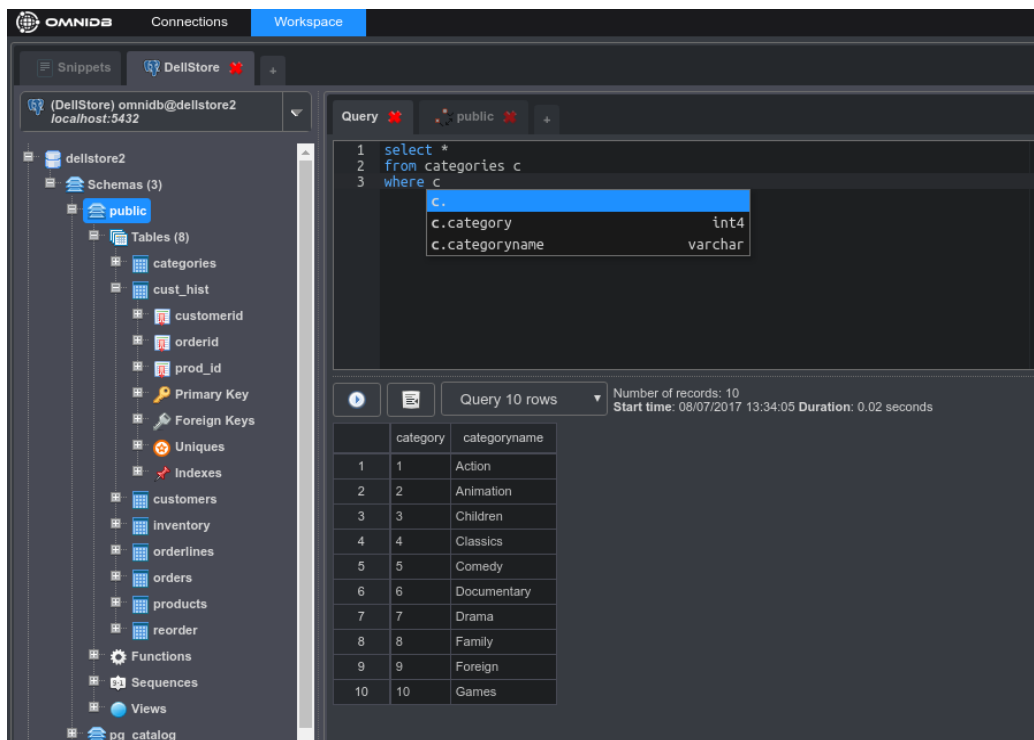
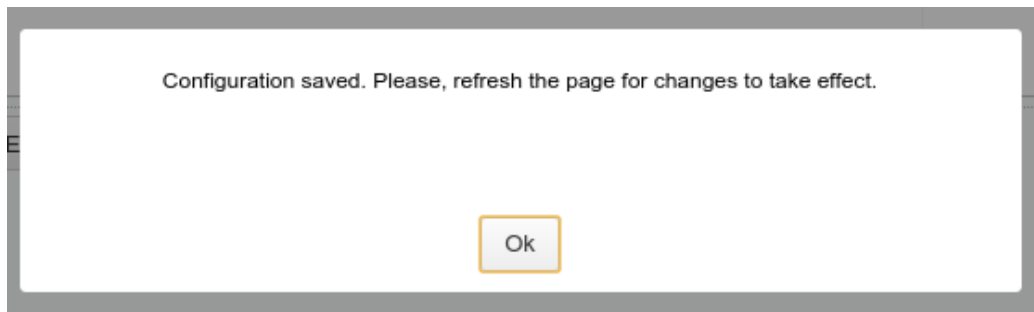
11.2 Configurações do usuário

Também no canto superior direito, clicando no ícone de engrenagem, o OmniDB irá abrir o pop-up *User Settings* (Configurações do usuário). É composto por duas abas: - **User (Usuário)**: permite que o usuário altere sua senha. Mais configurações do usuário serão adicionadas no futuro. - **Editor**: permite ao usuário alterar o tamanho da fonte do Editor SQL e também

altera todo o tema OmniDB. Há vários temas para OmniDB, cada um deles altera a cor de destaque de sintaxe do editor. Eles também são classificados em temas claros e escuros. Um tema leve é o padrão; um tema escuro mudará toda a interface do OmniDB.



Todas as alterações nas configurações do usuário exigem que você: - Atualize a página, se você estiver usando o OmniDB Server e a interface através de um navegador web; ou - Abra e feche o OmniDB, se você estiver usando o OmniDB-app.



12 Ferramenta de Configuração do OmniDB

Toda instalação do OmniDB também vem com um pequeno utilitário CLI chamado *OmniDB Config*. Ele terá um nome de arquivo diferente, dependendo da maneira como você instalou o OmniDB: - Se você estiver usando um pacote tarball ou zip, ele é chamado **omnidb-config**, para ambas versões: Server e app; - Se você usou um instalador (como o arquivo .deb) da versão Server, ele

é chamado **Omniadb-config-server**; - Se você usou um instalador da versão app, ele é chamado **omniadb-config-app**.

Apesar de ter nomes diferentes, o utilitário faz exatamente o mesmo. Se você usou um instalador, ele será colocado no seu `$PATH`.

```
user@machine:~$ omniadb-config-app --help
Usage: omniadb-config-app [options]
```

Options:

<code>--version</code>	show program's version number and exit
<code>-h, --help</code>	show this help message and exit
<code>-c username password, --createsuperuser=username password</code>	create super user: <code>-c username password</code>
<code>-a, --vacuum</code>	databases maintenance
<code>-r, --resetdatabase</code>	reset databases

12.1 Criar Super Usuário

Opção `-c` permite que você crie um novo super usuário, sem precisar abrir Interface OmniDB.

```
user @ machine: ~ $ omniadb-config-app -c william password
Criando o superusuário ...
Superuser criado.
```

12.2 Vacuum

O OmniDB possui dois bancos de dados: - `omniadb.db`: armazena todos os usuários e conexões, e outros relacionados ao OmniDB - *Session database*: armazena sessões de usuário do Django.

Ambos os bancos de dados são SQLite, por isso pode ser útil otimizá-los às vezes para reduzir o tamanho do arquivo. Isso pode ser feito com a opção `-a`.

```
user @ machine: ~ $ omniadb-config-app -a
Vacuuming OmniDB database ...
Done.
```



```
Vacuuming Sessions database...  
Done.
```

12.3 Redefinir o Banco de Dados

Se deseja eliminar todas as informações do OmniDB e obter um banco de dados limpo, conforme foi instalado, você pode usar a opção `-r`. Use com cuidado!

```
user@machine:~$ omnidb-config-app -r  
*** ATENTION *** ALL USERS DATA WILL BE LOST  
Would you like to continue? (y/n) y  
Cleaning users...  
Done.  
Cleaning sessions...  
Vacuuming OmniDB database...  
Done.  
Vacuuming Sessions database...  
Done.
```