

## 1. Introdução

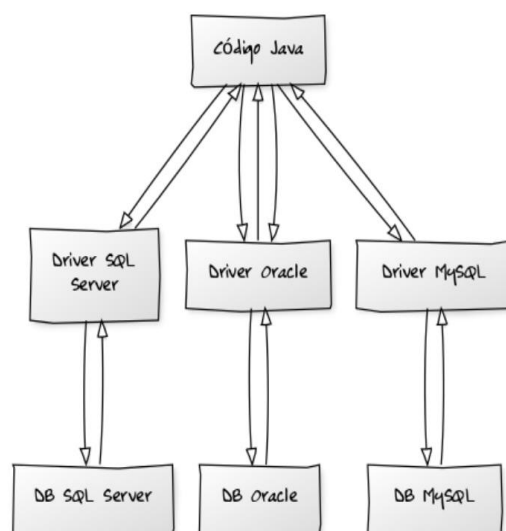
A biblioteca padrão do Java para persistência em banco de dados é conhecida como **JDBC**, de **Java Data Base Connectivity**.

O **JDBC** é, na verdade, um conjunto de interfaces bem elaborado que deve ser implementado de forma diferente para cada banco de dados. Dessa forma, evita-se que cada banco tenha seu próprio conjunto de classes e métodos.

Qual a vantagem disso? Manutenibilidade é uma das muitas. Migrar de um banco para outro é um processo fácil, já que todos implementam as mesmas interfaces, portanto possuem métodos com a mesma assinatura.

Esse conjunto de classes é conhecido como **driver**; elas fazem a ponte entre a **API de JDBC** e o **banco de dados**.

Os drivers que implementam as interfaces do **JDBC** possibilitam a comunicação entre um código Java e os diferentes bancos de dados existentes.



A plataforma Java é constituída de três componentes: A linguagem de programação Java, a biblioteca de classes e interfaces e a JVM – Java Virtual Machine.

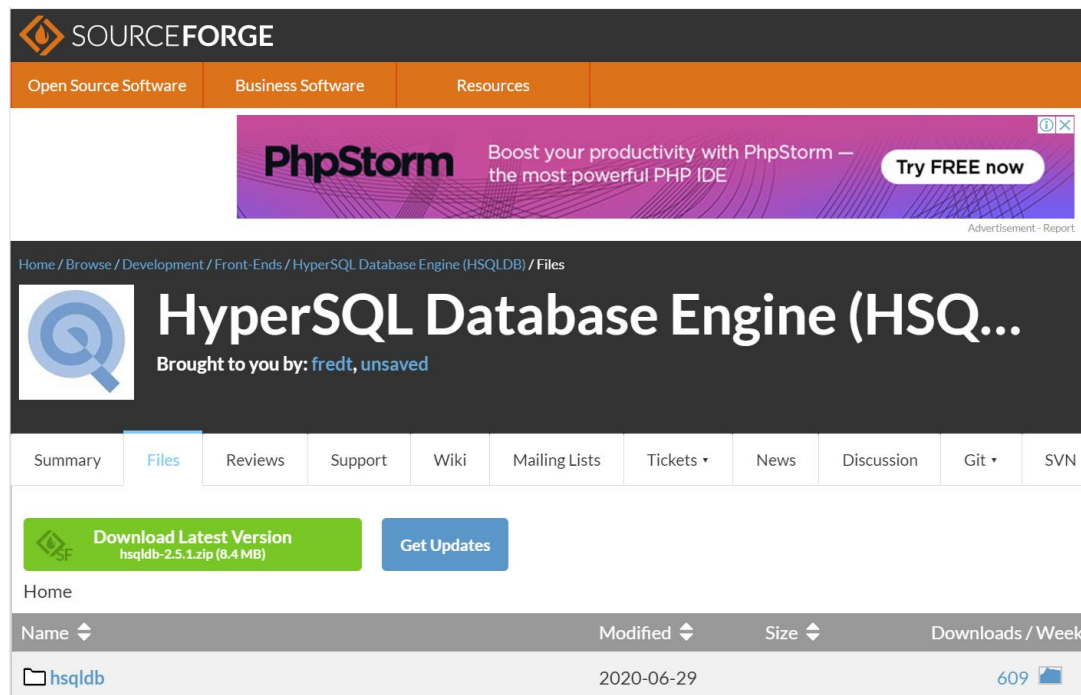
Neste exercício, assume-se que a plataforma **Java** está devidamente instalada e que o servidor de Banco de Dados MySQL está operacional. Da mesma forma, assume-se que a **IDE Eclipse** também esteja disponível para uso.

É muito comum que uma aplicação queira acessar um banco de dados para armazenar suas informações. Poderíamos escrever diretamente todo o código de conexão com um banco, por exemplo, **MySQL**. Mas se desejássemos mudar dele para um **PostgreSQL**, teríamos que reimplementar todo o protocolo de comunicação do servidor. Cada banco possui seu próprio protocolo, através do qual enviamos e recebemos requisições **SQL**, mas não queremos implementar algo tão baixo nível quanto o protocolo do banco de dados.

Portanto precisamos de alguém que faça uma ponte entre nós e o banco de dados. Algum tipo de interface comum a todos os bancos relacionais.

Neste exercício utilizaremos o **HSQLDB**, um banco de dados que suporta SQL e foi totalmente implementado em Java. Criaremos um novo projeto Java no Eclipse chamado Atividade-1.

**HSQLDB** pode ser baixado de: <http://hsqldb.org/>.



The screenshot shows the SourceForge project page for HyperSQL Database Engine (HSQLDB). The page includes a navigation bar with 'Open Source Software', 'Business Software', and 'Resources'. A banner for 'PhpStorm' is visible. The project title 'HyperSQL Database Engine (HSQLDB)' is prominently displayed, along with the text 'Brought to you by: fredt, unsaved'. Below the title, there are tabs for 'Summary', 'Files', 'Reviews', 'Support', 'Wiki', 'Mailing Lists', 'Tickets', 'News', 'Discussion', 'Git', and 'SVN'. A green button labeled 'Download Latest Version' (hsqldb-2.5.1.zip (8.4 MB)) and a blue button labeled 'Get Updates' are present. At the bottom, a table lists the files available for download:

Name	Modified	Size	Downloads / Week
hsqldb	2020-06-29		609

**HyperSQL Database (HSQLDB)** é um Sistema Gerenciador de Banco de Dados Relacional, compatível com **Java 8** e escrito na linguagem de programação **Java**, sendo executado em uma máquina virtual **Java**. Suporta a interface **JDBC** para acesso aos bancos de dados.

O package **jar HSQLDB**, **hsqldb.jar**, está localizado no **diretório /lib** do arquivo ZIP baixado e contém diversos componentes e programas, dentre os quais, o **HyperSQL RDBMS Engine**, o **HyperSQL JDBC Driver** e o **Database Manager (GUI database access tool)**.

Após baixar o **hsqldb.jar**, deve-se copiá-lo para o projeto **Java** criado no Eclipse.

Agora precisamos levantar o banco e para isso executaremos o arquivo **bat**, na pasta **bin**,



**runserver.bat** runServer.bat

Posicione-se no diretório **lib** da pasta onde foi baixado o **HSQLDB**, e execute o procedimento **runServer.bat**.

Será exibida uma janela (**console**) informando que o Servidor **HSQLDB** está ativo na porta, por padrão, **9001**.

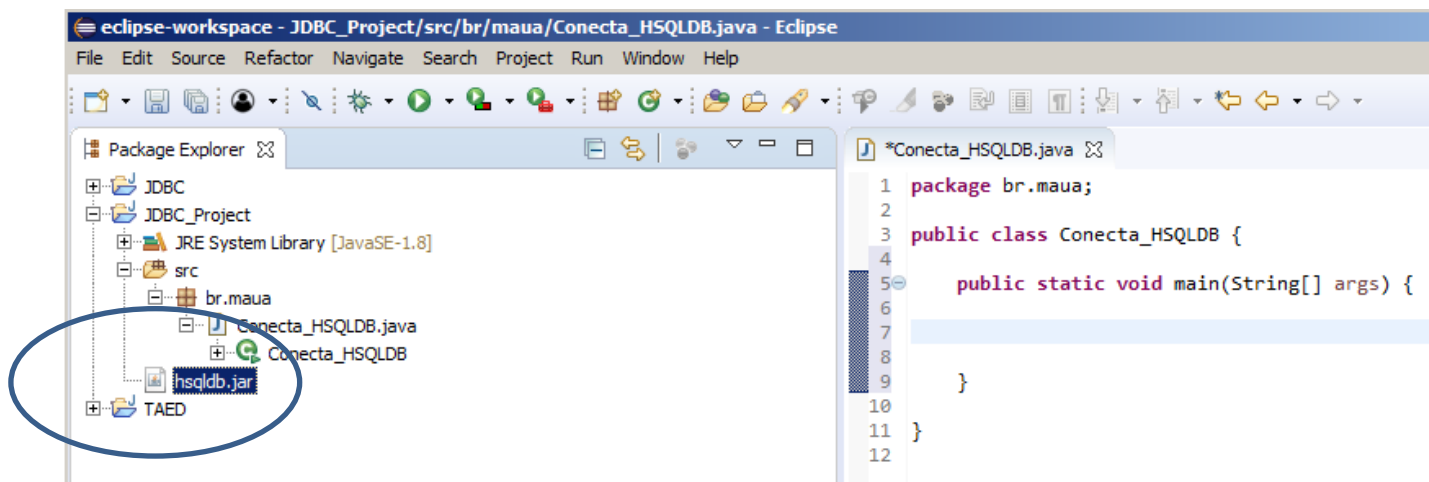
```
C:\Windows\system32\cmd.exe
C:\hsqldb_2.4.1\bin>cd ..\data
[Server@7adf9f5f]: Startup sequence initiated from main() method
[Server@7adf9f5f]: Could not load properties from file
[Server@7adf9f5f]: Using cli/default properties only
[Server@7adf9f5f]: Initiating startup sequence...
[Server@7adf9f5f]: Server socket opened successfully in 138 ms.
[Server@7adf9f5f]: Database [index=0, id=0, db=file:test, alias=] opened successfully in 1035 ms.
[Server@7adf9f5f]: Startup sequence completed in 1173 ms.
[Server@7adf9f5f]: 2018-06-12 04:59:13.124 HSQLDB server 2.4.1 is online on port 9001
[Server@7adf9f5f]: To close normally, connect and execute SHUTDOWN SQL
[Server@7adf9f5f]: From command line, use [Ctrl]+[C] to abort abruptly
```

Uma vez que o Servidor de Banco de Dados está ativo, vamos agora escrever um programa Java para fazer a conexão com o servidor de banco de dados **HSQLDB**.

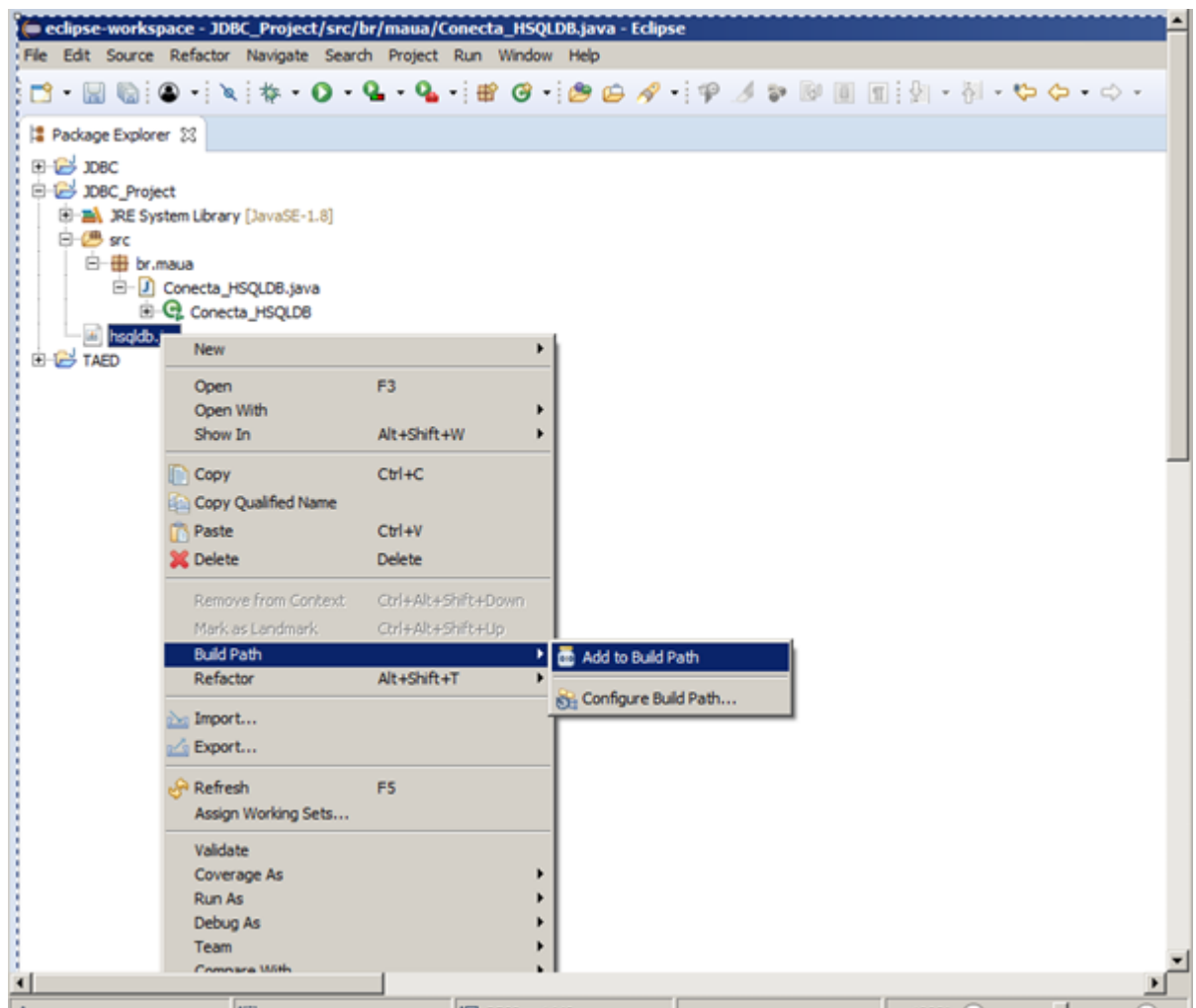
Abrir o Eclipse e criar um projeto Java. O nome do projeto Java será **JDBC\_Project**.

No projeto criar um **package** chamado **br.maua**.

Copiar para o projeto, a implementação do servidor **HSQLDB**, materializada no arquivo **hsqldb.jar**. (Esse arquivo está no diretório **lib** da instalação do servidor **HSQLDB**). Ctrl-C no arquivo .jar e Ctrl-V no projeto **JDBC\_Project** do Eclipse.



Definir no projeto a configuração correspondente ao **ClassPath**, para que o código a ser executado sob **Eclipse** encontre o driver **JDBC**, o qual também faz parte do **HSQLDB.jar**.



No projeto **JDBC\_Project** do Eclipse, botão direito do Mouse e selecionar **Build Path** e **Add Build Path**.

Neste **exercício** faremos uma configuração no Servidor de Banco de Dados **HSQLDB** para se criar um banco de dados. O nome do banco de dados será **db**.

Não há um banco de dados default no gerenciador de banco de dados **HSQLDB**. Assim, será necessário criar-se um banco de dados inicial para desenvolvermos uma aplicação.

No diretório de instalação do **HSQLDB**, há uma pasta chamada **DATA** e gravaremos o banco de dados **db** neste diretório.

Inicialmente configuraremos a variável **PATH** do ambiente. Clique com o botão direito do mouse sobre o ícone **Meu Computador**. Abrirá um menu flutuante, clique em **Propriedades**. Abrirá uma caixa de diálogo, escolha a **Guia Avançado**. Clique no botão Variáveis do Ambiente. Selecione a variável **PATH** e clique no botão Editar. Na caixa de texto Valor da Variável, clique no final do texto e acrescente: **C:\hsqldb\_2.4.1\lib\hsqldb.jar**

Clique no botão **OK** para fechar a edição da variável. Clique no botão **OK** para fechar a caixa de diálogo das variáveis do ambiente. Clique em **OK** para fechar a caixa de diálogo das Propriedades do Sistema.

Neste exercício, empregaremos o modo **HSQLDB** Servidor. Para executarmos o sistema gerenciador de banco de dados **HSQLDB** no modo Servidor, deve-se invocar o programa Server.

O programa recebe alguns argumentos para que se possa iniciar ou criar um novo banco de dados.

A configuração para a criação de um novo banco de dados será feita por meio da edição do arquivo **server.properties** no diretório **data**.

Crie um novo arquivo chamado **server.properties** no diretório **data** com as seguintes configurações:

```
server.port=59999
server.database.0=file:E:/hsqldb_2.5.1/data/db
server.dbname.0=db
```

Nesse arquivo, estamos por exemplo, definindo a porta **59999** e o banco de dados de nome **db**. O arquivo de configuração deve ser salvo sem a extensão **TXT**. Salvar o arquivo pelo nome **server.properties** no diretório **DATA**.

Executar o comando **cd \hsqldb\_2.5.1\bin** e execute:  
ar o procedimento **runserver.bat**.

Será exibida uma janela (console) informando que o Servidor **HSQLDB** está ativo na porta, **59999**.

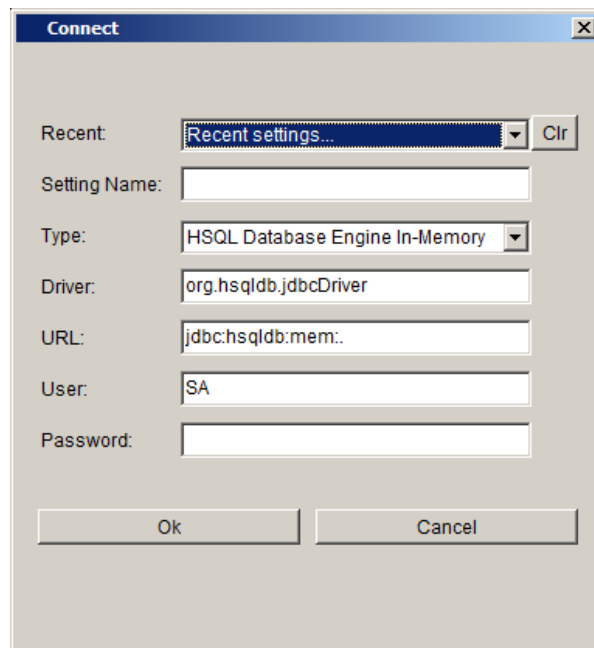
```
Command Prompt - runserver.bat
C:\hsqldb_2.4.1\bin>runserver.bat
C:\hsqldb_2.4.1\bin>cd ..\data
[Server@7adf9f5f]: Startup sequence initiated from main() method
[Server@7adf9f5f]: Loaded properties from [C:\hsqldb_2.4.1\data\server.properties]
[Server@7adf9f5f]: Initiating startup sequence...
[Server@7adf9f5f]: Server socket opened successfully in 16 ms.
[Server@7adf9f5f]: Database [index=0, id=0, db=file:c:/hsqldb_2.4.1/data/db, alias=] opened successfully in 562 ms.
[Server@7adf9f5f]: Startup sequence completed in 578 ms.
[Server@7adf9f5f]: 2018-06-12 13:17:05.013 HSQLDB server 2.4.1 is online on port 9099
[Server@7adf9f5f]: To close normally, connect and execute SHUTDOWN SQL
[Server@7adf9f5f]: From command line, use [Ctrl]+[C] to abort abruptly
```

Uma vez que o Servidor de Banco de Dados está ativo, vamos agora criar uma tabela chamada **Produtos**, no banco de dados **DB**.

Faremos a criação dessa tabela, abrindo um novo terminal e iniciando o ambiente visual (Cliente) que também está criado no hsqldb.jar, juntamente com o driver **JDBC**.

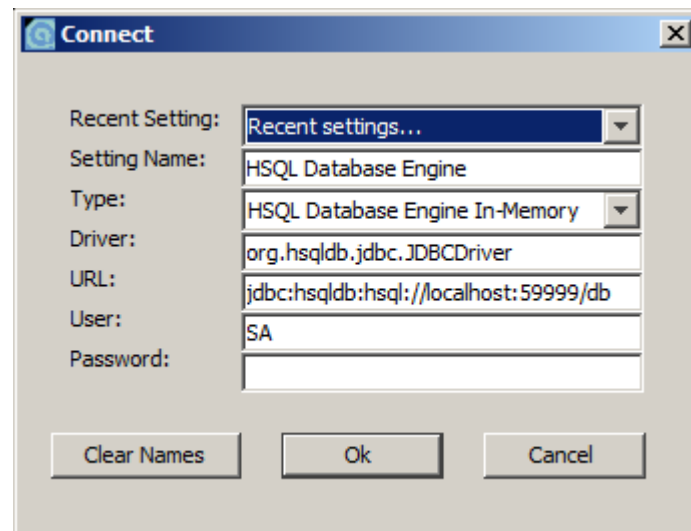
Nesse novo terminal, iniciar o cliente – no diretório **bin** -- por meio do procedimento **runManagerSwing.bat**

Será exibida a janela:

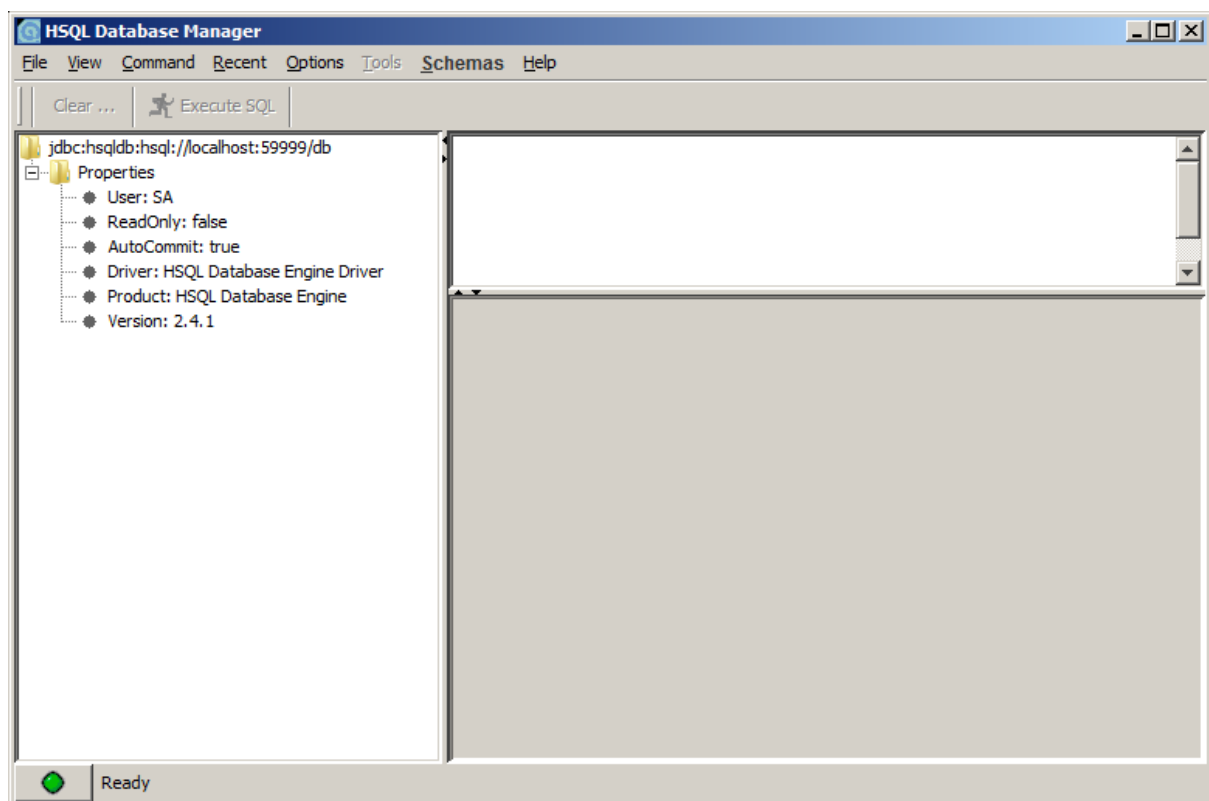


Considerando que o **HSQLDB** está sendo executado em Modo Servidor, escolhemos no campo **Type** a opção: **HSQL Database Engine Server**

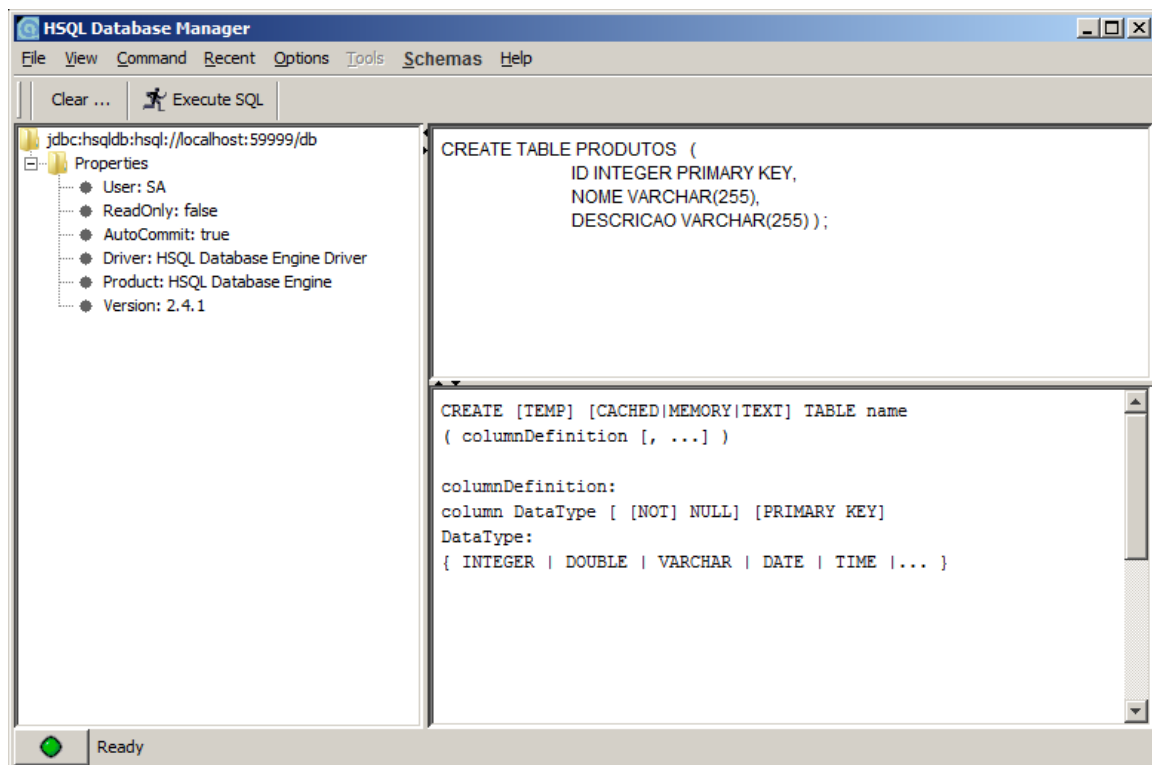
Além disso, deve-se complementar o campo URL com o nome do banco de dados que estaremos criando. Complementar com: **jdbc:hsqldb:hsq://localhost:59999/db**



Clique **OK**.

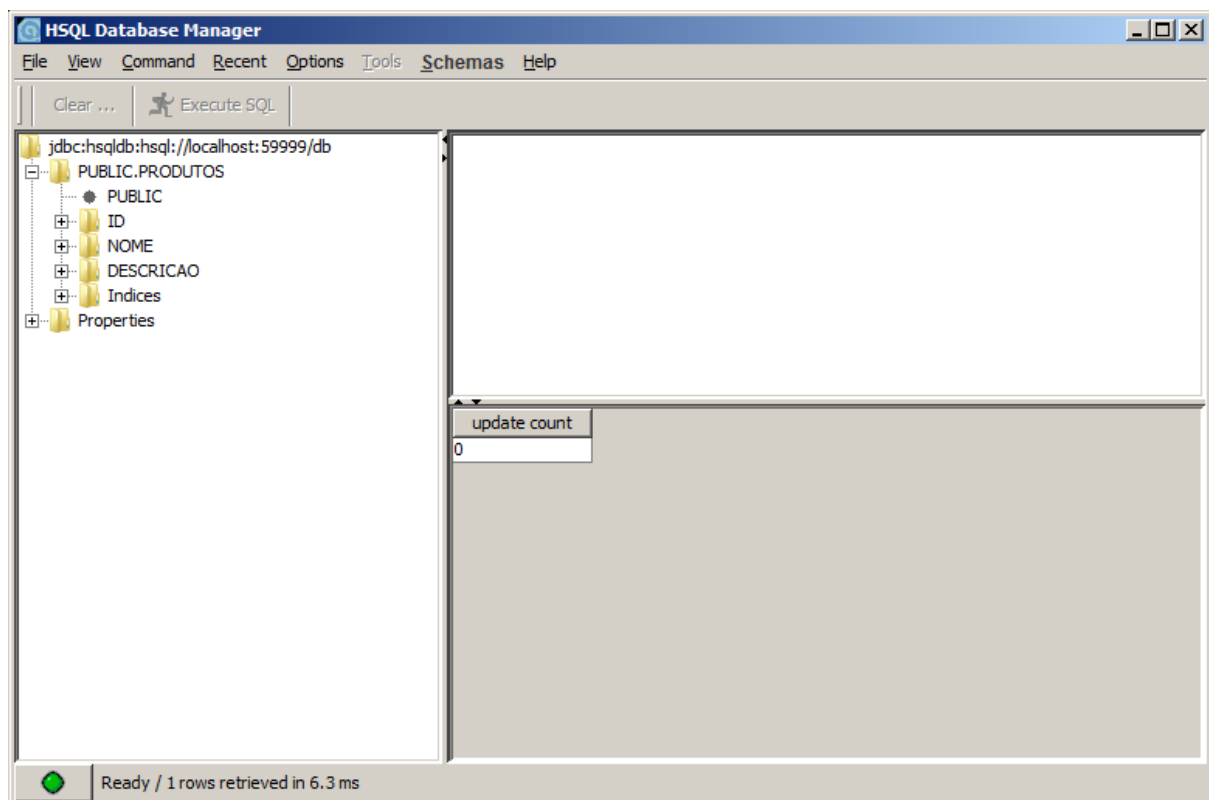


No cliente, vamos definir a tabela **produtos** com o comando:



A execução mostra **zero** atualizações.

No menu **View**, **Refresh**, pode-se agora ver a tabela **produtos**:

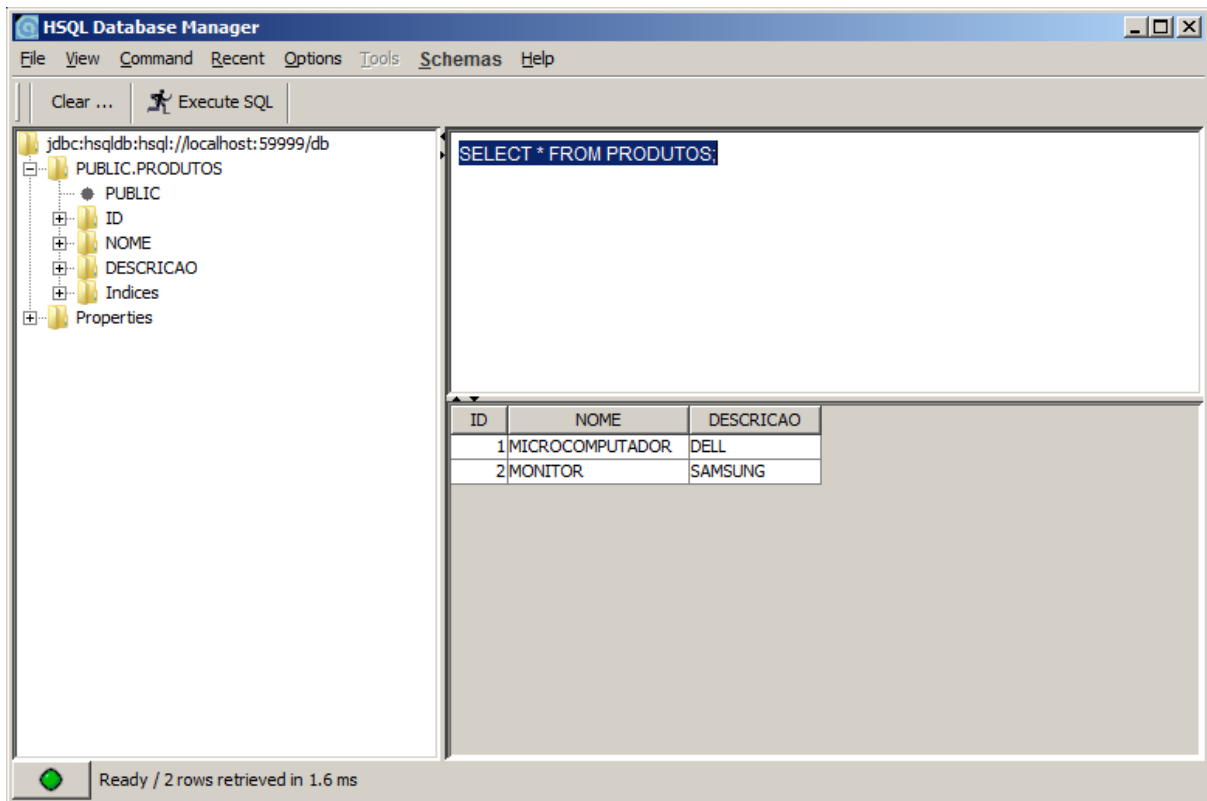


Vamos agora inserir dois novos produtos:

- **INSERT INTO PRODUTOS VALUES(1, 'MICROCOMPUTADOR', 'DELL');**



- **INSERT INTO PRODUTOS VALUES(2, 'MONITOR', 'SAMSUNG');**



Voltemos agora para o nosso código **Java**:

Abrir o Eclipse e no **package br.maua**, e vamos criar uma classe chamada **InsertProduto**, para inserir um terceiro produto com o comando SQL:

- **INSERT INTO PRODUTOS VALUES (3, 'CAMERA NIKON', 'D850');**

```

package br.maua;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class InsereProduto {

    public static void main(String[] args) throws SQLException {

        try {

            // Complemente o Código aqui

        }

        catch (SQLException e) {

            System.out.println("Erro SQLException....");

        }

        catch ( Exception e) {
            System.out.println("Problemas na conexao ao HSQLDB....");
        }

    }

}

```

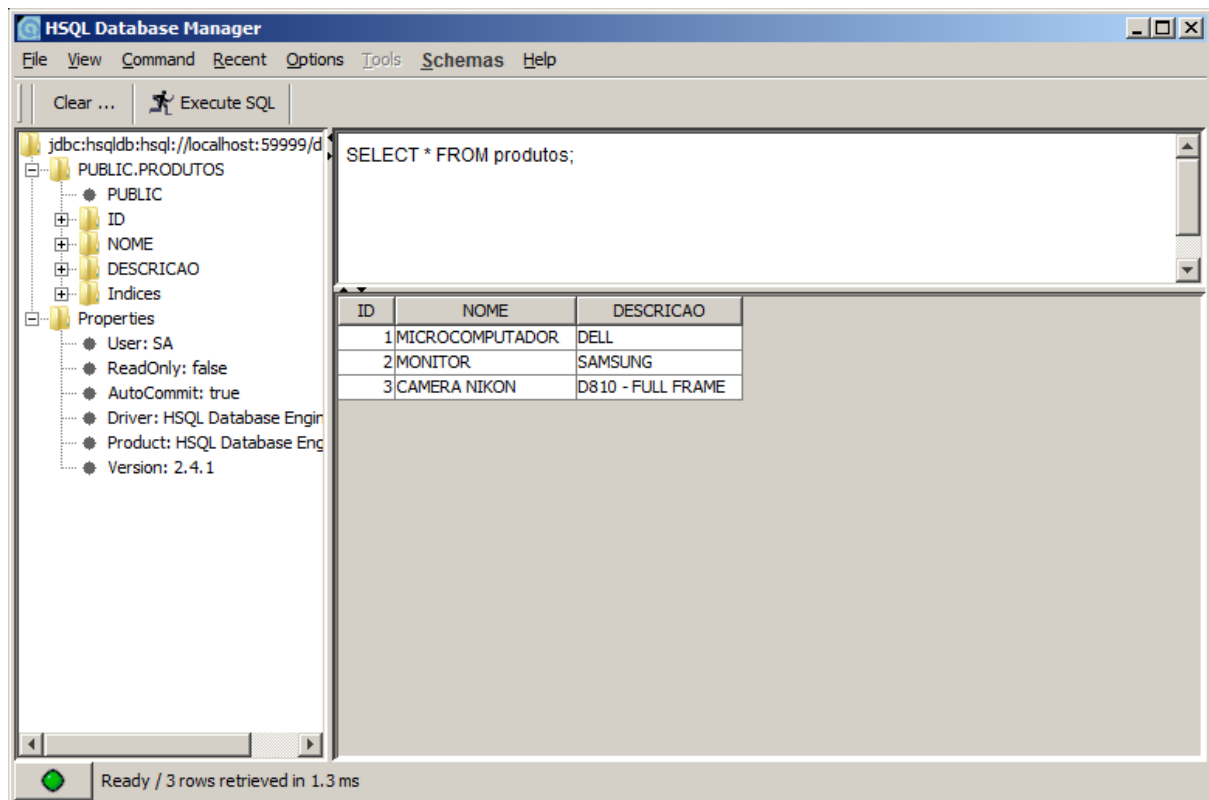
No **Eclipse**, após a execução do programa, deverá ser exibido na console:

```

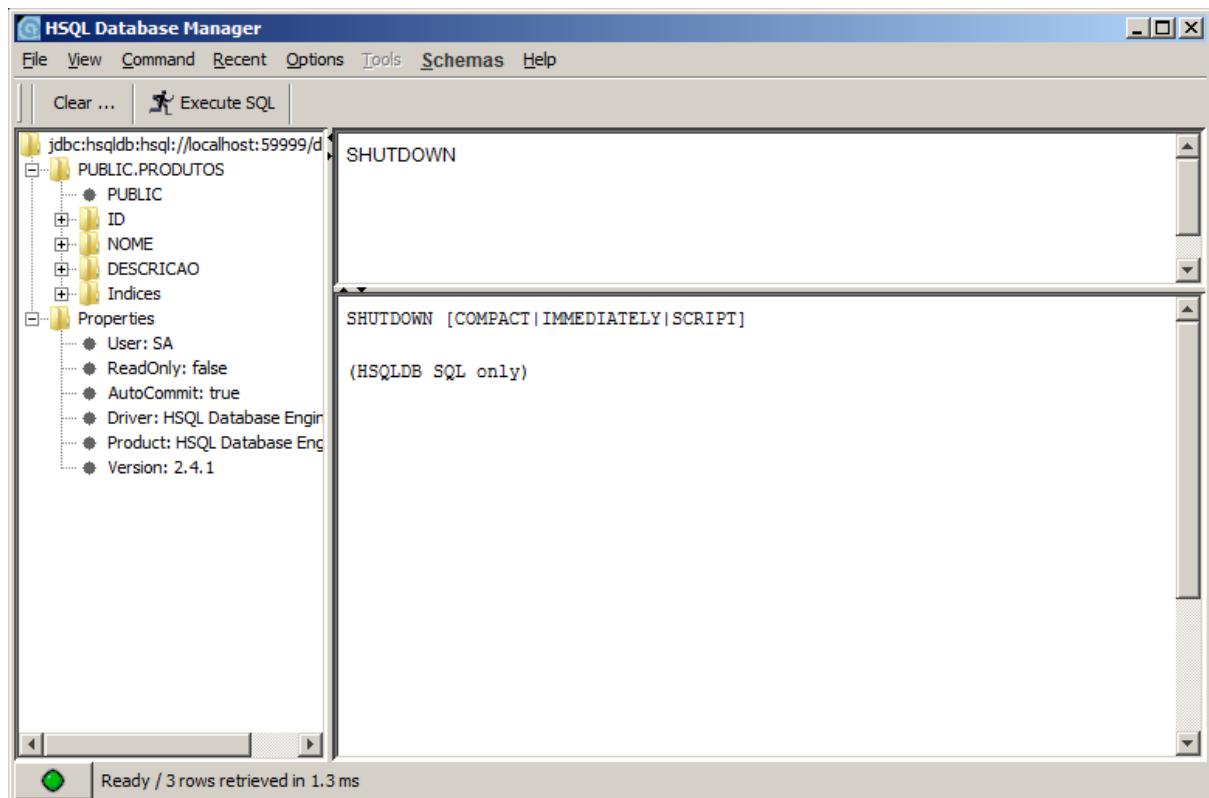
conexao ao HSQLDB feita com SUCESSO !
Insert executado com sucesso...

```

Vamos agora, ativar o cliente visual, para checar se o registro foi inserido no Banco de Dados.



Para desativar o servidor **HSQLDB**, no ambiente visual, deve-se entrar com o comando **SHUTDOWN**.



Escreva agora uma aplicação java para acessar o banco de dados **db** no servidor de banco de dados **HSQldb** e imprimir os registros armazenados na tabela **produtos**.

Abrir o Eclipse e no **package br.maua**, vamos criar uma classe chamada **ConsultaProdutos**.

```
package br.maua;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class ConsultaProdutos {

    public static void main(String[] args) throws SQLException {

        try {

            // Escreva o código aqui !

        }

        catch (SQLException e) {

            System.out.println("Erro SQLException....");

        }

        catch (Exception e) {

            System.out.println("Problemas na conexao ao HSQldb....");

        }

    }

}
```

```
        }  
    }  
}
```

Após a execução da consulta, deverá ser exibido na console do **Eclipse**:

```
conexao ao HSQLDB feita com SUCESSO !  
resultado = true  
1  
MICROCOMPUTADOR  
DELL  
2  
MONITOR  
SAMSUNG  
3  
CAMERA NIKON  
D810 - FULL FRAME  
Consulta feita com sucesso...
```