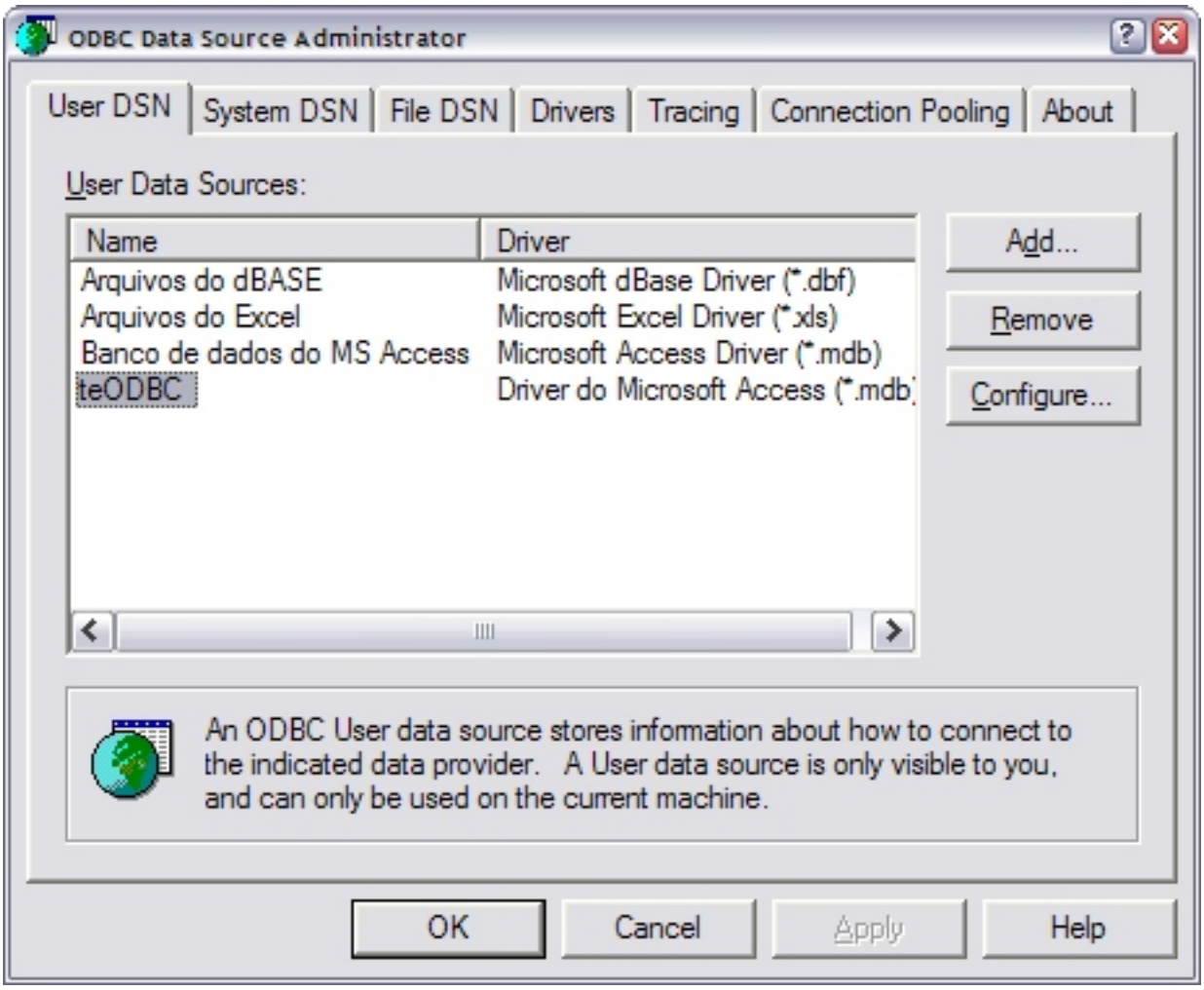
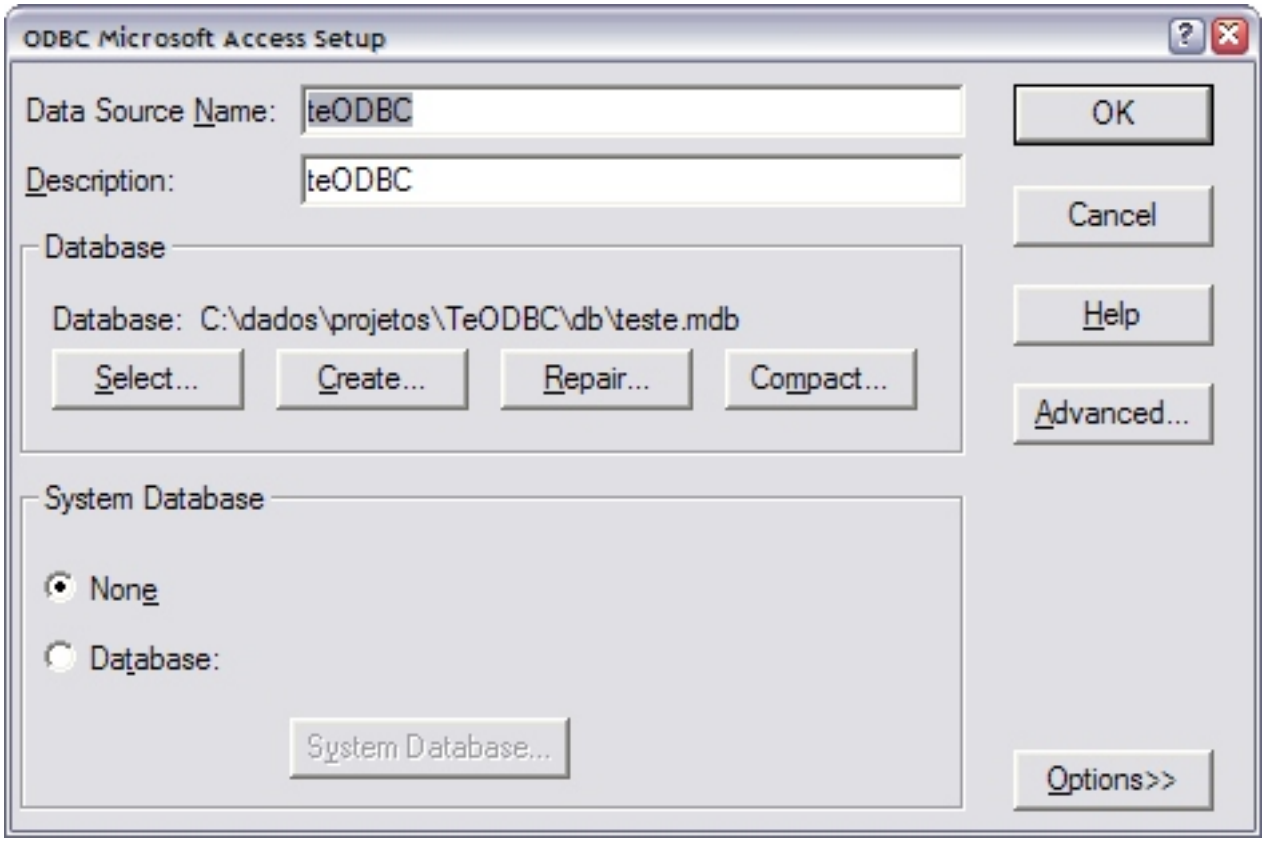
Utilizando o Database no TerraLib da mesma forma nas plataformas Linux e Windows

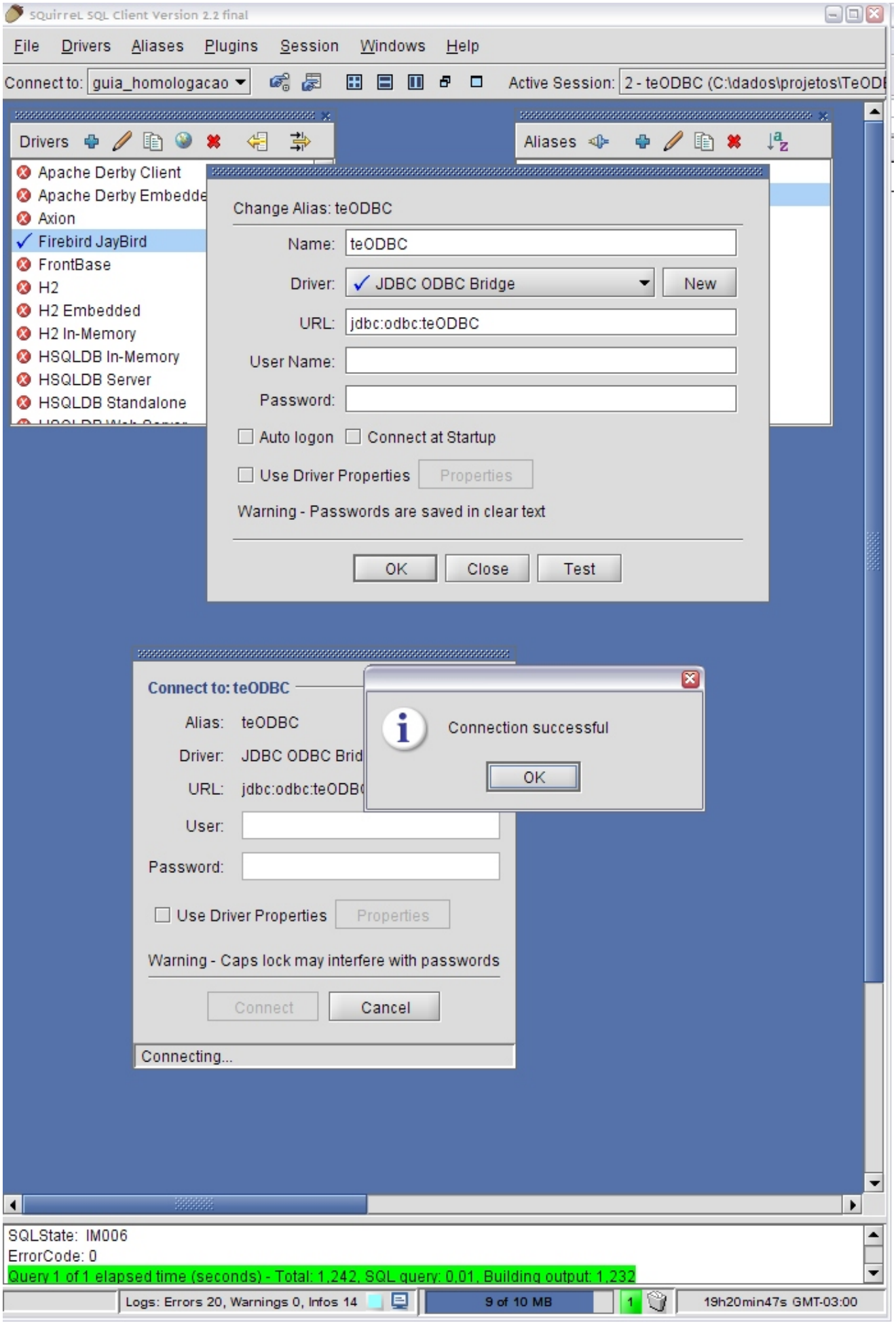
Primeiro passo (A): Configurei um Data source usando o ODBC do windows;
(Painel de Controle -> Ferramentas Administrativas -> Fonte de Dados (ODBC)



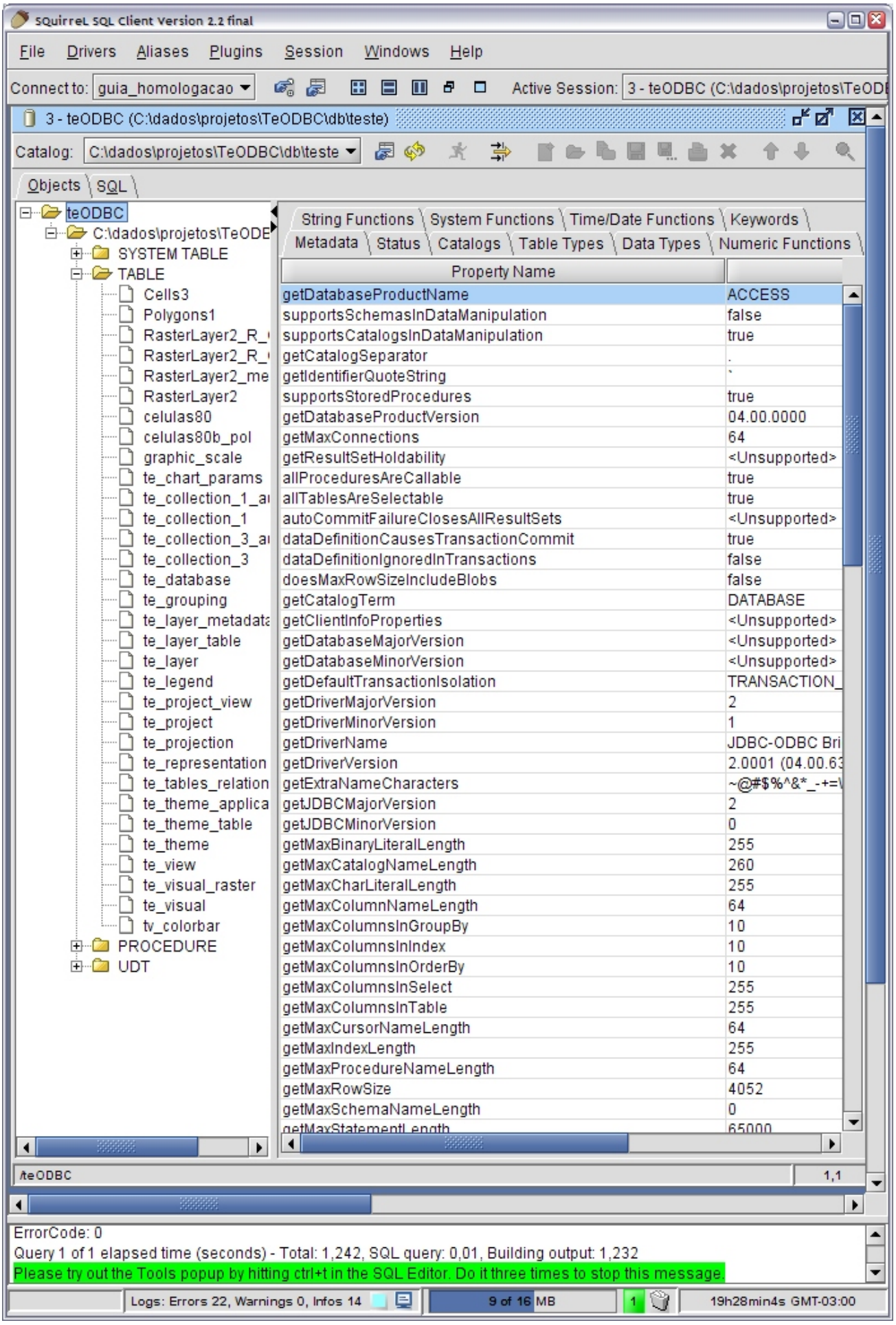
Coloquei o nome do Data Source como teODBC para testes:



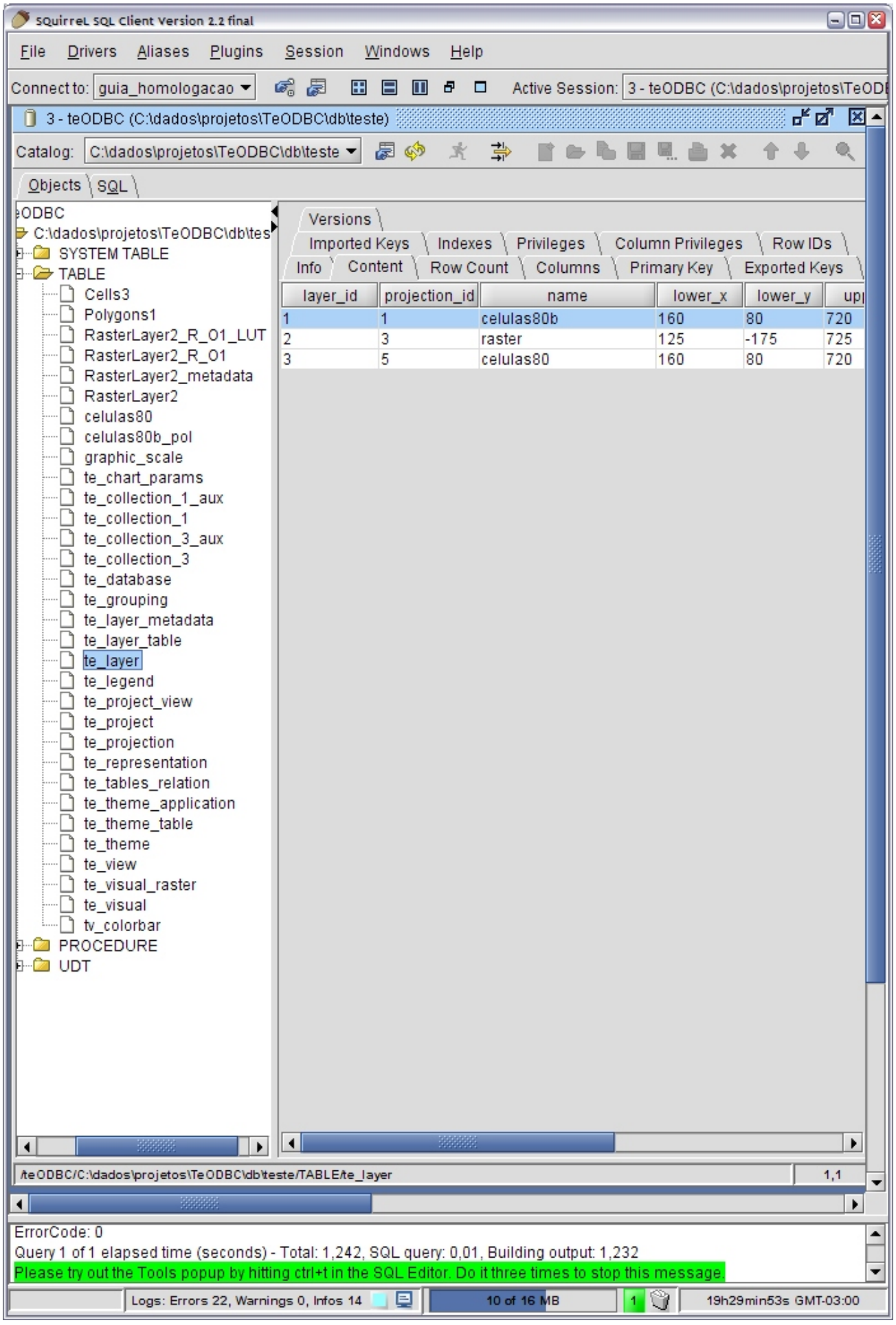
Segundo passo: Baixei o SQuirreL, um cliente SQL para uma infinidade de bancos de dados.



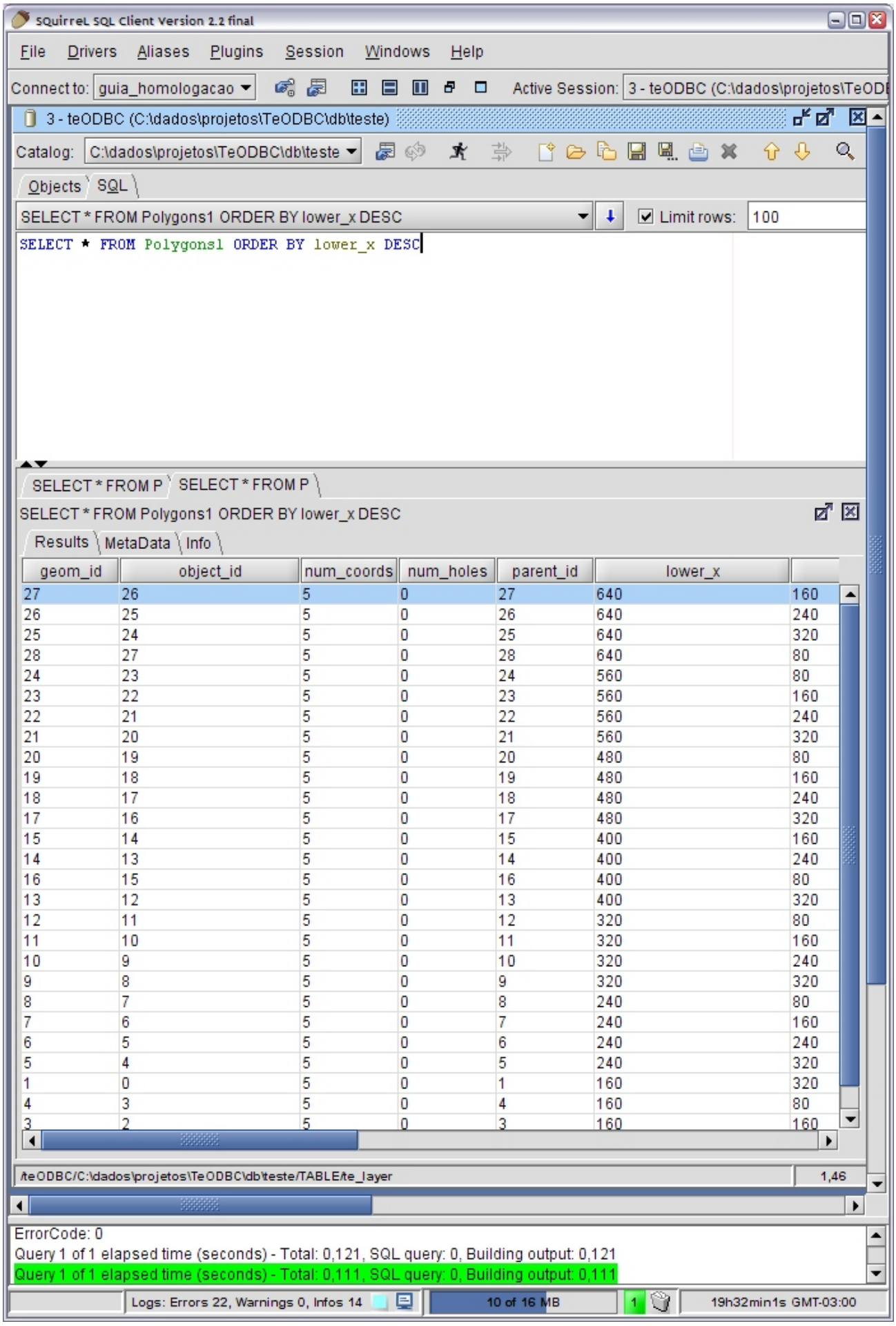
O SQuirreL mostra os metadados entre outras coisas:



O SQuirreL mostra o conteúdo das tabelas.



Terceiro passo: Fazer uma consulta no Squirrel com a cláusula ORDER BY



Quarto passo: Fazer um programa simples em java que use o ODBC no windows:

```
package com.terralib.db.odbc;

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

public class TestODBC {

/**
 * @param args
 */
public static void main(String[] args) {
try
     {
      // Load the database driver
      Class.forName( "sun.jdbc.odbc.JdbcOdbcDriver" ) ;

      // Get a connection to the database
      Connection conn = DriverManager.getConnection( "jdbc:odbc:teODBC" ) ;

      // Print all warnings
      for( SQLWarning warn = conn.getWarnings(); warn != null; warn = warn.getNextWarning() )
         {
          System.out.println( "SQL Warning:" ) ;
          System.out.println( "State  : " + warn.getSQLState()  ) ;
          System.out.println( "Message: " + warn.getMessage()   ) ;
          System.out.println( "Error  : " + warn.getErrorCode() ) ;
         }

      // Get a statement from the connection
      Statement stmt = conn.createStatement() ;

      // Execute the query
      ResultSet rs = stmt.executeQuery( "SELECT * FROM Polygons1 ORDER BY lower_x DESC" ) ;

      // Loop through the result set
      while( rs.next() ){
         System.out.println( rs.getString("lower_x") ) ;
      }

      // Close the result set, statement and the connection
      rs.close() ;
      stmt.close() ;
      conn.close() ;
     }
  catch( SQLException se )
     {
      System.out.println( "SQL Exception:" ) ;

      // Loop through the SQL Exceptions
      while( se != null )
         {
          System.out.println( "State  : " + se.getSQLState()  ) ;
          System.out.println( "Message: " + se.getMessage()   ) ;
          System.out.println( "Error  : " + se.getErrorCode() ) ;

          se = se.getNextException() ;
         }
     }
  catch( Exception e )
     {
      System.out.println( e ) ;
     }
 }

}
```

Resultado do Programa

```
640.0
640.0
640.0
640.0
560.0
560.0
560.0
560.0
480.0
480.0
480.0
480.0
400.0
400.0
400.0
400.0
320.0
320.0
320.0
320.0
240.0
240.0
240.0
240.0
160.0
160.0
160.0
160.0
```

Quinto passo: Configurar uma fonte de dados ODBC no linux usando outra API ODBC

Sexto passo: Rodar o programa java com a cláusula ORDER BY e verificar se não ocorreram erros. Caso ocorram erros, repetir o quinto passo, caso contrário vá para o sétimo passo.

Sétimo passo: Use esta fonte de dados ODBC com a atual API funcional, e compile o Terralib como se o objetivo fosse usar uma fonte de dados ODBC padrão no win.