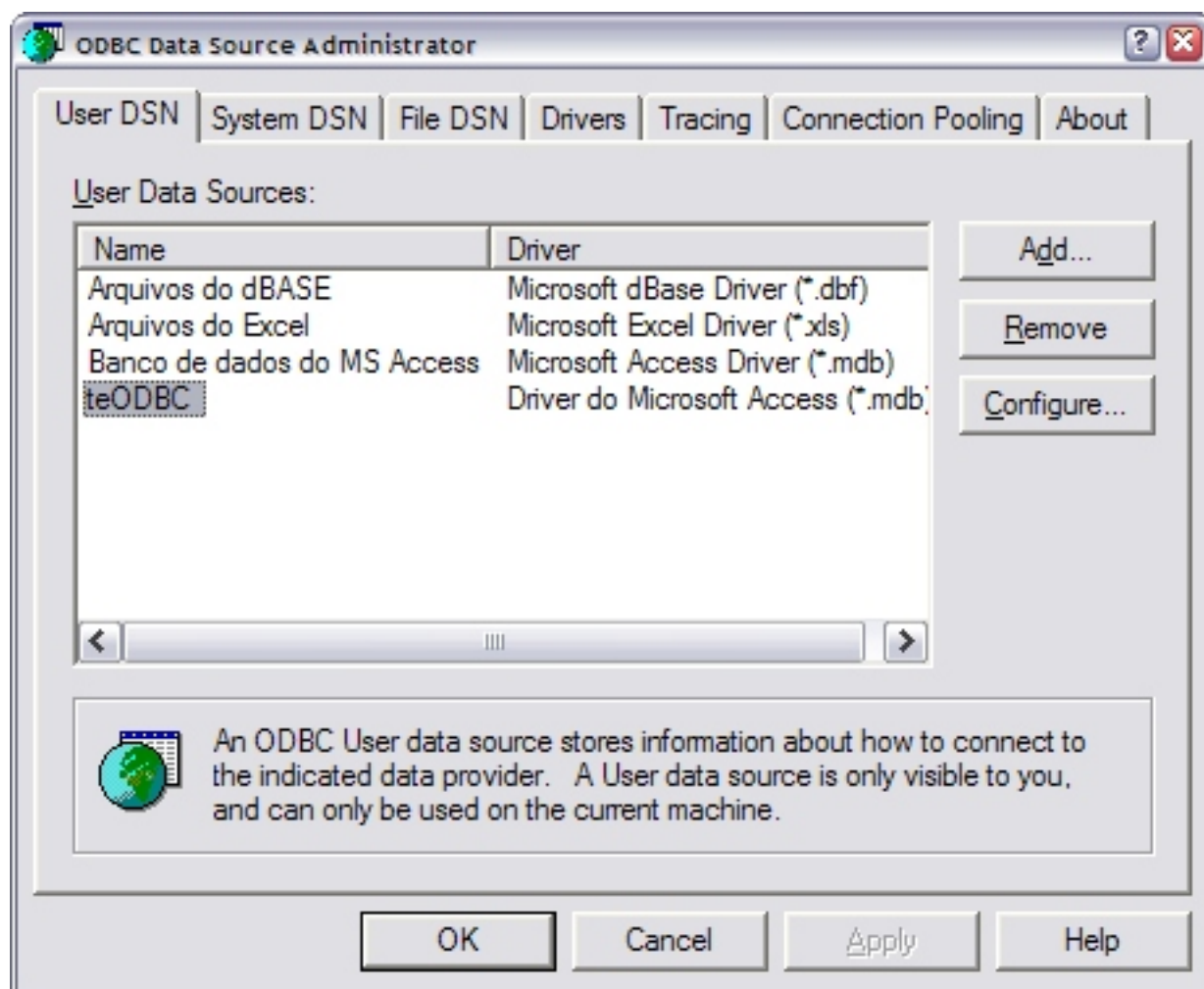
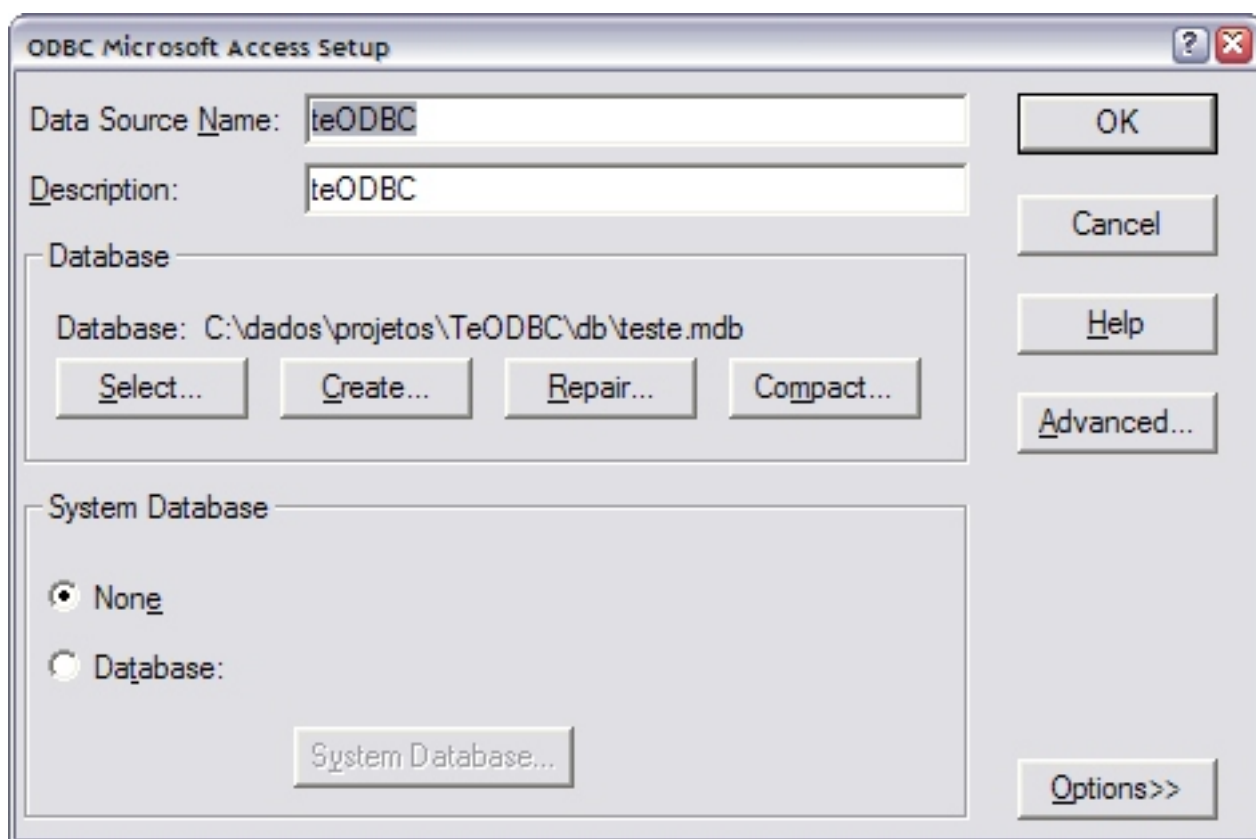


**Utilizando o Database no TerraLib da mesma forma nas
plataformas Linux e Windows**

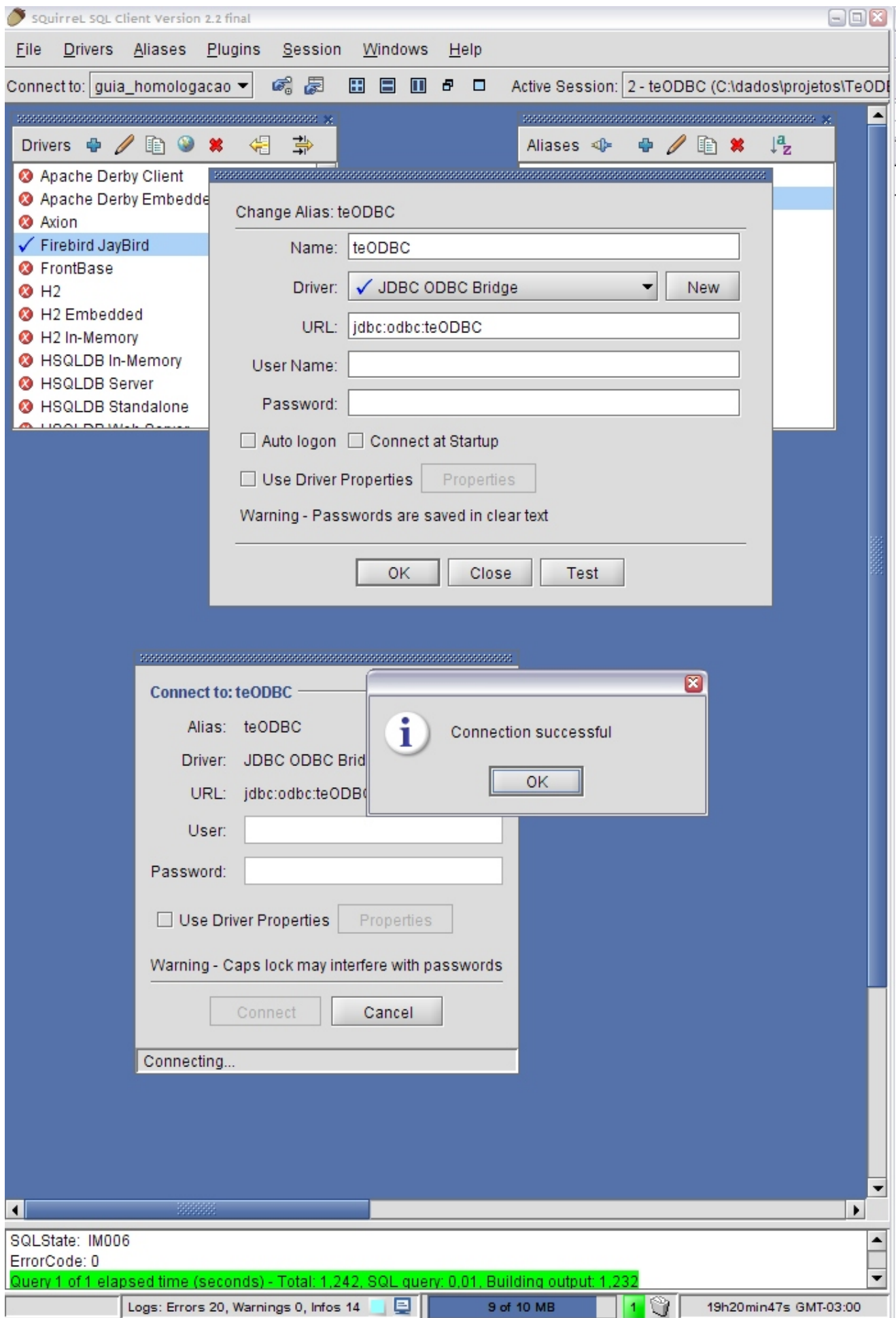
Primeiro passo (A): Configurei um Data source usando o ODBC do windows;
(Painel de Controle -> Ferramentas Administrativas -> Fonte de Dados (ODBC)



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:

The screenshot shows the Squirrel SQL Client interface. The main window displays the metadata for a connection to a teODBC database. The left pane shows a tree view of the database objects, including SYSTEM TABLE and TABLE. The right pane shows a list of metadata properties and their values.

| Property Name | Value |
|---------------------------------------|------------------|
| getDatabaseProductName | ACCESS |
| supportsSchemasInDataManipulation | false |
| supportsCatalogsInDataManipulation | true |
| getCatalogSeparator | . |
| getIdentifierQuoteString | . |
| supportsStoredProcedures | true |
| getDatabaseProductVersion | 04.00.0000 |
| getMaxConnections | 64 |
| getResultSetHoldability | <Unsupported> |
| allProceduresAreCallable | true |
| allTablesAreSelectable | true |
| autoCommitFailureClosesAllResultSets | <Unsupported> |
| dataDefinitionCausesTransactionCommit | true |
| dataDefinitionIgnoredInTransactions | false |
| doesMaxRowSizeIncludeBlobs | false |
| getCatalogTerm | DATABASE |
| getClientInfoProperties | <Unsupported> |
| getDatabaseMajorVersion | <Unsupported> |
| getDatabaseMinorVersion | <Unsupported> |
| getDefaultTransactionIsolation | TRANSACTION_ |
| getDriverMajorVersion | 2 |
| getDriverMinorVersion | 1 |
| getDriverName | JDBC-ODBC Bri |
| getDriverVersion | 2.0001 (04.00.63 |
| getExtraNameCharacters | ~@#\$\$%^&*_-+=\ |
| getJDBCMinorVersion | 0 |
| getMaxBinaryLiteralLength | 255 |
| getMaxCatalogNameLength | 260 |
| getMaxCharLiteralLength | 255 |
| getMaxColumnNameLength | 64 |
| getMaxColumnsInGroupBy | 10 |
| getMaxColumnsInIndex | 10 |
| getMaxColumnsInOrderBy | 10 |
| getMaxColumnsInSelect | 255 |
| getMaxColumnsInTable | 255 |
| getMaxCursorNameLength | 64 |
| getMaxIndexLength | 255 |
| getMaxProcedureNameLength | 64 |
| getMaxRowSize | 4052 |
| getMaxSchemaNameLength | 0 |
| getMaxStatementLength | 65000 |

At the bottom of the window, the status bar shows the following information:

- ErrorCode: 0
- Query 1 of 1 elapsed time (seconds) - Total: 1,242, SQL query: 0,01, Building output: 1,232
- Logs: Errors 22, Warnings 0, Infos 14
- 9 of 16 MB
- 19h28min4s GMT-03:00

O Squirrel mostra o conteúdo das tabelas.

The screenshot shows the Squirrel SQL Client interface. The left pane displays a tree view of the database objects, with 'te_layer' selected under the 'TABLE' folder. The right pane shows a table view of the 'te_layer' table with the following data:

| layer_id | projection_id | name | lower_x | lower_y | upper_x | upper_y |
|----------|---------------|------------|---------|---------|---------|---------|
| 1 | 1 | celulas80b | 160 | 80 | 720 | 720 |
| 2 | 3 | raster | 125 | -175 | 725 | 725 |
| 3 | 5 | celulas80 | 160 | 80 | 720 | 720 |

At the bottom of the window, the status bar shows 'Log: Errors 22, Warnings 0, Infos 14', '10 of 16 MB', and '19h29min53s GMT-03:00'. A message box at the bottom of the SQL editor reads: 'Please try out the Tools popup by hitting ctrl+t in the SQL Editor. Do it three times to stop this message'.

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

The screenshot shows the Squirrel SQL Client interface. The main window title is "Squirrel SQL Client Version 2.2 final". The "Connect to:" field is set to "guia_homologacao" and the "Active Session:" is "3 - teODBC (C:\dados\projetos\TeODBC\...". The "Catalog:" is "C:\dados\projetos\TeODBC\db\teste". The "SQL" tab is active, showing the query: `SELECT * FROM Polygons1 ORDER BY lower_x DESC`. The "Limit rows:" is set to 100. The "Results" tab is active, displaying a table with the following data:

| geom_id | object_id | num_coords | num_holes | parent_id | lower_x |
|---------|-----------|------------|-----------|-----------|---------|
| 27 | 26 | 5 | 0 | 27 | 640 |
| 26 | 25 | 5 | 0 | 26 | 640 |
| 25 | 24 | 5 | 0 | 25 | 640 |
| 28 | 27 | 5 | 0 | 28 | 640 |
| 24 | 23 | 5 | 0 | 24 | 560 |
| 23 | 22 | 5 | 0 | 23 | 560 |
| 22 | 21 | 5 | 0 | 22 | 560 |
| 21 | 20 | 5 | 0 | 21 | 560 |
| 20 | 19 | 5 | 0 | 20 | 480 |
| 19 | 18 | 5 | 0 | 19 | 480 |
| 18 | 17 | 5 | 0 | 18 | 480 |
| 17 | 16 | 5 | 0 | 17 | 480 |
| 15 | 14 | 5 | 0 | 15 | 400 |
| 14 | 13 | 5 | 0 | 14 | 400 |
| 16 | 15 | 5 | 0 | 16 | 400 |
| 13 | 12 | 5 | 0 | 13 | 400 |
| 12 | 11 | 5 | 0 | 12 | 320 |
| 11 | 10 | 5 | 0 | 11 | 320 |
| 10 | 9 | 5 | 0 | 10 | 320 |
| 9 | 8 | 5 | 0 | 9 | 320 |
| 8 | 7 | 5 | 0 | 8 | 240 |
| 7 | 6 | 5 | 0 | 7 | 240 |
| 6 | 5 | 5 | 0 | 6 | 240 |
| 5 | 4 | 5 | 0 | 5 | 240 |
| 1 | 0 | 5 | 0 | 1 | 160 |
| 4 | 3 | 5 | 0 | 4 | 160 |
| 3 | 2 | 5 | 0 | 3 | 160 |

The status bar at the bottom shows "Logs: Errors 22, Warnings 0, Infos 14", "10 of 16 MB", and "19h32min1s GMT-03:00". The "Error Code" is 0. The query execution time is shown as "Query 1 of 1 elapsed time (seconds) - Total: 0,111, SQL query: 0, Building output: 0,111".

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.