

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

Control de autenticación y
 acceso (Spring Security)
 UDDI
 Web Services
 Rest Services
 Social SSO
 SSO (Cas)

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)

¿Qué ofrece Autentia?



Somos su empresa de arquitectura y soporte a desarrollo

Java/.Net

En cualquier empresa mediana o grande donde exista un departamento de desarrollo informático encontramos un difícil equilibrio entre innovación y servicio.

- Si se trata de estar a la última, es bastante fácil perder la perspectiva y entrar en una espiral de innovación continua, olvidando los objetivos estratégicos de la propia empresa. Además, al disponer de unos recursos limitados en número y un ambiente muy acotado, no es posible alimentarse de experiencias (éxitos y fracasos) de entornos distintos. Se produce endogamia tecnológica.
- Si el día a día te puede, es realmente fácil quedarse obsoleto y perder competitividad.

Tener personal cualificado y disponible para ayudar puntualmente es complejo por los modelos organizativos y de costes (incluso para las grandes consultoras). En Autentia, nos hemos colocado en el centro de este problema, creando una empresa de arquitectura tecnológica y soporte al desarrollo en nuevas tecnologías. “Somos expertos en desarrollos empresariales compartidos por muchas empresas”.

Asistencia a responsables de áreas de Tecnología.

Asesoramiento personal técnico/organizativo.
Recomendaciones arquitectónicas: Frameworks
Técnicas de control de proyectos.
Auditorias de calidad y rendimiento.

Desarrollos de sistemas Web y componentes a medida.

Toma de requisitos, análisis, diseño y desarrollo.
Construcción de Sistemas transaccionales Web
Reingeniería de aplicaciones
Desarrollo y/o evolución de Frameworks

Formación a la carta

Dirección de proyectos.
Introducción a las nuevas tecnologías para directivos.
Gestión eficaz del tiempo.
Análisis y diseño orientado a objeto y UML.
Patrones de diseño
Java/J2EE a todos los niveles.
Buenas prácticas y técnicas avanzadas de desarrollo J2EE
Struts / JSF / EJBs / Hibernate
C/C++ en Windows y Linux.
Arquitectura de Aplicaciones Empresariales

Autentia: Nuevas soluciones para problemas antiguos... ¿Hablamos?

Isaac Newton 1, Local 28
Tres Cantos 28760
Madrid

Roberto Canales Mora: 655 99 11 72
Fax: 91 656 65 04
E-mail: rcanales@autentia.com

Tutorial desarrollado por: [Francisco Javier Martínez Páez](#)

Puedes encontrarme en [Autentia](#)
Somos expertos en [Java/J2EE](#)
Contacta en info@autentia.com



Descargar este documento en formato PDF [JDBCResultset.pdf](#)

[Firma en nuestro libro de Visitas](#)

[Master Java J2ee Oracle](#)

100% alumnos ya trabajan. Nuevo temario de Struts + J2ME.
www.grupoatrium.com

[Visual Studio 2005](#)

La diferencia es obvia Pruébalo y compara
www.microsoft.es

[diseño web Zaragoza](#)

Diseño y programación web a medida
Excelente relación calidad/precio
inicionet.com

PAGINAR UN RESULTSET

Los ejemplos de este tutorial están hechos con el siguiente entorno de desarrollo:

- Jboss Eclipse IDE Milestone 5.
- JDK 1.4
- MySQL 5.0
- MySQL Administrator (opcional)
- MySQL Connector/J (Driver tipo 4 que implementa la versión JDBC 3.0)

¿ JDBC ?

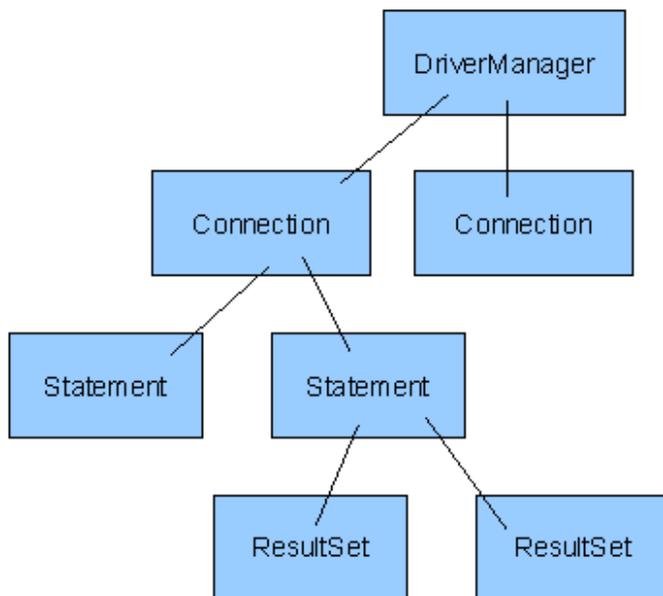
Aunque este tutorial está orientado a gente con algún conocimiento de JDBC, no está de más realizar una pequeña introducción, para refrescar ciertos conceptos.

El API de JDBC está expresado como un conjunto de interfaces abstractos que nos permiten principalmente:

- Manejar un conjunto de Drivers JDBC
- Abrir conexiones a la base de datos
- Ejecutar sentencias SQL.
- Procesar los resultados obtenidos.

Será por tanto cuestión de cada fabricante implementar dicha funcionalidad en los drivers.

La relación de los Interfaces principales es la siguiente:



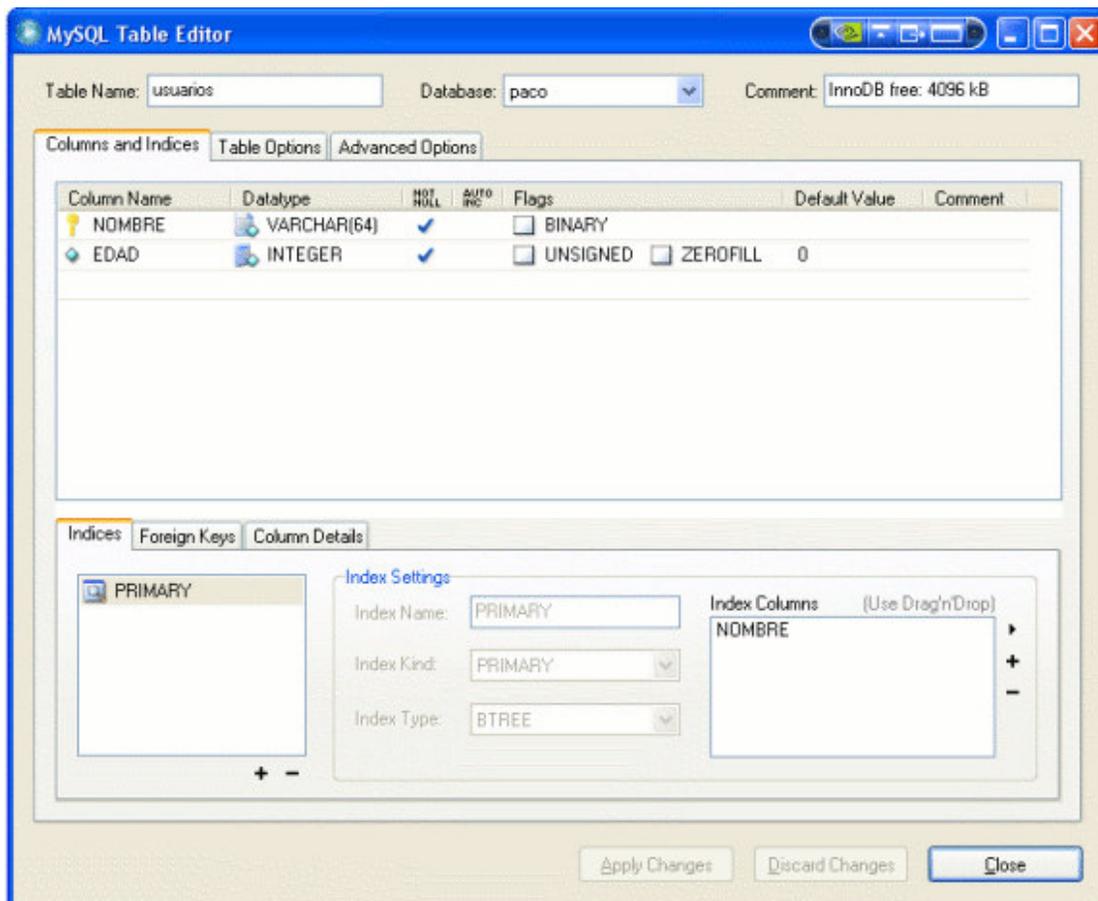
Y ya sin más dilación, y dando por supuesto que se manejan con cierta soltura estas clases, vamos a currar un poquito.

PAGINANDO

Pocas aplicaciones WEB no tienen una página que muestra de manera paginada los resultados de una consulta. Cuando nos disponemos a decidir cual es la mejor manera de paginar (o mejor dicho) donde paginar, surgen ciertas dudas al respecto. Yo siempre he creído (y si el driver lo permite) que la mejor manera de paginar es en el ResultSet, y no posteriormente en alguna lista o colección. Alguno me preguntará que porqué pienso así, la respuesta es simple, si paginamos en el ResultSet, no es necesario transmitir por la red todos los registros obtenidos, sino sólo los estrictamente necesarios. Hay una manera mejor que esta, y es usar alguna característica de nuestro motor de base de datos por ejemplo, el rownum de Oracle, pero esto no se incluye en todos los motores de base de datos.

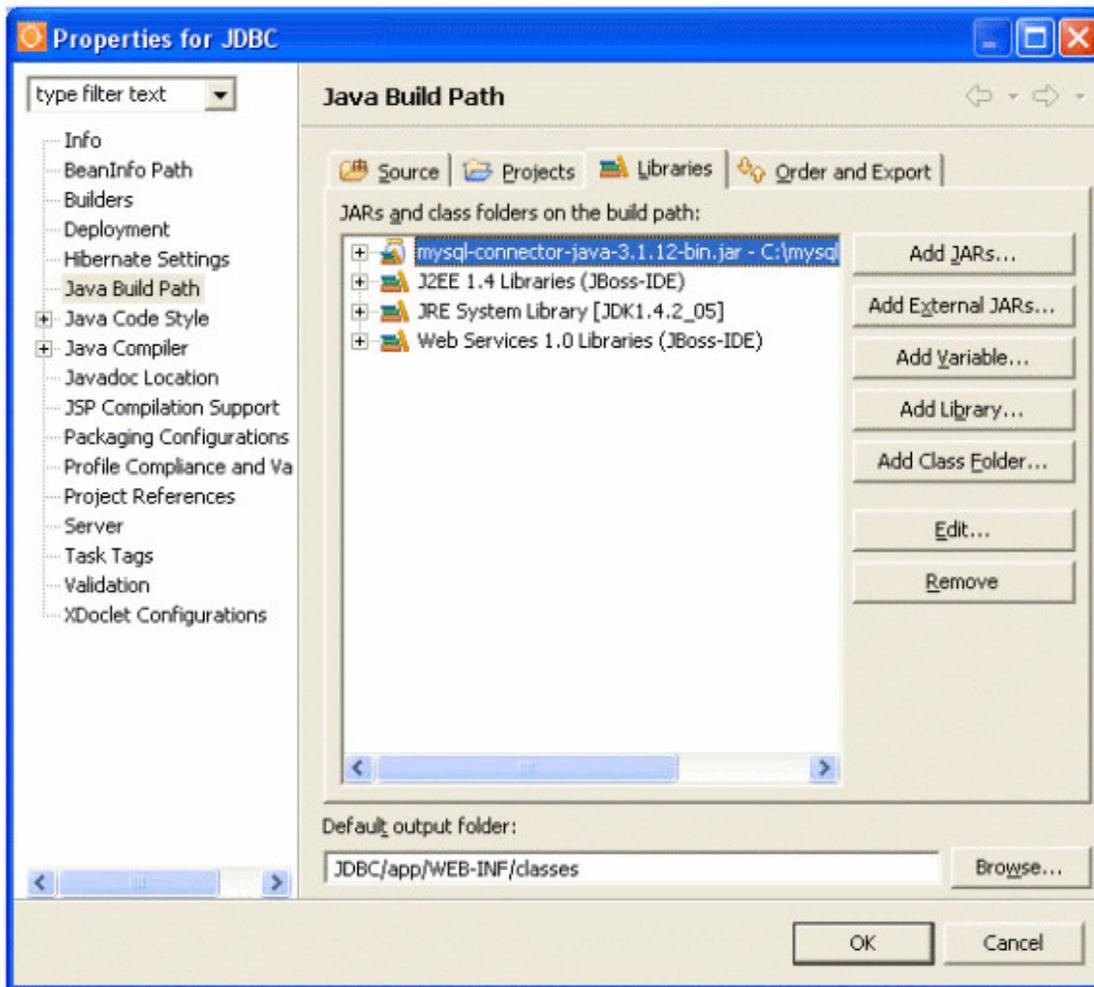
Lo primero que vamos a hacer es crearnos una tabla en nuestra base de datos:

Usando la herramienta MySQL Administrator, nos creamos la tabla USUARIOS, con los siguientes campos:

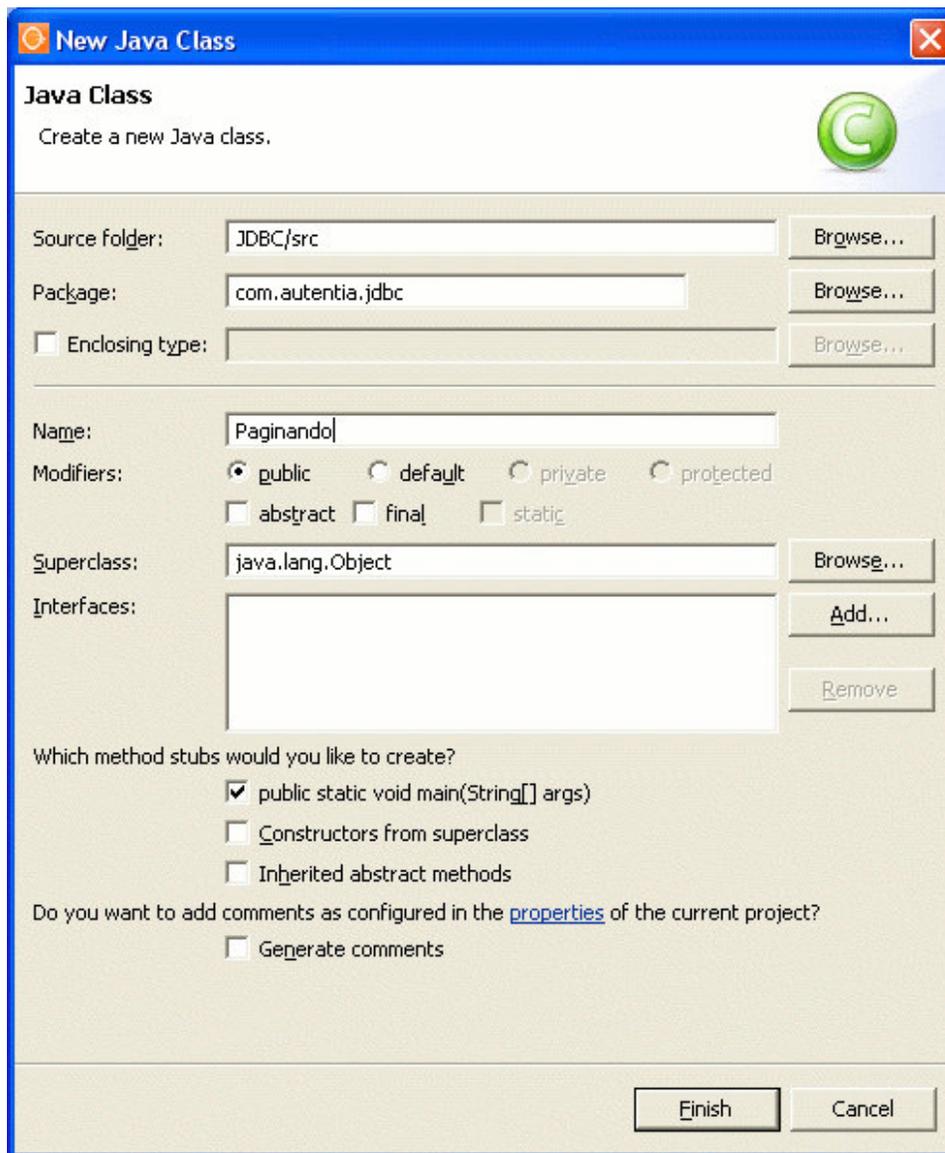


Una vez creada la tabla, podemos abrir el eclipse.

Nos generamos un proyecto nuevo (yo le he llamado JDBC). Recordad que debemos meter en el classpath el driver JDBC de mysql:



Creamos la clase que nos va a servir de ejemplo: La denominamos Paginando: (le decimos que nos genere el método main que después rellenaremos)



Vamos a crear nuestro primer método. getConnection() y dos constantes que se usarán dentro del método:

```
// Driver de MySQL
protected static String dbClass = "com.mysql.jdbc.Driver";

// Cadena de conexión
protected static String dbUrl = "jdbc:mysql://paco";

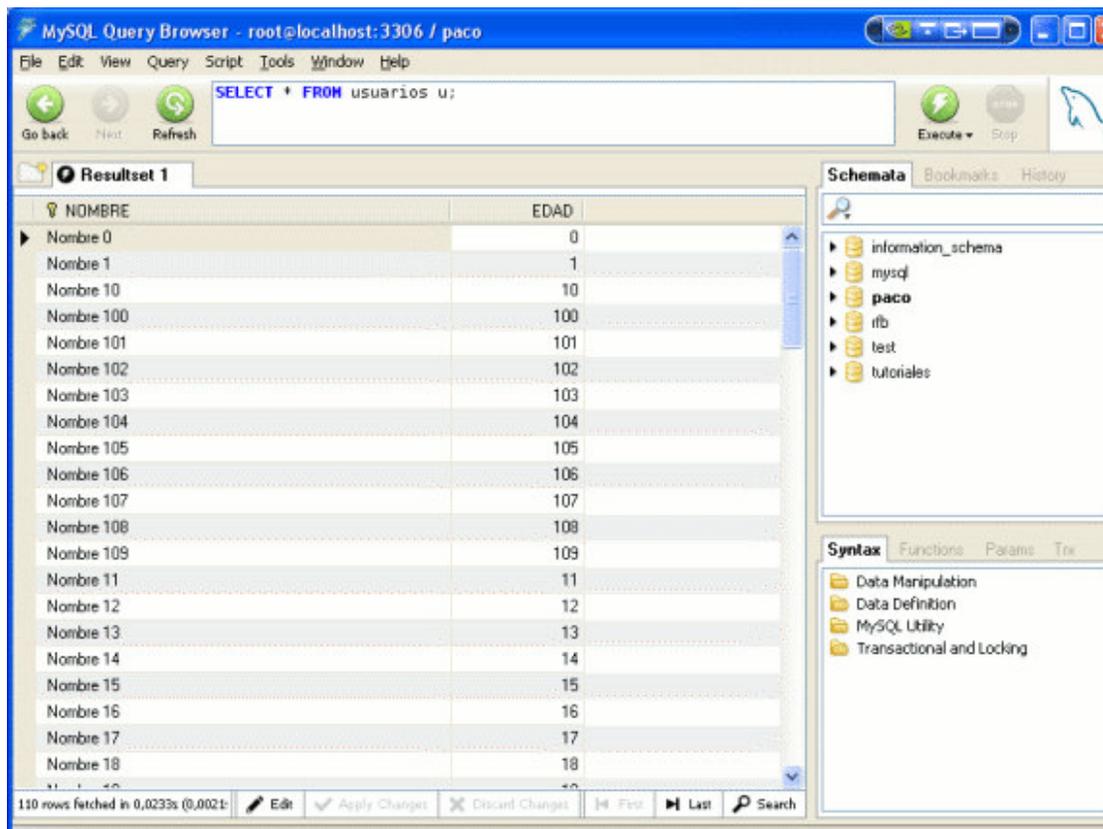
// Método que crea una conexión a nuestra Base de datos
protected Connection getConnection() {

    Properties props = new Properties();
    props.put("user", "tu_usuario");
    props.put("password", "tu_clave");
    Connection conBBDD = null;
    try {
        Class.forName(dbClass);
        conBBDD=DriverManager.getConnection(dbUrl, props);
    } catch(Exception e) {
        return null;
    }

    return conBBDD;
}
```

Nos vamos a crear un método para rellenar con datos nuestra tabla, que nos servirá también para comprobar que funcione el método getConnection().

```
protected void insertaRegistros() {
```

Ya tenemos datos suficientes en la tabla para nuestro ejemplo.

Vamos ahora a crear el código para paginar. Creamos un método que llamaremos ejecutaQueryPaginada. Este método recibe:

- String query: Consulta SQL a lanzar
- int numPagina: número de página que queremos consultar
- int numRegPagina: número de registros por página.

Este método retorna una lista de la siguiente manera:

- El primer elemento será un Integer con el número de elementos totales en la tabla que cumplen la query.
- El segundo elemento será una lista de Strings con los nombres de las columnas consultadas.
- A continuación devolverá listas de Strings con los valores correspondientes a la página buscada.

```
public ArrayList ejecutaQueryPaginada(String query, int numPagina,
    int numRegPagina) {
    ArrayList alResultado = new ArrayList();
    Connection conBBDD = null;
    Statement st = null;
    ResultSet rs = null;
    try {
        conBBDD=getConnection();
        st = conBBDD.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
        st.setFetchSize(numRegPagina);

        rs = st.executeQuery(query);

        int fila = numRegPagina * (numPagina - 1) + 1;
        int cont = 1;

        ResultSetMetaData md = rs.getMetaData();
        int numeroColumnas = md.getColumnCount();
        ArrayList alRegistro = new ArrayList(numeroColumnas);

        for (int i = 1; i <= numeroColumnas; i++) {
            String nomCol = md.getColumnName(i);
            alRegistro.add(nomCol);
        }
    }
}
```

```

alResultado.add(alRegistro);

if (rs.absolute(fila) && numRegPagina > 0) {
    do {
        alRegistro = new ArrayList();
        for (int i = 1; i <= numeroColumnas; i++) {
            alRegistro.add(rs.getString(i));
        }

        alResultado.add(alRegistro);
        cont++;
    }
    while (rs.next() && (cont <= numRegPagina));
}
// Se incluye el primer elemento del ArrayList con un objeto Integer
// con el Numero de Tuplas TOTAL de la query paginada
// Se mueve el cursor a ultima tupla
Integer numTuplasTotal = new Integer(0);
if (rs.last()) { // Existen tuplas y el cursor esta en la ultima,
    // basta con consultar el numero de esa tupla
    numTuplasTotal = new Integer(rs.getRow());
}
alResultado.add(0, numTuplasTotal);
} catch (Exception e) {
    e.printStackTrace();
} finally {
    if(st!=null) {
        try {
            st.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    if(rs!=null) {
        try {
            rs.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    if(conBBDD!=null) {
        try {
            conBBDD.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

return (alResultado);
}
}

```

Se ha resaltado **ResultSet.TYPE_SCROLL_INSENSITIVE**, porque para que nuestro código no genere un error, el drive de base de datos nos debe permitir generar "ResultSets scrollables", es decir, ResultSets que nos permitan navegar por el hacia adelante y hacia atrás.

Vamos a invocar el método desde main():

```

public static void main(String[] args) {

    Paginando pag = new Paginando();
    String consulta = "SELECT * FROM USUARIOS ORDER BY EDAD";
    List lista = pag.ejecutaQueryPaginada(consulta,1,10);

    Integer numeroRegistros = (Integer)lista.get(0);

    System.out.println("NUMERO DE REGISTROS TOTALES:"+numeroRegistros.toString());

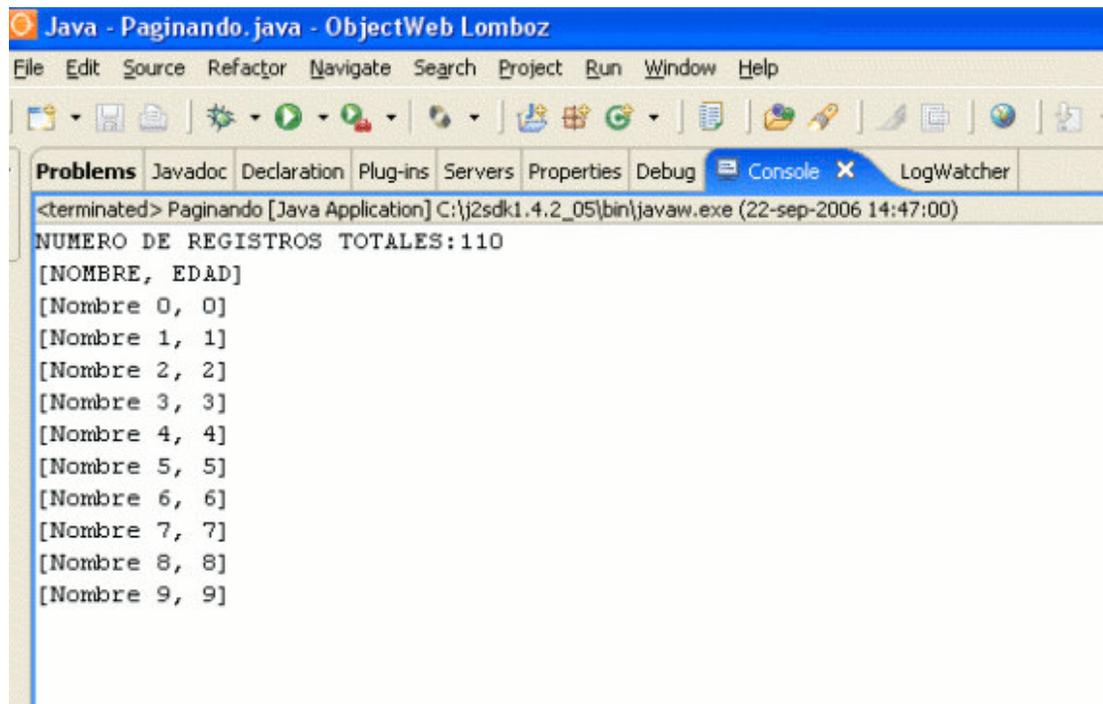
    for(int i=1;i<lista.size();i++) {
        List lista1 = (List) lista.get(i);
    }
}

```

```
        System.out.println(lista1.toString());
    }
}
```

Hemos solicitado los 10 primeros usuarios más jóvenes: (pagina 1/10)

Lo ejecutamos y mostramos el resultado en la consola:

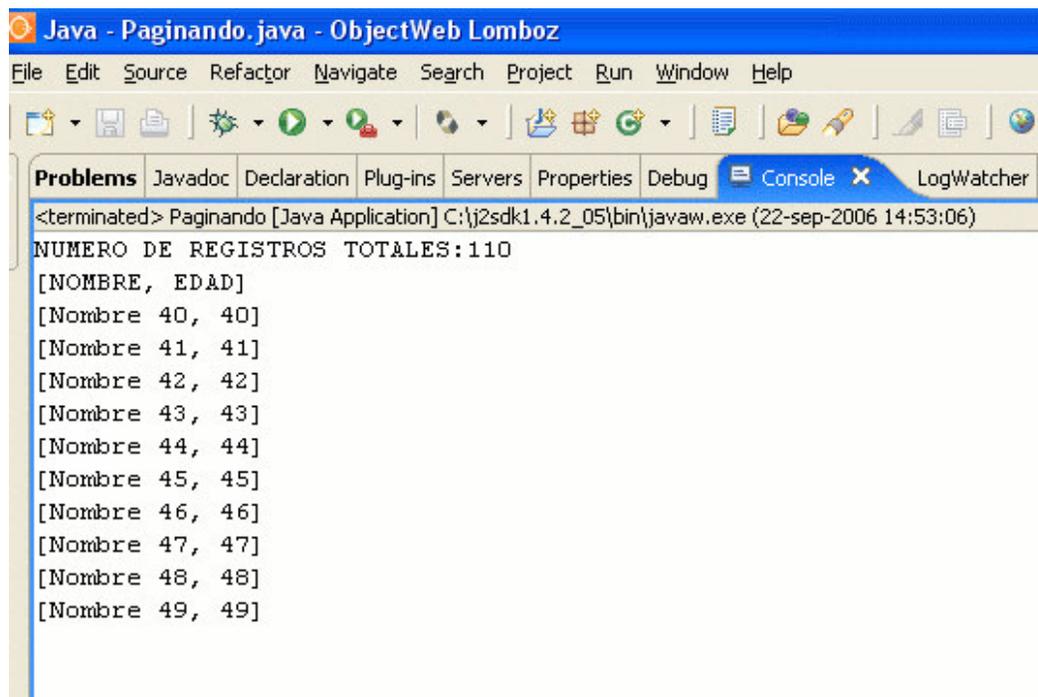


```
<terminated> Paginando [Java Application] C:\j2sdk1.4.2_05\bin\javaw.exe (22-sep-2006 14:47:00)
NUMERO DE REGISTROS TOTALES:110
[NOMBRE, EDAD]
[Nombre 0, 0]
[Nombre 1, 1]
[Nombre 2, 2]
[Nombre 3, 3]
[Nombre 4, 4]
[Nombre 5, 5]
[Nombre 6, 6]
[Nombre 7, 7]
[Nombre 8, 8]
[Nombre 9, 9]
```

Vamos a solicitar la página 5:

List lista = **pag.ejecutaQueryPaginada(consulta,5,10);**

Lo ejecutamos y mostramos el resultado en la consola:

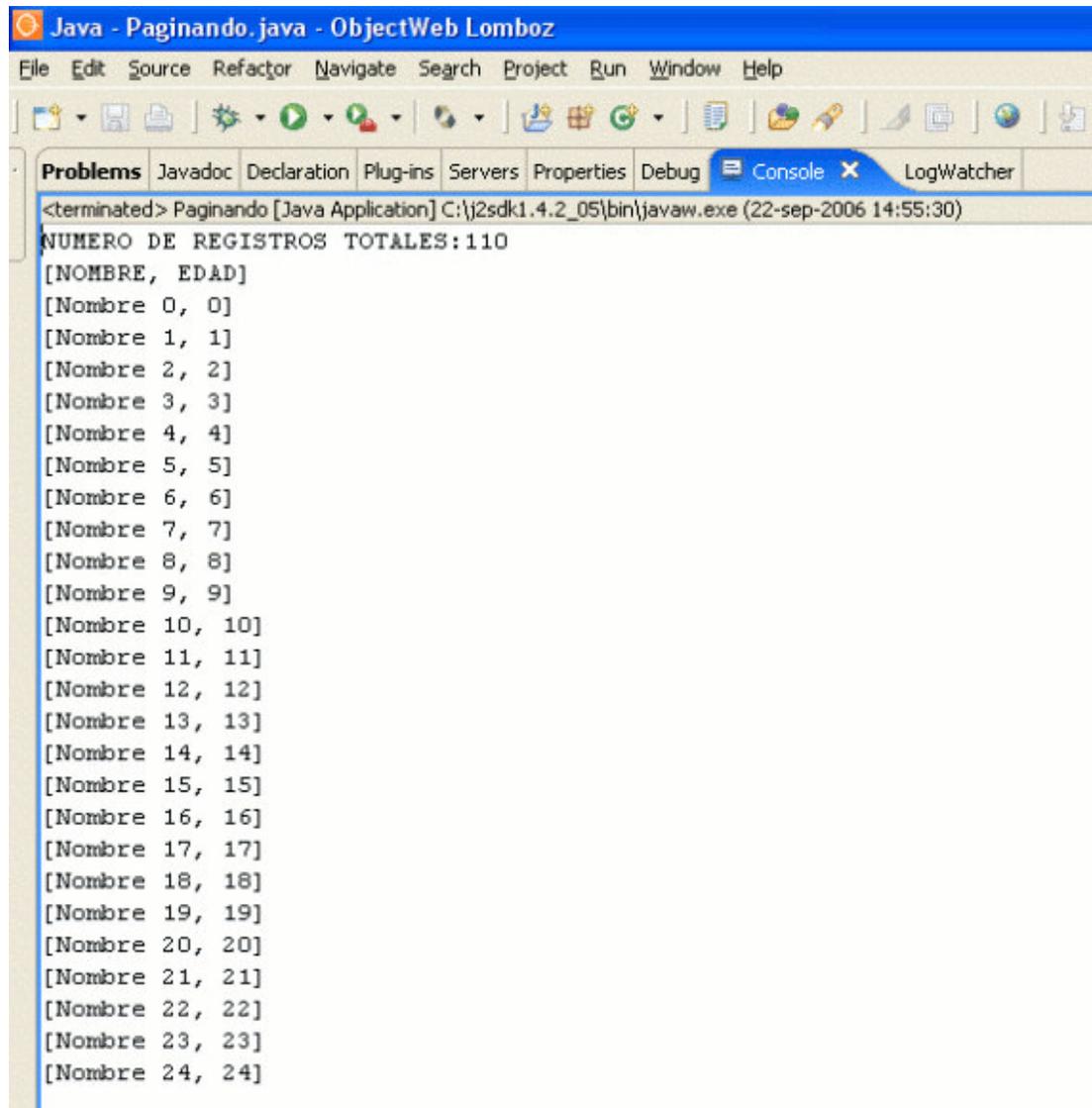


```
<terminated> Paginando [Java Application] C:\j2sdk1.4.2_05\bin\javaw.exe (22-sep-2006 14:53:06)
NUMERO DE REGISTROS TOTALES:110
[NOMBRE, EDAD]
[Nombre 40, 40]
[Nombre 41, 41]
[Nombre 42, 42]
[Nombre 43, 43]
[Nombre 44, 44]
[Nombre 45, 45]
[Nombre 46, 46]
[Nombre 47, 47]
[Nombre 48, 48]
[Nombre 49, 49]
```

Vamos a solicitar la página 1 devolviendo 25 registros por página:

List lista = **pag.ejecutaQueryPaginada(consulta,1,25);**

Lo ejecutamos y mostramos el resultado en la consola:



```
<terminated> Paginando [Java Application] C:\j2sdk1.4.2_05\bin\javaw.exe (22-sep-2006 14:55:30)
NUMERO DE REGISTROS TOTALES:110
[NOMBRE, EDAD]
[Nombre 0, 0]
[Nombre 1, 1]
[Nombre 2, 2]
[Nombre 3, 3]
[Nombre 4, 4]
[Nombre 5, 5]
[Nombre 6, 6]
[Nombre 7, 7]
[Nombre 8, 8]
[Nombre 9, 9]
[Nombre 10, 10]
[Nombre 11, 11]
[Nombre 12, 12]
[Nombre 13, 13]
[Nombre 14, 14]
[Nombre 15, 15]
[Nombre 16, 16]
[Nombre 17, 17]
[Nombre 18, 18]
[Nombre 19, 19]
[Nombre 20, 20]
[Nombre 21, 21]
[Nombre 22, 22]
[Nombre 23, 23]
[Nombre 24, 24]
```

Puedes seguir ejecutando las pruebas que quieras.

Podrías mejorar el código usando PreparedStatement en lugar de Statement para evitar SQLInjection. Necesitarías entonces un método para sustituir en el PreparedStatement los valores adecuados de manera dinámica. Pero eso lo dejo a tu imaginación...

En posteriores tutoriales publicaré uno acerca de las nuevas características que incorpora JDBC en Java 5 donde hablaremos de CachedRowSet, y verás que fácil es paginar...

Ya sabéis que si necesitáis ayuda <http://www.autentia.com>



[Puedes opinar sobre este tutorial aquí](#)

Recuerda

que el personal de [Autentia](#) te regala la mayoría del conocimiento aquí compartido ([Ver todos los tutoriales](#))

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?

¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?

info@autentia.com

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos

Autentia = Soporte a Desarrollo & Formación



[Autentia S.L.](#) Somos expertos en:

J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ..
y muchas otras cosas

Nuevo servicio de notificaciones

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales, inserta tu dirección de correo en el siguiente formulario.

Subscribirse a Novedades	
<i>e-mail</i>	<input type="text"/>
	<input type="button" value="Enviar"/>

Otros Tutoriales Recomendados ([También ver todos](#))

Nombre Corto

Descripción

Nota: Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento.

Los contenidos y comentarios de los tutoriales son responsabilidad de sus respectivos autores.

En algún caso se puede hacer referencia a marcas o nombres cuya propiedad y derechos es de sus respectivos dueños. Si algún afectado desea que incorporemos alguna reseña específica, no tiene más que solicitarlo.

Si alguien encuentra algún problema con la información publicada en este Web, rogamos que informe al administrador rcanales@adictosaltrabajo.com para su resolución.

[Patrocinados por enredados.com Hosting en Castellano con soporte Java/J2EE](#)



www.AdictosAlTrabajo.com Optimizado 800X600