001/\*  
002 \* Licensed to the Apache Software Foundation (ASF) under one or more  
003 \* contributor license agreements. See the NOTICE file distributed with  
004 \* this work for additional information regarding copyright ownership.  
005 \* The ASF licenses this file to You under the Apache License, Version 2.0  
006 \* (the "License"); you may not use this file except in compliance with  
007 \* the License. You may obtain a copy of the License at  
008 \*  
009 \* http://www.apache.org/licenses/LICENSE-2.0  
010 \*  
011 \* Unless required by applicable law or agreed to in writing, software  
012 \* distributed under the License is distributed on an "AS IS" BASIS,  
013 \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
014 \* See the License for the specific language governing permissions and  
015 \* limitations under the License.  
016 \*/  
017  
018  
019package org.apache.commons.beanutils;  
020  
021  
022import java.io.Serializable;  
023import java.sql.ResultSet;  
024import java.sql.SQLException;  
025import java.util.ArrayList;  
026import java.util.List;  
027  
028  
029/\*\*  
030 \* <p>Implementation of {@link DynaClass} that creates an in-memory collection  
031 \* of {@link DynaBean}s representing the results of an SQL query. Once the  
032 \* {@link DynaClass} instance has been created, the JDBC <code>ResultSet</code>  
033 \* and <code>Statement</code> on which it is based can be closed, and the  
034 \* underlying <code>Connection</code> can be returned to its connection pool  
035 \* (if you are using one).</p>  
036 \*  
037 \* <p>The normal usage pattern is something like:</p>  
038 \* <pre>  
039 \* Connection conn = ...; // Acquire connection from pool  
040 \* Statement stmt = conn.createStatement();  
041 \* ResultSet rs = stmt.executeQuery("SELECT ...");  
042 \* RowSetDynaClass rsdc = new RowSetDynaClass(rs);  
043 \* rs.close();  
044 \* stmt.close();  
045 \* ...; // Return connection to pool  
046 \* List rows = rsdc.getRows();  
047 \* ...; // Process the rows as desired  
048 \* </pre>  
049 \*  
050 \* <p>Each column in the result set will be represented as a {@link DynaBean}  
051 \* property of the corresponding name (optionally forced to lower case  
052 \* for portability). There will be one {@link DynaBean} in the  
053 \* <code>List</code> returned by <code>getRows()</code> for each  
054 \* row in the original <code>ResultSet</code>.</p>  
055 \*  
056 \* <p>In general, instances of {@link RowSetDynaClass} can be serialized  
057 \* and deserialized, which will automatically include the list of  
058 \* {@link DynaBean}s representing the data content. The only exception  
059 \* to this rule would be when the underlying property values that were  
060 \* copied from the <code>ResultSet</code> originally cannot themselves  
061 \* be serialized. Therefore, a {@link RowSetDynaClass} makes a very  
062 \* convenient mechanism for transporting data sets to remote Java-based  
063 \* application components.</p>  
064 \*  
065 \* @version $Id$  
066 \*/  
067  
068public class RowSetDynaClass extends JDBCDynaClass implements DynaClass, Serializable {  
069  
070  
071 // ----------------------------------------------------- Instance variables  
072  
073 /\*\*  
074 \* <p>Limits the size of the returned list. The call to  
075 \* <code>getRows()</code> will return at most limit number of rows.  
076 \* If less than or equal to 0, does not limit the size of the result.  
077 \*/  
078 protected int limit = -1;  
079  
080 /\*\*  
081 \* <p>The list of {@link DynaBean}s representing the contents of  
082 \* the original <code>ResultSet</code> on which this  
083 \* {@link RowSetDynaClass} was based.</p>  
084 \*/  
085 protected List<DynaBean> rows = new ArrayList<DynaBean>();  
086  
087 // ----------------------------------------------------------- Constructors  
088  
089  
090 /\*\*  
091 \* <p>Construct a new {@link RowSetDynaClass} for the specified  
092 \* <code>ResultSet</code>. The property names corresponding  
093 \* to column names in the result set will be lower cased.</p>  
094 \*  
095 \* @param resultSet The result set to be wrapped  
096 \*  
097 \* @throws NullPointerException if <code>resultSet</code>  
098 \* is <code>null</code>  
099 \* @throws SQLException if the metadata for this result set  
100 \* cannot be introspected  
101 \*/  
102 public RowSetDynaClass(final ResultSet resultSet) throws SQLException {  
103  
104 this(resultSet, true, -1);  
105  
106 }  
107  
108 /\*\*  
109 \* <p>Construct a new {@link RowSetDynaClass} for the specified  
110 \* <code>ResultSet</code>. The property names corresponding  
111 \* to column names in the result set will be lower cased.</p>  
112 \*  
113 \* If <code>limit</code> is not less than 0, max <code>limit</code>  
114 \* number of rows will be copied into the list.  
115 \*  
116 \* @param resultSet The result set to be wrapped  
117 \* @param limit The maximum for the size of the result.  
118 \*  
119 \* @throws NullPointerException if <code>resultSet</code>  
120 \* is <code>null</code>  
121 \* @throws SQLException if the metadata for this result set  
122 \* cannot be introspected  
123 \*/  
124 public RowSetDynaClass(final ResultSet resultSet, final int limit) throws SQLException {  
125  
126 this(resultSet, true, limit);  
127  
128 }  
129  
130  
131 /\*\*  
132 \* <p>Construct a new {@link RowSetDynaClass} for the specified  
133 \* <code>ResultSet</code>. The property names corresponding  
134 \* to the column names in the result set will be lower cased or not,  
135 \* depending on the specified <code>lowerCase</code> value.</p>  
136 \*  
137 \* If <code>limit</code> is not less than 0, max <code>limit</code>  
138 \* number of rows will be copied into the resultset.  
139 \*  
140 \*  
141 \* @param resultSet The result set to be wrapped  
142 \* @param lowerCase Should property names be lower cased?  
143 \*  
144 \* @throws NullPointerException if <code>resultSet</code>  
145 \* is <code>null</code>  
146 \* @throws SQLException if the metadata for this result set  
147 \* cannot be introspected  
148 \*/  
149 public RowSetDynaClass(final ResultSet resultSet, final boolean lowerCase)  
150 throws SQLException {  
151 this(resultSet, lowerCase, -1);  
152  
153 }  
154  
155 /\*\*  
156 \* <p>Construct a new {@link RowSetDynaClass} for the specified  
157 \* <code>ResultSet</code>. The property names corresponding  
158 \* to the column names in the result set will be lower cased or not,  
159 \* depending on the specified <code>lowerCase</code> value.</p>  
160 \*  
161 \* <p><strong>WARNING</strong> - If you specify <code>false</code>  
162 \* for <code>lowerCase</code>, the returned property names will  
163 \* exactly match the column names returned by your JDBC driver.  
164 \* Because different drivers might return column names in different  
165 \* cases, the property names seen by your application will vary  
166 \* depending on which JDBC driver you are using.</p>  
167 \*  
168 \* @param resultSet The result set to be wrapped  
169 \* @param lowerCase Should property names be lower cased?  
170 \* @param limit Maximum limit for the <code>List</code> of {@link DynaBean}  
171 \*  
172 \* @throws NullPointerException if <code>resultSet</code>  
173 \* is <code>null</code>  
174 \* @throws SQLException if the metadata for this result set  
175 \* cannot be introspected  
176 \*/  
177 public RowSetDynaClass(final ResultSet resultSet, final boolean lowerCase, final int limit)  
178 throws SQLException {  
179  
180 this(resultSet, lowerCase, limit, false);  
181  
182 }  
183  
184 /\*\*  
185 \* <p>Construct a new {@link RowSetDynaClass} for the specified  
186 \* <code>ResultSet</code>. The property names corresponding  
187 \* to the column names in the result set will be lower cased or not,  
188 \* depending on the specified <code>lowerCase</code> value.</p>  
189 \*  
190 \* <p><strong>WARNING</strong> - If you specify <code>false</code>  
191 \* for <code>lowerCase</code>, the returned property names will  
192 \* exactly match the column names returned by your JDBC driver.  
193 \* Because different drivers might return column names in different  
194 \* cases, the property names seen by your application will vary  
195 \* depending on which JDBC driver you are using.</p>  
196 \*  
197 \* @param resultSet The result set to be wrapped  
198 \* @param lowerCase Should property names be lower cased?  
199 \* @param useColumnLabel true if the column label should be used, otherwise false  
200 \*  
201 \* @throws NullPointerException if <code>resultSet</code>  
202 \* is <code>null</code>  
203 \* @throws SQLException if the metadata for this result set  
204 \* cannot be introspected  
205 \* @since 1.8.3  
206 \*/  
207 public RowSetDynaClass(final ResultSet resultSet, final boolean lowerCase, final boolean useColumnLabel)  
208 throws SQLException {  
209 this(resultSet, lowerCase, -1, useColumnLabel);  
210  
211 }  
212  
213 /\*\*  
214 \* <p>Construct a new {@link RowSetDynaClass} for the specified  
215 \* <code>ResultSet</code>. The property names corresponding  
216 \* to the column names in the result set will be lower cased or not,  
217 \* depending on the specified <code>lowerCase</code> value.</p>  
218 \*  
219 \* <p><strong>WARNING</strong> - If you specify <code>false</code>  
220 \* for <code>lowerCase</code>, the returned property names will  
221 \* exactly match the column names returned by your JDBC driver.  
222 \* Because different drivers might return column names in different  
223 \* cases, the property names seen by your application will vary  
224 \* depending on which JDBC driver you are using.</p>  
225 \*  
226 \* @param resultSet The result set to be wrapped  
227 \* @param lowerCase Should property names be lower cased?  
228 \* @param limit Maximum limit for the <code>List</code> of {@link DynaBean}  
229 \* @param useColumnLabel true if the column label should be used, otherwise false  
230 \*  
231 \* @throws NullPointerException if <code>resultSet</code>  
232 \* is <code>null</code>  
233 \* @throws SQLException if the metadata for this result set  
234 \* cannot be introspected  
235 \* @since 1.8.3  
236 \*/  
237 public RowSetDynaClass(final ResultSet resultSet, final boolean lowerCase, final int limit, final boolean useColumnLabel)  
238 throws SQLException {  
239  
240 if (resultSet == null) {  
241 throw new NullPointerException();  
242 }  
243 this.lowerCase = lowerCase;  
244 this.limit = limit;  
245 setUseColumnLabel(useColumnLabel);  
246 introspect(resultSet);  
247 copy(resultSet);  
248  
249 }  
250  
251 /\*\*  
252 \* <p>Return a <code>List</code> containing the {@link DynaBean}s that  
253 \* represent the contents of each <code>Row</code> from the  
254 \* <code>ResultSet</code> that was the basis of this  
255 \* {@link RowSetDynaClass} instance. These {@link DynaBean}s are  
256 \* disconnected from the database itself, so there is no problem with  
257 \* modifying the contents of the list, or the values of the properties  
258 \* of these {@link DynaBean}s. However, it is the application's  
259 \* responsibility to persist any such changes back to the database,  
260 \* if it so desires.</p>  
261 \*  
262 \* @return A <code>List</code> of {@link DynaBean} instances  
263 \*/  
264 public List<DynaBean> getRows() {  
265  
266 return (this.rows);  
267  
268 }  
269  
270  
271 // ------------------------------------------------------ Protected Methods  
272  
273  
274 /\*\*  
275 \* <p>Copy the column values for each row in the specified  
276 \* <code>ResultSet</code> into a newly created {@link DynaBean}, and add  
277 \* this bean to the list of {@link DynaBean}s that will later by  
278 \* returned by a call to <code>getRows()</code>.</p>  
279 \*  
280 \* @param resultSet The <code>ResultSet</code> whose data is to be  
281 \* copied  
282 \*  
283 \* @throws SQLException if an error is encountered copying the data  
284 \*/  
285 protected void copy(final ResultSet resultSet) throws SQLException {  
286  
287 int cnt = 0;  
288 while (resultSet.next() && (limit < 0 || cnt++ < limit) ) {  
289 final DynaBean bean = createDynaBean();  
290 for (DynaProperty propertie : properties) {  
291 final String name = propertie.getName();  
292 final Object value = getObject(resultSet, name);  
293 bean.set(name, value);  
294 }  
295 rows.add(bean);  
296 }  
297  
298 }  
299  
300  
301 /\*\*  
302 \* <p>Create and return a new {@link DynaBean} instance to be used for  
303 \* representing a row in the underlying result set.</p>  
304 \*  
305 \* @return A new <code>DynaBean</code> instance  
306 \*/  
307 protected DynaBean createDynaBean() {  
308  
309 return (new BasicDynaBean(this));  
310  
311 }  
312  
313  
314}