| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JoinRowSet.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/sql/rowset/Joinable.html)   [**NEXT CLASS**](http://docs.google.com/javax/sql/rowset/Predicate.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sql/rowset/JoinRowSet.html)    [**NO FRAMES**](http://docs.google.com/JoinRowSet.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#1t3h5sf) | DETAIL: [FIELD](#35nkun2) | CONSTR | [METHOD](#1y810tw) |

## **javax.sql.rowset**

Interface JoinRowSet

**All Superinterfaces:** [CachedRowSet](http://docs.google.com/javax/sql/rowset/CachedRowSet.html), [Joinable](http://docs.google.com/javax/sql/rowset/Joinable.html), [ResultSet](http://docs.google.com/java/sql/ResultSet.html), [RowSet](http://docs.google.com/javax/sql/RowSet.html), [WebRowSet](http://docs.google.com/javax/sql/rowset/WebRowSet.html), [Wrapper](http://docs.google.com/java/sql/Wrapper.html)

public interface **JoinRowSet**extends [WebRowSet](http://docs.google.com/javax/sql/rowset/WebRowSet.html)

The JoinRowSet interface provides a mechanism for combining related data from different RowSet objects into one JoinRowSet object, which represents an SQL JOIN. In other words, a JoinRowSet object acts as a container for the data from RowSet objects that form an SQL JOIN relationship.

The Joinable interface provides the methods for setting, retrieving, and unsetting a match column, the basis for establishing an SQL JOIN relationship. The match column may alternatively be set by supplying it to the appropriate version of the JointRowSet method addRowSet.

### 1.0 Overview

Disconnected RowSet objects (CachedRowSet objects and implementations extending the CachedRowSet interface) do not have a standard way to establish an SQL JOIN between RowSet objects without the expensive operation of reconnecting to the data source. The JoinRowSet interface is specifically designed to address this need.

Any RowSet object can be added to a JoinRowSet object to become part of an SQL JOIN relationship. This means that both connected and disconnected RowSet objects can be part of a JOIN. RowSet objects operating in a connected environment (JdbcRowSet objects) are encouraged to use the database to which they are already connected to establish SQL JOIN relationships between tables directly. However, it is possible for a JdbcRowSet object to be added to a JoinRowSet object if necessary.

Any number of RowSet objects can be added to an instance of JoinRowSet provided that they can be related in an SQL JOIN. By definition, the SQL JOIN statement is used to combine the data contained in two or more relational database tables based upon a common attribute. The Joinable interface provides the methods for establishing a common attribute, which is done by setting a *match column*. The match column commonly coincides with the primary key, but there is no requirement that the match column be the same as the primary key. By establishing and then enforcing column matches, a JoinRowSet object establishes JOIN relationships between RowSet objects without the assistance of an available relational database.

The type of JOIN to be established is determined by setting one of the JoinRowSet constants using the method setJoinType. The following SQL JOIN types can be set:

* CROSS\_JOIN
* FULL\_JOIN
* INNER\_JOIN - the default if no JOIN type has been set
* LEFT\_OUTER\_JOIN
* RIGHT\_OUTER\_JOIN

Note that if no type is set, the JOIN will automatically be an inner join. The comments for the fields in the JoinRowSet interface explain these JOIN types, which are standard SQL JOIN types.

### 2.0 Using a JoinRowSet Object for Creating a JOIN

When a JoinRowSet object is created, it is empty. The first RowSet object to be added becomes the basis for the JOIN relationship. Applications must determine which column in each of the RowSet objects to be added to the JoinRowSet object should be the match column. All of the RowSet objects must contain a match column, and the values in each match column must be ones that can be compared to values in the other match columns. The columns do not have to have the same name, though they often do, and they do not have to store the exact same data type as long as the data types can be compared.

A match column can be be set in two ways:

* By calling the Joinable method setMatchColumn  
  This is the only method that can set the match column before a RowSet object is added to a JoinRowSet object. The RowSet object must have implemented the Joinable interface in order to use the method setMatchColumn. Once the match column value has been set, this method can be used to reset the match column at any time.
* By calling one of the versions of the JoinRowSet method addRowSet that takes a column name or number (or an array of column names or numbers)  
  Four of the five addRowSet methods take a match column as a parameter. These four methods set or reset the match column at the time a RowSet object is being added to a JoinRowSet object.

### 3.0 Sample Usage

The following code fragment adds two CachedRowSet objects to a JoinRowSet object. Note that in this example, no SQL JOIN type is set, so the default JOIN type, which is *INNER\_JOIN*, is established.

In the following code fragment, the table EMPLOYEES, whose match column is set to the first column (EMP\_ID), is added to the JoinRowSet object *jrs*. Then the table ESSP\_BONUS\_PLAN, whose match column is likewise the EMP\_ID column, is added. When this second table is added to *jrs*, only the rows in ESSP\_BONUS\_PLAN whose EMP\_ID value matches an EMP\_ID value in the EMPLOYEES table are added. In this case, everyone in the bonus plan is an employee, so all of the rows in the table ESSP\_BONUS\_PLAN are added to the JoinRowSet object. In this example, both CachedRowSet objects being added have implemented the Joinable interface and can therefore call the Joinable method setMatchColumn.

JoinRowSet jrs = new JoinRowSetImpl();  
   
 ResultSet rs1 = stmt.executeQuery("SELECT \* FROM EMPLOYEES");  
 CachedRowSet empl = new CachedRowSetImpl();  
 empl.populate(rs1);  
 empl.setMatchColumn(1);   
 jrs.addRowSet(empl);  
   
 ResultSet rs2 = stmt.executeQuery("SELECT \* FROM ESSP\_BONUS\_PLAN");  
 CachedRowSet bonus = new CachedRowSetImpl();  
 bonus.populate(rs2);  
 bonus.setMatchColumn(1); // EMP\_ID is the first column  
 jrs.addRowSet(bonus);

At this point, *jrs* is an inside JOIN of the two RowSet objects based on their EMP\_ID columns. The application can now browse the combined data as if it were browsing one single RowSet object. Because *jrs* is itself a RowSet object, an application can navigate or modify it using RowSet methods.

jrs.first();  
 int employeeID = jrs.getInt(1);  
 String employeeName = jrs.getString(2);

Note that because the SQL JOIN must be enforced when an application adds a second or subsequent RowSet object, there may be an initial degradation in performance while the JOIN is being performed.

The following code fragment adds an additional CachedRowSet object. In this case, the match column (EMP\_ID) is set when the CachedRowSet object is added to the JoinRowSet object.

ResultSet rs3 = stmt.executeQuery("SELECT \* FROM 401K\_CONTRIB");  
 CachedRowSet fourO1k = new CachedRowSetImpl();  
 four01k.populate(rs3);  
 jrs.addRowSet(four01k, 1);

The JoinRowSet object *jrs* now contains values from all three tables. The data in each row in *four01k* in which the value for the EMP\_ID column matches a value for the EMP\_ID column in *jrs* has been added to *jrs*.

### 4.0 JoinRowSet Methods

The JoinRowSet interface supplies several methods for adding RowSet objects and for getting information about the JoinRowSet object.

* Methods for adding one or more RowSet objects  
  These methods allow an application to add one RowSet object at a time or to add multiple RowSet objects at one time. In either case, the methods may specify the match column for each RowSet object being added.
* Methods for getting information  
  One method retrieves the RowSet objects in the JoinRowSet object, and another method retrieves the RowSet names. A third method retrieves either the SQL WHERE clause used behind the scenes to form the JOIN or a text description of what the WHERE clause does.
* Methods related to the type of JOIN  
  One method sets the JOIN type, and five methods find out whether the JoinRowSet object supports a given type.
* A method to make a separate copy of the JoinRowSet object  
  This method creates a copy that can be persisted to the data source.

| **Field Summary** | |
| --- | --- |
| static int | [**CROSS\_JOIN**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#CROSS_JOIN)            An ANSI-style JOIN providing a cross product of two tables |
| static int | [**FULL\_JOIN**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#FULL_JOIN)            An ANSI-style JOIN providing a a full JOIN. |
| static int | [**INNER\_JOIN**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#INNER_JOIN)            An ANSI-style JOIN providing a inner join between two tables. |
| static int | [**LEFT\_OUTER\_JOIN**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#LEFT_OUTER_JOIN)            An ANSI-style JOIN providing a left outer join between two tables. |
| static int | [**RIGHT\_OUTER\_JOIN**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#RIGHT_OUTER_JOIN)            An ANSI-style JOIN providing a right outer join between two tables. |

| **Fields inherited from interface javax.sql.rowset.**[**WebRowSet**](http://docs.google.com/javax/sql/rowset/WebRowSet.html) |
| --- |
| [PUBLIC\_XML\_SCHEMA](http://docs.google.com/javax/sql/rowset/WebRowSet.html#PUBLIC_XML_SCHEMA), [SCHEMA\_SYSTEM\_ID](http://docs.google.com/javax/sql/rowset/WebRowSet.html#SCHEMA_SYSTEM_ID) |

| **Fields inherited from interface javax.sql.rowset.**[**CachedRowSet**](http://docs.google.com/javax/sql/rowset/CachedRowSet.html) |
| --- |
| [COMMIT\_ON\_ACCEPT\_CHANGES](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#COMMIT_ON_ACCEPT_CHANGES) |

| **Fields inherited from interface java.sql.**[**ResultSet**](http://docs.google.com/java/sql/ResultSet.html) |
| --- |
| [CLOSE\_CURSORS\_AT\_COMMIT](http://docs.google.com/java/sql/ResultSet.html#CLOSE_CURSORS_AT_COMMIT), [CONCUR\_READ\_ONLY](http://docs.google.com/java/sql/ResultSet.html#CONCUR_READ_ONLY), [CONCUR\_UPDATABLE](http://docs.google.com/java/sql/ResultSet.html#CONCUR_UPDATABLE), [FETCH\_FORWARD](http://docs.google.com/java/sql/ResultSet.html#FETCH_FORWARD), [FETCH\_REVERSE](http://docs.google.com/java/sql/ResultSet.html#FETCH_REVERSE), [FETCH\_UNKNOWN](http://docs.google.com/java/sql/ResultSet.html#FETCH_UNKNOWN), [HOLD\_CURSORS\_OVER\_COMMIT](http://docs.google.com/java/sql/ResultSet.html#HOLD_CURSORS_OVER_COMMIT), [TYPE\_FORWARD\_ONLY](http://docs.google.com/java/sql/ResultSet.html#TYPE_FORWARD_ONLY), [TYPE\_SCROLL\_INSENSITIVE](http://docs.google.com/java/sql/ResultSet.html#TYPE_SCROLL_INSENSITIVE), [TYPE\_SCROLL\_SENSITIVE](http://docs.google.com/java/sql/ResultSet.html#TYPE_SCROLL_SENSITIVE) |

| **Method Summary** | |
| --- | --- |
| void | [**addRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#addRowSet(javax.sql.rowset.Joinable))([Joinable](http://docs.google.com/javax/sql/rowset/Joinable.html) rowset)            Adds the given RowSet object to this JoinRowSet object. |
| void | [**addRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#addRowSet(javax.sql.RowSet%5B%5D,%20int%5B%5D))([RowSet](http://docs.google.com/javax/sql/RowSet.html)[] rowset, int[] columnIdx)            Adds one or more RowSet objects contained in the given array of RowSet objects to this JoinRowSet object and sets the match column for each of the RowSet objects to the match columns in the given array of column indexes. |
| void | [**addRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#addRowSet(javax.sql.RowSet%5B%5D,%20java.lang.String%5B%5D))([RowSet](http://docs.google.com/javax/sql/RowSet.html)[] rowset, [String](http://docs.google.com/java/lang/String.html)[] columnName)            Adds one or more RowSet objects contained in the given array of RowSet objects to this JoinRowSet object and sets the match column for each of the RowSet objects to the match columns in the given array of column names. |
| void | [**addRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#addRowSet(javax.sql.RowSet,%20int))([RowSet](http://docs.google.com/javax/sql/RowSet.html) rowset, int columnIdx)            Adds the given RowSet object to this JoinRowSet object and sets the designated column as the match column for the RowSet object. |
| void | [**addRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#addRowSet(javax.sql.RowSet,%20java.lang.String))([RowSet](http://docs.google.com/javax/sql/RowSet.html) rowset, [String](http://docs.google.com/java/lang/String.html) columnName)            Adds *rowset* to this JoinRowSet object and sets the designated column as the match column. |
| int | [**getJoinType**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#getJoinType())()            Returns a int describing the set SQL JOIN type governing this JoinRowSet instance. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getRowSetNames**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#getRowSetNames())()            Returns a String array containing the names of the RowSet objects added to this JoinRowSet object. |
| [Collection](http://docs.google.com/java/util/Collection.html)<?> | [**getRowSets**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#getRowSets())()            Returns a Collection object containing the RowSet objects that have been added to this JoinRowSet object. |
| [String](http://docs.google.com/java/lang/String.html) | [**getWhereClause**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#getWhereClause())()            Return a SQL-like description of the WHERE clause being used in a JoinRowSet object. |
| void | [**setJoinType**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#setJoinType(int))(int joinType)            Allow the application to adjust the type of JOIN imposed on tables contained within the JoinRowSet object instance. |
| boolean | [**supportsCrossJoin**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#supportsCrossJoin())()            Indicates if CROSS\_JOIN is supported by a JoinRowSet implementation |
| boolean | [**supportsFullJoin**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#supportsFullJoin())()            Indicates if FULL\_JOIN is supported by a JoinRowSet implementation |
| boolean | [**supportsInnerJoin**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#supportsInnerJoin())()            Indicates if INNER\_JOIN is supported by a JoinRowSet implementation |
| boolean | [**supportsLeftOuterJoin**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#supportsLeftOuterJoin())()            Indicates if LEFT\_OUTER\_JOIN is supported by a JoinRowSet implementation |
| boolean | [**supportsRightOuterJoin**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#supportsRightOuterJoin())()            Indicates if RIGHT\_OUTER\_JOIN is supported by a JoinRowSet implementation |
| [CachedRowSet](http://docs.google.com/javax/sql/rowset/CachedRowSet.html) | [**toCachedRowSet**](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#toCachedRowSet())()            Creates a new CachedRowSet object containing the data in this JoinRowSet object, which can be saved to a data source using the SyncProvider object for the CachedRowSet object. |

| **Methods inherited from interface javax.sql.rowset.**[**WebRowSet**](http://docs.google.com/javax/sql/rowset/WebRowSet.html) |
| --- |
| [readXml](http://docs.google.com/javax/sql/rowset/WebRowSet.html#readXml(java.io.InputStream)), [readXml](http://docs.google.com/javax/sql/rowset/WebRowSet.html#readXml(java.io.Reader)), [writeXml](http://docs.google.com/javax/sql/rowset/WebRowSet.html#writeXml(java.io.OutputStream)), [writeXml](http://docs.google.com/javax/sql/rowset/WebRowSet.html#writeXml(java.sql.ResultSet,%20java.io.OutputStream)), [writeXml](http://docs.google.com/javax/sql/rowset/WebRowSet.html#writeXml(java.sql.ResultSet,%20java.io.Writer)), [writeXml](http://docs.google.com/javax/sql/rowset/WebRowSet.html#writeXml(java.io.Writer)) |

| **Methods inherited from interface javax.sql.rowset.**[**CachedRowSet**](http://docs.google.com/javax/sql/rowset/CachedRowSet.html) |
| --- |
| [acceptChanges](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#acceptChanges()), [acceptChanges](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#acceptChanges(java.sql.Connection)), [columnUpdated](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#columnUpdated(int)), [columnUpdated](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#columnUpdated(java.lang.String)), [commit](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#commit()), [createCopy](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#createCopy()), [createCopyNoConstraints](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#createCopyNoConstraints()), [createCopySchema](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#createCopySchema()), [createShared](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#createShared()), [execute](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#execute(java.sql.Connection)), [getKeyColumns](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getKeyColumns()), [getOriginal](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getOriginal()), [getOriginalRow](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getOriginalRow()), [getPageSize](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getPageSize()), [getRowSetWarnings](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getRowSetWarnings()), [getShowDeleted](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getShowDeleted()), [getSyncProvider](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getSyncProvider()), [getTableName](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#getTableName()), [nextPage](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#nextPage()), [populate](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#populate(java.sql.ResultSet)), [populate](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#populate(java.sql.ResultSet,%20int)), [previousPage](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#previousPage()), [release](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#release()), [restoreOriginal](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#restoreOriginal()), [rollback](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#rollback()), [rollback](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#rollback(java.sql.Savepoint)), [rowSetPopulated](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#rowSetPopulated(javax.sql.RowSetEvent,%20int)), [setKeyColumns](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setKeyColumns(int%5B%5D)), [setMetaData](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setMetaData(javax.sql.RowSetMetaData)), [setOriginalRow](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setOriginalRow()), [setPageSize](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setPageSize(int)), [setShowDeleted](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setShowDeleted(boolean)), [setSyncProvider](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setSyncProvider(java.lang.String)), [setTableName](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setTableName(java.lang.String)), [size](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#size()), [toCollection](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#toCollection()), [toCollection](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#toCollection(int)), [toCollection](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#toCollection(java.lang.String)), [undoDelete](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#undoDelete()), [undoInsert](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#undoInsert()), [undoUpdate](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#undoUpdate()) |

| **Methods inherited from interface javax.sql.**[**RowSet**](http://docs.google.com/javax/sql/RowSet.html) |
| --- |
| [addRowSetListener](http://docs.google.com/javax/sql/RowSet.html#addRowSetListener(javax.sql.RowSetListener)), [clearParameters](http://docs.google.com/javax/sql/RowSet.html#clearParameters()), [execute](http://docs.google.com/javax/sql/RowSet.html#execute()), [getCommand](http://docs.google.com/javax/sql/RowSet.html#getCommand()), [getDataSourceName](http://docs.google.com/javax/sql/RowSet.html#getDataSourceName()), [getEscapeProcessing](http://docs.google.com/javax/sql/RowSet.html#getEscapeProcessing()), [getMaxFieldSize](http://docs.google.com/javax/sql/RowSet.html#getMaxFieldSize()), [getMaxRows](http://docs.google.com/javax/sql/RowSet.html#getMaxRows()), [getPassword](http://docs.google.com/javax/sql/RowSet.html#getPassword()), [getQueryTimeout](http://docs.google.com/javax/sql/RowSet.html#getQueryTimeout()), [getTransactionIsolation](http://docs.google.com/javax/sql/RowSet.html#getTransactionIsolation()), [getTypeMap](http://docs.google.com/javax/sql/RowSet.html#getTypeMap()), [getUrl](http://docs.google.com/javax/sql/RowSet.html#getUrl()), [getUsername](http://docs.google.com/javax/sql/RowSet.html#getUsername()), [isReadOnly](http://docs.google.com/javax/sql/RowSet.html#isReadOnly()), [removeRowSetListener](http://docs.google.com/javax/sql/RowSet.html#removeRowSetListener(javax.sql.RowSetListener)), [setArray](http://docs.google.com/javax/sql/RowSet.html#setArray(int,%20java.sql.Array)), [setAsciiStream](http://docs.google.com/javax/sql/RowSet.html#setAsciiStream(int,%20java.io.InputStream)), [setAsciiStream](http://docs.google.com/javax/sql/RowSet.html#setAsciiStream(int,%20java.io.InputStream,%20int)), [setAsciiStream](http://docs.google.com/javax/sql/RowSet.html#setAsciiStream(java.lang.String,%20java.io.InputStream)), [setAsciiStream](http://docs.google.com/javax/sql/RowSet.html#setAsciiStream(java.lang.String,%20java.io.InputStream,%20int)), [setBigDecimal](http://docs.google.com/javax/sql/RowSet.html#setBigDecimal(int,%20java.math.BigDecimal)), [setBigDecimal](http://docs.google.com/javax/sql/RowSet.html#setBigDecimal(java.lang.String,%20java.math.BigDecimal)), [setBinaryStream](http://docs.google.com/javax/sql/RowSet.html#setBinaryStream(int,%20java.io.InputStream)), [setBinaryStream](http://docs.google.com/javax/sql/RowSet.html#setBinaryStream(int,%20java.io.InputStream,%20int)), [setBinaryStream](http://docs.google.com/javax/sql/RowSet.html#setBinaryStream(java.lang.String,%20java.io.InputStream)), [setBinaryStream](http://docs.google.com/javax/sql/RowSet.html#setBinaryStream(java.lang.String,%20java.io.InputStream,%20int)), [setBlob](http://docs.google.com/javax/sql/RowSet.html#setBlob(int,%20java.sql.Blob)), [setBlob](http://docs.google.com/javax/sql/RowSet.html#setBlob(int,%20java.io.InputStream)), [setBlob](http://docs.google.com/javax/sql/RowSet.html#setBlob(int,%20java.io.InputStream,%20long)), [setBlob](http://docs.google.com/javax/sql/RowSet.html#setBlob(java.lang.String,%20java.sql.Blob)), [setBlob](http://docs.google.com/javax/sql/RowSet.html#setBlob(java.lang.String,%20java.io.InputStream)), [setBlob](http://docs.google.com/javax/sql/RowSet.html#setBlob(java.lang.String,%20java.io.InputStream,%20long)), [setBoolean](http://docs.google.com/javax/sql/RowSet.html#setBoolean(int,%20boolean)), [setBoolean](http://docs.google.com/javax/sql/RowSet.html#setBoolean(java.lang.String,%20boolean)), [setByte](http://docs.google.com/javax/sql/RowSet.html#setByte(int,%20byte)), [setByte](http://docs.google.com/javax/sql/RowSet.html#setByte(java.lang.String,%20byte)), [setBytes](http://docs.google.com/javax/sql/RowSet.html#setBytes(int,%20byte%5B%5D)), [setBytes](http://docs.google.com/javax/sql/RowSet.html#setBytes(java.lang.String,%20byte%5B%5D)), [setCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setCharacterStream(int,%20java.io.Reader)), [setCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setCharacterStream(int,%20java.io.Reader,%20int)), [setCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setCharacterStream(java.lang.String,%20java.io.Reader)), [setCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setCharacterStream(java.lang.String,%20java.io.Reader,%20int)), [setClob](http://docs.google.com/javax/sql/RowSet.html#setClob(int,%20java.sql.Clob)), [setClob](http://docs.google.com/javax/sql/RowSet.html#setClob(int,%20java.io.Reader)), [setClob](http://docs.google.com/javax/sql/RowSet.html#setClob(int,%20java.io.Reader,%20long)), [setClob](http://docs.google.com/javax/sql/RowSet.html#setClob(java.lang.String,%20java.sql.Clob)), [setClob](http://docs.google.com/javax/sql/RowSet.html#setClob(java.lang.String,%20java.io.Reader)), [setClob](http://docs.google.com/javax/sql/RowSet.html#setClob(java.lang.String,%20java.io.Reader,%20long)), [setCommand](http://docs.google.com/javax/sql/RowSet.html#setCommand(java.lang.String)), [setConcurrency](http://docs.google.com/javax/sql/RowSet.html#setConcurrency(int)), [setDataSourceName](http://docs.google.com/javax/sql/RowSet.html#setDataSourceName(java.lang.String)), [setDate](http://docs.google.com/javax/sql/RowSet.html#setDate(int,%20java.sql.Date)), [setDate](http://docs.google.com/javax/sql/RowSet.html#setDate(int,%20java.sql.Date,%20java.util.Calendar)), [setDate](http://docs.google.com/javax/sql/RowSet.html#setDate(java.lang.String,%20java.sql.Date)), [setDate](http://docs.google.com/javax/sql/RowSet.html#setDate(java.lang.String,%20java.sql.Date,%20java.util.Calendar)), [setDouble](http://docs.google.com/javax/sql/RowSet.html#setDouble(int,%20double)), [setDouble](http://docs.google.com/javax/sql/RowSet.html#setDouble(java.lang.String,%20double)), [setEscapeProcessing](http://docs.google.com/javax/sql/RowSet.html#setEscapeProcessing(boolean)), [setFloat](http://docs.google.com/javax/sql/RowSet.html#setFloat(int,%20float)), [setFloat](http://docs.google.com/javax/sql/RowSet.html#setFloat(java.lang.String,%20float)), [setInt](http://docs.google.com/javax/sql/RowSet.html#setInt(int,%20int)), [setInt](http://docs.google.com/javax/sql/RowSet.html#setInt(java.lang.String,%20int)), [setLong](http://docs.google.com/javax/sql/RowSet.html#setLong(int,%20long)), [setLong](http://docs.google.com/javax/sql/RowSet.html#setLong(java.lang.String,%20long)), [setMaxFieldSize](http://docs.google.com/javax/sql/RowSet.html#setMaxFieldSize(int)), [setMaxRows](http://docs.google.com/javax/sql/RowSet.html#setMaxRows(int)), [setNCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setNCharacterStream(int,%20java.io.Reader)), [setNCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setNCharacterStream(int,%20java.io.Reader,%20long)), [setNCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setNCharacterStream(java.lang.String,%20java.io.Reader)), [setNCharacterStream](http://docs.google.com/javax/sql/RowSet.html#setNCharacterStream(java.lang.String,%20java.io.Reader,%20long)), [setNClob](http://docs.google.com/javax/sql/RowSet.html#setNClob(int,%20java.sql.NClob)), [setNClob](http://docs.google.com/javax/sql/RowSet.html#setNClob(int,%20java.io.Reader)), [setNClob](http://docs.google.com/javax/sql/RowSet.html#setNClob(int,%20java.io.Reader,%20long)), [setNClob](http://docs.google.com/javax/sql/RowSet.html#setNClob(java.lang.String,%20java.sql.NClob)), [setNClob](http://docs.google.com/javax/sql/RowSet.html#setNClob(java.lang.String,%20java.io.Reader)), [setNClob](http://docs.google.com/javax/sql/RowSet.html#setNClob(java.lang.String,%20java.io.Reader,%20long)), [setNString](http://docs.google.com/javax/sql/RowSet.html#setNString(int,%20java.lang.String)), [setNString](http://docs.google.com/javax/sql/RowSet.html#setNString(java.lang.String,%20java.lang.String)), [setNull](http://docs.google.com/javax/sql/RowSet.html#setNull(int,%20int)), [setNull](http://docs.google.com/javax/sql/RowSet.html#setNull(int,%20int,%20java.lang.String)), [setNull](http://docs.google.com/javax/sql/RowSet.html#setNull(java.lang.String,%20int)), [setNull](http://docs.google.com/javax/sql/RowSet.html#setNull(java.lang.String,%20int,%20java.lang.String)), [setObject](http://docs.google.com/javax/sql/RowSet.html#setObject(int,%20java.lang.Object)), [setObject](http://docs.google.com/javax/sql/RowSet.html#setObject(int,%20java.lang.Object,%20int)), [setObject](http://docs.google.com/javax/sql/RowSet.html#setObject(int,%20java.lang.Object,%20int,%20int)), [setObject](http://docs.google.com/javax/sql/RowSet.html#setObject(java.lang.String,%20java.lang.Object)), [setObject](http://docs.google.com/javax/sql/RowSet.html#setObject(java.lang.String,%20java.lang.Object,%20int)), [setObject](http://docs.google.com/javax/sql/RowSet.html#setObject(java.lang.String,%20java.lang.Object,%20int,%20int)), [setPassword](http://docs.google.com/javax/sql/RowSet.html#setPassword(java.lang.String)), [setQueryTimeout](http://docs.google.com/javax/sql/RowSet.html#setQueryTimeout(int)), [setReadOnly](http://docs.google.com/javax/sql/RowSet.html#setReadOnly(boolean)), [setRef](http://docs.google.com/javax/sql/RowSet.html#setRef(int,%20java.sql.Ref)), [setRowId](http://docs.google.com/javax/sql/RowSet.html#setRowId(int,%20java.sql.RowId)), [setRowId](http://docs.google.com/javax/sql/RowSet.html#setRowId(java.lang.String,%20java.sql.RowId)), [setShort](http://docs.google.com/javax/sql/RowSet.html#setShort(int,%20short)), [setShort](http://docs.google.com/javax/sql/RowSet.html#setShort(java.lang.String,%20short)), [setSQLXML](http://docs.google.com/javax/sql/RowSet.html#setSQLXML(int,%20java.sql.SQLXML)), [setSQLXML](http://docs.google.com/javax/sql/RowSet.html#setSQLXML(java.lang.String,%20java.sql.SQLXML)), [setString](http://docs.google.com/javax/sql/RowSet.html#setString(int,%20java.lang.String)), [setString](http://docs.google.com/javax/sql/RowSet.html#setString(java.lang.String,%20java.lang.String)), [setTime](http://docs.google.com/javax/sql/RowSet.html#setTime(int,%20java.sql.Time)), [setTime](http://docs.google.com/javax/sql/RowSet.html#setTime(int,%20java.sql.Time,%20java.util.Calendar)), [setTime](http://docs.google.com/javax/sql/RowSet.html#setTime(java.lang.String,%20java.sql.Time)), [setTime](http://docs.google.com/javax/sql/RowSet.html#setTime(java.lang.String,%20java.sql.Time,%20java.util.Calendar)), [setTimestamp](http://docs.google.com/javax/sql/RowSet.html#setTimestamp(int,%20java.sql.Timestamp)), [setTimestamp](http://docs.google.com/javax/sql/RowSet.html#setTimestamp(int,%20java.sql.Timestamp,%20java.util.Calendar)), [setTimestamp](http://docs.google.com/javax/sql/RowSet.html#setTimestamp(java.lang.String,%20java.sql.Timestamp)), [setTimestamp](http://docs.google.com/javax/sql/RowSet.html#setTimestamp(java.lang.String,%20java.sql.Timestamp,%20java.util.Calendar)), [setTransactionIsolation](http://docs.google.com/javax/sql/RowSet.html#setTransactionIsolation(int)), [setType](http://docs.google.com/javax/sql/RowSet.html#setType(int)), [setTypeMap](http://docs.google.com/javax/sql/RowSet.html#setTypeMap(java.util.Map)), [setURL](http://docs.google.com/javax/sql/RowSet.html#setURL(int,%20java.net.URL)), [setUrl](http://docs.google.com/javax/sql/RowSet.html#setUrl(java.lang.String)), [setUsername](http://docs.google.com/javax/sql/RowSet.html#setUsername(java.lang.String)) |

| **Methods inherited from interface java.sql.**[**ResultSet**](http://docs.google.com/java/sql/ResultSet.html) |
| --- |
| [absolute](http://docs.google.com/java/sql/ResultSet.html#absolute(int)), [afterLast](http://docs.google.com/java/sql/ResultSet.html#afterLast()), [beforeFirst](http://docs.google.com/java/sql/ResultSet.html#beforeFirst()), [cancelRowUpdates](http://docs.google.com/java/sql/ResultSet.html#cancelRowUpdates()), [clearWarnings](http://docs.google.com/java/sql/ResultSet.html#clearWarnings()), [close](http://docs.google.com/java/sql/ResultSet.html#close()), [deleteRow](http://docs.google.com/java/sql/ResultSet.html#deleteRow()), [findColumn](http://docs.google.com/java/sql/ResultSet.html#findColumn(java.lang.String)), [first](http://docs.google.com/java/sql/ResultSet.html#first()), [getArray](http://docs.google.com/java/sql/ResultSet.html#getArray(int)), [getArray](http://docs.google.com/java/sql/ResultSet.html#getArray(java.lang.String)), [getAsciiStream](http://docs.google.com/java/sql/ResultSet.html#getAsciiStream(int)), [getAsciiStream](http://docs.google.com/java/sql/ResultSet.html#getAsciiStream(java.lang.String)), [getBigDecimal](http://docs.google.com/java/sql/ResultSet.html#getBigDecimal(int)), [getBigDecimal](http://docs.google.com/java/sql/ResultSet.html#getBigDecimal(int,%20int)), [getBigDecimal](http://docs.google.com/java/sql/ResultSet.html#getBigDecimal(java.lang.String)), [getBigDecimal](http://docs.google.com/java/sql/ResultSet.html#getBigDecimal(java.lang.String,%20int)), [getBinaryStream](http://docs.google.com/java/sql/ResultSet.html#getBinaryStream(int)), [getBinaryStream](http://docs.google.com/java/sql/ResultSet.html#getBinaryStream(java.lang.String)), [getBlob](http://docs.google.com/java/sql/ResultSet.html#getBlob(int)), [getBlob](http://docs.google.com/java/sql/ResultSet.html#getBlob(java.lang.String)), [getBoolean](http://docs.google.com/java/sql/ResultSet.html#getBoolean(int)), [getBoolean](http://docs.google.com/java/sql/ResultSet.html#getBoolean(java.lang.String)), [getByte](http://docs.google.com/java/sql/ResultSet.html#getByte(int)), [getByte](http://docs.google.com/java/sql/ResultSet.html#getByte(java.lang.String)), [getBytes](http://docs.google.com/java/sql/ResultSet.html#getBytes(int)), [getBytes](http://docs.google.com/java/sql/ResultSet.html#getBytes(java.lang.String)), [getCharacterStream](http://docs.google.com/java/sql/ResultSet.html#getCharacterStream(int)), [getCharacterStream](http://docs.google.com/java/sql/ResultSet.html#getCharacterStream(java.lang.String)), [getClob](http://docs.google.com/java/sql/ResultSet.html#getClob(int)), [getClob](http://docs.google.com/java/sql/ResultSet.html#getClob(java.lang.String)), [getConcurrency](http://docs.google.com/java/sql/ResultSet.html#getConcurrency()), [getCursorName](http://docs.google.com/java/sql/ResultSet.html#getCursorName()), [getDate](http://docs.google.com/java/sql/ResultSet.html#getDate(int)), [getDate](http://docs.google.com/java/sql/ResultSet.html#getDate(int,%20java.util.Calendar)), [getDate](http://docs.google.com/java/sql/ResultSet.html#getDate(java.lang.String)), [getDate](http://docs.google.com/java/sql/ResultSet.html#getDate(java.lang.String,%20java.util.Calendar)), [getDouble](http://docs.google.com/java/sql/ResultSet.html#getDouble(int)), [getDouble](http://docs.google.com/java/sql/ResultSet.html#getDouble(java.lang.String)), [getFetchDirection](http://docs.google.com/java/sql/ResultSet.html#getFetchDirection()), [getFetchSize](http://docs.google.com/java/sql/ResultSet.html#getFetchSize()), [getFloat](http://docs.google.com/java/sql/ResultSet.html#getFloat(int)), [getFloat](http://docs.google.com/java/sql/ResultSet.html#getFloat(java.lang.String)), [getHoldability](http://docs.google.com/java/sql/ResultSet.html#getHoldability()), [getInt](http://docs.google.com/java/sql/ResultSet.html#getInt(int)), [getInt](http://docs.google.com/java/sql/ResultSet.html#getInt(java.lang.String)), [getLong](http://docs.google.com/java/sql/ResultSet.html#getLong(int)), [getLong](http://docs.google.com/java/sql/ResultSet.html#getLong(java.lang.String)), [getMetaData](http://docs.google.com/java/sql/ResultSet.html#getMetaData()), [getNCharacterStream](http://docs.google.com/java/sql/ResultSet.html#getNCharacterStream(int)), [getNCharacterStream](http://docs.google.com/java/sql/ResultSet.html#getNCharacterStream(java.lang.String)), [getNClob](http://docs.google.com/java/sql/ResultSet.html#getNClob(int)), [getNClob](http://docs.google.com/java/sql/ResultSet.html#getNClob(java.lang.String)), [getNString](http://docs.google.com/java/sql/ResultSet.html#getNString(int)), [getNString](http://docs.google.com/java/sql/ResultSet.html#getNString(java.lang.String)), [getObject](http://docs.google.com/java/sql/ResultSet.html#getObject(int)), [getObject](http://docs.google.com/java/sql/ResultSet.html#getObject(int,%20java.util.Map)), [getObject](http://docs.google.com/java/sql/ResultSet.html#getObject(java.lang.String)), [getObject](http://docs.google.com/java/sql/ResultSet.html#getObject(java.lang.String,%20java.util.Map)), [getRef](http://docs.google.com/java/sql/ResultSet.html#getRef(int)), [getRef](http://docs.google.com/java/sql/ResultSet.html#getRef(java.lang.String)), [getRow](http://docs.google.com/java/sql/ResultSet.html#getRow()), [getRowId](http://docs.google.com/java/sql/ResultSet.html#getRowId(int)), [getRowId](http://docs.google.com/java/sql/ResultSet.html#getRowId(java.lang.String)), [getShort](http://docs.google.com/java/sql/ResultSet.html#getShort(int)), [getShort](http://docs.google.com/java/sql/ResultSet.html#getShort(java.lang.String)), [getSQLXML](http://docs.google.com/java/sql/ResultSet.html#getSQLXML(int)), [getSQLXML](http://docs.google.com/java/sql/ResultSet.html#getSQLXML(java.lang.String)), [getStatement](http://docs.google.com/java/sql/ResultSet.html#getStatement()), [getString](http://docs.google.com/java/sql/ResultSet.html#getString(int)), [getString](http://docs.google.com/java/sql/ResultSet.html#getString(java.lang.String)), [getTime](http://docs.google.com/java/sql/ResultSet.html#getTime(int)), [getTime](http://docs.google.com/java/sql/ResultSet.html#getTime(int,%20java.util.Calendar)), [getTime](http://docs.google.com/java/sql/ResultSet.html#getTime(java.lang.String)), [getTime](http://docs.google.com/java/sql/ResultSet.html#getTime(java.lang.String,%20java.util.Calendar)), [getTimestamp](http://docs.google.com/java/sql/ResultSet.html#getTimestamp(int)), [getTimestamp](http://docs.google.com/java/sql/ResultSet.html#getTimestamp(int,%20java.util.Calendar)), [getTimestamp](http://docs.google.com/java/sql/ResultSet.html#getTimestamp(java.lang.String)), [getTimestamp](http://docs.google.com/java/sql/ResultSet.html#getTimestamp(java.lang.String,%20java.util.Calendar)), [getType](http://docs.google.com/java/sql/ResultSet.html#getType()), [getUnicodeStream](http://docs.google.com/java/sql/ResultSet.html#getUnicodeStream(int)), [getUnicodeStream](http://docs.google.com/java/sql/ResultSet.html#getUnicodeStream(java.lang.String)), [getURL](http://docs.google.com/java/sql/ResultSet.html#getURL(int)), [getURL](http://docs.google.com/java/sql/ResultSet.html#getURL(java.lang.String)), [getWarnings](http://docs.google.com/java/sql/ResultSet.html#getWarnings()), [insertRow](http://docs.google.com/java/sql/ResultSet.html#insertRow()), [isAfterLast](http://docs.google.com/java/sql/ResultSet.html#isAfterLast()), [isBeforeFirst](http://docs.google.com/java/sql/ResultSet.html#isBeforeFirst()), [isClosed](http://docs.google.com/java/sql/ResultSet.html#isClosed()), [isFirst](http://docs.google.com/java/sql/ResultSet.html#isFirst()), [isLast](http://docs.google.com/java/sql/ResultSet.html#isLast()), [last](http://docs.google.com/java/sql/ResultSet.html#last()), [moveToCurrentRow](http://docs.google.com/java/sql/ResultSet.html#moveToCurrentRow()), [moveToInsertRow](http://docs.google.com/java/sql/ResultSet.html#moveToInsertRow()), [next](http://docs.google.com/java/sql/ResultSet.html#next()), [previous](http://docs.google.com/java/sql/ResultSet.html#previous()), [refreshRow](http://docs.google.com/java/sql/ResultSet.html#refreshRow()), [relative](http://docs.google.com/java/sql/ResultSet.html#relative(int)), [rowDeleted](http://docs.google.com/java/sql/ResultSet.html#rowDeleted()), [rowInserted](http://docs.google.com/java/sql/ResultSet.html#rowInserted()), [rowUpdated](http://docs.google.com/java/sql/ResultSet.html#rowUpdated()), [setFetchDirection](http://docs.google.com/java/sql/ResultSet.html#setFetchDirection(int)), [setFetchSize](http://docs.google.com/java/sql/ResultSet.html#setFetchSize(int)), [updateArray](http://docs.google.com/java/sql/ResultSet.html#updateArray(int,%20java.sql.Array)), [updateArray](http://docs.google.com/java/sql/ResultSet.html#updateArray(java.lang.String,%20java.sql.Array)), [updateAsciiStream](http://docs.google.com/java/sql/ResultSet.html#updateAsciiStream(int,%20java.io.InputStream)), [updateAsciiStream](http://docs.google.com/java/sql/ResultSet.html#updateAsciiStream(int,%20java.io.InputStream,%20int)), [updateAsciiStream](http://docs.google.com/java/sql/ResultSet.html#updateAsciiStream(int,%20java.io.InputStream,%20long)), [updateAsciiStream](http://docs.google.com/java/sql/ResultSet.html#updateAsciiStream(java.lang.String,%20java.io.InputStream)), [updateAsciiStream](http://docs.google.com/java/sql/ResultSet.html#updateAsciiStream(java.lang.String,%20java.io.InputStream,%20int)), [updateAsciiStream](http://docs.google.com/java/sql/ResultSet.html#updateAsciiStream(java.lang.String,%20java.io.InputStream,%20long)), [updateBigDecimal](http://docs.google.com/java/sql/ResultSet.html#updateBigDecimal(int,%20java.math.BigDecimal)), [updateBigDecimal](http://docs.google.com/java/sql/ResultSet.html#updateBigDecimal(java.lang.String,%20java.math.BigDecimal)), [updateBinaryStream](http://docs.google.com/java/sql/ResultSet.html#updateBinaryStream(int,%20java.io.InputStream)), [updateBinaryStream](http://docs.google.com/java/sql/ResultSet.html#updateBinaryStream(int,%20java.io.InputStream,%20int)), [updateBinaryStream](http://docs.google.com/java/sql/ResultSet.html#updateBinaryStream(int,%20java.io.InputStream,%20long)), [updateBinaryStream](http://docs.google.com/java/sql/ResultSet.html#updateBinaryStream(java.lang.String,%20java.io.InputStream)), [updateBinaryStream](http://docs.google.com/java/sql/ResultSet.html#updateBinaryStream(java.lang.String,%20java.io.InputStream,%20int)), [updateBinaryStream](http://docs.google.com/java/sql/ResultSet.html#updateBinaryStream(java.lang.String,%20java.io.InputStream,%20long)), [updateBlob](http://docs.google.com/java/sql/ResultSet.html#updateBlob(int,%20java.sql.Blob)), [updateBlob](http://docs.google.com/java/sql/ResultSet.html#updateBlob(int,%20java.io.InputStream)), [updateBlob](http://docs.google.com/java/sql/ResultSet.html#updateBlob(int,%20java.io.InputStream,%20long)), [updateBlob](http://docs.google.com/java/sql/ResultSet.html#updateBlob(java.lang.String,%20java.sql.Blob)), [updateBlob](http://docs.google.com/java/sql/ResultSet.html#updateBlob(java.lang.String,%20java.io.InputStream)), [updateBlob](http://docs.google.com/java/sql/ResultSet.html#updateBlob(java.lang.String,%20java.io.InputStream,%20long)), [updateBoolean](http://docs.google.com/java/sql/ResultSet.html#updateBoolean(int,%20boolean)), [updateBoolean](http://docs.google.com/java/sql/ResultSet.html#updateBoolean(java.lang.String,%20boolean)), [updateByte](http://docs.google.com/java/sql/ResultSet.html#updateByte(int,%20byte)), [updateByte](http://docs.google.com/java/sql/ResultSet.html#updateByte(java.lang.String,%20byte)), [updateBytes](http://docs.google.com/java/sql/ResultSet.html#updateBytes(int,%20byte%5B%5D)), [updateBytes](http://docs.google.com/java/sql/ResultSet.html#updateBytes(java.lang.String,%20byte%5B%5D)), [updateCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateCharacterStream(int,%20java.io.Reader)), [updateCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateCharacterStream(int,%20java.io.Reader,%20int)), [updateCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateCharacterStream(int,%20java.io.Reader,%20long)), [updateCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateCharacterStream(java.lang.String,%20java.io.Reader)), [updateCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateCharacterStream(java.lang.String,%20java.io.Reader,%20int)), [updateCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateCharacterStream(java.lang.String,%20java.io.Reader,%20long)), [updateClob](http://docs.google.com/java/sql/ResultSet.html#updateClob(int,%20java.sql.Clob)), [updateClob](http://docs.google.com/java/sql/ResultSet.html#updateClob(int,%20java.io.Reader)), [updateClob](http://docs.google.com/java/sql/ResultSet.html#updateClob(int,%20java.io.Reader,%20long)), [updateClob](http://docs.google.com/java/sql/ResultSet.html#updateClob(java.lang.String,%20java.sql.Clob)), [updateClob](http://docs.google.com/java/sql/ResultSet.html#updateClob(java.lang.String,%20java.io.Reader)), [updateClob](http://docs.google.com/java/sql/ResultSet.html#updateClob(java.lang.String,%20java.io.Reader,%20long)), [updateDate](http://docs.google.com/java/sql/ResultSet.html#updateDate(int,%20java.sql.Date)), [updateDate](http://docs.google.com/java/sql/ResultSet.html#updateDate(java.lang.String,%20java.sql.Date)), [updateDouble](http://docs.google.com/java/sql/ResultSet.html#updateDouble(int,%20double)), [updateDouble](http://docs.google.com/java/sql/ResultSet.html#updateDouble(java.lang.String,%20double)), [updateFloat](http://docs.google.com/java/sql/ResultSet.html#updateFloat(int,%20float)), [updateFloat](http://docs.google.com/java/sql/ResultSet.html#updateFloat(java.lang.String,%20float)), [updateInt](http://docs.google.com/java/sql/ResultSet.html#updateInt(int,%20int)), [updateInt](http://docs.google.com/java/sql/ResultSet.html#updateInt(java.lang.String,%20int)), [updateLong](http://docs.google.com/java/sql/ResultSet.html#updateLong(int,%20long)), [updateLong](http://docs.google.com/java/sql/ResultSet.html#updateLong(java.lang.String,%20long)), [updateNCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateNCharacterStream(int,%20java.io.Reader)), [updateNCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateNCharacterStream(int,%20java.io.Reader,%20long)), [updateNCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateNCharacterStream(java.lang.String,%20java.io.Reader)), [updateNCharacterStream](http://docs.google.com/java/sql/ResultSet.html#updateNCharacterStream(java.lang.String,%20java.io.Reader,%20long)), [updateNClob](http://docs.google.com/java/sql/ResultSet.html#updateNClob(int,%20java.sql.NClob)), [updateNClob](http://docs.google.com/java/sql/ResultSet.html#updateNClob(int,%20java.io.Reader)), [updateNClob](http://docs.google.com/java/sql/ResultSet.html#updateNClob(int,%20java.io.Reader,%20long)), [updateNClob](http://docs.google.com/java/sql/ResultSet.html#updateNClob(java.lang.String,%20java.sql.NClob)), [updateNClob](http://docs.google.com/java/sql/ResultSet.html#updateNClob(java.lang.String,%20java.io.Reader)), [updateNClob](http://docs.google.com/java/sql/ResultSet.html#updateNClob(java.lang.String,%20java.io.Reader,%20long)), [updateNString](http://docs.google.com/java/sql/ResultSet.html#updateNString(int,%20java.lang.String)), [updateNString](http://docs.google.com/java/sql/ResultSet.html#updateNString(java.lang.String,%20java.lang.String)), [updateNull](http://docs.google.com/java/sql/ResultSet.html#updateNull(int)), [updateNull](http://docs.google.com/java/sql/ResultSet.html#updateNull(java.lang.String)), [updateObject](http://docs.google.com/java/sql/ResultSet.html#updateObject(int,%20java.lang.Object)), [updateObject](http://docs.google.com/java/sql/ResultSet.html#updateObject(int,%20java.lang.Object,%20int)), [updateObject](http://docs.google.com/java/sql/ResultSet.html#updateObject(java.lang.String,%20java.lang.Object)), [updateObject](http://docs.google.com/java/sql/ResultSet.html#updateObject(java.lang.String,%20java.lang.Object,%20int)), [updateRef](http://docs.google.com/java/sql/ResultSet.html#updateRef(int,%20java.sql.Ref)), [updateRef](http://docs.google.com/java/sql/ResultSet.html#updateRef(java.lang.String,%20java.sql.Ref)), [updateRow](http://docs.google.com/java/sql/ResultSet.html#updateRow()), [updateRowId](http://docs.google.com/java/sql/ResultSet.html#updateRowId(int,%20java.sql.RowId)), [updateRowId](http://docs.google.com/java/sql/ResultSet.html#updateRowId(java.lang.String,%20java.sql.RowId)), [updateShort](http://docs.google.com/java/sql/ResultSet.html#updateShort(int,%20short)), [updateShort](http://docs.google.com/java/sql/ResultSet.html#updateShort(java.lang.String,%20short)), [updateSQLXML](http://docs.google.com/java/sql/ResultSet.html#updateSQLXML(int,%20java.sql.SQLXML)), [updateSQLXML](http://docs.google.com/java/sql/ResultSet.html#updateSQLXML(java.lang.String,%20java.sql.SQLXML)), [updateString](http://docs.google.com/java/sql/ResultSet.html#updateString(int,%20java.lang.String)), [updateString](http://docs.google.com/java/sql/ResultSet.html#updateString(java.lang.String,%20java.lang.String)), [updateTime](http://docs.google.com/java/sql/ResultSet.html#updateTime(int,%20java.sql.Time)), [updateTime](http://docs.google.com/java/sql/ResultSet.html#updateTime(java.lang.String,%20java.sql.Time)), [updateTimestamp](http://docs.google.com/java/sql/ResultSet.html#updateTimestamp(int,%20java.sql.Timestamp)), [updateTimestamp](http://docs.google.com/java/sql/ResultSet.html#updateTimestamp(java.lang.String,%20java.sql.Timestamp)), [wasNull](http://docs.google.com/java/sql/ResultSet.html#wasNull()) |

| **Methods inherited from interface java.sql.**[**Wrapper**](http://docs.google.com/java/sql/Wrapper.html) |
| --- |
| [isWrapperFor](http://docs.google.com/java/sql/Wrapper.html#isWrapperFor(java.lang.Class)), [unwrap](http://docs.google.com/java/sql/Wrapper.html#unwrap(java.lang.Class)) |

| **Methods inherited from interface javax.sql.rowset.**[**Joinable**](http://docs.google.com/javax/sql/rowset/Joinable.html) |
| --- |
| [getMatchColumnIndexes](http://docs.google.com/javax/sql/rowset/Joinable.html#getMatchColumnIndexes()), [getMatchColumnNames](http://docs.google.com/javax/sql/rowset/Joinable.html#getMatchColumnNames()), [setMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#setMatchColumn(int)), [setMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#setMatchColumn(int%5B%5D)), [setMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#setMatchColumn(java.lang.String)), [setMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#setMatchColumn(java.lang.String%5B%5D)), [unsetMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#unsetMatchColumn(int)), [unsetMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#unsetMatchColumn(int%5B%5D)), [unsetMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#unsetMatchColumn(java.lang.String)), [unsetMatchColumn](http://docs.google.com/javax/sql/rowset/Joinable.html#unsetMatchColumn(java.lang.String%5B%5D)) |

| **Field Detail** |
| --- |

### CROSS\_JOIN

static final int **CROSS\_JOIN**

An ANSI-style JOIN providing a cross product of two tables

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.sql.rowset.JoinRowSet.CROSS_JOIN)

### INNER\_JOIN

static final int **INNER\_JOIN**

An ANSI-style JOIN providing a inner join between two tables. Any unmatched rows in either table of the join should be discarded.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.sql.rowset.JoinRowSet.INNER_JOIN)

### LEFT\_OUTER\_JOIN

static final int **LEFT\_OUTER\_JOIN**

An ANSI-style JOIN providing a left outer join between two tables. In SQL, this is described where all records should be returned from the left side of the JOIN statement.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.sql.rowset.JoinRowSet.LEFT_OUTER_JOIN)

### RIGHT\_OUTER\_JOIN

static final int **RIGHT\_OUTER\_JOIN**

An ANSI-style JOIN providing a right outer join between two tables. In SQL, this is described where all records from the table on the right side of the JOIN statement even if the table on the left has no matching record.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.sql.rowset.JoinRowSet.RIGHT_OUTER_JOIN)

### FULL\_JOIN

static final int **FULL\_JOIN**

An ANSI-style JOIN providing a a full JOIN. Specifies that all rows from either table be returned regardless of matching records on the other table.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#javax.sql.rowset.JoinRowSet.FULL_JOIN)

| **Method Detail** |
| --- |

### addRowSet

void **addRowSet**([Joinable](http://docs.google.com/javax/sql/rowset/Joinable.html) rowset)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Adds the given RowSet object to this JoinRowSet object. If the RowSet object is the first to be added to this JoinRowSet object, it forms the basis of the JOIN relationship to be established.

This method should be used only when the given RowSet object already has a match column that was set with the Joinable method setMatchColumn.

Note: A Joinable object is any RowSet object that has implemented the Joinable interface.

**Parameters:**rowset - the RowSet object that is to be added to this JoinRowSet object; it must implement the Joinable interface and have a match column set **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if (1) an empty rowset is added to the to this JoinRowSet object, (2) a match column has not been set for *rowset*, or (3) *rowset* violates the active JOIN**See Also:**[Joinable.setMatchColumn(int)](http://docs.google.com/javax/sql/rowset/Joinable.html#setMatchColumn(int))

### addRowSet

void **addRowSet**([RowSet](http://docs.google.com/javax/sql/RowSet.html) rowset,  
 int columnIdx)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Adds the given RowSet object to this JoinRowSet object and sets the designated column as the match column for the RowSet object. If the RowSet object is the first to be added to this JoinRowSet object, it forms the basis of the JOIN relationship to be established.

This method should be used when *RowSet* does not already have a match column set.

**Parameters:**rowset - the RowSet object that is to be added to this JoinRowSet object; it may implement the Joinable interfacecolumnIdx - an int that identifies the column to become the match column **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if (1) *rowset* is an empty rowset or (2) *rowset* violates the active JOIN**See Also:**[Joinable.unsetMatchColumn(int)](http://docs.google.com/javax/sql/rowset/Joinable.html#unsetMatchColumn(int))

### addRowSet

void **addRowSet**([RowSet](http://docs.google.com/javax/sql/RowSet.html) rowset,  
 [String](http://docs.google.com/java/lang/String.html) columnName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Adds *rowset* to this JoinRowSet object and sets the designated column as the match column. If *rowset* is the first to be added to this JoinRowSet object, it forms the basis for the JOIN relationship to be established.

This method should be used when the given RowSet object does not already have a match column.

**Parameters:**rowset - the RowSet object that is to be added to this JoinRowSet object; it may implement the Joinable interfacecolumnName - the String object giving the name of the column to be set as the match column **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if (1) *rowset* is an empty rowset or (2) the match column for *rowset* does not satisfy the conditions of the JOIN

### addRowSet

void **addRowSet**([RowSet](http://docs.google.com/javax/sql/RowSet.html)[] rowset,  
 int[] columnIdx)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Adds one or more RowSet objects contained in the given array of RowSet objects to this JoinRowSet object and sets the match column for each of the RowSet objects to the match columns in the given array of column indexes. The first element in *columnIdx* is set as the match column for the first RowSet object in *rowset*, the second element of *columnIdx* is set as the match column for the second element in *rowset*, and so on.

The first RowSet object added to this JoinRowSet object forms the basis for the JOIN relationship.

This method should be used when the given RowSet object does not already have a match column.

**Parameters:**rowset - an array of one or more RowSet objects to be added to the JOIN; it may implement the Joinable interfacecolumnIdx - an array of int values indicating the index(es) of the columns to be set as the match columns for the RowSet objects in *rowset* **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if (1) an empty rowset is added to this JoinRowSet object, (2) a match column is not set for a RowSet object in *rowset*, or (3) a RowSet object being added violates the active JOIN

### addRowSet

void **addRowSet**([RowSet](http://docs.google.com/javax/sql/RowSet.html)[] rowset,  
 [String](http://docs.google.com/java/lang/String.html)[] columnName)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Adds one or more RowSet objects contained in the given array of RowSet objects to this JoinRowSet object and sets the match column for each of the RowSet objects to the match columns in the given array of column names. The first element in *columnName* is set as the match column for the first RowSet object in *rowset*, the second element of *columnName* is set as the match column for the second element in *rowset*, and so on.

The first RowSet object added to this JoinRowSet object forms the basis for the JOIN relationship.

This method should be used when the given RowSet object(s) does not already have a match column.

**Parameters:**rowset - an array of one or more RowSet objects to be added to the JOIN; it may implement the Joinable interfacecolumnName - an array of String values indicating the names of the columns to be set as the match columns for the RowSet objects in *rowset* **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if (1) an empty rowset is added to this JoinRowSet object, (2) a match column is not set for a RowSet object in *rowset*, or (3) a RowSet object being added violates the active JOIN

### getRowSets

[Collection](http://docs.google.com/java/util/Collection.html)<?> **getRowSets**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns a Collection object containing the RowSet objects that have been added to this JoinRowSet object. This should return the 'n' number of RowSet contained within the JOIN and maintain any updates that have occured while in this union.

**Returns:**a Collection object consisting of the RowSet objects added to this JoinRowSet object **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if an error occurs generating the Collection object to be returned

### getRowSetNames

[String](http://docs.google.com/java/lang/String.html)[] **getRowSetNames**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns a String array containing the names of the RowSet objects added to this JoinRowSet object.

**Returns:**a String array of the names of the RowSet objects in this JoinRowSet object **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if an error occurs retrieving the names of the RowSet objects**See Also:**[CachedRowSet.setTableName(java.lang.String)](http://docs.google.com/javax/sql/rowset/CachedRowSet.html#setTableName(java.lang.String))

### toCachedRowSet

[CachedRowSet](http://docs.google.com/javax/sql/rowset/CachedRowSet.html) **toCachedRowSet**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Creates a new CachedRowSet object containing the data in this JoinRowSet object, which can be saved to a data source using the SyncProvider object for the CachedRowSet object.

If any updates or modifications have been applied to the JoinRowSet the CachedRowSet returned by the method will not be able to persist it's changes back to the originating rows and tables in the in the datasource. The CachedRowSet instance returned should not contain modification data and it should clear all properties of it's originating SQL statement. An application should reset the SQL statement using the RowSet.setCommand method.

In order to allow changes to be persisted back to the datasource to the originating tables, the acceptChanges method should be used and called on a JoinRowSet object instance. Implementations can leverage the internal data and update tracking in their implementations to interact with the SyncProvider to persist any changes.

**Returns:**a CachedRowSet containing the contents of the JoinRowSet **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if an error occurs assembling the CachedRowSet object**See Also:**[RowSet](http://docs.google.com/javax/sql/RowSet.html), [CachedRowSet](http://docs.google.com/javax/sql/rowset/CachedRowSet.html), [SyncProvider](http://docs.google.com/javax/sql/rowset/spi/SyncProvider.html)

### supportsCrossJoin

boolean **supportsCrossJoin**()

Indicates if CROSS\_JOIN is supported by a JoinRowSet implementation

**Returns:**true if the CROSS\_JOIN is supported; false otherwise

### supportsInnerJoin

boolean **supportsInnerJoin**()

Indicates if INNER\_JOIN is supported by a JoinRowSet implementation

**Returns:**true is the INNER\_JOIN is supported; false otherwise

### supportsLeftOuterJoin

boolean **supportsLeftOuterJoin**()

Indicates if LEFT\_OUTER\_JOIN is supported by a JoinRowSet implementation

**Returns:**true is the LEFT\_OUTER\_JOIN is supported; false otherwise

### supportsRightOuterJoin

boolean **supportsRightOuterJoin**()

Indicates if RIGHT\_OUTER\_JOIN is supported by a JoinRowSet implementation

**Returns:**true is the RIGHT\_OUTER\_JOIN is supported; false otherwise

### supportsFullJoin

boolean **supportsFullJoin**()

Indicates if FULL\_JOIN is supported by a JoinRowSet implementation

**Returns:**true is the FULL\_JOIN is supported; false otherwise

### setJoinType

void **setJoinType**(int joinType)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Allow the application to adjust the type of JOIN imposed on tables contained within the JoinRowSet object instance. Implementations should throw a SQLException if they do not support a given JOIN type.

**Parameters:**joinType - the standard JoinRowSet.XXX static field definition of a SQL JOIN to re-configure a JoinRowSet instance on the fly. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if an unsupported JOIN type is set**See Also:**[getJoinType()](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#getJoinType())

### getWhereClause

[String](http://docs.google.com/java/lang/String.html) **getWhereClause**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Return a SQL-like description of the WHERE clause being used in a JoinRowSet object. An implementation can describe the WHERE clause of the SQL JOIN by supplying a SQL strings description of JOIN or provide a textual description to assist applications using a JoinRowSet

**Returns:**whereClause a textual or SQL description of the logical WHERE clause used in the JoinRowSet instance **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if an error occurs in generating a representation of the WHERE clause.

### getJoinType

int **getJoinType**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Returns a int describing the set SQL JOIN type governing this JoinRowSet instance. The returned type will be one of standard JoinRowSet types: CROSS\_JOIN, INNER\_JOIN, LEFT\_OUTER\_JOIN, RIGHT\_OUTER\_JOIN or FULL\_JOIN.

**Returns:**joinType one of the standard JoinRowSet static field definitions of a SQL JOIN. JoinRowSet.INNER\_JOIN is returned as the default JOIN type is no type has been explicitly set. **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if an error occurs determining the SQL JOIN type supported by the JoinRowSet instance.**See Also:**[setJoinType(int)](http://docs.google.com/javax/sql/rowset/JoinRowSet.html#setJoinType(int))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/JoinRowSet.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/sql/rowset/Joinable.html)   [**NEXT CLASS**](http://docs.google.com/javax/sql/rowset/Predicate.html) | [**FRAMES**](http://docs.google.com/index.html?javax/sql/rowset/JoinRowSet.html)    [**NO FRAMES**](http://docs.google.com/JoinRowSet.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#1t3h5sf) | DETAIL: [FIELD](#35nkun2) | CONSTR | [METHOD](#1y810tw) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).