| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DatabaseMetaData.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/java/sql/Connection.html)   [**NEXT CLASS**](http://docs.google.com/java/sql/DataTruncation.html) | [**FRAMES**](http://docs.google.com/index.html?java/sql/DatabaseMetaData.html)    [**NO FRAMES**](http://docs.google.com/DatabaseMetaData.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#3dy6vkm) | CONSTR | [METHOD](#kgcv8k) |

## **java.sql**

Interface DatabaseMetaData

**All Superinterfaces:** [Wrapper](http://docs.google.com/java/sql/Wrapper.html)

public interface **DatabaseMetaData**extends [Wrapper](http://docs.google.com/java/sql/Wrapper.html)

Comprehensive information about the database as a whole.

This interface is implemented by driver vendors to let users know the capabilities of a Database Management System (DBMS) in combination with the driver based on JDBCTM technology ("JDBC driver") that is used with it. Different relational DBMSs often support different features, implement features in different ways, and use different data types. In addition, a driver may implement a feature on top of what the DBMS offers. Information returned by methods in this interface applies to the capabilities of a particular driver and a particular DBMS working together. Note that as used in this documentation, the term "database" is used generically to refer to both the driver and DBMS.

A user for this interface is commonly a tool that needs to discover how to deal with the underlying DBMS. This is especially true for applications that are intended to be used with more than one DBMS. For example, a tool might use the method getTypeInfo to find out what data types can be used in a CREATE TABLE statement. Or a user might call the method supportsCorrelatedSubqueries to see if it is possible to use a correlated subquery or supportsBatchUpdates to see if it is possible to use batch updates.

Some DatabaseMetaData methods return lists of information in the form of ResultSet objects. Regular ResultSet methods, such as getString and getInt, can be used to retrieve the data from these ResultSet objects. If a given form of metadata is not available, an empty ResultSet will be returned. Additional columns beyond the columns defined to be returned by the ResultSet object for a given method can be defined by the JDBC driver vendor and must be accessed by their **column label**.

Some DatabaseMetaData methods take arguments that are String patterns. These arguments all have names such as fooPattern. Within a pattern String, "%" means match any substring of 0 or more characters, and "\_" means match any one character. Only metadata entries matching the search pattern are returned. If a search pattern argument is set to null, that argument's criterion will be dropped from the search.

| **Field Summary** | |
| --- | --- |
| static short | [**attributeNoNulls**](http://docs.google.com/java/sql/DatabaseMetaData.html#attributeNoNulls)            Indicates that NULL values might not be allowed. |
| static short | [**attributeNullable**](http://docs.google.com/java/sql/DatabaseMetaData.html#attributeNullable)            Indicates that NULL values are definitely allowed. |
| static short | [**attributeNullableUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#attributeNullableUnknown)            Indicates that whether NULL values are allowed is not known. |
| static int | [**bestRowNotPseudo**](http://docs.google.com/java/sql/DatabaseMetaData.html#bestRowNotPseudo)            Indicates that the best row identifier is NOT a pseudo column. |
| static int | [**bestRowPseudo**](http://docs.google.com/java/sql/DatabaseMetaData.html#bestRowPseudo)            Indicates that the best row identifier is a pseudo column. |
| static int | [**bestRowSession**](http://docs.google.com/java/sql/DatabaseMetaData.html#bestRowSession)            Indicates that the scope of the best row identifier is the remainder of the current session. |
| static int | [**bestRowTemporary**](http://docs.google.com/java/sql/DatabaseMetaData.html#bestRowTemporary)            Indicates that the scope of the best row identifier is very temporary, lasting only while the row is being used. |
| static int | [**bestRowTransaction**](http://docs.google.com/java/sql/DatabaseMetaData.html#bestRowTransaction)            Indicates that the scope of the best row identifier is the remainder of the current transaction. |
| static int | [**bestRowUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#bestRowUnknown)            Indicates that the best row identifier may or may not be a pseudo column. |
| static int | [**columnNoNulls**](http://docs.google.com/java/sql/DatabaseMetaData.html#columnNoNulls)            Indicates that the column might not allow NULL values. |
| static int | [**columnNullable**](http://docs.google.com/java/sql/DatabaseMetaData.html#columnNullable)            Indicates that the column definitely allows NULL values. |
| static int | [**columnNullableUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#columnNullableUnknown)            Indicates that the nullability of columns is unknown. |
| static int | [**functionColumnIn**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionColumnIn)            Indicates that the parameter or column is an IN parameter. |
| static int | [**functionColumnInOut**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionColumnInOut)            Indicates that the parameter or column is an INOUT parameter. |
| static int | [**functionColumnOut**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionColumnOut)            Indicates that the parameter or column is an OUT parameter. |
| static int | [**functionColumnResult**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionColumnResult)            Indicates that the parameter or column is a column in a result set. |
| static int | [**functionColumnUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionColumnUnknown)            Indicates that type of the parameter or column is unknown. |
| static int | [**functionNoNulls**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionNoNulls)            Indicates that NULL values are not allowed. |
| static int | [**functionNoTable**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionNoTable)            Indicates that the function does not return a table. |
| static int | [**functionNullable**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionNullable)            Indicates that NULL values are allowed. |
| static int | [**functionNullableUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionNullableUnknown)            Indicates that whether NULL values are allowed is unknown. |
| static int | [**functionResultUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionResultUnknown)            Indicates that it is not known whether the function returns a result or a table. |
| static int | [**functionReturn**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionReturn)            Indicates that the parameter or column is a return value. |
| static int | [**functionReturnsTable**](http://docs.google.com/java/sql/DatabaseMetaData.html#functionReturnsTable)            Indicates that the function returns a table. |
| static int | [**importedKeyCascade**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeyCascade)            For the column UPDATE\_RULE, indicates that when the primary key is updated, the foreign key (imported key) is changed to agree with it. |
| static int | [**importedKeyInitiallyDeferred**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeyInitiallyDeferred)            Indicates deferrability. |
| static int | [**importedKeyInitiallyImmediate**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeyInitiallyImmediate)            Indicates deferrability. |
| static int | [**importedKeyNoAction**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeyNoAction)            For the columns UPDATE\_RULE and DELETE\_RULE, indicates that if the primary key has been imported, it cannot be updated or deleted. |
| static int | [**importedKeyNotDeferrable**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeyNotDeferrable)            Indicates deferrability. |
| static int | [**importedKeyRestrict**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeyRestrict)            For the column UPDATE\_RULE, indicates that a primary key may not be updated if it has been imported by another table as a foreign key. |
| static int | [**importedKeySetDefault**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeySetDefault)            For the columns UPDATE\_RULE and DELETE\_RULE, indicates that if the primary key is updated or deleted, the foreign key (imported key) is set to the default value. |
| static int | [**importedKeySetNull**](http://docs.google.com/java/sql/DatabaseMetaData.html#importedKeySetNull)            For the columns UPDATE\_RULE and DELETE\_RULE, indicates that when the primary key is updated or deleted, the foreign key (imported key) is changed to NULL. |
| static int | [**procedureColumnIn**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureColumnIn)            Indicates that the column stores IN parameters. |
| static int | [**procedureColumnInOut**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureColumnInOut)            Indicates that the column stores INOUT parameters. |
| static int | [**procedureColumnOut**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureColumnOut)            Indicates that the column stores OUT parameters. |
| static int | [**procedureColumnResult**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureColumnResult)            Indicates that the column stores results. |
| static int | [**procedureColumnReturn**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureColumnReturn)            Indicates that the column stores return values. |
| static int | [**procedureColumnUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureColumnUnknown)            Indicates that type of the column is unknown. |
| static int | [**procedureNoNulls**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureNoNulls)            Indicates that NULL values are not allowed. |
| static int | [**procedureNoResult**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureNoResult)            Indicates that the procedure does not return a result. |
| static int | [**procedureNullable**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureNullable)            Indicates that NULL values are allowed. |
| static int | [**procedureNullableUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureNullableUnknown)            Indicates that whether NULL values are allowed is unknown. |
| static int | [**procedureResultUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureResultUnknown)            Indicates that it is not known whether the procedure returns a result. |
| static int | [**procedureReturnsResult**](http://docs.google.com/java/sql/DatabaseMetaData.html#procedureReturnsResult)            Indicates that the procedure returns a result. |
| static int | [**sqlStateSQL**](http://docs.google.com/java/sql/DatabaseMetaData.html#sqlStateSQL)            A possible return value for the method DatabaseMetaData.getSQLStateType which is used to indicate whether the value returned by the method SQLException.getSQLState is an SQLSTATE value. |
| static int | [**sqlStateSQL99**](http://docs.google.com/java/sql/DatabaseMetaData.html#sqlStateSQL99)            A possible return value for the method DatabaseMetaData.getSQLStateType which is used to indicate whether the value returned by the method SQLException.getSQLState is an SQL99 SQLSTATE value. |
| static int | [**sqlStateXOpen**](http://docs.google.com/java/sql/DatabaseMetaData.html#sqlStateXOpen)            A possible return value for the method DatabaseMetaData.getSQLStateType which is used to indicate whether the value returned by the method SQLException.getSQLState is an X/Open (now know as Open Group) SQL CLI SQLSTATE value. |
| static short | [**tableIndexClustered**](http://docs.google.com/java/sql/DatabaseMetaData.html#tableIndexClustered)            Indicates that this table index is a clustered index. |
| static short | [**tableIndexHashed**](http://docs.google.com/java/sql/DatabaseMetaData.html#tableIndexHashed)            Indicates that this table index is a hashed index. |
| static short | [**tableIndexOther**](http://docs.google.com/java/sql/DatabaseMetaData.html#tableIndexOther)            Indicates that this table index is not a clustered index, a hashed index, or table statistics; it is something other than these. |
| static short | [**tableIndexStatistic**](http://docs.google.com/java/sql/DatabaseMetaData.html#tableIndexStatistic)            Indicates that this column contains table statistics that are returned in conjunction with a table's index descriptions. |
| static int | [**typeNoNulls**](http://docs.google.com/java/sql/DatabaseMetaData.html#typeNoNulls)            Indicates that a NULL value is NOT allowed for this data type. |
| static int | [**typeNullable**](http://docs.google.com/java/sql/DatabaseMetaData.html#typeNullable)            Indicates that a NULL value is allowed for this data type. |
| static int | [**typeNullableUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#typeNullableUnknown)            Indicates that it is not known whether a NULL value is allowed for this data type. |
| static int | [**typePredBasic**](http://docs.google.com/java/sql/DatabaseMetaData.html#typePredBasic)            Indicates that the data type can be only be used in WHERE search clauses that do not use LIKE predicates. |
| static int | [**typePredChar**](http://docs.google.com/java/sql/DatabaseMetaData.html#typePredChar)            Indicates that the data type can be only be used in WHERE search clauses that use LIKE predicates. |
| static int | [**typePredNone**](http://docs.google.com/java/sql/DatabaseMetaData.html#typePredNone)            Indicates that WHERE search clauses are not supported for this type. |
| static int | [**typeSearchable**](http://docs.google.com/java/sql/DatabaseMetaData.html#typeSearchable)            Indicates that all WHERE search clauses can be based on this type. |
| static int | [**versionColumnNotPseudo**](http://docs.google.com/java/sql/DatabaseMetaData.html#versionColumnNotPseudo)            Indicates that this version column is NOT a pseudo column. |
| static int | [**versionColumnPseudo**](http://docs.google.com/java/sql/DatabaseMetaData.html#versionColumnPseudo)            Indicates that this version column is a pseudo column. |
| static int | [**versionColumnUnknown**](http://docs.google.com/java/sql/DatabaseMetaData.html#versionColumnUnknown)            Indicates that this version column may or may not be a pseudo column. |

| **Method Summary** | |
| --- | --- |
| boolean | [**allProceduresAreCallable**](http://docs.google.com/java/sql/DatabaseMetaData.html#allProceduresAreCallable())()            Retrieves whether the current user can call all the procedures returned by the method getProcedures. |
| boolean | [**allTablesAreSelectable**](http://docs.google.com/java/sql/DatabaseMetaData.html#allTablesAreSelectable())()            Retrieves whether the current user can use all the tables returned by the method getTables in a SELECT statement. |
| boolean | [**autoCommitFailureClosesAllResultSets**](http://docs.google.com/java/sql/DatabaseMetaData.html#autoCommitFailureClosesAllResultSets())()            Retrieves whether a SQLException while autoCommit is true inidcates that all open ResultSets are closed, even ones that are holdable. |
| boolean | [**dataDefinitionCausesTransactionCommit**](http://docs.google.com/java/sql/DatabaseMetaData.html#dataDefinitionCausesTransactionCommit())()            Retrieves whether a data definition statement within a transaction forces the transaction to commit. |
| boolean | [**dataDefinitionIgnoredInTransactions**](http://docs.google.com/java/sql/DatabaseMetaData.html#dataDefinitionIgnoredInTransactions())()            Retrieves whether this database ignores a data definition statement within a transaction. |
| boolean | [**deletesAreDetected**](http://docs.google.com/java/sql/DatabaseMetaData.html#deletesAreDetected(int))(int type)            Retrieves whether or not a visible row delete can be detected by calling the method ResultSet.rowDeleted. |
| boolean | [**doesMaxRowSizeIncludeBlobs**](http://docs.google.com/java/sql/DatabaseMetaData.html#doesMaxRowSizeIncludeBlobs())()            Retrieves whether the return value for the method getMaxRowSize includes the SQL data types LONGVARCHAR and LONGVARBINARY. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getAttributes**](http://docs.google.com/java/sql/DatabaseMetaData.html#getAttributes(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) typeNamePattern, [String](http://docs.google.com/java/lang/String.html) attributeNamePattern)            Retrieves a description of the given attribute of the given type for a user-defined type (UDT) that is available in the given schema and catalog. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getBestRowIdentifier**](http://docs.google.com/java/sql/DatabaseMetaData.html#getBestRowIdentifier(java.lang.String,%20java.lang.String,%20java.lang.String,%20int,%20boolean))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table, int scope, boolean nullable)            Retrieves a description of a table's optimal set of columns that uniquely identifies a row. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getCatalogs**](http://docs.google.com/java/sql/DatabaseMetaData.html#getCatalogs())()            Retrieves the catalog names available in this database. |
| [String](http://docs.google.com/java/lang/String.html) | [**getCatalogSeparator**](http://docs.google.com/java/sql/DatabaseMetaData.html#getCatalogSeparator())()            Retrieves the String that this database uses as the separator between a catalog and table name. |
| [String](http://docs.google.com/java/lang/String.html) | [**getCatalogTerm**](http://docs.google.com/java/sql/DatabaseMetaData.html#getCatalogTerm())()            Retrieves the database vendor's preferred term for "catalog". |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getClientInfoProperties**](http://docs.google.com/java/sql/DatabaseMetaData.html#getClientInfoProperties())()            Retrieves a list of the client info properties that the driver supports. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getColumnPrivileges**](http://docs.google.com/java/sql/DatabaseMetaData.html#getColumnPrivileges(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table, [String](http://docs.google.com/java/lang/String.html) columnNamePattern)            Retrieves a description of the access rights for a table's columns. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getColumns**](http://docs.google.com/java/sql/DatabaseMetaData.html#getColumns(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) tableNamePattern, [String](http://docs.google.com/java/lang/String.html) columnNamePattern)            Retrieves a description of table columns available in the specified catalog. |
| [Connection](http://docs.google.com/java/sql/Connection.html) | [**getConnection**](http://docs.google.com/java/sql/DatabaseMetaData.html#getConnection())()            Retrieves the connection that produced this metadata object. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getCrossReference**](http://docs.google.com/java/sql/DatabaseMetaData.html#getCrossReference(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) parentCatalog, [String](http://docs.google.com/java/lang/String.html) parentSchema, [String](http://docs.google.com/java/lang/String.html) parentTable, [String](http://docs.google.com/java/lang/String.html) foreignCatalog, [String](http://docs.google.com/java/lang/String.html) foreignSchema, [String](http://docs.google.com/java/lang/String.html) foreignTable)            Retrieves a description of the foreign key columns in the given foreign key table that reference the primary key or the columns representing a unique constraint of the parent table (could be the same or a different table). |
| int | [**getDatabaseMajorVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDatabaseMajorVersion())()            Retrieves the major version number of the underlying database. |
| int | [**getDatabaseMinorVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDatabaseMinorVersion())()            Retrieves the minor version number of the underlying database. |
| [String](http://docs.google.com/java/lang/String.html) | [**getDatabaseProductName**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDatabaseProductName())()            Retrieves the name of this database product. |
| [String](http://docs.google.com/java/lang/String.html) | [**getDatabaseProductVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDatabaseProductVersion())()            Retrieves the version number of this database product. |
| int | [**getDefaultTransactionIsolation**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDefaultTransactionIsolation())()            Retrieves this database's default transaction isolation level. |
| int | [**getDriverMajorVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDriverMajorVersion())()            Retrieves this JDBC driver's major version number. |
| int | [**getDriverMinorVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDriverMinorVersion())()            Retrieves this JDBC driver's minor version number. |
| [String](http://docs.google.com/java/lang/String.html) | [**getDriverName**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDriverName())()            Retrieves the name of this JDBC driver. |
| [String](http://docs.google.com/java/lang/String.html) | [**getDriverVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getDriverVersion())()            Retrieves the version number of this JDBC driver as a String. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getExportedKeys**](http://docs.google.com/java/sql/DatabaseMetaData.html#getExportedKeys(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table)            Retrieves a description of the foreign key columns that reference the given table's primary key columns (the foreign keys exported by a table). |
| [String](http://docs.google.com/java/lang/String.html) | [**getExtraNameCharacters**](http://docs.google.com/java/sql/DatabaseMetaData.html#getExtraNameCharacters())()            Retrieves all the "extra" characters that can be used in unquoted identifier names (those beyond a-z, A-Z, 0-9 and \_). |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getFunctionColumns**](http://docs.google.com/java/sql/DatabaseMetaData.html#getFunctionColumns(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) functionNamePattern, [String](http://docs.google.com/java/lang/String.html) columnNamePattern)            Retrieves a description of the given catalog's system or user function parameters and return type. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getFunctions**](http://docs.google.com/java/sql/DatabaseMetaData.html#getFunctions(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) functionNamePattern)            Retrieves a description of the system and user functions available in the given catalog. |
| [String](http://docs.google.com/java/lang/String.html) | [**getIdentifierQuoteString**](http://docs.google.com/java/sql/DatabaseMetaData.html#getIdentifierQuoteString())()            Retrieves the string used to quote SQL identifiers. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getImportedKeys**](http://docs.google.com/java/sql/DatabaseMetaData.html#getImportedKeys(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table)            Retrieves a description of the primary key columns that are referenced by the given table's foreign key columns (the primary keys imported by a table). |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getIndexInfo**](http://docs.google.com/java/sql/DatabaseMetaData.html#getIndexInfo(java.lang.String,%20java.lang.String,%20java.lang.String,%20boolean,%20boolean))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table, boolean unique, boolean approximate)            Retrieves a description of the given table's indices and statistics. |
| int | [**getJDBCMajorVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getJDBCMajorVersion())()            Retrieves the major JDBC version number for this driver. |
| int | [**getJDBCMinorVersion**](http://docs.google.com/java/sql/DatabaseMetaData.html#getJDBCMinorVersion())()            Retrieves the minor JDBC version number for this driver. |
| int | [**getMaxBinaryLiteralLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxBinaryLiteralLength())()            Retrieves the maximum number of hex characters this database allows in an inline binary literal. |
| int | [**getMaxCatalogNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxCatalogNameLength())()            Retrieves the maximum number of characters that this database allows in a catalog name. |
| int | [**getMaxCharLiteralLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxCharLiteralLength())()            Retrieves the maximum number of characters this database allows for a character literal. |
| int | [**getMaxColumnNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxColumnNameLength())()            Retrieves the maximum number of characters this database allows for a column name. |
| int | [**getMaxColumnsInGroupBy**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxColumnsInGroupBy())()            Retrieves the maximum number of columns this database allows in a GROUP BY clause. |
| int | [**getMaxColumnsInIndex**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxColumnsInIndex())()            Retrieves the maximum number of columns this database allows in an index. |
| int | [**getMaxColumnsInOrderBy**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxColumnsInOrderBy())()            Retrieves the maximum number of columns this database allows in an ORDER BY clause. |
| int | [**getMaxColumnsInSelect**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxColumnsInSelect())()            Retrieves the maximum number of columns this database allows in a SELECT list. |
| int | [**getMaxColumnsInTable**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxColumnsInTable())()            Retrieves the maximum number of columns this database allows in a table. |
| int | [**getMaxConnections**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxConnections())()            Retrieves the maximum number of concurrent connections to this database that are possible. |
| int | [**getMaxCursorNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxCursorNameLength())()            Retrieves the maximum number of characters that this database allows in a cursor name. |
| int | [**getMaxIndexLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxIndexLength())()            Retrieves the maximum number of bytes this database allows for an index, including all of the parts of the index. |
| int | [**getMaxProcedureNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxProcedureNameLength())()            Retrieves the maximum number of characters that this database allows in a procedure name. |
| int | [**getMaxRowSize**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxRowSize())()            Retrieves the maximum number of bytes this database allows in a single row. |
| int | [**getMaxSchemaNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxSchemaNameLength())()            Retrieves the maximum number of characters that this database allows in a schema name. |
| int | [**getMaxStatementLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxStatementLength())()            Retrieves the maximum number of characters this database allows in an SQL statement. |
| int | [**getMaxStatements**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxStatements())()            Retrieves the maximum number of active statements to this database that can be open at the same time. |
| int | [**getMaxTableNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxTableNameLength())()            Retrieves the maximum number of characters this database allows in a table name. |
| int | [**getMaxTablesInSelect**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxTablesInSelect())()            Retrieves the maximum number of tables this database allows in a SELECT statement. |
| int | [**getMaxUserNameLength**](http://docs.google.com/java/sql/DatabaseMetaData.html#getMaxUserNameLength())()            Retrieves the maximum number of characters this database allows in a user name. |
| [String](http://docs.google.com/java/lang/String.html) | [**getNumericFunctions**](http://docs.google.com/java/sql/DatabaseMetaData.html#getNumericFunctions())()            Retrieves a comma-separated list of math functions available with this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getPrimaryKeys**](http://docs.google.com/java/sql/DatabaseMetaData.html#getPrimaryKeys(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table)            Retrieves a description of the given table's primary key columns. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getProcedureColumns**](http://docs.google.com/java/sql/DatabaseMetaData.html#getProcedureColumns(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) procedureNamePattern, [String](http://docs.google.com/java/lang/String.html) columnNamePattern)            Retrieves a description of the given catalog's stored procedure parameter and result columns. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getProcedures**](http://docs.google.com/java/sql/DatabaseMetaData.html#getProcedures(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) procedureNamePattern)            Retrieves a description of the stored procedures available in the given catalog. |
| [String](http://docs.google.com/java/lang/String.html) | [**getProcedureTerm**](http://docs.google.com/java/sql/DatabaseMetaData.html#getProcedureTerm())()            Retrieves the database vendor's preferred term for "procedure". |
| int | [**getResultSetHoldability**](http://docs.google.com/java/sql/DatabaseMetaData.html#getResultSetHoldability())()            Retrieves this database's default holdability for ResultSet objects. |
| [RowIdLifetime](http://docs.google.com/java/sql/RowIdLifetime.html) | [**getRowIdLifetime**](http://docs.google.com/java/sql/DatabaseMetaData.html#getRowIdLifetime())()            Indicates whether or not this data source supports the SQL ROWID type, and if so the lifetime for which a RowId object remains valid. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getSchemas**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSchemas())()            Retrieves the schema names available in this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getSchemas**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSchemas(java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern)            Retrieves the schema names available in this database. |
| [String](http://docs.google.com/java/lang/String.html) | [**getSchemaTerm**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSchemaTerm())()            Retrieves the database vendor's preferred term for "schema". |
| [String](http://docs.google.com/java/lang/String.html) | [**getSearchStringEscape**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())()            Retrieves the string that can be used to escape wildcard characters. |
| [String](http://docs.google.com/java/lang/String.html) | [**getSQLKeywords**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSQLKeywords())()            Retrieves a comma-separated list of all of this database's SQL keywords that are NOT also SQL:2003 keywords. |
| int | [**getSQLStateType**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSQLStateType())()            Indicates whether the SQLSTATE returned by SQLException.getSQLState is X/Open (now known as Open Group) SQL CLI or SQL:2003. |
| [String](http://docs.google.com/java/lang/String.html) | [**getStringFunctions**](http://docs.google.com/java/sql/DatabaseMetaData.html#getStringFunctions())()            Retrieves a comma-separated list of string functions available with this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getSuperTables**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSuperTables(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) tableNamePattern)            Retrieves a description of the table hierarchies defined in a particular schema in this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getSuperTypes**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSuperTypes(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) typeNamePattern)            Retrieves a description of the user-defined type (UDT) hierarchies defined in a particular schema in this database. |
| [String](http://docs.google.com/java/lang/String.html) | [**getSystemFunctions**](http://docs.google.com/java/sql/DatabaseMetaData.html#getSystemFunctions())()            Retrieves a comma-separated list of system functions available with this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getTablePrivileges**](http://docs.google.com/java/sql/DatabaseMetaData.html#getTablePrivileges(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) tableNamePattern)            Retrieves a description of the access rights for each table available in a catalog. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getTables**](http://docs.google.com/java/sql/DatabaseMetaData.html#getTables(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String%5B%5D))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) tableNamePattern, [String](http://docs.google.com/java/lang/String.html)[] types)            Retrieves a description of the tables available in the given catalog. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getTableTypes**](http://docs.google.com/java/sql/DatabaseMetaData.html#getTableTypes())()            Retrieves the table types available in this database. |
| [String](http://docs.google.com/java/lang/String.html) | [**getTimeDateFunctions**](http://docs.google.com/java/sql/DatabaseMetaData.html#getTimeDateFunctions())()            Retrieves a comma-separated list of the time and date functions available with this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getTypeInfo**](http://docs.google.com/java/sql/DatabaseMetaData.html#getTypeInfo())()            Retrieves a description of all the data types supported by this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getUDTs**](http://docs.google.com/java/sql/DatabaseMetaData.html#getUDTs(java.lang.String,%20java.lang.String,%20java.lang.String,%20int%5B%5D))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schemaPattern, [String](http://docs.google.com/java/lang/String.html) typeNamePattern, int[] types)            Retrieves a description of the user-defined types (UDTs) defined in a particular schema. |
| [String](http://docs.google.com/java/lang/String.html) | [**getURL**](http://docs.google.com/java/sql/DatabaseMetaData.html#getURL())()            Retrieves the URL for this DBMS. |
| [String](http://docs.google.com/java/lang/String.html) | [**getUserName**](http://docs.google.com/java/sql/DatabaseMetaData.html#getUserName())()            Retrieves the user name as known to this database. |
| [ResultSet](http://docs.google.com/java/sql/ResultSet.html) | [**getVersionColumns**](http://docs.google.com/java/sql/DatabaseMetaData.html#getVersionColumns(java.lang.String,%20java.lang.String,%20java.lang.String))([String](http://docs.google.com/java/lang/String.html) catalog, [String](http://docs.google.com/java/lang/String.html) schema, [String](http://docs.google.com/java/lang/String.html) table)            Retrieves a description of a table's columns that are automatically updated when any value in a row is updated. |
| boolean | [**insertsAreDetected**](http://docs.google.com/java/sql/DatabaseMetaData.html#insertsAreDetected(int))(int type)            Retrieves whether or not a visible row insert can be detected by calling the method ResultSet.rowInserted. |
| boolean | [**isCatalogAtStart**](http://docs.google.com/java/sql/DatabaseMetaData.html#isCatalogAtStart())()            Retrieves whether a catalog appears at the start of a fully qualified table name. |
| boolean | [**isReadOnly**](http://docs.google.com/java/sql/DatabaseMetaData.html#isReadOnly())()            Retrieves whether this database is in read-only mode. |
| boolean | [**locatorsUpdateCopy**](http://docs.google.com/java/sql/DatabaseMetaData.html#locatorsUpdateCopy())()            Indicates whether updates made to a LOB are made on a copy or directly to the LOB. |
| boolean | [**nullPlusNonNullIsNull**](http://docs.google.com/java/sql/DatabaseMetaData.html#nullPlusNonNullIsNull())()            Retrieves whether this database supports concatenations between NULL and non-NULL values being NULL. |
| boolean | [**nullsAreSortedAtEnd**](http://docs.google.com/java/sql/DatabaseMetaData.html#nullsAreSortedAtEnd())()            Retrieves whether NULL values are sorted at the end regardless of sort order. |
| boolean | [**nullsAreSortedAtStart**](http://docs.google.com/java/sql/DatabaseMetaData.html#nullsAreSortedAtStart())()            Retrieves whether NULL values are sorted at the start regardless of sort order. |
| boolean | [**nullsAreSortedHigh**](http://docs.google.com/java/sql/DatabaseMetaData.html#nullsAreSortedHigh())()            Retrieves whether NULL values are sorted high. |
| boolean | [**nullsAreSortedLow**](http://docs.google.com/java/sql/DatabaseMetaData.html#nullsAreSortedLow())()            Retrieves whether NULL values are sorted low. |
| boolean | [**othersDeletesAreVisible**](http://docs.google.com/java/sql/DatabaseMetaData.html#othersDeletesAreVisible(int))(int type)            Retrieves whether deletes made by others are visible. |
| boolean | [**othersInsertsAreVisible**](http://docs.google.com/java/sql/DatabaseMetaData.html#othersInsertsAreVisible(int))(int type)            Retrieves whether inserts made by others are visible. |
| boolean | [**othersUpdatesAreVisible**](http://docs.google.com/java/sql/DatabaseMetaData.html#othersUpdatesAreVisible(int))(int type)            Retrieves whether updates made by others are visible. |
| boolean | [**ownDeletesAreVisible**](http://docs.google.com/java/sql/DatabaseMetaData.html#ownDeletesAreVisible(int))(int type)            Retrieves whether a result set's own deletes are visible. |
| boolean | [**ownInsertsAreVisible**](http://docs.google.com/java/sql/DatabaseMetaData.html#ownInsertsAreVisible(int))(int type)            Retrieves whether a result set's own inserts are visible. |
| boolean | [**ownUpdatesAreVisible**](http://docs.google.com/java/sql/DatabaseMetaData.html#ownUpdatesAreVisible(int))(int type)            Retrieves whether for the given type of ResultSet object, the result set's own updates are visible. |
| boolean | [**storesLowerCaseIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#storesLowerCaseIdentifiers())()            Retrieves whether this database treats mixed case unquoted SQL identifiers as case insensitive and stores them in lower case. |
| boolean | [**storesLowerCaseQuotedIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#storesLowerCaseQuotedIdentifiers())()            Retrieves whether this database treats mixed case quoted SQL identifiers as case insensitive and stores them in lower case. |
| boolean | [**storesMixedCaseIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#storesMixedCaseIdentifiers())()            Retrieves whether this database treats mixed case unquoted SQL identifiers as case insensitive and stores them in mixed case. |
| boolean | [**storesMixedCaseQuotedIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#storesMixedCaseQuotedIdentifiers())()            Retrieves whether this database treats mixed case quoted SQL identifiers as case insensitive and stores them in mixed case. |
| boolean | [**storesUpperCaseIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#storesUpperCaseIdentifiers())()            Retrieves whether this database treats mixed case unquoted SQL identifiers as case insensitive and stores them in upper case. |
| boolean | [**storesUpperCaseQuotedIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#storesUpperCaseQuotedIdentifiers())()            Retrieves whether this database treats mixed case quoted SQL identifiers as case insensitive and stores them in upper case. |
| boolean | [**supportsAlterTableWithAddColumn**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsAlterTableWithAddColumn())()            Retrieves whether this database supports ALTER TABLE with add column. |
| boolean | [**supportsAlterTableWithDropColumn**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsAlterTableWithDropColumn())()            Retrieves whether this database supports ALTER TABLE with drop column. |
| boolean | [**supportsANSI92EntryLevelSQL**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsANSI92EntryLevelSQL())()            Retrieves whether this database supports the ANSI92 entry level SQL grammar. |
| boolean | [**supportsANSI92FullSQL**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsANSI92FullSQL())()            Retrieves whether this database supports the ANSI92 full SQL grammar supported. |
| boolean | [**supportsANSI92IntermediateSQL**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsANSI92IntermediateSQL())()            Retrieves whether this database supports the ANSI92 intermediate SQL grammar supported. |
| boolean | [**supportsBatchUpdates**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsBatchUpdates())()            Retrieves whether this database supports batch updates. |
| boolean | [**supportsCatalogsInDataManipulation**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCatalogsInDataManipulation())()            Retrieves whether a catalog name can be used in a data manipulation statement. |
| boolean | [**supportsCatalogsInIndexDefinitions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCatalogsInIndexDefinitions())()            Retrieves whether a catalog name can be used in an index definition statement. |
| boolean | [**supportsCatalogsInPrivilegeDefinitions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCatalogsInPrivilegeDefinitions())()            Retrieves whether a catalog name can be used in a privilege definition statement. |
| boolean | [**supportsCatalogsInProcedureCalls**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCatalogsInProcedureCalls())()            Retrieves whether a catalog name can be used in a procedure call statement. |
| boolean | [**supportsCatalogsInTableDefinitions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCatalogsInTableDefinitions())()            Retrieves whether a catalog name can be used in a table definition statement. |
| boolean | [**supportsColumnAliasing**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsColumnAliasing())()            Retrieves whether this database supports column aliasing. |
| boolean | [**supportsConvert**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsConvert())()            Retrieves whether this database supports the JDBC scalar function CONVERT for the conversion of one JDBC type to another. |
| boolean | [**supportsConvert**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsConvert(int,%20int))(int fromType, int toType)            Retrieves whether this database supports the JDBC scalar function CONVERT for conversions between the JDBC types *fromType* and *toType*. |
| boolean | [**supportsCoreSQLGrammar**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCoreSQLGrammar())()            Retrieves whether this database supports the ODBC Core SQL grammar. |
| boolean | [**supportsCorrelatedSubqueries**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsCorrelatedSubqueries())()            Retrieves whether this database supports correlated subqueries. |
| boolean | [**supportsDataDefinitionAndDataManipulationTransactions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsDataDefinitionAndDataManipulationTransactions())()            Retrieves whether this database supports both data definition and data manipulation statements within a transaction. |
| boolean | [**supportsDataManipulationTransactionsOnly**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsDataManipulationTransactionsOnly())()            Retrieves whether this database supports only data manipulation statements within a transaction. |
| boolean | [**supportsDifferentTableCorrelationNames**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsDifferentTableCorrelationNames())()            Retrieves whether, when table correlation names are supported, they are restricted to being different from the names of the tables. |
| boolean | [**supportsExpressionsInOrderBy**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsExpressionsInOrderBy())()            Retrieves whether this database supports expressions in ORDER BY lists. |
| boolean | [**supportsExtendedSQLGrammar**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsExtendedSQLGrammar())()            Retrieves whether this database supports the ODBC Extended SQL grammar. |
| boolean | [**supportsFullOuterJoins**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsFullOuterJoins())()            Retrieves whether this database supports full nested outer joins. |
| boolean | [**supportsGetGeneratedKeys**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsGetGeneratedKeys())()            Retrieves whether auto-generated keys can be retrieved after a statement has been executed |
| boolean | [**supportsGroupBy**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsGroupBy())()            Retrieves whether this database supports some form of GROUP BY clause. |
| boolean | [**supportsGroupByBeyondSelect**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsGroupByBeyondSelect())()            Retrieves whether this database supports using columns not included in the SELECT statement in a GROUP BY clause provided that all of the columns in the SELECT statement are included in the GROUP BY clause. |
| boolean | [**supportsGroupByUnrelated**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsGroupByUnrelated())()            Retrieves whether this database supports using a column that is not in the SELECT statement in a GROUP BY clause. |
| boolean | [**supportsIntegrityEnhancementFacility**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsIntegrityEnhancementFacility())()            Retrieves whether this database supports the SQL Integrity Enhancement Facility. |
| boolean | [**supportsLikeEscapeClause**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsLikeEscapeClause())()            Retrieves whether this database supports specifying a LIKE escape clause. |
| boolean | [**supportsLimitedOuterJoins**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsLimitedOuterJoins())()            Retrieves whether this database provides limited support for outer joins. |
| boolean | [**supportsMinimumSQLGrammar**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsMinimumSQLGrammar())()            Retrieves whether this database supports the ODBC Minimum SQL grammar. |
| boolean | [**supportsMixedCaseIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsMixedCaseIdentifiers())()            Retrieves whether this database treats mixed case unquoted SQL identifiers as case sensitive and as a result stores them in mixed case. |
| boolean | [**supportsMixedCaseQuotedIdentifiers**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsMixedCaseQuotedIdentifiers())()            Retrieves whether this database treats mixed case quoted SQL identifiers as case sensitive and as a result stores them in mixed case. |
| boolean | [**supportsMultipleOpenResults**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsMultipleOpenResults())()            Retrieves whether it is possible to have multiple ResultSet objects returned from a CallableStatement object simultaneously. |
| boolean | [**supportsMultipleResultSets**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsMultipleResultSets())()            Retrieves whether this database supports getting multiple ResultSet objects from a single call to the method execute. |
| boolean | [**supportsMultipleTransactions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsMultipleTransactions())()            Retrieves whether this database allows having multiple transactions open at once (on different connections). |
| boolean | [**supportsNamedParameters**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsNamedParameters())()            Retrieves whether this database supports named parameters to callable statements. |
| boolean | [**supportsNonNullableColumns**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsNonNullableColumns())()            Retrieves whether columns in this database may be defined as non-nullable. |
| boolean | [**supportsOpenCursorsAcrossCommit**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsOpenCursorsAcrossCommit())()            Retrieves whether this database supports keeping cursors open across commits. |
| boolean | [**supportsOpenCursorsAcrossRollback**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsOpenCursorsAcrossRollback())()            Retrieves whether this database supports keeping cursors open across rollbacks. |
| boolean | [**supportsOpenStatementsAcrossCommit**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsOpenStatementsAcrossCommit())()            Retrieves whether this database supports keeping statements open across commits. |
| boolean | [**supportsOpenStatementsAcrossRollback**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsOpenStatementsAcrossRollback())()            Retrieves whether this database supports keeping statements open across rollbacks. |
| boolean | [**supportsOrderByUnrelated**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsOrderByUnrelated())()            Retrieves whether this database supports using a column that is not in the SELECT statement in an ORDER BY clause. |
| boolean | [**supportsOuterJoins**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsOuterJoins())()            Retrieves whether this database supports some form of outer join. |
| boolean | [**supportsPositionedDelete**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsPositionedDelete())()            Retrieves whether this database supports positioned DELETE statements. |
| boolean | [**supportsPositionedUpdate**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsPositionedUpdate())()            Retrieves whether this database supports positioned UPDATE statements. |
| boolean | [**supportsResultSetConcurrency**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsResultSetConcurrency(int,%20int))(int type, int concurrency)            Retrieves whether this database supports the given concurrency type in combination with the given result set type. |
| boolean | [**supportsResultSetHoldability**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsResultSetHoldability(int))(int holdability)            Retrieves whether this database supports the given result set holdability. |
| boolean | [**supportsResultSetType**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsResultSetType(int))(int type)            Retrieves whether this database supports the given result set type. |
| boolean | [**supportsSavepoints**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSavepoints())()            Retrieves whether this database supports savepoints. |
| boolean | [**supportsSchemasInDataManipulation**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSchemasInDataManipulation())()            Retrieves whether a schema name can be used in a data manipulation statement. |
| boolean | [**supportsSchemasInIndexDefinitions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSchemasInIndexDefinitions())()            Retrieves whether a schema name can be used in an index definition statement. |
| boolean | [**supportsSchemasInPrivilegeDefinitions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSchemasInPrivilegeDefinitions())()            Retrieves whether a schema name can be used in a privilege definition statement. |
| boolean | [**supportsSchemasInProcedureCalls**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSchemasInProcedureCalls())()            Retrieves whether a schema name can be used in a procedure call statement. |
| boolean | [**supportsSchemasInTableDefinitions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSchemasInTableDefinitions())()            Retrieves whether a schema name can be used in a table definition statement. |
| boolean | [**supportsSelectForUpdate**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSelectForUpdate())()            Retrieves whether this database supports SELECT FOR UPDATE statements. |
| boolean | [**supportsStatementPooling**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsStatementPooling())()            Retrieves whether this database supports statement pooling. |
| boolean | [**supportsStoredFunctionsUsingCallSyntax**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsStoredFunctionsUsingCallSyntax())()            Retrieves whether this database supports invoking user-defined or vendor functions using the stored procedure escape syntax. |
| boolean | [**supportsStoredProcedures**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsStoredProcedures())()            Retrieves whether this database supports stored procedure calls that use the stored procedure escape syntax. |
| boolean | [**supportsSubqueriesInComparisons**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSubqueriesInComparisons())()            Retrieves whether this database supports subqueries in comparison expressions. |
| boolean | [**supportsSubqueriesInExists**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSubqueriesInExists())()            Retrieves whether this database supports subqueries in EXISTS expressions. |
| boolean | [**supportsSubqueriesInIns**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSubqueriesInIns())()            Retrieves whether this database supports subqueries in IN expressions. |
| boolean | [**supportsSubqueriesInQuantifieds**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsSubqueriesInQuantifieds())()            Retrieves whether this database supports subqueries in quantified expressions. |
| boolean | [**supportsTableCorrelationNames**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsTableCorrelationNames())()            Retrieves whether this database supports table correlation names. |
| boolean | [**supportsTransactionIsolationLevel**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsTransactionIsolationLevel(int))(int level)            Retrieves whether this database supports the given transaction isolation level. |
| boolean | [**supportsTransactions**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsTransactions())()            Retrieves whether this database supports transactions. |
| boolean | [**supportsUnion**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsUnion())()            Retrieves whether this database supports SQL UNION. |
| boolean | [**supportsUnionAll**](http://docs.google.com/java/sql/DatabaseMetaData.html#supportsUnionAll())()            Retrieves whether this database supports SQL UNION ALL. |
| boolean | [**updatesAreDetected**](http://docs.google.com/java/sql/DatabaseMetaData.html#updatesAreDetected(int))(int type)            Retrieves whether or not a visible row update can be detected by calling the method ResultSet.rowUpdated. |
| boolean | [**usesLocalFilePerTable**](http://docs.google.com/java/sql/DatabaseMetaData.html#usesLocalFilePerTable())()            Retrieves whether this database uses a file for each table. |
| boolean | [**usesLocalFiles**](http://docs.google.com/java/sql/DatabaseMetaData.html#usesLocalFiles())()            Retrieves whether this database stores tables in a local file. |

| **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)) |

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

### procedureResultUnknown

static final int **procedureResultUnknown**

Indicates that it is not known whether the procedure returns a result.

A possible value for column PROCEDURE\_TYPE in the ResultSet object returned by the method getProcedures.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureResultUnknown)

### procedureNoResult

static final int **procedureNoResult**

Indicates that the procedure does not return a result.

A possible value for column PROCEDURE\_TYPE in the ResultSet object returned by the method getProcedures.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureNoResult)

### procedureReturnsResult

static final int **procedureReturnsResult**

Indicates that the procedure returns a result.

A possible value for column PROCEDURE\_TYPE in the ResultSet object returned by the method getProcedures.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureReturnsResult)

### procedureColumnUnknown

static final int **procedureColumnUnknown**

Indicates that type of the column is unknown.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureColumnUnknown)

### procedureColumnIn

static final int **procedureColumnIn**

Indicates that the column stores IN parameters.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureColumnIn)

### procedureColumnInOut

static final int **procedureColumnInOut**

Indicates that the column stores INOUT parameters.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureColumnInOut)

### procedureColumnOut

static final int **procedureColumnOut**

Indicates that the column stores OUT parameters.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureColumnOut)

### procedureColumnReturn

static final int **procedureColumnReturn**

Indicates that the column stores return values.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureColumnReturn)

### procedureColumnResult

static final int **procedureColumnResult**

Indicates that the column stores results.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureColumnResult)

### procedureNoNulls

static final int **procedureNoNulls**

Indicates that NULL values are not allowed.

A possible value for the column NULLABLE in the ResultSet object returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureNoNulls)

### procedureNullable

static final int **procedureNullable**

Indicates that NULL values are allowed.

A possible value for the column NULLABLE in the ResultSet object returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureNullable)

### procedureNullableUnknown

static final int **procedureNullableUnknown**

Indicates that whether NULL values are allowed is unknown.

A possible value for the column NULLABLE in the ResultSet object returned by the method getProcedureColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.procedureNullableUnknown)

### columnNoNulls

static final int **columnNoNulls**

Indicates that the column might not allow NULL values.

A possible value for the column NULLABLE in the ResultSet returned by the method getColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.columnNoNulls)

### columnNullable

static final int **columnNullable**

Indicates that the column definitely allows NULL values.

A possible value for the column NULLABLE in the ResultSet returned by the method getColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.columnNullable)

### columnNullableUnknown

static final int **columnNullableUnknown**

Indicates that the nullability of columns is unknown.

A possible value for the column NULLABLE in the ResultSet returned by the method getColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.columnNullableUnknown)

### bestRowTemporary

static final int **bestRowTemporary**

Indicates that the scope of the best row identifier is very temporary, lasting only while the row is being used.

A possible value for the column SCOPE in the ResultSet object returned by the method getBestRowIdentifier.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.bestRowTemporary)

### bestRowTransaction

static final int **bestRowTransaction**

Indicates that the scope of the best row identifier is the remainder of the current transaction.

A possible value for the column SCOPE in the ResultSet object returned by the method getBestRowIdentifier.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.bestRowTransaction)

### bestRowSession

static final int **bestRowSession**

Indicates that the scope of the best row identifier is the remainder of the current session.

A possible value for the column SCOPE in the ResultSet object returned by the method getBestRowIdentifier.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.bestRowSession)

### bestRowUnknown

static final int **bestRowUnknown**

Indicates that the best row identifier may or may not be a pseudo column.

A possible value for the column PSEUDO\_COLUMN in the ResultSet object returned by the method getBestRowIdentifier.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.bestRowUnknown)

### bestRowNotPseudo

static final int **bestRowNotPseudo**

Indicates that the best row identifier is NOT a pseudo column.

A possible value for the column PSEUDO\_COLUMN in the ResultSet object returned by the method getBestRowIdentifier.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.bestRowNotPseudo)

### bestRowPseudo

static final int **bestRowPseudo**

Indicates that the best row identifier is a pseudo column.

A possible value for the column PSEUDO\_COLUMN in the ResultSet object returned by the method getBestRowIdentifier.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.bestRowPseudo)

### versionColumnUnknown

static final int **versionColumnUnknown**

Indicates that this version column may or may not be a pseudo column.

A possible value for the column PSEUDO\_COLUMN in the ResultSet object returned by the method getVersionColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.versionColumnUnknown)

### versionColumnNotPseudo

static final int **versionColumnNotPseudo**

Indicates that this version column is NOT a pseudo column.

A possible value for the column PSEUDO\_COLUMN in the ResultSet object returned by the method getVersionColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.versionColumnNotPseudo)

### versionColumnPseudo

static final int **versionColumnPseudo**

Indicates that this version column is a pseudo column.

A possible value for the column PSEUDO\_COLUMN in the ResultSet object returned by the method getVersionColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.versionColumnPseudo)

### importedKeyCascade

static final int **importedKeyCascade**

For the column UPDATE\_RULE, indicates that when the primary key is updated, the foreign key (imported key) is changed to agree with it. For the column DELETE\_RULE, it indicates that when the primary key is deleted, rows that imported that key are deleted.

A possible value for the columns UPDATE\_RULE and DELETE\_RULE in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeyCascade)

### importedKeyRestrict

static final int **importedKeyRestrict**

For the column UPDATE\_RULE, indicates that a primary key may not be updated if it has been imported by another table as a foreign key. For the column DELETE\_RULE, indicates that a primary key may not be deleted if it has been imported by another table as a foreign key.

A possible value for the columns UPDATE\_RULE and DELETE\_RULE in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeyRestrict)

### importedKeySetNull

static final int **importedKeySetNull**

For the columns UPDATE\_RULE and DELETE\_RULE, indicates that when the primary key is updated or deleted, the foreign key (imported key) is changed to NULL.

A possible value for the columns UPDATE\_RULE and DELETE\_RULE in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeySetNull)

### importedKeyNoAction

static final int **importedKeyNoAction**

For the columns UPDATE\_RULE and DELETE\_RULE, indicates that if the primary key has been imported, it cannot be updated or deleted.

A possible value for the columns UPDATE\_RULE and DELETE\_RULE in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeyNoAction)

### importedKeySetDefault

static final int **importedKeySetDefault**

For the columns UPDATE\_RULE and DELETE\_RULE, indicates that if the primary key is updated or deleted, the foreign key (imported key) is set to the default value.

A possible value for the columns UPDATE\_RULE and DELETE\_RULE in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeySetDefault)

### importedKeyInitiallyDeferred

static final int **importedKeyInitiallyDeferred**

Indicates deferrability. See SQL-92 for a definition.

A possible value for the column DEFERRABILITY in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeyInitiallyDeferred)

### importedKeyInitiallyImmediate

static final int **importedKeyInitiallyImmediate**

Indicates deferrability. See SQL-92 for a definition.

A possible value for the column DEFERRABILITY in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeyInitiallyImmediate)

### importedKeyNotDeferrable

static final int **importedKeyNotDeferrable**

Indicates deferrability. See SQL-92 for a definition.

A possible value for the column DEFERRABILITY in the ResultSet objects returned by the methods getImportedKeys, getExportedKeys, and getCrossReference.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.importedKeyNotDeferrable)

### typeNoNulls

static final int **typeNoNulls**

Indicates that a NULL value is NOT allowed for this data type.

A possible value for column NULLABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typeNoNulls)

### typeNullable

static final int **typeNullable**

Indicates that a NULL value is allowed for this data type.

A possible value for column NULLABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typeNullable)

### typeNullableUnknown

static final int **typeNullableUnknown**

Indicates that it is not known whether a NULL value is allowed for this data type.

A possible value for column NULLABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typeNullableUnknown)

### typePredNone

static final int **typePredNone**

Indicates that WHERE search clauses are not supported for this type.

A possible value for column SEARCHABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typePredNone)

### typePredChar

static final int **typePredChar**

Indicates that the data type can be only be used in WHERE search clauses that use LIKE predicates.

A possible value for column SEARCHABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typePredChar)

### typePredBasic

static final int **typePredBasic**

Indicates that the data type can be only be used in WHERE search clauses that do not use LIKE predicates.

A possible value for column SEARCHABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typePredBasic)

### typeSearchable

static final int **typeSearchable**

Indicates that all WHERE search clauses can be based on this type.

A possible value for column SEARCHABLE in the ResultSet object returned by the method getTypeInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.typeSearchable)

### tableIndexStatistic

static final short **tableIndexStatistic**

Indicates that this column contains table statistics that are returned in conjunction with a table's index descriptions.

A possible value for column TYPE in the ResultSet object returned by the method getIndexInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.tableIndexStatistic)

### tableIndexClustered

static final short **tableIndexClustered**

Indicates that this table index is a clustered index.

A possible value for column TYPE in the ResultSet object returned by the method getIndexInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.tableIndexClustered)

### tableIndexHashed

static final short **tableIndexHashed**

Indicates that this table index is a hashed index.

A possible value for column TYPE in the ResultSet object returned by the method getIndexInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.tableIndexHashed)

### tableIndexOther

static final short **tableIndexOther**

Indicates that this table index is not a clustered index, a hashed index, or table statistics; it is something other than these.

A possible value for column TYPE in the ResultSet object returned by the method getIndexInfo.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.tableIndexOther)

### attributeNoNulls

static final short **attributeNoNulls**

Indicates that NULL values might not be allowed.

A possible value for the column NULLABLE in the ResultSet object returned by the method getAttributes.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.attributeNoNulls)

### attributeNullable

static final short **attributeNullable**

Indicates that NULL values are definitely allowed.

A possible value for the column NULLABLE in the ResultSet object returned by the method getAttributes.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.attributeNullable)

### attributeNullableUnknown

static final short **attributeNullableUnknown**

Indicates that whether NULL values are allowed is not known.

A possible value for the column NULLABLE in the ResultSet object returned by the method getAttributes.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.attributeNullableUnknown)

### sqlStateXOpen

static final int **sqlStateXOpen**

A possible return value for the method DatabaseMetaData.getSQLStateType which is used to indicate whether the value returned by the method SQLException.getSQLState is an X/Open (now know as Open Group) SQL CLI SQLSTATE value.

**Since:** 1.4 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.sqlStateXOpen)

### sqlStateSQL

static final int **sqlStateSQL**

A possible return value for the method DatabaseMetaData.getSQLStateType which is used to indicate whether the value returned by the method SQLException.getSQLState is an SQLSTATE value.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.sqlStateSQL)

### sqlStateSQL99

static final int **sqlStateSQL99**

A possible return value for the method DatabaseMetaData.getSQLStateType which is used to indicate whether the value returned by the method SQLException.getSQLState is an SQL99 SQLSTATE value.

**Note:**This constant remains only for compatibility reasons. Developers should use the constant sqlStateSQL instead.

**Since:** 1.4 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.sqlStateSQL99)

### functionColumnUnknown

static final int **functionColumnUnknown**

Indicates that type of the parameter or column is unknown.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getFunctionColumns.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionColumnUnknown)

### functionColumnIn

static final int **functionColumnIn**

Indicates that the parameter or column is an IN parameter.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionColumnIn)

### functionColumnInOut

static final int **functionColumnInOut**

Indicates that the parameter or column is an INOUT parameter.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionColumnInOut)

### functionColumnOut

static final int **functionColumnOut**

Indicates that the parameter or column is an OUT parameter.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionColumnOut)

### functionReturn

static final int **functionReturn**

Indicates that the parameter or column is a return value.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionReturn)

### functionColumnResult

static final int **functionColumnResult**

Indicates that the parameter or column is a column in a result set.

A possible value for the column COLUMN\_TYPE in the ResultSet returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionColumnResult)

### functionNoNulls

static final int **functionNoNulls**

Indicates that NULL values are not allowed.

A possible value for the column NULLABLE in the ResultSet object returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionNoNulls)

### functionNullable

static final int **functionNullable**

Indicates that NULL values are allowed.

A possible value for the column NULLABLE in the ResultSet object returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionNullable)

### functionNullableUnknown

static final int **functionNullableUnknown**

Indicates that whether NULL values are allowed is unknown.

A possible value for the column NULLABLE in the ResultSet object returned by the method getFunctionColumns.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionNullableUnknown)

### functionResultUnknown

static final int **functionResultUnknown**

Indicates that it is not known whether the function returns a result or a table.

A possible value for column FUNCTION\_TYPE in the ResultSet object returned by the method getFunctions.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionResultUnknown)

### functionNoTable

static final int **functionNoTable**

Indicates that the function does not return a table.

A possible value for column FUNCTION\_TYPE in the ResultSet object returned by the method getFunctions.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionNoTable)

### functionReturnsTable

static final int **functionReturnsTable**

Indicates that the function returns a table.

A possible value for column FUNCTION\_TYPE in the ResultSet object returned by the method getFunctions.

**Since:** 1.6 **See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#java.sql.DatabaseMetaData.functionReturnsTable)

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

### allProceduresAreCallable

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

Retrieves whether the current user can call all the procedures returned by the method getProcedures.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### allTablesAreSelectable

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

Retrieves whether the current user can use all the tables returned by the method getTables in a SELECT statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getURL

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

Retrieves the URL for this DBMS.

**Returns:**the URL for this DBMS or null if it cannot be generated **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getUserName

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

Retrieves the user name as known to this database.

**Returns:**the database user name **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### isReadOnly

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

Retrieves whether this database is in read-only mode.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### nullsAreSortedHigh

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

Retrieves whether NULL values are sorted high. Sorted high means that NULL values sort higher than any other value in a domain. In an ascending order, if this method returns true, NULL values will appear at the end. By contrast, the method nullsAreSortedAtEnd indicates whether NULL values are sorted at the end regardless of sort order.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### nullsAreSortedLow

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

Retrieves whether NULL values are sorted low. Sorted low means that NULL values sort lower than any other value in a domain. In an ascending order, if this method returns true, NULL values will appear at the beginning. By contrast, the method nullsAreSortedAtStart indicates whether NULL values are sorted at the beginning regardless of sort order.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### nullsAreSortedAtStart

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

Retrieves whether NULL values are sorted at the start regardless of sort order.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### nullsAreSortedAtEnd

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

Retrieves whether NULL values are sorted at the end regardless of sort order.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getDatabaseProductName

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

Retrieves the name of this database product.

**Returns:**database product name **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getDatabaseProductVersion

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

Retrieves the version number of this database product.

**Returns:**database version number **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getDriverName

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

Retrieves the name of this JDBC driver.

**Returns:**JDBC driver name **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getDriverVersion

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

Retrieves the version number of this JDBC driver as a String.

**Returns:**JDBC driver version **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getDriverMajorVersion

int **getDriverMajorVersion**()

Retrieves this JDBC driver's major version number.

**Returns:**JDBC driver major version

### getDriverMinorVersion

int **getDriverMinorVersion**()

Retrieves this JDBC driver's minor version number.

**Returns:**JDBC driver minor version number

### usesLocalFiles

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

Retrieves whether this database stores tables in a local file.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### usesLocalFilePerTable

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

Retrieves whether this database uses a file for each table.

**Returns:**true if this database uses a local file for each table; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsMixedCaseIdentifiers

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

Retrieves whether this database treats mixed case unquoted SQL identifiers as case sensitive and as a result stores them in mixed case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### storesUpperCaseIdentifiers

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

Retrieves whether this database treats mixed case unquoted SQL identifiers as case insensitive and stores them in upper case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### storesLowerCaseIdentifiers

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

Retrieves whether this database treats mixed case unquoted SQL identifiers as case insensitive and stores them in lower case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### storesMixedCaseIdentifiers

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

Retrieves whether this database treats mixed case unquoted SQL identifiers as case insensitive and stores them in mixed case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsMixedCaseQuotedIdentifiers

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

Retrieves whether this database treats mixed case quoted SQL identifiers as case sensitive and as a result stores them in mixed case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### storesUpperCaseQuotedIdentifiers

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

Retrieves whether this database treats mixed case quoted SQL identifiers as case insensitive and stores them in upper case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### storesLowerCaseQuotedIdentifiers

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

Retrieves whether this database treats mixed case quoted SQL identifiers as case insensitive and stores them in lower case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### storesMixedCaseQuotedIdentifiers

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

Retrieves whether this database treats mixed case quoted SQL identifiers as case insensitive and stores them in mixed case.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getIdentifierQuoteString

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

Retrieves the string used to quote SQL identifiers. This method returns a space " " if identifier quoting is not supported.

**Returns:**the quoting string or a space if quoting is not supported **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getSQLKeywords

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

Retrieves a comma-separated list of all of this database's SQL keywords that are NOT also SQL:2003 keywords.

**Returns:**the list of this database's keywords that are not also SQL:2003 keywords **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getNumericFunctions

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

Retrieves a comma-separated list of math functions available with this database. These are the Open /Open CLI math function names used in the JDBC function escape clause.

**Returns:**the list of math functions supported by this database **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getStringFunctions

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

Retrieves a comma-separated list of string functions available with this database. These are the Open Group CLI string function names used in the JDBC function escape clause.

**Returns:**the list of string functions supported by this database **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getSystemFunctions

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

Retrieves a comma-separated list of system functions available with this database. These are the Open Group CLI system function names used in the JDBC function escape clause.

**Returns:**a list of system functions supported by this database **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getTimeDateFunctions

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

Retrieves a comma-separated list of the time and date functions available with this database.

**Returns:**the list of time and date functions supported by this database **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getSearchStringEscape

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

Retrieves the string that can be used to escape wildcard characters. This is the string that can be used to escape '\_' or '%' in the catalog search parameters that are a pattern (and therefore use one of the wildcard characters).

The '\_' character represents any single character; the '%' character represents any sequence of zero or more characters.

**Returns:**the string used to escape wildcard characters **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getExtraNameCharacters

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

Retrieves all the "extra" characters that can be used in unquoted identifier names (those beyond a-z, A-Z, 0-9 and \_).

**Returns:**the string containing the extra characters **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsAlterTableWithAddColumn

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

Retrieves whether this database supports ALTER TABLE with add column.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsAlterTableWithDropColumn

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

Retrieves whether this database supports ALTER TABLE with drop column.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsColumnAliasing

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

Retrieves whether this database supports column aliasing.

If so, the SQL AS clause can be used to provide names for computed columns or to provide alias names for columns as required.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### nullPlusNonNullIsNull

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

Retrieves whether this database supports concatenations between NULL and non-NULL values being NULL.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsConvert

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

Retrieves whether this database supports the JDBC scalar function CONVERT for the conversion of one JDBC type to another. The JDBC types are the generic SQL data types defined in java.sql.Types.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsConvert

boolean **supportsConvert**(int fromType,  
 int toType)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves whether this database supports the JDBC scalar function CONVERT for conversions between the JDBC types *fromType* and *toType*. The JDBC types are the generic SQL data types defined in java.sql.Types.

**Parameters:**fromType - the type to convert from; one of the type codes from the class java.sql.TypestoType - the type to convert to; one of the type codes from the class java.sql.Types **Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[Types](http://docs.google.com/java/sql/Types.html)

### supportsTableCorrelationNames

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

Retrieves whether this database supports table correlation names.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsDifferentTableCorrelationNames

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

Retrieves whether, when table correlation names are supported, they are restricted to being different from the names of the tables.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsExpressionsInOrderBy

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

Retrieves whether this database supports expressions in ORDER BY lists.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsOrderByUnrelated

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

Retrieves whether this database supports using a column that is not in the SELECT statement in an ORDER BY clause.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsGroupBy

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

Retrieves whether this database supports some form of GROUP BY clause.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsGroupByUnrelated

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

Retrieves whether this database supports using a column that is not in the SELECT statement in a GROUP BY clause.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsGroupByBeyondSelect

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

Retrieves whether this database supports using columns not included in the SELECT statement in a GROUP BY clause provided that all of the columns in the SELECT statement are included in the GROUP BY clause.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsLikeEscapeClause

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

Retrieves whether this database supports specifying a LIKE escape clause.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsMultipleResultSets

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

Retrieves whether this database supports getting multiple ResultSet objects from a single call to the method execute.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsMultipleTransactions

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

Retrieves whether this database allows having multiple transactions open at once (on different connections).

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsNonNullableColumns

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

Retrieves whether columns in this database may be defined as non-nullable.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsMinimumSQLGrammar

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

Retrieves whether this database supports the ODBC Minimum SQL grammar.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCoreSQLGrammar

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

Retrieves whether this database supports the ODBC Core SQL grammar.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsExtendedSQLGrammar

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

Retrieves whether this database supports the ODBC Extended SQL grammar.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsANSI92EntryLevelSQL

boolean **supportsANSI92EntryLevelSQL**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves whether this database supports the ANSI92 entry level SQL grammar.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsANSI92IntermediateSQL

boolean **supportsANSI92IntermediateSQL**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves whether this database supports the ANSI92 intermediate SQL grammar supported.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsANSI92FullSQL

boolean **supportsANSI92FullSQL**()  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves whether this database supports the ANSI92 full SQL grammar supported.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsIntegrityEnhancementFacility

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

Retrieves whether this database supports the SQL Integrity Enhancement Facility.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsOuterJoins

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

Retrieves whether this database supports some form of outer join.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsFullOuterJoins

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

Retrieves whether this database supports full nested outer joins.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsLimitedOuterJoins

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

Retrieves whether this database provides limited support for outer joins. (This will be true if the method supportsFullOuterJoins returns true).

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getSchemaTerm

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

Retrieves the database vendor's preferred term for "schema".

**Returns:**the vendor term for "schema" **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getProcedureTerm

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

Retrieves the database vendor's preferred term for "procedure".

**Returns:**the vendor term for "procedure" **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getCatalogTerm

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

Retrieves the database vendor's preferred term for "catalog".

**Returns:**the vendor term for "catalog" **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### isCatalogAtStart

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

Retrieves whether a catalog appears at the start of a fully qualified table name. If not, the catalog appears at the end.

**Returns:**true if the catalog name appears at the beginning of a fully qualified table name; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getCatalogSeparator

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

Retrieves the String that this database uses as the separator between a catalog and table name.

**Returns:**the separator string **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSchemasInDataManipulation

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

Retrieves whether a schema name can be used in a data manipulation statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSchemasInProcedureCalls

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

Retrieves whether a schema name can be used in a procedure call statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSchemasInTableDefinitions

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

Retrieves whether a schema name can be used in a table definition statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSchemasInIndexDefinitions

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

Retrieves whether a schema name can be used in an index definition statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSchemasInPrivilegeDefinitions

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

Retrieves whether a schema name can be used in a privilege definition statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCatalogsInDataManipulation

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

Retrieves whether a catalog name can be used in a data manipulation statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCatalogsInProcedureCalls

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

Retrieves whether a catalog name can be used in a procedure call statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCatalogsInTableDefinitions

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

Retrieves whether a catalog name can be used in a table definition statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCatalogsInIndexDefinitions

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

Retrieves whether a catalog name can be used in an index definition statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCatalogsInPrivilegeDefinitions

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

Retrieves whether a catalog name can be used in a privilege definition statement.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsPositionedDelete

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

Retrieves whether this database supports positioned DELETE statements.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsPositionedUpdate

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

Retrieves whether this database supports positioned UPDATE statements.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSelectForUpdate

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

Retrieves whether this database supports SELECT FOR UPDATE statements.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsStoredProcedures

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

Retrieves whether this database supports stored procedure calls that use the stored procedure escape syntax.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSubqueriesInComparisons

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

Retrieves whether this database supports subqueries in comparison expressions.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSubqueriesInExists

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

Retrieves whether this database supports subqueries in EXISTS expressions.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSubqueriesInIns

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

Retrieves whether this database supports subqueries in IN expressions.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsSubqueriesInQuantifieds

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

Retrieves whether this database supports subqueries in quantified expressions.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsCorrelatedSubqueries

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

Retrieves whether this database supports correlated subqueries.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsUnion

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

Retrieves whether this database supports SQL UNION.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsUnionAll

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

Retrieves whether this database supports SQL UNION ALL.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsOpenCursorsAcrossCommit

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

Retrieves whether this database supports keeping cursors open across commits.

**Returns:**true if cursors always remain open; false if they might not remain open **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsOpenCursorsAcrossRollback

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

Retrieves whether this database supports keeping cursors open across rollbacks.

**Returns:**true if cursors always remain open; false if they might not remain open **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsOpenStatementsAcrossCommit

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

Retrieves whether this database supports keeping statements open across commits.

**Returns:**true if statements always remain open; false if they might not remain open **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsOpenStatementsAcrossRollback

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

Retrieves whether this database supports keeping statements open across rollbacks.

**Returns:**true if statements always remain open; false if they might not remain open **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxBinaryLiteralLength

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

Retrieves the maximum number of hex characters this database allows in an inline binary literal.

**Returns:**max the maximum length (in hex characters) for a binary literal; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxCharLiteralLength

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

Retrieves the maximum number of characters this database allows for a character literal.

**Returns:**the maximum number of characters allowed for a character literal; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxColumnNameLength

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

Retrieves the maximum number of characters this database allows for a column name.

**Returns:**the maximum number of characters allowed for a column name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxColumnsInGroupBy

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

Retrieves the maximum number of columns this database allows in a GROUP BY clause.

**Returns:**the maximum number of columns allowed; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxColumnsInIndex

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

Retrieves the maximum number of columns this database allows in an index.

**Returns:**the maximum number of columns allowed; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxColumnsInOrderBy

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

Retrieves the maximum number of columns this database allows in an ORDER BY clause.

**Returns:**the maximum number of columns allowed; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxColumnsInSelect

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

Retrieves the maximum number of columns this database allows in a SELECT list.

**Returns:**the maximum number of columns allowed; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxColumnsInTable

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

Retrieves the maximum number of columns this database allows in a table.

**Returns:**the maximum number of columns allowed; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxConnections

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

Retrieves the maximum number of concurrent connections to this database that are possible.

**Returns:**the maximum number of active connections possible at one time; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxCursorNameLength

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

Retrieves the maximum number of characters that this database allows in a cursor name.

**Returns:**the maximum number of characters allowed in a cursor name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxIndexLength

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

Retrieves the maximum number of bytes this database allows for an index, including all of the parts of the index.

**Returns:**the maximum number of bytes allowed; this limit includes the composite of all the constituent parts of the index; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxSchemaNameLength

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

Retrieves the maximum number of characters that this database allows in a schema name.

**Returns:**the maximum number of characters allowed in a schema name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxProcedureNameLength

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

Retrieves the maximum number of characters that this database allows in a procedure name.

**Returns:**the maximum number of characters allowed in a procedure name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxCatalogNameLength

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

Retrieves the maximum number of characters that this database allows in a catalog name.

**Returns:**the maximum number of characters allowed in a catalog name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxRowSize

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

Retrieves the maximum number of bytes this database allows in a single row.

**Returns:**the maximum number of bytes allowed for a row; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### doesMaxRowSizeIncludeBlobs

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

Retrieves whether the return value for the method getMaxRowSize includes the SQL data types LONGVARCHAR and LONGVARBINARY.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxStatementLength

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

Retrieves the maximum number of characters this database allows in an SQL statement.

**Returns:**the maximum number of characters allowed for an SQL statement; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxStatements

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

Retrieves the maximum number of active statements to this database that can be open at the same time.

**Returns:**the maximum number of statements that can be open at one time; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxTableNameLength

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

Retrieves the maximum number of characters this database allows in a table name.

**Returns:**the maximum number of characters allowed for a table name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxTablesInSelect

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

Retrieves the maximum number of tables this database allows in a SELECT statement.

**Returns:**the maximum number of tables allowed in a SELECT statement; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getMaxUserNameLength

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

Retrieves the maximum number of characters this database allows in a user name.

**Returns:**the maximum number of characters allowed for a user name; a result of zero means that there is no limit or the limit is not known **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getDefaultTransactionIsolation

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

Retrieves this database's default transaction isolation level. The possible values are defined in java.sql.Connection.

**Returns:**the default isolation level **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[Connection](http://docs.google.com/java/sql/Connection.html)

### supportsTransactions

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

Retrieves whether this database supports transactions. If not, invoking the method commit is a noop, and the isolation level is TRANSACTION\_NONE.

**Returns:**true if transactions are supported; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsTransactionIsolationLevel

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

Retrieves whether this database supports the given transaction isolation level.

**Parameters:**level - one of the transaction isolation levels defined in java.sql.Connection **Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[Connection](http://docs.google.com/java/sql/Connection.html)

### supportsDataDefinitionAndDataManipulationTransactions

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

Retrieves whether this database supports both data definition and data manipulation statements within a transaction.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsDataManipulationTransactionsOnly

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

Retrieves whether this database supports only data manipulation statements within a transaction.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### dataDefinitionCausesTransactionCommit

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

Retrieves whether a data definition statement within a transaction forces the transaction to commit.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### dataDefinitionIgnoredInTransactions

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

Retrieves whether this database ignores a data definition statement within a transaction.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getProcedures

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getProcedures**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) procedureNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the stored procedures available in the given catalog.

Only procedure descriptions matching the schema and procedure name criteria are returned. They are ordered by PROCEDURE\_CAT, PROCEDURE\_SCHEM, PROCEDURE\_NAME and SPECIFIC\_ NAME.

Each procedure description has the the following columns:

1. **PROCEDURE\_CAT** String => procedure catalog (may be null)
2. **PROCEDURE\_SCHEM** String => procedure schema (may be null)
3. **PROCEDURE\_NAME** String => procedure name
4. reserved for future use
5. reserved for future use
6. reserved for future use
7. **REMARKS** String => explanatory comment on the procedure
8. **PROCEDURE\_TYPE** short => kind of procedure:
   * procedureResultUnknown - Cannot determine if a return value will be returned
   * procedureNoResult - Does not return a return value
   * procedureReturnsResult - Returns a return value
9. **SPECIFIC\_NAME** String => The name which uniquely identifies this procedure within its schema.

A user may not have permissions to execute any of the procedures that are returned by getProcedures

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchprocedureNamePattern - a procedure name pattern; must match the procedure name as it is stored in the database **Returns:**ResultSet - each row is a procedure description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getProcedureColumns

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getProcedureColumns**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) procedureNamePattern,  
 [String](http://docs.google.com/java/lang/String.html) columnNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the given catalog's stored procedure parameter and result columns.

Only descriptions matching the schema, procedure and parameter name criteria are returned. They are ordered by PROCEDURE\_CAT, PROCEDURE\_SCHEM, PROCEDURE\_NAME and SPECIFIC\_NAME. Within this, the return value, if any, is first. Next are the parameter descriptions in call order. The column descriptions follow in column number order.

Each row in the ResultSet is a parameter description or column description with the following fields:

1. **PROCEDURE\_CAT** String => procedure catalog (may be null)
2. **PROCEDURE\_SCHEM** String => procedure schema (may be null)
3. **PROCEDURE\_NAME** String => procedure name
4. **COLUMN\_NAME** String => column/parameter name
5. **COLUMN\_TYPE** Short => kind of column/parameter:
   * procedureColumnUnknown - nobody knows
   * procedureColumnIn - IN parameter
   * procedureColumnInOut - INOUT parameter
   * procedureColumnOut - OUT parameter
   * procedureColumnReturn - procedure return value
   * procedureColumnResult - result column in ResultSet
6. **DATA\_TYPE** int => SQL type from java.sql.Types
7. **TYPE\_NAME** String => SQL type name, for a UDT type the type name is fully qualified
8. **PRECISION** int => precision
9. **LENGTH** int => length in bytes of data
10. **SCALE** short => scale - null is returned for data types where SCALE is not applicable.
11. **RADIX** short => radix
12. **NULLABLE** short => can it contain NULL.
    * procedureNoNulls - does not allow NULL values
    * procedureNullable - allows NULL values
    * procedureNullableUnknown - nullability unknown
13. **REMARKS** String => comment describing parameter/column
14. **COLUMN\_DEF** String => default value for the column, which should be interpreted as a string when the value is enclosed in single quotes (may be null)
    * The string NULL (not enclosed in quotes) - if NULL was specified as the default value
    * TRUNCATE (not enclosed in quotes) - if the specified default value cannot be represented without truncation
    * NULL - if a default value was not specified
15. **SQL\_DATA\_TYPE** int => reserved for future use
16. **SQL\_DATETIME\_SUB** int => reserved for future use
17. **CHAR\_OCTET\_LENGTH** int => the maximum length of binary and character based columns. For any other datatype the returned value is a NULL
18. **ORDINAL\_POSITION** int => the ordinal position, starting from 1, for the input and output parameters for a procedure. A value of 0 is returned if this row describes the procedure's return value. For result set columns, it is the ordinal position of the column in the result set starting from 1. If there are multiple result sets, the column ordinal positions are implementation defined.
19. **IS\_NULLABLE** String => ISO rules are used to determine the nullability for a column.
    * YES --- if the parameter can include NULLs
    * NO --- if the parameter cannot include NULLs
    * empty string --- if the nullability for the parameter is unknown
20. **SPECIFIC\_NAME** String => the name which uniquely identifies this procedure within its schema.

**Note:** Some databases may not return the column descriptions for a procedure.

The PRECISION column represents the specified column size for the given column. For numeric data, this is the maximum precision. For character data, this is the length in characters. For datetime datatypes, this is the length in characters of the String representation (assuming the maximum allowed precision of the fractional seconds component). For binary data, this is the length in bytes. For the ROWID datatype, this is the length in bytes. Null is returned for data types where the column size is not applicable.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchprocedureNamePattern - a procedure name pattern; must match the procedure name as it is stored in the databasecolumnNamePattern - a column name pattern; must match the column name as it is stored in the database **Returns:**ResultSet - each row describes a stored procedure parameter or column **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getTables

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getTables**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) tableNamePattern,  
 [String](http://docs.google.com/java/lang/String.html)[] types)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the tables available in the given catalog. Only table descriptions matching the catalog, schema, table name and type criteria are returned. They are ordered by TABLE\_TYPE, TABLE\_CAT, TABLE\_SCHEM and TABLE\_NAME.

Each table description has the following columns:

1. **TABLE\_CAT** String => table catalog (may be null)
2. **TABLE\_SCHEM** String => table schema (may be null)
3. **TABLE\_NAME** String => table name
4. **TABLE\_TYPE** String => table type. Typical types are "TABLE", "VIEW", "SYSTEM TABLE", "GLOBAL TEMPORARY", "LOCAL TEMPORARY", "ALIAS", "SYNONYM".
5. **REMARKS** String => explanatory comment on the table
6. **TYPE\_CAT** String => the types catalog (may be null)
7. **TYPE\_SCHEM** String => the types schema (may be null)
8. **TYPE\_NAME** String => type name (may be null)
9. **SELF\_REFERENCING\_COL\_NAME** String => name of the designated "identifier" column of a typed table (may be null)
10. **REF\_GENERATION** String => specifies how values in SELF\_REFERENCING\_COL\_NAME are created. Values are "SYSTEM", "USER", "DERIVED". (may be null)

**Note:** Some databases may not return information for all tables.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtableNamePattern - a table name pattern; must match the table name as it is stored in the databasetypes - a list of table types, which must be from the list of table types returned from [getTableTypes()](http://docs.google.com/java/sql/DatabaseMetaData.html#getTableTypes()),to include; null returns all types **Returns:**ResultSet - each row is a table description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getSchemas

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

Retrieves the schema names available in this database. The results are ordered by TABLE\_CATALOG and TABLE\_SCHEM.

The schema columns are:

1. **TABLE\_SCHEM** String => schema name
2. **TABLE\_CATALOG** String => catalog name (may be null)

**Returns:**a ResultSet object in which each row is a schema description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getCatalogs

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

Retrieves the catalog names available in this database. The results are ordered by catalog name.

The catalog column is:

1. **TABLE\_CAT** String => catalog name

**Returns:**a ResultSet object in which each row has a single String column that is a catalog name **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getTableTypes

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

Retrieves the table types available in this database. The results are ordered by table type.

The table type is:

1. **TABLE\_TYPE** String => table type. Typical types are "TABLE", "VIEW", "SYSTEM TABLE", "GLOBAL TEMPORARY", "LOCAL TEMPORARY", "ALIAS", "SYNONYM".

**Returns:**a ResultSet object in which each row has a single String column that is a table type **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getColumns

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getColumns**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) tableNamePattern,  
 [String](http://docs.google.com/java/lang/String.html) columnNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of table columns available in the specified catalog.

Only column descriptions matching the catalog, schema, table and column name criteria are returned. They are ordered by TABLE\_CAT,TABLE\_SCHEM, TABLE\_NAME, and ORDINAL\_POSITION.

Each column description has the following columns:

1. **TABLE\_CAT** String => table catalog (may be null)
2. **TABLE\_SCHEM** String => table schema (may be null)
3. **TABLE\_NAME** String => table name
4. **COLUMN\_NAME** String => column name
5. **DATA\_TYPE** int => SQL type from java.sql.Types
6. **TYPE\_NAME** String => Data source dependent type name, for a UDT the type name is fully qualified
7. **COLUMN\_SIZE** int => column size.
8. **BUFFER\_LENGTH** is not used.
9. **DECIMAL\_DIGITS** int => the number of fractional digits. Null is returned for data types where DECIMAL\_DIGITS is not applicable.
10. **NUM\_PREC\_RADIX** int => Radix (typically either 10 or 2)
11. **NULLABLE** int => is NULL allowed.
    * columnNoNulls - might not allow NULL values
    * columnNullable - definitely allows NULL values
    * columnNullableUnknown - nullability unknown
12. **REMARKS** String => comment describing column (may be null)
13. **COLUMN\_DEF** String => default value for the column, which should be interpreted as a string when the value is enclosed in single quotes (may be null)
14. **SQL\_DATA\_TYPE** int => unused
15. **SQL\_DATETIME\_SUB** int => unused
16. **CHAR\_OCTET\_LENGTH** int => for char types the maximum number of bytes in the column
17. **ORDINAL\_POSITION** int => index of column in table (starting at 1)
18. **IS\_NULLABLE** String => ISO rules are used to determine the nullability for a column.
    * YES --- if the parameter can include NULLs
    * NO --- if the parameter cannot include NULLs
    * empty string --- if the nullability for the parameter is unknown
19. **SCOPE\_CATLOG** String => catalog of table that is the scope of a reference attribute (null if DATA\_TYPE isn't REF)
20. **SCOPE\_SCHEMA** String => schema of table that is the scope of a reference attribute (null if the DATA\_TYPE isn't REF)
21. **SCOPE\_TABLE** String => table name that this the scope of a reference attribure (null if the DATA\_TYPE isn't REF)
22. **SOURCE\_DATA\_TYPE** short => source type of a distinct type or user-generated Ref type, SQL type from java.sql.Types (null if DATA\_TYPE isn't DISTINCT or user-generated REF)
23. **IS\_AUTOINCREMENT** String => Indicates whether this column is auto incremented
    * YES --- if the column is auto incremented
    * NO --- if the column is not auto incremented
    * empty string --- if it cannot be determined whether the column is auto incremented parameter is unknown

The COLUMN\_SIZE column the specified column size for the given column. For numeric data, this is the maximum precision. For character data, this is the length in characters. For datetime datatypes, this is the length in characters of the String representation (assuming the maximum allowed precision of the fractional seconds component). For binary data, this is the length in bytes. For the ROWID datatype, this is the length in bytes. Null is returned for data types where the column size is not applicable.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtableNamePattern - a table name pattern; must match the table name as it is stored in the databasecolumnNamePattern - a column name pattern; must match the column name as it is stored in the database **Returns:**ResultSet - each row is a column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getColumnPrivileges

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getColumnPrivileges**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table,  
 [String](http://docs.google.com/java/lang/String.html) columnNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the access rights for a table's columns.

Only privileges matching the column name criteria are returned. They are ordered by COLUMN\_NAME and PRIVILEGE.

Each privilige description has the following columns:

1. **TABLE\_CAT** String => table catalog (may be null)
2. **TABLE\_SCHEM** String => table schema (may be null)
3. **TABLE\_NAME** String => table name
4. **COLUMN\_NAME** String => column name
5. **GRANTOR** String => grantor of access (may be null)
6. **GRANTEE** String => grantee of access
7. **PRIVILEGE** String => name of access (SELECT, INSERT, UPDATE, REFRENCES, ...)
8. **IS\_GRANTABLE** String => "YES" if grantee is permitted to grant to others; "NO" if not; null if unknown

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in the databasecolumnNamePattern - a column name pattern; must match the column name as it is stored in the database **Returns:**ResultSet - each row is a column privilege description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getTablePrivileges

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getTablePrivileges**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) tableNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the access rights for each table available in a catalog. Note that a table privilege applies to one or more columns in the table. It would be wrong to assume that this privilege applies to all columns (this may be true for some systems but is not true for all.)

Only privileges matching the schema and table name criteria are returned. They are ordered by TABLE\_CAT, TABLE\_SCHEM, TABLE\_NAME, and PRIVILEGE.

Each privilige description has the following columns:

1. **TABLE\_CAT** String => table catalog (may be null)
2. **TABLE\_SCHEM** String => table schema (may be null)
3. **TABLE\_NAME** String => table name
4. **GRANTOR** String => grantor of access (may be null)
5. **GRANTEE** String => grantee of access
6. **PRIVILEGE** String => name of access (SELECT, INSERT, UPDATE, REFRENCES, ...)
7. **IS\_GRANTABLE** String => "YES" if grantee is permitted to grant to others; "NO" if not; null if unknown

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtableNamePattern - a table name pattern; must match the table name as it is stored in the database **Returns:**ResultSet - each row is a table privilege description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getBestRowIdentifier

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getBestRowIdentifier**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table,  
 int scope,  
 boolean nullable)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of a table's optimal set of columns that uniquely identifies a row. They are ordered by SCOPE.

Each column description has the following columns:

1. **SCOPE** short => actual scope of result
   * bestRowTemporary - very temporary, while using row
   * bestRowTransaction - valid for remainder of current transaction
   * bestRowSession - valid for remainder of current session
2. **COLUMN\_NAME** String => column name
3. **DATA\_TYPE** int => SQL data type from java.sql.Types
4. **TYPE\_NAME** String => Data source dependent type name, for a UDT the type name is fully qualified
5. **COLUMN\_SIZE** int => precision
6. **BUFFER\_LENGTH** int => not used
7. **DECIMAL\_DIGITS** short => scale - Null is returned for data types where DECIMAL\_DIGITS is not applicable.
8. **PSEUDO\_COLUMN** short => is this a pseudo column like an Oracle ROWID
   * bestRowUnknown - may or may not be pseudo column
   * bestRowNotPseudo - is NOT a pseudo column
   * bestRowPseudo - is a pseudo column

The COLUMN\_SIZE column represents the specified column size for the given column. For numeric data, this is the maximum precision. For character data, this is the length in characters. For datetime datatypes, this is the length in characters of the String representation (assuming the maximum allowed precision of the fractional seconds component). For binary data, this is the length in bytes. For the ROWID datatype, this is the length in bytes. Null is returned for data types where the column size is not applicable.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in the databasescope - the scope of interest; use same values as SCOPEnullable - include columns that are nullable. **Returns:**ResultSet - each row is a column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getVersionColumns

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getVersionColumns**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of a table's columns that are automatically updated when any value in a row is updated. They are unordered.

Each column description has the following columns:

1. **SCOPE** short => is not used
2. **COLUMN\_NAME** String => column name
3. **DATA\_TYPE** int => SQL data type from java.sql.Types
4. **TYPE\_NAME** String => Data source-dependent type name
5. **COLUMN\_SIZE** int => precision
6. **BUFFER\_LENGTH** int => length of column value in bytes
7. **DECIMAL\_DIGITS** short => scale - Null is returned for data types where DECIMAL\_DIGITS is not applicable.
8. **PSEUDO\_COLUMN** short => whether this is pseudo column like an Oracle ROWID
   * versionColumnUnknown - may or may not be pseudo column
   * versionColumnNotPseudo - is NOT a pseudo column
   * versionColumnPseudo - is a pseudo column

The COLUMN\_SIZE column represents the specified column size for the given column. For numeric data, this is the maximum precision. For character data, this is the length in characters. For datetime datatypes, this is the length in characters of the String representation (assuming the maximum allowed precision of the fractional seconds component). For binary data, this is the length in bytes. For the ROWID datatype, this is the length in bytes. Null is returned for data types where the column size is not applicable.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in the database **Returns:**a ResultSet object in which each row is a column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getPrimaryKeys

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getPrimaryKeys**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the given table's primary key columns. They are ordered by COLUMN\_NAME.

Each primary key column description has the following columns:

1. **TABLE\_CAT** String => table catalog (may be null)
2. **TABLE\_SCHEM** String => table schema (may be null)
3. **TABLE\_NAME** String => table name
4. **COLUMN\_NAME** String => column name
5. **KEY\_SEQ** short => sequence number within primary key( a value of 1 represents the first column of the primary key, a value of 2 would represent the second column within the primary key).
6. **PK\_NAME** String => primary key name (may be null)

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in the database **Returns:**ResultSet - each row is a primary key column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getImportedKeys

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getImportedKeys**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the primary key columns that are referenced by the given table's foreign key columns (the primary keys imported by a table). They are ordered by PKTABLE\_CAT, PKTABLE\_SCHEM, PKTABLE\_NAME, and KEY\_SEQ.

Each primary key column description has the following columns:

1. **PKTABLE\_CAT** String => primary key table catalog being imported (may be null)
2. **PKTABLE\_SCHEM** String => primary key table schema being imported (may be null)
3. **PKTABLE\_NAME** String => primary key table name being imported
4. **PKCOLUMN\_NAME** String => primary key column name being imported
5. **FKTABLE\_CAT** String => foreign key table catalog (may be null)
6. **FKTABLE\_SCHEM** String => foreign key table schema (may be null)
7. **FKTABLE\_NAME** String => foreign key table name
8. **FKCOLUMN\_NAME** String => foreign key column name
9. **KEY\_SEQ** short => sequence number within a foreign key( a value of 1 represents the first column of the foreign key, a value of 2 would represent the second column within the foreign key).
10. **UPDATE\_RULE** short => What happens to a foreign key when the primary key is updated:
    * importedNoAction - do not allow update of primary key if it has been imported
    * importedKeyCascade - change imported key to agree with primary key update
    * importedKeySetNull - change imported key to NULL if its primary key has been updated
    * importedKeySetDefault - change imported key to default values if its primary key has been updated
    * importedKeyRestrict - same as importedKeyNoAction (for ODBC 2.x compatibility)
11. **DELETE\_RULE** short => What happens to the foreign key when primary is deleted.
    * importedKeyNoAction - do not allow delete of primary key if it has been imported
    * importedKeyCascade - delete rows that import a deleted key
    * importedKeySetNull - change imported key to NULL if its primary key has been deleted
    * importedKeyRestrict - same as importedKeyNoAction (for ODBC 2.x compatibility)
    * importedKeySetDefault - change imported key to default if its primary key has been deleted
12. **FK\_NAME** String => foreign key name (may be null)
13. **PK\_NAME** String => primary key name (may be null)
14. **DEFERRABILITY** short => can the evaluation of foreign key constraints be deferred until commit
    * importedKeyInitiallyDeferred - see SQL92 for definition
    * importedKeyInitiallyImmediate - see SQL92 for definition
    * importedKeyNotDeferrable - see SQL92 for definition

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in the database **Returns:**ResultSet - each row is a primary key column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getExportedKeys(java.lang.String, java.lang.String, java.lang.String)](http://docs.google.com/java/sql/DatabaseMetaData.html#getExportedKeys(java.lang.String,%20java.lang.String,%20java.lang.String))

### getExportedKeys

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getExportedKeys**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the foreign key columns that reference the given table's primary key columns (the foreign keys exported by a table). They are ordered by FKTABLE\_CAT, FKTABLE\_SCHEM, FKTABLE\_NAME, and KEY\_SEQ.

Each foreign key column description has the following columns:

1. **PKTABLE\_CAT** String => primary key table catalog (may be null)
2. **PKTABLE\_SCHEM** String => primary key table schema (may be null)
3. **PKTABLE\_NAME** String => primary key table name
4. **PKCOLUMN\_NAME** String => primary key column name
5. **FKTABLE\_CAT** String => foreign key table catalog (may be null) being exported (may be null)
6. **FKTABLE\_SCHEM** String => foreign key table schema (may be null) being exported (may be null)
7. **FKTABLE\_NAME** String => foreign key table name being exported
8. **FKCOLUMN\_NAME** String => foreign key column name being exported
9. **KEY\_SEQ** short => sequence number within foreign key( a value of 1 represents the first column of the foreign key, a value of 2 would represent the second column within the foreign key).
10. **UPDATE\_RULE** short => What happens to foreign key when primary is updated:
    * importedNoAction - do not allow update of primary key if it has been imported
    * importedKeyCascade - change imported key to agree with primary key update
    * importedKeySetNull - change imported key to NULL if its primary key has been updated
    * importedKeySetDefault - change imported key to default values if its primary key has been updated
    * importedKeyRestrict - same as importedKeyNoAction (for ODBC 2.x compatibility)
11. **DELETE\_RULE** short => What happens to the foreign key when primary is deleted.
    * importedKeyNoAction - do not allow delete of primary key if it has been imported
    * importedKeyCascade - delete rows that import a deleted key
    * importedKeySetNull - change imported key to NULL if its primary key has been deleted
    * importedKeyRestrict - same as importedKeyNoAction (for ODBC 2.x compatibility)
    * importedKeySetDefault - change imported key to default if its primary key has been deleted
12. **FK\_NAME** String => foreign key name (may be null)
13. **PK\_NAME** String => primary key name (may be null)
14. **DEFERRABILITY** short => can the evaluation of foreign key constraints be deferred until commit
    * importedKeyInitiallyDeferred - see SQL92 for definition
    * importedKeyInitiallyImmediate - see SQL92 for definition
    * importedKeyNotDeferrable - see SQL92 for definition

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in this database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in this database **Returns:**a ResultSet object in which each row is a foreign key column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getImportedKeys(java.lang.String, java.lang.String, java.lang.String)](http://docs.google.com/java/sql/DatabaseMetaData.html#getImportedKeys(java.lang.String,%20java.lang.String,%20java.lang.String))

### getCrossReference

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getCrossReference**([String](http://docs.google.com/java/lang/String.html) parentCatalog,  
 [String](http://docs.google.com/java/lang/String.html) parentSchema,  
 [String](http://docs.google.com/java/lang/String.html) parentTable,  
 [String](http://docs.google.com/java/lang/String.html) foreignCatalog,  
 [String](http://docs.google.com/java/lang/String.html) foreignSchema,  
 [String](http://docs.google.com/java/lang/String.html) foreignTable)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the foreign key columns in the given foreign key table that reference the primary key or the columns representing a unique constraint of the parent table (could be the same or a different table). The number of columns returned from the parent table must match the number of columns that make up the foreign key. They are ordered by FKTABLE\_CAT, FKTABLE\_SCHEM, FKTABLE\_NAME, and KEY\_SEQ.

Each foreign key column description has the following columns:

1. **PKTABLE\_CAT** String => parent key table catalog (may be null)
2. **PKTABLE\_SCHEM** String => parent key table schema (may be null)
3. **PKTABLE\_NAME** String => parent key table name
4. **PKCOLUMN\_NAME** String => parent key column name
5. **FKTABLE\_CAT** String => foreign key table catalog (may be null) being exported (may be null)
6. **FKTABLE\_SCHEM** String => foreign key table schema (may be null) being exported (may be null)
7. **FKTABLE\_NAME** String => foreign key table name being exported
8. **FKCOLUMN\_NAME** String => foreign key column name being exported
9. **KEY\_SEQ** short => sequence number within foreign key( a value of 1 represents the first column of the foreign key, a value of 2 would represent the second column within the foreign key).
10. **UPDATE\_RULE** short => What happens to foreign key when parent key is updated:
    * importedNoAction - do not allow update of parent key if it has been imported
    * importedKeyCascade - change imported key to agree with parent key update
    * importedKeySetNull - change imported key to NULL if its parent key has been updated
    * importedKeySetDefault - change imported key to default values if its parent key has been updated
    * importedKeyRestrict - same as importedKeyNoAction (for ODBC 2.x compatibility)
11. **DELETE\_RULE** short => What happens to the foreign key when parent key is deleted.
    * importedKeyNoAction - do not allow delete of parent key if it has been imported
    * importedKeyCascade - delete rows that import a deleted key
    * importedKeySetNull - change imported key to NULL if its primary key has been deleted
    * importedKeyRestrict - same as importedKeyNoAction (for ODBC 2.x compatibility)
    * importedKeySetDefault - change imported key to default if its parent key has been deleted
12. **FK\_NAME** String => foreign key name (may be null)
13. **PK\_NAME** String => parent key name (may be null)
14. **DEFERRABILITY** short => can the evaluation of foreign key constraints be deferred until commit
    * importedKeyInitiallyDeferred - see SQL92 for definition
    * importedKeyInitiallyImmediate - see SQL92 for definition
    * importedKeyNotDeferrable - see SQL92 for definition

**Parameters:**parentCatalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means drop catalog name from the selection criteriaparentSchema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means drop schema name from the selection criteriaparentTable - the name of the table that exports the key; must match the table name as it is stored in the databaseforeignCatalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means drop catalog name from the selection criteriaforeignSchema - a schema name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means drop schema name from the selection criteriaforeignTable - the name of the table that imports the key; must match the table name as it is stored in the database **Returns:**ResultSet - each row is a foreign key column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**See Also:**[getImportedKeys(java.lang.String, java.lang.String, java.lang.String)](http://docs.google.com/java/sql/DatabaseMetaData.html#getImportedKeys(java.lang.String,%20java.lang.String,%20java.lang.String))

### getTypeInfo

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

Retrieves a description of all the data types supported by this database. They are ordered by DATA\_TYPE and then by how closely the data type maps to the corresponding JDBC SQL type.

If the database supports SQL distinct types, then getTypeInfo() will return a single row with a TYPE\_NAME of DISTINCT and a DATA\_TYPE of Types.DISTINCT. If the database supports SQL structured types, then getTypeInfo() will return a single row with a TYPE\_NAME of STRUCT and a DATA\_TYPE of Types.STRUCT.

If SQL distinct or structured types are supported, then information on the individual types may be obtained from the getUDTs() method.

Each type description has the following columns:

1. **TYPE\_NAME** String => Type name
2. **DATA\_TYPE** int => SQL data type from java.sql.Types
3. **PRECISION** int => maximum precision
4. **LITERAL\_PREFIX** String => prefix used to quote a literal (may be null)
5. **LITERAL\_SUFFIX** String => suffix used to quote a literal (may be null)
6. **CREATE\_PARAMS** String => parameters used in creating the type (may be null)
7. **NULLABLE** short => can you use NULL for this type.
   * typeNoNulls - does not allow NULL values
   * typeNullable - allows NULL values
   * typeNullableUnknown - nullability unknown
8. **CASE\_SENSITIVE** boolean=> is it case sensitive.
9. **SEARCHABLE** short => can you use "WHERE" based on this type:
   * typePredNone - No support
   * typePredChar - Only supported with WHERE .. LIKE
   * typePredBasic - Supported except for WHERE .. LIKE
   * typeSearchable - Supported for all WHERE ..
10. **UNSIGNED\_ATTRIBUTE** boolean => is it unsigned.
11. **FIXED\_PREC\_SCALE** boolean => can it be a money value.
12. **AUTO\_INCREMENT** boolean => can it be used for an auto-increment value.
13. **LOCAL\_TYPE\_NAME** String => localized version of type name (may be null)
14. **MINIMUM\_SCALE** short => minimum scale supported
15. **MAXIMUM\_SCALE** short => maximum scale supported
16. **SQL\_DATA\_TYPE** int => unused
17. **SQL\_DATETIME\_SUB** int => unused
18. **NUM\_PREC\_RADIX** int => usually 2 or 10

The PRECISION column represents the maximum column size that the server supports for the given datatype. For numeric data, this is the maximum precision. For character data, this is the length in characters. For datetime datatypes, this is the length in characters of the String representation (assuming the maximum allowed precision of the fractional seconds component). For binary data, this is the length in bytes. For the ROWID datatype, this is the length in bytes. Null is returned for data types where the column size is not applicable.

**Returns:**a ResultSet object in which each row is an SQL type description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### getIndexInfo

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getIndexInfo**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schema,  
 [String](http://docs.google.com/java/lang/String.html) table,  
 boolean unique,  
 boolean approximate)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the given table's indices and statistics. They are ordered by NON\_UNIQUE, TYPE, INDEX\_NAME, and ORDINAL\_POSITION.

Each index column description has the following columns:

1. **TABLE\_CAT** String => table catalog (may be null)
2. **TABLE\_SCHEM** String => table schema (may be null)
3. **TABLE\_NAME** String => table name
4. **NON\_UNIQUE** boolean => Can index values be non-unique. false when TYPE is tableIndexStatistic
5. **INDEX\_QUALIFIER** String => index catalog (may be null); null when TYPE is tableIndexStatistic
6. **INDEX\_NAME** String => index name; null when TYPE is tableIndexStatistic
7. **TYPE** short => index type:
   * tableIndexStatistic - this identifies table statistics that are returned in conjuction with a table's index descriptions
   * tableIndexClustered - this is a clustered index
   * tableIndexHashed - this is a hashed index
   * tableIndexOther - this is some other style of index
8. **ORDINAL\_POSITION** short => column sequence number within index; zero when TYPE is tableIndexStatistic
9. **COLUMN\_NAME** String => column name; null when TYPE is tableIndexStatistic
10. **ASC\_OR\_DESC** String => column sort sequence, "A" => ascending, "D" => descending, may be null if sort sequence is not supported; null when TYPE is tableIndexStatistic
11. **CARDINALITY** int => When TYPE is tableIndexStatistic, then this is the number of rows in the table; otherwise, it is the number of unique values in the index.
12. **PAGES** int => When TYPE is tableIndexStatisic then this is the number of pages used for the table, otherwise it is the number of pages used for the current index.
13. **FILTER\_CONDITION** String => Filter condition, if any. (may be null)

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in this database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschema - a schema name; must match the schema name as it is stored in this database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtable - a table name; must match the table name as it is stored in this databaseunique - when true, return only indices for unique values; when false, return indices regardless of whether unique or notapproximate - when true, result is allowed to reflect approximate or out of data values; when false, results are requested to be accurate **Returns:**ResultSet - each row is an index column description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

### supportsResultSetType

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

Retrieves whether this database supports the given result set type.

**Parameters:**type - defined in java.sql.ResultSet **Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2 **See Also:**[Connection](http://docs.google.com/java/sql/Connection.html)

### supportsResultSetConcurrency

boolean **supportsResultSetConcurrency**(int type,  
 int concurrency)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves whether this database supports the given concurrency type in combination with the given result set type.

**Parameters:**type - defined in java.sql.ResultSetconcurrency - type defined in java.sql.ResultSet **Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2 **See Also:**[Connection](http://docs.google.com/java/sql/Connection.html)

### ownUpdatesAreVisible

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

Retrieves whether for the given type of ResultSet object, the result set's own updates are visible.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if updates are visible for the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### ownDeletesAreVisible

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

Retrieves whether a result set's own deletes are visible.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if deletes are visible for the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### ownInsertsAreVisible

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

Retrieves whether a result set's own inserts are visible.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if inserts are visible for the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### othersUpdatesAreVisible

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

Retrieves whether updates made by others are visible.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if updates made by others are visible for the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### othersDeletesAreVisible

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

Retrieves whether deletes made by others are visible.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if deletes made by others are visible for the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### othersInsertsAreVisible

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

Retrieves whether inserts made by others are visible.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if inserts made by others are visible for the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### updatesAreDetected

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

Retrieves whether or not a visible row update can be detected by calling the method ResultSet.rowUpdated.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if changes are detected by the result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### deletesAreDetected

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

Retrieves whether or not a visible row delete can be detected by calling the method ResultSet.rowDeleted. If the method deletesAreDetected returns false, it means that deleted rows are removed from the result set.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if deletes are detected by the given result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### insertsAreDetected

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

Retrieves whether or not a visible row insert can be detected by calling the method ResultSet.rowInserted.

**Parameters:**type - the ResultSet type; one of ResultSet.TYPE\_FORWARD\_ONLY, ResultSet.TYPE\_SCROLL\_INSENSITIVE, or ResultSet.TYPE\_SCROLL\_SENSITIVE **Returns:**true if changes are detected by the specified result set type; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### supportsBatchUpdates

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

Retrieves whether this database supports batch updates.

**Returns:**true if this database supports batch upcates; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### getUDTs

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getUDTs**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) typeNamePattern,  
 int[] types)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the user-defined types (UDTs) defined in a particular schema. Schema-specific UDTs may have type JAVA\_OBJECT, STRUCT, or DISTINCT.

Only types matching the catalog, schema, type name and type criteria are returned. They are ordered by DATA\_TYPE, TYPE\_CAT, TYPE\_SCHEM and TYPE\_NAME. The type name parameter may be a fully-qualified name. In this case, the catalog and schemaPattern parameters are ignored.

Each type description has the following columns:

1. **TYPE\_CAT** String => the type's catalog (may be null)
2. **TYPE\_SCHEM** String => type's schema (may be null)
3. **TYPE\_NAME** String => type name
4. **CLASS\_NAME** String => Java class name
5. **DATA\_TYPE** int => type value defined in java.sql.Types. One of JAVA\_OBJECT, STRUCT, or DISTINCT
6. **REMARKS** String => explanatory comment on the type
7. **BASE\_TYPE** short => type code of the source type of a DISTINCT type or the type that implements the user-generated reference type of the SELF\_REFERENCING\_COLUMN of a structured type as defined in java.sql.Types (null if DATA\_TYPE is not DISTINCT or not STRUCT with REFERENCE\_GENERATION = USER\_DEFINED)

**Note:** If the driver does not support UDTs, an empty result set is returned.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema pattern name; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtypeNamePattern - a type name pattern; must match the type name as it is stored in the database; may be a fully qualified nametypes - a list of user-defined types (JAVA\_OBJECT, STRUCT, or DISTINCT) to include; null returns all types **Returns:**ResultSet object in which each row describes a UDT **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getConnection

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

Retrieves the connection that produced this metadata object.

**Returns:**the connection that produced this metadata object **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.2

### supportsSavepoints

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

Retrieves whether this database supports savepoints.

**Returns:**true if savepoints are supported; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### supportsNamedParameters

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

Retrieves whether this database supports named parameters to callable statements.

**Returns:**true if named parameters are supported; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### supportsMultipleOpenResults

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

Retrieves whether it is possible to have multiple ResultSet objects returned from a CallableStatement object simultaneously.

**Returns:**true if a CallableStatement object can return multiple ResultSet objects simultaneously; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a datanase access error occurs**Since:** 1.4

### supportsGetGeneratedKeys

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

Retrieves whether auto-generated keys can be retrieved after a statement has been executed

**Returns:**true if auto-generated keys can be retrieved after a statement has executed; false otherwise

If true is returned, the JDBC driver must support the returning of auto-generated keys for at least SQL INSERT statements

**Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### getSuperTypes

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getSuperTypes**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) typeNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the user-defined type (UDT) hierarchies defined in a particular schema in this database. Only the immediate super type/ sub type relationship is modeled.

Only supertype information for UDTs matching the catalog, schema, and type name is returned. The type name parameter may be a fully-qualified name. When the UDT name supplied is a fully-qualified name, the catalog and schemaPattern parameters are ignored.

If a UDT does not have a direct super type, it is not listed here. A row of the ResultSet object returned by this method describes the designated UDT and a direct supertype. A row has the following columns:

1. **TYPE\_CAT** String => the UDT's catalog (may be null)
2. **TYPE\_SCHEM** String => UDT's schema (may be null)
3. **TYPE\_NAME** String => type name of the UDT
4. **SUPERTYPE\_CAT** String => the direct super type's catalog (may be null)
5. **SUPERTYPE\_SCHEM** String => the direct super type's schema (may be null)
6. **SUPERTYPE\_NAME** String => the direct super type's name

**Note:** If the driver does not support type hierarchies, an empty result set is returned.

**Parameters:**catalog - a catalog name; "" retrieves those without a catalog; null means drop catalog name from the selection criteriaschemaPattern - a schema name pattern; "" retrieves those without a schematypeNamePattern - a UDT name pattern; may be a fully-qualified name **Returns:**a ResultSet object in which a row gives information about the designated UDT **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getSuperTables

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getSuperTables**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) tableNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the table hierarchies defined in a particular schema in this database.

Only supertable information for tables matching the catalog, schema and table name are returned. The table name parameter may be a fully- qualified name, in which case, the catalog and schemaPattern parameters are ignored. If a table does not have a super table, it is not listed here. Supertables have to be defined in the same catalog and schema as the sub tables. Therefore, the type description does not need to include this information for the supertable.

Each type description has the following columns:

1. **TABLE\_CAT** String => the type's catalog (may be null)
2. **TABLE\_SCHEM** String => type's schema (may be null)
3. **TABLE\_NAME** String => type name
4. **SUPERTABLE\_NAME** String => the direct super type's name

**Note:** If the driver does not support type hierarchies, an empty result set is returned.

**Parameters:**catalog - a catalog name; "" retrieves those without a catalog; null means drop catalog name from the selection criteriaschemaPattern - a schema name pattern; "" retrieves those without a schematableNamePattern - a table name pattern; may be a fully-qualified name **Returns:**a ResultSet object in which each row is a type description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getAttributes

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getAttributes**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) typeNamePattern,  
 [String](http://docs.google.com/java/lang/String.html) attributeNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the given attribute of the given type for a user-defined type (UDT) that is available in the given schema and catalog.

Descriptions are returned only for attributes of UDTs matching the catalog, schema, type, and attribute name criteria. They are ordered by TYPE\_CAT, TYPE\_SCHEM, TYPE\_NAME and ORDINAL\_POSITION. This description does not contain inherited attributes.

The ResultSet object that is returned has the following columns:

1. **TYPE\_CAT** String => type catalog (may be null)
2. **TYPE\_SCHEM** String => type schema (may be null)
3. **TYPE\_NAME** String => type name
4. **ATTR\_NAME** String => attribute name
5. **DATA\_TYPE** int => attribute type SQL type from java.sql.Types
6. **ATTR\_TYPE\_NAME** String => Data source dependent type name. For a UDT, the type name is fully qualified. For a REF, the type name is fully qualified and represents the target type of the reference type.
7. **ATTR\_SIZE** int => column size. For char or date types this is the maximum number of characters; for numeric or decimal types this is precision.
8. **DECIMAL\_DIGITS** int => the number of fractional digits. Null is returned for data types where DECIMAL\_DIGITS is not applicable.
9. **NUM\_PREC\_RADIX** int => Radix (typically either 10 or 2)
10. **NULLABLE** int => whether NULL is allowed
    * attributeNoNulls - might not allow NULL values
    * attributeNullable - definitely allows NULL values
    * attributeNullableUnknown - nullability unknown
11. **REMARKS** String => comment describing column (may be null)
12. **ATTR\_DEF** String => default value (may be null)
13. **SQL\_DATA\_TYPE** int => unused
14. **SQL\_DATETIME\_SUB** int => unused
15. **CHAR\_OCTET\_LENGTH** int => for char types the maximum number of bytes in the column
16. **ORDINAL\_POSITION** int => index of the attribute in the UDT (starting at 1)
17. **IS\_NULLABLE** String => ISO rules are used to determine the nullability for a attribute.
    * YES --- if the attribute can include NULLs
    * NO --- if the attribute cannot include NULLs
    * empty string --- if the nullability for the attribute is unknown
18. **SCOPE\_CATALOG** String => catalog of table that is the scope of a reference attribute (null if DATA\_TYPE isn't REF)
19. **SCOPE\_SCHEMA** String => schema of table that is the scope of a reference attribute (null if DATA\_TYPE isn't REF)
20. **SCOPE\_TABLE** String => table name that is the scope of a reference attribute (null if the DATA\_TYPE isn't REF)
21. **SOURCE\_DATA\_TYPE** short => source type of a distinct type or user-generated Ref type,SQL type from java.sql.Types (null if DATA\_TYPE isn't DISTINCT or user-generated REF)

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchtypeNamePattern - a type name pattern; must match the type name as it is stored in the databaseattributeNamePattern - an attribute name pattern; must match the attribute name as it is declared in the database **Returns:**a ResultSet object in which each row is an attribute description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### supportsResultSetHoldability

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

Retrieves whether this database supports the given result set holdability.

**Parameters:**holdability - one of the following constants: ResultSet.HOLD\_CURSORS\_OVER\_COMMIT or ResultSet.CLOSE\_CURSORS\_AT\_COMMIT **Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4 **See Also:**[Connection](http://docs.google.com/java/sql/Connection.html)

### getResultSetHoldability

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

Retrieves this database's default holdability for ResultSet objects.

**Returns:**the default holdability; either ResultSet.HOLD\_CURSORS\_OVER\_COMMIT or ResultSet.CLOSE\_CURSORS\_AT\_COMMIT **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### getDatabaseMajorVersion

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

Retrieves the major version number of the underlying database.

**Returns:**the underlying database's major version **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### getDatabaseMinorVersion

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

Retrieves the minor version number of the underlying database.

**Returns:**underlying database's minor version **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### getJDBCMajorVersion

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

Retrieves the major JDBC version number for this driver.

**Returns:**JDBC version major number **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### getJDBCMinorVersion

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

Retrieves the minor JDBC version number for this driver.

**Returns:**JDBC version minor number **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### getSQLStateType

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

Indicates whether the SQLSTATE returned by SQLException.getSQLState is X/Open (now known as Open Group) SQL CLI or SQL:2003.

**Returns:**the type of SQLSTATE; one of: sqlStateXOpen or sqlStateSQL **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### locatorsUpdateCopy

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

Indicates whether updates made to a LOB are made on a copy or directly to the LOB.

**Returns:**true if updates are made to a copy of the LOB; false if updates are made directly to the LOB **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.4

### supportsStatementPooling

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

Retrieves whether this database supports statement pooling.

**Returns:**true if so; false otherwise **Throws:** SQLExcpetion - if a database access error occurs [SQLException](http://docs.google.com/java/sql/SQLException.html)**Since:** 1.4

### getRowIdLifetime

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

Indicates whether or not this data source supports the SQL ROWID type, and if so the lifetime for which a RowId object remains valid.

The returned int values have the following relationship:

ROWID\_UNSUPPORTED < ROWID\_VALID\_OTHER < ROWID\_VALID\_TRANSACTION  
 < ROWID\_VALID\_SESSION < ROWID\_VALID\_FOREVER

so conditional logic such as

if (metadata.getRowIdLifetime() > DatabaseMetaData.ROWID\_VALID\_TRANSACTION)

can be used. Valid Forever means valid across all Sessions, and valid for a Session means valid across all its contained Transactions.

**Returns:**the status indicating the lifetime of a RowId **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.6

### getSchemas

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getSchemas**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves the schema names available in this database. The results are ordered by TABLE\_CATALOG and TABLE\_SCHEM.

The schema columns are:

1. **TABLE\_SCHEM** String => schema name
2. **TABLE\_CATALOG** String => catalog name (may be null)

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database;"" retrieves those without a catalog; null means catalog name should not be used to narrow down the search.schemaPattern - a schema name; must match the schema name as it is stored in the database; null means schema name should not be used to narrow down the search. **Returns:**a ResultSet object in which each row is a schema description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.6 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### supportsStoredFunctionsUsingCallSyntax

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

Retrieves whether this database supports invoking user-defined or vendor functions using the stored procedure escape syntax.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.6

### autoCommitFailureClosesAllResultSets

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

Retrieves whether a SQLException while autoCommit is true inidcates that all open ResultSets are closed, even ones that are holdable. When a SQLException occurs while autocommit is true, it is vendor specific whether the JDBC driver responds with a commit operation, a rollback operation, or by doing neither a commit nor a rollback. A potential result of this difference is in whether or not holdable ResultSets are closed.

**Returns:**true if so; false otherwise **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.6

### getClientInfoProperties

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

Retrieves a list of the client info properties that the driver supports. The result set contains the following columns

1. **NAME** String=> The name of the client info property
2. **MAX\_LEN** int=> The maximum length of the value for the property
3. **DEFAULT\_VALUE** String=> The default value of the property
4. **DESCRIPTION** String=> A description of the property. This will typically contain information as to where this property is stored in the database.

The ResultSet is sorted by the NAME column

**Returns:**A ResultSet object; each row is a supported client info property

**Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs

**Since:** 1.6

### getFunctions

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getFunctions**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) functionNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the system and user functions available in the given catalog.

Only system and user function descriptions matching the schema and function name criteria are returned. They are ordered by FUNCTION\_CAT, FUNCTION\_SCHEM, FUNCTION\_NAME and SPECIFIC\_ NAME.

Each function description has the the following columns:

1. **FUNCTION\_CAT** String => function catalog (may be null)
2. **FUNCTION\_SCHEM** String => function schema (may be null)
3. **FUNCTION\_NAME** String => function name. This is the name used to invoke the function
4. **REMARKS** String => explanatory comment on the function
5. **FUNCTION\_TYPE** short => kind of function:
   * functionResultUnknown - Cannot determine if a return value or table will be returned
   * functionNoTable- Does not return a table
   * functionReturnsTable - Returns a table
6. **SPECIFIC\_NAME** String => the name which uniquely identifies this function within its schema. This is a user specified, or DBMS generated, name that may be different then the FUNCTION\_NAME for example with overload functions

A user may not have permission to execute any of the functions that are returned by getFunctions

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchfunctionNamePattern - a function name pattern; must match the function name as it is stored in the database **Returns:**ResultSet - each row is a function description **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.6 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

### getFunctionColumns

[ResultSet](http://docs.google.com/java/sql/ResultSet.html) **getFunctionColumns**([String](http://docs.google.com/java/lang/String.html) catalog,  
 [String](http://docs.google.com/java/lang/String.html) schemaPattern,  
 [String](http://docs.google.com/java/lang/String.html) functionNamePattern,  
 [String](http://docs.google.com/java/lang/String.html) columnNamePattern)  
 throws [SQLException](http://docs.google.com/java/sql/SQLException.html)

Retrieves a description of the given catalog's system or user function parameters and return type.

Only descriptions matching the schema, function and parameter name criteria are returned. They are ordered by FUNCTION\_CAT, FUNCTION\_SCHEM, FUNCTION\_NAME and SPECIFIC\_ NAME. Within this, the return value, if any, is first. Next are the parameter descriptions in call order. The column descriptions follow in column number order.

Each row in the ResultSet is a parameter description, column description or return type description with the following fields:

1. **FUNCTION\_CAT** String => function catalog (may be null)
2. **FUNCTION\_SCHEM** String => function schema (may be null)
3. **FUNCTION\_NAME** String => function name. This is the name used to invoke the function
4. **COLUMN\_NAME** String => column/parameter name
5. **COLUMN\_TYPE** Short => kind of column/parameter:
   * functionColumnUnknown - nobody knows
   * functionColumnIn - IN parameter
   * functionColumnInOut - INOUT parameter
   * functionColumnOut - OUT parameter
   * functionColumnReturn - function return value
   * functionColumnResult - Indicates that the parameter or column is a column in the ResultSet
6. **DATA\_TYPE** int => SQL type from java.sql.Types
7. **TYPE\_NAME** String => SQL type name, for a UDT type the type name is fully qualified
8. **PRECISION** int => precision
9. **LENGTH** int => length in bytes of data
10. **SCALE** short => scale - null is returned for data types where SCALE is not applicable.
11. **RADIX** short => radix
12. **NULLABLE** short => can it contain NULL.
    * functionNoNulls - does not allow NULL values
    * functionNullable - allows NULL values
    * functionNullableUnknown - nullability unknown
13. **REMARKS** String => comment describing column/parameter
14. **CHAR\_OCTET\_LENGTH** int => the maximum length of binary and character based parameters or columns. For any other datatype the returned value is a NULL
15. **ORDINAL\_POSITION** int => the ordinal position, starting from 1, for the input and output parameters. A value of 0 is returned if this row describes the function's return value. For result set columns, it is the ordinal position of the column in the result set starting from 1.
16. **IS\_NULLABLE** String => ISO rules are used to determine the nullability for a parameter or column.
    * YES --- if the parameter or column can include NULLs
    * NO --- if the parameter or column cannot include NULLs
    * empty string --- if the nullability for the parameter or column is unknown
17. **SPECIFIC\_NAME** String => the name which uniquely identifies this function within its schema. This is a user specified, or DBMS generated, name that may be different then the FUNCTION\_NAME for example with overload functions

The PRECISION column represents the specified column size for the given parameter or column. For numeric data, this is the maximum precision. For character data, this is the length in characters. For datetime datatypes, this is the length in characters of the String representation (assuming the maximum allowed precision of the fractional seconds component). For binary data, this is the length in bytes. For the ROWID datatype, this is the length in bytes. Null is returned for data types where the column size is not applicable.

**Parameters:**catalog - a catalog name; must match the catalog name as it is stored in the database; "" retrieves those without a catalog; null means that the catalog name should not be used to narrow the searchschemaPattern - a schema name pattern; must match the schema name as it is stored in the database; "" retrieves those without a schema; null means that the schema name should not be used to narrow the searchfunctionNamePattern - a procedure name pattern; must match the function name as it is stored in the databasecolumnNamePattern - a parameter name pattern; must match the parameter or column name as it is stored in the database **Returns:**ResultSet - each row describes a user function parameter, column or return type **Throws:** [SQLException](http://docs.google.com/java/sql/SQLException.html) - if a database access error occurs**Since:** 1.6 **See Also:**[getSearchStringEscape()](http://docs.google.com/java/sql/DatabaseMetaData.html#getSearchStringEscape())

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/DatabaseMetaData.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/java/sql/Connection.html)   [**NEXT CLASS**](http://docs.google.com/java/sql/DataTruncation.html) | [**FRAMES**](http://docs.google.com/index.html?java/sql/DatabaseMetaData.html)    [**NO FRAMES**](http://docs.google.com/DatabaseMetaData.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#3dy6vkm) | CONSTR | [METHOD](#kgcv8k) |

[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).