程序员参考

SAP jConnect™ for JDBC 16.0

文档 ID： DC38606-01-0707100-01
最后修订日期： 2014 年 3 月

目录

SAP jConnect for JDBC

SAP® jConnect™ for JDBC 是 SAP® 高性能 JDBC 驱动程序。

SAP jConnect for JDBC

- 既是 native-protocol 或 all-Java 驱动程序，又是
- net-protocol 或 all-Java 驱动程序。

SAP jConnect 使用的协议是 Tabular Data Stream™ 5.0（TDS，版本 5），即 SAP® Adaptive Server® Enterprise (SAP® ASE) 和 SAP® Open Server™ 应用程序的本地协议。SAP jConnect 执行 JDBC 标准，提供与完整的 SAP 产品系列的最佳连接，允许访问 25 个以上的企业级系统和旧系统，其中包括：

- SAP ASE
- SAP® SQL Anywhere®
- SAP® IQ
- SAP® Replication Server®
- SAP® DirectConnect™

此外，SAP jConnect for JDBC 还能够访问 Oracle、AS/400 以及其它使用 SAP DirectConnect 的数据源。

在某些情况下，JDBC 的 SAP jConnect 实现会偏离 JDBC 规范。

另请参见

- JDBC 标准的约束与说明 （第 96 页）

Java 数据库连接 (JDBC)

Java 数据库连接 (JDBC) 由 Oracle Corporation 开发，是关于应用程序编程接口 (API) 的规范，它允许 Java 应用程序使用结构化查询语言 (SQL) 访问多个数据库管理系统。

JDBC 驱动程序管理器可以处理多个连接到不同数据库的驱动程序。

标准 JDBC API 和 JDBC 标准扩展 API 中包含一组接口，利用这些接口可以打开到数据库的连接，执行 SQL 命令并处理结果。

表 1. JDBC 接口

接口	说明
`java.sql.Driver`	定位数据库 URL 的驱动程序

接口	说明
`java.sql.Connection`	连接到特定数据库
`java.sql.Statement`	允许用户执行 SQL 语句。
`java.sql.PreparedStatement`	处理带参数的 SQL 语句
`java.sql.CallableStatement`	处理数据库存储过程调用
`java.sql.ResultSet`	获取 SQL 语句的结果
`java.sql.DatabaseMetaData`	此接口用于访问有关已获取连接的数据库的信息。
`java.sql.ResultSetMetaData`	此接口用于检索有关 ResultSet 的信息。
`javax.sql.Rowset`	处理 JDBC RowSet 实现
`javax.sql.DataSource`	处理与数据源的连接
`javax.sql.ConnectionPoolData-Source`	处理链接池

每个关系数据库管理系统都需要一个驱动程序来实现这些接口。 有如下四种类型的 JDBC 驱动程序：

- 类型 1 JDBC-ODBC 桥 – 将 JDBC 调用转换成 ODBC 调用，然后将其传递给 ODBC 驱动程序。 有些 ODBC 软件必须驻留在客户端计算机中。 有些客户端数据库代码可能也驻留在客户端计算机中。
- 类型 2 native-API partly-Java 驱动程序 – 将 JDBC 调用转换成数据库特定的调用。 该驱动程序能够直接与数据库服务器通信，同时还需要客户端计算机中的一些二进制代码。
- 类型 3 net-protocol all-Java 驱动程序 – 使用独立于 DBMS 的网络协议与中间层服务器通信。 然后，中间层网关将请求转换成供应商特定的协议。
- 类型 4 native-protocol all-Java 驱动程序 – 将 JDBC 调用转换成供应商特定的 DBMS 协议，允许客户端应用程序直接与数据库服务器通信。

有关 JDBC 及其规范的详细信息，请参见 Oracle Technology Network for Java。

编程信息

查看 SAP jConnect for JDBC 的基本组件和编程要求。

启动 SAP jConnect 驱动程序，设置连接属性，连接到数据库服务器并查看有关使用 SAP jConnect 功能的信息。 有关 JDBC 编程的信息，请转到 Oracle Technology Network for Java 中面向 Java 开发人员的资源页。

SAP jConnect 版本属性

JCONNECT_VERSION 连接属性决定了驱动程序的行为和激活的功能。

例如，SAP Adaptive Server 15.5 支持 SAP jConnect 6.05 和 7.0，但这两个版本处理 datetime 和 time 数据的方式有所不同。 在连接到 SAP Adaptive Server 15.5 时，支持微秒级时间数据精度的 SAP jConnect 7.0 使用 bigdatetime 或 bigtime，即使将目标 SAP Adaptive Server 列定义为 datetime 或 time 也是如此。 而不支持微秒级精度的 SAP jConnect 6.05 在连接到 SAP Adaptive Server 15.5 时则会始终转换 datetime 或 time 数据。

可以使用 SybDriver.setVersion 方法或 JCONNECT_VERSION 连接属性设置 SAP jConnect 版本。

SybDriver. setVersion 方法

setVersion 方法会影响由 SybDriver 对象创建的所有连接的 SAP jConnect 缺省行为。

可以多次调用 setVersion 来更改版本设置。 新连接会继承在建立连接时与版本设置相关联的行为。 在会话过程中更改版本设置不会影响当前连接。 可以使用 com.sybase.jdbcx.SybDriver.VERSION_LATEST 常量确保总是请求所使用的 SAP jConnect 驱动程序的最高可能版本值。 但是，在将版本设置为 com.sybase.jdbcx.SybDriver.VERSION_LATEST 后，如果用较新的 SAP jConnect 驱动程序替换当前的 SAP jConnect 驱动程序，可能会发现行为发生变化。

此代码示例演示如何装载 SAP jConnect 驱动程序和设置其版本：

```
import java.sql.DriverManager;
import com.sybase.jdbcx.SybDriver;
SybDriver sybDriver = (SybDriver)
   Class.forName("com.sybase.jdbc4.jdbc.SybDriver")
   .newInstance();
sybDriver.setVersion(com.sybase.jdbcx.SybDriver.
   VERSION_7);
DriverManager.registerDriver(sybDriver);
```

JCONNECT_VERSION 连接属性

使用 JCONNECT_VERSION 连接属性覆盖 SybDriver 版本设置，并为特定连接指定不同的版本设置。

请参见有效的 JCONNECT_VERSION 值以及与这些值相关联的 jConnect 特征。

表 2. 与 SAP jConnect 版本相关联的功能

JCONNECT_VERSION	功能
7.0	SAP jConnect 7.0 的行为方式与 SAP jConnect 6.05 相同，但在 7.0 中，SAP jConnect 要求为以下内容提供支持： • 来自服务器的 bigdatetime 和 bigtime SQL 数据类型。 低于 15.5 的 SAP Adaptive Server 版本将忽略该请求。 • JDBC 4.0。 • ENABLE_BULK_LOAD 的有效值为空（缺省值）、ARRAYINSERT_WITH_MIXED_STATEMENTS、ARRAYINSERT、BCP 和 LOG_BCP。
6.05	SAP jConnect 6.05 的行为方式与 SAP jConnect 6.0 相同，但在 6.05 中，SAP jConnect 要求为以下内容提供支持： • 计算列，包括元数据。 • 长标识符。 通过长标识符，您可以使用最多 255 个字节的标识符或对象名称。 长标识符适用于大多数用户定义的标识符，包括表名、列名和索引名。
6.0	SAP jConnect 6.0 的行为方式与 SAP jConnect 5.x 相同，但在 6.0 中，jConnect 要求为以下内容提供支持： • date 和 time SQL 数据类型。 低于 12.5.1 的 Adaptive Server 版本将忽略该请求。 • 来自服务器的 unichar 和 univarchar 数据类型。 低于 12.5.1 的 Adaptive Server 版本将忽略该请求。 • 来自服务器的宽表。 低于 12.5.1 的 Adaptive Server 版本将忽略该请求。 • DISABLE_UNICHAR_SENDING 的缺省值为 false。
5.0	SAP jConnect 5.x 与 SAP jConnect 4.0 的行为方式相同。

JCONNECT_VERSION	功能
4.0	SAP jConnect 4.0 的行为方式与 SAP jConnect 3.0 相同，但在 4.0 中，SAP jConnect 要求为以下内容提供支持： • LANGUAGE 连接属性的缺省值为空值。 • Statement.cancel 的缺省行为是只取消调用它的 **Statement** 对象。 此行为符合 JDBC 标准。 使用 CANCEL_ALL 可设置 Statement.cancel 的行为。 • 可以使用 JDBC 2.0 方法以列数据的形式存储和检索 Java 对象。
3.0	SAP jConnect 3.0 与 SAP jConnect 2.0 的行为方式相同，但在 3.0 中以下内容成立： • 如果 CHARSET 连接属性没有指定字符集，jConnect 将使用数据库的缺省字符集。 • CHARSET_CONVERTER 的缺省值为 CheckPureConverter 类。
2.0	• LANGUAGE 连接属性的缺省值为 us_english。 • 如果 CHARSET 连接属性没有指定字符集，则缺省字符集为 iso_1。 • CHARSET_CONVERTER 的缺省值为 TruncationConverter 类，除非 CHARSET 连接属性指定了多字节或 8 位字符集（在这种情况下，CHARSET_CONVERTER 的缺省值为 CheckPureConverter 类）。 • **Statement.cancel** 的缺省行为是取消调用它的对象，以及任何其它已开始执行并正在等待结果的 **Statement** 对象。 此行为不符合 JDBC 标准。 使用 CANCEL_ALL 可设置 Statement.cancel 的行为。

另请参见

- JDBC 4.0 规范支持 （第 86 页）
- JDBC 标准的约束与说明 （第 96 页）
- SAP jConnect 字符集转换程序 （第 38 页）
- Date 和 Time 数据类型 （第 65 页）
- JDBC 3.0 规范支持 （第 87 页）
- Adaptive Server 的宽表支持 （第 48 页）
- 将 Java 对象作为列数据存储在表中 （第 79 页）
- 使用 SAP jConnect 传递 Unicode 数据 （第 37 页）

调用 SAP jConnect 驱动程序

注册和调用 SAP jConnect，并将 SAP jConnect 添加到 `jdbc.drivers` 系统属性。

在初始化阶段，`DriverManager` 类试图装载 `jdbc.drivers` 中列出的驱动程序。这不如调用 `Class.forName` 的效率高。 可以在此属性中列出多个驱动程序，程序之间用冒号 (:) 分隔。

下面的示例代码演示如何在程序内向 `jdbc.drivers` 中添加驱动程序：

```
Properties sysProps = System.getProperties();
String drivers = "com.sybase.jdbc4.jdbc.SybDriver";
String oldDrivers =
sysProps.getProperty("jdbc.drivers");
if (oldDrivers != null)
   drivers += ":" + oldDrivers;
 sysProps.put("jdbc.drivers", drivers.toString());
```

注意： `System.getProperties` 不能用于 Java 小程序。 请改用 `Class.forName` 方法。

在 Java 6 和 JDBC 4 中，您可以使用 Java 系统属性 `jdbc.drivers` 来指定驱动程序类，例如：

```
java -Djdbc.drivers=com.sybase.jdbc4.jdbc.SybDriver UseDriver
```

无需使用 **UseDriver** 程序显式装载驱动程序：

```
public class UseDriver
{
   public static void main(String[] args)
   {
      try {
         Connection conn = java.sql.DriverManager.getConnection
            ("jdbc:sybase:Tds:localhost:5000?
USER=sa&PASSWORD=secret");
         // more code to use connection ...
      }
      catch (SQLException se){
         System.out.println("ERROR: SQLException "+se);
      }
   }
}
```

为 J2EE 服务器配置 SAP jConnect

使用 `com.sybase.jdbc4.jdbc.SybConnectionPoolDataSource` 类在 EAServer 等应用程序服务器中配置与 Adaptive Server 服务器的连接池。

`javax.sql.ConnectionPoolDataSource` 接口的 `com.sybase.jdbc4.jdbc.SybConnectionPoolDataSource` 实现可为每个连接属性提供 `getter` 和 `setter` 方法。

还可通过编程方式配置 SAP jConnect，例如：

```
private DataSource getDataSource ()
{
   SybConnectionPoolDataSource connectionPoolDataSource = new
      SybConnectionPoolDataSource();
   connectionPoolDataSource.setDatabaseName("pubs2");
   connectionPoolDataSource.setNetworkProtocol("Tds");
   connectionPoolDataSource.setServerName("localhost");
   connectionPoolDataSource.setPortNumber(5000);
   connectionPoolDataSource.setUser("sa");
   connectionPoolDataSource.setPassword(PASSWORD);
   return connectionPoolDataSource;
}
private void work () throws SQLException
{
   Connection conn = null;
   Statement stmt = null;
   DataSource ds = getDataSource();
   try {
      conn = ds.getConnection();
      stmt = conn.createStatement();
      // ...
   }
   finally {
      if (stmt != null) {
         try { stmt.close(); } catch (Exception ex) { /* ignore */ }
      }
      if (conn != null) {
         try { conn.close(); } catch (Exception ex) { /* ignore */ }
      }
   }
}
```

建立连接

使用 SAP jConnect 可建立与 SAP Adaptive Server 或 SAP SQL Anywhere 数据库的连接。

连接属性

连接属性指定登录到服务器所需的信息并定义预期的客户端和服务器行为。

连接属性名称不区分大小写。

设置连接属性

必须先设置连接属性才能连接到服务器。

通过以下任一方式设置连接属性：

- 在应用程序中使用 `DriverManager.getConnection` 方法，或，
- 在定义 URL 时设置连接属性。

注意： 在 URL 中设置的驱动程序连接属性不会覆盖在应用程序中使用 `DriverManager.getConnection` 方法设置的任何相应的连接属性。

下面的示例代码使用 `DriverManager.getConnection` 方法。 随 SAP jConnect 提供的示例程序也包含设置这些属性的示例。

```
Properties props = new Properties();
 props.put("user", "userid");
 props.put("password", "user_password");
/*
 * If the program is an applet that wants to access
 * a server that is not on the same host as the
 * web server, then it uses a proxy gateway.
 */
 props.put("proxy", "localhost:port");
/*
 * Make sure you set connection properties before
 * attempting to make a connection. You can also
 * set the properties in the URL.
 */
Connection con = DriverManager.getConnection
   ("jdbc:sybase:Tds:host:port", props);
```

当前连接设置

要查看驱动程序的当前连接设置，请使用 `Driver.getDriverPropertyInfo(String url, Properties props)`。

此代码将返回一个 `DriverPropertyInfo` 对象数组，其中包含以下内容：

- 驱动程序属性

- 驱动程序属性所基于的当前设置
- URL 和传入的属性

SAP jConnect 连接属性

SAP jConnect 的连接属性及其缺省值。

这些属性不区分大小写。

可以使用 `getClientInfo()` 和 `setClientInfo()` 标准方法按照指示动态设置这些属性。

表 3. 连接属性

属性	说明
ALTERNATE_ SERV-ER_NAME	指定镜像 SAP SQL Anywhere 环境中主数据库和辅助数据库使用的备用服务器名称。 主数据库和辅助数据库使用相同的备用服务器名称，因此客户端应用程序可以连接到当前主服务器，而无需事先知道两台服务器中的哪一台是主服务器。 JDBC URL 的语法为 `jdbc:sybase:Tds:<hostname>:<port#>/database?connection_property=val-ue;`。 但是，如果设置了 ALTERNATE_SERVER_NAME，SAP jConnect 将忽略 *hostname* 和 *port* 变量的值， 而是使用 SAP SQL Anywhere UDP 发现协议来确定当前主服务器。 有关数据库镜像的信息，请参见《SAP SQL Anywhere 服务器 - 数据库管理指南》。 也可以对未镜像的 SAP SQL Anywhere 服务器使用 `ALTERNATE_SERVER_NAME`。 不过，您将始终从单独的服务器获取相同的主机和端口值。 缺省值为空值。 此属性是静态属性。
APPLICATIONNAME	指定应用程序名称。 此属性是用户定义属性。 可对服务器端进行编程，以解释提供给此属性的值。 缺省值为空值。 此属性是静态属性。

属性	说明
BE_AS_JDBC_ COMPLIANT_AS_ POSSIBLE	调整其它属性以确保 SAP jConnect 方法的应答方式尽可能符合 JDBC 3.0 标准。 如果将该属性设置为 true，将影响（并覆盖）下列属性： • CANCEL_ALL（设置为 false） • LANGUAGE CURSOR（设置为 false） • SELECT_OPENS_CURSOR（设置为 true） • FAKE_METADATA（设置为 true） • GET_BY_NAME_USES_COLUMN_LABEL（设置为 false） 缺省值为 false。 此属性是静态属性。
CACHE_COLUMN_ METADATA	如果您重复使用 PreparedStatement 或 CallableStatement 对象执行 **SELECT** 查询，则将 CACHE_COLUMN_ METADATA 设置为 true 可以提高性能。 设置为 true 时，该语句将会记住与第一次执行该语句时所返回的 **SELECT** 查询结果关联的 ResultSet 元数据信息。 以后执行该语句时，将会重用这些元数据，而不必重新构建。 这可以通过使用更多内存来节省 CPU 时间。 在连接到 SAP Adaptive Server 15.7 ESD #1 及更高版本时，应使用 SUPPRESS_ROW_FORMAT 连接属性。 缺省值为 false。 此属性是静态属性。
PRE_CACHE_DATATYPE_INFO	如果重复使用 **Statement** 或其派生接口来获取数据类型元数据，那么将 **PRE_CACHE_DATATYPE_INFO** 设置为 true 可以提高性能。 如果将 **PRE_CACHE_DATATYPE_INFO** 设置为 true，连接时会对服务不同 ResultsetMetadata API（如 isCaseSensitive 和 isSearchable）的所有用户定义数据类型的相关信息进行高速缓存。 然后，可通过高速缓存对此信息进行后续访问。 如果 **PRE_CACHE_DATATYPE_INFO** 为 false（缺省值），SAP jConnect 不会高速缓存任何用户定义的数据类型信息。 **注意：** 根据要获取其连接的数据库中所存在的用户定义数据类型数，建立连接的时间可能会有所增加。 缺省值为 false。 此属性是动态属性。

属性	说明
CANCEL_ALL	指定 Statement.cancel 方法的行为: • 如果 CANCEL_ALL 为 false，那么调用 Statement.cancel 只会取消调用它的 Statement 对象。 因此，如果 stmtA 是 Statement 对象，stmtA.cancel 将取消执行数据库的 stmtA 中所含的 SQL 语句，但不会影响其它任何语句。 无论 stmtA 是在高速缓存中等待执行还是已开始执行并正在等待结果，都会遭到取消。 • 如果 CANCEL_ALL 为 true，那么调用 Statement.cancel 不仅会取消调用它的对象，还会取消同一连接上已开始执行并且正在等待结果的任何其它 Statement 对象。 下面的示例将 CANCEL_ALL 设置为 false。 *props* 是一个 Properties 对象，用于指定连接属性。 `props.put("CANCEL_ALL", "false");` 若要取消执行某个连接上的所有 Statement 对象，而不管其是否已开始在服务器上执行，请使用扩展方法 SybConnection.cancel。 缺省值为: • true – 针对 JCONNECT_VERSION <= “3” • false – 针对 JCONNECT_VERSION >= “4” 此属性是静态属性。

属性	说明
CAPABILITY_TIME	仅在 JCONNECT_VERSION >= 6 时使用。 此时，SAP jConnect 与支持 TIME 数据类型的服务器相连，并且所有类型为 java.sql.Time 或 escape literals {t ...} 的参数都被作为 TIME 进行处理。 低于 6.0 的 SAP jConnect 版本将这些参数作为 DATETIME 进行处理并在 java.sql.Time 参数前加上“1970-01-01”。 如果基础数据类型为 datetime 或 smalldatetime，则日期分量也会存储在数据库中。 在 SAP jConnect 6.0 或更高版本中，在处理 TIME 时，服务器会将时间转换成基础数据类型并在前面加上它自己的基准年。 这将导致旧数据与新数据之间不兼容。 如果要对 java.sql.Time 使用 datetime 或 smalldatetime 数据类型，应保留 CAPABILITY_TIME 为 false，以便实现向后兼容。 将此属性保留为 false 可强制 SAP jConnect 将 java.sql.Time 参数或 escape literals {t ...} 作为 DATETIME 进行处理，而不论服务器处理 TIME 数据类型的能力如何。 将此属性设置为 true 会使 SAP jConnect 在连接到 SAP Adaptive Server 时将 java.sql.Time 参数作为 TIME 数据类型进行处理。 如果您要使用 smalldatetime 或 datetime 列来存储时间值，SAP 建议您将此属性保留为 false。 缺省值为 false。 此属性是静态属性。
CAPABILITY_ WIDET-ABLE	如果您为了提高性能而不使用 JDBC ResultSetMetaData (如列名)，则可以将此属性设置为 false。 这将减少通过网络交换的数据，从而提高性能。 除非要使用 EAServer，否则 SAP 建议使用缺省设置。 缺省值为 false。 此属性是静态属性。

属性	说明
CHARSET	为传递给数据库的字符串指定字符集。 如果 CHARSET 值为空值，SAP jConnect 将使用服务器的缺省字符集向服务器发送字符串数据。 如果指定 CHARSET，数据库必须能够处理此格式的字符。 如果数据库不能处理此格式的字符，将生成一条消息，说明不能正确完成字符转换。 如果使用的是 SAP jConnect 6.05 或更高版本，并将 DISABLE_UNICHAR_SENDING 设置为 false，那么当客户端试图向服务器发送无法用连接所使用的字符集表示的字符时，SAP jConnect 将能够检测出来。 发生这种情况时，SAP jConnect 会将字符数据作为 unichar 数据发送给服务器，这样可使客户端能够在 unichar/univarchar 列和参数中插入 Unicode 数据。 缺省值为空值。 此属性是静态属性。
CHARSET_ CONVERTER_CLASS	指定希望 SAP jConnect 使用的字符集转换程序类。 SAP jConnect 使用来自 SybDriver.setVersion 的版本设置或随 JCONNECT_VERSION 属性传入的版本来确定要使用的缺省字符集转换程序类。 缺省值视版本而定。 此属性是静态属性。
CLASS_LOADER	该属性设置为您创建的 DynamicClassLoader 对象。 DynamicClassLoader 装载当应用程序启动时存储在数据库中但不在 CLASSPATH 中的 Java 类。 缺省值为空值。 此属性是静态属性。
CONNECTION_ FAILOVER	与 Java 命名和目录接口 (JNDI) 一起使用。 缺省值为 true。 此属性是静态属性。
CRC	当该属性设置为 true 时，返回的更新计数为累计计数，其中既包括受执行的语句直接影响的更新，也包括执行语句后调用的所有触发器。 缺省值为 false。 此属性是静态属性。

属性	说明
DATABASE	当从 SAP `interfaces` 文件获得连接信息时，使用该属性指定连接的数据库名称。 `interfaces` 文件的 URL 无法提供数据库名称。 缺省值为空值。 此属性是静态属性。
DEFAULT_QUERY_TIMEOUT	当设置该连接属性时，它将充当在该连接上创建的任何语句的缺省查询超时。 缺省值为 0（无超时）。 此属性是动态属性。
DELETE_WARNINGS_FROM_EXCEPTION_CHAIN	指定是要保留 SQLWarning 还是将其从 SQLException 链中删除。 值: • true – SAP jConnect 会将 SQLWarning 对象从 SQLException 链中删除。 • false – SAP jConnect 会将 SQLWarning 对象保留在 SQLException 链中。 缺省值为 true。 此属性是静态属性。
DISABLE_UNICHAR_SENDING	当客户端应用程序向服务器发送 `unichar` 字符（以及非 `unichar` 字符）时，会对发送到数据库的任何字符数据产生轻微的性能影响。 在 SAP jConnect 6.05 和更高版本中，此属性在缺省情况下设置为 false。 如果使用较低版本 SAP jConnect 的客户端希望向数据库发送 `unichar` 数据，则必须将此属性设置为 false。 缺省值视版本而定。 此属性是静态属性。
DISABLE_ UNPRO-CESSED_ PARAM_ WARNINGS	禁用警告。 在处理存储过程的结果时，SAP jConnect 经常读取行数据之外的返回值。 如果不处理返回值，SAP jConnect 将引发一个警告。 若要禁用这些警告（这样有助于提高性能），请将此属性设置为 true。 缺省值为 false。 此属性是静态属性。

属性	说明
DYNAMIC_PREPARE	决定是否在数据库中预编译动态 SQL 预准备语句。 缺省值为 true。 此属性是动态属性。
EARLY_BATCH_ READ_THRESHOLD	指定行数，超出该行数后，读取器线程应启动，以清除批处理的服务器响应。 如果无需较早读取，则将该值设置为 -1。 缺省值为 -1。 此属性是静态属性。
ENABLE_BULK_ LOAD	指定是否使用批量装载向数据库中插入行。 有效值为： • 空值 – 禁用批量装载。 • ARRAYINSERT_WITH_MIXED_STATEMENTS – 使用行级别日志记录启用批量装载，并允许应用程序在批量装载操作过程中执行其它语句。 • ARRAYINSERT – 使用行级别日志记录启用批量装载，但应用程序在执行批量装载操作期间无法执行其它语句。 • BCP – 使用页面级别日志记录启用批量装载，但应用程序在批量装载操作过程中无法执行其它语句。 • LOG_BCP – 与 BCP 相同，但对完整事务进行转储以供可能的完全恢复。 缺省值为空值。 此属性是动态属性。
ENABLE_LOB_ LOCA-TORS	指定 SAP jConnect 是应该创建客户端实现 LOB 还是服务器端 LOB 定位符。 有效值为： • false – SAP jConnect 使用客户端实现 LOB。 即，LOB 的所有数据均在客户端进行处理和高速缓存。 • true – 仅在自动提交设置为 false 时有效，否则，会在内部将值更改为 false。 设置为 true 后，将使用服务器定位符，而不是在客户端存储 LOB 数据。 缺省值为 false。 此属性是动态属性。

属性	说明
ENABLE_SERVER_ PACKETSIZE	指定是否将连接包大小设置为服务器建议的值。 如果为 true，驱动程序将不使用 PACKETSIZE 连接属性，并且服务器可以使用介于 512 和最大包大小之间的任何值。 如果为 false，将使用 PACKETSIZE 连接属性。 缺省值为 true。 此属性是静态属性。
ENCRYPT_ PASS-WORD	允许安全登录。 当该属性为 true 时，登录和远程站点口令都会被加密，然后再发送到服务器。 不再以明文形式发送这些口令。 ENCRYPT_PASSWORD 优先于 RETRY_WITH_NO_ENCRYP-TION。 缺省值为 false。 此属性是静态属性。
ESCAPE_ PROCESS-ING_ DEFAULT	避免处理 SQL 语句中的 JDBC 函数转义。 缺省情况下，SAP jConnect 会分析提交到数据库的所有 SQL 语句，以查找有效的 JDBC 函数转义。 如果应用程序不在其 SQL 调用中使用 JDBC 函数转义，可将此连接属性设置为 false 以避免此处理过程，这样做可以稍微改善性能。 此外，ESCAPE_PROCESSING_DEFAULT 可帮助在 SQL 语法中使用大括号的后端服务器（如 SAP® IQ）。 缺省值为 true。 此属性是静态属性。
EXECUTE_BATCH_ PAST_ERRORS	指定 SAP jConnect 是允许批处理更新操作忽略在执行各个语句时遇到的非致命错误并完成批处理更新，还是中止批处理更新操作。 有效值为： • true – 允许批处理更新操作忽略遇到的非致命错误并完成批处理更新。 • false – 在遇到非致命错误时中止批处理更新。 缺省值为 false。 此属性是静态属性。

属性	说明
EXPIRESTRING	包含许可证到期 `date`。 除 SAP jConnect 的评估副本之外，其余副本的有效期均设置为 Never。 缺省值为 never。 此属性是静态只读属性。
FAKE_METADATA	返回假元数据。 如果在调用 `ResultSetMetaData` 方法 `getCatalogName`、`getSchemaName` 和 `getTableName` 时此属性为 true，由于服务器不提供有用的元数据，调用过程将返回空字符串 ("")。 如果此属性为 false，那么调用这些方法将抛出“未实现”SQLException。 如果已启用宽表且正在使用 SAP Adaptive Server 12.5 或更高版本，则将忽略此属性设置，因为服务器提供了有用的元数据。 缺省值为 false。 此属性是静态属性。
GET_BY_NAME_ USES_COLUMN_ LABEL	提供与低于 6.0 的 SAP jConnect 版本的向后兼容性。 在 SAP Adaptive Server 12.5 和更高版本中，SAP jConnect 跟以前相比可以访问更多的元数据。 在 12.5 版之前，`column name` 和 `column alias` 代表同一数据类型。 现在，在使用 12.5 或更高版本的 SAP Adaptive Server 并启用宽表的情况下，SAP jConnect 可对这二者加以区分。 若要保持向后兼容性，请将此属性设置为 true。 如果希望调用 `getByte`、`getInt`、get* (`String columnName`) 以查看列的实际名称，请将此属性设置为 false。 缺省值为 true。 此属性是静态属性。

属性	说明
GET_COLUMN_ LABEL_FOR_NAME	保持与 SAP jConnect 5.5 或更低版本的向后兼容，其中对 `ResultMetaData.getColumnName` 的调用将返回列标签而不是列名称。 有效值为： • true – `ResultMetaData.getColumnName` 返回列标签。 • false – `ResultMetaData.getColumnName` 返回列名。 缺省值为 false。 此属性是静态属性。
GSSMANAGER_ CLASS	指定 `org.ietf.jgss.GSSManager` 类的第三方实现。 可以将此属性设置为字符串或 `GSSManager` 对象。 如果将此属性设置为字符串，其值应是第三方 GSSManager 实现的全限定类名。 如果将此属性设置为对象，该对象必须扩展 `org.ietf.jgss.GSSManager` 类。 缺省值为空值。 此属性是静态属性。
HOMOGENEOUS_ BATCH	调用 SAP Adaptive Server 优化批处理协议以加快 `PreparedStatement` 对象的批处理操作速度。 有效值为： • true – 使用优化批处理协议。 • false – 即使 SAP jConnect 连接到支持新优化批处理协议的 SAP Adaptive Server，也使用未优化的批处理协议。 缺省值为 true。 此属性是动态属性。
HOSTNAME	标识当前主机名。 缺省值为无。 最大长度是 30 个字符，如果超过则被截断至 30 个字符。 此属性是静态属性。

属性	说明
HOSTPROC	标识主机上的应用程序进程。 缺省值为无。 此属性是静态属性。
IGNORE_DONE_IN_PROC	不返回中间更新结果（如同在存储过程中），而只返回最终结果集。 缺省值为 false。 此属性是静态属性。
IGNORE_WARNINGS	指定是否检查并生成警告消息。 此属性仅检查有关将时间戳值存储为 SAP Adaptive Server `date` 和 `time` 数据类型（其精度低于 Java 时间戳）时的精度损失的警告。 有效值为: • true – SAP jConnect 不检查和生成警告消息，因此提高了性能。 • false – 缺省值，指示 SAP jConnect 检查并生成警告消息。 在将 `IGNORE_WARNINGS` 设置为 true 前，应就此类配置对应用程序产生的影响进行全面测试。 缺省值为 false。 此属性是静态属性。
IMPLICIT_CURSOR_FETCH_SIZE	将该属性与 `SELECT_OPENS_CURSOR` 属性结合使用，可强制 SAP jConnect 在发送到数据库的每个 **select** 查询上打开一个只读游标。 该游标具有此属性中所设置的值的获取大小，除非使用 `Statement.setFetchSize` 方法加以覆盖。 缺省值为 0。 此属性是静态属性。
INTERNAL_QUERY_TIMEOUT	使用该属性可设置查询超时，以供在内部创建并通过 SAP jConnect 执行的语句使用。 如果内部命令没有在适当的时间内完成，这可以防止应用程序失败。 缺省值为 0（无超时）。 此属性是动态属性。

属性	说明
IS_CLOSED_TEST	允许指定在调用 **Connection.isClosed** 时向数据库发送何种查询（如果有）。 缺省值为空值。 此属性是静态属性。
J2EE_TCK_ COMPLI-ANT	当该属性为 true 时，SAP jConnect 驱动程序会启用符合 J2EE 1.4 技术兼容包 (TCK) 测试套件的行为，这会导致性能有所下降。 因此，SAP 建议使用缺省值 false。 缺省值为 false。 此属性是静态属性。
JAVA_CHARSET_ MAPPING	指定用户定义的字符集映射，取代 SAP Adaptive Server 的缺省字符集映射。 缺省值为无。 此属性是静态属性。
JCE_PROVIDER_ CLASS	指定 RSA 加密算法中使用的 Java Cryptography Extension (JCE) 提供程序。 缺省值为捆绑的 JCE 提供程序。 此属性是静态属性。
JCONNECT_VERSION	设置特定于版本的特征。 缺省值为 7。 此属性是静态属性。
LANGUAGE	指定 SAP jConnect 和服务器中消息的显示语言。 该设置必须与 `syslanguages` 中的语言相符，因为服务器消息将根据您的当地环境中的语言设置进行本地化。 支持的语言包括：中文、美国英语、法语、德语、日语、韩语、波兰语、葡萄牙语和西班牙语。 缺省值视版本而定。 此属性是静态属性。
LANGUAGE_ CURSOR	确定 SAP jConnect 使用语言游标而不使用协议游标。 缺省值为 false。 此属性是静态属性。

属性	说明
LITERAL_PARAMS	如果为 true，那么 `setXXX` 方法在 `PreparedStatement` 接口中设置的任何参数都会在 SQL 语句执行时以文字形式插入该语句。 如果为 false，参数标记将留在 SQL 语句中，而参数值被单独发送给服务器。 缺省值为 false。 此属性是静态属性。
NEWPASSWORD	指定口令有效期处理过程中使用的新口令。 缺省值为空值。 此属性是静态属性。
OPTIMIZE_FOR_ PERFORMANCE	指定是否启用 SAP jConnect 性能增强属性。 此属性仅控制 `IGNORE_WARNINGS` 属性。 有效值为： • true – SAP jConnect 在增强性能模式下运行。 • false – 缺省值，表示 SAP jConnect 在正常模式下运行。 在将 `OPTIMIZE_FOR_PERFORMANCE` 设置为 true 前，应就此类配置对应用程序产生的影响进行全面测试。 缺省值为 false。 此属性是静态属性。
OPTIMIZE_STRING_ CONVERSIONS	指定是否启用字符串转换优化。 客户端在 SQL 预准备语句中使用字符数据类型时，此优化行为可能会提高 SAP jConnect 的性能。 有效值为： • 0 – 不启用字符串转换优化。 • 1 – 在 SAP jConnect 使用 UTF8 或服务器缺省字符集时启用字符串转换优化。 • 2 – 在所有情况下均启用字符串转换优化。 缺省值为 0。 此属性是静态属性。

属性	说明
PACKETSIZE	标识网络包大小。 如果使用的是 SAP Adaptive Server 15.0 或更高版本，SAP 建议您不要设置此属性，并让 SAP jConnect 和 SAP Adaptive Server 选用适合于您的环境的网络包大小。 缺省值为 512。 此属性是静态属性。
PASSWORD	标识登录口令。 如果使用 `getConnection(String, String, String)` 方法，将自动设置；如果使用 `getConnection(String, Props)`，将显式设置。 缺省值为无。 此属性是静态属性。
PRELOAD_JARS	包含以逗号分隔的 `.jar` 文件名列表，该列表与您指定的 `CLASS_LOADER` 相关联。 这些 `.jar` 文件在连接时装载，可供使用相同 SAP jConnect 驱动程序的任何其它连接使用。 缺省值为空值。 此属性是静态属性。
PROMPT_FOR_ NEW-PASSWORD	指定是执行透明口令更改还是提示输入新口令。 有效值为: • true – 提示手动设置新口令。 • false – SAP jConnect 检查 `NEWPASSWORD` 的值，如果不是空值，则使用该值替换到期口令。 缺省值为 false。 此属性是静态属性。
PROTOCOL_ CAP-TURE	指定用于捕获应用程序和 SAP Adaptive Server 间的 TDS 通信的文件。 缺省值为空值。 此属性是动态属性。

属性	说明
PROXY	指定网关地址。 对于 HTTP 协议，URL 是 http://host:port。 若要使用支持加密的 HTTPS 协议，URL 应为 https://host:port/servlet_alias。 缺省值为无。 此属性是静态属性。
QUERY_TIMEOUT_CANCELS_ALL	强制 SAP jConnect 在读取超时时取消连接上的所有语句。 当客户端调用 `execute()` 但由于死锁（例如，试图读取当前正在另一个事务中更新的表）而超时时，可使用此行为。 缺省值为 false。 此属性是动态属性。
RELEASE_LOCKS_ON_CURSOR_ CLOSE	指定在游标关闭时 SAP Adaptive Server 是否在隔离级别 2 和 3 释放共享只读游标锁: • false – 在游标关闭时不释放共享游标锁。 • true – 在游标关闭时释放共享游标锁。 缺省值为 false。 此属性是静态属性。
REMOTEPWD	包含用于通过服务器到服务器远程过程调用进行访问的远程服务器口令。 缺省值为无。 此属性是静态属性。
REPEAT_READ	确定驱动程序是否保留列和输出参数的副本，以便可以随机读取或重复读取列。 缺省值为 true。 此属性是静态属性。

属性	说明
REQUEST_HA_ SESSION	指示连接客户端是否希望开始高可用性 (HA) 故障切换会话。 建立连接后将不能重置此属性。 如果需要在请求故障切换会话时获得更大的灵活性，可以对客户端应用程序进行编码，使其在运行时设置 REQUEST_HA_SESSION。 此属性设置为 true 将导致 SAP jConnect 尝试进行故障切换登录。 如果没有设置此连接属性，即使已对服务器进行故障切换配置，也不会启动故障切换会话。 缺省值为 false。 此属性是静态属性。
REQUEST_ KERBEROS_SESSION	确定 SAP jConnect 是否使用 Kerberos 进行验证。 如果将此属性设置为 true，还必须为 SERVICE_PRINCIPAL_NAME 属性输入值。 可能还要为 GSSMANAGER_CLASS 属性提供值。 缺省值为 false。 此属性是静态属性。
RETRY_WITH_NO_ ENCRYPTION	允许服务器使用明文口令重新尝试登录。 当 ENCRYPT_PASSWORD 和 RETRY_WITH_NO_ENCRYPTION 属性都设置为 true 时，SAP jConnect 会首先使用加密口令登录。 如果登录失败，SAP jConnect 将使用明文口令登录。 缺省值为 false。 此属性是静态属性。
RMNAME	在使用分布式事务 (XA) 时设置资源管理器名称。 此属性将覆盖可能在 LDAP 服务器条目中设置的资源管理器名称。 缺省值为空值。 此属性是静态属性。
SECONDARY_ SERVER_HOSTPORT	在客户端使用 HA 故障切换会话时设置辅助服务器的主机名和端口。 此属性的值应采用的格式为 hostName:portNumber。 除非 REQUEST_HA_SESSION 也设置为 true，否则将忽略此属性。 缺省值为空值。 此属性是静态属性。

属性	说明
SELECT_OPENS_CURSOR	确定当 query 包含 **FOR UPDATE** 子句时是否自动调用 `Statement.executeQuery` 生成游标。 如果之前已在同一语句上调用 `Statement.setFetchSize` 或 `Statement.setCursorName`，那么将 `SELECT_OPENS_CURSOR` 设置为 true 将不起作用。 将 `SELECT_OPENS_CURSOR` 设置为 true 时，系统性能可能会下降。 缺省值为 false。 此属性是静态属性。
SEND_BATCHPARAMS_IMMEDIATE	指定 SAP jConnect 是在调用 `PreparedStatement.addBatch()` 后立即为当前行发送参数，还是仅在调用 `PreparedStatement.executeBatch()` 后再发送。 • true – SAP jConnect 在调用 `PreparedStatement.addBatch()` 后立即发送当前行的参数。 这最大限度地减少了客户端内存的使用，并让服务器有更多时间来处理批参数。 • false – SAP jConnect 仅在调用 `PreparedStatement.executeBatch()` 后才发送批处理参数。 缺省值为 false。 此属性是动态属性。
SERIALIZE_ REQUESTS	确定 SAP jConnect 在发送其它请求之前是否等待服务器的响应。 缺省值为 false。 此属性是静态属性。
SERVER_INITIATED_TRANSACTIONS	允许服务器控制事务。 缺省情况下，该属性设置为 true，并且 SAP jConnect 通过使用 SAP Transact-SQL 命令 **set chained on** 来允许服务器启动并控制事务。 如果设置为 false，则 SAP jConnect 通过使用 Transact-SQL 命令 **begin tran** 启动并控制事务。 SAP 建议允许服务器控制事务。 缺省值为 true。 此属性是静态属性。

属性	说明
SERVICENAME	指示 DirectConnect 网关服务的后端数据库服务器的名称。 还用于指示在连接到 SAP SQL Anywhere 之后应使用的数据库。 缺省值为无。 此属性是静态属性。
SERVERTYPE	在连接到 SAP® OpenSwitch 时将此属性设置为 OSW。这将允许 SAP jConnect 向 SAP OpenSwitch 发送某些指令，以允许 SAP OpenSwitch 在将连接重定向到另一个服务器实例时记住初始的连接设置，例如隔离级别、textsize、带引号的标识符和 autocommit。 缺省值为无。 此属性是静态属性。
SERVICE_ PRINCIPAL_ NAME	在建立到 SAP Adaptive Server 的 Kerberos 连接时使用。 此属性的值应与密钥分发中心 (KDC) 中的服务器条目和数据库运行时所使用的服务器名称对应。 如果 `REQUEST_KERBEROS_SESSION` 属性设置为 false，将会忽略 `SERVICE_PRINCIPAL_NAME` 属性的值。 缺省值为空值。 此属性是静态属性。
SESSION_ID	TDS 会话 ID。 如果设置此属性，SAP jConnect 将认为应用程序正在试图恢复由 TDS 隧道网关保持打开的现有 TDS 会话的通信。 SAP jConnect 将跳过登录协商并将所有来自应用程序的请求转发到指定的会话 ID。 缺省值为空值。 此属性是静态属性。
SESSION_TIMEOUT	指定 HTTP 隧道会话（使用 SAP jConnect TDS 隧道服务器小程序创建）在空闲时保持活动的时间量（以秒为单位）。 在达到指定时间后，连接会自动关闭。 缺省值为空值。 此属性是静态属性。

属性	说明
SETMAXROWS_ AFFECTS_SELECT_ ONLY	指定 `setMaxRows` 是否仅限制 **select** 语句返回的行，以与 JDBC 规范一致。 有效值为： • true – `Statement.setMaxRows(int max)` 仅限制作为 **select** 语句的结果返回的行数。 • false – `Statement.setMaxRows(int max)` 对作为 **select**、**insert**、**update** 和 **delete** 语句的结果返回的行数加以限制。 当连接到 SAP Adaptive Server 15.5 或更低版本时，`SETMAXROWS_AFFECTS_SELECT_ONLY` 会被忽略。 缺省值为 true。 此属性是静态属性。
SQLINITSTRING	定义要在连接打开时传递给数据库服务器的一组命令。 这些命令必须是可以用 `Statement.executeUpdate` 方法执行的 SQL 命令。 缺省值为空值。 此属性是静态属性。
STREAM_CACHE_ SIZE	指定用于高速缓存语句响应流的最大大小。 缺省值为空值（无限制的高速缓存大小）。 此属性是动态属性。
STRIP_BLANKS	强制服务器在将字符串值存储到表中之前，先删除其前导空白和尾随空白。 有效值为： • false – 客户端发送的字符串值“按原样”存储。 • true – 在将字符串值存储到表中之前，先删除其前导空白和尾随空白。 缺省值为 false。 此属性是静态属性。

属性	说明
SUPPRESS_CONTROL_TOKEN	取消发送控制令牌。 有效值为： • false – 发送控制令牌。 • true – 取消发送控制令牌。 缺省值为 false。 此属性是静态属性。
SUPPRESS_PARAM_FORMAT	在执行动态 SQL 预准备语句时，SAP jConnect 客户端可使用 SUPPRESS_PARAM_FORMAT 连接字符串属性来抑制参数格式元数据。 在可能的情况下，客户端会减少发送的参数元数据以提高性能。 有效值为： • false – 在 select、insert 和 update 操作中不抑制参数格式元数据。 • true – 缺省值；在可能情况下均抑制参数格式元数据。 缺省值为 true。 此属性是静态属性。
SUPPRESS_ROW_FORMAT	在 SAP jConnect 中，客户端可使用 SUPPRESS_ROW_FORMAT 连接字符串属性来强制 SAP Adaptive Server 仅在动态 SQL 预准备语句的行格式更改时发送 TDS_ROWFMT 或 TDS_ROWFMT2 数据。 这样，SAP Adaptive Server 可以尽量向客户端发送较少的数据，从而提高性能。 有效值为： • false – 即便行格式未发生更改，也发送 TDS_ROWFMT 或 TDS_ROWFMT2 数据。 • true – 缺省值；强制服务器仅在行格式更改时才发送 TDS_ROWFMT 或 TDS_ROWFMT2 数据。 缺省值为 true。 此属性是静态属性。

属性	说明
SUPPRESS_ROW_ FORMAT2	指定 SAP Adaptive Server 尽可能使用 TDS_ROWFMT 字节序列而不是 TDS_ROWFMT2 字节序列发送数据。 有效值为: • false – 缺省值；不禁止 TDS_ROWFMT2。 • true – 强制服务器尽可能以 TDS_ROWFMT 格式发送数据。 在连接到 SAP Adaptive Server 15.7 ESD #1 及更高版本时，应改用 SUPPRESS_ROW_FORMAT 连接属性。 缺省值为 false。 此属性是静态属性。
SYBSOCKET_ FACTO-RY	使 SAP jConnect 能够使用自定义套接字实现。 将 SYBSOCKET_FACTORY 设置为下列两项之一: • 实现 com.sybase.jdbcx.SybSocketFactory 的类的名称；或 • DEFAULT，这样将实例化新的 java.net.Socket( )。 使用此属性可与数据库建立 SSL 连接。 缺省值为空值。 此属性是静态属性。
TEXTSIZE	允许设置文本大小。缺省情况下，SAP Adaptive Server 和 SAP SQL Anywhere 允许从图像或文本列中读取 32,627 字节。 如果已经安装 SAP jConnect Meta Data 表，SAP jConnect 会将该值更改为 2GB。但是，如果在连接到 SAP OpenSwitch 时设置该值，则允许连接在被 SAP OpenSwitch 重定向到另一个服务器实例时记住此设置。 缺省值为 2GB。 此属性是静态属性。

属性	说明
USE_METADATA	在建立连接时创建并初始化 `DatabaseMetaData` 对象。 连接到指定的数据库必须要使用 `DatabaseMetaData` 对象。 SAP jConnect 将 `DatabaseMetaData` 用于某些功能，如分布式事务管理支持 (JTA/JTS) 和动态类装载 (DCL)。 如果收到错误 010SJ（表明应用程序需要元数据），请安装 SAP jConnect 附带的用于返回元数据的存储过程。 请参见《jConnect for JDBC 安装指南》中的“安装存储过程”。 缺省值为 true。 此属性是静态属性。
USER	指定登录 ID。 如果使用 `getConnection(String, String, String)` 方法，将自动设置；如果使用 `getConnection(String, Props)`，将显式设置。 缺省值为无。 此属性是静态属性。
VERSIONSTRING	提供 JDBC 驱动程序的只读版本信息。 缺省值是 SAP jConnect 驱动程序版本。 此属性是静态属性。

另请参见

- DYNAMIC_PREPARE 连接属性 （第 132 页）
- 口令加密 （第 77 页）
- 安全性 （第 101 页）
- SAP jConnect 中的优化批处理 （第 133 页）
- 游标性能 （第 135 页）
- 故障切换支持 （第 44 页）
- Adaptive Server 的宽表支持 （第 48 页）
- CONNECTION_FAILOVER 属性 （第 35 页）
- 大对象定位符支持 （第 68 页）
- 使用 Connection.isClosed 和 IS_CLOSED_TEST （第 96 页）
- 取代缺省字符集映射 （第 43 页）
- JCONNECT_VERSION 连接属性 （第 4 页）
- 在游标关闭时释放锁 （第 55 页）
- TDS 隧道 （第 141 页）

- 使用 DynamicClassLoader （第 83 页）
- 使用 SAP jConnect 传递 Unicode 数据 （第 37 页）
- 选择字符集转换程序 （第 39 页）
- 预装载 .jar 文件 （第 85 页）

连接到 Adaptive Server

在 Java 应用程序中，定义一个使用 SAP jConnect 驱动程序连接到 SAP Adaptive Server 的 URL。

此 URL 的基本格式如下：

```
jdbc:sybase:Tds:host:port
```

其中：

- *jdbc:sybase* 标识驱动程序。
- *Tds* 是 SAP Adaptive Server 的 SAP 通信协议。
- *host:port* 是 SAP Adaptive Server 主机名和监听端口。 有关数据库或 SAP Open Server 应用程序使用的条目，请参见 `$SYBASE/interfaces` (UNIX) 或 `%SYBASE%\ini\sql.ini` (Windows)。 可通过查询条目获取 *host:port*。

可使用下面的格式连接到特定数据库：

```
jdbc:sybase:Tds:host:port/database
```

注意： 使用 SAP SQL Anywhere 或 DirectConnect 连接到特定数据库。 使用 `SERVICENAME` 连接属性而不是“/database”来指定数据库名称。

以下代码将创建到主机“myserver”上的监听端口 3697 的 SAP Adaptive Server 的连接：

```
SysProps.put("user","userid");
SysProps.put("password","user_password");
String url = "jdbc:sybase:Tds:myserver:3697";
Connection con =
   DriverManager.getConnection(url,SysProps);
```

URL 连接属性参数

在定义 URL 时指定 SAP jConnect 驱动程序连接属性的值。

注意： 在 URL 中设置的驱动程序连接属性不会覆盖在应用程序中使用 `DriverManager.getConnection` 方法设置的任何相应的连接属性。

在 URL 中设置连接属性，并将属性名及其值附加到 URL 定义中。 使用以下语法：

```
jdbc:sybase:Tds:host:port/database?
     property_name=value
```

设置多个连接属性，并附加每个额外的连接属性和值（前面加上“&”）。 例如：

```
jdbc:sybase:Tds:myserver:1234/mydatabase?
   LITERAL_PARAMS=true&PACKETSIZE=512&HOSTNAME=myhost
```

如果其中某个连接属性的值包含“&”，请将“&”置于该连接属性值之前，并加上一个反斜杠 (\)。 例如，如果主机名是“a&bhost”，则使用下面的语法：

```
jdbc:sybase:Tds:myserver:1234/mydatabase?
   LITERAL_PARAMS=true&PACKETSIZE=512&HOSTNAME=
   a\&bhost
```

即使连接属性值是字符串，也不要对其使用引号。 例如，使用：

```
HOSTNAME=myhost
```

而不是：

```
HOSTNAME="myhost"
```

使用 sql.ini 和 interfaces 文件目录服务

使用 sql.ini 文件（对于 Windows）和 interfaces 文件（对于 UNIX）为 SAP jConnect for JDBC 提供服务器信息。

通过使用 sql.ini 或 interfaces 文件，企业可以集中管理企业网络中的可用服务的相关信息（包括 Adaptive Server 信息）。

使用连接字符串标识 sql.ini 或 interfaces 文件。 在 jConnect for JDBC 上，只能连接到单个目录服务 URL (DSURL)。

用于 SAP jConnect 的单个 DSURL 的连接字符串

当连接到 DSURL 时，必须指定 sql.ini 或 interfaces 文件的路径和服务器名称。

如果没有设置路径，SAP jConnect 将返回一个错误。

以下语法指定 sql.ini 文件的路径：

```
String url = "jdbc:sybase:jndi:file://D:/syb1252/ini/mysql.ini?
myaseISO1”
```

其中：

- 服务器名称为 myaseISO1
- sql.ini 文件路径为 file://D:/syb1252/ini/sql.ini?

以下语法指定 interfaces 文件的路径：

```
String url = "jdbc:sybase:jndi:file:///work/sybase/interfaces?myase"
```

其中：

- 服务器名称为 myase
- interfaces 文件路径为 file:///work/sybase/interfaces

适用于 SSL 的 sql.Ini 和 Interfaces 文件的格式

检查适用于 SSL 的 sql.ini 和 interfaces 文件的格式。

用于 SSL 的 sql.ini 文件的格式为：

```
[SYBSRV2]
master=nlwnsck,mango1,4100,ssl
query=nlwnsck,mango1,4100,ssl
query=nlwnsck,mango1,5000,ssl
```

interfaces 文件的格式为：

```
sybsrv2
master tcp ether mango1 5000 ssl
query tcp ether mango1 4100 ssl
query tcp ether mango1 5000 ssl
```

注意： SAP jConnect 支持在 `sql.ini` 或 `interfaces` 文件中的同一服务器名称下存在多个查询条目。 SAP jConnect 尝试按照 `sql.ini` 或 `interfaces` 文件中的顺序，通过查询条目连接到 `host` 或 `port` 值。 如果 SAP jConnect 在查询中找到 SSL，它将通过指定应用程序特定的套接字工厂来要求对应用程序进行编码以处理 SSL 连接，否则，连接可能会失败。

使用 JNDI 连接到服务器

在 SAP jConnect 中，您可以使用 Java 命名和目录接口 (JNDI) 来提供连接信息。

SAP jConnect 提供：

- 一个集中位置，可以从中指定主机名和端口以连接到服务器。 不需要在应用程序中对特定主机和端口号进行硬编码。
- 一个集中位置，可以从中指定供所有应用程序使用的连接属性和缺省数据库。
- 用于处理不成功的连接尝试的 SAP jConnect `CONNECTION_FAILOVER` 属性。 `CONNECTION_FAILOVER` 为 true 时，SAP jConnect 将尝试连接到 JNDI 名称空间中的主机/端口服务器地址序列，直到连接成功。

若要配合使用 SAP jConnect 和 JNDI，需要确保在 JNDI 访问的任何目录服务中都能获得特定的信息，并且在 `javax.naming.Context` 类中设置了所需的信息。

另请参见

- 使用 JNDI 的连接 URL （第 33 页）
- 所需的目录服务信息 （第 34 页）
- CONNECTION_FAILOVER 属性 （第 35 页）
- 提供 JNDI 上下文信息 （第 36 页）

使用 JNDI 的连接 URL

要指定 SAP jConnect 使用 JNDI 获取连接信息，请将“jndi”作为 URL 协议放在“sybase”后面。

例如：

```
jdbc:sybase:jndi:protocol-information-for-use-with-JNDI
```

URL 中“jndi”后的所有内容均通过 JNDI 进行处理。 例如，要将 JNDI 用于轻量目录访问协议 (LDAP)，可输入以下内容：

```
jdbc:sybase:jndi:ldap://LDAP_hostname:port_number/servername=
  Sybase11,o=MyCompany,c=US
```

此 URL 告知 JNDI 从 LDAP 服务器获取信息，提供要使用的 LDAP 服务器的主机名和端口号，并且以特定于 LDAP 形式的提供数据库服务器的名称。

所需的目录服务信息

在结合使用 JNDI 和 SAP jConnect 时，请检查所需的目录服务信息。

JNDI 必须为目标数据库服务器返回以下信息：

- 要连接到的主机名和端口号
- 要使用的数据库的名称
- 不允许单个应用程序自己设置的任何连接属性

在用于提供连接信息的任意目录服务中按照固定格式存储这些信息。 要求的格式包括一个数值对象标识符 (OID)，它标识所提供的信息的类型（如目标数据库），随后是格式化信息。

注意： 可以使用别名代替 OID 来引用属性。

表 4. JNDI 的目录服务信息

属性说明	别名	OID (object_id)
LDAP 目录服务中的接口条目替换	sybaseServer	1.3.6.1.4.1.897.4.1.1
sybaseServer LDAP 属性的集合点	sybaseServer	1.3.6.1.4.1.897.4.2
版本	sybaseVersion	1.3.6.1.4.1.897.4.2.1
服务器名	sybaseServer	1.3.6.1.4.1.897.4.2.2
服务	sybaseService	1.3.6.1.4.1.897.4.2.3
状态	sybaseStatus	1.3.6.1.4.1.897.4.2.4
（必需）地址	sybaseAddress	1.3.6.1.4.1.897.4.2.5
安全机制	sybaseSecurity	1.3.6.1.4.1.897.4.2.6
重试计数	sybaseRetryCount	1.3.6.1.4.1.897.4.2.7
循环延迟	sybaseRetryDelay	1.3.6.1.4.1.897.4.2.8
（必需）jConnect 连接协议	sybaseJconnectProtocol	1.3.6.1.4.1.897.4.2.9
（必需）jConnect 连接属性	sybaseJconnectProperty	1.3.6.1.4.1.897.4.2.10
（必需）数据库名	sybaseDatabasename	1.3.6.1.4.1.897.4.2.11
高可用性故障切换服务器名	sybaseHAservername	1.3.6.1.4.1.897.4.2.15
ResourceManager 名称	sybaseResourceManagerName	1.3.6.1.4.1.897.4.2.16

属性说明	别名	OID (object_id)
ResourceManager 类型	sybaseResourceManagerType	1.3.6.1.4.1.897.4.2.17
JDBCDataSource 接口	sybaseJdbcDataSource- Interface	1.3.6.1.4.1.897.4.2.18
ServerType	sybaseServerType	1.3.6.1.4.1.897.4.2.19

以下示例演示了为 LDAP 目录服务下的数据库服务器“SYBASE11”输入的连接信息。可以使用 OID 也可以使用别名。

- **示例 1** – 使用属性 OID:

```
dn: servername=SYBASE11,o=MyCompany,c=US
 servername:SYBASE11
1.3.6.1.4.1.897.4.2.5:TCP#1#giotto 1266
1.3.6.1.4.1.897.4.2.5:TCP#1#giotto 1337
1.3.6.1.4.1.897.4.2.5:TCP#1#standby1 4444
1.3.6.1.4.1.897.4.2.10:REPEAT_READ=false&
    PACKETSIZE=1024
 1.3.6.1.4.1.897.4.2.10:CONNECTION_FAILOVER=true
 1.3.6.1.4.1.897.4.2.11:pubs2
 1.3.6.1.4.1.897.4.2.9:Tds
```

- **示例 2** – 使用属性别名，别名不区分大小写:

```
dn: servername=SYBASE11,o=MyCompany,c=US
 servername:SYBASE11
sybaseAddress:TCP#1#giotto 1266
sybaseAddress:TCP#1#giotto 1337
sybaseAddress:TCP#1#standby1 4444
sybaseJconnectProperty:REPEAT_READ=false&
   PACKETSIZE=1024
sybaseJconnectProperty:CONNECTION_FAILOVER=true
sybaseDatabasename:pubs2
sybaseJconnectProtocol:Tds
```

在这些示例中，可通过主机“giotto”上的端口 1266 或 1337 访问 SYBASE11，也可通过主机“standby1”上的端口 4444 对其进行访问。 `REPEAT_READ` 和 `PACKETSIZE` 这两个连接属性在一个条目中设置。 `CONNECTION_FAILOVER` 连接属性是作为单独条目设置的。 连接到 SYBASE11 的应用程序开始时是与 `pubs2` 数据库连接。 不需要指定连接协议，但如果指定了，则必须以“`Tds`”而不是“`TDS`”的形式输入该属性。

CONNECTION_FAILOVER 属性

`CONNECTION_FAILOVER` 是布尔值连接属性，可以在 SAP jConnect 使用 JNDI 获取连接信息时使用该属性。

如果 `CONNECTION_FAILOVER` 为 true（缺省值），SAP jConnect 将多次尝试连接到服务器。 如果一次与服务器关联的主机和端口号的连接尝试失败，SAP jConnect 将使用 JNDI 获取与该服务器关联的下一个主机和端口号，并通过它们尝试连接。 连接尝试将按顺序使用与服务器关联的所有主机和端口。

例如，如果数据库服务器与下列主机和端口号相关联（如前面 LDAP 示例中所述）：

```
1.3.6.1.4.1.897.4.2.5:TCP#1#giotto 1266
1.3.6.1.4.1.897.4.2.5:TCP#1#giotto 1337
1.3.6.1.4.1.897.4.2.5:TCP#1#standby 4444
```

要连接到服务器，SAP jConnect 将尝试通过端口 1266 连接到主机“giotto”。 如果失败，SAP jConnect 将尝试“giotto”上的端口 1337。 如果再次失败，SAP jConnect 将尝试通过端口 4444 连接到主机“standby1”。

如果 CONNECTION_FAILOVER 为 false，SAP jConnect 将尝试连接到初始主机和端口号。 如果尝试失败，SAP jConnect 将抛出一个 SQL 异常并不再重试。

提供 JNDI 上下文信息

熟悉 JNDI 规范，以配合使用 SAP jConnect 和 JNDI。

请参见 JNDI specification from Oracle Technology Network。

特别是当 JNDI 和 SAP jConnect 一起使用时，需要确保在 javax.naming.directory.DirContext 中设置所需的初始化属性。 在系统级别或运行时设置这些属性。

这些属性包括：

- Context.INITIAL_CONTEXT_FACTORY – JNDI 使用的初始上下文工厂的全限定类名。 这确定了在 Context.PROVIDER_URL 属性中指定的 URL 使用的 JNDI 驱动程序。
- Context.PROVIDER_URL – 获取驱动程序（如 LDAP 驱动程序）要访问的目录服务的 URL。 此 URL 应该是一个字符串，如“ldap://ldaphost:427”。

此示例演示如何在运行时设置上下文属性以及如何使用 JNDI 和 LDAP 获取连接。 INITIAL_CONTEXT_FACTORY 上下文属性设置为调用 LDAP 服务提供程序的 Oracle 实现。 Context.PROVIDER_URL 属性被设置为位于主机“ldap_server1”上 389 端口的 LDAP 目录服务的 URL。

```
Properties props = new Properties();

/* We want to use LDAP, so INITIAL_CONTEXT_FACTORY is set to the
* class name of an LDAP context factory. In this case, the
* context factory is provided by Sun's implementation of a
* driver for LDAP directory service.
*/
props.put(Context.INITIAL_CONTEXT_FACTORY,
  "com.sun.jndi.ldap.LdapCtxFactory");

/* Now, we set PROVIDER_URL to the URL of the LDAP server that
* is to provide directory information for the connection.
*/
props.put(Context.PROVIDER_URL, "ldap://ldap_server1:389");

/* Set up additional context properties, as needed. */
props.put("user", "xyz");
props.put("password", "123");

/* get the connection */
```

```
Connection con = DriverManager.getConnection
   ("jdbc:sybase:jndi:ldap://ldap_server1:389" +
   "/servername=Sybase11,o=MyCompany,c=US",props);
```

传递给 `getConnection` 的连接字符串包含开发人员必须提供的特定于 LDAP 的信息。

在运行时设置 JNDI 属性后（如上例所示），SAP jConnect 将它们传递给要用于初始化服务器的 JNDI，如以下 SAP jConnect 代码所示：

```
javax.naming.directory.DirContext ctx =
   new javax.naming.directory.InitialDirContext(props);
```

SAP jConnect 然后会通过调用 `DirContext.getAtributes` 从 JNDI 获取所需的连接信息，如此示例所示，其中 *ctx* 是一个 **DirContext** 对象：

```
javax.naming.directory.Attributes attrs =
   ctx.getAttributes("ldap://ldap_server1:389/servername=" +
     "Sybase11", SYBASE_SERVER_ATTRIBUTES);
```

SYBASE_SERVER_ATTRIBUTES 是在 SAP jConnect 中定义的字符串数组。 数组值是“所需的目录服务信息”（第 34 页）中列出的所需目录信息的 OID。

国际化和本地化

检查与 SAP jConnect 有关的国际化和本地化问题。

使用 SAP jConnect 传递 Unicode 数据

在 SAP Adaptive Server 12.5 和更高版本中，数据库客户端可以使用 `unichar` 和 `univarchar` 数据类型

这两种数据类型可以实现 Unicode 数据的有效存储和检索，同时允许用户指定数据库表列存储 Unicode 数据，而不用考虑服务器的缺省字符集。

以下为从 Unicode 标准（版本 2.0）中引用的一段内容：

```
The Unicode Standard is a fixed-width, uniform encoding scheme for
encoding characters and text. The repertoire of this international
character code for information processing includes characters for the
major scripts of the world, as well as technical symbols in common.
The Unicode character encoding treats alphabetic characters,
ideographic characters, and symbols identically, which means they can
be used in any mixture and with equal facility. The Unicode Standard
is modeled on the ASCII character set, but uses a 16-bit encoding to
support full multilingual text.
```

注意： 在 SAP Adaptive Server 12.5 到 12.5.0.3 版本中，服务器必须具有缺省字符集 utf-8 才能使用 Unicode 数据类型。 但在 SAP Adaptive Server 12.5.1 和更高版本中，数据库用户无需考虑服务器的缺省字符集即可使用 `unichar` 和 `univarchar` 数据类型。

在可以使用 char 和 varchar 字符数据类型的任何位置均可以使用 unichar 和 univarchar 数据类型，而且不必更改语法。

- unichar – 使用 *n* 指定 Unicode 字符数（分配的存储空间量为每个字符 2 个字节）。
- univarchar – 使用 *n* 指定可变长度数据类型的字符的最大长度。

服务器接受 unichar 和 univarchar 数据时，SAP jConnect 将执行以下操作：

- 对于客户端要发送到服务器的所有字符数据 - 例如，使用 PreparedStatement.setString (int column, String value) - SAP jConnect 确定字符串是否能转换为服务器的缺省字符集。
- 如果 SAP jConnect 确定这些字符不能转换为服务器的字符集（例如，有些字符无法用服务器的字符集表示），它会将数据以 unichar/univarchar 数据编码发送给服务器。

例如，如果客户端尝试向以 iso_1 作为缺省字符集的 SAP Adaptive Server 12.5.1 发送 Unicode 日语字符，SAP jConnect 将检测到该日语字符不能转换为 iso_1 字符。 SAP jConnect 随后以 Unicode 数据发送字符串。

客户端向服务器发送 unichar/univarchar 数据会降低计算机的性能，这是因为 SAP jConnect 必须对不能直接映射到服务器缺省字符集的所有字符串和字符执行两次字符到字节的转换。

如果您使用的是 6.05 以前的 SAP jConnect 版本，并想要使用 unichar 和 univarchar 数据类型，您必须执行以下任务：

1. 设置 JCONNECT_VERSION = 6 或更高版本。
2. 您需要将 DISABLE_UNICHAR_SENDING 连接属性设置为 false。

   有关 unichar 和 univarchar 数据类型支持的详细信息，请参见《SAP Adaptive Server Enterprise 手册》(SAP Adaptive Server Enterprise Manuals)。

另请参见

- JCONNECT_VERSION 连接属性 （第 4 页）
- 设置连接属性 （第 8 页）

SAP jConnect 字符集转换程序

共有两个字符集转换类。 SAP jConnect 使用的转换类基于 JCONNECT_VERSION、CHARSET 和 CHARSET_CONVERTER_CLASS 连接属性。

- TruncationConverter 类只能用于使用 ASCII 字符的单字节字符集，如 iso_1 和 cp850。 它不能用于多字节字符集或使用非 ASCII 字符的单字节字符集。 当 JCONNECT_VERSION 设置为 2 时，TruncationConverter 类是缺省转换程序。

SAP jConnect 16.0 使用 TruncationConverter 类处理字符集的方式与 SAP jConnect 2.2 版相同。 当 JCONNECT_VERSION 为 2 时，TruncationConverter 类是缺省转换程序。

- PureConverter 类是纯 Java 多字节字符集转换程序。 如果 JCONNECT_VERSION 为 4 或更高，SAP jConnect 将使用此类。 当 JCONNECT_VERSION 为 2 时，如果 SAP jConnect 检测到 CHARSET 连接属性中所指定的字符集与 TruncationConverter 类不兼容，也将使用此转换程序。

  虽然 PureConverter 类能实现多字节字符集转换，但也可能降低 SAP jConnect 驱动程序的性能。

另请参见

- 提高字符集转换性能 （第 40 页）

选择字符集转换程序

SAP jConnect 使用 JCONNECT_VERSION 来确定要使用的缺省字符集转换程序类。

JCONNECT_VERSION = 2.0 or 3.0 时，缺省值为 TruncationConverter。
JCONNECT_VERSION = 4.0 或更高版本时，缺省值为 PureConverter。

也可以通过设置 CHARSET_CONVERTER_CLASS 连接属性指定希望 SAP jConnect 使用的字符集转换程序。 如果希望使用 jConnect 版本的缺省字符集转换程序之外的字符集转换程序，此方法将很有用。

例如，如果您设置 JCONNECT_VERSION = 4.0 或更高版本，但要使用 TruncationConverter 类而不使用多字节的 PureConverter 类，则可以设置 CHARSET_CONVERTER_CLASS：

```
...
 props.put("CHARSET_CONVERTER_CLASS",
   "com.sybase.jdbc4.charset.TruncationConverter")
```

设置 CHARSET 连接属性

可以通过设置 CHARSET 驱动程序属性指定要在应用程序中使用的字符集。

如果没有设置 CHARSET 属性：

- JCONNECT_VERSION = 2.0 时，SAP jConnect 将使用 iso_1 作为缺省字符集。
- JCONNECT_VERSION = 3.0 到 6.05 时，SAP jConnect 将使用数据库的缺省字符集并会在客户端自动调整以执行任何必要的转换。
- 对于从 6.05 开始的 SAP jConnect 版本，如果 SAP jConnect 无法将用户数据成功转换为协商的字符集，则在服务器支持 Unicode 字符时，它会向服务器发送未经转换的 Unicode 字符，否则会抛出异常。

也可以使用 **IsqlApp** 应用程序的 –J *charset* 命令行选项指定字符集。

若要确定 SAP Adaptive Server 上安装了哪些字符集，请在服务器上发出以下 SQL 查询：

```
select name from syscharsets
 go
```

对于 PureConverter 类，如果客户端 Java 虚拟机 (JVM) 不支持指定的 CHARSET，那么连接将失败并引发 SQLException，指出必须将 CHARSET 设置为 Adaptive Server 和客户端都支持的字符集。

如果使用 TruncationConverter 类，则无论指定的 CHARSET 是否是 7 位 ASCII 码，都将进行字符截断。 因此，如果您的应用程序必须处理非 ASCII 数据（例如任何亚洲语言），则不使用 TruncationConverter，因为这会导致数据损坏。

提高字符集转换性能

如果使用多字节字符集并需要提高驱动程序性能，可以使用 SAP jConnect 示例提供的 SunIoConverter 类。

另外，如果您的应用程序仅处理 7 位 ASCII 数据，则可以使用 TruncationConverter 来提高性能。

另请参见

- SunIoConverter 字符集转换 （第 128 页）

支持的字符集

SAP jConnect 支持的字符集，以及每个支持的字符集所对应的 JDK 字节转换程序。

虽然 SAP jConnect 支持 UCS-2, 但目前 SAP 数据库或 SAP Open Servers 都不支持 UCS-2。

SAP Adaptive Server 12.5 和更高版本支持一个称为 UTF-16 编码的 Unicode 版本。

表 5. 支持的 SAP jConnect 字符集

SybCharset 名称	JDK 字节转换程序
ascii_7	ASCII
big5	Big5
big5hk（针对 JDK 1.3 及更高版本）	Big5_HKSCS
cp037	Cp037
cp437	Cp437
cp500	Cp500
cp850	Cp850
cp852	Cp852
cp855	Cp855

SybCharset 名称	JDK 字节转换程序
cp857	Cp857
cp860	Cp860
cp863	Cp863
cp864	Cp864
cp866	Cp866
cp869	Cp869
cp874	Cp874
cp932	MS932
cp936	GBK
cp949	Cp949
cp950	Cp950
cp1250	Cp1250
cp1251	Cp1251
cp1252	Cp1252
cp1253	Cp1253
cp1254	Cp1254
cp1255	Cp1255
cp1256	Cp1256
cp1257	Cp1257
cp1258	Cp1258
deckanji	EUC_JP
eucgb	EUC_CN
eucjis	EUC_JP
eucksc	EUC_KR
gb18030	GB18030
ibm420	Cp420
ibm918	Cp918
iso_1	ISO8859_1

SybCharset 名称	JDK 字节转换程序
iso88592	ISO8859-2
is088595	ISO8859_5
iso88596	ISO8859_6
iso88597	ISO8859_7
iso88598	ISO8859_8
iso88599	ISO8859_9
iso15	ISO8859_15_FDIS
koi8	KOI8_R
mac	MacRoman
mac_cyr	MacCyrillic
mac_ee	MacCentralEurope
macgreek	MacGreek
macturk	MacTurkish
sjis	MS932
tis620	MS874
ucs2	Unicode
utf8	UTF8

不支持的字符集

SAP jConnect 不支持某些字符集，因为不存在与这些字符集相类似的 JDK 字节转换程序。

- cp1047
- euccns
- greek8
- roman8
- roman9
- turkish8

可将这些字符集用于 `TruncationConverter` 类，前提是应用程序仅使用这些字符的 7 位 ASCII 子集。

取代缺省字符集映射

使用 JAVA_CHARSET_MAPPING 连接属性可取代 SAP Adaptive Server 的缺省字符集映射。

- **示例** – 将服务器字符集 cp949 映射到 ms949：

```
props.put("CHARSET", "cp949"); /* Server character set */
props.put("JAVA_CHARSET_MAPPING", "ms949"); /* Java character set
mapping */
```

  大多数 SAP Adaptive Server 字符集与其所映射到的 Java 字符集同名。有关映射到 Java 字符集而使用不同名称的字符集，请参见支持的字符集 （第 40 页）。

欧洲货币符号支持

SAP jConnect 支持使用欧洲货币符号（或“euro”），并支持欧洲货币符号与 UCS-2 Unicode 间的相互转换。

以下 SAP jConnect 字符集中包括 euro：cp1250、cp1251、cp1252、cp1253、cp1254、cp1255、cp1256、cp1257、cp1258、cp874、iso885915 和 utf8。

要使用 euro 符号，请执行以下操作：

- 使用纯 Java 多字节字符集转换程序 PureConvertor 或 CheckPureConverter 类。
- 检验是否在服务器上安装了新的字符集。
- 在客户端上选择合适的字符集。

另请参见

- SAP jConnect 字符集转换程序 （第 38 页）
- 设置 CHARSET 连接属性 （第 39 页）

数据库问题

检查与 SAP jConnect 有关的数据库问题。

另请参见

- 批处理更新支持 （第 59 页）
- 数据类型 （第 61 页）
- 故障切换支持 （第 44 页）
- 服务器到服务器的远程过程调用 （第 47 页）
- Adaptive Server 的宽表支持 （第 48 页）
- 对结果集使用游标 （第 49 页）
- 包含 COMPUTE 子句的 Transact-SQL 查询 （第 59 页）
- DOL 锁定表中的可变长度行 （第 67 页）

- 大对象 (LOB) 支持 （第 68 页）
- 大对象定位符支持 （第 68 页）
- 访问数据库元数据 （第 48 页）
- 通过存储过程的结果集更新数据库 （第 60 页）

故障切换支持

SAP jConnect 支持 SAP Adaptive Server 故障切换。

SAP Adpative Server 故障切换允许配置两台 SAP Adaptive Server 作为协同服务器。

注意： 高可用性系统中的 SAP Adpative Server 故障切换与连接故障切换功能不同。如果希望同时使用这两个功能，SAP 强烈建议您仔细阅读本节内容。

如果主协同服务器发生故障，该服务器的设备、数据库和连接可以由辅助协同服务器接管。 您可以按非对称或对称方式来配置高可用性系统。

- 非对称配置包括两台 SAP Adaptive Server，它们在物理上位于不同的计算机上，但彼此相连，以便当一台服务器出现故障时，可以由另一台服务器承担它的工作负荷。 辅助 SAP Adaptive Server 充当“热备份”，它只有在出现故障切换时才工作。
- 对称配置也包括两台在不同计算机上运行的 SAP Adaptive Server。 但如果发生故障切换，其中任何一台 SAP Adaptive Server 都可以充当另一台 SAP Adaptive Server 的主协同服务器或辅助协同服务器。 在此配置中，每一台 SAP Adaptive Server 都具有完整的功能，还具有各自的系统设备、系统数据库、用户数据库和用户登录。

在上述两种设置中，两台计算机都配置为双向访问，这样使两台计算机都可以看到并访问对方的磁盘。 可以在 SAP jConnect 中启用故障切换，然后将客户端应用程序连接到进行过故障切换配置的 SAP Adaptive Server。 如果主服务器故障切换到辅助服务器，客户端应用程序也会自动切换到辅助服务器并重新建立网络连接。

有关更多详细信息，请参见 Adaptive Server 文档中的《在高可用性系统中使用故障切换》。

在使用 SAP jConnect 作为故障切换策略的一部分时：

- 必须配置两台 SAP Adaptive Server 用于故障切换。
- 当客户端发生故障切换时，只保留在发生故障切换之前提交给数据库的更改。
- 将 `REQUEST_HA_SESSION` jConnect 连接属性设置为 true。
- 发生故障切换时 SAP jConnect 事件通知不起作用。
- 请关闭所有不再使用的语句。 SAP jConnect 会存储有关语句的信息以启用故障切换。 未关闭的语句将导致内存泄漏。

在 SAP jConnect 中实现故障切换

在 SAP jConnect 中实现故障切换支持。

1. 设置：

- 将 REQUEST_HA_SESSION 设置为 true。
- 将 SECONDARY_SERVER_HOSTPORT 设置为辅助服务器监听的主机名和端口号。

2. 使用 JNDI 连接到服务器。 在 JNDI 所需的目录服务信息文件中加入一个主服务器条目和一个辅助服务器条目。

   主服务器条目有一个引用辅助服务器条目的属性 (HA OID)。

   使用 LDAP 作为 JNDI 的服务提供程序时，此 HA 属性可以有以下三种可能的形式：

   - 相对区分名 (RDN) – 此形式假定与此属性值结合使用的搜索库（通常由 java.naming.provider.url 属性提供）足以标识辅助服务器。
     例如，假定主服务器位于 hostname:4200 上，而辅助服务器位于 hostname:4202 上：

     ```
     dn: servername=haprimary, o=Sybase, c=US
     1.3.6.1.4.1.897.4.2.5: TCP#1#hostname 4200
     1.3.6.1.4.1.897.4.2.15: servername=hasecondary
     objectclass: sybaseServer
     ```

     ```
     dn: servername=hasecondary, o=Sybase, c=US
     1.3.6.1.4.1.897.4.2.5: TCP#1#hostname 4202
     objectclass: sybaseServer
     ```

   - 区分名 (DN) – 此形式假定 HA 属性的值唯一标识辅助服务器，因此可能复制也可能不复制在搜索库中找到的值。
     例如：

     ```
     dn: servername=haprimary, o=Sybase, c=US
     1.3.6.1.4.1.897.4.2.5: TCP#1#hostname 4200
     1.3.6.1.4.1.897.4.2.15: servername=hasecondary,
        o=Sybase, c=US ou=Accounting
     objectclass: sybaseServer
     ```

     ```
     dn: servername=hasecondary, o=Sybase, c=US, ou=Accounting
     1.3.6.1.4.1.897.4.2.5: TCP#1#hostname 4202
     objectclass: sybaseServer
     ```

     请注意，hasecondary 位于树的其它分支上（请参见附加的 ou=Accounting 限定符）。

   - Full LDAP URL – 此形式对搜索库没有任何假定。 HA 属性应是用于标识辅助服务器的完全限定 LDAP URL（它甚至可以指向不同的 LDAP 服务器）。
     例如：

     ```
     dn: servername=hafailover, o=Sybase, c=US
     1.3.6.1.4.1.897.4.2.5: TCP#1#hostname 4200
     1.3.6.1.4.1.897.4.2.15: ldap://ldapserver: 386/
     servername=secondary,
        o=Sybase, c=US ou=Accounting
     objectclass: sybaseServer
     ```

```
dn: servername=secondary, o=Sybase, c=US, ou=Accounting
1.3.6.1.4.1.897.4.2.5: TCP#1#hostname 4202
objectclass: sybaseServer
```

使用 REQUEST_HA_SESSION 连接属性表明连接客户端希望与配置为用于故障切换的 Adaptive Server 开始一个故障切换会话。 此属性设置为 true 将导致 SAP jConnect 尝试进行故障切换登录。 如果没有设置此连接属性，即使正确配置了服务器，也不会启动故障切换会话。 REQUEST_HA_SESSION 的缺省值是 false。

像设置其它任何连接属性一样设置此连接属性。 建立连接后将不能重置此属性。

如果希望在请求故障切换会话时更加灵活，可以对客户端应用程序进行编码，使其在运行时设置 REQUEST_HA_SESSION。

此示例演示为 LDAP 目录服务下的数据库服务器 SYBASE11 输入的连接信息，其中“tahiti”是主服务器，“moorea”是辅助协同服务器：

```
dn: servername=SYBASE11,o=MyCompany,c=US
1.3.6.1.4.1.897.4.2.5:TCP#1#tahiti 3456
1.3.6.1.4.1.897.4.2.10:REPEAT_READ=false&PACKETSIZE=1024
1.3.6.1.4.1.897.4.2.10:CONNECTION_FAILOVER=false
1.3.6.1.4.1.897.4.2.11:pubs2
1.3.6.1.4.1.897.4.2.9:Tds
1.3.6.1.4.1.897.4.2.15:servername=SECONDARY
1.3.6.1.4.1.897.4.2.10:REQUEST_HA_SESSION=true
```

```
dn:servername=SECONDARY, o=MyCompany, c=US
1.3.6.1.4.1.897.4.2.5:TCP#1#moorea 6000
```

3. 使用 JNDI 和 LDAP 请求连接：

   a) 使用 LDAP 服务器的目录确定主服务器和辅助服务器的名称和位置：

   ```
   /* get the connection */
   Connection con = DriverManager.getConnection
      ("jdbc:sybase:jndi:ldap://ldap_server1:389" +
       "/servername=Sybase11,o=MyCompany,c=US",props);
   ```

   或

   b) 指定搜索库：

   ```
   props.put(Context.PROVIDER_URL,
      "ldap://ldap_server1:389/ o=MyCompany, c=US");
   ```

   ```
   Connection con=DriverManager.getConnection
      ("jdbc:sybase:jndi:servername=Sybase11", props);
   ```

   故障切换过程允许：

   - **登录到主服务器** – 如果 SAP Adaptive Server 没有配置为用于故障切换或者不能批准故障切换会话，客户端将无法登录。

   ```
   'The server denied your request to use the high-
   availability feature.

   Please reconfigure your database, or do not request a
   high-availability session.'
   ```

- **故障切换到辅助服务器** – 发生故障切换时将抛出 SQL 异常 JZ0F2：

  ```
  'SAP Adaptive Server Enterprise high-availability failover has
  occurred. The
  current transaction is aborted, but the connection is
  still usable. Retry your transaction.'
  ```

  然后客户端使用 JNDI 自动重新连接到辅助数据库并允许：

  - 标识客户端连接到的数据库并保留任何提交的事务。
  - 部分读取的结果集、游标和存储过程调用将丢失。
  - 应用程序重新启动过程或返回到上一个完成的事务或活动。
- **故障恢复到主服务器** – 系统管理员通过在辅助服务器上发出 **sp_failback** 确定何时发生故障恢复。 客户端从辅助服务器故障恢复到主服务器。

  对于客户端而言，故障恢复后主服务器上的行为和结果与向辅助服务器进行故障切换期间相同。

另请参见

- 连接属性 （第 8 页）
- 使用 JNDI 连接到服务器 （第 33 页）

服务器到服务器的远程过程调用

在一台服务器上运行的 Transact-SQL 语言命令或存储过程可以执行位于另一台服务器上的存储过程。

应用程序已连接到的服务器将登录到远程服务器，并执行服务器到服务器的远程过程调用。

应用程序可以指定一个通用口令供服务器间通信使用，即用于所有服务器间连接的口令。 连接打开后，服务器便可使用此口令登录到任何远程服务器。 缺省情况下，SAP jConnect 使用当前连接的口令作为服务器间通信的缺省口令。

但如果同一用户在两台服务器上的口令不同，且该用户打算执行服务器到服务器的远程过程调用，则应用程序必须为要使用的每台服务器显式定义口令。

SAP jConnect 包括一个属性，用来设置通用远程口令或在多个服务器上设置不同的口令。

可使用 `setRemotePassword` 方法在 `SybDriver` 类中设置并配置此属性：

```
Properties connectionProps = new Properties();

public final void setRemotePassword(String serverName,
    String password, Properties connectionProps)
```

若要使用此方法，应用程序必须先导入 `SybDriver` 类，然后再调用此方法：

```
import com.sybase.jdbcx.SybDriver;
SybDriver sybDriver = (SybDriver)
    Class.forName("com.sybase.jdbc4.jdbc.SybDriver").newInstance();
```

```
sybDriver.setRemotePassword
    (serverName, password, connectionProps);
```

注意： 若要为不同的服务器设置不同的远程口令，请为每台服务器重复上述调用过程。

此调用将给定的“服务器名-口令”对添加到给定 Properties 对象中，该对象可以由应用程序在 DriverManager.getConnection (server_url, props) 中传递给 DriverManager。

如果 serverName 为空值，通用口令将设置为用于与所有服务器进行后续连接的口令，但由之前的 setRemotePassword 调用专门定义的除外。

如果应用程序设置了 REMOTEPWD 属性，SAP jConnect 将不再设置缺省的通用口令。

Adaptive Server 的宽表支持

SAP Adaptive Server 15.7 ESD #1 提供的限制和参数要大于以前版本的数据库服务器。

例如：

- 表可以包含 1,024 列。
- varchar 和 varbinary 列可以包含超过 255 个字节的数据。
- 在调用存储过程或者作为 PreparedStatement 的参数进行调用时，最多可以发送和检索 2048 个参数。
- 在连接到 SAP Adaptive Server 15.7 ESD #1 及更高版本时，最多可以发送和检索 32767 个 PreparedStatement 参数。

若要确保 SAP jConnect 从数据库请求宽表支持，JCONNECT_VERSION 的缺省设置必须为 6.0 或更高版本。

注意： 如果将 JCONNECT_VERSION 设置为低于 6.0，SAP jConnect 仍可继续使用 SAP Adaptive Server 12.5 和更高版本。 但如果尝试从需要宽表支持才能完全检索数据的表中选择数据，将可能遇到意外的错误或数据截断情形。

从不支持宽表的 SAP Adaptive Server 访问数据时，也可以将 JCONNECT_VERSION 设置为 6.0 或更高版本。 在这种情况下，服务器只是忽略宽表支持请求。

除了大量列和参数之外，宽表支持还提供扩展结果集元数据。 例如，在低于 SAP jConnect 6.0 的版本中，ResultSetMetaData 方法 getCatalogName、getSchemaName 和 getTableName 都返回未实现 SQLExceptions，因为服务器没有提供元数据。 启用宽表支持后，服务器现将发送回此信息，上述三个方法将返回有用的信息。

访问数据库元数据

为了支持 DatabaseMetaData 方法，SAP Adaptive Server 提供了一组存储过程，SAP jConnect 可以调用这些存储过程以获取数据库的元数据。

如果 SAP Adaptive Server 上尚未安装用于提供元数据的存储过程，则可以使用随 SAP jConnect 提供的存储过程脚本进行安装：

- `sql_server.sql` – 在低于 12.0 版的 SAP Adaptive Server 数据库中安装存储过程。
- `sql_server12.sql` – 在 SAP Adaptive Server 12.0.x 版本的数据库中安装存储过程。
- `sql_server12.5.sql` – 在 SAP Adaptive Server 12.5.x 版本的数据库中安装存储过程。
- `sql_server15.0.sql` – 为 SAP Adaptive Server 15.0 到 15.5 版安装存储过程。
- `sql_server15.7.sql` – 为 SAP Adaptive Server 15.7 到 15.7 ESD #1 版安装存储过程。
- `sql_server15.7.0.2.sql` – 为 SAP Adaptive Server 15.7 ESD #2 或更高版本安装存储过程。
- `sql_server16.0.sql` – 为 SAP Adaptive Server 16.0 版安装存储过程。
- `sql_asa.sql` – 在 SAP SQL Anywhere 9.x 版本的数据库中安装存储过程。
- `sql_asa10.sql` – 在 SAP SQL Anywhere 10.x 版本的数据库中安装存储过程。
- `sql_asa11.sql` – 在 SAP SQL Anywhere 11.x 版本的数据库中安装存储过程。
- `sql_asa12.sql` – 在 SAP SQL Anywhere 12.x 版本的数据库中安装存储过程。
- `sql_asa16.sql` – 在 SAP SQL Anywhere 16.x 版本的数据库中安装存储过程。

注意： 这些脚本的最新版本与所有版本的 SAP jConnect 都兼容。

有关安装存储过程的信息，请参见《SAP jConnect for JDBC 安装指南》和《SAP jConnect for JDBC 发行公告》。

此外，若要使用元数据方法，在建立连接时必须将 `USE_METADATA` 连接属性设置为 true（缺省值）。

不能获取来自数据库中临时表的元数据。

注意： `DatabaseMetaData.getPrimaryKeys` 方法可查找在表定义 (`CREATE TABLE`) 中声明或使用 alter table (`ALTER TABLE ADD CONSTRAINT`) 声明的主键。它不查找使用 **sp_primarykey** 定义的键。

对结果集使用游标

SAP jConnect 实现许多 JDBC 2.0 游标和更新方法。

使用这些方法可以更容易地使用游标，并能更容易地根据结果集中的值更新表中的行。

在 JDBC 2.0 中，`ResultSet` 的主要特点在于其类型和并发性。 类型和并发值是 `java.sql.ResultSet` 接口的一部分，由该接口的 Javadoc 描述。

在收到请求时，如果服务器是 SAP Adaptive Server 15.0 或更高版本，则 SAP jConnect 会打开服务器端可滚动游标。

表 6. SAP jConnect 中提供的 java.sql.ResultSet 选项

并发	类型		
	TYPE_FOR-WARD_ ONLY	**TYPE_SCROLL_ INSENSITIVE**	**TYPE_SCROLL_ SENSITIVE**
CONCUR_READ_ONLY	支持。	支持。	不可用
CONCUR_UPDATABLE	支持。	不可用	不可用

另请参见

- 使用 JDBC 2.0 方法进行定位型更新和删除 （第 53 页）
- PreparedStatement 对象的游标 （第 56 页）
- SAP jConnect 中的 TYPE_SCROLL_INSENSITIVE 结果集 （第 57 页）
- 使用 JDBC 1.x 方法进行定位型更新和删除 （第 52 页）
- 创建和使用游标 （第 51 页）

游标

使用 SAP jConnect 创建游标的方法。

- `SybStatement.setCursorName` – 为游标显式指派名称。

  `SybStatement.setCursorName` 的签名是：

  ```
  void setCursorName(String name) throws SQLException;
  ```
- `SybStatement.setFetchSize` – 创建游标并指定每次读取操作从数据库返回的行数。

  `SybStatement.setFetchSize` 的签名是：

  ```
  void setFetchSize(int rows) throws SQLException;
  ```

  使用 `setFetchSize` 创建游标时，SAP jConnect 驱动程序会为游标命名。 若要获取游标名称，请使用 `ResultSet.getCursorName`。

另一种创建游标的方法是，使用以下 JDBC 方法在连接上指定希望语句返回的 `ResultSet` 的类型：

```
Statement createStatement(int resultSetType, int
resultSetConcurrency)throws SQL Exception
```

如果请求的是不受支持的 `ResultSet`，将在连接上链接一个 SQL 警告。 执行返回的 **Statement** 时，您将收到与所请求的类型最接近的 `ResultSet` 类型。 有关此方法行为的详细信息，请参见 JDBC 规范。

如果不使用 **createStatement**，则 `ResultSet` 的缺省类型如下：

- 如果只调用 `Statement.executeQuery`，则返回的 `ResultSet` 是 `SybResultSet`，其类型和并发值分别为 TYPE_FORWARD_ONLY 和 CONCUR_READ_ONLY。

- 如果调用 setCursorName，则从 executeQuery 返回的 ResultSet 是 SybCursorResultSet，其类型和并发值分别为 TYPE_FORWARD_ONLY 和 CONCUR_UPDATABLE。
- 如果调用 setFetchSize，则从 executeQuery 返回的 ResultSet 是 SybCursorResultSet，其类型和并发值分别为 TYPE_FORWARD_ONLY 和 CONCUR_READ_ONLY。

若要检验 ResultSet 对象的类型是否是您所需要的类型，请使用以下两种 ResultSet 方法：

```
int getConcurrency() throws SQLException;
```

```
int getType() throws SQLException;
```

创建和使用游标

使用 Statement.setCursorName 或 SybStatement.setFetchSize 方法可以创建游标。

1. 使用 Statement.setCursorName 或 SybStatement.setFetchSize 创建游标。
2. 调用 Statement.executeQuery 为语句打开游标并返回游标结果集。
3. 调用 ResultSet.next 读取行并在结果集中定位游标。

   下面的示例分别使用上述两种方法创建游标并返回结果集。 它还使用 ResultSet.getCursorName 获取通过 SybStatement.setFetchSize 创建的游标的名称。

```
// With conn as a Connection object, create a
// Statement object and assign it a cursor using
// Statement.setCursorName().
Statement stmt = conn.createStatement();
stmt.setCursorName("author_cursor");

// Use the statement to execute a query and return
// a cursor result set.
ResultSet rs = stmt.executeQuery("SELECT au_id,
      au_lname, au_fname FROM authors
      WHERE city = 'Oakland'");
while(rs.next())
{
...
}

// Create a second statement object and use
// SybStatement.setFetchSize()to create a cursor
// that returns 10 rows at a time.
SybStatement syb_stmt = conn.createStatement();
syb_stmt.setFetchSize(10);

// Use the syb_stmt to execute a query and return
// a cursor result set.
```

```
SybCursorResultSet rs2 =
      (SybCursorResultSet)syb_stmt.executeQuery
      ("SELECT au_id, au_lname, au_fname FROM authors
       WHERE city = 'Pinole'");
while(rs2.next())
{
...
}

// Get the name of the cursor created through the
// setFetchSize() method.
String cursor_name = rs2.getCursorName();
 ...
```

```
// For jConnect 6.0, create a third statement
// object using the new method on Connection,
// and obtain a SCROLL_INSENSITIVE ResultSet.
// Note: you no longer have to downcast the
// Statement or the ResultSet.
```

```
Statement stmt = conn.createStatement(
                 ResultSet.TYPE_SCROLL_INSENSITIVE,
                 ResultSet.CONCUR_READ_ONLY);
```

```
ResultSet rs3 = stmt.executeQuery
   ("SELECT ... [whatever]");
```

```
// Execute any of the JDBC 2.0 methods that
// are valid for read only ResultSets.
```

```
rs3.next();
rs3.previous();
rs3.relative(3);
rs3.afterLast();
```

```
...
```

使用 JDBC 1.x 方法进行定位型更新和删除

检查使用 JDBC 1.x 的方法。

该示例创建两个 Statement 对象，一个用于选择游标结果集中的行，另一个用于通过结果集中的行更新数据库。

```
// Create two statement objects and create a cursor
// for the result set returned by the first
// statement, stmt1. Use stmt1 to execute a query
// and return a cursor result set.
Statement stmt1 = conn.createStatement();
Statement stmt2 = conn.createStatement();
stmt1.setCursorName("author_cursor");
ResultSet rs = stmt1.executeQuery("SELECT
    au_id,au_lname, au_fname
    FROM authors WHERE city = 'Oakland'
    FOR UPDATE OF au_lname");

// Get the name of the cursor created for stmt1 so
```

```
// that it can be used with stmt2.
String cursor = rs.getCursorName();

// Use stmt2 to update the database from the
// result set returned by stmt1.
String last_name = new String("Smith");
while(rs.next())
```

```
{
    if (rs.getString(1).equals("274-80-9391"))
     {
       stmt2.executeUpdate("UPDATE authors "+
       "SET au_lname = "+last_name +
       "WHERE CURRENT OF " + cursor);
    }
}
```

在结果集中删除
使用 **Statement** 对象 *stmt2* 执行定位型删除

```
stmt2.executeUpdate("DELETE FROM authors
          WHERE CURRENT OF " + cursor);
```

使用 JDBC 2.0 方法进行定位型更新和删除
JDBC 2.0 方法用于更新当前游标行中的列，并可用于通过结果集中的当前游标行更新数据库。

在结果集中更新列
JDBC 2.0 指定了多个方法，用于在客户端更新内存中结果集的列值。

然后可使用更新的值对基础数据库执行更新、插入或删除操作。 所有这些方法都在 `SybCursorResultSet` 类中实现。

下面是 SAP jConnect 中提供的一些 JDBC 2.0 更新方法示例：

```
void updateAsciiStream(String columnName, java.io.InputStream x, int
length)
   throws SQLException;
```

```
void updateBoolean(int columnIndex, boolean x) throws SQLException;
```

```
void updateFloat(int columnIndex, float x) throws SQLException;
```

```
void updateInt(String columnName, int x) throws SQLException;
```

```
void updateInt(int columnIndex, int x) throws SQLException;
```

```
void updateObject(String columnName, Object x) throws SQLException;
```

通过结果集更新数据库的方法
JDBC 2.0 指定了用于根据结果集中当前值来更新或删除数据库中的行的方法。

这些方法在形式上比 JDBC 1.x 中的 `Statement.executeUpdate` 简单，并且不需要使用游标名称。 它们在 `SybCursorResultSet` 中实现：

```
void updateRow() throws SQLException;
void deleteRow() throws SQLException;
```

注意： 结果集的并发值必须是 CONCUR_UPDATABLE。 否则上述方法将引发异常。 对于 `insertRow`，必须指定所有要求非空条目的表列。 `DatabaseMetaData` 提供的方法决定了这些更改何时可见。

示例

下面的示例将创建一个返回游标结果集的 `Statement` 对象。 对于结果集中的每一行，列值在内存中更新，然后数据库通过该行的新列值得到更新。

```
// Create a Statement object and set fetch size to
// 25. This creates a cursor for the Statement
// object Use the statement to return a cursor
// result set.
SybStatement syb_stmt =
(SybStatement)conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIV
E,
    ResultSet.CONCUR_UPDATABLE);
syb_stmt.setFetchSize(25);
SybCursorResultSet syb_rs =
(SybCursorResultSet)syb_stmt.executeQuery(
    "SELECT * from T1 WHERE ...")

// Update each row in the result set according to
// code in the following while loop. jConnect
// fetches 25 rows at a time, until fewer than 25
// rows are left. Its last fetch takes any
// remaining rows.
while(syb_rs.next())
{
  // Update columns 2 and 3 of each row, where
// column 2 is a varchar in the database and
// column 3 is an integer.
  syb_rs.updateString(2, "xyz");
syb_rs.updateInt(3,100);
//Now, update the row in the database.
  syb_rs.updateRow();
}
```

```
// Create a Statement object using the
// JDBC 2.0 method implemented in jConnect 6.0
Statement stmt = conn.createStatement
(ResultSet.TYPE_FORWARD_ONLY, ResultSet.CONCUR_UPDATABLE);
```

```
// In jConnect 6.0, downcasting to SybCursorResultSet is not
// necessary. Update each row in the ResultSet in the same
// manner as above
while (rs.next())
{
rs.updateString(2, “xyz”);
rs.updateInt(3,100);
  rs.updateRow();
```

```
// Use the Statement to return an updatable ResultSet
ResultSet rs = stmt.executeQuery( “SELECT * FROM T1 WHERE...” );
```

```
}
```

从结果集中删除行
从游标结果集删除行。

若要删除行，请使用 SybCursorResultSet.deleteRow：

```
while(syb_rs.next())
{
    int col3 = getInt(3);
    if (col3 >100)
    {
    syb_rs.deleteRow();
    }
}
```

将行插入结果集
使用 JDBC 2.0 API 插入行。

不需要下转到 SybCursorResultSet。

```
// prepare to insert
rs.moveToInsertRow();
```

```
// populate new row with column values
rs.updateString(1, "New entry for col 1");
rs.updateInt(2, 42);
```

```
// insert new row into db
rs.insertRow();
```

```
// return to current row in result set
rs.moveToCurrentRow();
```

在游标关闭时释放锁
SAP Adaptive Server 15.7 扩展了 declare cursor 语法，使其包括 release_locks_on_close 选项，用以在游标关闭时在隔离级别 2 和 3 释放共享游标锁。

SAP jConnect 对 release-lock-on-close 的语义提供相应支持。

要使用 SAP jConnect 连接，请将 RELEASE_LOCKS_ON_CURSOR_CLOSE 连接属性设置为 true。 缺省值为 false。

此设置只有在连接到支持 release_locks_on_close 的服务器时才会生效。

有关 release_locks_on_close 的信息，请参见 SAP Adaptive Server Enterprise《参考手册：命令》。

Select for Update 支持

SAP Adaptive Server 15.7 及更高版本支持 **select for update**，它可以为同一事务内的后续更新锁定行，并支持可更新游标的排它锁。

请参见查询：从表中选择数据（位于《SAP Adaptive Server Enterprise Transact-SQL 用户指南》中）。

当 `for update` 子句添加到 **select** 语句以及客户端内打开的任何可更新游标中后，此功能便可自动用于客户端。

PreparedStatement 对象的游标

可多次使用 `PreparedStatement`，每次使用时可为其输入参数指定相同或不同的值。

如果为 `PreparedStatement` 对象使用游标，则每次使用完游标后必须将其关闭，下次使用时再将其重新打开。关闭游标的结果集 (`ResultSet.close`) 时也会关闭该游标。执行游标的预准备语句 (`PreparedStatement.executeQuery`) 时会打开该游标。

下面的示例演示如何创建 `PreparedStatement` 对象，如何为其指定游标，以及如何执行两次 `PreparedStatement` 对象（关闭然后重新打开游标）。

```
// Create a prepared statement object with a
// parameterized query.
PreparedStatement prep_stmt =
conn.prepareStatement(
"SELECT au_id, au_lname, au_fname "+
"FROM authors WHERE city = ? "+
"FOR UPDATE OF au_lname");

//Create a cursor for the statement.
prep_stmt.setCursorName("author_cursor");

// Assign the parameter in the query a value.
// Execute the prepared statement to return a
// result set.
prep_stmt.setString(1, "Oakland");
ResultSet rs = prep_stmt.executeQuery();

//Do some processing on the result set.
while(rs.next())
```

```
{
    ...
}

// Close the result, which also closes the cursor.
rs.close();

// Execute the prepared statement again with a new
```

```
// parameter value.
prep_stmt.setString(1,"San Francisco");
rs = prep_stmt.executeQuery();

// reopens cursor
```

SAP jConnect 中的 TYPE_SCROLL_INSENSITIVE 结果集

SAP jConnect 支持 TYPE_SCROLL_INSENSITIVE 结果集。

SAP jConnect 使用 Tabular Data Stream (TDS)（即，其专有协议）与 SAP 数据库服务器进行通信。 SAP Adaptive Server 15.0 及更高版本支持 TDS 可滚动游标。 对于不支持 TDS 可滚动游标的服务器，在每次调用 `ResultSet.next` 时，SAP jConnect 都会根据需要在客户端高速缓存行数据。 但到达结果集的末尾时，整个结果集将存储到客户端内存中。因为这可能导致性能降低，因此SAP 建议您仅在使用 SAP Adaptive Server 15.0 或在结果集相当小的情况下才使用 TYPE_SCROLL_INSENSITIVE 结果集。

注意： 在 SAP jConnect 中使用 TYPE_SCROLL_INSENSITIVE `ResultSets` 时，如果服务器不支持 TDS 可滚动游标，则只能在读取完 **ResultSet** 的最后一行后才能调用 `isLast` 方法。 在未达到最后一行时调用 `isLast` 会导致抛出 `UnimplementedOperationException`。

SAP jConnect 在 `sample2` 目录中提供了 `ExtendResultSet`；此示例使用 JDBC 1.0 接口提供了一个受限制的 TYPE_SCROLL_INSENSITIVE `ResultSet`。

此实现使用标准 JDBC 1.0 方法生成无滚动功能的只读结果集，即基础数据的一个静态视图，它不会即时反映在结果集为打开状态时所进行的更改。
`ExtendedResultSet` 在客户端高速缓存所有 `ResultSet` 行。 对较大的结果集使用此类时应谨慎。

`sample.ScrollableResultSet` 接口：

- 是 JDBC 1.0 `java.sql.ResultSet` 的扩展。
- 定义了与 JDBC 2.0 `java.sql.ResultSet` 具有相同签名的其它方法。
- 只包含部分 JDBC 2.0 方法。 未包含其中用于修改 `ResultSet` 的方法。

来自 JDBC 2.0 API 的方法有：

```
boolean previous() throws SQLException;

boolean absolute(int row) throws SQLException;
boolean relative(int rows) throws SQLException;

boolean first() throws SQLException;
boolean last() throws SQLException;
void beforeFirst() throws SQLException;
void afterLast() throws SQLException;

boolean isFirst() throws SQLException;
boolean isLast() throws SQLException;
boolean isBeforeFirst() throws SQLException;
boolean isAfterLast() throws SQLException;

int getFetchSize() throws SQLException;
void setFetchSize(int rows) throws SQLException;
```

```
int getFetchDirection() throws SQLException;
void setFetchDirection(int direction) throws SQLException;
```

```
int getType() throws SQLException;
int getConcurrency() throws SQLException;
int getRow() throws SQLException;
```

若要使用示例类，请使用任意 JDBC 1.0 `java.sql.ResultSet` 创建一个 `ExtendedResultSet`。 以下为相关的代码段（假定为 Java 1.1 环境）：

```
// import the sample files
import sample.*;
```

```
//import the JDBC 1.0 classes
import java.sql.*;
```

```
// connect to some db using some driver;
// create a statement and a query;
```

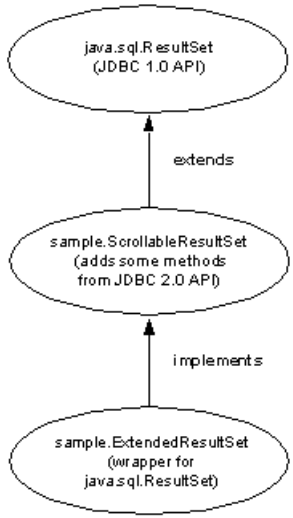
```
// Get a reference to a JDBC 1.0 ResultSet
ResultSet rs = stmt.executeQuery(_query);
```

```
// Create a ScrollableResultSet with it
ScrollableResultSet srs = new ExtendedResultSet(rs);
```

```
// invoke methods from the JDBC 2.0 API
srs.beforeFirst();
```

```
// or invoke methods from the JDBC 1.0 API
if (srs.next())
  String column1 = srs.getString(1);
```

图 1： 类框图显示了示例类和 JDBC API 之间的关系



有关详细信息，请参见 JDBC 2.0 API，网址为：Oracle Technology Network for Java。

包含 COMPUTE 子句的 Transact-SQL 查询

SAP jConnect for JDBC 支持包含 COMPUTE 子句的 Transact-SQL 查询。

COMPUTE 子句允许使用一个 **select** 语句显示明细和摘要结果。 摘要行显示在特定组的明细行的下面。 例如：

```
select type, price, advance
   from titles
   order by type
   compute sum(price), sum(advance) by type
```

```
type          price      advance
------------  ---------  ----------
UNDECIDED     NULL       NULL

Compute Result:
----------------------- -------------------------
NULL                    NULL

type          price      advance
------------  ---------  ----------
business      2.99       10,125.00
business      11.95      5,000.00

business      19.99      5,000.00
business      19.99      5,000.00

Compute Result:
----------------------- -------------------------
54.92                   25,125.00

...
...

(24 rows affected)
```

当 SAP jConnect 执行包含 COMPUTE 子句的 **select** 语句时，SAP jConnect 会向客户端返回多个结果集。 结果集的数量取决于可用的唯一分组的数量。 每个组包含一个用于明细行的结果集和一个用于摘要的结果集。 客户端必须处理所有结果集才能完全处理返回的行，否则返回的第一个结果集中将只包括第一个数据组的明细行。

有关 COMPUTE 子句的详细信息，请参见《SAP Adaptive Server Enterprise Transact-SQL 用户指南》。 有关处理多个结果集的详细信息，请参见 Oracle Technology Network for Java 站点上提供的 JDBC API 文档。

批处理更新支持

批处理更新允许一个 Statement 对象向基础数据库提交多个语句，这些语句作为一个单元（一批）一起进行处理。

添加到批处理的任何语句必须仅返回更新计数，不得返回 ResultSet。

有关对 Statement、PreparedStatement 和 CallableStatement 使用批处理更新的示例，请参见 sample2 子目录中的 BatchUpdates.java。

SAP jConnect 还支持动态 PreparedStatements 的批处理。

实现说明

SAP jConnect 按照 JDBC 2.0 API 中指定的方式实现批处理更新。

例外情况包括：

- EXECUTE_BATCH_PAST_ERRORS 连接属性控制故障在批处理执行中的处理方式。
  缺省情况下，EXECUTE_BATCH_PAST_ERRORS 设置为 false，SAP jConnect 会在出现第一个故障后停止处理。 BatchUpdateException.getUpdateCounts 返回长度为 M < N 的 int[] 数组，表示批处理中的前 M 个语句成功，第 M+1 个语句失败，第 M+2..N 个语句没有执行。 “N”表示批处理中的语句总数。
  如果将 EXECUTE_BATCH_PAST_ERRORS 设置为 true，则 SAP jConnect 会在出现非致命故障时继续处理。 BatchUpdateException.getUpdateCounts 返回长度为 N 的 int[] 数组，其中“N”表示批处理中的语句总数。 检查各项更新计数便可确定每个语句的执行状态。
- 若要以批处理（非链式）模式调用存储过程，必须以非链式模式创建存储过程。
- 如果 SAP Adaptive Server 在批处理执行过程中遇到致命错误，则 BatchUpdateException.getUpdateCounts 仅返回长度为零的 int[ ]。发生致命错误时整个事务将回退，因此成功操作的行数为零。
- 不支持批处理更新的数据库中的批处理更新：即使数据库不支持批处理更新，SAP jConnect 仍可在 executeUpdate 循环中执行批处理更新。 这样无论指向哪个数据库，都可以使用相同的批处理代码。

有关批处理更新的详细信息，请参见 JDBC API 文档。

另请参见

- 在非链式事务模式中执行存储过程 （第 124 页）

通过存储过程的结果集更新数据库

SAP jConnect 提供 **update** 方法和 **delete** 方法，用于在由存储过程返回的结果集中获取游标。

然后可使用游标的位置更新或删除提供结果集的基础表中的行。 这些方法位于 SybCursorResultSet 中：

```
void updateRow(String tableName) throws SQLException;

void deleteRow(String tableName) throws SQLException;
```

tableName 参数标识提供结果集的数据库表。

若要获取存储过程返回的结果集中的游标，先使用 SybCallableStatement.setCursorName 或

SybCallableStatement.setFetchSize，然后再执行包含该过程的可调用语句。此示例演示如何在存储过程的结果集中创建游标，更新结果集中的值，然后使用 SybCursorResultSet.update 方法更新基础表：

```
// Create a CallableStatement object for executing the stored
// procedure.
CallableStatement sproc_stmt =
   conn.prepareCall("{call update_titles}",
     ResultSet.TYPE_FORWARD_ONLY, ResultSet.CONCUR_UPDATABLE);

// Set the number of rows to be returned from the database with
// each fetch. This creates a cursor on the result set.
(SybCallableStatement)sproc_stmt.setFetchSize(10);

//Execute the stored procedure and get a result set from it.
SybCursorResultSet sproc_result = (SybCursorResultSet)
   sproc_stmt.executeQuery();

// Move through the result set row by row, updating values in the
// cursor's current row and updating the underlying titles table
// with the modified row values.
while(sproc_result.next())
{
   sproc_result.updateString(...);
   sproc_result.updateInt(...);
   ...
   sproc_result.updateRow(titles);
}
```

数据类型

检查 numeric、image、text、date、time 和 char 数据的使用情况。

Numeric 数据类型

SybPreparedStatement 扩展支持 SAP Adaptive Server 处理 NUMERIC 数据类型的方式，可为该数据类型指定精度（总位数）和标度（小数点后的位数）。

Java 中与此对应的数据类型 java.math.BigDecimal 稍有不同，当 SAP jConnect 应用程序使用 setBigDecimal 方法控制输入/输出参数的值时，这些差异会引发问题。具体地说，有时参数（无论是存储过程参数还是列）的精度和标度必须与对应的 SQL 对象的精度和标度完全一致。

SybPreparedStatement 扩展与以下方法结合使用来加强 SAP jConnect 应用程序对 setBigDecimal 的控制：

```
public void setBigDecimal (int parameterIndex, BigDecimal X, int
scale,
  int precision) throws SQLException
```

有关详细信息，请参见 SAP jConnect 安装目录下 /sample2 子目录中的 SybPrepExtension.java 示例。

Image 数据类型

SAP jConnect 的 TextPointer 类包含 sendData 方法，用于更新 SAP Adaptive Server 或 SAP SQL Anywhere 数据库中的 image 列。

在低于 4.0 的 SAP jConnect 版本中，必须在 java.sql.PreparedStatement 中使用 setBinaryStream 方法发送图像数据。 在 5.0 及更高版本中，TextPointer.sendData 方法使用 java.io.InputStream 将图像数据发送到 SAP Adaptive Server 数据库中，并且极大地提高了性能。

警告！ 使用 TextPointer 类的 sendData() 方法可能会影响应用程序，因为 TextPointer 不是标准的 JDBC 格式。

SAP 建议使用标准 JDBC 格式 PreparedStatement.setBinaryStream(int paramIndex, InputStream image) 或 LOB 定位符支持来发送图像数据。 但是，在处理大图像数据时，setBinaryStream() 消耗的过程高速缓存中的内存可能要远远多于 **TextPointer** 类。

在实现 TextPointer 类的替换之前，SAP 将继续支持它。

若要获取 TextPointer 类的实例，可以在 SybResultSet 中使用以下两种方法之一：

- `public TextPointer getTextPtr(String columnName)`
- `public TextPointer getTextPtr(int columnIndex)`

TextPointer 类中的公共方法

检查 SAP jConnect 中 TextPointer 类的公共方法。

com.sybase.jdbcx 包中包含 TextPointer 类。 其公共方法接口为：

```
public void sendData(InputStream is, boolean log)
   throws SQLException

public void sendData(InputStream is, int length,
   boolean log) throws SQLException

public void sendData(InputStream is, int offset,
   int length, boolean log) throws SQLException

public void sendData(byte[] byteInput, int offset,
   int length, boolean log) throws SQLEXception
```

其中：

- **sendData(InputStream** is**, boolean** *log)* 用指定的输入流中的数据更新 image 列。
- **sendData(InputStream** is**, int** *length*, **boolean** *log)* 用指定的输入流中的数据更新 image 列。 *length* 是发送的字节数。

- **sendData(InputStream** is **, int** *offset*, **int** *length*, **boolean** *log)* 用指定的输入流中的数据更新 image 列，从 *offset* 参数中给定的字节偏移处开始传送，直到传送完 *length* 参数中指定的字节数。
- **sendData(byte[ ]** *byteInput*, **int** *offset*, **int** *length*, **boolean** *log)* 用 *byteInput* 参数指定的字节数组中所包含的图像数据更新列。 更新从 *offset* 参数中给定的字节偏移处开始，一直持续到读取完 *length* 参数中指定的字节数为止。
- *log* 是各种方法的参数，用于指定 image 数据是否要完全记录到数据库的事务日志中。 如果 **log** 参数设置为 true，则整个二进制图像都将写入到事务日志中。 如果 **log** 参数设置为 false，则将记录更新操作，但图像本身并不写入日志。

TextPointer 对象

text 和 image 列包含 timestamp 和页位置信息，这些信息与列文本和图像数据分开存放。

在从 text 或 image 列中选取数据时，此额外信息作为结果集的一部分“隐含”起来。

用于更新 image 列的 TextPointer 对象需要此隐含信息，但不需要列数据的图像部分。 为了获取此信息，需将这一列选取到 ResultSet 对象中，然后使用 SybResultSet.getTextPtr 提取文本指针信息，忽略图像数据，并创建 TextPointer 对象。

当一列中包含大量的图像数据时，为一行或多行选取列并等待获取所有数据很可能效率很低，因为不需要使用这些数据。 为了缩短这一过程，请使用 **set textsize** 命令最小化数据包中返回的数据量。 为了达到这一目的，以下代码示例在获取 TextPointer 对象时使用了 **set textsize**。

```
/*
 * Define a string for selecting pic column data for author ID
 * 899-46-2035.
 */
 String getColumnData = "select pic from au_pix where au_id =
'899-46-2035'";

 /*
 * Use set textsize to return only a single byte of column data
 * to a Statement object. The packet with the column data will
 * contain the "hidden" information necessary for creating a
 * TextPointer object.
 */
 Statement stmt= connection.createStatement();
 stmt.executeUpdate("set textsize 1");

 /*
 * Select the column data into a ResultSet object--cast the
 * ResultSet to SybResultSet because the getTextPtr method is
 * in SybResultSet, which extends ResultSet.
 */
 SybResultSet rs = (SybResultSet)stmt.executeQuery(getColumnData);
```

```
/*
* Position the result set cursor on the returned column data
* and create the desired TextPointer object.
*/
rs.next();
TextPointer tp = rs.getTextPtr("pic");

/*
* Now, assuming we are only updating one row, and won't need
* the minimum textsize set for the next return from the server,
* we reset textsize to its default value.
*/
stmt.executeUpdate("set textsize 0");
```

使用 TextPointer.sendData 执行更新
借助 TextPointer 对象，使用 Anne_Ringer.gif 文件中的图像数据来更新 pic 列。

示例代码:

```
/*
 *First, define an input stream for the file.
 */
 FileInputStream in = new FileInputStream("Anne_Ringer.gif");

 /*
 * Prepare to send the input stream without logging the image data
 * in the transaction log.
 */
 boolean log = false;

 /*
 * Send the image data in Anne_Ringer.gif to update the pic
 * column for author ID 899-46-2035.
 */
 tp.sendData(in, log);
```

有关详细信息，请参见 SAP jConnect 安装目录下 sample2 子目录中的 TextPointers.java 示例。

使用 TextPointer.sendData 更新 Image 列
使用 TextPointer.sendData 更新具有图像数据的列。

1. 为要更新的行和列获取一个 TextPointer 对象。
2. 使用 TextPointer.sendData 执行更新操作。

本示例发送 Anne_Ringer.gif 文件中的 image 数据，用以更新 pubs2 数据库中 au_pix 表的 pic 列。 针对作者 ID 为 899-46-2035 的行执行这一更新操作。

Text 数据类型

在 SAP jConnect 3.0 及更低版本中，使用 `TextPointer` 类和 `sendData` 方法更新 SAP Adaptive Server 或 SAP SQL Anywhere 数据库中的 `text` 列。

Java 已不再支持 `TextPointer` 类，即在 Java 的后续版本中不再推荐使用，并且可能会取消该类。

如果使用的数据服务器是 SAP Adaptive Server 或 SAP SQL Anywhere，请使用标准 JDBC 格式发送文本数据：

```
PreparedStatement.setAsciiStream(int paramIndex,
   InputStream text, int length)
```

或：

```
PreparedStatement.setUnicodeStream(int paramIndex,
   InputStream text, int length)
```

或：

```
PreparedStatement.setCharacterStream(int paramIndex,    Reader
reader, int length)
```

Date 和 Time 数据类型

SAP jConnect for JDBC 支持 SAP Adaptive Server `datetime`、`smalldatetime`、`bigdatetime`、`bigtime`、`date` 和 `time` 数据类型：

- `datetime` 可保存从 1753 年 1 月 1 日到 9999 年 12 月 31 日之间的日期。在支持 1/300 秒精度级别的平台上，datetime 可精确到该级别。
- `smalldatetime` 可保存从 1900 年 1 月 1 日到 2079 年 6 月 6 日之间的日期，可以精确到分钟。
- `bigdatetime` 表示自 0000 年 1 月 1 日 0:00:00.000000 以来所经过的微秒数。`bigdatetime` 值的合法范围为 0001 年 1 月 1 日 00:00:00.000000 到 9999 年 12 月 31 日 23:59:59.999999。
- `bigtime` 表示自一天开始所经过的微秒数。`bigtime` 值的合法范围为 00:00:00.000000 到 23:59:59.999999。
- `date` 可保存从 0001 年 1 月 1 日到 9999 年 12 月 31 日之间的日期，与 `java.sql.Date` 中允许使用的值完全匹配。在 `java.sql.Date` 与 `date` 数据类型间存在直接映射。
- `time` 可保存从 00:00:00:000 到 23:59:59:990 之间的时间。在 `java.sql.Time` 与 `time` 数据类型间存在直接映射。

Date、Time、Datetime 和 Smalldatetime 数据类型

SAP jConnect for JDBC 支持 date、time、datetime 和 smalldatetime。

如果从包含 `date` 或 `time` 列的表中选择数据，并且尚未（通过设置 SAP jConnect 版本）在 SAP jConnect 中启用 `date/time` 支持，则服务器会在返回 `date` 或 `time` 之前尝试将其转换为 `datetime` 值。

- 如果要返回的日期早于 1753 年 1 月 1 日，可能会产生问题。 在这种情况下，会出现转换错误，数据库会向您通知该错误。
- SAP SQL Anywhere 支持 `date` 和 `time` 数据类型，但这些数据类型并不直接与 SAP Adaptive Server 12.5.1 及更高版本中的数据类型相兼容。 在使用 SAP jConnect 与 SAP SQL Anywhere 进行通信时，应继续使用 `datetime` 和 `smalldatetime` 数据类型。
- 在 SAP SQL Anywhere 中，`datetime` 列中的最大值是 1-1-7911 00:00:00。
- 使用 SAP jConnect 时，如果尝试将早于 1753 年 1 月 1 日的日期插入 `datetime` 列或参数中，则会收到转换错误。
- 有关 `date` 和 `time` 数据类型的详细信息，请参见 SAP Adaptive Server 手册；请特别注意其中有关可执行的隐式转换的信息。
- 如果对 SAP Adaptive Server `date`、`time` 或 `datetime` 列使用 `getObject`，则返回的值分别为 `java.sql.Date`、`java.sql.Time` 或 `java.sql.Timestamp` 数据类型。

Bigdatetime 和 Bigtime 数据类型

连接到 SAP Adaptive Server 15.5 及更高版本时，SAP jConnect 会使用 `bigdatetime` 和 `bigtime` 数据类型传输数据，即使接收 SAP Adaptive Server 列定义为 `datetime` 和 `time` 也是如此。

- 这意味着 SAP Adaptive Server 可能会不提示而直接截断来自 SAP jConnect 的值，从而满足 SAP Adaptive Server 列的要求。 例如，在数据类型为 `time` 的 Adaptive Server 列中，`bigtime` 值 23:59:59.999999 保存为 23:59:59.996。
- 连接到 SAP Adaptive Server 15.0.x 及更低版本时，SAP jConnect for JDBC 使用 `datetime` 和 `time` 数据类型传输数据。

Char、Varchar、Text 和 GetByte 数据类型

如果数据不是十六进制、八进制或十进制数据，请勿对 `char`、`univarchar`、`unichar`、`varchar` 或 `text` 字段使用 `rs.getByte`。

支持的其它数据类型

检查 SAP jConnect 所支持的其它 SAP Adaptive Server 数据库。

SAP jConnect 支持以下 SAP Adaptive Server 数据类型：

- `bigint` – 一种精确数值数据类型，设计为在现有 `int` 类型范围不足时使用。
- `unsigned int` – 无符号形式的精确数值整数数据类型：`unsignedsmallint`、`unsignedint` 和 `unsignedbigint`。
- `unitext` – 用于 Unicode 字符的可变长度数据类型。

Bigint 数据类型

`bigint` 是一种 64 位整数数据类型，它作为本机 SAP Adaptive Server 数据类型而受到支持。

`bigint` 映射到 Java 数据类型 `long`。 若要将此数据类型用作参数，可以调用 `PreparedStatement.setLong(int index, long value)`，SAP jConnect 即会将数据作为`bigint`发送给 SAP Adaptive Server。 从`bigint`列进行检索时，可以使用 `ResultSet.getLong(int index)` 方法。

Unitext 数据类型

使用 `unitext` 列时，SAP jConnect 将在 SAP Adaptive Server 内部存储并检索数据。

Unsigned Int 数据类型

SAP Adaptive Server 支持将 `unsigned bigint`、`unsigned int` 和 `unsigned smallint` 作为本机 SAP Adaptive Server 数据类型。

由于在 Java 中并没有与之相对应的无符号数据类型，因此必须用 `set` 和 `get` 语句设置并获取下一个较大整数才能正确处理数据。 例如，如果要从 *unsigned int* 中检索数据，使用 Java 数据类型 `int` 则会太小，不能包含大的正值，因此，`ResultSet.getInt (int index)` 可能会返回不正确的数据或抛出异常。 若要正确处理数据，应该用 `get` 语句获取下一个较大整数值 `ResultSet.getLong()`。

SAP Adaptive Server 数据类型	Java 数据类型
`unsigned smallint`	`setInt()`、`getInt()`
`unsigned int`	`setLong()`、`getLong()`
`unsigned bigint`	`setBigDecimal()`、`getBigDecimal()`

DOL 锁定表中的可变长度行

如果可变长度列从行起始位置后超过 8191 字节处开始，则配置了 16K 逻辑页大小的低于 15.7 版的 SAP Adaptive Server 无法创建含有可变长度行的 DOL 锁定表。

从 SAP Adaptive Server 15.7 开始删除了这一限制。 请参见“数据存储”（SAP Adaptive Server Enterprise《性能和调优系列： 物理数据库调优》）。

JDBC 客户端不需要特殊配置即可使用此功能。 当连接到配置为接收 DOL 宽行的 SAP Adaptive Server 15.7 版时，这些客户端会自动使用宽偏移插入记录。 如果客户端尝试向 SAP Adaptive Server 的早期版本或禁用了 DOL 宽行选项的 15.7 版 SAP Adaptive Server 发送 DOL 宽行，则会收到一条错误消息。

大对象 (LOB) 支持

SAP jConnect 支持使用大对象 (LOB) 数据类型 `text`、`unitext` 和 `image`

- 具有行内存储的 LOB 列 – 在 SAP Adaptive Server 中，当有足够的内存存储整个行时，标记为行内的 LOB 列将存储在行内。 如果由于对行中的列进行更新而使行大小增大并超过定义的限制值，在行内存储的 LOB 列会移动到行外以使其仍不超过限制值。 请参见《SAP Adaptive Server Enterprise Transact-SQL 用户指南》中的“行内、行外 LOB”。

  SAP jConnect 中的批量插入例程支持 SAP Adaptive Server 中 `text`、`image` 和 `unitext` LOB 列的行内和行外存储。 早期客户端版本中的批量插入例程始终将 LOB 列存储在行外。
- 用作存储过程参数的 LOB 对象 – SAP jConnect 支持将 `text`、`unitext` 和 `image` 用作存储过程中的输入参数以及参数标记数据类型。

大对象定位符支持

SAP jConnect 支持大对象 (LOB) 定位符。 LOB 定位符包含指向 LOB 数据的逻辑指针，而不是数据本身，减少了通过网络在 SAP Adaptive Server 和其客户端之间传送的数据量。

SAP Adaptive Server 15.7 中引入了对 LOB 定位符的服务器支持。

当连接到支持 LOB 定位符的 SAP Adaptive Server 并关闭 `autocommit` 时，SAP jConnect 使用服务器端定位符访问 LOB 数据。 否则，jConnect 会在客户端实现 LOB 数据。 可以将整个 LOB API 用于客户端实现的 LOB 数据，但由于数据较大，API 性能可能会与用于 LOB 定位符时不同。

注意： 当您使用 LOB 定位符时，检索每个行上都包括 LOB 数据的大结果集可能会影响应用程序的性能。 SAP Adaptive Server 将 LOB 定位符作为结果集的一部分返回，为获取 LOB 数据，SAP jConnect 必须缓存剩余的结果集。 SAP 建议您让结果集小一些，或者启用游标支持来限制要缓存的数据的大小。

要启用 LOB 定位符支持，请在 `ENABLE_LOB_LOCATORS` 连接属性设置为 true 的情况下与 SAP Adaptive Server 建立连接。 启用后，客户端应用程序便可使用 `java.sql` 软件包中的 `Blob`、`Clob` 和 `NClob` 类访问定位符。

注意： 如果 LOB 定位符和 `autocommit` 均已启用，SAP jConnect 会自动将 LOB 定位符切换为客户端实现的 LOB，即使 SAP Adaptive Server 能够支持 LOB 定位符也是如此。 这会增加客户端使用的内存，而且可能会降低性能。 因此，建议您在设置 `autocommit off` 的条件下使用 LOB 定位符。

有关 `Blob`、`Clob` 和 `NClob` 类的详细信息，请参见 Java 文档。

SAP jConnect 中的高级功能

SAP jConnect 提供高级功能，如事件通知、错误消息处理、口令加密、动态类装载以及 JDBC 规范支持。

查看此说明以使用 SAP jConnect 支持的高级功能。

另请参见

- BCP 插入 （第 69 页）
- 受支持的 SAP Adaptive Server Cluster Edition 功能 （第 70 页）
- 事件通知 （第 71 页）
- 错误消息 （第 73 页）
- 口令加密 （第 77 页）
- JDBC 4.0 规范支持 （第 86 页）
- 将 Java 对象作为列数据存储在表中 （第 79 页）
- 动态类装载 （第 83 页）
- JDBC 3.0 规范支持 （第 87 页）
- JDBC 2.0 选件工具包扩展支持 （第 89 页）

BCP 插入

SAP jConnect 支持使用 bulk-load 插入功能将大量行插入 SAP Adaptive Server 12.5.2 和更高版本中。

使用此功能不需要在服务器上进行特殊配置，但如果增加页大小和网络包大小并最大化内存大小，将能够显著提升性能。

此外，根据客户端内存，使用较大的批处理文件也可改进性能。

要启用 bulk-load 插入，请将 `ENABLE_BULK_LOAD` 设置为以下值之一：

- ARRAYINSERT_WITH_MIXED_STATEMENTS – 使用行级别日志记录启用批量装载，并允许应用程序在批量装载操作过程中执行其它语句。
- ARRAYINSERT – 使用行级别日志记录启用批量装载，但应用程序在执行批量装载操作期间无法执行其它语句。
- BCP – 使用页面级别日志记录启用批量装载，但应用程序在批量装载操作过程中无法执行其它语句。
- LOG_BCP – 使用页面级别日志记录（使用 SAP Adaptive Server 快速记录 BCP 功能）启用批量装载；但应用程序在批量装载操作过程中无法执行其它语句。

使用预准备语句并且将 `ENABLE_BULK_LOAD` 设置为有效值时，SAP jConnect 会使用 BULK 例程将一批记录插入 SAP 数据库。

启用批量装载的限制
ENABLE_BULK_LOAD 的一些限制：

- 忽略选定表上的触发器。
- 不验证空值和引用约束。
- 不支持计算列和加密列。
- 如果指定了重叠的标识值范围，可能会创建重复的标识值。
- 发生并发数据输入时，可能会插入相冲突的 IDENTITY 值。

受支持的 SAP Adaptive Server Cluster Edition 功能

SAP jConnect 支持 SAP Adaptive Server Cluster Edition 环境，其中多个 SAP Adaptive Server 与一组共享磁盘和高速专用互连相连接。 这样，SAP Adaptive Server 即可使用多个物理和逻辑主机进行扩展。

有关 Cluster Edition 的详细信息，请参见《SAP Adaptive Server Enterprise 集群用户指南》。

登录重定向

当客户端应用程序尝试连接到繁忙的服务器时，登录重定向可允许服务器将客户端连接重定向到集群中不太繁忙的服务器，从而帮助平衡服务器的负荷。

在任何给定时间，集群环境中通常有一些服务器的工作负荷比其它服务器高。 登录重定向过程在登录序列中进行，客户端应用程序不会收到被重定向的通知。 当客户端应用程序连接到支持登录重定向功能的服务器时，会自动启用此功能。

注意： 当客户端应用程序连接到配置为重定向客户端的服务器时，登录时间可能会增加，因为在将客户端连接重定向到另一台服务器时，会重新启动登录过程。

连接迁移

连接迁移允许集群环境中的服务器动态分配负荷，并将现有客户端连接及其上下文无缝迁移到集群中的另一台服务器。

通过此功能，集群环境能够充分利用资源并减少计算时间。 由于服务器间进行的连接迁移是无缝迁移，因此通过这种方式还可帮助创建零停机时间的高可用性环境。 当客户端应用程序连接到支持连接迁移功能的服务器时，会自动启用此功能。

注意： 在服务器迁移过程中，命令执行时间可能会增加。 SAP 建议您相应地增加命令超时时间。

连接故障切换

连接故障切换功能允许客户端应用程序在主服务器因意外事件（例如断电或套接字失败）变得不可用时切换到备用 SAP Adaptive Server。

在集群环境中，客户端应用程序可以使用动态故障切换地址多次故障切换到多台服务器。

启用高可用性后，客户端应用程序无需配置为知道可能的故障切换目标。 SAP Adaptive Server 始终使用基于集群成员资格、逻辑集群使用情况和负荷分配的最佳故障切换列表更新客户端。 在故障切换过程中，客户端参照有序的故障切换列表来尝试重新连接。 如果驱动程序成功连接到服务器，则驱动程序会在内部根据返回的列表更新主机值列表。 否则，驱动程序会引发连接失败异常。

注意： 连接属性 **DEFAULT_QUERY_TIMEOUT** 和 **INTERNAL_QUERY_TIMEOUT** 或 **DriverManager.setLoginTimeout(xx)** 在发生故障切换后将失败的节点切换到高可用性节点的过程中起关键作用。

启用连接故障切换

可以使用连接字符串通过将 `REQUEST_HA_SESSION` 设置为 true 来启用连接故障切换。

例如：

```
URL="jdbc:sybase:Tds:server1:port1,server2:port2,...,
serverN:portN/mydb?REQUEST_HA_SESSION=true"
```

其中 server1:port1, server2:port2, ... , serverN:portN 是有序故障切换列表。

SAP jConnect 会尝试连接到故障切换列表中指定的第一个主机和端口。 如果失败，则会遍历列表，直到建立连接，或直到到达列表末尾。

注意： 连接字符串中指定的备用服务器列表只在初始连接过程中使用。 使用任何可用实例建立连接后，如果客户端支持高可用性，则客户端会从服务器收到最可能成为故障切换目标的更新列表。 此新列表将覆盖指定列表。

事件通知

可以使用事件通知让应用程序在执行 SAP Open Server 过程时获得通知。

若要使用这一功能，必须使用 `SybConnection` 类，此类扩展了 `Connection` 接口。 `SybConnection` 包含 `regWatch` 方法和 `regNoWatch` 方法，分别用于打开事件通知和关闭事件通知。

应用程序还必须实现 `SybEventHandler` 接口。 该接口包含一个公共方法 `void event(String proc_name, ResultSet params)`，在发生指定事件时将调用该方法。 事件的参数被传递给 **event**，后者告知应用程序如何进行响应。

若要在应用程序中使用事件通知，请调用 `SybConnection.regWatch( )` 将应用程序注册到已注册过程的通知列表中：

```
SybConnection.regWatch(proc_name,eventHdlr,option)
```

其中：

- *proc_name* 是一个字符串，是用于生成通知的注册过程的名称。
- *eventHdler* 是实现的 `SybEventHandler` 类的实例。

- *option* 是 NOTIFY_ONCE 或 NOTIFY_ALWAYS。 如果希望应用程序仅在过程首次执行时得到通知，请使用 NOTIFY_ONCE。 如果希望应用程序在过程每次执行时均得到通知，请使用 NOTIFY_ALWAYS。

每当 SAP Open Server 上发生具有指定 *proc_name* 的事件时，SAP jConnect 都会从独立线程中调用 **eventHdlr.event**。 **eventHdlr.event** 在执行时会接收传递来的事件参数。 由于这是一个独立的线程，因此事件通知不会阻止应用程序的执行。

如果 *proc_name* 不是已注册过程，或者 SAP Open Server 无法将客户端添加到通知列表中，则调用 **regWatch** 会抛出 SQL 异常。

要关闭事件通知，请执行以下命令：

```
SybConnection.regNoWatch(proc_name)
```

警告！ 使用 SAP jConnect 事件通知扩展时，应用程序必须对连接调用 **close** 方法，以删除首次调用 **regWatch** 时创建的子线程。 否则，当退出应用程序时可能会导致虚拟机停止响应。

事件通知示例

建立连接后，查看说明以实现事件处理程序，然后向该事件处理程序的某个实例注册事件。

事件通知示例代码：

```
public class MyEventHandler implements SybEventHandler
{
  // Declare fields and constructors, as needed.
  ...
  public MyEventHandler(String eventname)
  {
    ...
  }

  // Implement SybEventHandler.event.
  public void event(String eventName, ResultSet params)
  {
    try
    {
      // Check for error messages received prior to event
      // notification.
      SQLWarning sqlw = params.getWarnings();
      if sqlw != null
      {
        // process errors, if any
        ...
      }
      // process params as you would any result set with
      // one row.
      ResultSetMetaData rsmd = params.getMetaData();
      int numColumns = rsmd.getColumnCount();
      while (params.next())          // optional
      {
        for (int i = 1; i <= numColumns; i++)
```

```
        {
          System.out.println(rsmd.getColumnName(i) + " =
            " + params.getString(i));
        }
        // Take appropriate action on the event. For example,
        // perhaps notify application thread.
        ...
      }
    }
    catch (SQLException sqe)
    {
      // process errors, if any
      ...
    }
  }
}

public class MyProgram
{
  ...
  // Get a connection and register an event with an instance
  // of MyEventHandler.
  Connection conn = DriverManager.getConnection(...);
  MyEventHandler myHdlr = new  MyEventHandler("MY_EVENT");

  // Register your event handler.
  ((SybConnection)conn).regWatch("MY_EVENT", myHdlr,
    SybEventHandler.NOTIFY_ALWAYS);
  ...
 conn.regNoWatch("MY_EVENT");
  conn.close();
}
```

错误消息

SAP jConnect 提供两个用于返回特定于 SAP jConnect 的错误信息的类：`SybSQLException` 和 `SybSQLWarning`，并提供一个 `SybMessageHandler` 接口，用于自定义 SAP jConnect 处理来自服务器的错误消息的方式。

作为警告返回的数字错误

在 SAP Adaptive Server12.0 到 12.5 中，缺省情况下，将数字错误作为严重级 10 进行处理。

严重级为 10 的消息归类为状态信息性消息，而不是归类为错误，其内容将传输到 `SQLWarning` 对象。

以下代码演示了这一过程：

```
static void processWarnings(SQLWarning warning)
{
if (warning != null)
 {
  System.out.println ("\n -- Warning received -- \n");
```

```
 }//end if
  while (warning != null)
 {
  System.out.println ("Message: " +   warning.getMessage());
  System.out.println("SQLState: " +   warning.getSQLState());
  System.out.println ("ErrorCode: " +
  warning.getErrorCode());
  System.out.println ("----------------------------");
  warning = warning.getNextWarning();
 }//end while
}//end processWarnings
```

当出现数字错误时，返回的 ResultSet 对象中不包含结果集数据，必须从 SQLWarning 获取此错误的相关信息。 因此，在 JDBC 应用程序中，检查并处理 SQLWarning 的代码不应依赖于结果集。 例如，以下代码通过处理 while 循环来检查并处理结果集内部和外部的 SQLWarning 数据：

```
while (rs.next())
{
 String value = rs.getString(1);
 System.out.println ("Fetched value: " + value);

 // Check for SQLWarning on the result set.
 processWarnings (rs.getWarnings());

}//end while

 // Check for SQLWarning on the result set.
 processWarnings (rs.getWarnings());
```

此处，即使没有结果集数据（`rs.next( )` 为 false），此代码也将检查 SQLWarning。 以下示例为正确写入的程序输出，用于检测并报告数字错误。 错误是除数为零：

```
-- Warning received --

Message: Divide by zero occurred.
SQLState: 01012
ErrorCode: 3607
```

检索特定于 SAP jConnect 的错误信息

SAP jConnect 提供 `EedInfo` 接口来指定获取特定于 SAP jConnect 的错误信息的方法。

`EedInfo` 接口在 `SybSQLException` 和 `SybSQLWarning` 中实现，它们是 `SQLException` 和 `SQLWarning` 的扩展类。

`SybSQLException` 和 `SybSQLWarning` 包含以下方法：

- `public ResultSet getEedParams`，返回包含附带了错误消息的所有参数值的单行结果集。
- `public int getStatus`，如果消息中包含参数值，则返回 1；如果不包含参数值，则返回 0。

- `public int getLineNumber`，返回引发了错误消息的存储过程或查询的行号。
- `public String getProcedureName`，返回引发了错误消息的过程的名称。
- `public String getServerName`，返回生成消息的服务器的名称。
- `public int getSeverity`，返回错误消息的严重性。
- `public int getState`，返回有关服务器中错误消息的内部源信息，仅供 SAP 技术支持使用。
- `public int getTranState`，返回以下事务状态之一：
  - 0 - 连接当前处于扩展事务中。
  - 1 - 前一事务已成功提交。
  - 3 - 前一事务已中止。

有些错误消息可能是 `SQLException` 或 `SQLWarning` 消息，但不是 `SybSQLException` 或 `SybSQLWarning` 消息。 应用程序应先检查其正在处理的异常类型，然后下转到 `SybSQLException` 或 `SybSQLWarning`。

自定义错误消息处理

使用 `SybMessageHandler` 接口来自定义 SAP jConnect 处理服务器生成的错误消息的方式。

通过在自己的类中实现 `SybMessageHandler` 来处理错误消息有如下好处：

- 通用错误处理 – 错误处理逻辑可放在错误消息处理程序中，而不需要在整个应用程序中重复。
- 通用错误记录 – 错误消息处理程序可包含处理所有错误记录的逻辑。
- 根据应用程序要求重新映射错误消息严重性 - 错误消息处理程序可以包含识别特定错误消息的逻辑，并根据应用程序的要求而不是根据服务器的严重性级别来降低或提高错误的严重级。 例如，在进行删除旧行的清除操作期间，可能想降低消息“行不存在”的严重级。 然而，在其它环境中又可能需要提高严重级。

注意： 实现 `SybMessageHandler` 接口的错误消息处理程序仅接收服务器生成的消息；它们不处理由 SAP jConnect 生成的消息。

SAP jConnect 接收到错误消息时，它会检查是否已注册 `SybMessageHandler` 类来处理该消息。如果已注册，SAP jConnect 将调用 `messageHandler` 方法，该方法接受 SQL 异常作为其参数。 然后，SAP jConnect 根据 `messageHandler` 返回的值处理消息。 错误消息处理程序可以：

- 依原样返回 SQL 异常。
- 返回空值。 因此，SAP jConnect 将忽略此消息。
- 根据 SQL 异常创建 SQL 警告，并将其返回。 警告结果被添加到警告消息链中。
- 如果初始消息是 SQL 警告，`messageHandler` 可将此 SQL 警告评估为紧急，并在控制权返回到 SAP jConnect 后创建并返回 SQL 异常。

安装错误消息处理程序

通过从 SybDriver、SybConnection 或 SybStatement 调用 setMessageHandler 方法可安装实现 SybMessageHandler 的错误消息处理程序。

如果从 SybDriver 安装错误消息处理程序，则所有后续 SybConnection 对象都会继承此处理程序。 如果从 SybConnection 对象安装错误消息处理程序，则由该 SybConnection 对象创建的所有 SybStatement 对象都会继承此处理程序。

此继承关系仅从安装了错误消息处理程序对象后才起作用。 例如，如果创建一个名为“myConnection”的 SybConnection 对象，然后调用 SybDriver.setMessageHandler 来安装错误消息处理程序对象，则“myConnection”无法使用此对象。

若要返回当前错误消息处理程序对象，请使用 getMessageHandler。

错误消息处理程序示例

SAP jConnect 中错误消息处理程序的示例。

```
import java.io.*;
import java.sql.*;
import com.sybase.jdbcx.SybMessageHandler;
import com.sybase.jdbcx.SybConnection;
import com.sybase.jdbcx.SybStatement;
import java.util.*;

public class MyApp
{
  static SybConnection conn = null;
  static SybStatement stmt = null
  static ResultSet rs = null;
  static String user = "guest";
  static String password = "sybase";
  static String server = "jdbc:sybase:Tds:192.138.151.39:4444";
  static final int AVOID_SQLE = 20001;

  public MyApp()
  {
     try
     {
             Class.forName("com.sybase.jdbc4.jdbc.SybDriver").newIn
stance();
              Properties props = new Properties();
              props.put("user", user);
              props.put("password", password);
              conn = (SybConnection)
              DriverManager.getConnection(server, props);
              conn.setMessageHandler(new NoResultSetHandler());
              stmt =(SybStatement) conn.createStatement();
              stmt.executeUpdate("raiserror 20001 'your error'");
```

```
            for (SQLWarning sqw = _stmt.getWarnings();
            sqw != null;
            sqw = sqw.getNextWarning());
            {
               if (sqw.getErrorCode() == AVOID_SQLE);
               {
                System.out.println("Error" + sqw.getErrorCode()+
                " was found in the Statement's warning list.");
                break;
               }
            }
            stmt.close();
            conn.close();
  }
  catch(Exception e)
  {
    System.out.println(e.getMessage());
    e.printStackTrace();
  }
}

class NoResultSetHandler implements SybMessageHandler
{
  public SQLException messageHandler(SQLException sqe)
  {
    int code = sqe.getErrorCode();
    if (code == AVOID_SQLE)
    {
      System.out.println("User " + _user + " downgrading " +
        AVOID_SQLE + " to a warning");
      sqe = new SQLWarning(sqe.getMessage(),
        sqe.getSQLState(),sqe.getErrorCode());
    }
    return sqe;
  }
}

public static void main(String args[])
{
  new MyApp();
}
```

口令加密

缺省情况下，SAP jConnect for JDBC 通过网络向 SAP Adaptive Server 发送明文口令以进行验证。

但是，SAP jConnect 也支持对称和非对称口令加密，并可以在通过网络发送口令之前对口令进行加密。

对称加密机制使用同一密钥来加密和解密口令，而非对称加密机制使用一个密钥（公用密钥）加密口令，使用另一个密钥（私有密钥）解密口令。 由于私有密钥不在网络上共享，因此认为非对称加密比对称加密更安全。 启用口令加密后，如果服务器支持非对称加密，则使用此格式，而不使用对称加密。

注意： 若要使用非对称口令加密功能，必须有一个支持口令加密的服务器，例如 SAP Adaptive Server 15.0.2 及更高版本。

启用口令加密

ENCRYPT_PASSWORD 连接属性指定是否以加密格式传输口令。

此属性用于启用非对称密钥加密。 启用口令加密后，如果服务器支持非对称密钥加密，则使用此格式，而不使用对称密钥加密。

将 ENCRYPT_PASSWORD 连接属性设置为 true 将启用口令加密。 缺省值为 false。

注意： 如果服务器配置为要求客户端使用加密口令，则输入纯文本口令会导致用户登录失败。

使用明文口令重试登录

如果 ENCRYPT_PASSWORD 属性设置为 true，但服务器不支持口令加密，服务器登录将失败。

若要对不支持口令加密的服务器使用明文口令，请将 RETRY_WITH_NO_ENCRYPTION 连接属性设置为 true。

当 ENCRYPT_PASSWORD 和 RETRY_WITH_NO_ENCRYPTION 属性都设置为 true 时，jConnect 会首先使用加密口令登录。 如果登录失败，jConnect 将使用明文口令登录。

设置 Java Cryptography Extension (JCE) 提供程序

非对称口令加密机制使用 RSA 加密算法加密要传输的口令。

若要执行此 RSA 加密，请为 JRE 配置合适的 Java Cryptography Extension (JCE) 提供程序。 配置的 JCE 提供程序应能够支持“RSA/ECB/OAEPWithSHA1AndMGF1Padding”或“RSA/NONE/OAEPWithSHA1AndMGF1Padding”转换。 SAP jConnect 将 FIPS 140-2 认证的 JCE 提供程序与产品一起提供并在缺省情况下使用。

可以使用 JCE_PROVIDER_CLASS 连接属性指定备用 JCE 提供程序。 您可以从大量商业和开放源代码 JCE 提供程序中进行选择。 例如，“Bouncy Castle Crypto APIs for Java”就是一种广泛使用的开放源代码 Java JCE 提供程序。 如果选择不指定 JCE_PROVIDER_CLASS 属性，SAP jConnect 将使用 FIPS 140-2 认证的捆绑的提供程序。

使用缺省 JCE 提供程序执行口令加密

未指定 JCE_PROVIDER_CLASS 时，SAP jConnect 使用缺省捆绑的 FIPS 140-2 认证的 JCE 提供程序执行 RSA 口令加密。

指定自定义 JCE 提供程序

在 SAP jConnect 中指定自定义 JCE 提供程序。

1. 将 JCE_PROVIDER_CLASS 属性设置为要使用的提供程序的完全限定类名。

例如，要使用 Bouncy Castle JCE：

```
String url = "jdbc:sybase:Tds:myserver:3697";
Properties props = new Properties();
props.put("ENCRYPT_PASSWORD ", “true”);
props.put("JCE_PROVIDER_CLASS",
"org.bouncycastle.jce.provider.BouncyCastleProvider");

/* Set up additional connnection properties as needed */
props.put("user", "xyz");
props.put("password", "123");

/* get  the connection */
Connection con = DriverManager.getConnection(url, props);
```

2. 配置 JCE 提供程序。

   下列方法任选其一：

   - 将 JCE 提供程序 `jar` 文件复制到 JRE 标准扩展目录中：
     - 对于 UNIX 平台： `${JAVA_HOME}/jre/lib/ext`
     - 对于 Windows： `%JAVA_HOME%\jre\lib\ext`
   - 如果无法将 JCE `jar` 文件复制到适当的目录中，请参见《JCE 参考指南》(JCE Reference Guide) 以了解设置外部 JCE 提供程序的相关说明。

   如果没有配置任何其它 JCE 提供程序，或者配置的提供程序不支持所需转换并且启用了口令加密，则连接将失败。

将 Java 对象作为列数据存储在表中

数据库产品允许直接将 Java 对象作为列数据存储到数据库中。

在这样的数据库中，Java 类被当作数据类型，可以声明以 Java 类作为其数据类型的列。

通过实现定义于 `PreparedStatement` 接口中的 `setObject` 方法以及定义于 `CallableStatement` 和 `ResultSet` 接口中的 `getObject` 方法，SAP jConnect 支持将 Java 对象存储到数据库中。 这样，便可将 SAP jConnect 与使用本地 JDBC 类和方法的应用程序结合使用，从而直接将 Java 对象作为列数据进行存储和检索。

注意： 要使用 `getObject` 和 `setObject`，请将 SAP jConnect 版本设置为 `com.sybase.jdbcx.SybDriver.VERSION_4` 或更高版本。

SAP Adaptive Server 12.0 及更高版本和 SAP SQL Anywhere 6.0.x 及更高版本可在表中存储 Java 对象，但有一些限制。 请参见《SAP jConnect for JDBC 发行公告》。

另请参见

- 将 Java 对象作为列数据存储的前提条件 （第 80 页）
- 接收来自数据库的 Java 对象 （第 81 页）
- JCONNECT_VERSION 连接属性 （第 4 页）
- 将 Java 对象发送到数据库 （第 80 页）

将 Java 对象作为列数据存储的前提条件

将属于用户定义的 Java 类的 Java 对象存储在列中。

- 该类必须实现 `java.io.Serializable` 接口。这是因为 SAP jConnect 使用本地 Java 序列化和反序列化将对象发送到数据库和接收从数据库返回的对象。
- 类定义必须安装在目标数据库中，或者您必须使用 `DynamicClassLoader` (DCL) 直接从 SAP SQL Anywhere 或 SAP Adaptive Server 服务器装载类，并在使用类时视其位于本地 CLASSPATH 中。
- 客户端系统必须在 `.class` 文件中包含类定义，该文件可通过本地 CLASSPATH 环境变量访问。

另请参见

- 动态类装载 （第 83 页）

将 Java 对象发送到数据库

使用 `setObject` 方法将 Java 对象发送到数据库。

要将用户定义的类的实例作为列数据发送，请使用 `setObject` 方法之一（按照 `PreparedStatement` 接口中指定的内容）：

```
void setObject(int parameterIndex, Object x, int targetSqlType,
   int scale) throws SQLException;
```

```
void setObject(int parameterIndex, Object x, int targetSqlType)
   throws SQLException;
```

```
void setObject(int parameterIndex, Object x) throws SQLException;
```

在 jConnect 中，若要发送 Java 对象，可使用 `java.sql.Types.JAVA_OBJECT` 目标 `sql.Type`，或者使用 `java.sql.Types.OTHER`。

以下示例定义一个 `Address` 类，显示 `Friends` 表（该表包含数据类型为 `Address` 类的 `Address` 列）的定义，并向该表中插入一行。

```
public class Address implements Serializable
```

```
{
```

```
  public String   streetNumber;
   public String   street;
   public String   apartmentNumber;
   public String   city;
   public int   zipCode;
```

```
  //Methods
```

```
  ...
```

```
}
```

```
/* This code assumes a table with the following structure
**  Create table Friends:
```

```
**  (firstname varchar(30)  ,
**   lastname varchar(30),
**  address Address,
**  phone varchar(15))
```

```
*/
```

```
// Connect to the database containing the Friends table.
Connection conn =
   DriverManager.getConnection("jdbc:sybase:Tds:localhost:5000",
     "username", "password");

// Create a Prepared Statement object with an insert statement
//for updating the Friends table.
PreparedStatement ps = conn.prepareStatement("INSERT INTO
   Friends values (?,?,?,?)");

// Now, set the values in the prepared statement object, ps.
// set firstname to "Joan."
ps.setString(1, "Joan");

// Set last name to "Smith."
ps.setString(2, "Smith");

// Assuming that we already have "Joan_address" as an instance
// of Address, use setObject(int parameterIndex, Object x) to
// set the address column to "Joan_address."
ps.setObject(3, Joan_address);

// Set the phone column to Joan's phone number.
ps.setString(4, "123-456-7890");

// Perform the insert.
ps.executeUpdate();
```

接收来自数据库的 Java 对象

客户端 JDBC 应用程序可将接收到的来自数据库的 Java 对象放到结果集中，或作为从存储过程返回的输出参数的值。

如果结果集包含作为列数据的 Java 对象，请在 `ResultSet` 接口中使用 `getObject` 方法之一检索该对象：

```
Object getObject(int columnIndex) throws SQLException;
```

```
Object getObject(String columnName) throws SQLException;
```

如果从存储过程返回的输出参数包含 Java 对象，请在 `CallableStatement` 接口中使用此 `getObject` 方法检索该对象：

```
Object getObject(int parameterIndex) throws SQLException;
```

以下示例说明如何使用 `ResultSet.getObject(int parameterIndex)` 将在结果集中接收的对象指派给类变量。 本示例使用 `Address` 类和 `Friends` 表，并提供了一个可在信封上打印姓名与地址的简单应用程序。

```
/*
 ** This application takes a first and last name, gets the
 ** specified person's address from the Friends table in the
 ** database, and addresses an envelope using the name and
 ** retrieved address.
 */
 public class Envelope
 {
   Connection conn = null;
   String firstName = null;
   String lastName = null;
   String street = null;
   String city = null;
   String zip = null;

   public static void main(String[] args)
   {
     if (args.length < 2)
     {
     System.out.println("Usage: Envelope <firstName>
       <lastName>");
     System.exit(1);
     }
     // create a 4" x 10" envelope
     Envelope e = new Envelope(4, 10);
     try
     {
       // connect to the database with the Friends table.
       conn = DriverManager.getConnection(
         "jdbc:sybase:Tds:localhost:5000", "username",
           "password");
       // look up the address of the specified person
       firstName = args[0];
       lastName = args[1];
       PreparedStatement ps = conn.prepareStatement(
         "SELECT address FROM friends WHERE " +
           "firstname = ? AND lastname = ?");
       ps.setString(1, firstName);
       ps.setString(2, lastName);
       ResultSet rs = ps.executeQuery();
       if (rs.next())
       {
         Address a = (Address) rs.getObject(1);
         // set the destination address on the envelope
         e.setAddress(firstName, lastName, a);
       }
       conn.close();
     }
     catch (SQLException sqe)
     {
       sqe.printStackTrace();
       System.exit(2);
     }
     // if everything was successful, print the envelope
     e.print();
```

```
    }
    private void setAddress(String fname, String lname, Address a)
    {
      street = a.streetNumber + " " + a.street + " " +
        a.apartmentNumber;
      city = a.city;
      zip = "" + a.zipCode;
    }
    private void print()
    {
      // Print the name and address on the envelope.
      ...
    }
}
```

可在 SAP jConnect 安装目录下的 `sample2` 子目录中找到更详细的 `HandleObject.java` 示例。

动态类装载

SAP SQL Anywhere 和 SAP Adaptive Server 允许指定 Java 类。

- SQL 列的数据类型
- Transact-SQL 变量的数据类型
- SQL 列的缺省值

SAP jConnect 6.05 和更高版本通过实现 `DynamicClassLoader` (DCL) 来直接从 SAP SQL Anywhere 或 SAP Adaptive Server 服务器装载类，并在使用类时视其位于本地 CLASSPATH 中。

在 SAP jConnect 6.0 和更低版本中，只有出现在 SAP jConnect CLASSPATH 中的类才是可访问的，也就是说，如果 SAP jConnect 应用程序试图访问不在本地 CLASSPATH 中的类的实例，将会产生 `java.lang.ClassNotFound` 异常。

会继承父类中的所有安全性功能。 Java 2 中实现的装载程序委托模型仍然存在 - SAP jConnect 首先尝试从 CLASSPATH 中装载请求的类；如果失败，SAP jConnect 会尝试 `DynamicClassLoader`。

有关使用 Java 和 SAP Adaptive Server 的详细信息，请参见《Adaptive Server 中的 Java》。

使用 DynamicClassLoader

使用 `CLASS_LOADER` 连接属性提供一个方便的机制，以在多个连接中共享一个类装载程序。

1. 创建并配置类装载程序。

   SAP jConnect 应用程序的代码应该类似于以下形式：

   ```
   Properties props = new Properties();// URL of the server where the
   classes live.
   String classesUrl = "jdbc:sybase:Tds:myase:1200"; // Connection
   properties for connecting to above server.
   ```

```
props.put("user", "grinch");
props.put("password", "meanone");
... // Ask the SybDriver for a new class loader.
DynamicClassLoader loader = driver.getClassLoader(classesUrl,
props);
```

2. 使用 CLASS_LOADER 连接属性，使得新的类装载程序可用于执行查询的语句。

   创建类装载程序后，即可将其传递给后续连接，如下所示（接着第 1 步中的代码示例）：

```
// Stash the class loader so that other connection(s)
// can know about it.
props.put("CLASS_LOADER", loader);// Additional connection
properties
props.put("user", "joeuser");
props.put("password", "joespassword");// URL of the server we now
want to connect to.
String url = "jdbc:sybase:Tds:jdbc.sybase.com:4446";// Make a
connection and go.
Connection conn = DriverManager.getConnection(url, props);
```

   假定 Java 类定义为：

```
class Addr {
      String street;
      String city;
      String state;
}
```

   假定 SQL 表定义为：

```
create table employee (char(100) name, int empid, Addr address)
```

3. 如果客户端应用程序 CLASSPATH 中缺少 Addr 类，请使用以下客户端代码：

```
Statement stmnt = conn.createStatement();

// Retrieve some rows from the table that has a Java class
// as one of its fields.

ResultSet rs = stmnt.executeQuery(

      "select * from employee where empid = ’19’ ");

if (rs.next() {

      // Even though the class is not in our class path,

      // we should be able to access its instance.

      Object obj = rs.getObject("address");

     // The class has been loaded from the server, so let's take a
look.

      Class c = obj.getClass();

    // Some Java Reflection can be done here to access the fields
of obj.
```

```
        ...
}
```

应确保在多个连接中共享一个类装载程序不会导致类冲突。例如，假定 `org.foo.Bar` 类的两个不同且不兼容的实例存在于两个不同的数据库中，当使用同一装载程序访问这两个类时，就可能引发问题。在检查来自第一个连接的结果集时装载第一个类。当检查来自第二个连接的结果集时，已经装载了这个类。因此，将不会装载第二个类，而 SAP jConnect 无法直接检测出此情况。

不过，Java 具有一种内置机制，可确保类的版本与反序列化对象的版本信息匹配。Java 至少会检测并报告上述情况。

类及其实例不需要驻留在同一数据库或服务器中，但装载程序和后续连接没有理由不引用同一数据库或服务器。

反序列化

该序列化对象是一个驻留在服务器但不存在于 CLASSPATH 中的类的实例。

本示例说明了如何将本地文件中的对象反序列化：

`SybResultSet.getObject()` 使用 `DynamicObjectInputStream`（装载来自 `DynamicClassLoader` 的类定义的 `ObjectInputStream` 的子类），而不是缺省系统（“boot”）类装载程序。

```
// Make a stream on the file containing the
//serialized object.
FileInputStream fileStream = new FileInputStream("serFile");
// Make a "deserializer" on it. Notice that, apart
//from the additional parameter, this is the same
//as ObjectInputStreamDynamicObjectInputStream
stream = new DynamicObjectInputStream(fileStream, loader);
// As the object is deserialized, its class is
//retrieved through the loader from our server.
Object obj = stream.readObject();stream.close();
```

预装载 .jar 文件

SAP jConnect 6.05 或更高版本具有一个名为 `PRELOAD_JARS` 的连接属性。将 `.jar` 文件定义为以逗号分隔的 `.jar` 文件名的列表时，这些文件会被全部装载。

在这种情况下，“JAR”会引用服务器使用的“保留的 JARname”。它是在安装 Java 程序中指定的 `.jar` 文件名，例如：

```
install java new jar 'myJarName' from file '/tmp/mystuff.jar'
```

如果设置 PRELOAD_JARS，则 .jar 文件会与类装载程序关联，因此不需要为每个连接预装载这些文件。 您只需为一个连接指定 PRELOAD_JARS。 以后尝试预装载相同的 .jar 文件可能会影响性能，因为不必要地从服务器检索 .jar 文件数据。

注意： SAP SQL Anywhere 不能将 .jar 文件作为一个实体返回，因此 SAP jConnect 会依次迭代检索每个类。 不过，SAP Adaptive Server 可检索整个 .jar 文件，并装载其包含的所有类。

附加动态类装载功能

附加功能包括能够在期望进行一系列类装载时使装载程序的数据库连接保持活动状态，并能按照类名称显式装载单个类。

您可使用继承自 java.lang.ClassLoader 的公共方法。 java.lang.Class 中处理装载类的方法也是可用的；不过，使用这些方法时要小心，因为其中某些方法会对使用哪些类装载程序进行假定。 尤其应使用 Class.forName 的三参数版本，否则会使用系统（“boot”）类装载程序。

DynamicClassLoader 中有许多公共方法。 有关详细信息，请参见 JDBC_HOME/docs/en/javadocs 中的 Javadoc。

另请参见

- 错误消息 （第 73 页）

JDBC 4.0 规范支持

SAP jConnect 支持的一些 JDBC 4.0 规范。

- 连接管理
- 自动 SQL 驱动程序装载
- 数据库元数据
- 国家字符集转换
- 包装模式
- 标量函数 CHAR_LENGTH、CHARACTER_LENGTH、CURRENT_DATE、CURRENT_TIME、CURRENT_TIMESTAMP、EXTRACT 和 OCTET_LENGTH、POSITION

有关 JDBC 4.0 规范的信息，请参见 Oracle Technology Network for Java。

JDBC 3.0 规范支持

SAP jConnect 16.0 中所支持的 JDBC 3.0 功能。

保存点支持

您可使用 `Savepoint` 接口，其中包含设置、释放事务或将事务回退到指定保存点的方法。

- **在事务中使用保存点** – 在 JDBC 3.0 中，`Savepoint` 接口允许将一个事务分成几个逻辑断点，以此对回退的事务量加以控制。
- **设置和回退到某个保存点** – JDBC 3.0 API 包含方法 `Connection.setSavepoint`，该方法在当前事务内设置一个保存点，并返回一个 `Savepoint` 对象。 重载 `Connection.rollback` 方法来采用 `Savepoint` 对象参数。
- **释放保存点** – `Connection.releaseSavepoint` 方法以 `Savepoint` 对象作为参数并将其从当前事务中删除。

  释放 `Savepoint` 后，如果尝试在回退操作中引用它，则会发生 SQLException。 当事务被提交或整个事务回退时，在事务中创建的任何保存点都自动释放并变成无效保存点。 如果将事务回退到某个保存点，它会自动释放任何其它的在该保存点之后创建的保存点并使之失效。

  注意： 可以使用 `DatabaseMetaData.supportsSavepoints` 方法确定 JDBC API 实现是否支持保存点。

参数元数据检索

JDBC 3.0 `ParameterMetaData` 接口描述预准备语句参数的数量、类型和属性，并支持最新的 `DatabaseMetaData` 方法。

自动生成键检索

JDBC 3.0 能够满足从列获取自动生成键或自动递增键的常见需求。

要检索自动生成键，请将常量 `Statement.RETURN_GENERATED_KEYS` 作为 `Statement.execute()` 方法的第二个参数进行传递。

执行该语句后，调用 `Statement.getGeneratedKeys()` 检索生成的键。 结果集将包含检索到的每个键对应的行。

注意： SAP Adaptive Server 无法返回生成键的结果集。 如果执行一批 **insert** 命令，则调用 `Statement.getGeneratedKeys()` 将只返回最后一个生成键的值。

有关检索自动生成键的详细信息（包括示例代码），请在 Oracle Java 网站上搜索“检索自动生成的键”。

多个打开的 ResultSet 对象

JDBC 3.0 包含 `getMoreResults(int)`，它采用的参数指定 `Statement` 对象所返回的 `ResultSet` 对象是否应在返回任何后续 `ResultSet` 对象前关闭。

JDBC 3.0 规范允许 `Statement` 接口支持多个打开的 `ResultSets`，从而取消了 JDBC 2.0 规范限制，即，返回多个结果的语句在任何指定时间只能打开一个 `ResultSet`。为支持多个打开的结果，`Statement` 接口增加了方法 `getMoreResults()` 的一个重载版本。`getMoreResults(int)` 方法采用整数标志，当调用 `getResultSet()` 方法时，该标志指定以前打开的 `ResultSets` 的行为。该接口定义的标志如下所示：

- CLOSE_ALL_RESULTS - 当调用 `getMoreResults()` 时，关闭之前打开的所有 `ResultSet` 对象。
- CLOSE_CURRENT_RESULT – 当调用 `getMoreResults()` 时，关闭当前 `ResultSet` 对象。
- KEEP_CURRENT_RESULT - 当调用 `getMoreResults()` 时，不关闭当前 `ResultSet` 对象。

按照名称将参数传递到 CallableStatement 对象

允许用字符串标识要为 `CallableStatement` 对象设置的参数。

可使用 `CallableStatement` 接口按参数名称而非参数索引指定参数。当过程中包含很多具有缺省值的参数时，这种方法非常有用。可以使用命名参数来仅指定没有缺省值的值。

可保持游标支持

可保持游标（或结果）在包含该游标的事务被提交后不能自动关闭。必须指定 `ResultSet` 对象的可保持性。

JDBC 3.0 支持指定游标的可保持性。在使用 `createStatement()`、`prepareStatement()` 或 `prepareCall()` 方法准备语句时，必须指定 `ResultSet` 的可保持性。可保持性可以为下列常量之一：

- HOLD_CURSORS_OVER_COMMIT – `ResultSet` 对象（游标）不关闭；当隐式或显式执行 **commit** 操作时，它们保持打开状态。
- CLOSE_CURSORS_AT_COMMIT – `ResultSet` 对象（游标）在隐式或显式执行 **commit** 操作时关闭。

如果在提交事务时关闭游标，通常能够提高性能。SAP 建议除非在完成事务后需要游标，否则最好在执行 **commit** 操作时关闭游标。因为规范没有定义 `ResultSet` 的缺省可保持性，所以其行为将取决于实现。

JDBC 2.0 选件工具包扩展支持

《JDBC 2.0 选件工具包》(JDBC 2.0 Optional Package)（旧称《JDBC 2.0 标准扩展 API》(JDBC 2.0 Standard Extension API)）中定义了多个 JDBC 2.0 驱动程序可实现的功能。

SAP jConnect 6.05 及更高版本已实现其中多项选件工具包扩展功能：

- 用于命名约定的 JNDI - 可用于 SAP jConnect 支持的任何 SAP DBMS
- 连接池 - 可用于 SAP jConnect 支持的任何 SAP DBMS
- 分布式事务管理支持 - 仅用于 SAP Adaptive Server

SAP 建议使用 JNDI 1.2，它可与 Java 1.1.6 及更高版本兼容。

用于命名数据库的 JNDI

查看用于命名数据库的 JNDI 的相关信息。

参考

《JDBC 2.0 选件工具包》(JDBC 2.0 Optional Package)（旧称《JDBC 2.0 标准扩展 API》(JDBC 2.0 Standard Extension API)）。

相关接口

相关接口为 JDBC 客户端提供了按标准方法获取数据库连接的替代方法。

- `javax.sql.DataSource`
- `javax.naming.Referenceable`
- `javax.naming.spi.ObjectFactory`

客户端不再通过调用 `Class.forName`（“com.sybase.jdbc4.jdbc.SybDriver”）并将 JDBC URL 传递到 DriverManager 的 `getConnection()` 方法，而是使用逻辑名访问 JNDI 命名服务器来检索 `javax.sql.DataSource` 对象。 此对象负责装载驱动程序，并与它代表的物理数据库建立连接。 客户端代码更简单并且是可重用的，因为特定于供应商的信息已放入 `DataSource` 对象中。

`DataSource` 对象的 SAP 实现是 `com.sybase.jdbcx.SybDataSource`（有关详细信息，请参见 Javadoc）。 此实现通过使用 JavaBean 组件的设计模式支持以下标准属性：

- `databaseName`
- `dataSourceName`
- `description`
- `networkProtocol`
- `password`
- `portNumber`
- `serverName`
- `user`

注意： roleName 不受支持。

SAP jConnect 提供 javax.naming.spi.ObjectFactory 接口的实现，因此 DataSource 对象可根据命名服务器条目的属性构建。 给定 javax.naming.Reference，或 javax.naming.Name 和 javax.naming.DirContext 时，此 factory 可构建 com.sybase.jdbcx.SybDataSource 对象。 要使用此 factory，请将 java.naming.object.factory 系统属性设置为包括 com.sybase.jdbc4.SybObjectFactory。

用法

DataSource 可以在不同的应用程序中以不同的方式使用。

所有选项均随附一些代码示例。有关详细信息，请参见《JDBC 2.0 选件工具包》(JDBC 2.0 Optional Package)（旧称《JDBC 2.0 标准扩展 API》(JDBC 2.0 Standard Extension API)）以及 Oracle Java 网站上的 JNDI 文档。

管理员进行的配置：LDAP

SAP jConnect 自 4.0 版本开始已经支持 LDAP 连接。 因此，建议的方法（不要求自定义软件）是使用 LDAP 数据交换格式 (LDIF) 将 DataSources 配置为 LDAP 条目。

例如：

```
dn:servername:myASE, o=MyCompany, c=US

1.3.6.1.4.1.897.4.2.5:TCP#1# mymachine 4000

1.3.6.1.4.1.897.4.2.10:PACKETSIZE=1024&user=me&password=secret

1.3.6.1.4.1.897.4.2.11:userdb
```

通过客户端访问

JDBC 客户端应用程序可通过访问服务器名称获取对 DataSource 对象的引用，而不是通过访问 DriverManager 并提供 JDBC URL。

这是典型的 JDBC 客户端应用程序。 获取连接后，客户端代码就和所有其它 JDBC 客户端代码相同了。 此代码为通用代码，且设置对象 factory 属性（可在设置环境时进行）时仅引用 SAP jConnect。

SAP jConnect 安装包含了示例程序 sample2/SimpleDataSource.java 以说明 DataSource 的使用。 此示例仅用于参考，也就是说，除非适当地配置环境并编辑此示例，否则不能运行它。 SimpleDataSource.java 包含以下重要代码：

```
import javax.naming.*;

import javax.sql.*;

import java.sql.*;

// set necessary JNDI properties for your environment (same as above)
```

```
Properties jndiProps = new Properties();

// used by JNDI to build the SybDataSource

jndiProps.put(Context.OBJECT_FACTORIES,

    "com.sybase.jdbc4.jdbc.SybObjectFactory");

// nameserver that JNDI should talk to

jndiProps.put(Context.PROVIDER_URL, "ldap: some_ldap_server:238/" +
"o=MyCompany,c=Us");

// used by JNDI to establish the naming context

jndiProps.put(Context.INITIAL_CONTEXT_FACTORY,

    "com.sun.jndi.ldap.LdapCtxFactory");

// obtain a connection to your name server

Context ctx = new InitialContext(jndiProps);

DataSource ds = (DataSource) ctx.lookup("servername=myASE");

// obtains a connection to the server as configured earlier.

// in this case, the default username and password will be used

Connection conn = ds.getConnection();

// do standard JDBC methods

...
```

如果已经在虚拟机中定义属性，则不必将 `Properties` 显式传递给 `InitialContext` 构造函数，也就是说，要么在将 Java 设置为浏览器属性的一部分时传递，要么使用下面的方法传递：

```
java -
Djava.naming.object.factory=com.sybase.jdbc4.jdbc.SybObjectFactory
```

有关设置环境属性的详细信息，请参见 Java VM 文档。

编程配置

编程配置的目的是定义一个数据源，然后将其以一个逻辑名部署到命名服务器。

如果需要重新配置服务器（例如，移到其它计算机、端口等），管理员将运行此配置实用程序（概述如下），并将此逻辑名重新指派给新的数据源配置。 此阶段通常由负责为公司执行数据库系统管理或应用程序集成的人员来完成。 因此，客户端代码不会更改，因为它只认识此逻辑名。

```
import javax.sql.*;

import com.sybase.jdbcx.*;

.....

// create a SybDataSource, and configure it
```

```
SybDataSource ds = new com.sybase.jdbc4.jdbc.SybDataSource();

ds.setUser("my_username");

ds.setPassword("my_password");

ds.setDatabaseName("my_favorite_db");

ds.setServerName("db_machine");

ds.setPortNumber(4000);

ds.setDescription("This DataSource represents the Adaptive Server
   Enterprise server running on db_machine at port 2638.  The default
    username and password have been set to 'me' and 'mine'
respectively.
   Upon connection, the user will access the my_favorite_db database
on
    this server.");
Properties props = newProperties()
props.put("REPEAT_READ","false");
props.put("REQUEST_HA_SESSION","true");
ds.setConnectionProperties(props);

// store the DataSource object. Typically this is
// done by setting JNDI properties specific to the
// type of JNDI service provider you are using.
// Then, initialize the context and bind the object.

Context ctx = new InitialContext();

ctx.bind("java:comp/env/jdbc/myASE", ds);
```

设置 DataSource 后，需要决定信息的存储位置和存储方式。 SybDataSource 提供了 java.io.Serializable 和 javax.naming.Referenceable 来帮助您作出决定，但仍需管理员根据使用的 JNDI 服务提供程序来决定如何存储数据。

通过客户端检索 Datasource 对象

客户端通过采用部署 DataSource 的方法来设置 DataSource 对象的 JNDI 属性，从而检索该对象。

客户端需要一个可转换该对象的可用对象 factory，因为它会被存储（例如，序列化）到 Java 对象中。

```
Context ctx = new InitialContext();
DataSource ds = (DataSource) ctx.lookup("java:comp/env/jdbc/myASE");
Connection conn = ds.getConnection();
```

连接池
查看 SAP jConnect 中的连接池说明。

参考
查看《JDBC 2.0 选件工具包》(JDBC 2.0 Optional Package)（旧称《JDBC 2.0 标准扩展 API》(JDBC 2.0 Standard Extension API)）。

相关接口
查看 JDBC 中的相关接口。

- `javax.sql.ConnectionPoolDataSource`
- `javax.sql.PooledConnection`

概述
传统数据库应用程序可与用于应用程序的每次会话的数据库创建一个连接。 不过，使用此应用程序时，基于 Web 的数据库应用程序可能需要多次打开和关闭新连接。

处理基于 Web 的数据库连接的一个有效方法是使用连接池，它可维护打开的数据库连接并管理在不同用户请求间共享的连接，从而保证性能并减少空闲连接的数目。 对于每个连接请求，连接池首先确定池中是否有空闲连接。 如果有，连接池会返回空闲连接，而不是与数据库建立新连接。

提供 `com.sybase.jdbc4.jdbc.ConnectionPoolDataSource` 类来与连接池实现交互。 使用 `ConnectionPoolDataSource` 时，池实现会监听 `PooledConnection`。 当您关闭连接或有错误破坏了连接时，会通知该实现。 此时，池实现会决定如何处理 `PooledConnection`。

如果没有连接池，事务会：

1. 创建数据库连接。
2. 向数据库发送查询。
3. 获得结果集。
4. 显示结果集。
5. 破坏连接。

有连接池时，序列大致如下：

1. 事务确定连接“池”中是否存在未用连接。
2. 如果有，则使用此连接；否则创建新连接。
3. 向数据库发送查询。
4. 获得结果集。
5. 显示结果集。
6. 将连接返回池。 用户仍然调用 `close()`，但连接保持打开状态，且池会得到 **close** 请求的通知。

与每次客户端需要建立与数据库的连接时都创建一个新的连接相比，重新使用连接的开销要少。

为使第三方能够实现连接池，jConnect 实现让 ConnectionPoolDataSource 接口生成 PooledConnections，其方式与 DataSource 接口生成 Connections 的方式类似。池实现使用 ConnectionPoolDataSource 的 getPooledConnection( ) 方法创建真正的数据库连接。 然后，池实现将自己注册为针对 PooledConnection 的监听器。 这样一来，当客户端请求连接时，池实现就会在可用的 PooledConnection 上调用 getConnection()。 当客户端完成连接并调用 close 时，池实现就会通过 ConnectionEventListener 接口得到通知，告知连接空闲，可以重用。

如果客户端因某种原因损坏了数据库连接，池实现也会通过 ConnectionEventListener 接口收到通知，这样，池实现便可从池中删除该连接。有关详细信息，请参见《JDBC 2.0 选件工具包》(JDBC 2.0 Optional Package)（旧称《JDBC 2.0 标准扩展 API》(JDBC 2.0 Standard Extension API)）中的“Appendix B”(附录 B）。

管理员进行的配置：LDAP

通过向 LDIF 条目中输入附加行来配置 LDAP。

在本示例中，添加的代码行以粗体显示以供参考。

```
dn:servername=myASE, o=MyCompany, c=US
1.3.6.1.4.1.897.4.2.5:TCP#1# mymachine 4000
1.3.6.1.4.1.897.4.2.10:PACKETSIZE=1024&user=me&password=secret
1.3.6.1.4.1.897.4.2.11:userdb
1.3.6.1.4.1.897.4.2.18:ConnectionPoolDataSource
```

另请参见

- 用于命名数据库的 JNDI （第 89 页）
- 管理员进行的配置：LDAP （第 90 页）

通过中间层客户端进行访问

初始化三个属性（INITIAL_CONTEXT_FACTORY、PROVIDER_URL 和 OBJECT_FACTORIES），并检索 ConnectionPoolDataSource 对象。

有关更完整的代码示例，请参见 sample2/SimpleConnectionPool.java。 通过客户端访问和通过中间层客户端访问的根本区别是：

```
...
ConnectionPoolDatabase cpds = (ConnectionPoolDataSource)
    ctx.lookup("servername=myASE");
PooledConnection pconn = cpds.getPooledConnection();
```

分布式事务管理支持
为 SAP Adaptive Server 提供了用于执行分布式事务的标准 Java API。 此功能设计用于大型多层环境。

参考
《JDBC 2.0 选件工具包》(JDBC 2.0 Optional Package)（旧称《JDBC 2.0 标准扩展 API》(JDBC 2.0 Standard Extension API)）。

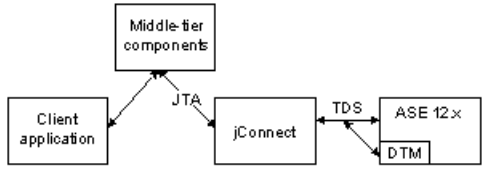
相关接口
查看 JDBC 中的相关接口。

- `javax.sql.XADataSource`
- `javax.sql.XAConnection`
- `javax.transaction.xa.XAResource`

后台和系统要求
使用 `dtm_tm_role` 启用分布式事务管理支持。

- SAP jConnect 必须直接与 SAP Adaptive Server 12.0 和更高版本中的资源管理器通信，并且安装必须支持分布式事务管理。
- 任何想参与分布式事务的用户都必须授予 `dtm_tm_role`，否则事务将失败。
- 要使用分布式事务，必须在 `/sp` 目录中安装存储过程。 请参见《SAP jConnect for JDBC 安装指南》中的“安装存储过程”。

图 2： 12.x 版本中的分布式事务管理支持



管理员进行的配置：LDAP
通过向 LDIF 条目中输入附加行来配置 LDAP。

在本示例中，添加的代码行显示为粗体。

```
dn:servername:myASE, o=MyCompany, c=US
1.3.6.1.4.1.897.4.2.5:TCP#1# mymachine 4000
1.3.6.1.4.1.897.4.2.10:PACKETSIZE=1024&user=me&password=secret
1.3.6.1.4.1.897.4.2.11:userdb
1.3.6.1.4.1.897.4.2.18:XADataSource
```

另请参见

- 用于命名数据库的 JNDI （第 89 页）
- 管理员进行的配置： LDAP （第 90 页）

通过中间层客户端进行访问

初始化三个属性（INITIAL_CONTEXT_FACTORY、PROVIDER_URL 和 OBJECT_FACTORIES），并检索 XADataSource 对象。

例如：

```
...
XADataSoruce xads = (XADatasource) ctx.lookup ("server=myASE");
XAConnection xaconn = xads.getXAConnection ();
```

或覆盖用户名和口令的缺省设置：

```
...
XADataSource xads = (XADatasource) ctx.lookup("servername=myASE");
XAConnection xaconn = xads.getXAConnection("my_username",
"my_password");
```

JDBC 标准的约束与说明

JDBC 的 SAP jConnect 实现会偏离 JDBC 标准。

另请参见

- 多线程的调整 （第 98 页）
- 不受支持的 JDBC 4.0 规范要求 （第 96 页）
- 使用 Connection.isClosed 和 IS_CLOSED_TEST （第 96 页）
- 带有未处理结果的 Statement.close （第 97 页）
- ResultSet.getCursorName （第 98 页）
- 执行存储过程 （第 98 页）

不受支持的 JDBC 4.0 规范要求

查看此版本中不受支持的 JDBC 4.0 语句。

- java.sql.RowID
- JDBC 4.0 中引入的 XML API

使用 Connection.isClosed 和 IS_CLOSED_TEST

SAP jConnect 为 isClosed 方法提供的缺省说明与 JDBC 4.0 规范中定义的行为不同。

当您调用 Connection.isClosed 时，SAP jConnect 会验证是否已在此连接上调用 Connection.close。 如果已调用 close，则 SAP jConnect 会为 isClosed 返回

true。 但是，如果未调用 Connection.close，则 SAP jConnect 将尝试在数据库上执行 **sp_mda**。 **sp_mda** 是 SAP jConnect 用户在数据库上使用 SAP jConnect 时必须安装的标准元数据的一部分。

根据 JDBC 4.0 规范的第 11.1 节：

仅能确保 Connection.isClosed 方法在调用 Connection.close 后返回 true。 通常情况下，不能调用 Connection.isClosed 来确定数据库连接是否有效。 典型客户端可通过捕获试图执行操作时引发的异常来确定连接无效。

调用 **sp_mda** 的目的是使 SAP jConnect 可执行已知（或至少可推断出）驻留在数据库服务器上的过程。 如果存储过程正常执行，则 SAP jConnect 会为 isClosed 返回 false，因为其已检验出数据库连接有效且正在运行。 但是，如果调用 **sp_mda** 引发 SQLException，则 SAP jConnect 会捕获该异常并为 isClosed 返回 true，因为连接似乎有问题。

如果想强制 SAP jConnect 更加严格地遵循 isClosed() 的标准 JDBC 行为，则可将 IS_CLOSED_TEST 连接属性设置为特殊值“INTERNAL”。 INTERNAL 设置表示只有在已调用 Connection.close 时，或者当 SAP jConnect 检测到禁用了连接的 IOException 时，SAP jConnect 才会为 isClosed 返回 true。

也可指定在调用 **isClosed** 时使用的除 sp_mda 外的其它查询。 例如，如果想让 SAP jConnect 在调用 **isClosed** 时尝试 select 1，可将 IS_CLOSED_TEST 连接属性设置为 **select 1**。

带有未处理结果的 Statement.close

关于首先调用 Statement.execute，然后对同一语句对象调用 close，而没有处理 Statement 返回的所有结果（更新计数和 ResultSet）时驱动程序应如何响应，JDBC 规范中未作明确说明。

例如，假定数据库上有一个执行七次行插入的存储过程。 某应用程序随后使用 Statement.execute 执行该存储过程。 在这种情况下，SAP 数据库向应用程序返回七个更新计数（每个插入的行分别计一次）。 根据常规 JDBC 应用程序逻辑，应该使用 getMoreResults、getResultSet 和 getUpdateCount 方法在循环中处理这些更新计数。 java.sql.* 软件包的 Javadoc 中的 Java SE 文档中对此有明确说明。

不过，应用程序程序员可能会在通览返回的所有更新计数前错误地调用 Statement.close。 在这种情况下，SAP jConnect 会向数据库发送 cancel，而这会造成无法预料的不必要的负面影响。

在这个特殊的示例中，如果应用程序在数据库完成插入前调用 Statement.close，则数据库可能不会执行所有插入。 例如，它可能会在插入 5 行后就停止，因为存储过程尚未结束就在数据库上取消了。 如果您试图在仍存在未处理结果时关闭 Statement，SAP jConnect 则会抛出 SQLException。

此时不会向您报告丢失的插入。 强烈建议 SAP jConnect 程序员遵循以下原则：

- 调用 Statement.close 时，如果未处理完所有结果（更新计数和 ResultSet），将向服务器发送 cancel。 在只执行 select 语句的情况下，这完全可以。 但是，在执行 **insert**/**update**/**delete** 操作的情况下，这可能会导致某些操作无法按预期完成。
- 因此，在执行 **select** 以外的其它操作时，切勿在仍存在未处理结果时调用 close。
- 反之，如果调用 Statement.execute，请用 getUpdateCount、getMoreResults 和 getResultSet 方法确认您的代码已处理所有结果。

多线程的调整

如果多个线程同时调用同一 Statement 实例（CallableStatement 或 PreparedStatement）上的方法（SAP 不建议这样做），您必须手动同步对 Statement 上的方法的调用；SAP jConnect 不会自动执行此过程。

例如，如果有两个在同一 Statement 实例上运行的线程（一个线程发送查询，另一个线程处理警告），则必须同步对 Statement 上的方法的调用，否则可能会发生冲突。

ResultSet.getCursorName

JDBC 驱动程序可为任何 SQL 查询生成游标名，这样可确保始终能够返回一个字符串。不过，调用 ResultSet.getCursorName 时，SAP jConnect 不会返回名称。

假设您：

- 调用相应的语句上的 setFetchSize 或 setCursorName，或者
- 将 SELECT_OPENS_CURSOR 连接属性设置为 true，且查询的格式为 SELECT... FOR UPDATE。 例如：

```
select au_id from authors for update
```

如果不调用相应语句上的 setFetchSize 或 setCursorName，或不将 SELECT_OPENS_CURSOR 连接属性设置为 true，则会返回空值。

根据 JDBC 2.0 API 文档，所有其它 SQL 语句都不需要打开游标及返回名称。

有关如何在 SAP jConnect 中使用游标的详细信息，请参见“带有结果集的游标”。

另请参见

- 对结果集使用游标 （第 49 页）

执行存储过程

如果执行用问号代表参数值的 CallableStatement 对象中的存储过程，会比对参数既使用问号又使用实际值获得更好的性能。

而且，如果混合使用实际值和问号，就不能对存储过程使用输出参数。

本示例将 *sp_stmt* 创建为 CallableStatement 对象，以执行存储过程 **MyProc**：

```
CallableStatement sp_stmt = conn.prepareCall(
   "{call MyProc(?,?)}");
```

MyProc 中的两个参数用问号表示。 可以使用 **CallableStatement** 接口中的 `registerOutParameter` 方法将其中的一个或两个参数注册均为输出参数。

在本示例中，*sp_stmt2* 是 `CallableStatement` 对象，用于执行存储过程 **MyProc2**。

```
CallableStatement sp_stmt2 = conn.prepareCall(
   {"call MyProc2(?,'javelin')}");
```

在 *sp_stmt2* 中，其中一个给定参数值为实际值，另一个为问号。 不能将两个参数中的任一个注册为输出参数。

若要使用参数的名称绑定来通过 RPC 命令执行存储过程，请使用以下两个过程之一：

- 使用语言命令，用 `PreparedStatement` 类将输入参数直接从 Java 变量传递给语言命令。

  ```
  // Prepare the statement
  System.out.println("Preparing the statement...");
  String stmtString = "exec " + procname + " @p3=?, @p1=?";
  PreparedStatement pstmt = con.preparedStatement(stmtString);

  // Set the values
  pstmt.setString(1, "xyz");
  pstmt.setInt(2, 123);

  // Send the query
  System.out.println("Executing the query...");
  ResultSet rs = pstmt.executeQuery();
  ```

- 在 SAP jConnect 6.05 及更高版本中，可使用 `com.sybase.jdbcx.SybCallableStatement` 接口：

  ```
  import com.sybase.jdbcx.*;

  ....

  // prepare the call for the stored procedure to execute as an RPC

  String execRPC = "{call " + procName + " (?, ?)}";
  SybCallableStatement scs = (SybCallableStatement)
  con.prepareCall(execRPC);


  // set the values and name the parameters

  // also (optional) register for any output parameters
  scs.setString(1, "xyz");
  scs.setParameterName(1, "@p3");
  scs.setInt(2, 123);
  scs.setParameterName(2, "@p1");


  // execute the RPC
  // may also process the results using getResultSet()
  // and getMoreResults()
  ```

```
// see the samples for more information on processing results
ResultSet rs = scs.executeQuery();
```

安全性

SAP jConnect 提供安全套接字层 (SSL) 和 Kerberos 选项来保护客户端/服务器通信的安全。

- SSL - 使用 SSL 加密客户端和服务器应用程序之间的通信，包括登录名交换。
- Kerberos - 使用 Kerberos 为 SAP Adaptive Server 验证 Java 应用程序或 Java 应用程序用户，而不需要通过网络发送用户名或口令。 还可以使用 Kerberos 设置单点登录 (SSO) 环境，并提供 Java 应用程序的数字标识和 SAP Adaptive Server Enterprise 的数字标识之间的相互验证。

  注意： 您也可以使用 Kerberos 加密通信并提供数据完整性检查，但尚未对 SAP jConnect 实现这些功能。

可以将 Kerberos 和 SSL 结合使用，这样可以提供 SSO 以及客户端和服务器应用程序之间传输的数据加密的优点。

限制

Kerberos 和 SSL 可以与 SAP Adaptive Server 结合使用；SAP SQL Anywhere 当前不支持 SSL 和 Kerberos 这两种安全机制。

SAP 建议在 SAP jConnect 中使用 SSL 或 Kerberos 前先阅读其相关文档。 安装信息假定要使用的服务器已经进行配置，可正常使用 SSL、Kerberos 或上述两者。

有关 Kerberos、SSL 和配置 SAP Adaptive Server Enterprise 的详细信息，请参见“相关文档” （第 115 页）。 另请参见《SAP jConnect for JDBC 发行公告》中引用的 Kerberos 设置白皮书。

实现自定义 SSL 套接字插件

将自定义套接字实现插入到应用程序中，以自定义客户端与服务器间的通信。

`javax.net.ssl.SSLSocket` 是可自定义以启用加密的套接字的示例。

`com.sybase.jdbcx.SybSocketFactory` 是包含返回 `java.net.Socket` 的 `createSocket(String, int, Properties)` 方法的 SAP jConnect 扩展接口。 要在 SAP jConnect 中使用自定义套接字工厂，应用程序必须通过定义 `createSocket()` 方法来实现此接口。

SAP jConnect 使用套接字进行后续输入或输出操作。 实现 `SybSocketFactory` 的类将创建套接字并提供一般框架以添加公共套接字级功能，如下所示：

```
/**
 * Returns a socket connected to a ServerSocket on the named host,
```

```
 * at the given port.
 * @param host  the server host
 * @param port  the server port
 * @param props  Properties passed in through the connection
 * @returns Socket
 * @exception IOException, UnknownHostException
 */
public java.net.Socket createSocket(String host, int port,
Properties props)
   throws IOException, UnknownHostException;
```

传入属性将允许 SybSocketFactory 的实例使用连接属性实现智能套接字。

实现 SybSocketFactory 时，通过向应用程序传递创建套接字的不同种类的工厂或伪工厂可以使相同的应用程序代码使用不同种类的套接字。

可以使用在套接字结构中使用的参数自定义工厂。 例如，可以使用已配置的不同的网络超时或安全参数自定义工厂以返回套接字。 返回到应用程序的套接字可以是 java.net.Socket 的子类，以直接公开一些功能（如压缩、安全性、记录标记、统计信息收集或防火墙隧道）的新 API (javax.net.SocketFactory)。

注意： SybSocketFactory 是过度简化的 javax.net.SocketFactory，可以使应用程序从 java.net.* 过渡到 javax.net.*

在 jConnect 中使用自定义套接字

查看在 SAP jConnect 中使用自定义套接字的步骤。

1. 提供一个实现 com.sybase.jdbcx.SybSocketFactory 的 Java 类。
2. 设置 SYBSOCKET_FACTORY 连接属性以使 SAP jConnect 可以使用您的实现获取套接字。

   要在 SAP jConnect 中使用自定义套接字，请将 SYBSOCKET_FACTORY 连接属性设置为以下两者之一：
   - 实现 com.sybase.jdbcx.SybSocketFactory 的类名称，或者
   - DEFAULT（这将实例化新的 java.net.Socket）。

另请参见

- 连接属性 （第 8 页）
- 创建和配置自定义套接字 （第 102 页）

创建和配置自定义套接字

您可创建 SSL 套接字实例并对其进行配置，然后再由 SAP jConnect 获取该套接字。

SAP jConnect 使用该套接字连接到服务器。

本示例显示 SSL 的实现如何创建、配置和返回 SSLSocket 实例。 MySSLSocketFactory 类实现 SybSocketFactory 并扩展 javax.net.ssl.SSLSocketFactory 以实现 SSL。 它包含两个 createSocket

方法：一个用于 SSLSocketFactory，另一个用于 SybSocketFactory，功能如下：

- 创建 SSL 套接字
- 调用 SSLSocket.setEnableCipherSuites 以指定可用于加密的密码成套程序
- 返回 SAP jConnect 要使用的套接字

示例

```
public class MySSLSocketFactory extends SSLSocketFactory
   implements SybSocketFactory
 {
 /**
 * Create a socket, set the cipher suites it can use, return
 * the socket.
 * Demonstrates how cither suites could be hard-coded into the
 * implementation.
 *
 * See javax.net.SSLSocketFactory#createSocket
 */
```

```
public Socket createSocket(String host, int port)
   throws IOException, UnknownHostException
 {
   // Prepare an array containing the cipher suites that are to
   // be enabled.
   String enableThese[] =
   {
       "SSL_DH_DSS_EXPORT_WITH_DES40_CBC_SHA",
       "SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5",
       "SSL_DH_RSA_EXPORT_WITH_DES40_CBC_SHA"
   }
   ;
   Socket s =
       SSLSocketFactory.getDefault().createSocket(host, port);
   ((SSLSocket)s).setEnabledCipherSuites(enableThese);
   return s;
 }
```

```
/**
 * Return an SSLSocket.
 * Demonstrates how to set cipher suites based on connection
 * properties like:
 * Properties _props = new Properties();
 * Set other url, password, etc. properties.
 * _props.put(("CIPHER_SUITES_1",
 *    "SSL_DH_DSS_EXPORT_WITH_DES40_CBC_SHA");
 * _props.put("CIPHER_SUITES_2",
 *    "SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5");
 * _props.put("CIPHER_SUITES_3",
 *     "SSL_DH_RSA_EXPORT_WITH_DES40_CBC_SHA");
 * _conn = _driver.getConnection(url, _props);
 *
```

```
 * See com.sybase.jdbcx.SybSocketFactory#createSocket
 */
```

```
public Socket createSocket(String host, int port,
   Properties props)
   throws IOException, UnknownHostException
 {
   // check to see if cipher suites are set in the connection
   // properites
   Vector cipherSuites = new Vector();
   String cipherSuiteVal = null;
   int cipherIndex = 1;
   do
   {
       if((cipherSuiteVal = props.getProperty("CIPHER_SUITES_"
           + cipherIndex++)) == null)
       {
           if(cipherIndex <= 2)
           {
               // No cipher suites available
               // return what the object considers its default
               // SSLSocket, with cipher suites enabled.
               return createSocket(host, port);
           }
           else
           {
               // we have at least one cipher suite to enable
               // per request on the connection
               break;
           }
           else
           }
               // add to the cipher suit Vector, so that
               // we may enable them together
               cipherSuites.addElement(cipherSuiteVal);
           }
       }
       while(true);
```

```
      // lets you create a String[] out of the created vector
       String enableThese[] = new String[cipherSuites.size()];
       cipherSuites.copyInto(enableThese);
```

```
       Socket s =
           SSLSocketFactory.getDefault().createSocket
             (host, port);
       // enable the cipher suites
       ((SSLSocket)s).setEnabledCipherSuites(enableThese);
```

```
      // return the SSLSocket
       return s;
   }
```

```
  // other methods
 }
```

由于 jConnect 不需要套接字的种类信息，因此必须在返回套接字之前完成所有配置。

有关详细信息，请参见：

- `EncryptASE.java` - 位于 SAP jConnect 安装的 `sample2` 子目录下，此示例显示如何在 SAP jConnect 应用程序中使用 `SybSocketFactory` 接口。
- `MySSLSocketFactoryASE.java` - 也位于 SAP jConnect 安装的 `sample2` 子目录下，这是一个 `SybSocketFactory` 接口的实现示例，可以插入应用程序并使用。

SAP jConnect 中的 SSL 支持

要在低于 15.7 SP 100 版的 SAP jConnect 中使用 SSL 套接字，必须创建“**SybSocketFactory**” 接口的实现并通过设置 **SYBSOCKET_FACTORY** 连接属性来使用该接口的实现。

从 15.7 SP 100 版开始，SAP jConnect 具有对使用 SSL 套接字连接到 SAP Adaptive Server 的内置支持。 在设置为以下值时，新的连接属性 **ENABLE_SSL** 如下：

- false -（缺省值）SAP jConnect 将不使用 SSL 套接字。
- true - SAP jConnect 将使用 SSL 套接字，且目标 SAP Adaptive Server 必须启用 SSL 套接字连接。 SAP jConnect 将忽略 **SYBSOCKET_FACTORY** 连接属性。

注意： SAP 建议使用 **DriverManager.setLoginTimeout** 属性设置登录超时，从而允许在未启用 SSL 的 SAP Adaptive Server 上尝试建立 SSL 连接时出现连接超时。

SSL 套接字功能取决于以下标准 Java 属性：

- **javax.net.ssl.keyStore**
- **javax.net.ssl.keyStorePassword**
- **javax.net.ssl.trustStore**
- **javax.net.ssl.trustStorePassword**
- **javax.net.ssl.trustStore**
- **javax.net.ssl.trustStoreType**

有关 Java 标准属性的详细信息，请参见 Java J2SE 6 文档。

Kerberos

Kerberos 是一种网络验证协议，对客户端/服务器应用程序的验证使用加密。

Kerberos 为用户和系统管理员提供以下优势：

- Kerberos 数据库可用作用户的集中仓库。
- Kerberos 便于建立单点登录 (SSO) 环境，在此环境中用户系统登录可提供访问数据库所需的证书。
- Kerberos 是一种 IETF 标准。 Kerberos 的不同实现间支持互操作。

为 SAP jConnect 配置 Kerberos

查看配置 SAP jConnect 以使用 Kerberos 安全机制的说明。

前提条件

为 SAP jConnect 配置 Kerberos 存在几项前提条件：

- JDK 6 或更高版本
- Java 通用安全服务 (GSS) 管理器：
  - 缺省 GSS 管理器（JDK 的一部分），或
  - Wedgetail JCSI Kerberos 2.6 或更高版本，或
  - CyberSafe TrustBroker Application Security Runtime Library 3.1.0 或更高版本，或
  - 其它供应商提供的 GSS 管理器实现。
- 在服务器端得到支持，可与 GSS 库互操作，同时在客户端也得到支持，可与 GSSManager 互操作的密钥分发中心 (KDC)。

过程

1. 将 `REQUEST_KERBEROS_SESSION` 属性设置为 true。
2. 将 `SERVICE_PRINCIPAL_NAME` 属性设置为正在运行的 SAP Adaptive Server Enterprise 的名称。 通常就是服务器启动时用 `-s` 选项设置的名称。 服务主体名称也必须用 KDC 注册。 如果不为此属性设置任何值，SAP jConnect 会使用客户端计算机的主机名。
3. （可选）设置 `GSSMANAGER_CLASS` 属性。

   有关 REQUEST_KERBEROS_SESSION 和 SERVICE_PRINCIPAL_NAME 的详细信息，请参见“jConnect 连接属性” （第 9 页）

另请参见

- GSSMANAGER_CLASS 连接属性 （第 106 页）
- 编程信息 （第 3 页）

GSSMANAGER_CLASS 连接属性

使用 Kerberos 时，SAP jConnect 依赖于实现通用安全服务 (GSS) API 的几个 Java 类。

这个功能的大部分都是由 `org.ietf.jgss.GSSManager` 类提供的。

SAP jConnect 检查要用于 Kerberos 验证的 `GSSManager` 类对象的 `GSSMANAGER_CLASS` 值。

如果将 `GSSMANAGER_CLASS` 的值设置为字符串而不是类对象，则 SAP jConnect 将使用该字符串创建指定类的实例，并在 Kerberos 验证中使用新实例。

如果将 GSSMANAGER_CLASS 的值设置为 GSSManager 类对象和字符串以外的其它内容，或如果 SAP jConnect 遇到 ClassCastException，则 SAP jConnect 将抛出指明问题的 SQLException。

Java 允许供应商提供自己的 GSSManager 类实现。

如 Wedgetail Communications 和 CyberSafe Limited 提供的实现就是供应商提供的 GSSManager 实现。 用户可配置供应商编写的 GSSManager 类，以使其在特定的 Kerberos 环境中工作。 供应商提供的 GSSManager 类提供的与 Windows 的互操作性可能比标准的 Java GSSManager 类提供的还要多。

在使用供应商提供的 GSSManager 实现前，一定要阅读供应商文档。 供应商使用属性设置而不是用于 Kerberos 的标准 Java 系统属性，而且可能会定位领域名和密钥分发中心 (KDC) 条目而不使用配置文件。

设置 GSSMANAGER_CLASS 属性

通过设置 GSSMANAGER_CLASS 连接属性可以在 SAP jConnect 中使用供应商提供的 GSSManager 实现。

有两种设置此属性的方法：

- 创建一个 GSSManager 实例，并将此实例设置为 GSSMANAGER_CLASS 属性的值。
- 将 GSSMANAGER_CLASS 属性的值设置为指定 GSSManager 对象的完全限定类名称的字符串。 SAP jConnect 使用该字符串调用 Class.forName().newInstance()，并将返回的对象转换为 GSSManager 类。

在任一情况下，应用程序 CLASSPATH 变量必须包含供应商实现的类和 .jar 文件的位置。

注意： 如果不设置 GSSMANAGER_CLASS 连接属性，SAP jConnect 将使用 org.ietf.jgss.GSSManager.getInstance 方法装载缺省 Java GSSManager 实现。

使用 GSSMANAGER_CLASS 连接属性传递完全限定类名时，SAP jConnect 将调用 GSSManager 的不带参数的构造方法。 此处例示了供应商实现的一个处于缺省配置的 GSSManager，因此您无法控制 GSSManager 对象的精确配置。 如果创建自己的 GSSManager 实例，就可以使用构造方法参数来设置配置选项。

GSS 管理器示例
查看根据需求创建 GSSManager 实例或允许 SAP jConnect 在 GSSMANAGER_CLASS 连接属性设置为完全限定类名时创建 GSSManager 对象的说明。

创建 GSSManager 的实例
创建 GSSManager 的实例，并将其传递给 GSSMANAGER_CLASS 属性。

1. 在应用程序代码中实例化 GSSManager：

```
GSSManager gssMan =  new
com.dstc.security.kerberos.gssapi.GSSManager();
```

   此示例使用了不带参数的缺省构造方法。 也可以使用允许设置各种配置选项的其它供应商提供的构造方法。
2. 将新 GSSManager 实例传递给 GSSMANAGER_CLASS 连接属性：

```
Properties props = new Properties();
props.put("GSSMANAGER_CLASS", gssMan);
```

3. 在连接中使用这些连接属性（包括 GSSMANAGER_CLASS）：

```
Connection conn = DriverManager.getConnection (url, props);
```

将字符串传递给 GSSMANAGER_CLASS
在应用程序中，将字符串传递给 GSSMANAGER_CLASS。

1. 创建指定 GSSManager 对象的完全限定类名的字符串。 例如：

```
String gssManClass =
"com.dstc.security.kerberos.gssapi.GSSManager";
```

2. 将该字符串传递给 GSSMANAGER_CLASS 连接属性。 例如：

```
Properties props = new Properties();
props.put("GSSMANAGER_CLASS", gssManClass);
```

3. 在连接中使用这些连接属性（包括 GSSMANAGER_CLASS）。 例如：

```
Connection conn = DriverManager.getConnection (url, props);
```

Kerberos 环境

您可将 SAP jConnect 与三种不同的 Kerberos 实现结合使用。

- CyberSafe
- MIT
- Microsoft Active Directory

请参见 Kerberos 白皮书。

CyberSafe

查看 SAP jConnect 中的 CyberSafe Kerberos 实现。

- **加密密钥** – 在 CyberSafe KDC 中创建要由 Java 使用的主体时指定数据加密标准 (DES) 密钥。

  Java 参考实现不支持三倍数据加密标准 (3DES) 密钥。

  注意： 如果将 CyberSafe GSSManager 与 CyberSafe KDC 结合使用且设置了 `GSSMANAGER_CLASS` 属性，则可使用 3DES 密钥。

- **地址映射和领域信息** – CyberSafe 使用 DNS 记录来定位 KDC 地址映射和领域信息。

  CyberSafe Kerberos 不使用 `krb5.conf` 配置文件。 或者，CyberSafe 分别在 `krb.conf` 和 `krb.realms` 文件中定位 KDC 地址映射和领域信息。 有关详细信息，请参见 CyberSafe 文档。

  如果使用的是标准 Java `GSSManager` 实现，还必须创建 `krb5.conf` 文件以供 Java 使用。 CyberSafe `krb.conf` 文件的格式与 `krb5.conf` 文件的格式不同。 根据 Java SE 文档或 MIT 文档中指定的内容创建 `krb5.conf` 文件。 如果正在使用 CyberSafe `GSSManager`，就不需要 `krb5.conf` 文件。

  有关 `krb5.conf` 文件的示例，请参见 Kerberos 设置白皮书，该 URL 在《SAP jConnect for JDBC 发行公告》中引用。

- **Solaris** – 在 Solaris 上使用 CyberSafe 客户端库时，应确保库搜索路径在任何其它 Kerberos 库前包含 CyberSafe 库。

客户端将 `krb5.conf` 文件与 CyberSafe 或 MIT KDC 结合使用。 例如：

```
# Please note that customers must alter the
# default_realm, [realms] and [doamin_realm]
# information to reflect their Kerberos environment.
# Customers should *not* attempt to use this file as is.
#

[libdefaults]
        default_realm = ASE
        default_tgs_enctypes = des-cbc-crc
        default_tkt_enctypes = des-cbc-crc
        kdc_req_checksum_type = 2
        ccache_type = 2

[realms]

        ASE = {
            kdc = kdchost
            admin_server = kdchost
        }

[domain_realm]
        .sybase.com = ASE
```

```
        sybase.com = ASE

[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log
 kdc_rotate = {

# How often to rotate kdc.log. Logs will get rotated
# no more often than the period, and less often if the
# KDC is not used frequently.

  period = 1d

# how many versions of kdc.log to keep around
# (kdc.log.0, kdc.log.1, ...)

  versions = 10
 }

[appdefaults]
 kinit = {
  renewable = true
  forwardable= true
 }
```

MIT

创建 MIT KDC 中要由 Java 使用的主体时应指定 DES 密钥。

Java 参考实现不支持 3DES 密钥。

如果计划只使用标准 Java `GSSManager` 实现，应指定 `des-cbc-crc` 或 `des-cbc-md5` 类型的加密密钥。 将加密类型指定为：

```
des-cbc-crc:normal
```

其中 `normal` 是密钥 salt 的类型。 也可以使用其它 salt 类型。

注意： 如果正在使用 Wedgetail `GSSManager`，则可以在 MIT KDC 中创建类型为 `des3-cbc-sha1-kd` 的主体。

Microsoft Active Directory

查看适用于 Kerberos 的 Microsoft Active Directory 服务器中的组件。

- **用户帐户和服务主体** – 确保已在 Active Directory 中为用户主体用户（用户）和服务主体（代表数据库服务器的帐户）设置了帐户。 用户主体和服务主体都应在 Active Directory 中创建为 Users 。
- **客户端计算机** – 修改 Windows 注册表以使用 Java 参考实现设置 SSO 环境。

  有关修改 Windows 注册表的说明，请参见 Microsoft 支持站点。
- **配置文件** – 在 Windows 上，Kerberos 配置文件称为 `krb5.ini`。 缺省情况下，Java 会在 `C:\WINNT\krb5.ini` 中查找 `krb5.ini`。

  Java 允许指定该文件的位置。 `krb5.ini` 的格式与 `krb5.conf` 的格式相同。

有关 krb5.conf 文件的示例，请参见《SAP jConnect for JDBC 发行公告》中引用的 Kerberos 白皮书。

有关 Microsoft Active Directory 的 Kerberos 的详细信息，请参见 Microsoft Developer Network。

客户端将 krb5.conf 文件与 Active Directory 结合使用以作为 KDC。 例如：

```
# Please note that customers must alter the
# default_realm, [realms] and [domain_realm]
# information to reflect their Kerberos environment.
# Customers should *not* attempt to use this file as is.
#

[libdefaults]
        default_realm = W2K.SYBASE.COM
        default_tgs_enctypes = des-cbc-crc
        default_tkt_enctypes = des-cbc-crc
        kdc_req_checksum_type = 2
        ccache_type = 2

[realms]

        W2K.SYBASE.COM = {
           kdc = 1.2.3.4:88
           admin_server = adserver
         }

[domain_realm]
        .sybase.com = W2K.SYBASE.COM
        sybase.com = W2K.SYBASE.COM

[logging]
        default = FILE:/var/krb5/kdc.log
        kdc = FILE:/var/krb5/kdc.log
 kdc_rotate = {

# How often to rotate kdc.log. Logs will get rotated no
# more often than the period, and less often if the KDC
# is not used frequently.

  period = 1d

# how many versions of kdc.log to keep around
# (kdc.log.0, kdc.log.1, ...)

  versions = 10
 }

[appdefaults]
 kinit = {
  renewable = true
  forwardable= true
 }
```

设置 DES 加密

如果要使用 Java 参考 GSS 管理器实现，就必须对用户和服务主体使用 DES 加密。

1. 在 Active Directory 中，右键单击特定用户主体或服务主体名。
2. 选择**“属性”(Properties)**。
3. 单击**“帐号”(Account)** 选项卡。
4. 为用户主体和服务主体指定应使用的 DES 加密类型。

示例应用程序

在 `jConnect-16_0/sample2` 目录中有两个注释的代码示例，说明了如何建立与 Adaptive Server Enterprise 的 Kerberos 连接。

- `ConnectKerberos.java` - 登录 Adaptive Server Enterprise 的简单 Kerberos 示例。
- `ConnectKerberosJAAS.java` - 更详细的示例，显示了如何在应用程序/服务器代码中实现 Kerberos 登录。

运行 ConnectKerberos.java

查看运行 `ConnectKerberos.java` 文件示例应用程序的说明。

1. 确保计算机具有有效 Kerberos 证书。 该任务根据计算机和环境的不同而不同。
   - Windows - 通过使用 Kerberos 验证成功登录，可以为 Active Directory 环境中的计算机建立 Kerberos 证书。
   - UNIX 或 Linux - 可以使用 Kerberos 客户端的 **kinit** 实用程序为 UNIX 或 Linux 计算机建立 Kerberos 证书。 如果未使用 **kinit** 获取初始证书，在试图运行示例应用程序时，系统会提示您输入用户名和口令。

   注意：一般情况下，由标准 JDK 提供的 GSSManager 提供程序实现只能使用 DES_CBC_MD5 和 DES_CBC_CRC 加密类型。 通过使用第三方软件并设置 `GSSMANAGER_CLASS` 也许能够使用其它加密类型。
2. 确定计算机证书的位置。
   - Windows - 对于运行在 Active Directory 环境中的计算机，Kerberos 证书存储在内存中的票据高速缓存中。
   - UNIX 或 Linux - 对于使用 Kerberos 的 JRE、CyberSafe、Solaris 或 MIT 实现的 UNIX 或 Linux 计算机，缺省情况下，**kinit** 将证书放在 `/tmp/krb5cc_{user_id_number}` 中，其中 *{user_id_number}* 对于您的用户名来说是唯一的。

   如果证书位于其它位置，则需通过设置 `ticketCache` 属性在 `sample2/exampleLogin.conf` 文件中指定该位置。

3. 向 Java 参考实现指定 KDC 计算机的缺省领域和主机名。 Java 可从 `krb5.conf` 或 `krb5.ini` 配置文件或从 Java System 属性获取该信息。 如果使用供应商提供的 GSS 管理器实现，则此实现可能从 DNS SRV 记录获取主机及领域信息。

   SAP 推荐使用 Kerberos 配置文件，它允许对 Kerberos 环境进行更多控制，包括向 Java 指定验证期间请求的加密类型的功能。

   注意： 在 Linux 上，Java 参考实现在 `/etc/krb5.conf` 中查找 Kerberos 配置文件。

   如果不使用 Kerberos 配置文件，且未将 Kerberos 配置设置为使用 DNS SRV 记录，就可以使用 **java.security.krb5.realm** 和 **java.security.krb5.kdc** 系统属性指定领域和 KDC。
4. 编辑 `ConnectKerberos.java` 以使连接 URL 指向您的数据库。
5. 编译 `ConnectKerberos.java`。

   确保您使用的是 JDK 版本 6 或更高版本。 通览源代码注释，并确保已在 CLASSPATH 环境变量中指定了 jConnect 安装中的 `jconn4.jar`。
6. 执行 `ConnectKerberos.class`：

   ```
   java ConnectKerberos
   ```

   确保您使用的是 Java 6 版本的可执行文件。 示例应用程序输出说明已成功建立连接并执行 SQL：

   ```
   select 1
   ```

   - 要在不使用 Kerberos 配置文件的情况下执行示例，请使用：

     ```
     java -Djava.security.krb5.realm=your_realm
     -Djava.security.krb5.kdc=your_kdc ConnectKerberos
     ```

     其中，*your_realm* 是缺省领域，而 *your_kdc* 是您的 KDC。
   - 如有必要，可在调试模式下运行示例应用程序，以查看 Java Kerberos 层的调试输出：

     ```
     java -Dsun.security.krb5.debug=true ConnectKerberos
     ```

   也可以使用位于 `jConnect-16_0/classes` 目录的 **IsqlApp**（**isql** 的 Java 版本）来建立 Kerberos 连接：

   ```
   java IsqlApp -S jdbc:sybase:Tds:hostName:portNum
   -K service_principal_name
   -F path_to_JAAS_login_module_config_file
   ```

互操作性

SAP jConnect 支持 KDC、GSS 库和平台的互操作性组合，SAP jConnect 在这些平台上成功地建立了到 SAP Adaptive Server Enterprise 的连接。

缺少任何特定组合并不表示不能与该组合建立连接。 您可在 jConnect for JDBC 网站上找到最新状态。

表 7. 互操作性组合

客户端平台	KDC	GSSManager	GSS C 库 \[a]	SAP ASE 平台
Solaris 8[b]	CyberSafe	Java GSS	CyberSafe	Solaris 8
Solaris 8	Active Directory[c]	Java GSS	CyberSafe	Solaris 8
Solaris 8	MIT	Java GSS	CyberSafe	Solaris 8
Solaris 8	MIT	Wedgetail GSS[d]	MIT	Solaris 8
Solaris 8	CyberSafe	Wedgetail GSS[e]	CyberSafe	Solaris 8
Windows 2000	Active Directory	Java GSS	CyberSafe	Solaris 8
Windows XP	Active Directory	Java GSS[f]	CyberSafe	Solaris 8

a. 这些是 SAP Adaptive Server Enterprise 用于提供 GSS 功能的库。

b. 表中所有 Solaris 8 平台均为 32 位。

c. 表中所有 Active Directory 条目均指 Windows 2000 上运行的 Active Directory 服务器。 要实现 Kerberos 互操作性，Active Directory 用户必须设置为“为此帐户使用 DES 加密类型”。

d. 使用 Wedgetail JCSI Kerberos 2.6。 加密类型为 3DES。

e. 使用 Wedgetail JCSI Kerberos 2.6。 加密类型为 DES。

f. Java 1.4.x 有一个错误，它要求客户端使用 System.setProperty("os.name", "Windows 2000"); 来确保 Java 可在 Windows XP 客户端找到内存中的证书。

SAP 建议使用这些库的最新版本。 如果想要使用较旧版本，或者非 SAP 产品有问题，请联系供应商。

加密类型

典型 JRE 提供的标准 Java GSS 实施仅支持 DES 加密。

若要使用 3DES、RC4-HMAC、AES-256 或 AES-128 加密标准，就必须使用 CyberSafe 或 Wedgetail GSSManager。

有关 Wedgetail 和 CyberSafe 的详细信息，请参见各自的文档。

Kerberos 故障排除

查看 Kerberos 安全问题故障排除期间的注意事项。

- Java 参考实现仅支持 DES 加密类型。 必须配置 Active Directory 和 KDC 主体使用 DES 加密。
- `SERVICE_PRINCIPAL_NAME` 属性的值必须设置为启动数据服务器时用 `-s` 选项指定的同一名称。

- 检查 `krb5.conf` 和 `krb5.ini` 文件。对于 CyberSafe 客户端，检查 `krb.conf` 和 `krb.realms` 文件或 DNS SRV 记录。
- 可在 JAAS 登录配置文件中将 `debug` 属性设置为 true。
- 可在命令行中将 `debug` 属性设置为 true：

```
-Dsun.security.krb5.debug=true
```

- JAAS 登录配置文件提供了多个可进行设置以满足特定需要的选项。有关 JAAS 和 Java GSS API 的信息，请参见：
  - JAAS 登录配置文件
  - 类 Krb5LoginModule
  - JGSS 故障排除

相关文档

查看有关 Kerberos 安全机制的附加信息。

- 有关 JAAS 和 Java GSS API 的 Java 教程
- MIT Kerberos 文档和下载站点
- CyberSafe Limited
- 有关 Windows-Kerberos 互操作性的 CyberSafe Limited 文档
- Kerberos RFC 1510

故障排除

查看 SAP jConnect 使用过程中可能出现的问题的解决方案和解决方法。

使用 SAP jConnect 进行调试

SAP jConnect 包括一个 Debug 类，该类包含一组调试函数。

Debug 方法包括多个断言函数、跟踪函数和计时器函数，用于定义调试过程的范围以及调试结果的输出位置。

SAP jConnect 安装还包括一组完整的具有调试功能的类。 这些类位于 SAP jConnect 安装目录的 devclasses 子目录下。 为了进行调试，必须重定向 CLASSPATH 环境变量以引用调试模式运行时类 (devclasses/jconn4d.jar)，而不是引用标准的 SAP jConnect classes 目录。 也可以在运行 Java 程序时，通过将 **-classpath** 参数显式提供给 **java** 命令来实现此目的。

获取 Debug 类的实例

导入 Debug 接口并通过调用 SybDriver 类的 getDebug 方法获取 Debug 类的实例。

```
import com.sybase.jdbcx.Debug;
//
...
SybDriver sybDriver = (SybDriver)
Class.forName("com.sybase.jdbc4.jdbc.SybDriver").newInstance();
```

```
Debug sybdebug = sybDriver.getDebug();
...
```

在应用程序中打开调试程序

使用 Debug 对象的 debug 方法打开应用程序中的调试程序。

添加以下调用：

```
sybdebug.debug(true, [classes], [printstream]);
```

classes 参数是一个字符串，以冒号分隔的形式列出要调试的特定类。 例如：

```
sybdebug.debug(true,"MyClass")
```

以及：

```
sybdebug.debug(true,"MyClass:YourClass")
```

在类字符串中使用“STATIC”可为 SAP jConnect 中的所有静态方法以及指定类打开调试程序。 例如：

```
sybdebug.debug(true,"STATIC:MyClass")
```

可以指定“ALL”为所有类打开调试程序。 例如：

```
sybdebug.debug(true,"ALL");
```

printstream 参数是可选的。 如果未指定 printstream 参数，调试输出将传送到通过 DriverManager.setLogStream 指定的输出文件中。

在应用程序中关闭调试程序

查看关闭调试程序方法的说明。

添加以下调用：

```
sybdebug.debug(false);
```

为调试程序设置 CLASSPATH

在运行启用了调试的应用程序之前，请将经过优化的 SAP jConnect jconn4.jar 文件替换为调试版本 jconn4d.jar，后者位于 SAP jConnect 安装目录下的 devclasses 子目录中。

若要设置环境变量，则：

- 在 UNIX 系统中，将 $JDBC_HOME/classes/jconn4.jar 替换为 $JDBC_HOME/devclasses/jconn4d.jar。
- 在 Windows 系统中，将 %JDBC_HOME%\classes\jconn4.jar 替换为 %JDBC_HOME%\devclasses\jconn4d.jar。

使用调试方法

在 SAP jConnect 中自定义调试方法。

可以添加对其它 Debug 方法的调用。

如果上述方法中存在静态方法，则对象参数需使用空值。

- println – 如果已启用调试程序并且对象包含在要调试的类列表中，请使用该方法定义要在输出日志中输出的消息。 调试输出将传送到 sybdebug.debug 所指定的文件中。
  语法为：
  ```
  sybdebug.println(object,message string);
  ```
  例如：
  ```
  sybdebug.println(this,"Query: "+ query);
  ```
  在输出日志中生成如下消息：
  ```
  myApp(thread[x,y,z]): Query: select * from authors
  ```
- assert – 使用该方法声明条件，并在不满足该条件时抛出一个运行时异常。 也可以定义在不满足条件时将消息输出到输出日志中。
  语法为：

```
sybdebug.assert(object,boolean condition,message
   string);
```

例如：

```
sybdebug.assert(this,amount<=buf.length,amount+"
   too big!");
```

如果“amount”超出 `buf.length` 的值，将会在输出日志中生成如下消息：

```
java.lang.RuntimeException:myApp(thread[x,y,z]):
Assertion failed: 513 too big!
at jdbc.sybase.utils.sybdebug.assert(
sybdebug.java:338)
at myApp.myCall(myApp.java:xxx)
at .... more stack:
```

- `startTimer` 和 `stopTimer` – 使用这些方法启动和停止用于测量事件所占用的时间（以毫秒计）的计时器。 该方法为每个对象保留一个计时器，并为所有静态方法保留一个计时器。 启动计时器的语法为：

  ```
  sybdebug.startTimer(object);
  ```

  停止计时器的语法为：

  ```
  sybdebug.stopTimer(object,message string);
  ```

  例如：

  ```
  sybdebug.startTimer(this);
  stmt.executeQuery(query);
  sybdebug.stopTimer(this,"executeQuery");
  ```

  在输出日志中生成如下消息：

  ```
  myApp(thread[x,y,z]):executeQuery elapsed time =
    25ms
  ```

动态记录

自 15.7 ESD #4 开始，SAP jConnect for JDBC 通过实现标准 Java Logger 机制支持记录机制。

示例
应用程序可获取 SAP jConnect 记录器的句柄，并根据需要打开或关闭记录。

```
try
{
// Get logger for all classes present in
//"com.sybase.jdbc4.jdbc" package

Logger LOG = Logger.getLogger("com.sybase.jdbc4.jdbc");

// To log class-specific log message,
// provide complete class name, for example:
```

```
//Logger.getLogger("com.sybase.jdbc4.jdbc.
//SybConnection");
//Get handle as per user's requirement
Handler handler = new ConsoleHandler();

//Set logging level
handler.setLevel(Level.ALL);

//Added user specific handler to logger object
LOG.addHandler(handler);

//Set logging level
LOG.setLevel(Level.ALL);

Class.forName("com.sybase.jdbc4.jdbc.SybDriver");
Properties properties = new Properties();
properties.put("USER", USER_NAME);
properties.put("PASSWORD", PASSWORD);
Connection con =   DriverManager.getConnection("jdbc:sybase:Tds:" +
    HOST_PORT, properties);
Statement stmt = con.createStatement();
stmt.execute("select @@version");

//Dynamically turn off logging mechanism
LOG.setLevel(Level.OFF);
con.close();
...
}
```

记录级别

SAP jConnect 允许应用程序用户将消息粒度设置为 Level.FINE、Level.FINER 和 Level.FINEST。 例如：

- 如果用户将 `SybConnection` 类的记录级别设置为 Level.FINE，jConnect 将报告：`Dr1_Col setClientInfo(Properties)`
- 如果将 `SybConnection` 类的记录级别设置为 Level.FINER，将报告：`Dr1_Co1 setClientInfo(Properties.size = [3])`
- 如果将 `SybConnection` 类的记录级别设置为 Level.FINEST，将报告：`Dr1_Co1 setClientInfo(Properties = [[ClientUserValue, ApplicationNameValue, ClientHostnameValue]])`

在 SAP jConnect 中动态启用记录

在应用程序中通过编程方式使用 `LogHandler` API 可以动态启用记录。

1. 通过编程方式使用 `LogHandler` API 以启用或禁用记录。 输入以下代码以在 `SybConnection` 和 `SybStatement` 类上启用主控台级记录：

```
LogManager logManager = LogManager.getLogManager();
    Handler handler = new ConsoleHandler();
    handler.setLevel(Level.ALL);
    Logger connLOG =
```

```
Logger.getLogger(SybConnection.class.getName());
    connLOG.addHandler(handler);
    connLOG.setLevel(Level.FINE);
    logManager.addLogger(connLOG);
    Logger stmtLOG =
Logger.getLogger(SybStatement.class.getName());
    stmtLOG.addHandler(handler);
    stmtLOG.setLevel(Level.FINE);
    logManager.addLogger(stmtLOG);
```

2. 运行应用程序。
3. 可以像步骤 1 中给出的那样使用 `LogHandler` 来动态调整记录设置。 下面提供一些可以实现此目的的建议方法：
   - 如果可在某个界面中更改用于内部管理记录级别的属性，显示该界面。
   - 如果某个代理线程可根据应用程序中的需要来调整记录，运行该代理线程。

在 SAP jConnect 中静态启用记录

在实施了标准 Java 记录机制的 SAP jConnect 中静态启用记录。

1. 使用文本编辑器在 `$JRE_DIR/lib/logging.properties` 中修改标准记录文件的内容。

   ```
   handlers= java.util.logging.FileHandler
   java.util.logging.FileHandler.formatter =
   com.sybase.jdbc4.utils.LogUtil
   .level= ALL
   ```

2. 在文件中添加或输入以下内容：

   ```
   com.sybase.jdbc4.jdbc.SybDriver.level = FINEST
   com.sybase.jdbc4.jdbc.SybConnection.level = FINEST
   com.sybase.jdbc4.jdbc.SybStatement.level = FINER
   com.sybase.jdbc4.jdbc.SybPreparedStatement.level = FINE
   com.sybase.jdbc4.jdbc.SybResultSet.level = FINE
   ```

3. 将记录级别调整为 Level.FINE、Level.FINER 和 Level.FINEST，以设置相应的记录粒度。

   注意： SAP jConnect 不支持软件包级记录。

4. 保存 `logging.properties` 文件。

捕获 TDS 通信

TDS 是用于处理客户端应用程序与 SAP Adaptive Server 间通信的 SAP jConnect 专有协议。

SAP jConnect 包括一个 `PROTOCOL_CAPTURE` 连接属性，用于将原始 TDS 包捕获到文件。

如果应用程序出现故障，而且无法在应用程序或服务器内部加以解决，则可使用 `PROTOCOL_CAPTURE` 捕获客户端和服务器之间的通信，并将其存放到一个文件中。

该文件包含二进制数据，不能直接解读。您可以将该文件发送给 SAP 技术支持部门进行分析。

注意： 捕获并保存到文件中的 TDS 协议数据包含敏感的用户验证信息，也可能含有公司或客户数据等机密信息。 为避免未经授权或意外泄露此类机密数据，请使用文件权限或加密对包含捕获数据的文件加以正确保护。

PROTOCOL_CAPTURE 连接属性

使用 PROTOCOL_CAPTURE 连接属性指定一个文件，用来接收应用程序和 SAP Adaptive Server 之间交换的 TDS 包。

PROTOCOL_CAPTURE 会立即生效，这样在建立连接过程中交换的 TDS 包就会被写入到指定的文件中。 所有的 TDS 包将继续被写入到该文件中，直到执行 **Capture.pause** 或关闭该会话为止。

本示例显示如何使用 PROTOCOL_CAPTURE 将 TDS 数据发送到文件 tds_data 中：

```
...
props.put("PROTOCOL_CAPTURE", "tds_data")
Connection conn = DriverManager.getConnection(url, props);
```

其中，*url* 是连接的 URL，*props* 是用于指定连接属性的 Properties 对象。

Capture 类中的 Pause 和 Resume 方法

Capture 类在 com.sybase.jdbcx 包中，并且包含 pause 和 resume 方法。

Capture.pause 停止将原始 TDS 包捕获到文件中；Capture.resume 重新开始捕获。

整个会话的 TDS 捕获文件可能会变得很大。 如果您知道要捕获的 TDS 数据在应用程序中的位置，则可限制捕获文件的大小。

限制捕获文件的大小

查看限制捕获文件大小的说明。

1. 在建立连接后，立即获取用于该连接的 Capture 对象，并使用 pause 方法停止捕获 TDS 数据：

   ```
   Capture cap = ((SybConnection)conn).getCapture();
    cap.pause();
   ```

2. 将 cap.resume 放置在希望开始捕获 TDS 数据的位置。
3. 将 cap.pause 放置在希望停止捕获数据的位置。

解决连接错误

解决在试图建立连接或启用网关时出现的问题。

```
Gateway connection refused:
```

```
HTTP/1.0 502 Bad Gateway|Restart Connection
```

该错误消息表明，用于连接到 Adaptive Server 的 *hostname* 或 *port#* 出错。 检查 `$SYBASE/interfaces` (UNIX) 或 `%SYBASE%\ini\sql.ini` (Windows) 中的 [query] 项。

如果在检验了 *hostname* 和 *port#* 之后问题仍然存在，请使用“verbose”系统属性启动 HTTP 服务器以进一步了解相关信息。

在 Windows 系统中，转至 DOS 提示符并输入：

```
httpd -Dverbose=1 > filename
```

在 UNIX 系统中，输入：

```
sh httpd.sh -Dverbose=1 > filename &
```

其中，*filename* 是调试消息输出文件。

您的 Web 服务器可能不支持 `connect` 方法。 小程序仅能够连接到可下载这些小程序的主机。

HTTP 网关和 Web 服务器必须在同一主机上运行。 在这种情况下，小程序可以通过 HTTP 网关（HTTP 网关能够将请求路由到相关数据库）控制的端口连接到同一主机。

若要查看该过程是如何实现的，请查阅位于 jConnect 安装目录下的 `sample2` 子目录中的 `Isql.java` 和 `gateway.html` 源文件。 搜索“proxy”。

管理 SAP jConnect 应用程序所使用的内存

如果注意到 SAP jConnect 应用程序所使用的内存有所增加，请使用 `Statement` 对象和子类。

- 在 SAP jConnect 应用程序中，显式关闭最近一次使用的所有 `Statement` 对象以及子类（例如，`PreparedStatement`、`CallableStatement`）以阻止语句累积在内存中。 仅关闭 `ResultSet` 是不够的。

  例如，此语句将引发问题：

  ```
  ResultSet rs = _conn.prepareCall(_query).execute();

  ...
  rs.close();
  ```

  请改用：

  ```
  PreparedStatement ps = _conn.prepareCall(_query);
  ResultSet rs = ps.executeQuery();

  ...
  rs.close();
  ps.close();
  ```

- 根据所连接到的 SAP Adaptive Server 或 SAP SQL Anywhere 数据库版本，可能不存在对可滚动游标或可更新式可滚动游标的内在支持。 为了在后端服务器不支持

可滚动游标或可更新式可滚动游标的情况下对这些游标提供支持，SAP jConnect 会根据需要在每次调用 `ResultSet.next` 时在客户端上高速缓存行数据。 但到达结果集的末尾时，整个结果集将存储到客户端内存中。 由于这可能会导致性能降低，因此 SAP 建议仅在结果集相当小的情况下使用 TYPE_SCROLL_INSENSITIVE 结果集。 SAP jConnect 可确定 SAP Adaptive Server 连接是否支持本机可滚动游标功能，以及是否使用该功能代替客户端高速缓存。 这样一来，大多数应用程序都可以在访问无序行的过程中获得显著性能提高并可降低客户端内存要求。

解决存储过程错误

解决在试图使用 SAP jConnect 和存储过程时出现的问题。

RPC 返回比已注册参数少的输出参数

如果通过调用 `CallableStatement.registerOutParam` 注册的参数多于在存储过程中声明为“OUTPUT”的参数，则会发生错误。

```
SQLState: JZ0SG - An RPC did not return as many output parameters as
the application had registered for it.
```

确保已将所有相应参数声明为“OUTPUT”。 注意内容如下的代码行：

```
create procedure yourproc (@p1 int OUTPUT, ...
```

注意： 如果在使用 SAP SQL Anywhere 时收到此错误，请升级到 SAP SQL Anywhere 5.5.04 或更高版本。

Fetch/State 错误

如果查询不返回行数据，则会发生 `Fetch/State` 错误。

可以使用 `CallableStatement.executeUpdate` 或 `execute` 方法，而不使用 `executeQuery` 方法。

根据 JDBC 标准的要求，如果 `executeQuery` 没有结果集，SAP jConnect 会抛出一个 SQL 异常。

在非链式事务模式中执行存储过程

如果 JDBC 试图在 `autocommit(true)` 模式下发送连接，则会发生此错误。

`SAP Adaptive Server 错误 7713 – 只能在非链式事务模式中执行存储过程。`

应用程序可使用 `Connection.setAutoCommit(false)` 或通过使用 **“set chained on”** 语言命令将连接更改为链式模式。 如果存储过程不是在兼容模式中创建的，则会出现该错误。

要修复该问题，请使用：

```
sp_procxmode procedure_name,"anymode"
```

解决自定义套接字实现错误

如果试图在调用 `sun.security.ssl.SSLSocketImpl.setEnabledCipherSuites` 时设置 SSL 套接字，则会发生自定义套接字实现错误。

```
java.lang.IllegalArgumentException:
SSL_SH_anon_EXPORT_WITH_RC4_40_MDS
```

检验 SSL 库是否位于系统库路径中。

性能和调优

查看使用 SAP jConnect 时微调或改进性能的说明。

改进 SAP jConnect 性能

查看用于通过 SAP jConnect 优化应用程序性能的选项。

- 使用 `TextPointer.sendData` 方法将文本和图像数据发送到 SAP Adaptive Server 数据库。
- 创建在会话过程中重复使用的动态 SQL 语句的预编译 `PreparedStatement` 对象。
- 使用批处理从而减少网络通信量而提高性能：具体来说，所有的查询通过组的方式发送到服务器，并且所有返回到客户端的响应也通过组发出。
- 对于可能会移动图像数据、较大行集以及过长文本数据的会话，使用 `PACKETSIZE` 连接属性设置最大可用包大小。
- 对于通过 TDS 建立隧道的 HTTP，设置最大 TDS 包大小并配置 Web 服务器以支持 HTTP1.1 Keep-Alive 功能。 此外，将 *SkipDoneProc* 服务器小程序参数设置为 true。
- 使用协议游标，即 `LANGUAGE_CURSOR` 连接属性的缺省设置。
- 如果使用 TYPE_SCROLL_INSENSITIVE 结果集，应仅在结果集相当小时使用它们。

另请参见

- 批处理更新支持 （第 59 页）
- Image 数据类型 （第 62 页）
- 对动态 SQL 中的预准备语句的性能调优 （第 129 页）
- SAP jConnect 中的 TYPE_SCROLL_INSENSITIVE 结果集 （第 57 页）
- LANGUAGE_CURSOR 连接属性 （第 135 页）

BigDecimal 范围重设

JDBC 1.0 规范要求 `getBigDecimal` 方法有一个范围因子。

当从服务器返回 `BigDecimal` 对象时，必须通过 `getBigDecimal` 使用的原始范围因子对其进行范围重设。

若要减少范围重设所需的时间，请使用 JDBC 2.0 `getBigDecimal` 方法（SAP jConnect 在 `SybResultSet` 类中实现，且不需要 *scale* 值）：

```
public BigDecimal getBigDecimal(int columnIndex)
   throws SQLException
```

例如：

```
SybResultSet rs =
   (SybResultSet)stmt.executeQuery("SELECT
   numeric_column from T1");
 while (rs.next())
 {
   BigDecimal bd rs.getBigDecimal(
     "numeric_column");
      ...
 }
```

REPEAT_READ 连接属性

如果将 REPEAT_READ 连接属性设置为 false，则可改进从数据库中检索结果集的性能。

当 REPEAT_READ 为 false 时：

- 必须按照列索引顺序读取列值。 如果要按名称而不是按列编号访问列将是很困难的。
- 不能多次读取行中的列值。

SunIoConverter 字符集转换

如果使用多字节字符集并需要改进驱动程序性能，可以使用随 SAP jConnect 示例提供的 SunIoConverter 类。

此转换程序以 Oracle Corporation 提供的 sun.io 类为基础。

SunIoConverter 类不是字符集转换程序功能的纯 Java 实现，因此未与标准 SAP jConnect 产品相集成。它提供此转换程序类以供您参考，您可将其与 SAP jConnect 驱动程序结合使用以改进字符集转换性能。

注意： 根据 SAP 测试，SunIoConverter 类在所测试的所有虚拟机上都改善了性能。不过，Oracle Corporation 保留在 JDK 未来版本中删除或更改 sun.io 类的权利。 因此，此 SunIoConverter 类可能会与 JDK 的更高版本不兼容。

要使用 SunIoConverter 类，必须先安装 SAP jConnect 示例应用程序。 示例安装完毕后，设置 CHARSET_CONVERTER_CLASS 连接属性，使其引用 jConnect 安装目录 sample2 子目录中的 SunIoConverter 类。

有关安装 SAP jConnect 及其组件（包括示例应用程序）的完整说明，请参见《SAP jConnect for JDBC 安装指南》。

若正在使用缺省字符集为 iso_1 的数据库或仅前 7 位 ASCII，则通过使用 TruncationConverter 可获得显著的性能优势。

另请参见

- SAP jConnect 字符集转换程序 （第 38 页）

对动态 SQL 中的预准备语句的性能调优

在 Embedded SQL™ 中，动态语句是需要在运行期编译而不是静态编译的 SQL 语句。

通常，动态语句包含输入参数，但这不是必需的。 在 SQL 中，**prepare** 命令用于预编译动态语句并将其保存，以使其在会话期间不必重新编译便可重复执行。

如果语句在一个会话中使用多次，预编译将比每次使用时将其发送到数据库并进行编译提供更好的性能。 语句越复杂，性能优势就越显著。

如果可能仅使用几次语句，预编译可能会降低效率，因为在数据库中的预编译、保存以及随后的释放都会增加开销。

预编译要执行的动态 SQL 语句并将其保存到内存中会耗用时间和资源。 如果在会话中不太可能多次使用同一个语句，执行数据库 **prepare** 的开销可能会大大超过其性能优势。 另外需要考虑的是，只要数据库中预准备了动态 SQL 语句，它就非常类似于存储过程。 在某些情况下，创建存储过程并使其驻留在服务器中可能会比在应用程序中定义预准备语句更可取。

可以使用 SAP jConnect 优化 SAP 数据库中动态 SQL 语句的性能，方法为：

- 在同一语句在会话中可能多次执行的情况下，创建包含预编译语句的 `PreparedStatement` 对象。
- 在同一语句在会话中很少使用的情况下，创建包含未编译 SQL 语句的 `PreparedStatement` 对象。

设置 `DYNAMIC_PREPARE` 连接属性并创建 `PreparedStatement` 对象的最佳方法可能取决于应用程序是否需要跨 JDBC 驱动程序移植，或者所编写的应用程序是否允许特定于 SAP jConnect 的 JDBC 扩展。

SAP jConnect 提供了动态 SQL 语句的性能调优功能。

另请参见

- 选择预准备语句和存储过程 （第 129 页）

选择预准备语句和存储过程

如果创建包含预编译动态 SQL 语句的 `PreparedStatement` 对象，一旦该语句在数据库中编译，它实际上就变成了驻留在内存中且附加至会话相关数据结构的存储过程。

在决定是否维护数据库中的存储过程或在应用程序中创建包含已编译 SQL 语句的 `PreparedStatement` 对象时，资源需求以及数据库和应用程序维护都是需要考虑的重要因素：

- 存储过程一旦被编译，就跨所有的连接在全局都可用。 相反，PreparedStatement 对象中的动态 SQL 语句必须在每个使用它的会话中进行编译和释放。
- 如果应用程序访问多个数据库，使用存储过程意味着相同的存储过程必须在所有目标数据库上都可用。 这样便产生了数据库维护问题。 如果对动态 SQL 语句使用 PreparedStatement 对象，就可避免出现这一问题。
- 如果应用程序创建了 CallableStatement 对象以调用存储过程，则可在该存储过程中封装 SQL 代码和表引用。 然后，可修改基础数据库或 SQL 代码而不必更改应用程序。

可移植应用程序中的预准备语句

如果在来自不同供应商的数据库上运行应用程序，而且想要一些 PreparedStatement 对象包含预编译语句，而其它对象包含未编译语句，请在可移植应用程序中使用 PreparedStatement。

- 访问 SAP 数据库时，确保将 DYNAMIC_PREPARE 连接属性设置为 true。
- 要返回包含预编译语句的 PreparedStatement 对象，请以标准方式使用 Connection.prepareStatement：

```
PreparedStatement ps_precomp =
   Connection.prepareStatement(sql_string);
```

- 要返回包含未编译语句的 PreparedStatement 对象，请使用 Connection.prepareCall。

  Connection.prepareCall 返回 CallableStatement 对象，但由于 CallableStatement 是 PreparedStatement 的一个子类，因此可将 CallableStatement 对象向上转换为 PreparedStatement 对象，如下所示：

```
PreparedStatement ps_uncomp =
   Connection.prepareCall(sql_string);
```

  PreparedStatement 对象 *ps_uncomp* 保证包含未编译语句，因为仅需执行 **Connection.prepareStatement** 即可返回包含预编译语句的 PreparedStatement 对象。

具有 SAP jConnect 扩展的预准备语句

如果不关心跨驱动程序的可移植性，可编写使用 SybConnection.prepareStatement 的代码，以指定 PreparedStatement 对象是否包含预编译或未编译语句。

在这种情况下，如何编码预准备语句取决于应用程序中的大多数动态语句在会话期间可能执行多次还是仅执行几次。

如果大多数动态语句不常执行

动态 SQL 语句仅在应用程序的会话中执行一次或两次。

- 将连接属性 DYNAMIC_PREPARE 设置为 false。
- 要返回包含未编译语句的 PreparedStatement 对象，请以标准方式使用 Connection.prepareStatement：

```
PreparedStatement ps_uncomp =
   Connection.prepareStatement(sql_string);
```

- 要返回包含预编译语句的 PreparedStatement 对象，请使用 SybConnection.prepareStatement 并将 *dynamic* 设置为 true。 例如：

```
PreparedStatement ps_precomp =
   (SybConnection)conn.prepareStatement(sql_string, true);
```

如果大多数动态语句在会话中多次执行

使用 DYNAMIC_PREPARE 和 PreparedStatement 对象于会话期间在应用程序中多次执行动态语句。

- 将连接属性 DYNAMIC_PREPARE 设置为 true。
- 要返回包含预编译语句的 PreparedStatement 对象，请以标准方式使用 Connection.prepareStatement：

```
PreparedStatement ps_precomp =
Connection.prepareStatement(sql_string);
```

- 要返回包含未编译语句的 PreparedStatement 对象，则可以使用 Connection.prepareCall 或 SybConnection.prepareStatement，并将 *dynamic* 设置为 false。 例如：

```
PreparedStatement ps_uncomp =
   (SybConnection)conn.prepareStatement(sql_string, false);
```

```
PreparedStatement ps_uncomp = Connection.prepareCall(sql_string);
```

另请参见

- 可移植应用程序中的预准备语句 （第 130 页）

Connection.PrepareStatement

SAP jConnect 实现了 Connection.prepareStatement，因此可对其进行设置，以在 **PreparedStatement** 对象中返回预编译 SQL 语句或未编译 SQL 语句。

如果设置 Connection.prepareStatement 返回 PreparedStatement 对象中的预编译 SQL 语句，它会将动态 SQL 语句发送到数据库中进行预编译，并且如同直接执行 **prepare** 命令时一样，被准确地保存下来。 如果设置 Connection.prepareStatement 返回未编译 SQL 语句，它将返回 PreparedStatement 对象中的未编译 SQL 语句，而不将其发送到数据库中。

Connection.prepareStatement 返回的 SQL 语句类型由连接属性 DYNAMIC_PREPARE 确定，而且适用于整个会话。

对于 SAP 特定的应用程序，SAP jConnect 6.05 及更高版本将提供 SAP jConnect SybConnection 类下的 prepareStatement 方法。SybConnection.prepareStatement 用于指定是否对单个动态 SQL 语句进行预编译，不考虑 DYNAMIC_PREPARE 连接属性的会话级设置。

DYNAMIC_PREPARE 连接属性

DYNAMIC_PREPARE 是用于启用动态 SQL 预准备语句的布尔值连接属性。

- 如果将 DYNAMIC_PREPARE 设置为 true（缺省），则每次在会话期间调用 Connection.prepareStatement 时，都将尝试返回 PreparedStatement 对象中的预编译语句。
  在这种情况下，在执行 PreparedStatement 时，其所包含的语句已在数据库中进行预编译，拥有动态赋值的占位符，而且仅需执行该语句。
- 如果连接的 DYNAMIC_PREPARE 为 false，由 Connection.prepareStatement 返回的 PreparedStatement 对象将不包含预编译语句。
  在这种情况下，每次执行 PreparedStatement 时，其所包含的动态 SQL 语句必须发送到数据库中进行编译和执行。

在本示例中，DYNAMIC_PREPARE 为 false，可禁用动态 SQL 语句的预编译，而 **props** 则是用于指定连接属性的 **Properties** 对象。

```
...
props.put("DYNAMIC_PREPARE", "false")
Connection conn = DriverManager.getConnection(url, props);
```

当 DYNAMIC_PREPARE 为 true 时：

- 并非所有的动态语句均可在 **prepare** 命令下预编译。 SQL-92 标准对可用于 **prepare** 命令的语句做了一些限制，而且每个数据库供应商可能会有各自不同的约束。
- 如果数据库因为不能预编译和不能保存通过 Connection.prepareStatement 发送到该数据库的语句而产生错误，SAP jConnect 会捕获该错误，并返回包含未编译动态 SQL 语句的 PreparedStatement 对象。 每次执行 PreparedStatement 对象时，都会将该语句重新发送到数据库进行编译和执行。
- 在会话结束或显式关闭预编译语句的 PreparedStatement 对象之前，预编译语句将一直驻留在数据库的内存中。 PreparedStatement 对象的碎片收集不能从数据库中删除预准备语句。

一般来讲，应在最后一次使用 PreparedStatement 对象之后显式将其关闭，以避免因会话期间预准备语句在服务器内存中累积而降低性能。

SybConnection.PrepareStatement 方法

使用 SybConnection.prepareStatement 扩展方法在 PreparedStatement 对象中返回动态 SQL 语句。

如果应用程序允许对 JDBC 进行 SAP jConnect 特定的扩展：

```
PreparedStatement SybConnection.prepareStatement(String sql_stmt,
   boolean dynamic) throws SQLException
```

SybConnection.prepareStatement 会根据 *dynamic* 参数的设置，返回包含预编译或未编译 SQL 语句的 PreparedStatement 对象。 如果 *dynamic* 为 true，则 SybConnection.prepareStatement 返回具有预编译 SQL 语句的 PreparedStatement 对象。 如果 *dynamic* 为 false，则它会返回具有未编译 SQL 语句的 PreparedStatement 对象。

本示例显示了如何使用 SybConnection.prepareStatement 返回包含预编译语句的 PreparedStatement 对象：

```
PreparedStatement precomp_stmt = ((SybConnection)
conn).prepareStatement
   ("SELECT * FROM authors WHERE au_fname LIKE ?", true);
```

在本示例中，连接对象 *conn* 被转换为 SybConnection 对象，以允许使用 SybConnection.prepareStatement。 传递给 SybConnection.prepareStatement 的 SQL 字符串在数据库中预编译，即使连接属性 DYNAMIC_PREPARE 设置为 false。

如果数据库因为不能预编译通过 SybConnection.prepareStatement 发送到该数据库的语句而产生错误，则 SAP jConnect 会抛出 SQLException，并且调用无法返回 PreparedStatement 对象。 这与 Connection.prepareStatement 不同，后者会捕获 SQL 错误，并且如果产生上述错误，它会返回包含未编译语句的 PreparedStatement 对象。

ESCAPE_PROCESSING_DEFAULT 连接属性

缺省情况下，SAP jConnect 会分析提交到数据库的所有 SQL 语句，以查找有效的 JDBC 函数转义。

如果应用程序不在其 SQL 调用中使用 JDBC 函数转义，可将此连接属性设置为 false 以回避此分析过程。 这样做可以稍微改善性能。

SAP jConnect 中的优化批处理

SAP jConnect 实施内部算法以加快 PreparedStatement 对象的批处理操作速度。

此算法在 HOMOGENEOUS_BATCH 连接属性为 true 时被调用。

注意： 只有客户端应用程序连接到支持同类批处理的服务器时，此功能才可用。 SAP Adaptive Server Enterprise 15.7 引入了对同类批处理的支持。

以下示例说明使用 addBatch 和 executeBatch 方法的 PreparedStatement 批处理操作：

```
String sql = "update members set lastname = ? where member_id = ?";
prep_stmt = connection.prepareStatement(sql);
prep_stmt.setString(1, "Forrester");
prep_stmt.setLong(2, 45129);
prep_stmt.addBatch();
prep_stmt.setString(1, "Robinson");
prep_stmt.setLong(2, 45130);
prep_stmt.addBatch();
prep_stmt.setString(1, "Servo");
prep_stmt.setLong(2, 45131);
prep_stmt.addBatch();
prep_stmt.executeBatch();
```

其中，connection 表示连接实例，prep_stmt 表示预准备语句实例，而 ? 表示预准备语句的参数占位符。

大对象 (LOB) 列的同类批处理

如果 HOMOGENEOUS_BATCH 和 ENABLE_LOB_LOCATORS 属性为 true，客户端应用程序就无法将 LOB 和非 LOB 预准备语句 setter 方法混合在同一批处理中。

例如，以下语句是无效的：

```
String sql = "update members SET catchphrase = ? WHERE member_id
= ?";
prep_stmt = connection.prepareStatement(sql);
prep_stmt.setString(1, "Push the button, Frank!");
prep_stmt.setLong(2, 45129);
prep_stmt.addBatch();
Clob myclob = con.createClob();
myclob.setString(1, "Hi-keeba!");
prep_stmt.setClob(1, myclob);
prep_stmt.setLong(2, 45130);
prep_stmt.addBatch();
pstmt.executeBatch();
```

其中，catchphrase 是类型为 text 的列。 此代码失败的原因是 setString 方法和 setClob 方法用在同一列的同一批处理中。

游标性能

当在 SybCursorResultSet 类中使用 Statement.setCursorName 方法或 setFetchSize() 方法时，SAP jConnect 会在数据库中创建游标。

使用其它方法可以使 SAP jConnect 打开、读取和更新游标。

SAP jConnect 可通过将 SQL 语句发送到数据库或通过将游标命令编码为 TDS 通信协议内部的标识来创建和操纵游标。 第一种类型的游标是语言游标，第二种类型的游标是协议游标。

协议游标能够提供比语言游标更高的性能。 另外，并非所有数据库都支持语言游标。例如，SAP SQL Anywhere 数据库就不支持语言游标。

在 SAP jConnect 中，缺省条件是所有游标都是协议游标。 但是，LANGUAGE_CURSOR 连接属性允许您在数据库中使用语言命令创建和处理游标。

LANGUAGE_CURSOR 连接属性

LANGUAGE_CURSOR 在 SAP jConnect 中是一个布尔值连接属性，用于确定将游标创建为协议游标还是语言游标：

- 如果 LANGUAGE_CURSOR 为 false（缺省），会话期间创建的所有游标均为协议游标，这有助于提高性能。 SAP jConnect 通过将游标命令作为 TDS 协议中的标识发送来创建和操纵游标。
- 如果 LANGUAGE_CURSOR 为 true，会话期间创建的所有游标均为语言游标。 SAP jConnect 通过将 SQL 语句发送到数据库进行分析和编译来创建和操纵游标。

  将 LANGUAGE_CURSOR 设置为 true 无任何已知优点，但 LANGUAGE_CURSOR 为 false 时应用程序显示意外行为，在这种情况下可将其设置为 true。

迁移 SAP jConnect 应用程序

查看从 SAP jConnect 5.x、6.x、7.x 和 SAP jConnect 16.x 迁移应用程序的说明。

向 SAP jConnect 16.x 迁移应用程序

查看向 SAP jConnect 16.x 迁移应用程序的说明。

1. 如果代码使用 SAP jConnect 扩展，或者您要在代码中显式导入任何 SAP jConnect 类，则可以根据需要更改软件包导入语句。

   例如，将导入语句：

   ```
   import com.sybase.jdbc.*
   ```

   以及：

   ```
   import com.sybase.jdbc2.jdbc.*
   ```

   更改为：

   ```
   import com.sybase.jdbcx.*
   ```

2. 将 JDBC_HOME 设置为 SAP jConnect 驱动程序的顶层安装目录：

   ```
   JDBC_HOME=jConnect-16_0
   ```

3. 更改 CLASSPATH 环境变量以反映新的安装；它必须包括：

   ```
   JDBC_HOME/classes/jconn4.jar
   ```

4. 更改用于装载驱动程序的源代码，并重新编译该应用程序以使用新的驱动程序：

   ```
   Class.forName("com.sybase.jdbc4.jdbc.SybDriver");
   ```

5. 检验 SAP jConnect 16.0 驱动程序是否为 CLASSPATH 环境变量中指定的第一个 SAP jConnect 驱动程序。

另请参见

- 更改 SAP jConnect 扩展 （第 137 页）

更改 SAP jConnect 扩展

SAP jConnect 版本 4.1 及更高版本包括软件包 `com.sybase.jdbcx`，该软件包含 JDBC 的所有 SAP jConnect 扩展。

在 SAP jConnect 4.1 以前的版本中，可在 `com.sybase.jdbc` 和 `com.sybase.utils` 软件包中找到这些扩展。

com.sybase.jdbcx 软件包为不同版本的 SAP jConnect 提供一致的接口。 所有 SAP jConnect 扩展都定义为 Java 接口，从而可以在不影响使用这些接口建立的应用程序的情况下更改底层实现。

当开发使用 SAP jConnect 扩展的新应用程序时，可使用 com.sybase.jdbcx。 该软件包中的接口允许以最小的更改将应用程序升级到高于 SAP jConnect 4.0 的版本。

某些 SAP jConnect 扩展已经过更改以容纳 com.sybase.jdbcx 接口。

扩展更改示例

查看应用程序使用 SybMessageHandler 时的代码差异。

- SAP jConnect 4.0 代码：

```
import com.sybase.jdbc.SybConnection;
import com.sybase.jdbc.SybMessageHandler;
.
.
Connection con = DriverManager.getConnection(url, props);
SybConnection sybCon = (SybConnection) con;
sybCon.setMessageHandler(new ConnectionMsgHandler());
```

- SAP jConnect 6.0 代码：

```
import com.sybase.jdbcx.SybConnection;
import com.sybase.jdbcx.SybMessageHandler;
.
.
Connection con = DriverManager.getConnection(url, props);
SybConnection sybCon = (SybConnection) con;
sybCon.setSybMessageHandler(new ConnectionMsgHandler());
```

有关如何使用 SAP jConnect 扩展的更多示例，请参见随 SAP jConnect 一起提供的示例。

方法名称

查看接口中重命名的方法的列表。

表 8. 方法名称更改

实际方法名称	版本 4.0 及更低版本	版本 4.0 及更高版本
SybConnection	getCapture( )	createCapture( )
SybConnection	setMessageHandler( )	setSybMessageHandler( )
SybConnection	getMessageHandler( )	getSybMessageHandler( )
SybStatement	setMessageHandler( )	setSybMessageHandler( )
SybStatement	getMessageHandler( )	getSybMessageHandler( )

Debug 类

不再支持对 Debug 类的直接静态引用，但在 com.sybase.utils 软件包中存在此不受支持的引用。

要使用 SAP jConnect 调试功能，请使用 SybDriver 类的 getDebug 方法来获得对 Debug 类的引用。 例如：

```
import com.sybase.jdbcx.SybDriver;
import com.sybase.jdbcx.Debug;
.
.
.
SybDriver sybDriver =
    SybDriver)Class.forName
    ("com.sybase.jdbc4.jdbc.SybDriver") newInstance();
Debug sybDebug = sybDriver.getDebug();
sybDebug.debug(true, "ALL", System.out);
```

SAP jConnect Javadoc 文档中有 SAP jConnect 扩展的完整列表，该文档位于 SAP jConnect 安装目录的 docs/ 目录中。

Web 服务器网关

如果数据库服务器与 Web 服务器运行在不同的主机上，或者正在开发的 Internet 应用程序必须通过防火墙连接到安全的数据库服务器，则需要一个网关充当代理，以提供到数据库服务器的路径。

为使用 SSL 协议连接到服务器，SAP jConnect 提供了一个 Java 服务器小程序，该小程序可安装在支持 `javax.servlet` 接口的任何 Web 服务器上。 该服务器小程序使 SAP jConnect 能够通过将 Web 服务器用作网关来支持加密。

注意： SAP jConnect 支持客户端系统上的 SSL。

另请参见

- 实现自定义 SSL 套接字插件 （第 101 页）

TDS 隧道

SAP jConnect 使用 TDS 与数据库服务器通信。 从客户端到后端数据库的请求经过了整个网关，并在请求的正文中包含 TDS。

通过 HTTP 建立隧道的 TDS 可用于转发请求。 请求的标头指示请求包中包含的 TDS 的长度。

TDS 是一种面向连接的协议，但 HTTP 不是。 为支持安全性功能（如为 Internet 应用程序加密），SAP jConnect 使用 TDS 隧道服务器小程序来维护各 HTTP 请求间的逻辑连接。 服务器小程序在初始登录请求的过程中生成一个会话 ID，并且每个后续请求的标头中都包含此会话 ID。 使用会话 ID 可以标识活动会话甚至恢复会话，只要服务器小程序拥有一个使用该特定会话 ID 的开放式连接即可。

TDS 隧道服务器小程序提供的逻辑连接使 SAP jConnect 能够支持两个系统间的加密通信，例如，SAP jConnect 客户端将 `CONNECT_PROTOCOL` 连接属性设置为 https 后可连接到运行 TDS 隧道服务器小程序的 Web 服务器。

配置 SAP jConnect 和网关

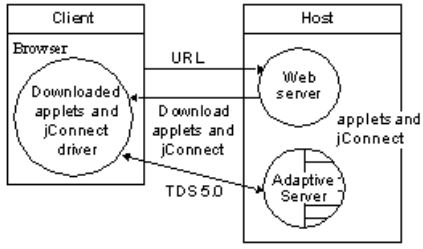
可通过多种选项设置 Web 服务器和 SAP Adaptive Server，以安装 SAP jConnect 驱动程序并将网关与 TDS 隧道服务器小程序结合使用。

Web 服务器和 SAP Adaptive Server 在同一主机上

在两层配置中，Web 服务器和 SAP Adaptive Server 安装在同一主机上。

- 在 Web 服务器主机上安装 SAP jConnect。
- 不需要任何网关。

图 3： Web 服务器和 SAP Adaptive Server 在同一主机上



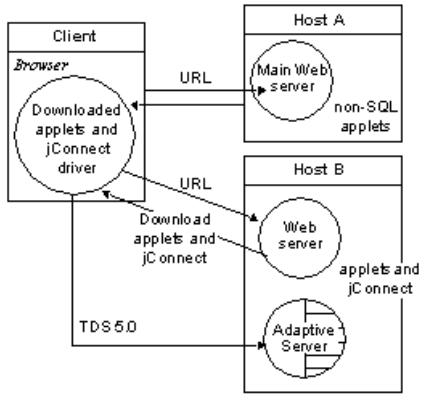
专用 JDBC Web 服务器和 SAP Adaptive Server 在同一主机上

在单个主机配置中，主 Web 服务器在一台单独的主机上。

另一台主机由专用于 SAP Adaptive Server 访问的 Web 服务器和 SAP Adaptive Server 共享。 来自主服务器的链接发送请求，要求 SQL 访问专用 Web 服务器。

- 在第二台 (SAP Adaptive Server) 主机上安装 SAP jConnect。
- 第二台 (SAP Adaptive Server) 主机不需要网关。

图 4： 专用 JDBC Web 服务器和 SAP Adaptive Server 在同一主机上

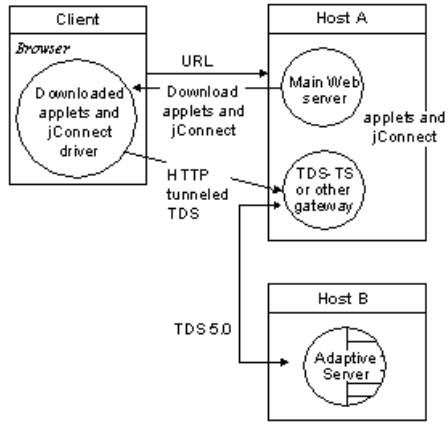


Web 服务器和 SAP Adaptive Server 在不同的主机上

在三层配置中， SAP Adaptive Server 和 Web 服务器位于不同主机。 SAP jConnect 需要使用网关来充当 SAP Adaptive Server 的代理。

- 在 Web 服务器主机上安装 SAP jConnect。
- 安装 TDS 隧道服务器小程序或其它网关。

图 5： Web 服务器和 SAP Adaptive Server 在不同的主机上



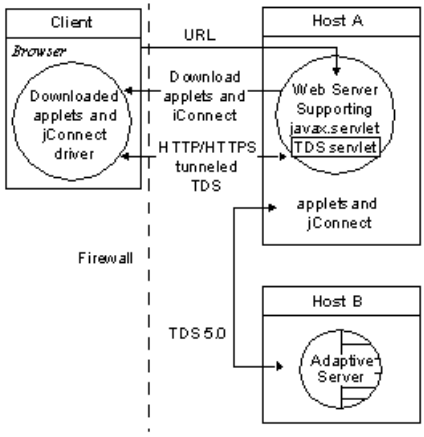
通过防火墙连接到服务器

连接到受防火墙保护的服务器

必须使用带有 TDS 隧道服务器小程序的 Web 服务器来支持在 Internet 上传输数据库请求响应。

- 在 Web 服务器主机上安装 SAP jConnect。
- 需要一台支持 **javax.servlet** 接口的 Web 服务器。

图 6： 通过防火墙连接到服务器



使用要求

查看 Web 服务器网关的使用要求。

查看 Index.html 文件

使用 Web 浏览器查看 SAP jConnect 安装目录中的 `index.html` 文件。 `index.html` 提供 SAP jConnect 文档和示例代码的链接。

如果在安装有 SAP jConnect 的同一台计算机上使用 Netscape，请确保浏览器无权访问 CLASSPATH 环境变量。 请参见《SAP jConnect for JDBC 安装指南和发行公告》中的“使用 Netscape 时设置 CLASSPATH 的限制”。

1. 打开 Web 浏览器。
2. 输入与安装相匹配的 URL。 例如，如果浏览器和 Web 服务器运行在同一主机上，则输入：

```
http://localhost:8000/index.html
```

如果浏览器和 Web 服务器运行在不同的主机上，则输入：

```
http://host:port/index.html
```

其中，*host* 是运行 Web 服务器的主机的名称，*port* 是监听端口。

运行示例小程序

在 SAP jConnect 中执行示例小程序的说明。

1. 单击“**运行示例 JDBC 小程序**”(Run Sample JDBC Applets)。
2. 在“可执行的示例”(Executable Samples) 表中，找到 `Isql.java` 并单击位于行末的“**运行**”(Run)。

示例 `Isql.java` 小程序提示对示例数据库执行简单查询并显示结果。 小程序显示缺省的 SAP Adaptive Server 主机名、端口号、用户名 (*guest*)、口令 (*sybase*)、数据库和查询。 使用这些缺省值，小程序连接到 SAP jConnect 示例数据库，并在您单击“**执行**”(Go) 后返回结果。

修改小程序的屏幕维度

在 UNIX 平台上，如果小程序未能按预期显示，则可以修改小程序的屏幕维度。

1. 使用文本编辑器打开 `$JDBC_HOME/sample2/gateway.html`。
2. 将第 7 行的高度参数更改为 650。 可尝试使用不同的高度设置。
3. 在浏览器上重新装载网页。

TDS 隧道服务器小程序

要使用 TDS 隧道服务器小程序，需要一台支持 `javax.servlet` 接口的 Web 服务器（如 Oracle Corporation Java Web 服务器）。

安装 Web 服务器时，将 SAP jConnect TDS 隧道服务器小程序添加到活动服务器小程序列表中。 还可以通过设置服务器小程序参数来定义连接超时和最大包大小。

使用 TDS 隧道服务器小程序时，从客户端到后端服务器的请求隧道网关，这样的请求包括 **GET** 或 **POST** 命令、TDS 会话 ID（在初始请求后）、后端地址和请求状态。

TDS 在请求正文中。 两个标头字段表示 TDS 流的长度和网关指派的会话 ID。

当客户端发送请求时，Content-Length 标头字段表示 TDS 内容的大小，请求命令为 **POST**。 如果由于客户端正在检索来自服务器的响应数据的下一部分内容或正在关闭连接，从而造成请求中没有任何 TDS 数据，此时请求命令为 **GET**。

以下示例演示了如何使用 TDS 隧道 HTTPS 协议在客户端和 HTTPS 网关之间传递信息；该示例显示了一个到端口号为 1234 且名为“DBSERVER”的后端服务器的连接。

- **客户端到网关的登录请求** – 无需会话 ID。
  - 查询 – POST/tds?ServerHost=dbserver&ServerPort=1234& Operation=more HTTP/1.0
  - 标头 – 内容长度： 605
  - 内容 (TDS) – 登录请求
- **网关到客户端** – 标头包含由 TDS 服务器小程序指派的会话 ID。
  - 查询 – 200 SUCCESS HTTP/1.0
  - 标头 – 内容长度： 210 TDS-会话： TDS00245817298274292
  - 内容 (TDS) – 登录确认 EED
- **客户端到网关** – 所有后续请求的标头都包含会话 ID。
  - 查询 – POST/tds?TDS-Session=TDS00245817298274292&Operation=more HTTP/1.0
  - 标头 – 内容长度： 32
  - 内容 (TDS) – 查询“SELECT * from authors”
- **网关到客户端** – 所有后续响应的标头都包含会话 ID
  - 查询 – 200 SUCCESS HTTP/1.0
  - 标头 – 内容长度： 2048 TDS-会话： TDS00245817298274292
  - 内容 (TDS) – 行格式与某些来自查询响应的行

检查要求

要使用 SAP jConnect TDS-隧道服务器小程序，必须具有支持 `javax.servlet` 接口的 Web 服务器。

要安装该服务器，请遵循 Java 服务器小程序提供的操作说明。

安装和设置服务器小程序参数

SAP jConnect 安装包括 `classes` 目录下的 `gateway2` 子目录。 该子目录包含 TDS 隧道服务器小程序所需的文件。

1. 将 SAP jConnect **gateway** 包复制到 Web 服务器的 `servlets` 目录下的 `gateway2` 子目录中。

   复制好服务器小程序后，按照 Web 服务器操作说明激活服务器小程序。
2. 将服务器小程序添加到 Web 服务器，可设置以下可选参数以自定义性能：
   - **SkipDoneProc** [true|false] – 数据库通常会在查询执行期间执行中间处理步骤时返回行计数信息。 通常，客户端应用程序会忽略此数据。 如果将 **SkipDoneProc** 设置为 true，服务器小程序会随即从响应中删除此额外信息，这将减少网络使用和客户端上的处理要求。 这在使用 HTTPS/SSL 时尤其有效，因为不需要的数据不会被加密/解密。

- **TdsResponseSize** – 为隧道的 HTTPS 设置最大 TDS 包大小。 如果只有几个用户有大量数据，较大的 **TdsResponseSize** 会更加有效。 如果有许多用户都执行较小的事务，请使用较小的 **TdsResponseSize**。
- **TdsSessionIdleTimeout** – 定义在服务器连接自动关闭前该连接能够维持空闲状态的时间量（以毫秒为单位）。 `TdsSessionIdleTimeout` 缺省值为 600,000（10 分钟）。

  如果交互式客户端程序可能长时间处于空闲状态而您又不希望中断该连接，则增加 **TdsSessionIdleTimeout**。

  还可从 SAP jConnect 客户端使用 `SESSION_TIMEOUT` 连接属性设置连接超时值。 这在存在可能长时间处于空闲状态的特定应用程序时很有用。 在这种情况下，可通过 `SESSION_TIMEOUT` 连接属性为这些连接设置更长的超时值，而不为服务器小程序进行设置。
- **Debug** – 打开调试程序。

另请参见

- 使用 SAP jConnect 进行调试 （第 117 页）

调用服务器小程序

SAP jConnect 根据 `proxy` 连接属性的路径扩展来确定何时使用安装有 TDS 隧道服务器小程序的网关。

SAP jConnect 能够识别 `proxy` 的服务器小程序路径扩展，并调用指定网关上的服务器小程序。

使用下面的格式定义连接 URL：

```
http://host:port/TDS-servlet-path
```

SAP jConnect 通过调用 Web 服务器上的 TDS 隧道服务器小程序使 TDS 隧道 HTTP。 服务器小程序的路径必须是您在 Web 服务器的服务器小程序别名列表中定义的路径。

跟踪活动的 TDS 会话

查看活动 TDS 会话的相关信息，其中包括每个会话的服务器连接。

使用 Web 浏览器打开管理 URL：

```
http://host:port/TDS-servlet-path?Operation=list
```

例如，如果服务器为“myserver”，TDS 服务器小程序路径为 `/tds`，则输入：

```
http://myserver:8080/tds?Operation=list
```

这将显示活动 TDS 会话的列表。 单击会话可查看更多信息，包括服务器连接。

终止 TDS 会话

要终止 TDS 会话，请使用在任意活动的 TDS 会话中定义的 URL。

从第一页的会话列表中选择一个活动会话，然后单击**“终止会话”** (Terminate This Session)。

恢复 TDS 会话

当您指定 SESSION_ID 后，SAP jConnect 将跳过协议的登录阶段，并使用指定的会话 ID 恢复与网关的连接。

设置 SESSION_ID 连接属性，以便在必要时恢复现有的开放式连接。

如果指定的会话 ID 不在服务器小程序中，当您首次尝试使用连接时，SAP jConnect 会抛出一个 SQL 异常。

SAP jConnect 示例程序

查看 SAP jConnect 示例程序。

运行 IsqlApp

IsqlApp 允许从命令行发出 **isql** 命令，而且允许运行 SAP jConnect 示例程序。

IsqlApp 的语法：

```
IsqlApp [-U username]
   [-P password]
   [-S servername]
   [-G gateway]
   [-p {http|https}]
   [-D debug_class_list]
   [-v]
   [-I input_command_file]
   [-c command_terminator]
   [-C charset]
   [-L language]
   [-K service_principal_name]
   [-F JAAS_login_config_file_path]
   [-T sessionID]
   [-V <version {2,3,4,5}>]
```

参数	说明
-U	用于连接到服务器的登录 ID。
-P	指定的登录 ID 的口令。
-S	要连接的服务器的名称。
-G	网关地址。 对于 HTTP，URL 是：http://*host:port*。 要使用 HTTPS，URL 应为 https://*host:port*/*servlet_alias*。
-p	指定是要使用 HTTP 还是 HTTPS。
-D	为所有类或指定的类打开调试程序，类之间用逗号分隔。 例如： -D ALL - 显示所有类的调试输出。 -D SybConnection, Tds - 仅显示 `SybConnection` 和 `Tds` 类的调试输出。
-v	打开详细输出以显示或打印输出。

参数	说明
-I	让 **IsqlApp** 接受来自文件而不是来自键盘的命令。 在此参数之后，指定用于 **IsqlApp** 输入的文件的名称。 文件必须包含命令终止符（缺省为“go”）。
-c	允许指定一个关键字（如“go”），当独占一行输入该关键字时可终止命令。 这使得在使用终止符关键字之前可以输入多行命令。 如果不指定命令终止符，每个新行都会终止命令。
-C	为通过 TDS 传递的字符串指定字符集。 如果不指定字符集，**IsqlApp** 将使用服务器的缺省字符集。
-L	为 SAP jConnect 消息和从服务器返回的错误消息指定显示语言。
-K	表示用户想要通过 Kerberos 登录到 SAP Adaptive Server。 此参数设置服务主体名称。 例如： `-K myASE` 其中 **myASE** 是服务器的服务主体名称。
-F	指定 JAAS 登录配置文件的路径。 如果使用 -K 选项，则必须设置此属性。 例如： `-F /foo/bar/exampleLogin.conf` 请参见 SAP jConnect 安装的 `sample2` 目录中的 `ConnectKerberos.java` 示例。
-T	如果设置此参数，SAP jConnect 将认为应用程序正在尝试恢复由 TDS 隧道网关保持打开的现有 TDS 会话的通信。 SAP jConnect 将跳过登录协商并将所有来自应用程序的请求转发到指定的会话 ID。
-V	启用特定于版本的特性。

必须在每个选项标志后面输入一个空格。

要获取命令行选项的完整描述，请输入：

```
java IsqlApp -help
```

以下示例显示了如何通过端口“3756”连接到主机“myserver”上的数据库，并运行名为“myscript”的 **isql** 脚本：

```
java IsqlApp -U sa -P sapassword
   -S jdbc:sybase:Tds:myserver:3756
   -I $JDBC_HOME/sp/myscript -c run
```

提供了对 **isql** 命令进行 GUI 访问的小程序，如下所示： `$JDBC_HOME/sample2/gateway.html` (UNIX) `%JDBC_HOME%\sample2\gateway.html` (Windows)。

另请参见

- 安全性 （第 101 页）
- JCONNECT_VERSION 连接属性 （第 4 页）

SAP jConnect 示例程序和代码

SAP jConnect 包含多个示例程序，这些程序旨在帮助您了解 SAP jConnect 如何使用各种 JDBC 类和方法。

示例应用程序

安装 SAP jConnect 时，也可安装包含源代码的示例程序，这样可以查看 SAP jConnect 是如何实现各种 JDBC 类和方法的。

有关安装示例程序的完整说明，请参见《SAP jConnect for JDBC 安装指南》。

注意： SAP jConnect 示例程序仅用于示范。

示例程序安装在 SAP jConnect 安装目录的 `sample2` 子目录下。 `sample2` 子目录中的文件 `index.html` 包含可用示例的完整列表以及对每个示例的说明。 `index.html` 还允许以小程序形式查看并运行示例程序。

运行示例小程序

可在 Web 浏览器中以小程序形式运行某些示例程序，这样便可在检查输出结果时查看源代码。

要以小程序形式运行示例程序，请在 Web 浏览器上输入 `http://localhost:8000/sample2/index.html` 以启动 Web 服务器网关。

与 SAP SQL Anywhere 一起运行示例程序

所有示例程序均与 SAP Adaptive Server 兼容，但只有有限数目的示例程序与 SAP SQL Anywhere 兼容。

有关与 SAP SQL Anywhere 兼容的示例程序的当前列表，请参见 `sample2` 子目录中的 `index.html`。

要运行可用于 SAP SQL Anywhere 的示例程序，必须在 SAP SQL Anywhere 服务器上安装 `pubs2_any.sql` 脚本。 此脚本位于 `sample2` 子目录中。

在 Windows 系统中，进入 DOS 命令窗口并输入：

```
java IsqlApp -U dba -P password
 -S jdbc:sybase:Tds:[hostname]:[port]
 -I %JDBC_HOME%\sample2\pubs2_any.sql -c go
```

在 UNIX 系统中，输入：

```
java IsqlApp -U dba -P password
 -S jdbc:sybase:Tds:[hostname]:[port]
 -I $JDBC_HOME/sample2/pubs2_any.sql -c go
```

示例代码

查看说明如何调用 SAP jConnect 驱动程序、建立连接、发出 SQL 语句，以及处理结果的示例代码。

```
import java.io.*;
 import java.sql.*;

   public class SampleCode
   {
       public static void main(String args[])
       {
           try
           {
           /*
            * Open the connection. May throw a SQLException.
            */
              DriverManager.registerDriver(

              (Driver) Class.forName(

               "com.sybase.jdbc4.jdbc.SybDriver").newInstance());

               Connection con = DriverManager.getConnection(
                  "jdbc:sybase:Tds:myserver:3767", "sa", "");
           /*
            * Create a statement object, the container for the SQL
            * statement. May throw a SQLException.
            */
              Statement stmt = con.createStatement();
           /*
            * Create a result set object by executing the query.
            * May throw a SQLException.
            */
              ResultSet rs = stmt.executeQuery("Select 1");
           /*
            * Process the result set.
            */

              if (rs.next())
              {
                 int value = rs.getInt(1);
                 System.out.println("Fetched value " + value);
              }

            rs.close()

            stmt.close()

            con.close()
          }//end try
      /*
       * Exception handling.
       */
```

```
            catch (SQLException sqe)
            {
                System.out.println("Unexpected exception : " +
                            sqe.toString() + ", sqlstate = " +
                                sqe.getSQLState());
                System.exit(1);
            }//end catch
```

```
            catch (Exception e)
            {
                e.printStackTrace();
```

```
                System.exit(1);
            }//end catch

            System.exit(0);
        }
    }
```

SQL 异常与警告消息

查看在使用 SAP jConnect 时可能会遇到的 SQL 异常和警告消息。

SQL 状态	消息/说明/操作
010AF	`SEVERE WARNING: An assertion has failed, please use dev-classes todetermine the source of this serious bug. Message = _____.` 操作： 使用 devclasses 调试类确定显示此消息的原因并向 SAP 技术支持部门报告此问题。
010CP	`AutoCommit option has changed to true. All pending state-ments on this transaction (if any) are committed.`
010DF	`Attempt to set database at login failed. Error mes-sage:_____.` 说明： SAP jConnect 无法连接到在连接 URL 中指定的数据库。 操作： 确保 URL 中的数据库名称正确无误。 此外，如果连接到 SAP SQL Anywhere，请使用 `SERVICENAME` 连接属性指定数据库。
010DP	`Duplicate connection property _____ ignored.` 说明： 某个连接属性被定义了两次。 可能是在驱动程序连接属性列表中用不同的大小写形式对该连接属性定义了两次，例如“password”和“PASSWORD”。 SAP jConnect 不会区分名称相同但大小写不同的属性名。 也可以同时在连接属性列表和 URL 中定义连接属性。 在这种情况下，连接属性列表中的属性值优先。 操作： 确保应用程序只定义一次连接属性。 不过，或许您希望应用程序利用属性列表中定义的连接属性优先于 URL 中定义的属性这一特性。 这时，可以安全地忽略此警告。
010HA	`The server denied your request to use the high-availability feature. Please reconfigure your database, or do not request a high-availability session.` 操作： 重新配置服务器使其支持高可用性故障切换，或者不要将 `REQUEST_HA_SESSION` 设置为 true
010HD	`SAP Adaptive Server high-availability failover is not sup-ported by this type of database server.` 操作： 应当只连接到支持高可用性故障切换的数据库服务器。

SQL 状态	消息/说明/操作
010HN	`The client did not specify a SERVICE_PRINCIPAL_NAME Connection property. Therefore, jConnect is using the hostname of _____ as the service principal name` 操作：通过连接属性显式指定服务主体名称。
010HT	`Hostname property truncated, maximum length is 30.` 操作：无需任何操作，这只是提醒您 SAP jConnect 将把名称截断到 30 个字节。然而，如果希望避免此警告，应将 HOSTNAME 设置为长度小于或等于 30 个字节的字符串。
010KF	`The server rejected your Kerberos login attempt. Most likely, this was because of a Generic Security Services (GSS)exception. Please check your Kerberos environment and configuration.` 操作：检查 Kerberos 环境，确保已向 KDC 进行正确验证。 有关详细信息，请参见“安全性”（第 101 页）。
010MX	`Metadata accessor information was not found on this database. Please install the required tables as mentioned in the jConnect documentation. Error encountered while attempting to retrieve metadata information: _____.` 说明：服务器可能不具有返回元数据信息所需的存储过程。 操作：确保服务器上安装有用于提供元数据的存储过程。 请参见《SAP jConnect for JDBC 安装指南》中的“安装存储过程”。
010P4	`An output parameter was received and ignored.` 说明：执行的查询返回一个输出参数，但应用程序的结果处理代码未读取该参数，因此将其忽略。 操作：如果应用程序需要输出参数数据，应重写该应用程序以便能获取数据。 这可能需要使用 **CallableStatement** 来执行查询，并添加对 **registerOutputParameter** 和 **getXXX** 的调用。 也可以将 DISABLE_UNPROCESSED_PARAM_WARNINGS 连接属性设置为 true，从而禁止 SAP jConnect 返回此警告，这样做或许还可以提高性能。
010P6	`A row was received and ignored.` 说明：正在处理的结果集中出现类型为 0xD1 的意外对象，该对象已被忽略。 操作：检查生成结果集的查询，如有需要予以更正。

SQL 状态	消息/说明/操作
010PF	`One or more jars specified in the PRELOAD_JARS connection property could not be loaded.` 说明：如果在使用 DynamicClassLoader 时将 PRELOAD_JARS 连接属性设置为以逗号分隔的 .jar 文件名列表，则会出现此问题。当 DynamicClassLoader 打开与要装载的类所在服务器的连接时，会尝试预装载此连接属性中提到的所有 .jar 文件。如果服务器上不存在指定的一个或多个 .jar 文件名，则会出现上述错误消息。 操作：验证在应用程序的 PRELOAD_JARS 连接属性中提到的所有 .jar 文件是否均位于服务器上且均可访问。
010PO	`Property LITERAL_PARAM has been reset to false because DYNAMIC_PREPARE was set to true.` 说明：要使用预编译的动态语句，应允许向这些语句发送参数（在语句带参数的情况下）。将 LITERAL_PARAMS 设置为 true 会强制以送往服务器的 SQL 中的文本值的形式发送所有参数。所以不能将这两个属性都设置为 true。 操作：为了避免出现此警告，当您想使用动态 SQL 时，不要将 LITERAL_PARAMS 设置为 true。请参见“对动态 SQL 中的预准备语句的性能调优”（第 129 页）。
010RC	`The requested ResultSet type and concurrency is not supported. They have been converted.` 说明：有关 SAP jConnect 中可用结果集类型和并发的详细信息，请参见“对结果集使用游标”（第 49 页） 操作：请求受支持的结果集类型和并发组合。
010SJ	`Metadata accessor information was not found on this database. Please install the required tables as mentioned in the jConnect documentation.` 说明：服务器上没有配置元数据信息。 操作：如果应用程序需要元数据，请安装 jConnect 中附带的用于返回元数据的存储过程（请参见《jConnect for JDBC 安装指南》中的“安装存储过程”）。如果不需要元数据，请将 USE_METADATA 属性设置为 false
010SK	`Database cannot set connection option _____.` 说明：连接的数据库不支持应用程序尝试的操作。 操作：升级数据库，或者确保安装了最新版本的元数据信息。

SQL 状态	消息/说明/操作
010SL	`Out-of-date metadata accessor information was found on this database. Ask your database administrator to load the latest scripts.` 说明：服务器上的元数据信息已过时，需要更新。 操作：安装随 SAP jConnect 提供的用于返回元数据的存储过程。请参见《SAP jConnect for JDBC 安装指南》中的“安装存储过程”。
010SM	`This database does not support the initial proposed set of capabilities, retrying.` 说明：SAP Adaptive Server Enterprise 11.9.2 版及更低版本存在一个问题，该问题使这些版本在服务器没有客户端所请求的功能时拒绝客户端登录。此警告说明 SAP jConnect 检测到这种情况，且正以服务器可接受的最多功能数重试该连接。当 SAP jConnect 遇到这项错误时，它会进行两次服务器连接尝试。 操作：客户端可以安全地忽略此警告，但若想消除此警告并确保 SAP jConnect 只进行一次连接尝试，客户端可以将 `ELIMINATE_010SM` 连接属性设置为 true。在连接到 SAP Adaptive Server 12.0 及更高版本时，请勿将此属性设置为 true。
010SN	`Permission to write to file was denied. File: _____. Error message: _____.` 说明：因 VM 中的安全冲突，对 `PROTOCOL_CAPTURE` 连接属性中指定的文件的写入权限被拒绝。当一个小程序尝试写入指定文件时，会出现此消息。 操作：如果要通过小程序写入文件，必须确保该小程序可以访问目标文件系统。
010SP	`File could not be opened for writing. File: _____. Error message: _____.` 操作：确保文件名正确无误且文件可写。
010SQ	`The connection or login was refused, retrying connection with the host/port address.` 说明：`CONNECTION_FAILOVER` 连接属性被设置为 true，SAP jConnect 无法连接到要连接的服务器列表中的某个数据库服务器。因此，SAP jConnect 现在尝试连接到列表中的下一个服务器。 操作：只要 SAP jConnect 能连接到另一数据库服务器，就无需任何操作。不过，应确定 SAP jConnect 为何无法连接到导致连接警告的特定服务器。

SQL 状态	消息/说明/操作
010TP	`The connection's initial character set,_____, could not be converted by the server. The server's proposed character set,_____, will be used, with conversions performed by jConnect.` 说明：服务器无法使用最初由 SAP jConnect 请求的字符集，已经用不同的字符集响应。 SAP jConnect 接受此更改，并执行必要的字符集转换。 本消息只用来提供信息，无其它影响。 操作：为避免出现此消息，可将 CHARSET 连接属性设置为服务器支持的字符集。
010TQ	`jConnect could not determine the server's default character set. This is likely because of a metadata problem. Please install the required tables as mentioned in the jConnect documentation. The connection is defaulting to the ascii_7 character set, which can handle only characters in the range from 0x00 through 0x7F.` 说明：当发生这种情况时，只有前 127 个 ASCII 字符能够确保得以正确转换。 因此，SAP jConnect 恢复为 7 位 ASCII。 本消息只用来提供信息，无其它影响。 操作：安装随 SAP jConnect 提供的用于返回元数据的存储过程。 请参见《SAP jConnect for JDBC 安装指南》中的“安装存储过程”。
010UF	`Attempt to execute use database command failed. Error message:_____.` 说明：SAP jConnect 无法连接到在连接 URL 中指定的数据库。 两种可能的原因是： • URL 中输入的名称有误。 • USE_METADATA 为 true（缺省设置），但未安装用于返回元数据的存储过程。 结果，SAP jConnect 尝试对 URL 中的数据库执行 **use** *database* 命令，但命令失败。 这可能是由于您尝试访问不支持 **use***databse* 命令的 SQL Anywhere 数据库。 操作：确保 URL 中的数据库名称正确无误。 确保服务器上安装了用于返回元数据的存储过程。 请参见《SAP jConnect for JDBC 安装指南》中的“安装存储过程”以及《SAP jConnect for JDBC 发行公告》。 若要尝试访问 SQL Anywhere 数据库，则不要在 URL 中指定数据库名称，或者将 USE_METADATA 设置为 false。
010UP	`Unrecognized connection property _____ ignored.` 说明：您试图在 URL 中设置一个 jConnect 目前无法识别的连接属性。jConnect 将忽略该无法识别的属性。 操作：检查应用程序中的 URL 定义，确保其只引用有效的 SAP jConnect 驱动程序连接属性。

SQL 状态	消息/说明/操作
0100V	`The version of TDS protocol being used is too old. Version: _____.` 说明： SAP jConnect 要求 5.0 或更高版本。 操作： 使用支持所需 TDS 版本的服务器。 请参见《SAP jConnect 安装指南》中的“系统要求”。
01S07	`Adaptive Server may round or truncate nanosecond values.` 说明： 遇到一个精度高于 1/300 秒的时间值。 由于 SAP Adaptive Server 不支持如此高的精度，jConnect 拒绝了这个值。 操作： 确保时间值的精度不高于 1/300 秒。
01S08	`This connection has been enlisted in a Global transaction. All pending statements on the current local transaction (if any) have been rolled back.` 说明： SAP jConnect 发出回退命令以清除当前所有本地事务。 如果在发出 `XAResource.start()` 后征用全局事务，则会出现这种情况。 操作： 在发出 `XAResource.start()` 方法前提交或回退活动的本地事务。
01S09	`The local transaction method _____ cannot be used while a global transaction is active on this connection.` 说明： 对连接调用 commit() 方法就属于一种本地操作。 其它不能使用的操作有：`rollback()`、`rollback(Savepoint)`、`setSavepoint()`、`setSavepoint(String)`、`releaseSavepoint(Savepoint)` 和 `setAutoCommit()`。 操作： 将本地事务与全局事务分开。 先完成所有本地事务及其操作，然后再启动全局事务。
01S10	`The local transaction method _____ cannot be used on a pre-System 12 XAConnection.` 说明： 您使用的本地事务方法对低于 12 的 SAP SQL Anywhere 版本无效。 操作： 请不要使用该方法。
01S11	`WARNING: Your data might be truncated.` 说明： 用户指定的流或 LOB 长度大于 **ResultSet.updateXXX** 方法中的限制 (Integer.MAX_VALUE)。 操作： 确保长度在限制范围内。

SQL 状态	消息/说明/操作
01S12	`Unable to continue with HOMOGENEOUS_BATCH protocol, falling back to normal batching.` 说明：如果将 DYNAMIC_PREPARE 设置为 false，SAP Adaptive Server 不会发送参数元数据。如果 HOMOGENEOUS_BATCH 设置为 true，SAP jConnect 需要使用此信息进行优化。因此，jConnect 将恢复为正常批处理。 操作：仅以预编译的动态 SQL 预准备语句（将 DYNAMIC_PREPARE 设置为 true）使用优化的批处理（将 HOMOGENEOUS_BATCH 设置为 true）。

01S13	`Connected Adaptive Server server does not support the set option 'logbulkcopy' needed for logging BCP.`
	`Falling back to normal bulk load without logging which is equivalent to setting ENABLE_BULK_LOAD=BCP.`
	说明：如果 SAP Adaptive Server 版本低于 15.7 ESD #1 并且不支持带记录功能的批量装载，则 SAP jConnect 将恢复为正常批量处理。
	操作：仅以 SAP Adaptive Server 15.7 ESD #1 或更高版本使用 ENABLE_BULK_LOAD=LOG_BCP 设置。

01ZZZ	错误代码 4022： `Password has expired Please set the NEWPASSWORD property with the new password or use sp_password to change passwords.` 操作：重置用于连接 SAP Adaptive Server 的口令。
JZ001	`User name property ‘_____’ too long. Maximum length is 30.` 操作：不要超出 30 字节的最大长度限制。
JZ002	`Password property ‘_____’ too long. Maximum length is 30.` 操作：不要超出 30 字节的最大长度限制。
JZ003	`Incorrect URL format. URL: _____.` 操作：检查 URL 格式。请参见“URL 连接属性参数”（第 31 页）。 如果使用的是 PROXY 连接属性并且 PROXY 属性的格式有误，则在尝试连接时会发生 JZ003 异常。 • 该级联代理的 PROXY 格式为：*ip_address*:*port_number* • TDS 隧道服务器小程序的 PROXY 格式为：http[s]://*host*:*port*/*tunneling_servlet_alias*
JZ004	`User name property missing in DriverManager.getConnection(..., Properties)` 操作：提供所需的用户属性。

SQL 状态	消息/说明/操作
JZ006	`Caught IOException: _____.` 说明：从低层检测到意外的 I/O 错误。当捕获到这种 I/O 异常时，会使用 `ERR_IO_EXCEPTION JZ006` sqlstate 将其作为 SQL 异常再次抛出。这些错误通常是由网络通信问题引起的。如果 I/O 异常导致数据库连接被关闭，则 SAP jConnect 会将 JZ0C1 异常链接到 JZ006。客户端应用程序可以查找链中的 JZ0C1 异常，以查看该连接是否仍然可用。 操作：检查原始 I/O 异常消息的文本，并从该处继续。
JZ008	`Invalid column index value _____.` 说明：请求的列索引值小于 1 或大于最大可用值。 操作：验证对 `getXXX` 方法的调用和原始查询的文本，或者调用 **rs.next**。
JZ009	`Error encountered in conversion. Error message: _____.` 说明：可能的原因是： • 尝试在两种不兼容的数据类型之间进行转换，例如 `date` 到 `int`。 • 试图将包含非数值字符的字符串转换为数值类型。 • 存在格式错误，例如 `time/date` 字符串格式有误。 操作：确保 JDBC 规范支持尝试进行的类型转换。确保字符串格式正确无误。如果字符串包含非数值字符，不要试图将其转换为数值类型。
JZ00A	`Invalid precision and scale specified for a numeric value.` 说明：使用 `setBigDecimal` 方法时，将 BigDecimal 值设置为以下几种值：小于 1 的精度值、负标度值、小于标度值的精度，或大于 127 的精度值。 操作：检查查询并予以更正，以指定合法的精度/标度值。
JZ00B	`Numeric overflow.` 说明：您试图将 **BigInteger** 作为 TDS 数值发送，而该值过大；或者您试图将 Java `long` 作为 `int` 发送，而该值过大。 操作：不能在 SAP jConnect 中存储这些值。对于 `long`，请考虑使用 SAP jConnect 数值类型。尚无法解决 **Bignum** 的问题。
JZ00C	`The precision and scale specified cannot accommodate numeric value _____.` 说明：在使用 setBigDecimal 方法时，BigDecimal 值的精度或标度超出指定范围。 操作：确保指定的精度和标度与 BigDecimal 值相匹配。

SQL 状态	消息/说明/操作
JZ00E	`Attempt to call execute() or executeUpdate() for a statement where setCursorName()has been called.` 操作：请单独使用一条语句删除或更新游标。请参见“对结果集使用游标”（第49页）。
JZ00F	`Cursor name has already been set by setCursorName().` 操作：不要为同一语句设置两次游标名。关闭当前游标语句的结果集。
JZ00G	`No column values were set for this row update.` 说明：您试图更新的行中列值并未发生更改。 操作：在调用 **updateRow** 之前调用 **updateXX** 方法。
JZ00H	`The result set is not updatable. Use Statement.setResultSetConcurrencyType().` 操作：要将结果集从只读改为可更新，请使用 `Statement.setResultSetConcurrencyType` 方法，或将 **for update** 子句添加到 SQL **select** 语句中。
JZ00I	`Invalid scale. The specified scale must be >=0.` 说明：标度值必须大于零。 操作：确保标度值非负。
JZ00L	`Login failed. Examine the SQLWarnings chained to this exception for the reason(s).` 操作：查看消息文本；根据给出的登录失败原因继续操作。

JZ00M	`Login timed out. Check that your database server is running on the host and port number you specified.`
	`Also check the database server for other conditions (such as a full tempdb) that might be causing it to hang.`
	操作：按照错误消息中提供的建议进行操作。

JZ010	`Unable to deserialize an Object value. Error text: _____.` 操作：确保数据库中的 Java 对象实现 `Serializable` 接口并位于本地 CLASSPATH 变量中。
JZ011	`Number format exception encountered while parsing numeric connection property _____.` 说明：为数值连接属性指定了非整数值。 操作：为连接属性指定整数值。

SQL 状态	消息/说明/操作
JZ012	`Internal Error. Please report it to SAP technical support. Wrong access type for connection property _____.` 操作：与 SAP 技术支持部门联系。
JZ013	`Error obtaining JNDI entry: _____.` 操作：更正 JNDI URL，或在目录服务中创建一个新条目。
JZ014	`You may not setTransactionIsolation(Connection.TRANSACTION_NONE). This level cannot be set; it can only be returned by a server.` 操作：检查调用 `Connection.setTransactionIsolation` 的应用程序代码，并验证传递给该方法的值。

JZ015	`Illegal value set for the GSSMANAGER_CLASS connection property.`
	`The property value must be a String or an Object that extends org.ietf.jgss.GSSManager.`
	操作：检查为 `GSSMANAGER_CLASS` 属性设置的值。

JZ017	`Savepoint is not valid.` 说明：为回退或释放指定的保存点不存在。 操作：检查查询并予以更正，以指定存在的保存点。
JZ018	`This method can not be applied to this type of savepoint.` 说明：`getSavepointId()` 方法不适用于命名保存点（没有 ID），`getSavepointName()` 方法不适用于未命名保存点（没有名称）。 操作：检查查询并予以更正。
JZ019	`Error obtaining SERVERNAME : _____.` 说明：用 **jdbc::jndi:file** 设置的 URL 未指定 `sql.ini` 文件 (Windows) 或 `interfaces` 文件 (UNIX) 或者服务器名。 操作：检查 URL 命令并予以更正。
JZ021	`The Specified _____ file not found.` 说明：未找到在连接 URL 中指定的 `sql.ini` 文件 (Windows) 或 `interfaces` 文件 (UNIX)。 操作：检查连接 URL 并予以更正。

SQL 状态	消息/说明/操作
JZ022	`The Specified _____ file has an unknown format.` 说明：`sql.ini` 文件 (Windows) 或 `interfaces` 文件 (UNIX) 中的连接 URL 字符串格式不正确。 操作：检查连接 URL 字符串并予以更正。
JZ024	`The Specified Server : _____ has no entry in the interfaces/sql.ini file :_____.` 操作：检查连接 URL 字符串并予以更正。
JZ025	`The TLI format for Specified Server in interfaces/sql.ini is Invalid.` 操作：检查设置并予以更正。
JZ026	`The Specified Protocol : _____ for Server : _____ in interfaces/sql.ini file :_____ is not Supported.` 说明：在 `sql.ini` 文件 (Windows) 或 `interfaces` 文件 (UNIX) 中指定了 TLI、TCP 和 NLWNSCK 以外的协议。 操作：指定受支持的协议。
JZ027	`The Specified SECMECH entrys: _____ for Server : _____ in interfaces/sql.ini file :_____ are not Supported.` 说明：在 Kerberos 连接 URL 中指定的值无效。 操作：检查 URL 并予以更正。
JZ028	`Illegal value set for JCE_PROVIDER_CLASS connection property. The property value must be a fully qualified provider class name passed as a String or an instance of java.security.Provider.` 操作：指定合法值。
JZ029	`Error looking up address for ALTERNATE_SERVER_NAME _____, (_____).` 说明：SAP jConnect 无法使用 SAP SQL Anywhere UDP 发现协议查找以 `ALTERNATE_SERVER_NAME` 属性指定的服务器。 操作：检查以 `ALTERNATE_SERVER_NAME` 连接属性指定的服务器名并予以更正。
JZ030	`The method _____ is not supported.`

SQL 状态	消息/说明/操作
JZ031	`Failed to unwrap the object of _____.` 说明： SAP jConnect 无法打开自定义类的对象，因为该自定义类不在类路径中。 操作： 将类添加到类路径中。
JZ032	`A Date or Timestamp parameter exceeds the BigDateTime/BigTime range. The server can only support BigDateTime values between 0001/01/01 12:00:00:000000AM to 9999/12/31 11:59:59.999999PM or BigTime values between 12:00:00:000000AM to 11:59:59.999999PM.`
JZ033	`Unknown Blob type returned by the server.` 说明： SAP jConnect 不能将 SAP Adaptive Server 列数据类型映射到 BLOB 数据类型。 操作： 确保 SAP Adaptive Server 列可以转换为 BLOB 数据类型。
JZ034	`The connected server is not capable of handling Large Objects [LOB].` 操作： 使用常规流方法访问 LOB。
JZ035	`To handle Large object [LOB], please set connection property 'ENABLE_LOB_LOCATOR' to true.`
JZ036	`Reference to this Large Object [LOB] is no longer valid in database. Check if you have called free() or check if transaction ended.`
JZ037	`Value of offset/position/start should be in the range [1, len] where len is the length of Large Object[LOB].`
JZ038	`Length of the object should be >= 0.`
JZ040	`_____ operation failed. The _____ has been closed alread.` 说明： 读取（写入）操作失败，因为输入流或 LOB 读取程序（输出流或 LOB 写入程序）已关闭 操作： 检查应用程序找出冲突原因并予以更正。
JZ041	`_____ operation failed on the _____.` 说明： read(write)(available()) 操作失败，因为输入流或读取程序（输出流或写入程序）（输入流）已关闭。 操作： 检查应用程序找出冲突原因并予以更正。

SQL 状态	消息/说明/操作
JZ042	`Large Object setters can not be mixed with other setters when ENABLE_LOB_LOCATOR and HOMOGENEOUS_BATCH are set to TRUE. java.sql.Types _____ was mixed with java.sql.Types _____.`
JZ043	`LOB objects are not supported for any of the possible variants of 'ENABLE_BULK_LOAD property', but false. Please consider using alternate setter APIs to insert the data.`
JZ044	`Server-side locators can not be created within the batch if,SEND_BATCHPARAMS_IMMEDIATE is TRUE. Try using client side LOBs or set SEND_BATCHPARAMS_IMMEDIATE to FALSE.`
JZ0BD	`Out of range or invalid value used for method parameter.`
JZ0BI	`setFetchSize: The fetch size should be set with the following restrictions – 0 <= rows <= (maximum number of rows in the ResultSet).` 操作：验证调用 **setFetchSize** 时使用的参数值是否位于上述范围内。
JZ0BJ	`The value set for the IMPLICIT_CURSOR_FETCH_SIZE connection property must be > 0.`
JZ0BP	`Output parameters are not allowed in Batch Update Statements.`
JZ0BR	`The cursor is not positioned on a row that supports the _____ method.` 操作：不要调用对当前行位置无效的 **ResultSet** 方法。
JZ0BS	`Batch Statements not supported.` 操作：在数据库中通过最新版本安装或更新 SAP jConnect 元数据存储过程。
JZ0BT	`The _____ method is not supported for ResultSets of type _____.` 操作：不要尝试调用对 **ResultSet** 类型无效的 **ResultSet** 方法。
JZ0C0	`Connection is already closed.` 操作：修正代码，以便在连接关闭时清空连接对象引用。

SQL 状态	消息/说明/操作
JZ0C1	`An IOException occurred which closed the connection.` 说明：该连接不能再用于其它任何数据库活动。如果出现此异常，则它始终与 JZ006 异常一起出现在一个异常链中。 操作：确定导致连接中断的 IOException 的起因。
JZ0CL	`You must define the CLASS_LOADER property when using the PRELOAD_JARS property.`
JZ0D4	`Unrecognized protocol in JDBC URL:_____.` 说明：您使用非 TDS 协议指定了连接 URL，但目前 SAP jConnect 只支持 TDS 协议。 操作：检查 URL 定义。如果 URL 指定 TDS 作为子协议，请确保该条目使用以下格式和大小写形式： jdbc::Tds:*host*:*port* 如果 URL 指定 JNDI 作为子协议，请确保它的开头为： jdbc::jndi:
JZ0D5	`Error loading protocol _____.` 操作：检查 CLASSPATH 系统变量的设置。
JZ0D6	`Unrecognized version number _____ specified in setVersion. Choose one of the SybDriver.VERSION_* values, and make sure that the version of jConnect that you are using is at or beyond the version you specify.`
JZ0D7	`Error loading url provider _____. Error message: _____.` 操作：检查 JNDI URL，确保其正确无误。
JZ0D8	`Error initializing url provider: _____.` 操作：检查 JNDI URL，确保其正确无误。
JZ0EM	`End of data.` 操作：请向 SAP 技术支持部门报告此错误。

SQL 状态	消息/说明/操作
JZ0F1	`SAP Adaptive Server Enterprise high-availability failover connection was requested but the companion server address is missing.` 说明：将 `REQUEST_HA_SESSION` 连接属性设置为 true 时，必须同时指定故障切换服务器。 操作：可以用 `SECONDARY_SERVER_HOSTPORT` 连接属性指定辅助服务器，或者用 JNDI 设置辅助服务器。
JZ0F2	`SAP Adaptive Server Enterprise high-availability failover has occurred. The current transaction is aborted, but the connection is still usable. Retry your transaction.` 说明：之前连接的后端数据库服务器停止响应，但因为您已经故障切换到辅助服务器，所以数据库连接仍可用。 操作：客户端代码应捕获这一异常，再从上次提交点重新启动事务。 如果正确处理了此异常，就可以对同一连接对象继续执行 JDBC 调用。
JZ0FP	`Incorrect value passed for parameter ____.` 说明：为当前结果集的状态指定的参数值无效。 操作：确保所指定的值有效：CLOSE_CURRENT_RESULT、KEEP_CURRENT_RESULT、CLOSE_ALL_RESULTS。
JZ0GC	`Error casting a ____ as a GSSManager. Please check the value you are setting for the GSSMANAGER_CLASS connection property. The value must be a String that specifies the fully qualified class name of a GSSManager implementation. Or, it must be an Object that extends org.ietf.jgss.GSSManager.`
JZ0GK	`The _____ array can not be null and has to contain only one key.` 说明：自动生成的键列名/索引阵列为 NULL 或含有多个键。 在阵列中只允许使用一个键，因为它与 IDENTITY 列相关。 操作：检查查询并予以更正。
JZ0GN	`Error instantiating the class ___ as a GSSManager. The exception was ____. Please check your CLASSPATH and make sure the GSSMANAGER_CLASS property value refers to a fully qualified class name of a GSSManager implementation.` 操作：确保 CLASSPATH 环境变量包含第三方 GSSManager 实现所需的所有 `.jar` 文件。

SQL 状态	消息/说明/操作
JZ0GS	`A Generic Security Services API exception occurred. The major error code is ___. The major error message is ___. The minor error code is ___. The minor error message is ____.` 操作：检查主要和次要错误代码及错误消息。检查 Kerberos 配置。请参见“安全性”（第 101 页）。
JZ0H0	`Unable to start thread for event handler; event name = _____.` 操作：向 SAP 技术支持部门报告此错误。
JZ0H1	`An event notification was received but the event handler was not found; event name = _____.` 操作：向 SAP 技术支持部门报告此错误。
JZ0HC	`Illegal character '_____' encountered while parsing hexadecimal number.` 说明：用于表示二进制值的字符串中包含不在十六进制数限定范围内（0 – 9、a – f）的字符。 操作：检查字符串中的字符值，确保它们在要求的范围内。
JZ0I3	`Incorrect argument _____ passed to method _____.` 操作：在产品文档中或在 JDBC API 中验证是否传递了正确的参数。
JZ0I5	`An unrecognized CHARSET property was specified: _____.` 操作：为 CHARSET 连接属性输入有效的字符集代码。请参见“jConnect 字符集转换程序”（第 38 页）。
JZ0I6	`An error occurred converting UNICODE to the charset used by the server. Error message: _____.` 操作：在 SAP jConnect 客户端为 CHARSET 连接属性选择另外的字符集代码，所选字符集代码应支持要发送到服务器的所有字符。可能也需要在服务器上安装另一个字符集。此外，如果您使用的是 SAP jConnect 6.05 或更高版本以及 SAP Adaptive Server Enterprise 12.5 或更高版本，则可以在向服务器发送数据时使用 unichar/univarchar 数据类型。
JZ0I7	`No response from proxy gateway.` 说明：无法建立连接，因为以 PROXY 连接属性指定的代理网关无响应。 操作：检查 PROXY 设置并予以更正。
JZ0I8	`Proxy gateway connection refused.  Gateway response: %1s.` 操作：检查代理网关设置。

SQL 状态	消息/说明/操作
JZ0I9	`This InputStream was closed.` 说明：您试图读取从 `getAsciiStream`、`getUnicodeStream` 或 `getBinaryStream` 获取的 InputStream，但该 InputStream 已关闭。该流被关闭的原因可能是您移到了另一列或取消了结果集，且没有足够的资源来高速缓存数据。 操作：增加高速缓存大小，或者按顺序读取列。
JZ0IA	`Truncation error trying to send_____.` 说明：在发送字符串之前进行字符集转换时发生截断错误。所转换字符串的长度超出为其指派的大小。 操作：在 SAP jConnect 客户端为 `CHARSET` 连接属性选择另外的字符集代码，所选字符集代码应支持要发送到服务器的所有字符。可能也需要在服务器上安装另一个字符集。
JZ0IB	`The server's default charset of _____ does not map to an encoding that is available in the client Java environment. Because jConnect will not be able to do client-side conversion, the connection is unusable and is being closed. Try using a later Java version, or try including your Java installation's i18n.jar or charsets.jar file in the classpath.`
JZ0IR	`getXXX may not be called on a column after it has been updated in the result set with a java.io.Reader.` 操作：删除对使用读取器更新过的 ResultSet 列的 `getXXX` 调用。
JZ0IS	`getXXXStream may not be called on a column after it has been updated in the result set.` 说明：在更新结果集中的列之后，您试图用以下 `SybResultSet` 方法之一读取更新后的列值：`getAsciiStream`、`getUnicodeStream`、`getBinaryStream`。SAP jConnect 不支持这种用法。 操作：不要试图从正在更新的列中读取输入流。
JZ0J0	`Offset and/or length values exceed the actual text/image length.`
JZ0LA	`Failed to instantiate Cipher object. Transformation %1s is not implemented by any of the loaded JCE providers.` 操作：确保在 CLASSPATH 的 `JCE_PROVIDER_CLASS` 连接属性中正确指定了实现。

SQL 状态	消息/说明/操作
JZ0LC	`You cannot call the ____ method on a ResultSet which is using a language cursor to fetch rows. Try setting the LANGUAGE_CURSOR connection property to false.` 说明：应用程序试图在通过语言游标创建的 ResultSet 上调用其中一种 ResultSet 游标滚动方法。
JZ0MD	`ResultSet metadata is not available.` 操作：安装元数据存储过程。
JZ0NC	`wasNull called without a preceding call to get a column.` 说明：只能在诸如 **getInt** 或 **getBinaryStream** 等用于获取列的调用之后才能调用 **wasNull**。 操作：更改代码，移动对 **wasNull** 的调用。
JZ0NE	`Incorrect URL format. URL: _____. Error message: _____.` 操作：在 URL 中，确保端口号仅包含数值字符。
JZ0NF	`Unable to load SybSocketFactory. Make sure that you have spelled the class name correctly, that the package is fully specified, that the class is available in your class path, and that it has a public zero-argument constructor.`
JZ0NK	`Generated keys are not available because either the Statement.NO_GENERATED_KEYS was used or no keys were automatically generated.` 说明：`getGeneratedKeys()` 方法无法返回自动生成的键，因为语句是通过 `.NO_GENERATED_KEYS` 执行的或者该语句未产生自动生成的键。 操作：仅对以 `.RETURN_GENERATED_KEYS` 执行的语句或应该自动生成键的语句使用 `getGeneratedKeys()`。
JZ0NS	`The method ____ is not supported and should not be called.`
JZ0P1	`Unexpected result type.` 说明：数据库返回的结果不能由语句返回给应用程序，或者应用程序此时不需要该结果。通常这表明应用程序在错误地使用 JDBC 执行查询或存储过程。如果 JDBC 应用程序连接到一个 SAP Open Server 应用程序，这可能表明该 SAP Open Server 应用程序中有错误，该错误导致 SAP Open Server 发送异常结果序列。 操作：使用 `com..utils.Debug(true, “ALL”)` 调试工具，尝试找出异常结果及其发生原因。

SQL 状态	消息/说明/操作
JZ0P4	`Protocol error. This message indicates an internal product problem. Report this error to SAP technical support.`
JZ0P7	`Column is not cached; use RE-READABLE_COLUMNS property.` 说明：当 `REPEAT_READ` 连接属性设置为 false 时，尝试再次读取列或以错误顺序读取列。 当 `REPEAT_READ` 为 false 时，只能读取一次某行的列值，且必须按列索引升序顺序读取。例如，在读取一行的第 3 列后，就不能再次读取该列的值，也不能读取该行第 2 列。 操作：要么将 `REPEAT_READ` 设置为 true，要么不要试图重复读取某列值并确保按列索引升序顺序进行读取。
JZ0P8	`The RSMDA Column Type Name you requested is unknown. This is a SAP internal error; please report it to technical support.` 说明：SAP jConnect 无法在 `ResultSetMetaData.getColumnTypeName` 方法中确定列类型的名称。 操作：确保数据库具有用于元数据的最新的存储过程。
JZ0P9	`A COMPUTE BY query has been detected. That type of result is unsupported and has been cancelled.` 操作：更改查询或存储过程，使其不使用 **COMPUTE BY**。
JZ0PA	`The query has been cancelled and the same response discarded.` 操作：检查各语句中的 SQL 异常和警告链，以找出原因。
JZ0PB	`The server does not support a requested operation.` 说明：当 SAP jConnect 建立与服务器的连接时，它会将需要服务器支持的功能通知服务器，服务器再将自身支持的功能通知 SAP jConnect。当应用程序请求的操作在最初的功能协商中被拒绝时，会发出此错误消息。 例如，如果数据库不支持动态 SQL 语句预编译，而代码调用 `SybConnection.prepareStatement(`*sql_stmt,* dynamic)，并且 *dynamic* 设置为 true，那么 SAP jConnect 将生成此消息。 操作：修改代码，使其不请求不受支持的功能。

SQL 状态	消息/说明/操作
JZ0PC	`The number and size of parameters in your query require wide table support. But either the server does not offer such support, or it was not requested during the login sequence. Try setting the JCONNECT_VERSION property to >=6 if you wish to request widetable support.` 说明：您正试图执行的语句中所含的参数数量大于服务器被配置为可处理的参数数量。可引起此异常的参数数量因所发送的数据类型不同而不同。如果发送的参数数量小于等于 481，绝对不会收到此异常。 操作：必须在 SAP Adaptive Server 12.5 或更高版本服务器中运行此查询。在连接到数据库时，请将 JCONNECT_VERSION 属性设置为“6”。
JZ0PD	`The size of the query in your dynamic prepare is large enough that you require widetable support. But either the server does not offer such support, or it was not requested during the login sequence. Try setting the JCONNECT_VERSION property to >=6 if you wish to request widetable support.` 说明：您正试图执行的动态准备语句中所含的参数数量大于服务器被配置为可处理的参数数量。 操作：必须在 SAP Adaptive Server 12.5 或更高版本服务器中运行此查询。在连接到数据库时，请将 JCONNECT_VERSION 属性设置为“6”。
JZ0PE	`The number of columns in your cursor declaration OR the size of your cursor declaration itself are large enough that you require widetable support. But either the server does not offer such support, or it was not requested during the login sequence. Try setting the JCONNECT_VERSION property to >= 6 if you wish to request wide table support.` 说明：当 SELECT 语句试图从 255 个以上的列中返回数据时，或者当 SELECT 语句的实际长度很大时（超过约 65500 个字符），会出现此错误。 操作：必须在 SAP Adaptive Server 12.5 或更高版本中运行此查询。在连接到数据库时，请将 JCONNECT_VERSION 属性设置为“6”。
JZ0PN	`Specified port number of ____ was out of range. Port numbers must meet the following conditions: 0<= portNumber <=65535.` 操作：检查数据库 URL 中指定的端口号。
JZ0R0	`Result set has already been closed.` 操作：修正代码，从而在每当关闭结果集时将 ResultSet 对象引用设置为空值。

SQL 状态	消息/说明/操作
JZ0R1	`Result set is IDLE as you are not currently accessing a row.` 说明：应用程序调用了其中一个 `ResultSet.getXXX` 列数据检索方法，但不存在当前行；应用程序没有调用 `ResultSet.next`，或者 `ResultSet.next` 返回 false 以指示不存在数据。 操作：确保先将 `rs.next` 设置为 true 后再调用 `rs.getXXX`。
JZ0R2	`No result set for this query.` 说明：您使用了 `Statement.executeQuery`，但语句没有返回任何行。 操作：对未返回任何行的语句使用 `executeUpdate`。
JZ0R3	`Column is DEAD. This is an internal error. Please report it to SAP technical support.`
JZ0R4	`Column does not have a text pointer. It is not a text/image column or the column is NULL.` 操作：确保不要更新或获取指向不支持文本/图像数据的列的文本指针，并验证您没有尝试更新值为空值的 `text/image` 列。 先插入数据，然后再进行更新。
JZ0R5	`The ResultSet is currently positioned beyond the last row. You cannot perform a get* operation to read data in this state.` 操作：更改代码，使 `ResultSet` 定位在最后一行之后时不再读取列数据。
JZ0RD	`You cannot call any of the ResultSet.get* methods on a row that has been deleted with the deleteRow() method.` 操作：更改代码，使应用程序不从已删除的行中检索数据。
JZ0RM	`refreshRow may not be called after updateRow or deleteRow.` 说明：在用 `SybCursorResult.updateRow` 更新数据库中的行之后，或用 `SybCursorResult.deleteRow` 删除行之后，又用 `SybCursorResult.refreshRow` 刷新了数据库中的行。 操作：在更新或删除数据库中的行以后，不要尝试刷新该行。
JZ0S0	`Statement state machine: Statement is BUSY.` 说明：出现此错误的唯一情况是在使用 `Statement.setCursorname` 方法时，当语句已在使用中并且具有需要读取的非游标结果时，如果应用程序试图设置游标名称，则会出现此错误。 操作：在用语句执行任何查询之前先为其设置游标名称，或者在设置游标名称之前调用 `Statement.cancel`，以确保该语句不处于繁忙状态。

SQL 状态	消息/说明/操作
JZ0S1	`Statement state machine: Trying to FETCH on IDLE statement.` 操作：关闭该语句，打开另一语句。
JZ0S2	`Statement object has already been closed.` 操作：修正应用程序，从而每当关闭语句时都将语句对象引用设置为空值。
JZ0S3	`The inherited method _____ cannot be used in this subclass.` 说明：`PreparedStatement` 不支持 `executeQuery(String)`、`executeUpdate(String)` 和 `orexecute(String)`。 操作：要传递查询字符串，应使用 `Statement`，而非 `PreparedStatement`。
JZ0S4	`Cannot execute an empty (zero-length) query.` 操作：不要执行空查询 (“”)。
JZ0S5	`The local transaction method ____ cannot be used while a global transaction is active on this connection.` 说明：使用分布式事务时可能会出现该异常。 操作：请参见“分布式事务管理支持”（第 95 页）。
JZ0S6	`The local transaction method _____ cannot be used on a pre-System 12 XAConnection.` 说明：使用分布式事务时可能会出现该异常。 操作：请参见“分布式事务管理支持”（第 95 页）。
JZ0S8	`An escape sequence in a SQL Query was malformed: ‘_____.` 操作：检查 JDBC 文档的语法是否正确。
JZ0S9	`Cannot execute an empty (zero-length) query.` 操作：不要执行空查询 (“”)。
JZ0SA	`Prepared Statement: Input parameter not set, index: _____.` 操作：确保为每个输入参数赋值。
JZ0SB	`Parameter index out of range: _____.` 操作：检查查询中的参数数量。

SQL 状态	消息/说明/操作
JZ0SC	`Callable Statement: attempt to set the return status as an InParameter.` 说明：您已准备调用一个可返回状态的存储过程，但却试图设置参数 1，该参数是返回状态。 操作：此类调用中可供设置的参数是从 2 开始的。
JZ0SD	`No registered parameter found for output parameter.` 说明：这表明存在应用程序逻辑错误。您尝试对参数调用 `getXXX` 或 `wasNull`，但尚未读取任何参数，或没有输出参数。 操作：检查以确保应用程序已经对 CallableStatement 注册了输出参数，并确保语句已执行且输出参数已读取。
JZ0SE	`Invalid object type specified for setObject().` *说明：* 向 `PreparedStatement.setObject` 传递了非法类型参数。 *操作：* 检查 JDBC 文档。参数必须是来自 `java.sql.Types` 的一个常数。
JZ0SF	`No Parameters expected. Has query been sent?` 说明：您试图对不含参数的语句设置参数。 操作：在设置参数前确保查询已发送。

JZ0SG	`An RPC did not return as many output parameters as the application had registered for it.`
	说明：如果通过调用 `CallableStatement.registerOutParam` 注册的参数多于在存储过程中声明为 OUTPUT 的参数，则会出现此错误。请参见“RPC 返回比已注册参数少的输出参数”（第 124 页）。
	操作：检查存储过程和 `registerOutParameter` 调用。确保已将所有相应参数声明为 OUTPUT。注意内容如下的代码行：
	`create procedure yourproc (@p1 int OUTPUT, ...`
	注意： 如果在使用 SQL Anywhere 时出现此错误，请升级到 SQL Anywhere 5.5.04 版。

JZ0SH	`A static function escape was used, but the metadata accessor information was not found on this server.` 操作：在使用静态函数转义之前，先安装元数据访问程序信息。
JZ0SI	`A static function escape _____ was used which is not supported by this server.` 操作：不要使用此转义。

SQL 状态	消息/说明/操作
JZ0SJ	`Metadata accessor information was not found on this database.` 操作：在进行元数据调用前，先安装元数据信息。
JZ0SK	`The oj escape is not supported for this type of database server. Workaround: use server-specific outer join syntax, if supported. Consult server documentation.` 操作：除按照消息中的说明进行操作之外，还应安装最新版本的 SAP jConnect 元数据。
JZ0SL	`Unsupported SQL type _____.` 操作：如果可能，请将参数声明为 SAP jConnect 支持的类型。不要使用 `Types.NULL` 或 `PreparedStatement.setObject (null)`。

JZ0SM	`jConnect could not execute a stored procedure because there was a problem sending the parameter(s).`
	`This problem was likely caused because the server does not support a specific datatype, or because jConnect did not request`
	`support for that datatype at connect time. Try setting the JCONNECT_VERSION connection property to a higher value.`
	`Or, if possible, try sending your procedure execution command as a language statement.`

JZ0SN	`setMaxFieldSize: field size cannot be negative.`
JZ0SO	`Invalid ResultSet concurrency type:_____.` 操作：确保声明的并发为 `ResultSet.CONCUR_READ_ONLY` 或 `ResultSet.CONCUR_UPDATABLE`。
JZ0SP	`Invalid ResultSet type:_____.` 操作：确保声明的 ResultSet 类型为 `ResultSet.TYPE_FORWARD_ONLY` 或 `ResultSet.TYPE_SCROLL_INSENSITIVE`。SAP jConnect 不支持 `ResultSet.TYPE_SCROLL_SENSITIVE` ResultSet 类型。
JZ0SQ	`In valid UDT type ____.` 说明：在调用 `DatabaseMetaData.getUDTs` 方法时，如果用户定义的类型不是 `Types.JAVA_OBJECT`、`Types.STRUCT` 或 `Types.DISTINCT`，jConnect 便会抛出此异常。 操作：使用上述三种 UDT 之一。
JZ0SR	`setMaxRows: max rows cannot be negative.`

SQL 状态	消息/说明/操作
JZ0SS	`setQueryTimeout:query timeout cannot be negative.`
JZ0ST	`jConnect cannot send a Java object as a literal parameter in a query. Make sure that your database server supports Java objects and that the LITERAL_PARAMS connection property is set to false when you execute this query.`
JZ0SU	`A Date or Timestamp parameter was set with a year of _____, but the server can only support year values between _____ and _____. If you're trying to send data to date or timestamp columns or parameters on Adaptive Server Anywhere, you may wish to send your data as Strings, and let the server convert them.` 说明： SAP Adaptive Server Enterprise 和 SAP SQL Anywhere 对 `datetime` 和 `date` 的取值范围有不同规定。 `datetime` 值的年份必须大于或等于 1753。 而 `date` 数据类型可以采用大于或等于 1 的年份值。 操作： 确保发送的 `date/timestamp` 值在可接受的范围内。
JZ0SV	`Combination of setting parameters by name and by index is not allowed in the same CallableStatement.` 说明： CallableStatement 拥有按名称和索引（顺序位置）指定的两种参数。 混合使用无效。 操作： 仅按名称或仅按索引（顺序位置）指定参数。
JZ0SW	`Invalid ResultSet holdability type: _____.` 说明： 您以 **setHoldability()** 方法指定的值无效。 操作： 仅使用以下合法值： HOLD_CURSORS_OVER_COMMIT 或 CLOSE_CURSORS_AT_COMMIT。
JZ0T2	`Listener thread read error.` 操作： 检查网络通信。
JZ0T3	`Read operation timed out.` 操作： 调用 **Statement.setQueryTimeout** 以增大超时期限。
JZ0T4	`Write operation timed out. Timeout in milliseconds: _____.` 操作： 调用 **Statement.setQueryTimeout** 以增大超时期限。
JZ0T5	`Cache used to store responses is full.` 操作： 为 `STREAM_CACHE_SIZE` 连接属性使用缺省值或更大的值。

SQL 状态	消息/说明/操作
JZ0T6	`Error reading tunneled TDS URL.` 说明：读取 URL 标头时隧道协议失败。 操作：检查为连接定义的 URL。
JZ0T7	`Listener thread read error -- caught ThreadDeath. Check network connection.` 操作：检查网络连接，尝试重新运行应用程序。如果线程仍被中止，请联系 SAP 技术支持部门。
JZ0T8	`Data received for an unknown request. Please report this error to SAP Technical Support.`
JZ0TC	`Attempted conversion between an illegal pair of types.` 说明：在 Java 类型和 SQL 类型之间进行的转换失败。 操作：检查请求的类型转换，确保 JDBC 规范支持该转换。
JZ0TD	`Caught ThreadDeath.` 说明：在 SAP jConnect 执行定时 IO 操作时，调用的应用程序线程被注销。 操作：检查应用程序代码以找出冲突并予以更正。
JZ0TE	`Attempted conversion between an illegal pair of types. Valid database types are: '_____'.` 说明：数据库列的数据类型和 `ResultSet.getXXX` 调用中请求的数据类型不能进行隐式转换。 操作：使用错误消息中列出的有效数据类型之一。
JZ0TI	`jConnect cannot make a meaningful conversion between the database type of _____ and the requested type of _____.` 说明：例如，应用程序试图对从数据库返回的 `time` 值调用 `ResultSet.getObject(int, Types.DATE)` 时，可能会发生这种异常。 操作：确保数据库的数据类型能隐式转换成要检索的对象类型。
JZ0TO	`Read operation timed out.` 说明：如果套接字读取超时，则会发生此异常。 操作：调用 `Statement.setQueryTimeout` 以增大超时期限。同时，检查正在执行的查询或存储过程以确定超时原因。

SQL 状态	消息/说明/操作
JZ0TS	`Truncation error trying to send _____.` 说明：应用程序指定的字符串的长度大于应用程序要发送的字符串长度。因此，字符串被截断为声明的长度。 操作：正确设置长度以避免发生截断。
JZ0US	`The SybSocketFactory connection property was set, and the PROXY connection property was set to the URL of a servlet. The jConnect driver does not support this combination. If you want to send secure HTTP from an applet running within a browser, use a proxy URL beginning with “https://”.`
JZ0XC	`_____ is an unrecognized transaction coordinator type.` 说明：元数据信息指示服务器支持分布式事务，但 SAP jConnect 不支持所用的协议。 操作：检验是否安装了最新的元数据脚本。如果此错误仍然存在，请联系 SAP 技术支持部门。
JZ0XS	`The server does not support XA-style transactions. Please verify that the transaction feature is enabled and licensed on this server.` 说明：SAP jConnect 尝试连接的服务器不支持分布式事务。 操作：不要为此服务器使用 `XADataSource`，或者对服务器进行升级或配置以使其支持分布式事务。
JZ0XU	`Current user does not have permission to do XA-style transactions. Be sure user has _____ role.` 说明：连接到数据库的用户无权执行分布式事务，最可能的原因是该用户不具备适当的角色。 操作：按错误消息中的提示授予用户相应的角色，或让具有该角色的另一用户执行此事务。
JZBK1	`SybBCP class is NOT initialized Please Re-Run MDA sqls to update the MDA stored procedures.` 操作：安装 MDA 存储过程。
JZBK3	`Bulk load table does not exists.` 操作：更正表名。

SQL 状态	消息/说明/操作
JZBK4	`Illegal usage of sql statements mixed with batches in bcp/arrayinsert mode.` 说明：您尝试在批处理操作过程中执行非批处理操作。 操作：等待批处理操作完成后再尝试执行非批处理操作。
JZBK5	`autocommit should be set to true when running bulk load in bcp mode.`
JZBK6	`Adaptive Server 15.7 or latter and 'allow wide dol rows' DB option must be enabled to insert rows with offsets greater than 8191.`
JZBK7	`Failed to insert data. Total data size (_____ bytes) exceeds the maximum row size (_____ bytes)allowed for the table _____.`
JZBKI	`Invalid value set for property ENABLE_BULK_LOAD.` 操作：只能将 `ENABLE_BULK_LOAD` 设置为以下有效值之一 `-ARRAYINSERT_WITH_MIXED_STATEMENTS`、`ARRAYINSERT`、`BCP` 或 `LOG_BCP`。
JZNNA	`Column does not allow null values.` 说明：您尝试在预准备语句中使用 `setNull()` 将位类型列设置为 NULL 值。 操作：检查查询并予以更正，将位类型列的值设置为 0 或 1。
S0022	`Invalid column name '_____'.` 操作：检查列名的拼写是否正确以及是否存在命名的列。
ZZ00A	`The method _____ has not been completed and should not be called.` 操作：查阅您的 SAP jConnect 版本附带的发行公告以获取更多信息。也可以访问 jConnect 网页，查看是否有较新版本的 SAP jConnect 实现了该方法。如果没有，请不要使用该方法。

词汇表

SAP jConnect for JDBC 中所用术语的词汇表。

- **应用程序编程接口 (API)** – 一种基于源代码的规范，旨在用作软件组件相互通信的接口。
- **SAP Adaptive Server Enterprise** – 管理多个数据库和多个用户、跟踪数据在磁盘上的实际位置、维护逻辑数据描述到物理数据存储的映射，以及维护内存中的数据和过程高速缓存。
- **Certicom Security Builder GSE-J** – 一款 Java Cryptography Extension (JCE) 软件加密提供程序，支持 FIPS 140-2 验证的加密算法。
- **CyberSafe TrustBroker** – 可用于 jConnect 的通用安全服务 (GSS) 管理器。
- **数据库服务器** – 使用客户端或服务器架构的数据库应用程序的后端系统。
- **数据类型** – 一个用于描述对于变量来说合法的类型、值和操作的定义属性。
- **死锁** – 在有权锁定一部分数据的两位用户都尝试锁定对方那部分数据时出现的一种情况。
- **DirectConnect** – 提供与非 SAP 数据源的基本连接的 ECDA 组件。 特别地，它通过 DirectConnect 管理器提供访问管理、事务管理以及远程系统管理。
- **区分名 (DN)** – 一种用于在目录服务器中唯一标识某个条目的字符串。 DN 包含多个（或不包含）相对区分名 (RDN) 组成部分，此部分标识目录信息树 (DIT) 中条目的位置。 DN 同时指定了名称和层级位置，所以它与文件系统中绝对路径相类似。
- **GSS 库** – 实现通用安全服务应用程序编程接口 (GSS-API) 的库。
- **IETF** – Internet Engineering Task Force。 Internet 的主要标准化组织。 IETF 是由网络设计人员、操作人员、供应商和研究员组成的大型开放式国际社区，着重于 Internet 的体系结构演变及其流畅操作。
- **SAP jConnect 驱动程序** – 一款 JDBC 驱动程序，适用于 SAP 服务器，例如使用 Tabular Data Stream (TDS) 通信协议的 Adaptive Server Enterprise。
- **Java Cryptography Extension (JCE)** – 是为实现 Java 安全功能提供统一框架的 API。
- **JDBC** – Java 数据库连接。 JDBC 是允许 Java 程序执行 SQL 语句的 Java API。
- **JDK** – Java 开发工具包。 一款用于制作 Java 程序的软件开发工具包。
- **Java 通用安全服务 (GSS) 管理器** – 为进行验证和安全消息传送提供通用接口。
- **JNDI** – Java 命名和目录接口 JNDI 使基于 Java 平台的应用程序能够访问多个命名和目录服务。

  JNDI 是来自 Oracle 的 API，用于将 Java 程序连接到命名和目录服务（如 DNS、LDAP 和 NDS）。
- **Java Runtime Environment (JRE)** – Java 开发工具包 (JDK) 的一部分，JDK 是一组用于开发 Java 应用程序的编程工具。 也称作 Java Runtime。

- **Java Transaction API (JTA)** – 一种允许应用程序和 J2EE 服务器访问事务的 API。
- **Java Transaction Service (JTS)** – 指定如何实现一个支持 Java Transaction API (JTA) 并在 API 以下级别实现对象管理组织对象事务服务 1.1 规范的 Java 映射的事务管理器。
- **Java 虚拟机 (JVM)** – 一种用于提供独立于平台的执行环境的虚拟机，在此环境中可将 Java 字节码转换为计算机语言并加以执行。
- **J2EE** – Java 2 平台企业版。 J2EE 是一种独立于平台的以 Java 为中心的环境，用于在线开发、构建和部署基于 Web 的企业应用程序。
- **Kerberos** – Kerberos 是一种用于在计算机网络中验证服务请求的安全方法。
- **密匙分发中心 (KDC)** – 是执行验证和票据生成任务的单点登录 (SSO) 设置的一部分。
- **大对象 (LOB) 数据类型** – 通常为大型字符对象（文本）或二进制对象（图像）。
- **大对象 (LOB) 定位符** – 包含指向 LOB 数据的逻辑指针，而不是数据本身，减少了通过网络在 Adaptive Server 和其客户端之间传送的数据量。
- **LDAP** – 轻量目录访问协议 LDAP 是一套软件协议，能让任何人在网络（无论是公共 Internet 还是在公司内部网）中找到组织、个人及其它资源（如文件和设备）。
- **LDAP 数据交换格式 (LDIF)** – 一种以文本形式表示目录数据的机制形式。 LDIF 规范包含在 RFC 2849 中，该规范不仅描述了一种用于表示目录数据的格式，还描述了该数据的更改机制。
- **native-protocol** – DBMS 支持的本地协议，用于在客户端和服务器之间交换请求或响应。
- **net-protocol** – 用于在中间层网关之间交换请求或响应，进而与数据库进行通信的协议。
- **对象标识符 (OID)** – 用于命名对象的标识符。 从结构上说，OID 由按层级分配的命名空间中的节点构成。
- **主服务器** – 在高可用性 (HA) 环境中，主服务器是客户端应首先尝试连接的服务器。
- **Replication Server®** – 维护多个数据库中的复制数据，同时确保这些数据的完整性和一致性。 它为使用复制系统中数据库的客户端提供本地数据访问功能，从而降低网络和集中计算机系统的负载。
- **相对区分名 (RDN)** – 区分名的一个组成部分。 RDN 包含一个或多个名值对，其中名称和值由等号分隔（例如 RDN 为“*uid=ann*”时，名称为“*uid*”，值为“*ann*”），如果存在多个名值对，则应以加号将其分隔（例如 RDN 为“*cn=Jon Doe+employeeNumber=12345*”时，名值对为“*cn=John Doe*”和“*employeeNumber=12345*”）。 实际上，包含多个名值对的 RDN（称为“*多值 RDN*”）很少见，但在条目中不存在唯一属性或者要确保条目的 DN 中包含一些有用标识信息的情况下，这种 RDN 非常有用。
- **RPC** – 远程过程调用 RPC 是一项协议，一个程序可使用它向网络中另一台计算机上的程序请求服务，而无需了解网络详情。 （有时，过程调用也称作函数调用或子例程调用。） RPC 使用客户端/服务器模型。 发出请求的程序是客户端，提供服务的程序是服务器。 与常规或本地过程调用类似，RPC 也是同步操作，需要

挂起发出请求的程序直至返回远程过程的结果。 但使用轻量进程或共享相同地址空间的线程可以并发执行多个 RPC。

- **RSA 加密** – 一种高度安全的加密方法。
- **辅助服务器** – 在高可用性 (HA) 环境中，辅助服务器是客户端在连接主服务器失败的情况下，应尝试连接的服务器。
- **单点登录 (SSO)** – 一种会话或用户验证过程，允许用户输入一个名称和口令以访问多个应用程序。 该过程针对用户拥有权限的所有应用程序对其进行验证，当用户在特定会话过程中切换应用程序时消除其它提示。
- **SAP SQL Anywhere** – 功能全面的关系数据库和数据管理工具。
- **SSL** – 安全套接字层。 SSL 是用于管理 Internet 中消息传输安全性的常用协议。
- **SAP IQ** – 专为数据仓储应用设计的高性能决策支持服务器。

  SAP IQ 是包括 SAP Adaptive Server Enterprise 和 SAP SQL Anywhere 的 SAP 产品系列的一部分。 SAP IQ 中的组件集成服务提供对大型机、UNIX 或 Windows 服务器上关系和非关系数据库的直接访问。
- **Tabular Data Stream (TDS)** – TDS 是用于描述两台计算机之间数据传输的应用程序级协议。 TDS 定义可发送的消息类型以及消息发送顺序。 TDS 依赖于面向连接的传输服务。
- **TDS 隧道服务器小程序** – 通过 HTTP 或 HTTPS 数据包穿过 TDS 流的服务器小程序。
- **UCS-2** – 通用字符集是用于对字符集进行编码的 ISO/IEC 格式。 ISO/IEC 10646 过去与 Unicode 同步；但 Unicode 现在添加了一些其它约束，所以符合 10646 并不保证兼容 Unicode。
- **UTF-16** – Unicode 转换格式 16 (UTF-16) 是 Unicode 编码系统中的一种双字节格式。
- **Wedgetail JCSI** – 可用于 SAP jConnect 的通用安全服务 (GSS) 管理器。

索引

A

B

C

D

F

G

H

I

J

K

L

M

U

W

X

Y

Z