**JDBC常见面试题集锦**

## 一、概念

### 1.什么是JDBC，在什么时候会用到它？

JDBC的全称是Java DataBase Connection，也就是Java数据库连接，我们可以用它来操作关系型数据库。JDBC接口及相关类在java.sql包和javax.sql包里。我们可以用它来连接数据库，执行SQL查询，存储过程，并处理返回的结果。JDBC接口让Java程序和JDBC驱动实现了松耦合，使得切换不同的数据库变得更加简单。

### 2.有哪些不同类型的JDBC驱动？

有四类JDBC驱动。和数据库进行交互的Java程序分成两个部分，一部分是JDBC的API，实际工作的驱动则是另一部分。**（1）**JDBC-ODBC Bridge plus ODBC Driver（类型1）：它使用ODBC驱动连接数据库。需要安装ODBC以便连接数据库，正因为这样，这种方式现在已经基本淘汰了。**（2）**Native API partly Java technology-enabled driver（类型2）：这种驱动把JDBC调用适配成数据库的本地接口的调用。**（3）**Pure Java Driver for Database Middleware（类型3）：这个驱动把JDBC调用转发给中间件服务器，由它去和不同的数据库进行连接。用这种类型的驱动需要部署中间件服务器。这种方式增加了额外的网络调用，导致性能变差，因此很少使用。**（4）**Direct-to-Database Pure Java Driver（类型4）：这个驱动把JDBC转化成数据库使用的网络协议。这种方案最简单，也适合通过网络连接数据库。不过使用这种方式的话，需要根据不同数据库选用特定的驱动程序，比如OJDBC是Oracle开发的Oracle数据库的驱动，而MySQL Connector/J是MySQL数据库的驱动。

### 3.JDBC是如何实现Java程序和JDBC驱动的松耦合的？

JDBC API使用Java的反射机制来实现Java程序和JDBC驱动的松耦合。随便看一个简单的JDBC示例，你会发现所有操作都是通过JDBC接口完成的，而驱动只有在通过Class.forName反射机制来加载的时候才会出现。

### 4.什么是JDBC连接，在Java中如何创建一个JDBC连接？

JDBC连接是和数据库服务器建立的一个会话。你可以想像成是一个和数据库的Socket连接。

创建JDBC连接很简单，只需要两步：

A. 注册并加载驱动：使用Class.forName()，驱动类就会注册到DriverManager里面并加载到内存里。 B. 用DriverManager获取连接对象：调用DriverManager.getConnnection()方法并传入数据库连接的URL，用户名及密码，就能获取到连接对象。

Connection con = null;

try{

// load the Driver Class

Class.forName("com.mysql.jdbc.Driver");

// create the connection now

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/UserDB",

"pankaj",

"pankaj123");

}catch (SQLException e) {

System.out.println("Check database is UP and configs are correct");

e.printStackTrace();

}catch (ClassNotFoundException e) {

System.out.println("Please include JDBC MySQL jar in classpath");

e.printStackTrace();

}

### 5.JDBC的DriverManager是用来做什么的？

JDBC的DriverManager是一个工厂类，我们通过它来创建数据库连接。当JDBC的Driver类被加载进来时，它会自己注册到DriverManager类里面，你可以看下JDBC Driver类的源码来了解一下。然后我们会把数据库配置信息传成DriverManager.getConnection()方法，DriverManager会使用注册到它里面的驱动来获取数据库连接，并返回给调用的程序。

### 6.在Java程序中，如何获取数据库服务器的相关信息？

使用DatabaseMetaData可以获取到服务器的信息。当和数据库的连接成功建立了之后，可以通过调用getMetaData()方法来获取数据库的元信息。DatabaseMetaData里面有很多方法，通过它们可以获取到数据库的产品名称，版本号，配置信息等。

DatabaseMetaData metaData = con.getMetaData();

String dbProduct = metaData.getDatabaseProductName();

### 7.JDBC的Statement是什么？

Statement是JDBC中用来执行数据库SQL查询语句的接口。通过调用连接对象的getStatement()方法我们可以生成一个Statement对象。我们可以通过调用它的execute()，executeQuery()，executeUpdate()方法来执行静态SQL查询。

由于SQL语句是程序中传入的，如果没有对用户输入进行校验的话可能会引起SQL注入的问题，如果想了解更多关于SQL注入的，可以看下这里。默认情况下，一个Statement同时只能打开一个ResultSet。如果想操作多个ResultSet对象的话，需要创建多个Statement。Statement接口的所有execute方法开始执行时都默认会关闭当前打开的ResultSet。

### 6.execute，executeQuery，executeUpdate的区别是什么？

Statement的execute(String query)方法用来执行任意的SQL查询，如果查询的结果是一个ResultSet，这个方法就返回true。如果结果不是ResultSet，比如insert或者update查询，它就会返回false。我们可以通过它的getResultSet方法来获取ResultSet，或者通过getUpdateCount()方法来获取更新的记录条数。

Statement的executeQuery(String query)接口用来执行select查询，并且返回ResultSet。即使查询不到记录返回的ResultSet也不会为null。我们通常使用executeQuery来执行查询语句，这样的话如果传进来的是insert或者update语句的话，它会抛出错误信息为 “executeQuery method can not be used for update”的java.util.SQLException。

Statement的executeUpdate(String query)方法用来执行insert或者update/delete（DML）语句，或者 什么也不返回DDL语句。返回值是int类型，如果是DML语句的话，它就是更新的条数，如果是DDL的话，就返回0。

只有当你不确定是什么语句的时候才应该使用execute()方法，否则应该使用executeQuery或者executeUpdate方法。

### 7.JDBC的PreparedStatement是什么？

PreparedStatement对象代表的是一个预编译的SQL语句。用它提供的setter方法可以传入查询的变量。由于PreparedStatement是预编译的，通过它可以将对应的SQL语句高效的执行多次。由于PreparedStatement自动对特殊字符转义，避免了SQL注入攻击，因此应当尽量的使用它。PreparedStatement中如何注入NULL值？

可以使用它的setNull方法来把null值绑定到指定的变量上。setNull方法需要传入参数的索引以及SQL字段的类型，像这样：

ps.setNull(10, java.sql.Types.INTEGER);.

Statement中的getGeneratedKeys方法有什么用？

有的时候表会生成主键，这时候就可以用Statement的getGeneratedKeys()方法来获取这个自动生成的主键的值了。相对于Statement，PreparedStatement的优点是什么？

它和Statement相比优点在于：PreparedStatement有助于防止SQL注入，因为它会自动对特殊字符转义。PreparedStatement可以用来进行动态查询。PreparedStatement执行更快。尤其当你重用它或者使用它的拼量查询接口执行多条语句时。使用PreparedStatement的setter方法更容易写出面向对象的代码，而Statement的话，我们得拼接字符串来生成查询语句。如果参数太多了，字符串拼接看起来会非常丑陋并且容易出错。

### 8.PreparedStatement的缺点是什么，怎么解决这个问题？

PreparedStatement的一个缺点是，我们不能直接用它来执行in条件语句；需要执行IN条件语句的话，下面有一些解决方案：分别进行单条查询——这样做性能很差，不推荐。

使用存储过程——这取决于数据库的实现，不是所有数据库都支持。动态生成PreparedStatement——这是个好办法，但是不能享受PreparedStatement的缓存带来的好处了。在PreparedStatement查询中使用NULL值——如果你知道输入变量的最大个数的话，这是个不错的办法，扩展一下还可以支持无限参数。

关于这个问题更详细的分析可以看下这篇文章。

### 9.JDBC的ResultSet是什么？

在查询数据库后会返回一个ResultSet，它就像是查询结果集的一张数据表。ResultSet对象维护了一个游标，指向当前的数据行。开始的时候这个游标指向的是第一行。如果调用了ResultSet的next()方法游标会下移一行，如果没有更多的数据了，next()方法会返回false。可以在for循环中用它来遍历数据集。默认的ResultSet是不能更新的，游标也只能往下移。也就是说你只能从第一行到最后一行遍历一遍。不过也可以创建可以回滚或者可更新的ResultSet，像下面这样。

Statement stmt = con.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE,

ResultSet.CONCUR\_UPDATABLE);

当生成ResultSet的Statement对象要关闭或者重新执行或是获取下一个ResultSet的时候，ResultSet对象也会自动关闭。可以通过ResultSet的getter方法，传入列名或者从1开始的序号来获取列数据。

### 10.有哪些不同的ResultSet？

根据创建Statement时输入参数的不同，会对应不同类型的ResultSet。如果你看下Connection的方法，你会发现createStatement和prepareStatement方法重载了，以支持不同的ResultSet和并发类型。一共有三种ResultSet对象。

ResultSet.TYPE\_FORWARD\_ONLY：这是默认的类型，它的游标只能往下移。

ResultSet.TYPE\_SCROLL\_INSENSITIVE：游标可以上下移动，一旦它创建后，数据库里的数据再发生修改，对它来说是透明的。

ResultSet.TYPE\_SCROLL\_SENSITIVE：游标可以上下移动，如果生成后数据库还发生了修改操作，它是能够感知到的。

### 11.ResultSet有两种并发类型。

ResultSet.CONCUR\_READ\_ONLY:ResultSet是只读的，这是默认类型。

ResultSet.CONCUR\_UPDATABLE:我们可以使用ResultSet的更新方法来更新里面的数据。

### 12.Statement中的setFetchSize和setMaxRows方法有什么用处？

setMaxRows可以用来限制返回的数据集的行数。当然通过SQL语句也可以实现这个功能。比如在MySQL中我们可以用LIMIT条件来设置返回结果的最大行数。

setFetchSize理解起来就有点费劲了，因为你得知道Statement和ResultSet是怎么工作的。当数据库在执行一条查询语句时，查询到的数据是在数据库的缓存中维护的。ResultSet其实引用的是数据库中缓存的结果。

假设我们有一条查询返回了100行数据，我们把fetchSize设置成了10，那么数据库驱动每次只会取10条数据，也就是说得取10次。当每条数据需要处理的时间比较长的时候并且返回数据又非常多的时候，这个可选的参数就变得非常有用了。我们可以通过Statement来设置fetchSize参数，不过它会被ResultSet对象设置进来的值所覆盖掉。

### 13.如何使用JDBC接口来调用存储过程？

存储过程就是数据库编译好的一组SQL语句，可以通过JDBC接口来进行调用。我们可以通过JDBC的CallableStatement接口来在数据库中执行存储过程。初始化CallableStatement的语法是这样的：

CallableStatement stmt = con.prepareCall("{call insertEmployee(?,?,?,?,?,?)}");

stmt.setInt(1, id);

stmt.setString(2, name);

stmt.setString(3, role);

stmt.setString(4, city);

stmt.setString(5, country);

//register the OUT parameter before calling the stored procedure

stmt.registerOutParameter(6, java.sql.Types.VARCHAR);

stmt.executeUpdate();

我们得在执行CallableStatement之前注册OUT参数。关于这个更详细的资料可以看这里。

### 13.JDBC的批处理是什么，有什么好处？

有时候类似的查询我们需要执行很多遍，比如从CSV文件中加载数据到关系型数据库的表里。我们也知道，执行查询可以用Statement或者PreparedStatement。除此之外，JDBC还提供了批处理的特性，有了它，我们可以在一次数据库调用中执行多条查询语句。JDBC通过Statement和PreparedStatement中的addBatch和executeBatch方法来支持批处理。批处理比一条条语句执行的速度要快得多，因为它需要很少的数据库调用，想进一步了解请点这里。

### 14.JDBC的事务管理是什么，为什么需要它？

默认情况下，我们创建的数据库连接，是工作在自动提交的模式下的。这意味着只要我们执行完一条查询语句，就会自动进行提交。因此我们的每条查询，实际上都是一个事务，如果我们执行的是DML或者DDL，每条语句完成的时候，数据库就已经完成修改了。

有的时候我们希望由一组SQL查询组成一个事务，如果它们都执行OK我们再进行提交，如果中途出现异常了，我们可以进行回滚。

JDBC接口提供了一个setAutoCommit(boolean flag)方法，我们可以用它来关闭连接自动提交的特性。我们应该在需要手动提交时才关闭这个特性，不然的话事务不会自动提交，每次都得手动提交。数据库通过表锁来管理事务，这个操作非常消耗资源。因此我们应当完成操作后尽快的提交事务。在这里有更多关于事务的示例程序。

### 15.如何回滚事务？

通过Connection对象的rollback方法可以回滚事务。它会回滚这次事务中的所有修改操作，并释放当前连接所持有的数据库锁。

### 16.JDBC的保存点(Savepoint)是什么，如何使用？

　　有时候事务包含了一组语句，而我们希望回滚到这个事务的某个特定的点。JDBC的保存点可以用来生成事务的一个检查点，使得事务可以回滚到这个检查点。

　　一旦事务提交或者回滚了，它生成的任何保存点都会自动释放并失效。回滚事务到某个特定的保存点后，这个保存点后所有其它的保存点会自动释放并且失效。可以读下这个了解更多关于JDBC Savepoint的信息。

### 17.JDBC的DataSource是什么，有什么好处？

　　DataSource即数据源，它是定义在javax.sql中的一个接口，跟DriverManager相比，它的功能要更强大。我们可以用它来创建数据库连接，当然驱动的实现类会实际去完成这个工作。除了能创建连接外，它还提供了如下的特性：

1.缓存PreparedStatement以便更快的执行

2.可以设置连接超时时间

3.提供日志记录的功能

4.ResultSet大小的最大阈值设置

通过JNDI的支持，可以为servlet容器提供连接池的功能

### 18.如何通过JDBC的DataSource和Apache Tomcat的JNDI来创建连接池？

对部署在servlet容器中的WEB程序而言，创建数据库连接池非常简单，仅需要以下几步。

在容器的配置文件中创建JDBC的JNDI资源，通常在server.xml或者context.xml里面。像这样：

<Resource name="jdbc/MyDB"

global="jdbc/MyDB"

auth="Container"

type="javax.sql.DataSource"

driverClassName="com.mysql.jdbc.Driver"

url="jdbc:mysql://localhost:3306/UserDB"

username="pankaj"

password="pankaj123"

maxActive="100"

maxIdle="20"

minIdle="5"

maxWait="10000"/>

<ResourceLink name="jdbc/MyLocalDB"

global="jdbc/MyDB"

auth="Container"

type="javax.sql.DataSource" />

在WEB应用程序中，先用InitialContext来查找JNDI资源，然后获取连接。

Context ctx = new InitialContext();

DataSource ds = (DataSource) ctx.lookup("java:/comp/env/jdbc/MyLocalDB");

### 19.Apache的DBCP是什么？

　　如果用DataSource来获取连接的话，通常获取连接的代码和驱动特定的DataSource是紧耦合的。另外，除了选择DataSource的实现类，剩下的代码基本都是一样的。

　　Apache的DBCP就是用来解决这些问题的，它提供的DataSource实现成为了应用程序和不同JDBC驱动间的一个抽象层。Apache的DBCP库依赖commons-pool库，所以要确保它们都在部署路径下。

### 20.什么是数据库的隔离级别？

　　当我们为了数据的一致性使用事务时，数据库系统用锁来防止别人访问事务中用到的数据。数据库通过锁来防止脏读，不可重复读(Non-Repeatable Reads)及幻读（Phantom-Read）的问题。

　　数据库使用JDBC设置的隔离级别来决定它使用何种锁机制，我们可以通过Connection的getTransactionIsolation和setTransactionIsolation方法来获取和设置数据库的隔离级别。

隔离级别 事务 脏读 不可重复读 幻读

TRANSACTION\_NONE 不支持 不可用 不可用 不可用

TRANSACTION\_READ\_COMMITTED 支持 阻止 允许 允许

TRANSACTION\_READ\_UNCOMMITTED 支持 允许 允许 允许

TRANSACTION\_REPEATABLE\_READ 支持 阻止 阻止 允许

TRANSACTION\_SERIALIZABLE 支持 阻止 阻止 阻止

### 21.JDBC的RowSet是什么，有哪些不同的RowSet？

　　RowSet用于存储查询的数据结果，和ResultSet相比，它更具灵活性。RowSet继承自ResultSet，因此ResultSet能干的，它们也能，而ResultSet做不到的，它们还是可以。RowSet接口定义在javax.sql包里。

### 22.RowSet提供的额外的特性有：

提供了Java Bean的功能，可以通过settter和getter方法来设置和获取属性。RowSet使用了JavaBean的事件驱动模型，它可以给注册的组件发送事件通知，比如游标的移动，行的增删改，以及RowSet内容的修改等。

RowSet对象默认是可滚动，可更新的，因此如果数据库系统不支持ResultSet实现类似的功能，可以使用RowSet来实现。

　　RowSet分为两大类：

　　A. 连接型RowSet——这类对象与数据库进行连接，和ResultSet很类似。JDBC接口只提供了一种连接型RowSet，javax.sql.rowset.JdbcRowSet，它的标准实现是com.sun.rowset.JdbcRowSetImpl。 B. 离线型RowSet——这类对象不需要和数据库进行连接，因此它们更轻量级，更容易序列化。它们适用于在网络间传递数据。有四种不同的离线型RowSet的实现。

CachedRowSet——可以通过他们获取连接，执行查询并读取ResultSet的数据到RowSet里。我们可以在离线时对数据进行维护和更新，然后重新连接到数据库里，并回写改动的数据。

WebRowSet继承自CachedRowSet——他可以读写XML文档。

JoinRowSet继承自WebRowSet——它不用连接数据库就可以执行SQL的join操作。

FilteredRowSet继承自WebRowSet——我们可以用它来设置过滤规则，这样只有选中的数据才可见。

### 23.RowSet和ResultSet的区别是什么？

　　RowSet继承自ResultSet，因此它有ResultSet的全部功能，同时它自己添加了些额外的特性。RowSet一个最大的好处是它可以是离线的，这样使得它更轻量级，同时便于在网络间进行传输。

　　具体使用哪个取决于你的需求，不过如果你操作ResultSet对象的时间较长的话，最好选择一个离线的RowSet，这样可以释放数据库连接。

### 24.常见的JDBC异常有哪些？

有以下这些：

java.sql.SQLException——这是JDBC异常的基类。

java.sql.BatchUpdateException——当批处理操作执行失败的时候可能会抛出这个异常。这取决于具体的JDBC驱动的实现，它也可能直接抛出基类异常java.sql.SQLException。

java.sql.SQLWarning——SQL操作出现的警告信息。

java.sql.DataTruncation——字段值由于某些非正常原因被截断了（不是因为超过对应字段类型的长度限制）。

### 25.JDBC里的CLOB和BLOB数据类型分别代表什么？

　　CLOB意思是Character Large OBjects，字符大对象，它是由单字节字符组成的字符串数据，有自己专门的代码页。这种数据类型适用于存储超长的文本信息，那些可能会超出标准的VARCHAR数据类型长度限制（上限是32KB）的文本。

　　BLOB是Binary Larget OBject，它是二进制大对象，由二进制数据组成，没有专门的代码页。它能用于存储超过VARBINARY限制（32KB）的二进制数据。这种数据类型适合存储图片，声音，图形，或者其它业务程序特定的数据。

### 26.JDBC的脏读是什么？哪种数据库隔离级别能防止脏读？

　　当我们使用事务时，有可能会出现这样的情况，有一行数据刚更新，与此同时另一个查询读到了这个刚更新的值。这样就导致了脏读，因为更新的数据还没有进行持久化，更新这行数据的业务可能会进行回滚，这样这个数据就是无效的。

　　数据库的TRANSACTIONREADCOMMITTED，TRANSACTIONREPEATABLEREAD，和TRANSACTION\_SERIALIZABLE隔离级别可以防止脏读。

### 27.什么是两阶段提交？

　　当我们在分布式系统上同时使用多个数据库时，这时候我们就需要用到两阶段提交协议。两阶段提交协议能保证是分布式系统提交的原子性。在第一个阶段，事务管理器发所有的事务参与者发送提交的请求。如果所有的参与者都返回OK，它会向参与者正式提交该事务。如果有任何一个参与方返回了中止消息，事务管理器会回滚所有的修改动作。

### 28.JDBC中存在哪些不同类型的锁？

　　从广义上讲，有两种锁机制来防止多个用户同时操作引起的数据损坏。

　　乐观锁——只有当更新数据的时候才会锁定记录。 悲观锁——从查询到更新和提交整个过程都会对数据记录进行加锁。

　　不仅如此，一些数据库系统还提供了行锁，表锁等锁机制。

### 29.DDL和DML语句分别代表什么？

　　DDL（数据定义语言，Data Definition Language）语句用来定义数据库模式。Create，Alter, Drop, Truncate, Rename都属于DDL语句，一般来说，它们是不返回结果的。

　　DML（数据操作语言，Data Manipulation Language）语句用来操作数据库中的数据。select, insert, update, delete, call等，都属于DML语句。

### 30.java.util.Date和java.sql.Date有什么区别？

　　java.util.Date包含日期和时间，而java.sql.Date只包含日期信息，而没有具体的时间信息。如果你想把时间信息存储在数据库里，可以考虑使用Timestamp或者DateTime字段。

### 31.如何把图片或者原始数据插入到数据库中？

可以使用BLOB类型将图片或者原始的二进制数据存储到数据库里。

### 32.什么是幻读，哪种隔离级别可以防止幻读？

　幻读是指一个事务多次执行一条查询返回的却是不同的值。假设一个事务正根据某个条件进行数据查询，然后另一个事务插入了一行满足这个查询条件的数据。之后这个事务再次执行了这条查询，返回的结果集中会包含刚插入的那条新数据。这行新数据被称为幻行，而这种现象就叫做幻读。只有TRANSACTION\_SERIALIZABLE隔离级别才能防止产生幻读。

### 33.SQLWarning是什么，在程序中如何获取SQLWarning？

　　SQLWarning是SQLException的子类，通过Connection, Statement, Result的getWarnings方法都可以获取到它。 SQLWarning不会中断查询语句的执行，只是用来提示用户存在相关的警告信息。

### 34.如果Oracle的存储过程的入参出参中包含数据库对象，应该如何进行调用？

　　如果Oracle的存储过程的入参出参中包含数据库对象，我们需要在程序创建一个同样大小的对象数组，然后用它来生成Oracle的STRUCT对象。然后可以通过数据库对象的setSTRUCT方法传入这个struct对象，并对它进行使用。

### 35.什么是JDBC的最佳实践？

下面列举了其中的一些：

1、 数据库资源是非常昂贵的，用完了应该尽快关闭它。Connection, Statement, ResultSet等JDBC对象都有close方法，调用它就好了。

2、 养成在代码中显式关闭掉ResultSet，Statement，Connection的习惯，如果你用的是连接池的话，连接用完后会放回池里，但是没有关闭的ResultSet和Statement就会造成资源泄漏了。

3、 在finally块中关闭资源，保证即便出了异常也能正常关闭。

4、 大量类似的查询应当使用批处理完成。

5、 尽量使用PreparedStatement而不是Statement，以避免SQL注入，同时还能通过预编译和缓存机制提升执行的效率。

6、 如果你要将大量数据读入到ResultSet中，应该合理的设置fetchSize以便提升性能。

7、 你用的数据库可能没有支持所有的隔离级别，用之前先仔细确认下。

8、 数据库隔离级别越高性能越差，确保你的数据库连接设置的隔离级别是最优的。

9、 如果在WEB程序中创建数据库连接，最好通过JNDI使用JDBC的数据源，这样可以对连接进行重用。

10、如果你需要长时间对ResultSet进行操作的话，尽量使用离线的RowSet。

## 二、JDBC相关测试题

##### 1.JDBC连接数据库6步

**答案：**Load the JDBC Driver

Establish the Database Connection

Create a Statement Object

Execute a Query

Process the Results

Close the Connection

##### 2.事务的4大特性

**答案：**原子性A，一致性C，隔离性I，永久性D

##### 3.select count（\*） from student 和select count（id） from student 之间的区别。

**答案：**

select count(\*) 统计所有学生的记录个数，包括空记录。

Select count(Id) 统计所有学生的记录个数，不包括null记录。

##### 4.假设现在有表system.table1，表中有三个字段：id(数值型)、name（字符型）、age（数值型）写出SQL语句完成如下功能：在表中查出年龄大于20，且名字以“王”开头的记录，并且按照年龄的倒叙排列出来（年龄大的在前面）。

**答案：**

Select \* from system.table1 where age>20 and name like ‘王%’ order by age DESC;

##### 5 .创建CUSTOMERS表，字段为：ID：（非空，主键）bigint，NAME：（非空）varchar，AGE：int类型；创建ORDERS表，字段为：ID：（非空，主键，）bigint，ORDER\_NUMBER：（非空）varchar，PRICE：double，CUSTOMER\_ID ：（外键）bigint，设置级连删除；

**答案：**

create table CUSTOMBERS(ID bigint not null,NAME varchar(15),AGE int,primary key (ID));

create table ORDERS(ID bigint not null,ORDER\_NUMBER varchar(15) not nulll,PRICE double precision,CUSTOMER\_ID bigint,primary key(ID),);

alter table ORDERS add constraint FK\_CUSTOMER foreign key (CUSTOMER\_ID) references CUSTOMERS(ID) on delete cascade;

##### 6.使用左外连接查询，ORDERS 和 CUSTOMERS 表，

**答案：**

select c.ID, o.CUSTOMER\_ID,c.NAME, o.ID ORDER\_ID,ORDER\_NUMBER from CUSTOMERS c left outer join ORDERS o no c.ID=o.CUSTOMER\_ID;

##### 7 .简述数据库事务的生命周期？（可画流程图）

**答案：无。**

##### 8.delete from tablea & truncate table tablea的区别

**答案：**

truncate 语句执行速度快,占资源少,并且只记录页删除的日志；

delete 对每条记录的删除均需要记录日志

## 三、JDBC相关测试题（二）

#### 选择题

1. 有关JDBC的选项正确的是哪一个？
2. JDBC是一种被设计成通用的数据库连接技术，JDBC技术不光可以应用在Java程序里面，还可以用在C++这样的程序里面。
3. JDBC技术是SUN公司设计出来专门用在连接Oracle数据库的技术，连接其他的数据库只能采用微软的ODBC解决方案
4. 微软的ODBC和SUN公司的JDBC解决方案都能实现跨平台使用，只是JDBC的性能要高于ODBC
5. JDBC只是个抽象的调用规范，底层程序实际上要依赖于每种数据库的驱动文件
6. 选择JDBC可以执行的语句（多选）
7. DDL
8. DCL
9. DML
10. 以上都可以
11. 选择Java程序开发中推荐使用的常用数据库（多选）
12. Oracle
13. SQL Server 2000
14. MySQL
15. DB2
16. 哪个不是JDBC用到的接口和类？
17. System
18. Class
19. Connection
20. ResultSet
21. 使用Connection 的哪个方法可以建立一个PreparedStatement接口？
22. createPrepareStatement()
23. prepareStatement()
24. createPreparedStatement()
25. preparedStatement()
26. 下面的描述正确的是什么？
27. PreparedStatement继承自Statement
28. Statement继承自PreparedStatement
29. ResultSet继承自Statement
30. CallableStatement继承自PreparedStatement
31. 下面的描述错误的是什么？
32. Statement的executeQuery()方法会返回一个结果集
33. Statement的executeUpdate()方法会返回是否更新成功的boolean值
34. 使用ResultSet中的getString()可以获得一个对应于数据库中char类型的值
35. ResultSet中的next()方法会使结果集中的下一行成为当前行
36. 如果数据库中某个字段为numberic型,可以通过结果集中的哪个方法获取？
37. getNumberic()
38. getDouble()
39. getBigDecimal ()
40. getFloat()
41. 在Jdbc中使用事务,想要回滚事务事务的方法是什么？
42. Connection的commit()
43. Connection的setAutoCommit()
44. Connection的rollback()
45. Connection的close()
46. 在JDBC编程中执行完下列SQL语句SELECT name, rank, serialNo

FROM employee，能得到rs的第一列数据的代码是哪两个？

1. rs.getString(0);
2. rs.getString("name");
3. rs.getString(1);
4. rs.getString("ename");
5. 下面关于PreparedStatement的说法错误的是什么
6. PreparedStatement继承了Statement
7. PreparedStatement可以有效的防止SQL注入
8. PreparedStatement不能用于批量更新的操作
9. PreparedStatement可以存储预编译的Statement，从而提升执行效率
10. 下面的选项加载MySQL驱动正确的是哪一个？
11. Class.forname("org.gjt.mm.mysql.Driver");
12. Class.forname("org.gjt.mysql.jdbc.Driver");
13. Class.forname("org.git.mm.mysql.Driver");
14. Class.forname("org.git.mysql.jdbc.Driver");
15. 下面的选项加载MySQL驱动正确的是哪一个？
16. Class.forname("com.mysql.JdbcDriver");
17. Class.forname("com.mysql.jdbc.Driver");
18. Class.forname("com.mysql.driver.Driver");
19. Class.forname("com.mysql.jdbc.MySQLDriver");
20. 下面代码加载Oracle驱动正确的是哪一个？
21. DriverManager.register("oracle.driver.OracleDriver");
22. DriverManager.forname("oracle.driver.OracleDriver");
23. DriverManager.load("oracle.driver.OracleDriver");
24. DriverManager.newInstance("oracle.driver.OracleDriver");
25. 有关Connection描述错误的是哪一个？
26. Connection是Java程序与数据库建立的连接对象，这个对象只能用来连接数据库，不能执行SQL语句。
27. JDBC的数据库事物控制要靠Connection对象完成。
28. Connection对象使用完毕后要及时关闭，否则会对数据库造成负担。
29. 只用MySQL和Oracle数据库的JDBC程序需要创建Connection对象，其他数据库的JDBC程序不用创建Connection对象就可以执行CRUD操作。
30. 下面有关JDBC事物的描述正确的是哪一个？
31. JDBC事物默认为自动提交，没执行一条SQL语句就会开启一个事物，执行完毕之后自动提交事物，如果出现异常自动回滚事物。
32. JDBC的事物不同于数据库的事物，JDBC的事物依赖于JDBC驱动文件，拥有独立于数据库的日志文件，因此JDBC的事物可以替代数据库事物。
33. 如果需要开启手动提交事物需要调用Connection对象的start()方法。
34. 如果事物没有提交就关闭了Connection连接，那么JDBC会自动提交事物。
35. 下列的预编译SQL哪一个是正确的？
36. SELECT \* FROM ？;
37. SELECT ?,?,? FROM emp ;
38. SELECT \* FROM emp WHERE salary>(?)
39. 以上都不对
40. 能执行预编译SQL的是哪一个选项？
41. Statement
42. PreparedStatement
43. PrepareStatement
44. 以上都不是
45. 如果为下列预编译SQL的第三个问号赋值，那么正确的选项是哪一个？

UPDATE emp SET ename=?,job=?,salary=? WHERE empno=?;

1. pst.setInt("3",2000);
2. pst.setInt(3,2000);
3. pst.setFloat("salary",2000);
4. pst.setString("salary","2000");
5. 有关PreparedStatement说法正确的是哪一个？
6. 该对象只能执行带问号占位符的预编译SQL，不能执行SQL语句。
7. 该对象执行的时候，只能执行查询语句，其他预编译SQL语句只能由Statement执行。
8. 该对象因为只能执行查询语句，所以该对象不能用在JDBC事物中。
9. 该对象与一条SQL预编译语句绑定，不能执行其他预编译SQL语句。
10. 有关预编译SQL的有点说法错误的是哪一个？
11. 预编译SQL可以被PreparedStatement反复执行
12. 预编译SQL语句在PreparedStatement对象创建之后就被传递给数据库解析，之后PreparedStatement执行预编译的时候，其实传递给数据库的只有占位符的参数。如果需要批量插入1000条记录的时候，预编译SQL只被数据库解析一次，其余都是数据库接受参数数据然后执行，这样的速度大为提高。
13. 预编译SQL的安全性好，可以抵御数据库脚本注入攻击，而这却是Statement所不具备的。
14. 预编译SQL的占位符既可以替代数据表，也可以替代表达式的数据，甚至是子查询语句。
15. 下列选项有关ResultSet说法错误的是哪一个？
16. ResultSet是查询结果集对象，如果JDBC执行查询语句没有查询到数据，那么ResultSet将会是null值。
17. 判断ResultSet是否存在查询结果集，可以调用它的next()方法。
18. 如果Connection对象关闭，那么ResultSet也无法使用。
19. 如果一个事物没有提交，那么ResultSet中是看不到事物过程中的临时数据。
20. SELECT COUNT(\*) FROM emp;这条SQL语句执行，如果员工表中没有任何数据，那么ResultSet中将会是什么样子？
21. null
22. 有数据
23. 不为null，但是没有数据
24. 以上都选项都不对
25. 下面选项的MySQL数据库URL正确的是哪一个？
26. jdbc:mysql://localhost/company
27. jdbc:mysql://localhost:3306:company
28. jdbc:mysql://localhost:3306/company
29. jdbc:mysql://localhost/3306/company
30. 编写JDBC的时候下列选项哪一个错误的使用了某个包的类？
31. java.sql.Connection
32. java.sql.Statement
33. java.sql.PreparedStatement
34. java.sql.Driver

#### 编程题

多年以来，我国的啤酒生产企业一直是群雄割据、各自为战的局面。各地都有自己的生产企业，它们受到地方主义的保护，企业规模小，经营业绩不佳，抵御市场风险的能力差。近几年来，这种格局正在被打破，国内的一些啤酒巨头，如青岛啤酒、燕京啤酒等，开始进行跨地区的大规模重组，对市场进行重新整合，逐步扩大企业的生产规模，形成新的市场竞争优势，比如，青岛啤酒目前在国内已有48家生产企业。

啤酒的生产和销售所需的时间相对也比较短，库存量比较低。原因是啤酒在短时间内可能会变质，而库存费用和生产费用相对也比较高。要减少库存量，又要保持较强的市场竞争能力，就需要对生产和需求量的变化做出迅速反应。这就要求对需求量做出科学的预测，以作为制定下一年度生产计划的依据。

宁波一家啤酒生产企业现正着手制定2009年的生产计划，这就需要对市场需求量做出预测，作为制定计划的参考依据。经理室要求财务部负责测算工作，并提出预测数据。计划财务部认为企业最近十五年的销售数据对预测有用，于是找来企业历年的销售数据如下表：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序 号 | 年 份 | 季 度 （单位：万吨） | | | | 合 计 |
| 一 | 二 | 三 | 四 |
| 1 | 1995 | 6 | 7 | 9 | 8 | 30 |
| 2 | 1996 | 9 | 12 | 13 | 10 | 44 |
| 3 | 1997 | 13 | 14 | 17 | 13 | 57 |
| 4 | 1998 | 15 | 18 | 19 | 14 | 66 |
| 5 | 1999 | 18 | 20 | 24 | 19 | 81 |
| 6 | 2000 | 22 | 24 | 29 | 23 | 98 |
| 7 | 2001 | 23 | 26 | 32 | 24 | 105 |
| 8 | 2002 | 25 | 32 | 37 | 26 | 120 |
| 9 | 2003 | 30 | 38 | 42 | 30 | 140 |
| 10 | 2004 | 29 | 39 | 50 | 35 | 153 |
| 11 | 2005 | 30 | 39 | 51 | 37 | 157 |
| 12 | 2006 | 29 | 42 | 55 | 38 | 164 |
| 13 | 2007 | 31 | 43 | 54 | 41 | 169 |
| 14 | 2008 | 33 | 45 | 58 | 42 | 178 |
| 15 | 2009 | 34 | 46 | 60 | 45 | 185 |

编写程序完成如下要求：

1. 编写JDBC程序创建数据output表，表结构如下

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 列名 | 数据类型 | 主键 | 非空 | 唯一 | 索引 | 备注 |
| num | number(10) | 是 |  |  |  | 序号 |
| year | number(4) |  | 是 | 是 | 是 | 年份 |
| q1 | number(10) |  | 是 |  |  | 第一季产量 |
| q2 | number(10) |  | 是 |  |  | 第二季产量 |
| q3 | number(10) |  | 是 |  |  | 第三季产量 |
| q4 | number(10) |  | 是 |  |  | 第四季产量 |
| total | number(10) |  | 是 |  |  | 年产量 |

1. 编写JDBC程序将表1中的数据插入到数据库，要求使用整体性事务，不是没插入一条提交一次事务。
2. 编写JDBC程序读取数据库中的数据，在控制台输出年增长量的平均值

4. 利用数据表中最近四年的数据预测未来五年中每年啤酒的产量数据（四个季度和总产量），输出到控制台上面。

5. 用户在控制台上面输入一个任意的年份，将该年的记录打印到控制台上面，需要输出年份、四个季度的产量和年总产量

(1) 如果用户输入的是1995年以前的年份，控制台输出“没有数据”

(2) 如果用户输入的是1995-2009年之间的年份，输出的数据从数据库中查询

(3) 如果输入的是2009年以后的年份，输出的数据是预测的数据（参见第四题的思路）