**1. 如何配置tomcat数据源**

方法一：  
1.配置tomcat下的conf下的context.xml文件,在之间添加连接池配置:  
< Resource name=”jdbc/mydb”  
auth=”Container”  
type=”javax.sql.DataSource”  
driverClassName=”oracle.jdbc.driver.OracleDriver”  
url=”jdbc:oracle:thin:@localhost:1521:mydb”  
username=”bmgis”  
password=”bmgis”  
maxActive=”100″  
maxIdle=”30″  
maxWait=”10000″ />  
2.配置你的应用下的web.xml中的之间加入:  
< resource-ref>  
< description>DB Connection</description>  
< res-ref-name>jdbc/mydb</res-ref-name>  
< res-type>javax.sql.DataSource</res-type>  
< res-auth>Container</res-auth>  
< /resource-ref>  
在以往的tomcat当中还需要在web.xml指定相应的resource，在tomcat 5.5以后的版本不写也可以，但建议还是配置。

3.把连接数据库的第三方驱动放到${CATALINA\_HOME}/lib下面就ok了  
4.测试程序test.jsp如下：  
<%@ page import=”javax.naming.\*”%>  
< %@ page import=”java.sql.\*”%>  
< %@ page import=”javax.sql.\*”%>  
< %  
Context initContext = new InitialContext();  
Context envContext  = (Context)initContext.lookup(“java:/comp/env”);  
DataSource ds = (DataSource)envContext.lookup(“jdbc/myoracle”);  
Connection conn = ds.getConnection();  
conn.close();  
%>

方法二：  
我们只需要在WebRoot目录下，新建一个META-INF的目录（假如不存在,注意目录名称大写），  
在该目录下创建一个context.xml文件，并且在context.xml文件当添加以下的配置信息：  
<Context>  
< Resource name=”jdbc/mydb”  
auth=”Container”  
type=”javax.sql.DataSource”  
driverClassName=”oracle.jdbc.driver.OracleDriver”  
url=”jdbc:oracle:thin:@localhost:1521:mydb”  
username=”bmgis”  
password=”bmgis”  
maxActive=”100″  
maxIdle=”30″  
maxWait=”10000″  
logAbandoned=”true” />  
< /Context>  
其中：  
name 表示指定的jndi名称  
auth 表示认证方式，一般为Container  
type 表示数据源床型，使用标准的javax.sql.DataSource  
maxActive 表示连接池当中最大的数据库连接  
maxIdle 表示最大的空闲连接数  
maxWait 当池的数据库连接已经被占用的时候，最大等待时间  
logAbandoned 表示被丢弃的数据库连接是否做记录，以便跟踪  
username 表示数据库用户名  
password 表示数据库用户的密码  
driverClassName 表示JDBC DRIVER  
url 表示数据库URL地址

**2. oracle数据库内外联接查询**

内连接（inner join）。  
外连接：  
全连接（full join）、左连接（left join）、右连接（right join）。  
交叉联接(cross join)。  
外连接与内连接不一样，外连接返回的查询结果中不仅包含符合条件的行，还包括左表(左外连接),右表(右外连接)或者两个连接表(全外连接)中的所有不符合条件的数据行。  
1.左连接 (left [outer] join)  
左外连接就是将左表的所有数据分别于右表的每条数据进行连接组合，返回的结果除内连接的数据外，还有左表中不符合条件的数据，并在右表的相应列中填上null值。  
SQL语句如下：  
select \* from mt\_pb\_org o left join mt\_pb\_orgframe f on o.PB\_ORGFRAMEID = f.PB\_ORGFRAMEID;  
等价语句：  
select \* from mt\_pb\_org o,mt\_pb\_orgframe f where o.pb\_orgframeid = f.pb\_orgframeid(+);  
2.右连接 (right [outer] join)  
右外连接就是将右表中的所有数据分别与左表的每条数据进行连接组合，返回的结果除了内连接的数据外，还有右表中不符合条件的数据，并在左表相应的列中填上null值。  
SQL语句如下：  
select \* from mt\_pb\_org o right join mt\_pb\_orgframe on o.pb\_orgframeid = f.pb\_orgframeid;  
等价语句：  
select \* from mt\_pb\_org o,mt\_pb\_orgframe f where o.pb\_orgframeid(+) = f.pb\_orgframeid;  
3.全外连接 (full [outer] join)  
全外连接就是将左表的所有数据分别与右表的每条数据进行连接组合，返回的结果除了内连接的数据外，还有两个表中不符合条件的数据，并在左表或者右表的相应列中填上null值。  
SQL语句如下：  
select \* from mt\_pb\_org o full join mt\_pb\_orgframe o.pb\_orgframeid = f.pb\_orgframeid;  
4.交叉连接(cross join)  
交叉连接不带WHERE 子句，它返回被连接的两个表所有数据行的笛卡尔积，返回到结果集合中的数据行数等于第一个表中符合查询条件的数据行数乘以第二个表中符合查询条件的数据行数。  
SQL语句如下：  
select \* from mt\_pb\_org o cross join mt\_pb\_orgframe f;

**3. 存储过程概念**

存储过程（Stored Procedure）是在大型数据库系统中，一组为了完成特定功能的SQL 语句集，经编译后存储在数据库中，用户通过指定存储过程的名字并给出参数（如果该存储过程带有参数）来执行它。存储过程是数据库中的一个重要对象，任何一个设计良好的数据库应用程序都应该用到存储过程。

**4.Windows下如何启动tomcat**

直接bat启动安装为windows service.

**5. Tomcat对外提供服务的访问端口**

默认8080

**6. Spring 依赖注入原理**

所谓依赖注入就是指：在运行期，由外部容器动态地将依赖对象注入到组件中。当spring容器启动后，spring容器初始化，创建并管理bean对象，以及销毁它。所以我们只需从容器直接获取Bean对象就行，而不用编写一句代码来创建bean对象。这种现象就称作控制反转，即应用本身不负责依赖对象的创建及维护，依赖对象的创建及维护是由外部容器负责的。这样控制权就由应用转移到了外部容器，控制权的转移就是所谓反转。虽然平时只需要按要求将bean配置到配置文件中，但是了解其实现过程对理解spring的实现原理是有好处的  
**7. Spring 依赖注入两种方式**

（1）：通过 setter 方法注入：

          <property name=“ ” ></property>

     其中，name属性的取值依setter方法名而定，要求这个类里面这个对应的属性必须有setter方法

 （2）：通过构造方法注入：

          <constructor-arg index=””></ constructor-arg>

     其中，index表示构造方法中的参数索引(第一个参数索引为0)

     要求这个类里面必须有想对应的构造方法