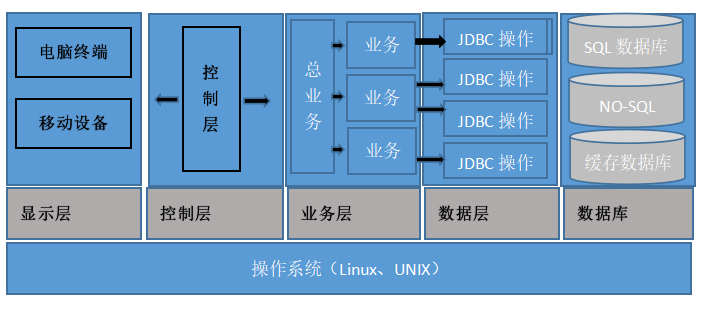
**程序分层：**

在一个完整的项目中，对程序进行合理的分层，可以让开发变得更加方便。每一层有每 一层的开发人员。

1. 在一个项目中，最核心的是业务层。
2. 不同层之间最为重要的连接部分就是接口，并且在后台代码中一定要有业务层接口 和数据层接口。数据层给以后业务层使用，业务层以后的控制层使用。

·数据层（数据访问层，Dtat Access Object）也叫持久层或数据访问对象，也称DAO 设计模式。用于执行数据的具体操作，主要任务是负责完成数据的CRUD，即 create（创建数据）、read（读取数据）、Update（修改数据）、delete（删除数据）。 使用java.sql.PreparedStatement接口完成。

·业务层（业务对象，Business Object，BD或者称为Service，服务层）。服务层负 责根据业务需求进行数据层的操作，一个业务层要包含多个数据层。

**业务层与数据层的区分：**

以雇员管理为例，来进行业务层与数据层的区别。

1）【业务层】增加一个雇员信息

【数据层】根据增加的雇员编号查看此雇员是否存在

【数据层】如果雇员不存在则执行插入操作，如果不存在则不插入

2）【业务层】修改一个雇员的信息

【数据层】直接修改一个雇员的信息，如果没有修改返回的更新行数是0

1. 【业务层】删除多个雇员的信息

【数据层】接收一个Set集合保存若干个要删除的雇员编号，然后使用IN进 行删除

1. 【业务层】根据编号查询一个雇员的信息

【数据层】返回一个雇员的完整信息

1. 【业务层】取得全部雇员的信息，要求可以实现模糊查询和分页显示，查询结果除 了返回数据还要求取得模糊查询或全部查询时所返回的全部数据量

【数据层】模糊或查询全部满足条件的雇员数据，多个数据

【数据层】使用Count( )方法对满足条件的数据进行统计

**DAO设计模式：**

DAO设计模式表示数据访问对象，负责完成数据的CRUD，即对数据库进行操作。

DAO设计模式指的是程序分层思想中数据层的设计。

**DAO的组成：**

1. VO类
2. DatabaseConnection类
3. 表接口
4. 表接口的实现类
5. DAO工厂类

**VO类（负责数据传输与包装）：**

在程序分层中，除了数据层要操作SQL之外，其他层操作的数据都是以对象的形式存在。 不同层之间需要进行数据传递，就应该有一个负责传输的数据对象，此对象的类应该为 简单类，这种类在DAO设计模式中称为Value Object，简称VO。这种简单Java类的开 发原则如下：

1. 类名称与表名称一致，且类中的属性与表中的字段相对应。
2. 类中不允许出现基本数据类型，只能使用包装类。
3. 类之所有的属性都必须封装，并添加setter和getter方法。
4. 类中必须存在一个无参构造方法。
5. 类必须实现java.io.Serializable接口。

如果项目的总包名是cn.mldn.oracle，那么VO类保存的包名称应该为cn.mldn.vo

**DatabaseConnection类（负责数据库连接）：**

在数据层的开发中，DatabaseConnection类专门负责取得数据连接和关闭数据连接。

如果项目的总包名是cn.mldn.oracle，DatabaseConnection类要求定义在cn.mldn.dbc包 中。

**DAO接口（数据层开发标准）：**

1）不同层之间的操作依靠的是接口，所以数据层要定义的就是标准。既然是标准， 则需要定义的就是一个接口。

2）通常为每一个表定义一个接口，接口名称为“表名称DAO”；为了从名称上区别接 口和类，建议在接口名称前增加一个字母I，表示Interface的含义。

3）这个接口保存在cn.mldn.dao包中。

4）对于接口的开发主要是针对数据的两种操作（更新、查询），所以从开发标准上对于 命名也有要求，必须严格遵循。基本标准如下：

·更新操作：以“doXxx( )”的方式命名，例如doCreate( )、doUpdate( )、doRemove( )

·查询操作，因为查询操作分为两类：

1. 查询数据：以“findXxx( )”为主，例如findAll( )、findById( )、findByJob( )
2. 统计查询：以“getXxx( )”或“getByXxx( )”为主，例如getAllCount( )、 getByJobCount( )

**DAO工厂类：**

由于不同层之间只能依靠接口取得对象，所以需要定义工厂操作类。

工厂类定义在cn.mldn.factory包之中，名称为DAOFactory。

DAO中类与接口的存放位置：