- Mybatis

1. JDBC
    a. 注册驱动
    b. 获取连接
    c. 执行 sql 语句
    d. 得到结果集，处理结果集 CURD（增，删，改，查）
    e. 关闭资源

```
Connection c = null;
Statement stmt = null;
try {
Class.forName("org.sqlite.JDBC");
c = DriverManager.getConnection("jdbc:sqlite:test.db");
c.setAutoCommit(false);
System.out.println("Opened database successfully");

stmt = c.createStatement();
ResultSet rs = stmt.executeQuery( "SELECT * FROM COMPANY;" );
while ( rs.next() ) {
int id = rs.getInt("id");
String  name = rs.getString("name");
int age  = rs.getInt("age");
String  address = rs.getString("address");
float salary = rs.getFloat("salary");
}
rs.close();
stmt.close();
c.close();
} catch ( Exception e ) {

}
```

2. Mybatis 简介

1) MyBatis 的前身就是 iBatis, iBatis 本是 apache 的一个开源项目，2010 年这个项目由 apahce sofeware foundation 迁移到了 google code，并且改名为 MyBatis。
2) MyBatis 主要完成两件事：
    a. 根据 JDBC 规范建立与数据库的连接；
    b. 通过 Annotaion/XML + JAVA 反射技术，实现 Java 对象与关系数据库之间相互转化

3. Mybatis & hibernate

共同点：

- 从配置文件(通常是 XML 配置文件中)得到 sessionfactory.
- 由 sessionfactory 产生 session
- 在 session 中完成对数据的增删改查和事务提交等.
- 在用完之后关闭 session 。
- 在 java 对象和 数据库之间有做 mapping 的配置文件，也通常是 xml 文件。

a. object/ relational mapping，ORM 代表一种将对象和关系数据库表相互转换的技术

b. 对 Hibernate"O/R"而言

iBATIS 是一种"Sql Mapping"的 ORM 实现。

Hibernate 的 O/R Mapping 实现了 POJO 和数据库表之间的映射，以及 SQL 的自动生成和执行。

程序员往往只需定义好了 POJO 到数据库表的映射关系，即可通过 Hibernate 提供的方法完成持久层操作。程序员甚至不需要对 SQL 的熟练掌握，Hibernate/OJB 会根据制定的存储逻辑，自动生成对应的 SQL 并调用 JDBC 接口加以执行。

Mybatis 的着力点在于 POJO 与 SQL 之间的映射关系。也就是说，Mybatis 并不会为程序员在运行期自动生成 SQL 执行。具体的 SQL 需要程序员编写，然后通过映射配置文 件，将 SQL 所需的参数，以及返回的结果字段映射到指定 POJO。

二者的对比：

1． Mybatis 非常简单易学，Hibernate 相对较复杂，门槛较高。

2．Sql 文件易于管理维护，优化不用改动 java 代码。

3．当系统属于二次开发，无法对数据库结构做到控制和修改, 那 Mybatis 的灵活性将比

Hibernat 更适合。

4. 系统数据处理量巨大，性能要求极为苛刻，这往往意味着我们必须通过经过高度优化的 SQL 语句（或存储过程）才能达到系统性能设计指标。在这种情况下 iBATIS 会有更好的可控性和表现。

5． Hibernate 和 MyBatis 都有相应的代码生成工具。可以生成简单基本的 DAO 层方法。针对高级查询，Mybatis 需要手动编写 SQL 语句，以及 ResultMap。而 Hibernate 有良好的映射机制，开发者无需关心 SQL 的生成与结果映射，可以更专注于业务流程。类似的，如果涉及到数据库字段的修改，Hibernate 修改的地方很少，而 Mybatis 要把那些 sql mapping 的地方一一修改。

6． 以数据库字段一一对应映射得到的 PO 和 Hibernte 这种对象化映射得到的 PO 是截然不同的，本质区别在于这种 PO 是扁平化的，不像 Hibernate 映射的 PO 是可以表达立体的对象继承，聚合等等关系的，这将会直接影响到你的整个软件系统的设计思路。

7．hibernate 数据库无关性比较好，如果数据库有改动，mybatis 要修改较多，而 hibernate 不需要。

4. Mybatis 使用教程
   a) Mybatis 的两个 jar 包，一个 config.xml 配置文件，DAO 层的 java 接口和 sql 映射关系的 xml 文件。
   b) CDATA 标签的使用

      在 XML 元素中，"<" 和 "&" 是非法的。

      "<" 会产生错误，因为解析器会把该字符解释为新元素的开始。

      "&" 也会产生错误，因为解析器会把该字符解释为字符实体的开始。

      包含大量 "<" 或 "&" 字符。为了避免错误，可以将脚本代码定义为 CDATA。

      CDATA 部分中的所有内容都会被解析器忽略，当作普通文本处理。
   c) Like 查询的写法

      *select * from person where name like "%"#{name}"%"*

      *select * from person where name like '%'||#{name}||'%'*

      *select * from person where name like '%${name}%'*

      *select * from person where name like concat('%',#{ name},'%')*
   d) 插入操作获取自增长主键

Oracle(sequence):

```
<insert id="insert" parameterType="User">
        <selectKey keyProperty="id" resultType="int" order="BEFORE">
                select SEQ_TEST_USER_ID.nextval from dual
        </selectKey>
                insert into TEST_USER (ID,NAME,AGE)values (#{id},#{name},#{age})
</insert>
```

Mysql(auto_increament):

```
<sql id='TABLE_NAME'>TEST_USER</sql>
<insert id=”insert” useGeneratedKeys=”true” keyProperty=”id” parameterType="User">
        insert into TEST_USER ( NAME,AGE) values (#{name},#{age})
</insert>
```

或者类似 oracle 的方法：

```
<insert id="insert" parameterType="User">
        <selectKey keyProperty="id" resultType="int" order="BEFORE">
                SELECT LAST_INSERT_ID()
        </selectKey>
        insert into TEST_USER(ID,NAME,AGE)values (#{id},#{name},#{age})
</insert>
```

e) Association 和 Connection

Association:

```
<resultMap type="User" id="userResult">
    <result property="id" column="id"/>
    <result property="userCode" column="userCode" />
    <result property="userName" column="userName" />
    <result property="roleId" column="roleId" />
        <association property="role" javaType="Role" >
      <result property="id" column="id"/>
      <result property="roleCode" column="roleCode"/>
      <result property="roleName" column="roleName"/>
    </association>
</resultMap>
```

```
<select id="getUserListByRoleId" parameterType="Role" resultMap="userResult">
        select u.*,r.roleCode as roleCode,r.roleName as roleName from user u,role r where
   u.roleId = r.id and u.roleId = #{id}
</select>
```

Connection:

获取指定用户的地址列表(user 表-address 表：1 对多关系)

```
<resultMap type="User" id="userMap">
                <id property="id" column="userId"/>
    //property 是属性名 ofType 是 List《Address》的类对象
                <collection property="addressList" ofType="Address">
                <id property="id" column="a_id"/>
                <result property="postCode" column="postCode"/>
                <result property="addressContent" column="addressContent"/>
                </collection>
</resultMap>
```

f) 缓存机制

1．MyBatis 提供了**一级缓存**和**二级缓存**的支持

a. 一级缓存: 存储作用域为 Session，当 Session flush 或 close 之后，该 Session 中的所有 Cache 就将清空，默认的以及缓存是开启的。
b. 二级缓存与一级缓存其机制相同，其存储作用域为 Mapper(Namespace)，并且可自定义存储源，如 Ehcache，二级缓存默认是关闭的 。
c. 对于缓存数据更新机制，当某一个作用域(一级缓存 Session/二级缓存 Namespaces)的进行了 C/U/D 操作后，默认该作用域下所有 select 中的缓存将被 clear。
d. 当执行一条查询 SQL 时，流程为： 从二级缓存中进行查询 -> [如果缓存中没有，委托给 BaseExecutor] -> 进入一级缓存中查询 -> [如果也没有] -> 则执行 JDBC 查询。

一级缓存的使用：

二级缓存的使用：

a. 二级缓存默认为关闭，在 mybatis.xml 中配置:

元素的必需按照如下顺序配置:(properties>>settings>>typeAliases>>typeHandlers>>objectFactory>>objectWrapperFactory>>plugins>>environments>>databaseIdProvider>>mappers)

```
<settings>
<setting name="cacheEnabled" value="true"/>
</settings>
```

b. 当全局的二级缓存（setting 中配置）设置为关闭时可以在 Mapper XML 中配置单个 mapper 的二级缓存为打开，配置如下：

```
<cache />
```

c. 当全局的二级缓存（setting 中配置）设置为打开时，mapper 中这个配置无效，即 mapper 中配置为关闭该 mapper 的二级缓存也是打开。

d. 实体类必需实现序列化接口 Serializable

e. 如果想对某条 SQL 单独对待，可以在 SELECE 语句中配置 useCache,配置如下:

```
<select  id="selectByPrimaryKey" parameterType="string" resultType="User" useCache="false">
```

深入研究缓存：http://www.iteye.com/topic/1112327

g) 利用工具生成 java 实体类，xml 映射文件和接口

需要两个 jar 包（mybatis-generator-core-1.3.2.jar 和 mysql-connector-java-5.1.8-bin.jar），一个 xml 配置文件。

CMD 命令行运行命令:

java -jar C:\workarea.c\eclipse\workspace\CDT_UI3.0\TicketCMS\src\test\mybatis-generator-core-1.3.2.jar –configfile C:\workarea.c\eclipse\workspace\CDT_UI3.0\TicketCMS\src\test\mbgConfiguration.xm -overwrite