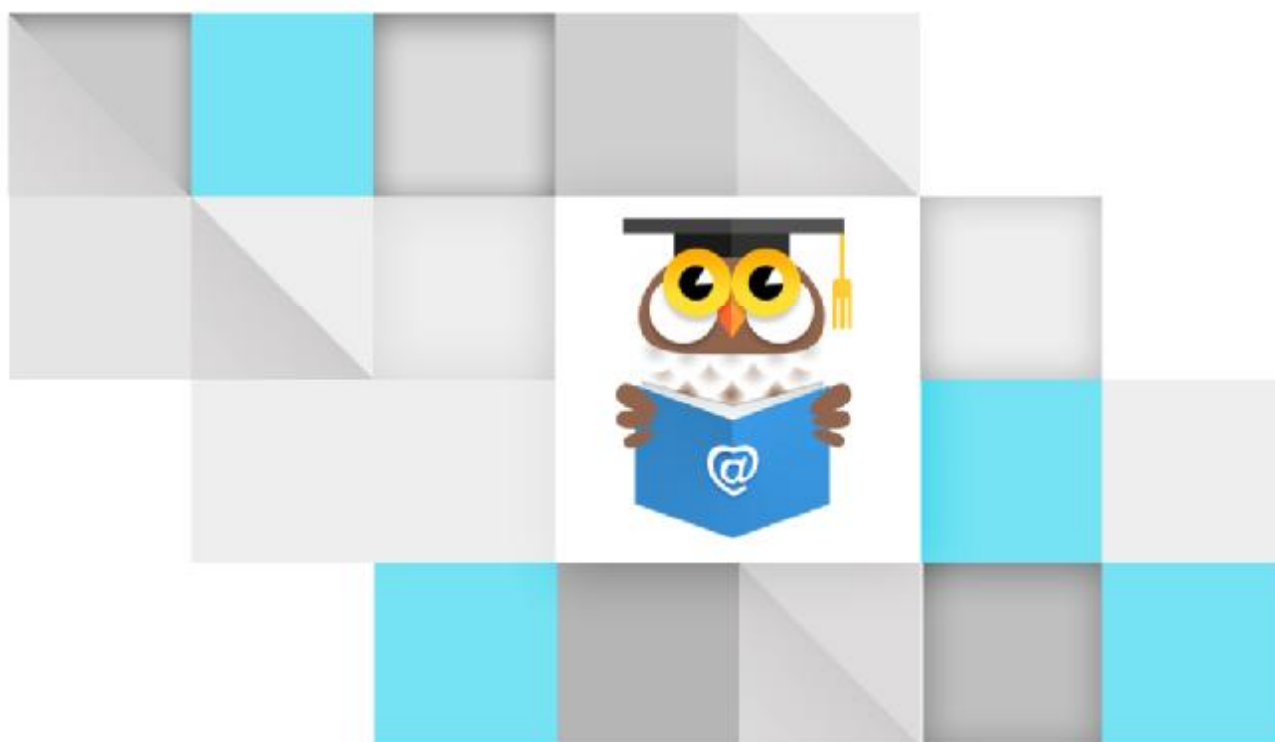


去哪玩旅游网（动态 WEB）

教学指导手册

PRJ-WTP-JEE-009 – 检索页信息查询（EX）



Campus Solution Group

目 录

一、场景说明	1
1、实现效果	1
2、业务描述	1
3、实现思路	2
4、核心组件	2
二、实训技能	3
1、重点演练	3
2、相关技能	4
3、相关知识点	4
4、前置条件	4
5、搭建环境	4
三、场景任务	5
任务 1、创建数据源，并获取数据库连接	5
任务 2、获取搜索的行程数	10
任务 3、分页获取行程集	13

一、场景说明

1、实现效果



图 1-1-1

2、业务描述

本场景用于实现，在《去哪儿玩》旅游网的产品查询页中，根据检索条件检索旅游产品数据。

注意：本场景仅实现“旅游产品按条件检索”业务的数据库访问操作，在界面上显示检索结果的业务，将由后续场景PRJ-WTP-JEE-010中实现（界面显示效果如下图）。



图 1-3-1

2-1. 通过数据源访问《去哪儿玩》旅游网后台数据库：qunawan。

2-2. 获得满足查询条件的记录总数，记录数显示在产品查询页的左上角。

2-3. 分页检索旅游产品（行程）数据，每页显示10条记录。

3、实现思路

3-1. 本场景建议将“旅游产品按条件检索”的数据库访问业务，分为三个任务依次实现：

3-1.1. 任务1. 通过数据源获取《去哪儿玩》旅游网的数据库连接：

1) 配置基于Tomcat容器环境的数据源。

2) 创建数据库访问公共类，并利用数据源（DataSource）获取数据库连接。

3-1.2. 任务2. 基于任务1，访问Trip表，查询满足条件的旅游产品的记录数。

3-1.3. 任务3. 基于任务1，实现分页检索满足条件的旅游产品，每页显示10条记录。

4、核心组件

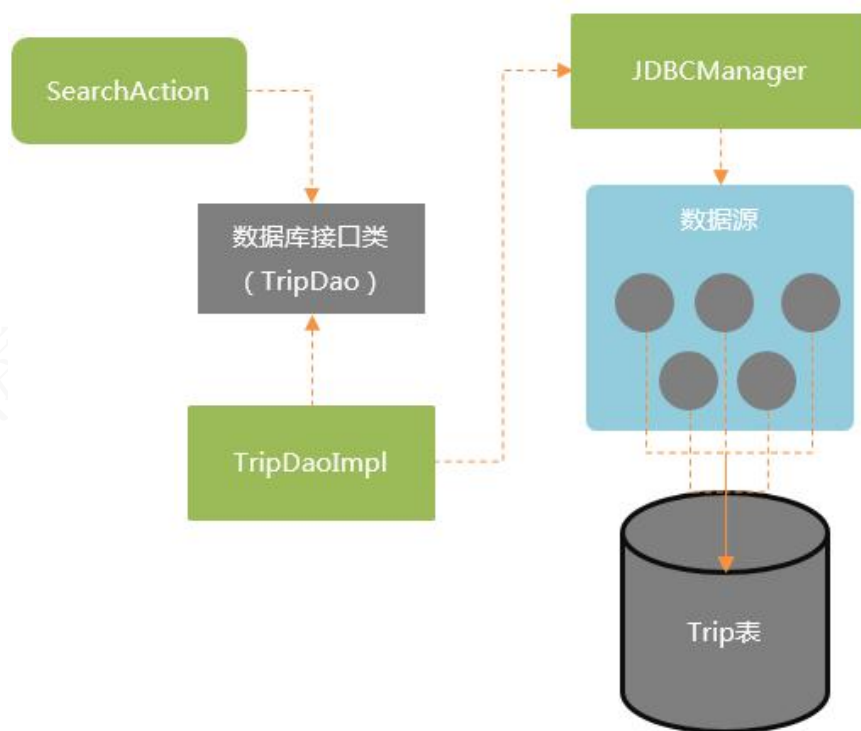


图 1-4-1

4-1. SearchAction (需实现) :

4-1.1. 【旅游产品查询页面】Servlet

4-1.2. 负责访问检索旅游产品的DAO类 (TripDaoImpl) , 获取旅游产品数据。

4-2. JDBCManager (需实现) :

4-2.1. 数据库公共访问类。

4-2.2. 负责数据源对象的创建, 管理数据库连接的获取与归还。

4-3. TripDaoImpl (需实现) :

4-3.1. 旅游产品行程表 (Trip) 的数据访问类。

4-3.2. 获取满足查询条件的行程总数, 用于页面显示以及分页数的计算。

4-3.3. 分页获取满足查询条件的行程对象。

4-4. 数据源 (需实现) :

4-4.1. 通过配置实现的基于Tomcat容器的数据源。

二、实训技能

1、重点演练

1-1. 配置基于Tomcat容器的数据源。

1-2. 创建数据源。

1-3. 通过数据源获取数据库连接。

1-4. 按条件检索数据表记录。

1-5. 分页检索数据库记录。

1-6. 分页页数计算。

2、相关技能

- I 数据源与 JNI 应用

3、相关知识点

- I 配置 Tomcat 数据源
- I 查找数据源
- I 访问数据源

4、前置条件

4-1. 前置场景：无

4-2. 已学技能：

4-2.1. Java开发工具（Eclipse）。

4-2.2. JDBC检索数据库。

4-2.3. 数据源。

5、搭建环境



三、场景任务

任务 1、创建数据源，并获取数据库连接

1. 任务说明：

1-1. 完成效果：

```
Std: Server startup in 2856 ms  
jdbc:mysql://localhost:3306/qunawan, UserName=root@localhost, MySQL-AB JDBC Driver
```

图 3-1-1

1-2. 任务目标：

1-2.1. 配置基于Tomcat容器环境的**数据源**。

1-2.2. 创建数据库访问公共类，并利用**数据源 (DataSource)** 获取数据库连接。

1-3. 任务要求：

1-3.1. 配置基于Tomcat容器环境的**数据源**。

1) 使用context.xml配置数据源。

2) 配置文件固定存放于：**站点根目录/META-INF/**下。

1-3.2. 创建数据库访问公共类，实现利用数据源**获取**与**归还**数据库连接的操作。

1) 限制1. 数据库公共类，命名为：**JDBCManager**。

2) 限制2. 数据库公共类，创建在：**campsg.qunawan.dao.jdbc**包中。

2. 实现思路：

2-1. 实现思路描述的技术结构，如下图（黑点为数据库连接）：

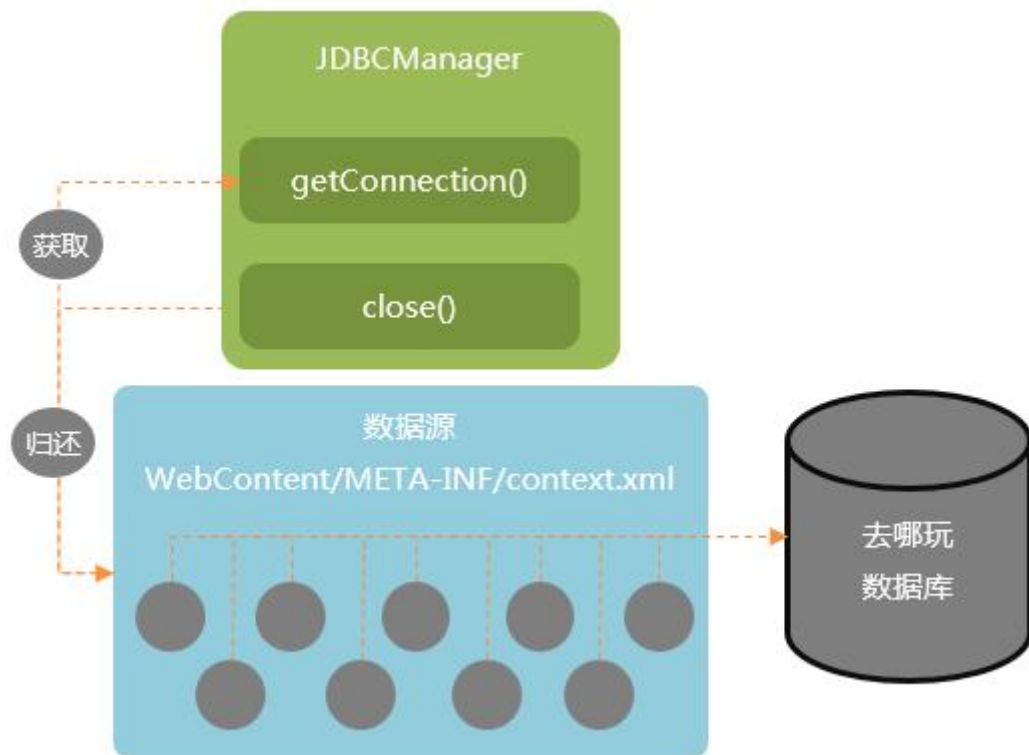


图 3-1-2

2-2. 配置基于Tomcat容器环境的数据源：

2-2.1. 在/**WebContent/META-INF**/目录下新建：context.xml。

2-2.2. 创建<Context></Context>节点。

2-2.3. 在Context节点下创建：<Resource />节点，并依次设置数据源属性如下表：

编号	属性名	属性值	含义
1	name	jdbc/mydb	数据源名称
2	auth	Container	数据源管理者
3	type	javax.sql.DataSource	数据源类型
4	driverClassNam	com.mysql.jdbc.Driver	驱动类
5	url	jdbc:mysql://localhost:3306/qunawan	连接的 URL
6	username	root	数据库用户名
7	password	root	数据库密码
8	maxActive	100	最大活动连接数
9	maxIdle	30	最大限制连接数

10	maxWait	10000	无可用连接最长等待时间
----	---------	-------	-------------

2-3. 创建数据库访问公共类。

2-3.1. 在`campsg.qunawan.dao.jdbc`包中创建类：`JDBCManager`。

2-4. 利用“数据源”获取与归还数据库连接。

2-4.1. 初始化，利用`InitialContext`组件创建数据源：`DataSource`。

2-4.2. 为方便外部调用，定义：用于数据库连接获取的静态函数，形式如下：

```
public static Connection getConnection() {
}

```

2-4.3. 为方便外部调用，定义：用于归还数据库连接的静态函数，形式如下：

```
// 数据库连接归还函数获
// 判断参数类型后调用close方法关闭或归还数据库对象。
public static void close(Object o) {
}

```

2-5. 任务开发流程见下图：

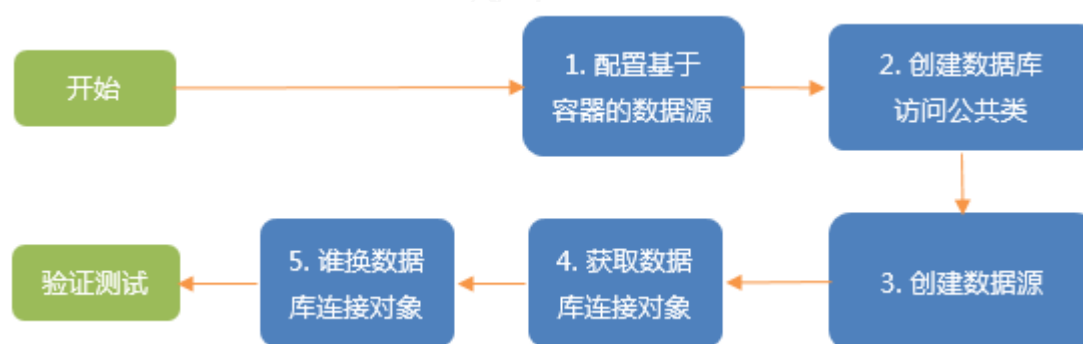


图 3-1-3

3. 推荐步骤：

3-1. 配置数据源：

3-1.1. 创建数据源配置文件`context.xml`：

3-1.2. 创建数据源的`Context`与`Resource`节点：

3-1.3. 按实现思路，依次配置数据源节点`Resource`的属性。

3-2. 创建数据库操作类：**JDBCManager**。

3-3. 获取数据源对象：

3-3.1. 创建静态代码块，加载数据源对象：

- 1) 创建数据源的初始化组件**InitialContext**的实例；
- 2) 通过**InitialContext**实例获取数据源对象（DataSource）；

+ 提示：

1) 静态代码块的语法如下：

```
static {  
  
}
```

注意：静态代码块会在静态函数之前执行，它是类中最先被执行的函数且仅被执行一次。

2) 创建数据源的初始化组件：

```
InitialContext context = new InitialContext();
```

3) 获取数据源对象：

```
DataSource ds = (DataSource) context.lookup("");
```

4) 设置待获取数据源的名称：

4-1. 已知，在context.xml中配置了名为：**jdbc/mydb**的数据源。

4-2. 因此lookup的入参字符串应设置为：**jdbc/mydb**。

注意：创建数据源的字符串格式规定，必须在数据源名前拼接：**java:/comp/env/**字符。

```
DataSource ds = (DataSource) context.lookup("java:/comp/env/jdbc/mydb");
```

3-4. 获取数据库连接对象：

3-4.1. 在JDBCManager中创建静态方法：getConnection；

3-4.2. 通过数据源对象，从数据源获取连接Connection；

3-5. 归还数据库连接对象：

3-5.1. 按实现思路，需在JDBCManager中创建数据库连接归还函数：

- 1) 判断参数的类型是否为ResultSet，是则强转后关闭。
- 2) 判断参数的类型是否为Statement，是则强转后关闭。
- 3) 判断参数的类型是否为Connection，是则强转后关闭（归还）。

4. 验证与测试：

4-1. 获取数据库连接对象：

4-1.1. 在SearchAction的doGet方法获取数据库连接。

- 1) 调用JDBCManager的getConnection函数。
- 2) 将获取的连接对象输出到控制台。

4-2. 运行项目工程。

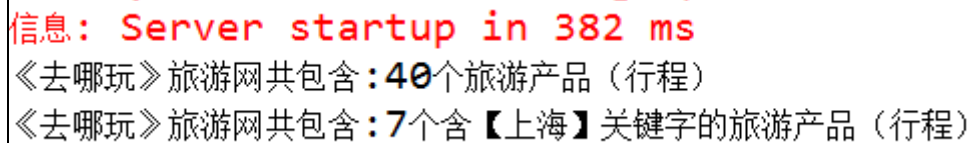
4-3. 并在浏览器地址栏输出 http://localhost/PRJ_WTP_JEE_009/ 进行访问

4-4. 应见【任务完成效果】截图。

任务 2、获取搜索的行程数

1. 任务说明：

1-1. 完成效果：



```
信息: Server startup in 382 ms
《去哪玩》旅游网共包含: 40个旅游产品 (行程)
《去哪玩》旅游网共包含: 7个含【上海】关键字的旅游产品 (行程)
```

图 3-2-1

1-2. 任务目标：

1-2.1. 在任务1的基础上，访问Trip表，查询满足条件的旅游产品的记录数。

1-3. 任务要求：

1-3.1. 在TripDaoImpl类的getTripNum中，实现按条件检索旅游产品记录数的业务。

1) 说明：getTripNum方法的page参数：翻页的页码。

2) 说明：getTripNum方法的返回值：满足条件的记录数。

1-3.2. 按用户输入的检索条件，检索满足条件的旅游产品记录数：

1) 当用户输入检索条件（例如：上海）后执行检索操作时，应该只返回满足检索条件的记录数。

2) 当用户未输入关键字执行检索操作时，应该返回全部旅游产品的记录数。

3) SQL语句拼接规则：用户输入的检索条件应与Trip表中的title字段模糊匹配。

2. 实现思路：

2-1. 检索满足条件的旅游产品数量：

2-1.1. 在TripDaoImpl类的getTripNum中，通过JDBCManager获取数据库连接。

2-1.2. 使用SQL语句中的count函数，获取旅游产品的记录数。

2-1.3. SQL语句的Where条件应确保：Trip表的title字段与检索条件模糊匹配：

- 1) 用户未输入关键字，不设置查询条件（查询全部记录）。
- 2) 用户输入了关键字，设置查询条件（查询满足条件的记录）。

2-2. 任务开发流程见下图：

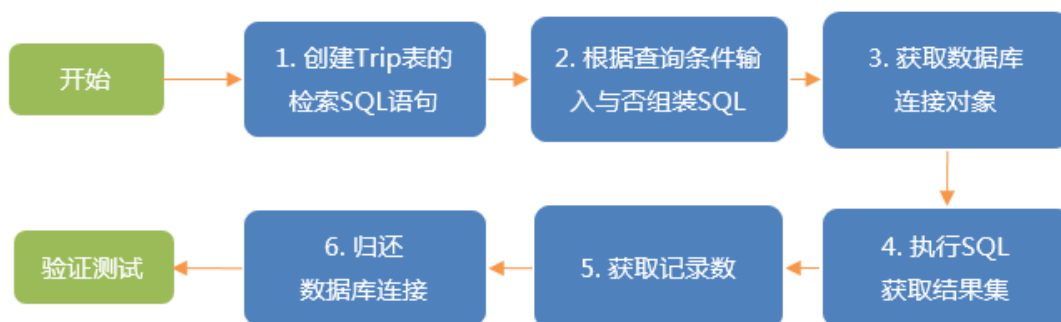


图 3-2-2

3. 推荐步骤：

3-1. 在TripDaoImpl的getTripNum方法中，创建查询的SQL语句：

3-1.1. 利用count(*)函数检索数据库，获取记录数。

3-1.2. 根据用户是否输入检索条件，设置Where语句。

- 1) 判断getTripNum方法参数（用户输入的条件）是否为null。
- 2) 判断getTripNum方法参数（用户输入的条件）是否为空（""）。
- 3) 两个条件都不满足，表示用户输入了查询条件，则在SQL语句尾部追加title字段的模糊匹配条件：like '%' + key + '%'。
- 4) 反之，表示用户没有输入查询条件，则SQL语句尾部不追加查询条件。

+ 提示：

1) 一个常用的按需组装SQL语句的技巧：

1-1. SQL语句固定设置为：SELECT 字段集 FROM 数据表 WHERE 1=1；

说明：1=1虽然没有意义，但是却可省略判断是否增加WHERE关键的代码。

1-2. 随后依次判断查询条件是否为空（""）和null。

1-3. 满足条件，不增加查询条件。

1-4. 不满足条件，增加查询条件。

例如：

```
String sql = "SELECT * FROM MyTable WHERE 1=1" ;
```

```
if (key !=null && !"".equals(key))
```

```
    sql += "and title like '%" + key + "%'";
```

3-2. 通过JDBCManager获取数据库连接对象。

3-3. 执行SQL语句。

3-3.1. 通过PreparedStatement执行SQL语句。

3-3.2. 通过PreparedStatement返回结果集（ResultSet）。

3-3.3. 获取结果集中第一条记录的第一个字段，其固定为int类型，该字段就是满足条件的记录数量。

3-4. 调用JDBCManager类中的close函数，归还数据库连接。

4. 验证与测试：

4-1. 在SearchAction中编写测试语句：

4-1.1. 实例化TripDaoImpl类。

4-1.2. 调用getTripNum方法，参数设置为null。

4-1.3. 向控制台输出：《去哪玩》旅游网共包含：X个旅游产品（行程）。

4-1.4. 调用getTripNum方法，参数设置为上海。

4-1.5. 向控制台输出：《去哪玩》旅游网共包含：X个含【上海】关键字的旅游产品（行

程)。

说明：X为getTripNum方法的返回结果。

4-2. 运行项目工程，控制台应显示与【任务完成效果】截图匹配的结果。

任务 3、分页获取行程集

1. 任务说明：

1-1. 完成效果：

显示含【上海】关键字的，第1页旅游产品：

```
Trip [id=2, num=0, title=【上海松江1天1夜】住上海旗山大酒店1晚，免费游佘山森林公园，
Trip [id=3, num=0, title=【享受住宿温泉“零”距离，上海2天1晚】住上海之根雪浪湖度假，
Trip [id=10, num=0, title=【激情燃烧，欢乐谷2天1晚】住1晚上海松江世茂睿选酒店，嗨翻，
Trip [id=23, num=0, title=上海欢乐谷4, s_title=null, traffic=大巴, hot
Trip [id=24, num=0, title=上海野生动物园, s_title=null, traffic=轮船, hc
Trip [id=30, num=0, title=上海-厦门双飞4日自由行（世纪寰岛酒店 赠接机服务），s_tit
Trip [id=35, num=0, title=塞班岛5晚6日半自助游（已含税上海直飞入住海景酒店或黄金海滩
```

图 3-3-1

1-2. 任务目标：

1-2.1. 在任务1的基础上，实现分页检索满足条件的旅游产品，每页显示10条记录。

1-3. 任务要求：

1-3-1. 在TripDaoImpl类的getTripByCondition中，实现分页检索满足条件的旅游产品的业务。

1) 说明：getTripByCondition方法的key参数：检索条件。

2) 说明：getTripByCondition方法的page参数：翻页的页码。

3) 说明：getTripByCondition方法的返回值：满足条件的对象集。

1-3-2. 按用户输入的检索条件，分页检索满足条件的旅游产品记录：

1) 由任务2的任务要求可知，查询旅游产品记录，应满足用户输入的检索条件。

2) 在任务2的基础上，需要实现分页检索，每页显示10条记录。

1-3-3. 例如：检索含【上海】关键字的旅游产品共包含12条记录，分页规则如下：

- 1) 第1页，显示第0到第9条记录（0为起始记录）。
- 2) 第2页，显示第10到11条记录（10为起始记录）。
- 3) 第3页，无显示记录。

1-3-4. `getTripByCondition`方法获取记录后，需将每条记录封装入Trip实体类中。

1-3-5. Trip实体类需要被加入集合中，并从`getTripByCondition`方法中返回。

2. 实现思路：

2-1. 实现分页检索满足条件的旅游产品的业务：

2-1.1. 按翻页页码计算数据表的起始记录号：

- 1) 一页显示10条记录，起始记录号 = (翻页页码 - 1) * 10。
- 2) 即，第1页，从第0条记录开始，一共显示10条记录。
- 3) 即，第2页，从第10条记录开始，一共显示10条记录。

2-1.2. 根据用户输入的检索条件检索旅游产品记录，同任务2的实现思路。

2-1.3. SQL语句实现MySQL数据表的翻页查询。

- 1) MySQL数据库允许使用limit关键实现翻页检索。
- 2) 语法：SELECT * FROM MyTable WHERE id limit 0,10

注意：limit前的字段，一般为表的主键。

注意：limit后的数据，第1位是起始记录号，第2位是获取的记录总数。

注意：当有多个条件设置在WHERE语句后，limit语句始终处于SQL的末尾。

2-2. 封装Trip对象，Trip实体与Trip表的对应关系如下表：

编号	数据表字段	类的属性	字段含义
1	id	setId()	Trip 表的主键，无意义
2	title	setTitle()	旅游产品的标题

3	time	setTime()	行程包含的天数
4	traffic	setTraffic()	行程包含的交通
5	hotel	setHotel()	行程包含的酒店
6	good_rate	setGood_rate()	好评率

2-3. 创建Trip实体集合，本任务可考虑使用ArrayList类型保存Trip实体。

2-4. 任务开发流程见下图：

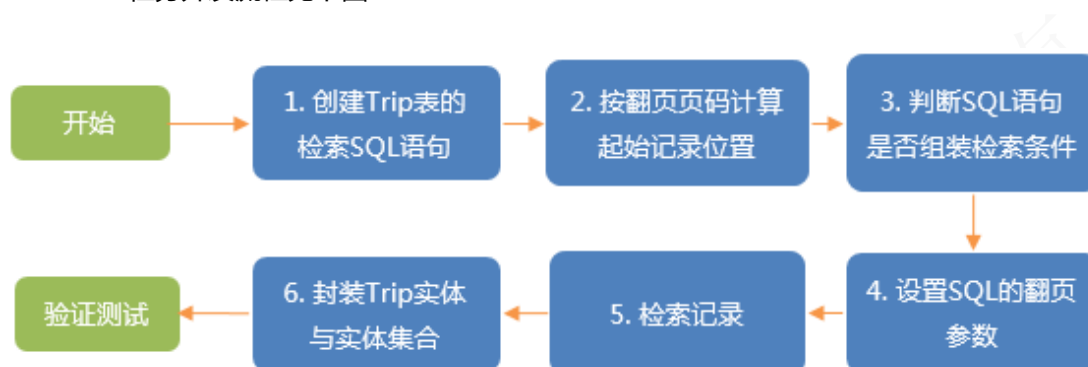


图 3-3-2

3. 推荐步骤：

3-1. 在TripDaoImpl类的getTripByCondition中，实现分页检索满足条件的旅游产品。

3-2. 按翻页页码计算起始记录。

+ 提示：

1) 由实现思路可知，翻页起始记录的计算公式为：

```
int start = (page - 1) * 10。
```

3-3. 根据用户输入的检索条件检索旅游产品记录，同任务2的推荐步骤。

3-4. SQL语句实现MySQL数据表的翻页查询

+ 提示：

1) 翻页SQL语句如下：

```
SELECT * FROM Trip
```

WHERE 1=1 AND 按检索条件查询 AND id limit 起始记录号 , 10

3-5. 检索记录：

3-5.1. 通过JDBCManager获得数据库连接。

3-5.2. 通过PreparedStatement对象执行SQL语句。

3-5.3. 获取ResultSet结果。

3-6. 封装Trip实体类与实体集合：

3-6.1. 循环ResultSet，依次封装每个Trip实体对象，封装方法见实现思路中的对应表。

3-6.2. 将封装完的实体加入实体集合中。

3-7. 归还数据库连接。

3-8. 返回实体集合。

4. 验证与测试：

4-1. 在SearchAction中编写测试语句：

4-1.1. 实例化TripDaoImpl类。

4-1.2. 调用getTripByCondition方法，参数1设置为：上海，参数2设置为：1。

4-1.3. 获取检索记录集。

4-1.4. 向控制台输出：显示含【上海】关键字的，第1页旅游产品：

4-1.5. 循环getTripByCondition方法返回的结果集，打印每个Trip对象。

说明：直接打印Trip对象即可，无需逐个属性打印。

4-2. 运行项目工程，控制台应显示与【任务完成效果】截图匹配的结果。

4-3. 有兴趣的用户可以尝试修改getTripByCondition方法的参数，将看到不同的结果。

作者：Fox.Sun