**分页：**

分页是一个网站系统中非常重要也十分常用的功能，在 MVC 开发模式下我们通常是把它放入返回对象中并在页面代码中循环遍历并渲染到页面中，也有通过接口返回，并通过前端插件来实现，这两种方式我们都会介绍到，将其设计为一个通用的分页接口，并将分页数据放到 Result 对象中并通过 json 格式返回。

**分页的作用：**

减少系统资源的消耗，数据查询出来后是放在内存里的，如果在数据量很大的情况下一次性将所有内容都查询出来，会占用过多的内存，通过分页可以减少这种消耗；

提高性能，应用与数据库间通过网络传输数据，一次传输 10 条数据结果集与一次传输 20000 条数据结果集肯定是传输 10 条消耗更少的网络资源；

提升访问速度，浏览器与应用间的传输也是通过网络，返回 10 条数据明显那比返回 20000 条数据速度更快，因为数据包的大小有差别；

符合用户习惯，比如搜索结果或者商品展示，通常用户可能只看最近前 30 条，将所有数据都查询出来比较浪费；

基于展现层面的考虑，由于设备屏幕的大小比较固定，一个屏幕能够展示的信息并不是特别多，如果一次展现太多的数据，不管是排版还是页面美观度都有影响，一个屏幕的范围就是那么大，展示信息条数有限。

分页功能的使用可以提升系统性能，也比较符合用户习惯，符合页面设计，这也是为什么大部分系统都会有分页功能。

**分页设计：**

**分页参数设计：**

前端分页区比较重要的几个信息是：

页码展示

当前页码

每页条数

当然，有些页面也会加上首页、尾页、跳转页码等功能，这些信息都根据功能需要和页面设计去做增加和删减。

后端功能设计

前端页面的工作是渲染数据和分页信息展示，而后端则需要按照前端传输过来的请求将分页所需的数据正确的查询出来并返回给前端，两端的侧重点并不相同，比如前端需要展示所有页码，而后端则只需要提供总页数即可，并不需要对这个总页码进行其他操作，比如前端需要根据用户操作记录当前页码这个参数以便对页码信息进行调整和限制，而后端则并不是这么注重当前页码，只需要接收前端传输过来的页码进行相应的判断和查询操作即可。

对于后端必不可少的是两个参数：

页码(需要第几页的数据)

每页条数(每次查询多少条数据,一般默认 10 条或者 20 条)

因为数据库查询语句如下，不同数据库可能关键字有些差别，比如 SQL Server 是通过 top 关键字、Oracle 通过 rownum 关键字，MySQL 实现分页功能基本都是使用 limit 关键字。

//下面是mysql的实现语句：

select \* from tb\_xxxx limit 10,20

分页功能的最终实现既是如此，通过页码和条数确定数据库需要查询的是从第几条到第几条的数据，比如查询第 1 页每页 20 条数据就是查询数据库中从 0 到 20 条数据，查询第 4 页每页 10 条数据就是查询数据库中第 30 到 40 条数据，因此对于后端来说页码和条数两个参数就显得特别重要，缺少这两个参数根本无法继续之后的查询逻辑，分页数据也就无从查起。

虽然如此，为了前端分页区展示还要将数据总量或者总页数返回给前端，数据总量是必不可少的，因为总页数可以计算出来，即数据总量除以每页条数，数据总量的获取方式：

select count(\*) from tb\_xxxx

之后将数据封装，并返回给前端即可。

**新增加数据：**

CREATE TABLE `admin\_user` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`user\_name` varchar(50) NOT NULL,

`password\_md5` varchar(50) NOT NULL,

`user\_token` varchar(50) NOT NULL,

`is\_deleted` tinyint(4) DEFAULT '0',

`create\_time` date NOT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `admin\_user`(`id`,`user\_name`,`password\_md5`,`user\_token`,`is\_deleted`,`create\_time`) VALUES (1,'admin','e10adc3949ba59abbe56e057f20f883e','d87edfdd63674b9591602b26bfb7f93f',0,'2018-07-04 11:21:14'),(2,'test2','098f6bcd4621d373cade4e832627b4f6','\'\'',0,'2018-07-09 17:22:28'),(3,'test3','098f6bcd4621d373cade4e832627b4f6','\'\'',0,'2018-07-09 17:22:32'),(4,'test4','098f6bcd4621d373cade4e832627b4f6','\'\'',0,'2018-07-09 17:22:32'),(5,'test5','098f6bcd4621d373cade4e832627b4f6','\'\'',0,'2018-07-09 17:22:33'),(6,'test6','098f6bcd4621d373cade4e832627b4f6','\'\'',0,'2018-07-09 17:22:34'),(7,'test7','098f6bcd4621d373cade4e832627b4f6','\'\'',0,'2018-07-09 17:22:35')；

**后端分页功能代码：**

**AdminUserDao.xml：**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN" "<http://mybatis.org/dtd/mybatis-3-mapper.dtd>">

<mapper namespace="com.project.springboot.dao.AdminUserDao">

<resultMap type="com.lou.springboot.entity.AdminUser" id="AdminUserResult">

<id property="id" column="id" jdbcType="BIGINT"/>

<result property="userName" column="user\_name" jdbcType="VARCHAR"/>

<result property="password" column="password\_md5" jdbcType="VARCHAR"/>

<result property="userToken" column="user\_token" jdbcType="VARCHAR"/>

<result property="isDeleted" column="is\_deleted" jdbcType="TINYINT"/>

<result property="createTime" column="create\_time" jdbcType="TIMESTAMP"/>

</resultMap>

<insert id="addUser" parameterType="com.lou.springboot.entity.AdminUser">

insert into tb\_admin\_user(user\_name,password\_md5)

values(#{userName},#{password})

</insert>

<insert id="insertUsersBatch">

insert into tb\_admin\_user(user\_name,password\_md5) VALUES

<foreach collection="adminUsers" index="index" item="adminUser" open="" separator="," close="">

(#{adminUser.userName}, #{adminUser.password})

</foreach>

</insert>

<select id="findAdminUsers" parameterType="Map" resultMap="AdminUserResult">

select id,user\_name,create\_time from tb\_admin\_user

where is\_deleted=0

order by id desc

<if test="start!=null and limit!=null">

limit #{start},#{limit}

</if>

</select>

<select id="getTotalAdminUser" parameterType="Map" resultType="int">

select count(\*) from tb\_admin\_user

where is\_deleted=0

</select>

<select id="getAdminUserByUserNameAndPassword" resultMap="AdminUserResult">

select id,user\_name,user\_token

from tb\_admin\_user

where user\_name = #{userName} and password\_md5 = #{passwordMD5}

and is\_deleted = 0

ORDER BY id DESC limit 1

</select>

<select id="getAdminUserByToken" resultMap="AdminUserResult">

select id,user\_name,user\_token

from tb\_admin\_user

where user\_token = #{userToken}

and is\_deleted = 0

ORDER BY id DESC limit 1

</select>

<select id="getAdminUserById" resultMap="AdminUserResult">

select user\_name,user\_token

from tb\_admin\_user

where id=#{id}

and is\_deleted = 0

ORDER BY id DESC limit 1

</select>

<select id="getAdminUserByUserName" resultMap="AdminUserResult">

select id,user\_name,user\_token

from tb\_admin\_user

where user\_name = #{userName}

and is\_deleted = 0

ORDER BY id DESC limit 1

</select>

<update id="updateUserToken">

update tb\_admin\_user set user\_token = #{newToken} where id =#{userId} and is\_deleted = 0

</update>

<update id="updateUserPassword">

update tb\_admin\_user set password\_md5 = #{newPassword},user\_token ='' where id =#{userId} and is\_deleted = 0

</update>

<update id="deleteBatch">

update tb\_admin\_user

set is\_deleted=1 where id in

<foreach item="id" collection="array" open="(" separator="," close=")">

#{id}

</foreach>

</update>

<select id="getAllAdminUsers" resultMap="AdminUserResult">

select id,user\_name,is\_deleted,create\_time from tb\_admin\_user

order by id desc

</select>

</mapper>

**业务层代码：**