

第8章 图形报表、Excel报表

- 了解Echarts
- 掌握Echarts实现会员数量折线图的实现过程
- 掌握Echarts实现套餐预约占比饼形图的实现过程
- 掌握运营数据统计的实现过程
- 掌握运营数据统计报表导出的实现过程
- 掌握报表sql语句的编写步骤

1. 图形报表ECharts

【目标】

了解Echarts

【路径】

- 1: ECharts简介
- 2: 5分钟上手ECharts
- 3: 查看ECharts官方实例

【讲解】

1.1. ECharts简介

ECharts缩写来自Enterprise Charts，商业级数据图表，是百度的一个开源的使用JavaScript实现的数据可视化工具，可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器（IE8/9/10/11，Chrome，Firefox，Safari等），底层依赖轻量级的矢量图形库 [ZRender](#)，提供直观、交互丰富、可高度个性化定制的数据可视化图表。

官网: <https://echarts.baidu.com/>

下载地址: <https://echarts.baidu.com/download.html>



下载完成可以得到如下文件：



解压上面的zip文件：



我们只需要将dist目录下的echarts.js文件（已经压缩）引入到页面上就可以使用了



1.2. 5分钟上手ECharts

我们可以参考官方提供的5分钟上手ECharts文档感受一下ECharts的使用方式，地址如下：

<https://www.echartsjs.com/tutorial.html#5%20%E5%88%86%E9%92%9F%E4%B8%8A%E6%89%8B%20Echarts>

第一步：在health_web中创建html页面并引入echarts.js文件



echartsDemo.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Title</title>
6 </head>
7 <!-- 引入 ECharts 文件 -->
8 <script src="js/echarts.js"></script>
9 <body>
10
11 </body>
12 </html>
```

第二步：在页面中准备一个具备宽高的DOM容器。

```
1 <body>
2     <!-- 为 ECharts 准备一个具备大小（宽高）的 DOM -->
3     <div id="main" style="width: 600px;height:400px;"></div>
4 </body>
```

第三步：通过echarts.init方法初始化一个 echarts 实例并通过setOption方法生成一个简单的柱状图

```
1 <script type="text/javascript">
2     // 基于准备好的dom，初始化echarts实例
3     var myChart = echarts.init(document.getElementById('main'));
4
5     // 指定图表的配置项和数据
6     var option = {
7         title: {
8             text: 'Echarts 入门示例'
9         },
10        tooltip: {},
11        legend: {
12            data: ['销量']
13        },
14        xAxis: {
15            data: ["衬衫", "羊毛衫", "雪纺衫", "裤子", "高跟鞋", "袜子"]
16        },
17        yAxis: {},
18        series: [{
19            name: '销量',
20            type: 'bar',
21            data: [5, 20, 36, 10, 10, 20]
22        }]
23    };
24    // 使用刚指定的配置项和数据显示图表。
25    myChart.setOption(option);
26 </script>
```

效果如下：



1.3. 查看ECharts官方实例

ECharts提供了很多官方实例，我们可以通过这些官方实例来查看展示效果和使用方法。

官方实例地址：<https://www.echartsjs.com/examples/>



可以点击具体的一个图形会跳转到编辑页面，编辑页面左侧展示源码（js部分源码），右侧展示图表效果，如下：



要查看完整代码可以点击右下角的Download按钮将完整页面下载到本地。

通过官方案例我们可以发现，使用ECharts展示图表效果，关键点在于确定此图表所需的数据格式，然后按照此数据格式提供数据就可以了，我们无须关注效果是如何渲染出来的。

在实际应用中，我们要展示的数据往往存储在数据库中，所以我们可以发送ajax请求获取数据库中的数据并转为图表所需的数据即可。

【小结】

1: ECharts简介

- ECharts百度开发的一套商业报表工具(库)，数据加载使用JavaScript

2: 5分钟上手ECharts 引入js, 创建一个div, 编写js代码(初始化, 配置信息, 设置配置信息) xAxis data, yAxis data

3: 查看ECharts官方实例

学习方法

- 看官网, 关注案例, 无需关注api, 根据项目需求, 选择合适报表, 从数据库中构造需要的数据
- 页面加载时发送请求获取数据, 再设置echarts的配置项

2. 会员数量折线图

【目标】

使用ECharts实现注册会员数量折线图

【路径】

1: 需求分析

2: 前台代码

(1) 导入ECharts库

(2) 参考官方实例导入折线图

3: 后台代码

(1) ReportController类 getMemberReport

- 创建12个月的数据
- 使用Calendar 对日期操作 对年-1, 循环12次
- 每次+1个月, 就得到12个月 List months 2020-01
- 调用memberService 查询12个月的数据 List memberCount
- 把得到的12个月的月份数据及统计数据封装到map
- result{data: {months, memberCount}}

(2) MemberService服务接口

(3) MemberServiceImpl服务实现类

- 遍历12个月, 每个月拼接-31, 查询, 返回的数据添加到List
- 返回List

(4) MemberDao接口

(5) Mapper映射文件 (MemberDao.xml) , 大小于号要转义

```
1  -- 大于号转义 &gt;      小于号转义 &lt;
2  select count(1) from t_member where regTime <='2020-09-31'
```

【讲解】

2.1. 需求分析

会员信息是体检机构的核心数据, 其会员数量和增长数量可以反映出机构的部分运营情况。通过折线图可以直观的反映出会员数量的增长趋势。本章节我们需要展示过去一年时间内每个月的会员总数据量。展示效果如下图:

到某个月最后一天为止时, 会员总数量

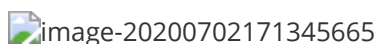


需要的sql:


```
1  #2019年05月31日之前注册会员的人数(11)
2
3  SELECT COUNT(id) FROM t_member WHERE regTime <= '2019-05-31'
4
5  #2019年06月31日之前注册会员的人数(12)
6
7  SELECT COUNT(id) FROM t_member WHERE regTime <= '2019-06-31'
8
9  #2019年07月31日之前注册会员的人数(16)
10
11 SELECT COUNT(id) FROM t_member WHERE regTime <= '2019-07-31'
```

2.2. 前台代码

会员数量折线图对应的页面为/pages/report_member.html。复制资料中的



到health_web中webapp/pages下


image-20200702171428753

2.2.1. 导入ECharts库

第一步：将echarts.js文件复制到health_web工程中

image-20200702171506435

第二步：在report_member.html页面引入echarts.js文件

image-20200702171528489

2.2.2. 参照官方实例导入折线图

1: 定义

```
1 <div class="app-container">
2   <div class="box">
3     <!-- 为 ECharts 准备一个具备大小（宽高）的 DOM -->
4     <div id="chart1" style="height:600px;"></div>
5   </div>
6 </div>
```

2: 定义script

```
1 <script type="text/javascript">
2   // 基于准备好的dom，初始化echarts实例
3   var myChart1 = echarts.init(document.getElementById('chart1'));
4
5   // 使用刚指定的配置项和数据显示图表。
6   //myChart.setOption(option);
7
8   axios.get("/report/getMemberReport.do").then((res)=>{
9     if(res.data.flag) {
10       myChart1.setOption(
11         {
12           title: {
13             text: '会员数量'
14           },
15           tooltip: {},
16           legend: {
17             data: ['会员数量']
18           },
19           xAxis: {
20             data: res.data.data.months
21           },
22           yAxis: {
23             type: 'value'
24           },
25           series: [{
26             name: '会员数量',
27             type: 'line',
28             data: res.data.data.memberCount
29           }]
30         }
31       );
32     }
33   });
```

```

31         }else{
32             this.$message.error(res.data.message);
33         }
34     });
35 </script>

```

根据折线图对数据格式的要求，我们发送ajax请求，服务端需要返回如下格式的数据：

其中months和memberCount可以使用hashmap封装

返回Result: data、flag、message

其中data，是map集合

map集合的key: months; map集合的值: List

map集合的key: memberCount; map集合的值: List

```

1  {
2      "data":{
3          "months":["2019-01","2019-02","2019-03","2019-04"],
4          "memberCount":[3,4,8,10]
5      },
6      "flag":true,
7      "message":"获取会员统计数据成功"
8  }

```

2.3. 后台代码

2.3.1. Controller

在health_web工程中创建ReportController并提供getMemberReport方法

```

1  package com.itheima.health.controller;
2
3  import com.alibaba.dubbo.config.annotation.Reference;
4  import com.itheima.health.constant.MessageConstant;
5  import com.itheima.health.entity.Result;
6  import com.itheima.health.service.MemberService;
7  import org.springframework.web.bind.annotation.RequestMapping;
8  import org.springframework.web.bind.annotation.RestController;
9
10 import java.text.SimpleDateFormat;
11 import java.util.*;
12
13 /**
14  * 统计报表
15  */
16 @RestController
17 @RequestMapping("/report")
18 public class ReportController {
19     @Reference
20     private MemberService memberService;
21
22     /**
23      * 会员折线图
24      * @return

```

```

25     */
26     @GetMapping("/getMemberReport")
27     public Result getMemberReport(){
28         // 组装过去12个月的数据，前端是一个数组
29         List<String> months = new ArrayList<String>();
30         // 使用java中的calendar来操作日期，日历对象
31         Calendar calendar = Calendar.getInstance();
32         // 设置过去一年的时间 年-月-日 时:分:秒，减去12个月
33         calendar.add(Calendar.MONTH, -12);
34         SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM");
35         // 构建12个月的数据
36         for (int i = 0; i < 12; i++) {
37             // 每次增加一个月就
38             calendar.add(Calendar.MONTH, 1);
39             // 过去的日期，设置好的日期
40             Date date = calendar.getTime();
41             // 2020-06 前端只展示年-月
42             months.add(sdf.format(date));
43         }
44
45         // 调用服务查询过去12个月会员数据 前端也是一数组 数值
46         List<Integer> memberCount =memberService.getMemberReport(months);
47         // 放入map
48         Map<String,Object> resultMap = new HashMap<String,Object>();
49         resultMap.put("months",months);
50         resultMap.put("memberCount",memberCount);
51
52         // 再返回给前端
53         return new Result(true,
54             MessageConstant.GET_MEMBER_NUMBER_REPORT_SUCCESS,resultMap);
55     }

```

2.3.2. 服务接口

在MemberService服务接口中扩展方法findMemberCountByMonth

```

1     /**
2     * 统计过去一年的会员总数
3     * @param months
4     * @return
5     */
6     List<Integer> getMemberReport(List<String> months);

```

2.3.3. 服务实现类

在MemberServiceImpl服务实现类中实现findMemberCountByMonth方法

```

1     /**
2     * 统计过去一年的会员总数
3     * @param months
4     * @return

```

```

5    */
6    @Override
7    public List<Integer> getMemberReport(List<String> months) {
8        //select count(1) from t_member where regTime<='2020-06-31';  <= Before
9        List<Integer> memberCount = new ArrayList<Integer>();
10       if(months != null) {
11           // 循环12个，每个月查询一下
12           for (String month : months) {
13               // 到这个month为，一会有多少个会员
14               Integer count = memberDao.findMemberCountBeforeDate(month +
15               "-31");
16               memberCount.add(count);
17           }
18       }
19       return memberCount;
20   }

```

2.3.4. Dao接口

在MemberDao接口中扩展方法findMemberCountBeforeDate

```

1 public Integer findMemberCountBeforeDate(String date);

```

2.3.5. Mapper映射文件

在MemberDao.xml映射文件中提供SQL语句

```

1 <!--根据日期统计会员数，统计指定日期之前的会员数-->
2 <select id="findMemberCountBeforeDate" parameterType="string"
3     resultType="int">
4     select count(id) from t_member where regTime <= #{value}
5 </select>

```

```

1 注意：在xml文件中，    <号需要转义"&lt;"
2                        >号需要转义"&gt;"
3                        &号需要专业 "&amp;"

```

测试

1: 修改main.html

 image-20200702171856199

2: 测试效果

 img

【小结】

1: 需求分析

会员数量统计， t_member。每个月的会员【总】数量。到某个月的最后一天为止时会员表中总数量

2: 前台页面

(1) 导入ECharts库

(2) 参考官方实例导入折线图案例（折线图、柱状图、饼图...），参考API完成开发

res.data.data.months => [] => List

res.data.data.memberCount => [] => List

3: 后台代码

(1) ReportController类

计算出过去12个月，利用Calendar日历工具对月份-12，再循环12次，每次加1个月

(2) MemberService服务接口

(3) MemberServiceImpl服务实现类

(4) MemberDao接口

(5) Mapper映射文件（MemberDao.xml）

3.套餐预约占比饼形图

3.1. 需求分析

【目标】

会员可以通过移动端自助进行体检预约，在预约时需要选择预约的体检套餐。本章节我们需要通过饼形图直观的展示出会员预约的各个套餐占比情况。展示效果如下图：



【路径】

分析：

写sql报表的步骤

1. 找出数据所在的表及其字段 t_setmeal.name, t_order
2. 找出条件所在表 没条件
3. 找出数据表之间的关系，如果没有直接关系，则找中间表 t_setmeal.id=t_order.setmeal_id
4. 找出条件所在表之间的关系，如果没有直接关系，则找中间表
5. 找出数据表与条件表之间的关系，如果没有直接关系，则找中间表
6. 如果是多条合成一条，就用分组统计

```
1 -- 统计每个套餐的数量，同一个套餐有多条记录
2 -- 每个套餐只有一条记录，把多条合成一条记录。用聚合函数，就得用分组group by
3 -- 语法不严格的写法，mysql5.7以下可以这样写，5.7以上就会报错
4 -- 严格的语法：分组统计时，select后的列必须在group by后出现过的
5 -- 这里的查询出来的列名必须与前端需要的数据格式一致(data:[{name:?,value:0}])
6 select s.name,count(1) value from t_setmeal s, t_order o
7 where s.id=o.setmeal_id group by s.id,s.name
```

1: 前台页面

(1) 修改main.html

(2) 导入ECharts库

(3) 参照官方实例导入饼形图

(4) 分析需要构造的数据格式和sql语句

```
1 {  
2     flag:true,  
3     message:"",  
4     data:{  
5         setmealNames:List<String>,  
6         setmealCount:List<map<String,Object>>  
7     }  
8 }
```

(5) 饼图API介绍

2: 后台代码

(1) ReportController

Map<String,Object> {setmealNames, setmealCount}, setmealNames从setmealCount提取

(2) SetmealService

(3) SetmealServiceImpl

(4) SetmealDao.java

(5) SetmealDao.xml

【讲解】

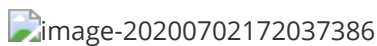
3.2. 前台页面

套餐预约占比饼形图对应的页面为/pages/report_setmeal.html。



3.2.1. 修改main.html

添加report_setmeal.html的url



3.2.2. 导入ECharts库



3.2.3. 参照官方实例导入饼形图

```
1 <div class="app-container">  
2     <div class="box">  
3         <!-- 为 Echarts 准备一个具备大小（宽高）的 DOM -->  
4         <div id="chart1" style="height:600px;"></div>  
5     </div>  
6 </div>
```

Js代码:

```
1 <script type="text/javascript">
2   // 基于准备好的dom，初始化echarts实例
3   var myChart1 = echarts.init(document.getElementById('chart1'));
4
5   // 使用刚指定的配置项和数据显示图表。
6   //myChart.setOption(option);
7
8   axios.get("/report/getSetmealReport.do").then((res) => {
9     if (res.data.flag) {
10       myChart1.setOption({
11         title: {
12           text: '套餐预约占比',
13           subtext: '',
14           x: 'center'
15         },
16         tooltip: { //提示框组件
17           trigger: 'item', //触发类型，在饼形图中为item
18           formatter: "{a} <br/>{b} : {c} ({d}%) " //提示内容格式
19         },
20         legend: {
21           orient: 'vertical',
22           left: 'left',
23           data: res.data.data.setmealNames
24         },
25         series: [
26           {
27             name: '套餐预约占比',
28             type: 'pie',
29             radius: '55%',
30             center: ['50%', '60%'],
31             data: res.data.data.setmealCount,
32             itemStyle: {
33               emphasis: {
34                 shadowBlur: 10,
35                 shadowOffsetX: 0,
36                 shadowColor: 'rgba(0, 0, 0, 0.5)'
37               }
38             }
39           }
40         ]
41       });
42     } else {
43       alert(res.data.message);
44     }
45   });
46 </script>
```

3.2.4 .分析需要构造的数据格式和sql语句

1: 根据饼形图对数据格式的要求，我们发送ajax请求，服务端需要返回如下格式的数据：

```

1  {
2      "data":{
3          "setmealNames":["套餐1","套餐2","套餐3"],
4          "setmealCount":[
5              {"name":"套餐1","value":10},
6              {"name":"套餐2","value":30},
7              {"name":"套餐3","value":25}
8          ]
9      },
10     "flag":true,
11     "message":"获取套餐统计数据成功"
12 }

```

2: 组织数据结构:

```

1  Map<String,Object>;
2  map.put("setmealNames",List<String>);
3  map.put("setmealCount",List<Map>)
4

```

3: 只需要把List setmealCount查询出来, setmealNames的数据也就有了

```

1  SELECT s.name, COUNT(o.id) value FROM t_order o, t_setmeal s WHERE
   o.setmeal_id = s.id GROUP BY s.name

```

3.2.5 .饼图: API介绍:

第一步: 查看图例



第二步: 查看代码



第三步: 查看api



3.3. 后台代码

3.3.1. Controller

在health_web工程的ReportController中提供getSetmealReport方法

```

1  /**
2   * 套餐预约占比
3   */
4  @GetMapping("/getSetmealReport")
5  public Result getSetmealReport(){
6      // 调用服务查询
7      // 套餐数量

```

```

8      List<Map<String, Object>> setmealCount =
setmealService.findSetmealCount();
9      // 套餐名称集合
10     List<String> setmealNames = new ArrayList<String>();
11     // [{name:,value}]
12     // 抽取套餐名称
13     if(null != setmealCount){
14         for (Map<String, Object> map : setmealCount) {
15             //map {name:,value}
16             // 获取套餐的名称
17             setmealNames.add((String) map.get("name"));
18         }
19     }
20     // 封装返回的结果
21     Map<String, Object> resultMap = new HashMap<String, Object>();
22     resultMap.put("setmealNames", setmealNames);
23     resultMap.put("setmealCount", setmealCount);
24
25     return new Result(true,
MessageConstant.GET_SETMEAL_COUNT_REPORT_SUCCESS, resultMap);
26 }

```

3.3.2. 服务接口

在SetmealService服务接口中扩展方法findSetmealCount

```

1  /**
2   * 获取套餐的预约数量
3   * @return
4   */
5  List<Map<String, Object>> findSetmealCount();

```

3.3.3. 服务实现类

在SetmealServiceImpl服务实现类中实现findSetmealCount方法

```

1  /**
2   * 获取套餐的预约数量
3   * @return
4   */
5  @Override
6  public List<Map<String, Object>> findSetmealCount() {
7      return setmealDao.findSetmealCount();
8  }

```

3.3.4. Dao接口

在SetmealDao接口中扩展方法findSetmealCount

```

1  /**
2   * 获取套餐的预约数量
3   * @return
4   */
5  List<Map<String, Object>> findSetmealCount();

```

3.3.5. Mapper映射文件

在SetmealDao.xml映射文件中提供SQL语句

```

1  <select id="findSetmealCount" resultType="map">
2  select s.name,t.value from t_setmeal s, (
3      select setmeal_id,count(1) value from t_order group by setmeal_id
4  ) t where s.id=t.setmeal_id
5  </select>

```

【小结】

核心地方 sql语句的编写, 分析 预约套餐的占比 预约(t_order), 占比 (个数/总数) 每个套餐的数量/总数量

每个套餐的数量 在订单表中, 一个套餐是有多条记录, 多条记录要合计数(count), 分组, 按套餐来计算, 按套餐来分组

1. 分析出数据格式, 返回Result

```

1  {
2      flag:true,
3      message:'成功',
4      data:{
5          setmealNames:['套餐A','套餐B','套餐C'],
6          setmealCount:[
7              {value:1, name:'套餐A'},
8              {value:2, name:'套餐B'},
9              {value:1, name:'套餐C'},
10         ]
11     }
12 }
13 }

```

1.数据封装

```

1  data--->Map
2  setmealNames--->List<String>
3  setmealCount--->List<Map>

```

2.数据怎么来, 分析出查询的sql语句 mysql 5.7 会报错

```

1  -- 报错
2  SELECT s.name, COUNT(1) value FROM t_order o, t_setmeal s WHERE o.setmeal_id
   = s.id GROUP BY o.setmeal_id
3  -- 不会报错
4  SELECT s.name, COUNT(1) value FROM t_order o, t_setmeal s WHERE o.setmeal_id
   = s.id GROUP BY s.name

```

注意事项

在SetmealDao.xml映射文件中提供SQL语句

写SQL的返回值的时候, resultMap="对象实体",也可以使用resultType="map"

```
1 <select id="findSetmealCount" resultType="map">
2   select s.name,t.value from t_setmeal s, (
3       select setmeal_id,count(1) value from t_order group by setmeal_id
4   ) t where s.id=t.setmeal_id
5 </select>
```

4. 运营数据统计

4.1. 需求分析

【目标】

通过运营数据统计可以展示出体检机构的运营情况,包括会员数据、预约到诊数据、热门套餐(前4)等信息。本章节就是要通过一个表格的形式来展示这些运营数据。效果如下图:



【路径】

1: 前台代码

- (1) 修改main.html
- (2) 定义数据模型
- (3) 对应后台sql语句
- (4) 发送请求获取动态数据

2: 后台代码

- (1) ReportController.java
- (2) ReportService.java
- (3) ReportServiceImpl.java
- (4) OrderDao.java

MemberDao.java

- (5) OrderDao.xml

MemberDao.xml

【讲解】

4.2. 前台代码

运营数据统计对应的页面为/pages/report_business.html。



4.2.1. 修改main.html

```
1  {
2    "path": "/5-1",
3    "title": "会员数量统计",
4    "linkUrl": "report_member.html",
5    "children": []
6  },
7  {
8    "path": "/5-2",
9    "title": "预约套餐占比统计",
10   "linkUrl": "report_setmeal.html",
11   "children": []
12 },
13 {
14   "path": "/5-3",
15   "title": "运营数据统计",
16   "linkUrl": "report_business.html",
17   "children": []
18 }
```

4.2.2. 定义模型数据

(1)：定义数据模型，通过VUE的数据绑定展示数据

```
1  <script>
2    var vue = new Vue({
3      el: '#app',
4      data: {
5        reportData: {
6          reportDate: null,
7          todayNewMember : 0,
8          totalMember : 0,
9          thisWeekNewMember : 0,
10         thisMonthNewMember : 0,
11         todayOrderNumber : 0,
12         todayVisitsNumber : 0,
13         thisWeekOrderNumber : 0,
14         thisWeekVisitsNumber : 0,
15         thisMonthOrderNumber : 0,
16         thisMonthVisitsNumber : 0,
17         hotSetmeal : [
18           {name: '升级肿瘤12项筛查（男女单人）体检套餐', setmeal_count: 200, proportion: 0.222},
19           {name: '升级肿瘤12项筛查体检套餐', setmeal_count: 200, proportion: 0.222}
20         ]
21       }
22     }
23   })
```


(2) : 数据展示

```

1  <div class="app-container">
2    <div class="box" style="height: 900px">
3      <div class="excelTitle" >
4        <el-button @click="exportExcel">导出Excel</el-button>运营数据统计
5      </div>
6      <div class="excelTime">日期: {{reportData.reportDate}}</div>
7      <table class="exceTable" cellspacing="0" cellpadding="0">
8        <tr>
9          <td colspan="4" class="headBody">会员数据统计</td>
10        </tr>
11        <tr>
12          <td width='20%' class="tabletrBg">新增会员数</td>
13          <td width='30%'>{{reportData.todayNewMember}}</td>
14          <td width='20%' class="tabletrBg">总会员数</td>
15          <td width='30%'>{{reportData.totalMember}}</td>
16        </tr>
17        <tr>
18          <td class="tabletrBg">本周新增会员数</td>
19          <td>{{reportData.thisWeekNewMember}}</td>
20          <td class="tabletrBg">本月新增会员数</td>
21          <td>{{reportData.thisMonthNewMember}}</td>
22        </tr>
23        <tr>
24          <td colspan="4" class="headBody">预约到诊数据统计</td>
25        </tr>
26        <tr>
27          <td class="tabletrBg">今日预约数</td>
28          <td>{{reportData.todayOrderNumber}}</td>
29          <td class="tabletrBg">今日到诊数</td>
30          <td>{{reportData.todayVisitsNumber}}</td>
31        </tr>
32        <tr>
33          <td class="tabletrBg">本周预约数</td>
34          <td>{{reportData.thisWeekOrderNumber}}</td>
35          <td class="tabletrBg">本周到诊数</td>
36          <td>{{reportData.thisWeekVisitsNumber}}</td>
37        </tr>
38        <tr>
39          <td class="tabletrBg">本月预约数</td>
40          <td>{{reportData.thisMonthOrderNumber}}</td>
41          <td class="tabletrBg">本月到诊数</td>
42          <td>{{reportData.thisMonthVisitsNumber}}</td>
43        </tr>
44        <tr>
45          <td colspan="4" class="headBody">热门套餐</td>
46        </tr>
47        <tr class="tabletrBg textCenter">
48          <td>套餐名称</td>
49          <td>预约数量</td>
50          <td>占比</td>
51          <td>备注</td>
52        </tr>
53        <tr v-for="s in reportData.hotSetmeal">

```

```

54         <td>{{s.name}}</td>
55         <td>{{s.setmeal_count}}</td>
56         <td>{{s.proportion}}</td>
57         <td>{{s.remark}}</td>
58     </tr>
59 </table>
60 </div>
61 </div>

```

4.2.3.对应后台sql语句

需求

```

1  -- 今天新增会员数
2
3  -- 总会员数
4
5  -- 本周新增会员数(>=本周的周一的日期)
6
7  -- 本月新增会员数(>=本月的第一天的日期)
8
9  -----
10 ---
11
12 -- 今日预约数
13
14 -- 今日到诊数
15
16 -- 本周预约数(>=本周的周一的日期 <=本周的周日的日期)
17
18 -- 本周到诊数(>=本周的周一的日期 <=本周的周日的日期) + (状态=已到诊)
19
20 -- 本月预约数(>=每月的第一天的日期 <=每月的最后一天的日期)
21
22 -- 本月到诊数(>=每月的第一天的日期 <=每月的最后一天的日期) + (状态=已到诊)
23
24 -- 热门套餐

```

所需sql语句

```

1  -- 今天新增会员数
2  SELECT COUNT(*) FROM t_member WHERE regTime = '2019-06-26'
3  -- 总会员数
4  SELECT COUNT(*) FROM t_member
5  -- 本周新增会员数(>=本周的周一的日期)
6  SELECT COUNT(*) FROM t_member WHERE regTime >= '2019-06-24'
7  -- 本月新增会员数(>=本月的第一天的日期)
8  SELECT COUNT(*) FROM t_member WHERE regTime >= '2019-06-01'
9  -----
10 ---
11
12 -- 今日预约数
13 SELECT COUNT(*) FROM t_order WHERE orderDate = '2019-06-26'
14 -- 今日到诊数
15 SELECT COUNT(*) FROM t_order WHERE orderDate = '2019-06-26' AND orderStatus
   = '已到诊'
16 -- 本周预约数(>=本周的周一的日期 <=本周的周日的日期)

```

```

15 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-24' and '2019-
06-31'
16 -- 本周到诊数
17 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-24' and '2019-
06-31' AND orderStatus = '已到诊'
18 -- 本月预约数(>=每月的第一天的日期 <=每月的最后一天的日期)
19 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-01' and '2019-
06-31'
20 -- 本月到诊数
21 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-01' and '2019-
06-31' AND orderStatus = '已到诊'
22
23 -- 热门套餐
24 SELECT s.name, COUNT(o.id) setmeal_count, COUNT(o.id)/(SELECT COUNT(id) FROM
t_order ) proportion FROM t_setmeal s, t_order o WHERE s.id = o.setmeal_id
25 GROUP BY s.name ORDER BY setmeal_count DESC LIMIT 0,4

```

4.2.4. 发送请求获取动态数据

(1) 添加VUE的created()钩子函数中发送ajax请求获取动态数据，通过VUE的数据绑定将数据展示到页面

```

1 created() {
2     axios.get("/report/getBusinessReportData.do").then((res)=>{
3         if(res.data.flag){
4             this.reportData = res.data.data;
5         }else{
6             this.$message.error(res.data.message);
7         }
8     });
9 },

```

(2) 根据页面对数据格式的要求，我们发送ajax请求，服务端需要返回如下格式的数据：

```

1 {
2     "data":{
3         "reportDate":"2019-04-25",
4         "todayNewMember":0,
5         "totalMember":10,
6         "thisMonthNewMember":2,
7         "thisWeekNewMember":0,
8         "todayOrderNumber":0,
9         "todayVisitsNumber":0,
10        "thisWeekVisitsNumber":0,
11        "thisWeekOrderNumber":0,
12        "thisMonthOrderNumber":2,
13        "thisMonthVisitsNumber":0,
14        "hotSetmeal":[
15            {"proportion":0.4545,"name":"粉红珍爱(女)升级TM12项筛查体检套餐",
16            "setmeal_count":5},
17            {"proportion":0.1818,"name":"美丽爸妈升级肿瘤12项筛查体检套餐",
18            "setmeal_count":2},
19            {"proportion":0.1818,"name":"珍爱高端升级肿瘤12项筛查",
20            "setmeal_count":2},

```

```

18         {"proportion":0.0909,"name":"孕前检查套餐","setmeal_count":1}
19     ],
20     },
21     "flag":true,
22     "message":"获取运营统计数据成功"
23 }

```

4.3. 后台代码

4.3.1. Controller

在ReportController中提供getBusinessReportData方法

```

1  @Reference
2  private ReportService reportService;
3
4  /**
5   * 运营统计数据
6   * @return
7   */
8  @GetMapping("/getBusinessReportData")
9  public Result getBusinessReportData() {
10      Map<String, Object> businessReport = reportService.getBusinessReport();
11      return new Result(true,
12          MessageConstant.GET_BUSINESS_REPORT_SUCCESS,businessReport);
13  }

```

4.3.2. 服务接口

在health_interface工程中创建ReportService服务接口并声明getBusinessReport方法

```

1  public interface ReportService {
2
3      /**
4       * 获得运营统计数据
5       * Map数据格式:
6       *     reportDate (当前时间) --String
7       *     todayNewMember (今日新增会员数) -> number
8       *     totalMember (总会员数) -> number
9       *     thisWeekNewMember (本周新增会员数) -> number
10      *     thisMonthNewMember (本月新增会员数) -> number
11      *     todayOrderNumber (今日预约数) -> number
12      *     todayVisitsNumber (今日到诊数) -> number
13      *     thisWeekOrderNumber (本周预约数) -> number
14      *     thisWeekVisitsNumber (本周到诊数) -> number
15      *     thisMonthOrderNumber (本月预约数) -> number
16      *     thisMonthVisitsNumber (本月到诊数) -> number
17      *     hotSetmeal (热门套餐 (取前4)) -> List<Map<String,Object>>
18      */
19      Map<String,Object> getBusinessReport() throws Exception;
20  }

```

复制传智健康day06中资料的DateUtils到health_common中的utils包下

image-20200702173042795

给DateUtils添加 getLastDayOfThisMonth方法

```
1 //获得本月最后一日的日期
2 public static Date getLastDayOfThisMonth(){
3     Calendar calendar = Calendar.getInstance();
4     calendar.set(Calendar.DAY_OF_MONTH,1);
5     // 增加一个月
6     calendar.add(Calendar.MONTH,1);
7     // 再减去1天
8     calendar.add(Calendar.DATE,-1);
9     return calendar.getTime();
10 }
```

4.3.3. 服务实现类

在health_service工程中创建服务实现类ReportServiceImpl并实现ReportService接口

```
1 package com.itheima.health.service.impl;
2
3 import com.alibaba.dubbo.config.annotation.Service;
4 import com.itheima.health.dao.MemberDao;
5 import com.itheima.health.dao.OrderDao;
6 import com.itheima.health.service.ReportService;
7 import com.itheima.health.utils.DateUtils;
8 import org.springframework.beans.factory.annotation.Autowired;
9
10 import java.text.SimpleDateFormat;
11 import java.util.Date;
12 import java.util.HashMap;
13 import java.util.List;
14 import java.util.Map;
15
16 /**
17  * Description: No Description
18  * User: Eric
19  */
20 @Service(interfaceClass = ReportService.class)
21 public class ReportServiceImpl implements ReportService {
22
23     @Autowired
24     private MemberDao memberDao;
25     @Autowired
26     private OrderDao orderDao;
27
28     @Override
29     public Map<String, Object> getBusinessReport() {
30         Map<String, Object> reportData = new HashMap<String, Object>();
31         //reportDate
32         Date today = new Date();
33         SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
34         // 星期一
```

```

35     String monday = sdf.format(DateUtils.getFirstDayOfWeek(today));
36     // 星期天
37     String sunday = sdf.format(DateUtils.getLastDayOfWeek(today));
38     // 1号
39     String firstDayOfThisMonth =
sdf.format(DateUtils.getFirstDay4ThisMonth());
40     // 本月最后一天
41     String lastDayOfThisMonth =
sdf.format(DateUtils.getLastDayOfThisMonth());
42
43
44     String reportDate = sdf.format(today);
45     //=====会员数量=====
46     //todayNewMember 今日新增会员
47     int todayNewMember = memberDao.findMemberCountByDate(reportDate);
48     //totalMember
49     int totalMember = memberDao.findMemberTotalCount();
50     //thisweekNewMember 本周新增会员数
51     int thisweekNewMember = memberDao.findMemberCountAfterDate(monday);
52     //thisMonthNewMember 本月新增会员数
53     int thisMonthNewMember =
memberDao.findMemberCountAfterDate(firstDayOfThisMonth);
54     //=====
55
56     //=====订单统计=====
57     //todayOrderNumber 今日预约数
58     int todayOrderNumber = orderDao.findOrderCountByDate(reportDate);
59     //todayVisitsNumber 今日到诊数
60     int todayVisitsNumber = orderDao.findVisitsCountByDate(reportDate);
61     //thisweekOrderNumber 本周预约数
62     int thisweekOrderNumber = orderDao.findOrderCountBetweenDate(monday,
sunday);
63     //thisweekVisitsNumber 本周到诊数
64     int thisweekVisitsNumber =
orderDao.findVisitsCountAfterDate(monday);
65     //thisMonthOrderNumber 本月预约数
66     int thisMonthOrderNumber =
orderDao.findOrderCountBetweenDate(firstDayOfThisMonth, lastDayOfThisMonth);
67     //thisMonthVisitsNumber 本月到诊数
68     int thisMonthVisitsNumber =
orderDao.findVisitsCountAfterDate(firstDayOfThisMonth);
69
70     //===== 热门套餐=====
71     //hotSetmeal
72     List<Map<String,Object>> hotSetmeal = orderDao.findHotSetmeal();
73
74     reportData.put("reportDate", reportDate);
75     reportData.put("todayNewMember", todayNewMember);
76     reportData.put("totalMember", totalMember);
77     reportData.put("thisweekNewMember", thisweekNewMember);
78     reportData.put("thisMonthNewMember", thisMonthNewMember);
79     reportData.put("todayOrderNumber", todayOrderNumber);
80     reportData.put("todayVisitsNumber", todayVisitsNumber);
81     reportData.put("thisweekOrderNumber", thisweekOrderNumber);
82     reportData.put("thisweekVisitsNumber", thisweekVisitsNumber);
83     reportData.put("thisMonthOrderNumber", thisMonthOrderNumber);
84     reportData.put("thisMonthVisitsNumber", thisMonthVisitsNumber);
85     reportData.put("hotSetmeal", hotSetmeal);

```

```

86
87         return reportData;
88     }
89 }
90

```

4.3.4. Dao接口

在OrderDao和MemberDao中声明相关统计查询方法

4.3.4.1. OrderDao.java

```

1  @Repository
2  public interface OrderDao {
3
4      Integer findOrderCountByDate(String today);
5
6      Integer findOrderCountBetweenDate(Map<String,Object> map);
7
8      Integer findVisitsCountByDate(String today);
9
10     Integer findVisitsCountAfterDate(Map<String,Object> map);
11
12     List<Map> findHotSetmeal();
13
14     /**
15      * 统计日期范围内预约数量
16      * @param startDate
17      * @param endDate
18      * @return
19      */
20     int findOrderCountBetweenDate(@Param("startDate") String startDate,
21     @Param("endDate")String endDate);
22 }

```

4.3.4.2. MemberDao.java

```

1  @Repository
2  public interface MemberDao {
3      public Integer findMemberCountBeforeDate(String date);
4      public Integer findMemberCountByDate(String date);
5      public Integer findMemberCountAfterDate(String date);
6      public Integer findMemberTotalCount();
7  }

```

4.3.5. Mapper映射文件

在OrderDao.xml和MemberDao.xml中定义SQL语句

4.3.5.1. OrderDao.xml 添加:

```

1  <!--统计日期范围内预约数量-->
2  <select id="findOrderCountBetweenDate" parameterType="string"
3  resultType="int">

```

```

3      select count(id) from t_order where orderDate between #{startDate} and #
      {endDate}
4  </select>
5
6  <!-- 修改查询热门套餐的sql 如下-->
7  <!--热门套餐，查询前5条-->
8      <select id="findHotSetmeal" resultType="map">
9          select s.name, count(o.id) setmeal_count ,count(o.id)/t.total
      proportion,s.remark
10         from t_order o, t_setmeal s,(select count(id) total from t_order) t
11         where s.id = o.setmeal_id
12         group by o.setmeal_id
13         order by setmeal_count desc limit 0,4
14     </select>

```

4.3.5.2. MemberDao.xml:

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
3      "http://mybatis.org/dtd/mybatis-3-mapper.dtd" >
4  <mapper namespace="com.itheima.health.dao.MemberDao">
5
6      <!--根据日期统计会员数，统计指定日期之前的会员数-->
7      <select id="findMemberCountBeforeDate" parameterType="string"
      resultType="int">
8          select count(id) from t_member where regTime <= #{value}
9      </select>
10
11      <!--根据日期统计会员数-->
12      <select id="findMemberCountByDate" parameterType="string"
      resultType="int">
13          select count(id) from t_member where regTime = #{value}
14      </select>
15
16      <!--根据日期统计会员数，统计指定日期之后的会员数-->
17      <select id="findMemberCountAfterDate" parameterType="string"
      resultType="int">
18          select count(id) from t_member where regTime >= #{value}
19      </select>
20
21      <!--总会员数-->
22      <select id="findMemberTotalCount" resultType="int">
23          select count(id) from t_member
24      </select>
25  </mapper>

```

【小结】

1. 数据查询的比较多, 数据怎么封装 --->选择Map
2. 数据怎么查询出来, 并放置到vue中的模型

```

1  -- 今天新增会员数
2  SELECT COUNT(*) FROM t_member WHERE regTime = '2019-06-26'
3  -- 总会员数
4  SELECT COUNT(*) FROM t_member
5  -- 本周新增会员数(>=本周的周一的日期)

```



```

6  SELECT COUNT(*) FROM t_member WHERE regTime >= '2019-06-24'
7  -- 本月新增会员数(>=本月的第一天的日期)
8  SELECT COUNT(*) FROM t_member WHERE regTime >= '2019-06-01'
9
10 -- 今日预约数
11 SELECT COUNT(*) FROM t_order WHERE orderDate = '2019-06-26'
12 -- 今日到诊数
13 SELECT COUNT(*) FROM t_order WHERE orderDate = '2019-06-26' AND orderStatus
   = '已到诊'
14 -- 本周预约数(>=本周的周一的日期 <=本周的周日的日期)
15 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-24' and '2019-
   06-31'
16 -- 本周到诊数
17 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-24' and '2019-
   06-31' AND orderStatus = '已到诊'
18 -- 本月预约数(>=每月的第一天的日期 <=每月的最后一天的日期)
19 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-01' and '2019-
   06-31'
20 -- 本月到诊数
21 SELECT COUNT(*) FROM t_order WHERE orderDate between '2019-06-01' and '2019-
   06-31' AND orderStatus = '已到诊'
22
23 -- 热门套餐
24 select s.name,t.setmeal_count,t.setmeal_count/t1.total proportion,s.remark
   from (
25     select setmeal_id,count(1) setmeal_count from t_order group by
   setmeal_id
26 ) t, (select count(1) total from t_order) t1, t_setmeal s
27 where s.id=t.setmeal_id
28 order by t.setmeal_count desc limit 0,4

```

编写报表sql的步骤

1. 找出数据所在表, 看到的数据在哪些表里
2. 找出条件所在的表, 【注意】隐藏条件 本周, 本月
3. 找出数据表之间的关系, 如果没有则找中间表
4. 找出条件表之间的关系, 如果没有则找中间表
5. 找出数据表与条件表之间的关系, 如果没有则找中间表
6. 如果是多条合成一条就使用聚合函数, 使用分组
7. 如果不能用一条语句完成, 就使用子查询, 都可以放到from后面
8. 如果要从小到大, 从大到小。。。使用排序
9. 尽量不要在where后面使用函数, 可以在select后使用

5. 运营数据统计报表导出

【目标】

运营数据统计报表导出就是将统计数据写入到Excel并提供给客户端浏览器进行下载, 以便体检机构管理人员对运营数据的查看和存档。

【路径】

1: 提供模板文件

2: 前台代码

在report_business.html页面提供“导出”按钮并绑定事件

3: 后台代码

ReportController

- 获取模板所在
- 获取报表数据
- 创建Workbook传模板所在路径
- 获取工作表
- 获取行，单元格，设置相应的数据
- 热门套餐，遍历输出填值
 - 数量的类型为Long
 - 占比的值的类型为bigdecimal，转成dubbo
- 设置响应体内容的格式application/vnd.ms-excel
- 设置响应头信息，告诉浏览器下载的文件名叫什么 Content-Disposition, attachment;filename=文件名
- Workbook.write响应输出流
- 关闭workbook

【讲解】

5.1. 提供模板文件

本章节我们需要将运营统计数据通过POI写入到Excel文件，对应的Excel效果如下：



通过上面的Excel效果可以看到，表格比较复杂，涉及到合并单元格、字体、字号、字体加粗、对齐方式等的设置。如果我们通过POI编程的方式来设置这些效果代码会非常繁琐。

在企业实际开发中，对于这种比较复杂的表格导出一般我们会提前设计一个Excel模板文件，在这个模板文件中提前将表格的结构和样式设置好，我们的程序只需要读取这个文件并在文件中的相应位置写入具体的值就可以了。

在本章节资料中已经提供了一个名为report_template.xlsx的模板文件



需要将这个文件复制到 health_web工程中



5.2. 前台代码

(1) 在report_business.html页面提供“导出”按钮并绑定事件

```

1 <div class="excelTitle" >
2   <el-button @click="exportExcel">导出Excel</el-button>运营数据统计
3 </div>

```

(2) 导出方法

```

1 methods:{
2   exportExcel(){
3     window.location.href = '/report/exportBusinessReport.do';
4   }
5 }

```

5.3. 后台代码

在ReportController中提供exportBusinessReport方法，基于POI将数据写入到Excel中并通过输出流下载到客户端

```

1 /**
2  * 导出运营统计数据报表
3  */
4 @GetMapping("/exportBusinessReport")
5 public void exportBusinessReport(HttpServletRequest req, HttpServletResponse res){
6     // 获取模板的路径，getRealPath("/") 相当于到webapp目录下
7     String template =
8     req.getSession().getServletContext().getRealPath("/template/report_template.
9     xlsx");
10    // 创建工作簿(模板路径)
11    try (// 写在try()里的对象，必须实现closable接口，try()catch()中的finally
12         OutputStream os = res.getOutputStream();
13         XSSFWorkbook wk = new XSSFWorkbook(template);){
14        // 获取工作表
15        XSSFSheet sht = wk.getSheetAt(0);
16        // 获取运营统计数据
17        Map<String, Object> reportData = reportService.getBusinessReport();
18        // 日期 坐标 2,5
19
20        sht.getRow(2).getCell(5).setCellValue(reportData.get("reportDate").toString());
21
22        //===== 会员 =====
23        // 新增会员数 4,5
24
25        sht.getRow(4).getCell(5).setCellValue((Integer)reportData.get("todayNewMember"));
26
27        // 总会员数 4,7
28
29        sht.getRow(4).getCell(7).setCellValue((Integer)reportData.get("totalMember"));
30
31        // 本周新增会员数5,5
32
33        sht.getRow(5).getCell(5).setCellValue((Integer)reportData.get("thisWeekNewMember"));
34        // 本月新增会员数 5,7

```

```

26     sht.getRow(5).getCell(7).setCellValue((Integer)reportData.get("thisMonthNew
Member"));
27
28     //===== 预约 =====
29
30     sht.getRow(7).getCell(5).setCellValue((Integer)reportData.get("todayOrderNu
mber"));
31
32     sht.getRow(7).getCell(7).setCellValue((Integer)reportData.get("todayVisitsN
umber"));
33
34     sht.getRow(8).getCell(5).setCellValue((Integer)reportData.get("thisweekOrde
rNumber"));
35
36     sht.getRow(8).getCell(7).setCellValue((Integer)reportData.get("thisweekvisi
tsNumber"));
37
38     sht.getRow(9).getCell(5).setCellValue((Integer)reportData.get("thisMonthOrd
erNumber"));
39
40     sht.getRow(9).getCell(7).setCellValue((Integer)reportData.get("thisMonthVis
itsNumber"));
41
42     // 热门套餐
43     List<Map<String,Object>> hotSetmeal = (List<Map<String,Object>>
)reportData.get("hotSetmeal");
44     int row = 12;
45     for (Map<String, Object> setmealMap : hotSetmeal) {
46
47         sht.getRow(row).getCell(4).setCellValue((String)setmealMap.get("name"));
48
49         sht.getRow(row).getCell(5).setCellValue((Long)setmealMap.get("setmeal_count
"));
50
51         BigDecimal proportion = (BigDecimal)
setmealMap.get("proportion");
52
53         sht.getRow(row).getCell(6).setCellValue(proportion.doubleValue());
54
55         sht.getRow(row).getCell(7).setCellValue((String)setmealMap.get("remark"));
56         row++;
57     }
58
59     // 工作簿写给reponse输出流
60     res.setContentType("application/vnd.ms-excel");
61     String filename = "运营统计数据报表.xlsx";
62     // 解决下载的文件名 中文乱码
63     filename = new String(filename.getBytes(), "ISO-8859-1");
64     // 设置头信息, 告诉浏览器, 是带附件的, 文件下载
65     res.setHeader("Content-Disposition", "attachment; filename=" +
filename);
66     wk.write(os);
67     os.flush();
68 } catch (IOException e) {
69     e.printStackTrace();
70 }
71 }

```

【小结】


如果发现导出Excel有些复杂, 一般先把Excel制作一个模版. 把模版通过POI读取到内存里面. 获得数据, 动态的给模版里面填充数据, 再响应(Response)文件, 使用IO流的方式输出。

登陆修改为ajax


login.html

```
1  <script>
2      var vue = new Vue({
3          el: '#app',
4          data: {
5              username: 'admin',
6              password: 'admin'
7          },
8          methods: {
9              login() {
10                 axios.post('/login.do?username=' + this.username +
11                    '&password=' + this.password).then(res => {
12                     if(res.data.flag){
13                         window.location.href="/pages/main.html"
14                     }else{
15                         this.$message.error(res.data.message);
16                     }
17                 })
18             }
19         });
20 </script>
```

添加数据绑定和事件绑定

 image-20200928165523950

修改spring-security.xml

 image-20200928165625734

UserController中添加方法

```
1  @RequestMapping("/loginSuccess")
2  public Result loginSuccess(){
3      return new Result(true, MessageConstant.LOGIN_SUCCESS);
4  }
5
6  @RequestMapping("/loginFail")
7  public Result loginFail(){
8      return new Result(false, "用户名或密码不正确");
9  }
```