“3-1 分栏 Web 统计报表”生成技术①

邓子云, 杨晓峰, 翦象慧

(湖南现代物流职业技术学院 物流信息系, 长沙 410131)

摘　要: 总结出了“3-1 分栏 Web 统计报表”的 3 个基本特征, 归结了生成“3-1 分栏 Web 统计报表”要解决的 3 个关键问题, 给出了解决问题的策略, 描述了采取的生成技术, 包括算法假设与算法设计, 并在实际工程中应用. 该种生成技术算法复杂度降为, 且只进行了一次 SQL 语句操作, 具备良好的效率.

关键词: 3-1 分栏 Web 统计报表; 生成技术; 算法假设; 算法设计; 算法实现

Web Information Extraction and Knowledge Presentation System

DENG Zi-Yun, YANG Xiao-Feng, JIAN Xiang-Hui

(Department of Logistic Information, Hunan Modern Logistics Occupation Technical College, Changsha 410131, China)

Abstract: This article summarizes 3 basic characteristics of 3-1 Split Web Report, summed up 3 key problems to be solved about the generation 3-1 Split Web Report, gives the strategy to solve these problems, describes take generation technology, including the algorithms hypothesis and algorithm design, and application in practical engineering. This paper studies the generation algorithm complexity reduction, and only one SQL statement operation, with good efficiency.

Key words: 3-1 split Web report; generation technology; algorithms hypothesis; algorithm design; algorithm realization

1　定义与问题提出

1.1 报表研究情况

基于 Web 的报表是报表发展的一个主流方向[1], 在国内外出现过大量的报表组件或专门的报表中间件软件, 如水晶报表、JasperReport 等[2]. 比如 JFreeReport 是一个用来生成报表的 Java 类库, 它为 Java 应用程序提供一个灵活的打印功能并支持输出到打印机和 PDF、Excel、HTML 和 XHTML、PlainText、XML 和 CSV 文件中.

然而复杂的报表仍然需要复杂的 SQL(Structured Query Language, 结构化查询语言) 语句和统计分析算法来实现, 很多工具难以支持复杂的报表应用需求[3]. 为研发出应用系统的通用软件报表组件, 首先需要专门研究一些复杂的报表生成技术, 以简化它们的开发工作, 比如本文将讨论的“3-1 分栏 Web 统计报表”就尚还没有报表工具做出专门的有针对性的研究.

1.2 “3-1 分栏统计报表”特征

“3-1 分栏 Web 统计报表”是一种相对较为复杂而又常用的报表, 形如图 1 所示.

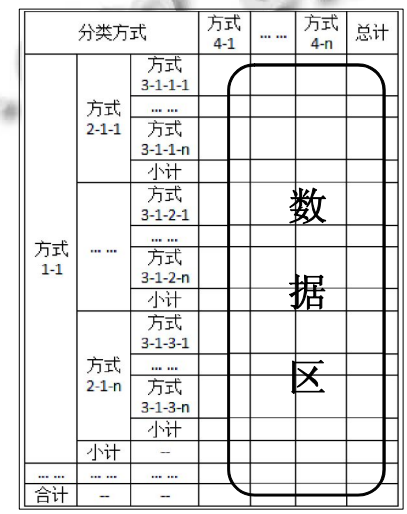

图 1　3-1 分栏统计报表

① 基金项目:湖南省重大科技专项(2010FJ1005-1);长沙市重点科技计划项目(K1113042-11)
收稿时间:2012-03-23;收到修改稿时间:2012-06-05

“3-1 分栏统计报表”具有以下基本特征:

(1) 纵向分 3 栏统计, 为 3 种分类方式, 位于报表的最左侧, 其中第 1 栏包含第 2 栏, 第 2 栏包含第 3 栏; 第 1 栏要进行“合计”, 第 2 栏要进行“小计”, 第 3 栏要进行“小计”.

(2) 横向也要进行统计, 为第 4 种分类方式.

(3) 纵向最后 1 栏为总计栏, 可对第 1、2、3 种分类方式进行“总计”.

1.3 生成“3-1 分栏统计报表”的关键问题

要生成“3-1 分栏统计报表”需要解决以下 3 个关键问题:

(1) 数据矩阵的生成问题. 要生成数据区的数据矩阵元素值, 需要程序不断进行统计, 特别是方式 1 和方式 2 的 2 个“小计”以及“合计”、“总计”数据的生成.

(3) 数据库的操作频繁问题. 由于不断要统计分析数据, 则要打开、关闭很多次数据库连接, 并且得到数据后还要不断遍历并统计分析, 而对于 Web 系统来说, 打开、关闭数据库连接是十分昂贵的操作[4], 如果生成报表的时间很难被用户所接受, 那生成报表也就意义不大了[5].

(4) 算法的复杂度太高问题. 统计共有 4 种分类方式, 且计算出一个数据矩阵元素的值至少应遍历一次 SQL 查询得到的结果记录集, 使得统计报表的算法复杂度达到 $O(n^5)$甚至更高, 需要设计出更为巧妙的算法以降低复杂度.

1.4 解决问题的策略

以解决生成“3-1 分栏统计报表”的 3 个关键问题, 可采取以下策略:

(1) 先用“Group by”语句对数据库作 select 分组查询, 并将得到的结果记录集转换为二维字符串数组, 以后进行统计数据生成时只操作这个二维字符串数组就可以了, 这样可不必再频繁开关数据库连接, 仅需进行一次数据库 select 操作即可.

(2) 根据(1)得到的二维字符串数据, 利用一定的算法来生成数据矩阵, 在 Web 界面中只需简单地显示矩阵元素的值就可以了.

2 生成技术

2.1 “Group by”查询

按前文所述解决问题的策略, 可先行根据第 1、2、3、4 种分类方式进行“Group by” SQL 查询, 如下所示:

Select 方式 1,方式 2,方式 3,方式 4,
sum(统计数据列或 O-R 对象属性)
from 数据表或 O-R 对象
Group by 方式 1,方式 2,方式 3,方式 4

2.2 算法假设

设根据“Group by”结果记录集对“方式 1”字段作 Distinct 运算(Distinct 运算是指的唯一性运算) 后得到的“方式 1”的长度为 l, L 表示“方式 1”元素所对应的“方式 2”元素个数的数组, 则 $L[i]$ 表示“方式 1”的第 i 个元素对应的“方式 2”对“Group by”结果记录集作 Distinct 运算后得到的“方式 2”元素个数.

用 M 表示某个“方式 2”元素所对应的“方式 3”元素个数数组, 则 $M[i]$ 表示第 i 个“方式 2”元素所对应的“方式 3”元素个数, 其值也是对“Group by”结果记录集作 Distinct 运算后得到的“方式 3”元素个数.

设“方式 4”的记录个数为 o, 其值是对“Group by”结果记录集作 Distinct 运算后得到的方式 4 元素个数.

设结果数据矩阵为 R, 将“Group by”结果记录集转换为一个二维的字符串数组 G, 其中第 1、2、3、4 列分别为方式 1、方式 2、方式 3、方式 4, 第 5 列为统计数据

2.3 算法设计

根据以上假设, 可得到“方式 3”的总元素个数为:

$$\sum_{j=1}^{l}\sum_{i=1}^{L[j]}M[i]$$

“方式 3”加上“方式 1”和“方式 2”的 2 个“小计”以及“合计”后的总行数为:

$$\sum_{j=1}^{l}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1$$

从而也可知第 x 个“方式 1”的第 y 个“方式 2”的“小计”在 R 中的行号为:

$$\sum_{i=1}^{y-1}(M[i]+1)+M[y]+\sum_{j=1}^{x}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1$$

据此在每次生成“小计”元素的值时, 可根据对应的方式 1、方式 2、方式 4 名称对 G 进行匹配查找, 如相同则进行累加, 可得到“小计”元素的值.

如果要求第 x 个“方式 1”的第 y 个“方式 2”的“小计”的数据行的值, 则有:

$$\begin{cases} k=\sum_{i=1}^{y-1}(M[i]+1)+M[y]+\sum_{j=1}^{x}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1 \\ R[k,j]=\sum find(G,condition(x,y,j)) \end{cases}$$

$find(G,condition(x,y,j))$ 函数表示查找字符串数组 G，匹配条件是“方式 1”为第 x 个，“方式 2”为第 y 个，“方式”4 为第 j 个.

如果要求第 x 个方式 1 的小结的数据行的值，则有(k 为行号,l 为列号)：

$$\begin{cases} k=\sum_{j=1}^{x}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1 \\ R[k,j]=\sum find(G,condition(x,j)) \end{cases}$$

$find(G,condition(x,j))$ 函数表示查找字符串数组 G，“方式 1”为第 x 个，“方式”4 为第 j 个.

如果要求合计值，则有：

$$\begin{cases} R[k,l]=\sum find(G,condition(x)) \\ k=\sum_{j=1}^{l}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1 \end{cases}$$

$find(G,condition(x))$ 函数表示查找字符串数组 G，“方式 1”为第 x 个.

如果要求报表最右边的“总计”列元素的值，即求 R 的第 o+1 列的值，可将数据 R 的当前行元素值累加：

$$R[i,o+1]=\sum_{j=1}^{o}R[i,j]$$

其中，$1\le x\le\sum_{j=1}^{l}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1$

据以上分析，可知数据矩阵中每个元素的值，则算法复杂度降为 $O(n^3)$，相对 $O(n^5)$ 降低了 2 个幂次，只进行了一次 SQL 语句(即“Group by”语句) 操作，整体算法性能得到大幅提升.

3 算法实现与应用

在湖南某集团企业的 ERP(Enterprise Resource Plan，企业资源计划) 系统研发中，为生成销售“3-1 分栏统计报表”，采用了本文所述的技术. 该系统采用了 SSH 技术研发，数据库操作采用了 Hibernate 中间件软件.

3.1 算法实现描述

“Group by”HQL 语句描述如下：

Select 业务员,区域,客户,产品系列,sum(销售金额)
from 销售明细
Group by 业务员,区域,客户,产品系列

生成数据矩阵 R 的算法如下：

//将结果记录集转换为 G
$G\leftarrow$Result;
//计算 R 的行数与列数
int iMAX = $\sum_{j=1}^{l}(\sum_{i=1}^{L[j]}(M[i]+1)+1)+1$;
int jMAX= $o+1$;
//生成 R 中元素的值
for(int i=0;i<iMAX;i++){
  if(为非“小计”和“合计”行){
    for(int j=0;j<jMAX;j++){
      //得到业务员,区域,客户,产品系列
      x =getXName(G);
      y =getYName(G);
      z =getZName(G);
      u =getUName(G);
      //设置 R 中元素的值
       ;
    }
  }
  If(为“区域”的“小计”行){
    for(int j=0;j<jMAX;j++){
      //得到业务员,区域,产品系列
      x =getXName(G);
      y =getYName(G);
      u =getUName(G);
      //设置 R 中元素的值
      $R_{[i,j]}=\sum find(G,condition(x,y,u))$;
    }
}
If(为“业务员”的“小计”行){
for(int j=0;j<jMAX;j++){
      //得到业务员,产品系列
      x =getXName(G);
      u =getUName(G);
      //设置 R 中元素的值
      $R_{[i,j]}=\sum find(G,condition(x,u))$;
    }
  }
If(为“合计”行){
    for(int j=0;j<jMAX;j++){
      //得到产品系列

u =getUName(G);
//设置 R 中元素的值
$R_{[i,j]} = \sum find(G, condition(u))$;
}
}
}
//生成 R 的“总计”列元素值
for(i=0;i<iMAX;i++)

$$R[i,o+1] = \sum_{j=1}^{o} R[i,j];$$

3.2 技术架构

在某集团的应用系统中为生成有关销售业绩的“3-1 分栏 Web 统计报表”采用了如图 2 所示的技术架构.

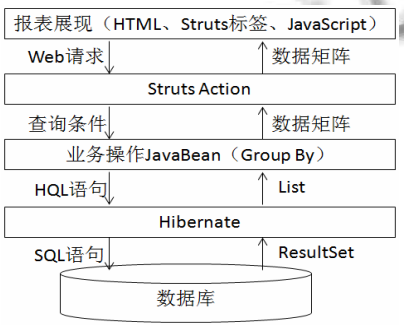

图 2 技术架构

Web 页面发起 Web 请求到给 Struts 的代表 Action 的 POJO 类; Action 将各个属性所代表的查询条件传递给业务操作 JavaBean 的生成报表的方法; 在方法体内使用“Group by” HQL 语句查询出报表数据, 得到 List 对象; 再根据本文所述算法构建出数据矩阵; 数据矩阵在 Action 中以属性的形式返回到 Web 页面; Web 页面再使用 Struts 标签遍历只需要显示数据即可.

3.3 生成的报表效果

生成的报表如图 3 所示: 有了能够生成复杂的“3-1 分栏统计报表”, 将为构建出应用系统报表中间件软件提供基础.

3.4 对比测试

在 Pentium(R) 双核 CPU(2.16GHz、2.17GHz) 、2G 内存的电脑运行 Tomcat6 软件, 数据库采用 SQL Server 2008, 将使用 Hibernate 对象操作不断访问对象属性并进行统计计算生成报表数据的简单方法, 与本文所述生成技术对比分析, 两者使用的硬件环境与软件环境相同, 对比结果情况如表 1 所示.

地宝龙（集团）装饰材料有限公司 2012 年 3 月客户销售统计表

业务员	区域	客户名称	销售合计		城市月光		流金岁月		万象更新		…	综合本 A5	
			数量合计	金额合计	数量	金额	数量	金额	数量	金额		数量	金额
袁思思	南区	钟在华	5	938	3	562	0	0	0	0	…	0	0
		小计	5	938	3	562	0	0	0	0	…	0	0
	北区	潘晓	8	288	0	0	0	0	0	0	…	0	0
		小计	8	288	0	0	0	0	0	0	…	0	0
	小计		13	1226	3	562	0	0	0	0	…	0	0
吴昕鑫	南区	彭革平	2	136	0	0	1	68	0	0	…	0	0
		小计	2	136	0	0	1	68	0	0	…	0	0
	北区	李克友	0	0	0	0	0	0	0	0	…	0	0
		郑锦	10	690	0	0	0	0	0	0	…	0	0
		小计	10	690	0	0	0	0	0	0	…	0	0
	小计		12	826	0	0	1	68	0	0	…	0	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…
唐红英	南区	黄加芳	1	180	0	0	0	0	0	0	…	0	0
		梁金凤	2	300	0	0	0	0	0	0	…	0	0
		邓家钟	4	700	0	0	2	350	0	0	…	1	200
		肖振辉	3	940	0	0	0	0	0	0		0	0
		梁江晓	2	400	0	0	0	0	0	0	…	0	0
		小计	12	2520	0	0	2	350	0	0	…	0	0
	小计		12	2520	0	0	2	350	0	0	…	0	0
合计			29512	1087650	7396	179214	7508	186982.5	1256	45397	…	1	200

图 3 生成的销售“3-1 分栏统计报表”

表 1 简单方法与本文所述生成技术对比分析

生成技术	算法复杂度	测试次数	平均生成时间
简单方法	$O(n^5)$	12	124.84 秒
本文所述生成技术	$O(n^3)$	15	1.52 秒

4 结语

“3-1 分栏 Web 统计报表”就是一种相对较为复杂而又常用的报表, 研究这种 Web 报表的生成技术具有重要的现实意义.

然而需要解决生数据矩阵的生成、数据库的操作频繁、算法复杂度太高的问题. 在本文中给出了一种生成技术的策略、详细的算法, 及其应用与实现, 这种生成技术算法复杂度降为 $O(n^3)$, 且只进行了一次 SQL 语句操作, 具备良好的效率. 从而可进一步形成通用的报表组件, 供其它应用系统复用.

参考文献

1 王野.基于 Web 的柔性报表系统的研究与实现[硕士学位论文].哈尔滨:哈尔滨工业大学,2010.
2 刘妍.基于 MVC 模式的 Web 报表系统的设计与实现[硕士学位论文].沈阳:中国科学院沈阳计算机技术研究所,2007.
3 唐远涛.基于 Web 报表开发的研究与应用[硕士学位论文].成都:成都理工大学,2006.
4 靳其兵,李晓波.基于 JSP 的数据库连接技术的研究.计算机仿真,2007,24(4):108−111.
5 李航.复杂数据源报表模型的研究与实现[硕士学位论文].郑州:河南大学,2008.