密级: 公 开

信息工程大学

操作系统实验报告

Linux初识

单 位：四院

队 别：十一大队一队

学 号：4042017002

姓 名：单琳茜

学科专业：网络工程

日 期： 2019 年 09 月 08 日

说 明

一、本实验的目的是帮助学员熟悉Linux下基本命令行使用、gcc编译器的使用等操作。

二、 要求学员使用C语言进行代码实现下面的题目。

**数据说明：**

总共包括3个数据文件：customer.txt, lineitem.txt和orders.txt。分别行数为100，1000和4000，列和列之间使用|来分隔。所有文件均为文本文件。

每一个顾客在customer表中有一个唯一的顾客编号c\_custkey，和对应的某个市场部门c\_mkgsegment。每个顾客可能会有任意多个历史订单，每个订单在orders表中有一行，它有一个唯一的订单号o\_orderkey和下单顾客编号o\_custkey和一个下单时间c\_orderdate。每个订单又由多件商品组成，每件商品在lineitem中有一行，记录了它所属的订单号l\_orderkey、价格l\_extendedprice及发货时间l\_shipdate。

目标是找出每个订单延迟发货的商品的总销售额，以供销售方有针对性地分析优化。如下图所示，我们将该问题抽象为以下的一个计算问题：每张表看作一个multi-set，对customer、orders和lineitem三张表建立笛卡尔集合，得到一个mnl行的一个集合（其中m,n,l分别是三张表的行数），然后对该集合按以下条件做过滤：

c\_mktsegment = ？ and c\_custkey=o\_custkey and o\_orderkey=l\_orderkey and o\_orderdate < ? and l\_shipdate > ? ：其中？是动态变参

对满足过滤条件的结果，按照l\_orderkey列的值进行分组，并对l\_orderkey相同的记录的l\_extendedprice做求和，最后结果按照l\_extendedprice求和值排序，并返回topn的结果，其中topn值也是动态变参。

该计算过程，也可以使用以下SQL来描述：

select

l\_orderkey,

o\_orderdate,

sum(l\_extendedprice) as revenue

from

customer,

orders,

lineitem

where

c\_mktsegment = ？

and l\_orderkey = o\_orderkey

and c\_custkey = o\_custkey

and o\_orderdate < ？

and l\_shipdate > ？

group by

l\_orderkey,

o\_orderdate

order by

revenue desc

LIMIT ？

**提交要求：**

需要提供5个文件（夹）

Readme.txt：编译和运行方式

Compile.sh：编译脚本

Run.sh: 执行脚本

Sourcecode 文件夹：包括所有source code和配置文件

可执行文件：可执行文件

**程序执行方式：**

选手需要提前将3个数据文件放到run.sh 相同目录，提交的程序将会以如下格式的命令运行、进行评测。

./run.sh customer.txt orders.txt lineitem.txt n n4个参数

第四个参数表示总共计算的次数

第5~8个参数为第一次计算时的4个参数

第9~12个参数为第二次计算时的4个参数

…

第1+4n~4+4\*n 个参数为第n次计算时的4个参数

四个参数分别对应以下的条件值：

c\_mktsegment = ？

o\_orderdate < ？

l\_shipdate > ？

LIMIT ？

执行示例：

示例1：

./run.sh customer.txt orders.txt lineitem.txt 1 BUILDING 1995-03-29 1995-03-27 5

示例2：

./run.sh customer.txt orders.txt lineitem.txt 3 BUILDING 1995-03-29 1995-03-27 5 BUILDING 1995-02-29 1995-04-27 10 BUILDING 1995-03-28 1995-04-27 2

程序输出：所有结果按行输出，多列结果使用|分割

输出示例：

示例1：

l\_orderkey|o\_orderdate|revenue

249739810|1995-02-28|513890.13

180639074|1995-03-20|502044.66

202071367|1995-03-12|499760.62

31617348|1995-03-27|498949.38

461984355|1995-03-13|491959.42

示例2：

l\_orderkey|o\_orderdate|revenue

249739810|1995-02-28|513890.13

180639074|1995-03-20|502044.66

202071367|1995-03-12|499760.62

31617348|1995-03-27|498949.38

461984355|1995-03-13|491959.42

l\_orderkey|o\_orderdate|revenue

594317283|1995-02-11|457168.01

328093060|1995-02-03|453863.74

322499142|1995-02-25|427615.4

319693698|1995-02-05|425802.97

323689092|1995-02-23|416615.42

128919045|1995-02-28|414996.73

540656743|1995-02-23|411107.69

311469825|1995-02-21|402946.11

164796770|1995-02-27|401863.1

131043138|1995-02-20|399884.41

l\_orderkey|o\_orderdate|revenue

31617348|1995-03-27|498949.38

461984355|1995-03-13|491959.42

|  |  |  |
| --- | --- | --- |
| 一、基本情况 | | |
| 实验环境 | | 说明：笔记本型号：LAPTOP-JL6E1NND 虚拟机：VMWare Linux版本：CentOS 6 |
| 实验过程  1 安装gcc  1 安装gcc  2 编写代码  由于在Linux环境下编写比较复杂，我是在Windows10下编写的。具体代码由于过长不在实验报告展示，详见文件夹中的report.c。在这里解释几个比较重要的我自定义的函数。  2.1 filter\_chain  如图。三个函数组成filter\_chain函数，实现三个txt文件的连接，并直接筛选出符合要求的项。    2.2 print\_middle\_items  将filter\_chain函数得出的符合要求的项作为一个“中间量”，输出符合要求项的信息，方便检验最后得到的结果是否为正确结果。  2.3 sum\_by\_group  将得到的“中间量”汇总起来，删掉重复的项，并将重复项金额累加。  2.4 sort\_result  将上一步得到的项进行排序。这里我用的是一个一个插入的方法。  3 输出结果 | | |
| 三、总结部分  1 总结  实验代码部分因为比较复杂，学会了要将复杂问题拆分成简单的数个小问题，并逐个解决。而且要先搞清楚实验要求做什么，逐条列出来会使思路更加清晰。    设计思路  2 建议  最大的问题是，感觉本次实验过难。  而且数据有问题，所以我用Excel自动输入了随机数字进行的测试。具体数据也一并提交到在了文件夹中。 | | |
|  | | |