**银行数据库建立与数据分析**

**一、创建一个名为Bank的数据库，数据文件名为bank.mdf和bank\_log.ldf**

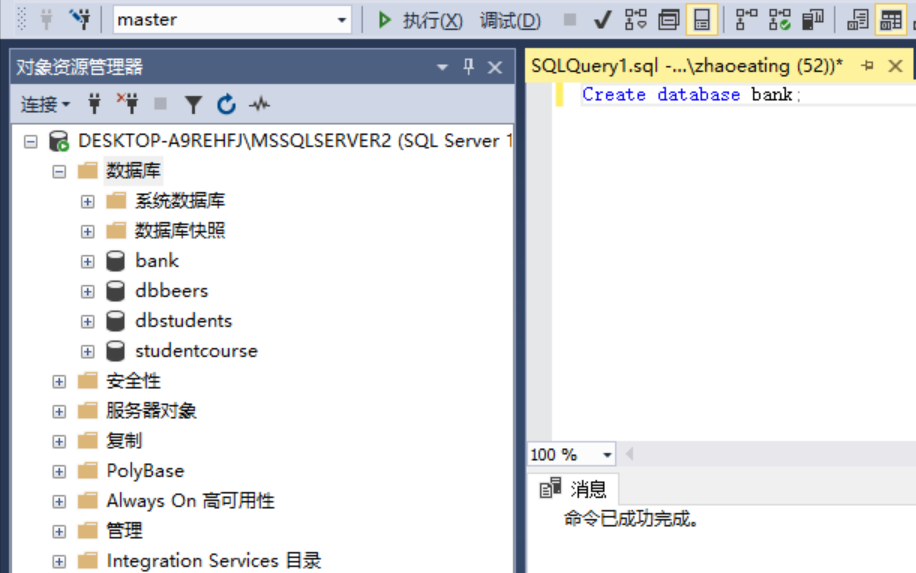
**1.1 解题思路**

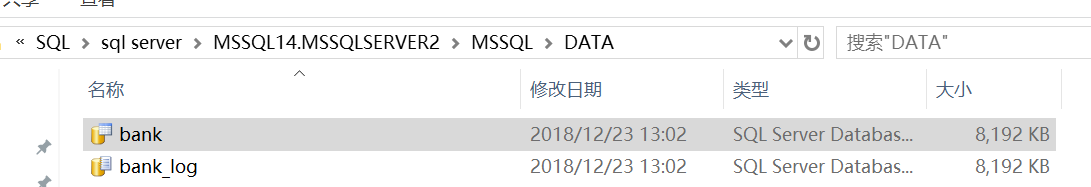
在sqlserver management里运行这条语句，并且可用数据库选择master，创建数据库。

**1.2 代码**

Create database bank;

**1.3 结果截图**





**二、设计数据库的关系用以保存所获得的相关数据**

**2.1 解题思路**

用BCNF范式和第四范式分析这几张表，把cdtrans拆分成了open和record，并标出主键。

**2.2关系模式**

Customer(cno, cname, csex, cphone)

Employee(eno, ename, esex, ephone)

Open(account, cno, aeno, date)

Record(tno, account, balance, teno, tdate)

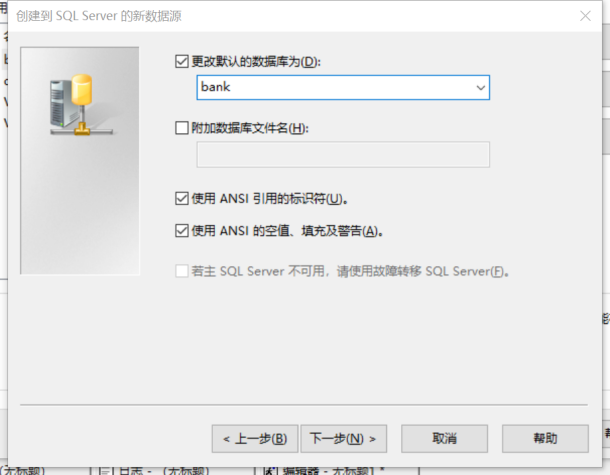
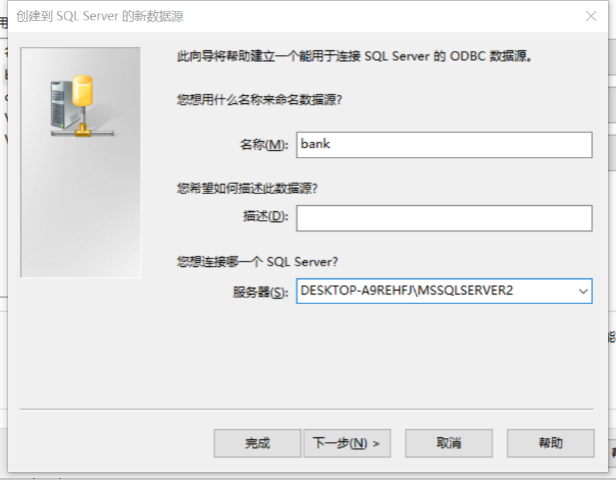
Rate(cdate, crate)

**三、创建banksql逻辑库连接SQL Server 的Bank数据库，利用SQL和SAS数据管理将获得的所有数据保存到数据库中**

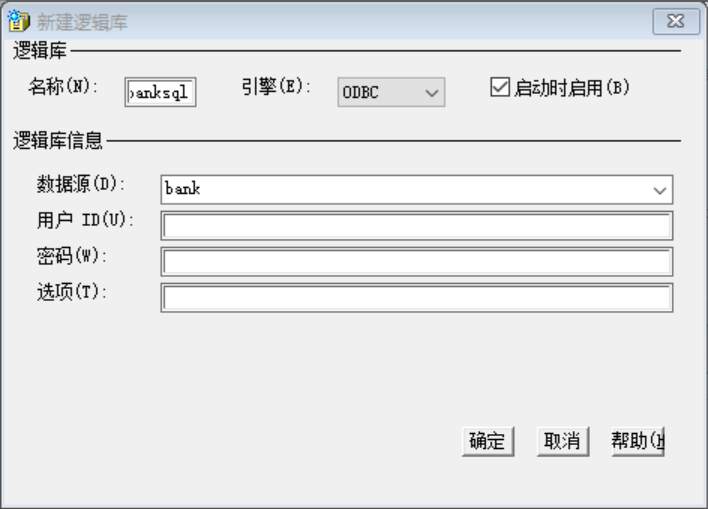
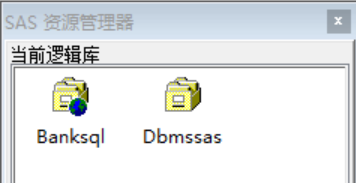
**3.1 创建banksql逻辑库连接SQL Server 的Bank数据库**

**3.1.1 解题思路**

Windows+R——输入odbcad32——添加——选择SQL Server——接下来步骤看截图



打开SAS——工具——新建逻辑库——设置

**3.2 导入数据**

**3.2.1 导入employee表**

options user=banksql;

**DATA** employee;

INPUT eno$ ename$ esex$ ephone$11.;

DATALINES;

e106001 宋慧敏 女 19307806332

e106002 严卫林 男 19325808538

e106003 李元群 女 19341352395

e106004 周良明 男 19326106297

e106005 魏东成 男 19363617796

e106006 赵伊娜 女 19311426482

e106007 王绍松 男 19359251232

e106008 郑寅波 男 19364738510

e106009 孙少伟 男 19354097935

e106010 程慧丽 女 19338467464

e106011 刘佳英 女 19362505855

e106012 陈兴颖 女 19389837518

e106013 任刚 男 19331257159

e106014 薛丽 女 19366507563

e106015 葛娟 女 19399873325

;

**RUN**;

**导入结果：**



**3.2.2 导入customer数据**

**DATA** customer;

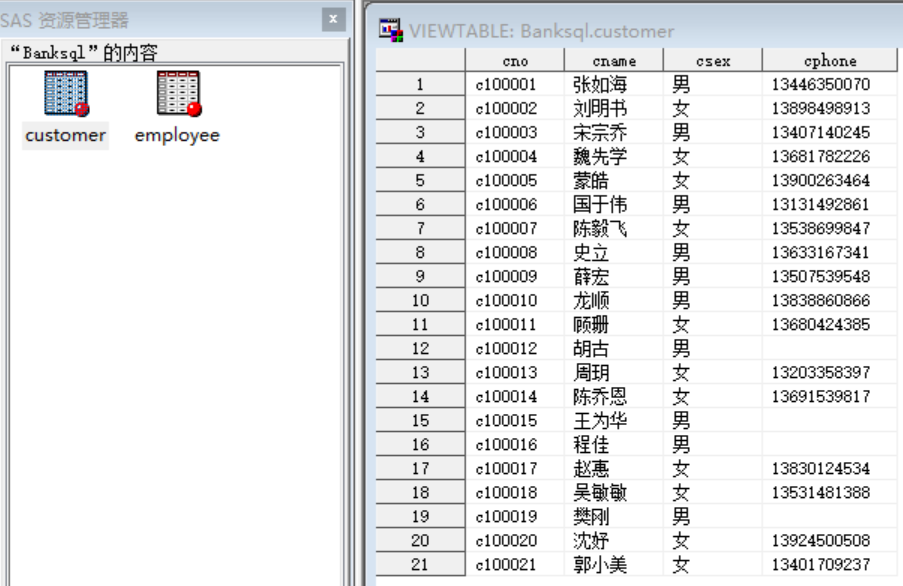
INFILE 'D:\bankdata\customer.txt'

FIRSTOBS = **3** MISSOVER;

INPUT cno$ cname$ csex$ cphone$11.;

**RUN**;

**导入结果：**



**3.2.3 导入rate数据**

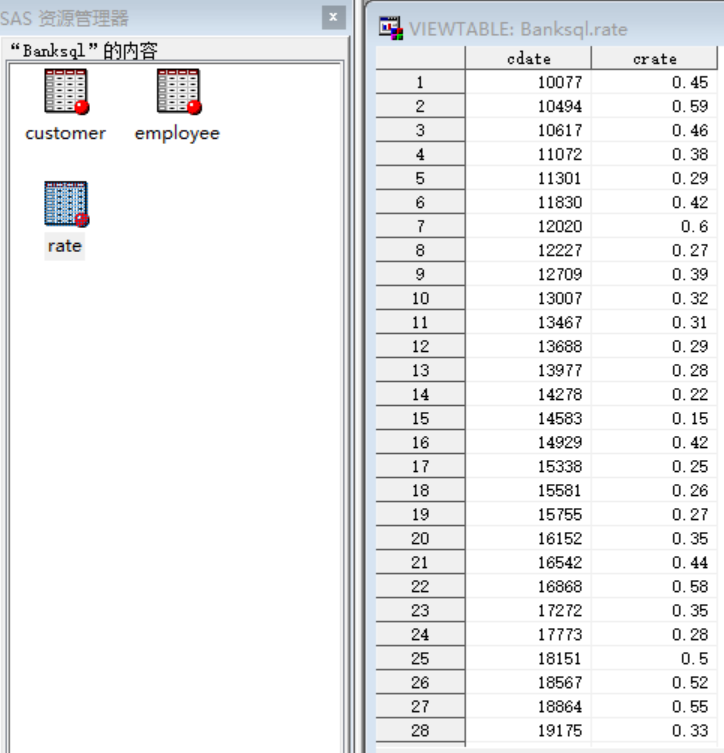
**DATA** rate;

INFILE 'D:\bankdata\cdrate.csv' DLM = ','FIRSTOBS = **3** ;

INPUT cdate yymmdd10. crate ;

**RUN**;

**导入结果：**



**3.2.4 导入cdtrans**

**3.2.4.1 解题思路**

由于45行之后的balance是空值，一起导入会出现错误，所以我们分成了两部分读入，cdtrans1存的是第3行到第45行的数据，cdtrans2存的是第46行及以后的数据，之后再用set语句把cdtrans1和cdtrans2合并成cdtrans。之后根据第二题中做的关系模式把cdtrans拆分成open和record两张表。删除cdtrans1和cdtrans2和cdtrans三张多余的表。

**3.2.4.2 代码**

**DATA** cdtrans1;

INFILE 'D:\bankdata\cdtrans.txt' DLM = '09'X

FIRSTOBS = **3** OBS = **45**;

LENGTH tno $ **12** account $ **10**;

INPUT tno $ account $ cno $ aeno $ date YYMMDD10. amount balance teno $ tdate YYMMDD10.;

**RUN**;

**DATA** cdtrans2;

INFILE 'D:\bankdata\cdtrans.txt' DLM = '09'X

FIRSTOBS = **46**;

LENGTH tno $ **12** account $ **10**;

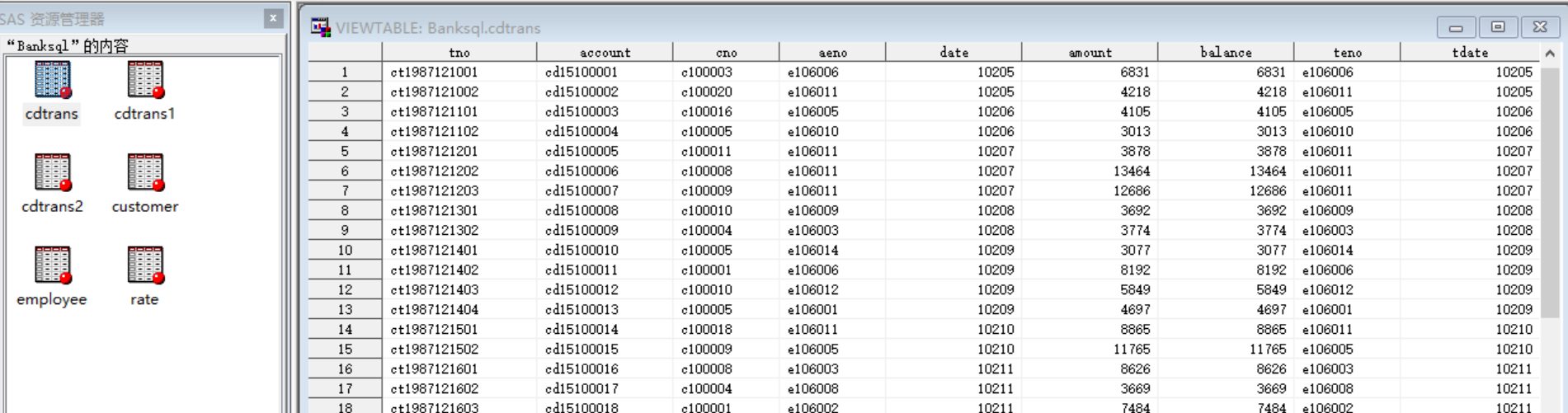
INPUT tno $ account $ cno $ aeno $ date YYMMDD10. amount teno $ tdate YYMMDD10.;

**RUN**;

**DATA** cdtrans;

SET banksql.cdtrans1 banksql.cdtrans2;

**RUN**;



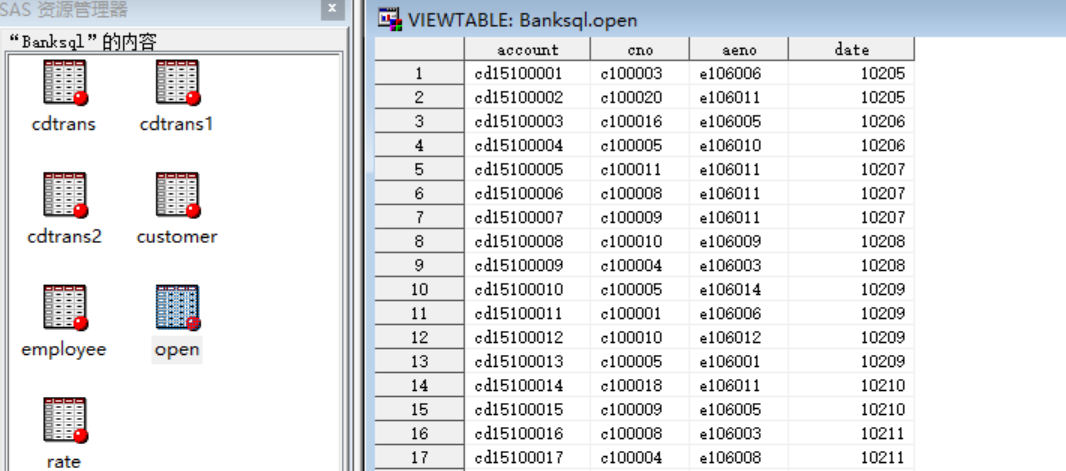
**PROC SQL**;

CREATE TABLE open AS

SELECT account,cno,aeno,date

FROM cdtrans;

**QUIT**;

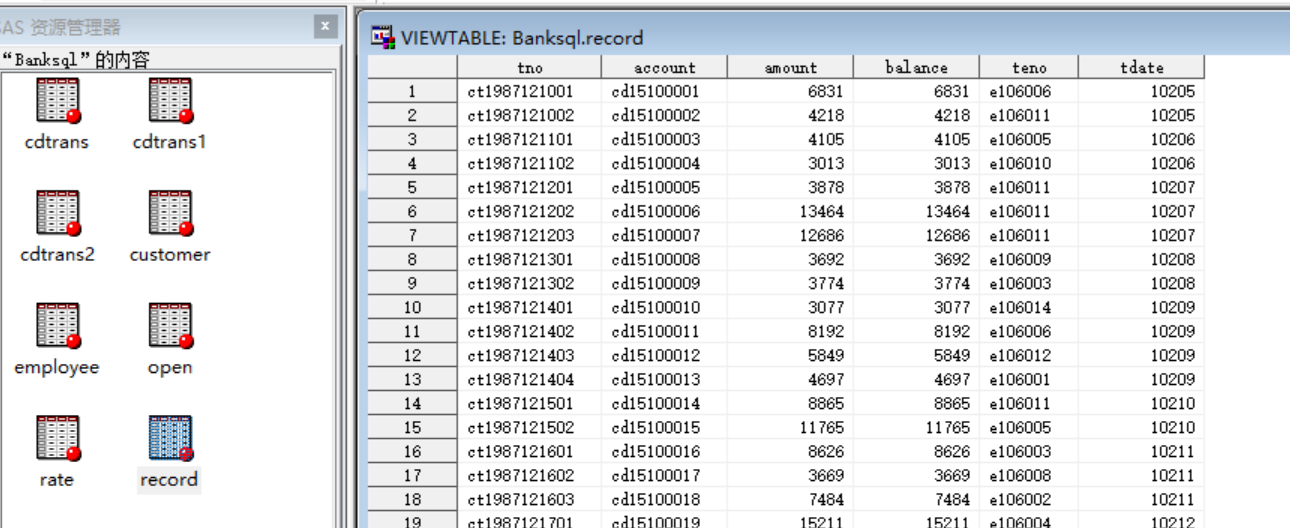


**DATA** record;

SET cdtrans;

DROP cno aeno date;

**RUN**;

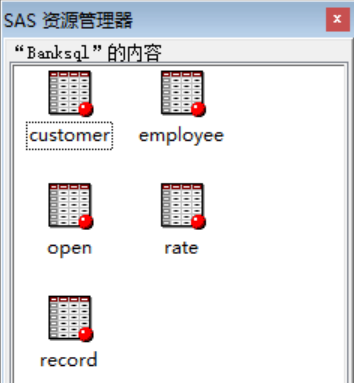


OPTIONS USER=banksql;

**PROC** **DATASETS** LIBRARY=banksql;

DELETE cdtrans1 cdtrans2 cdtrans;

**QUIT**;



**四、创建bankloc逻辑库对应到本地硬盘的一个文件夹如：D:\Bank，将banksql逻辑库中的数据集都拷贝至bankloc逻辑库中，在bankloc逻辑库中进行数据处理和分析**

**4.1 先在D盘新建一个文件夹bank**

**4.1.1 代码**

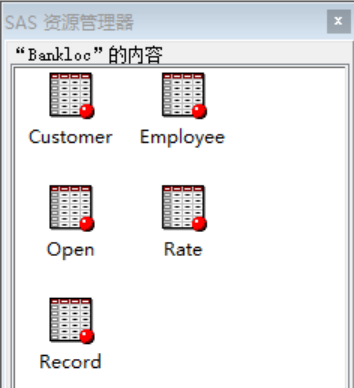
LIBNAME bankloc 'D:\bank';

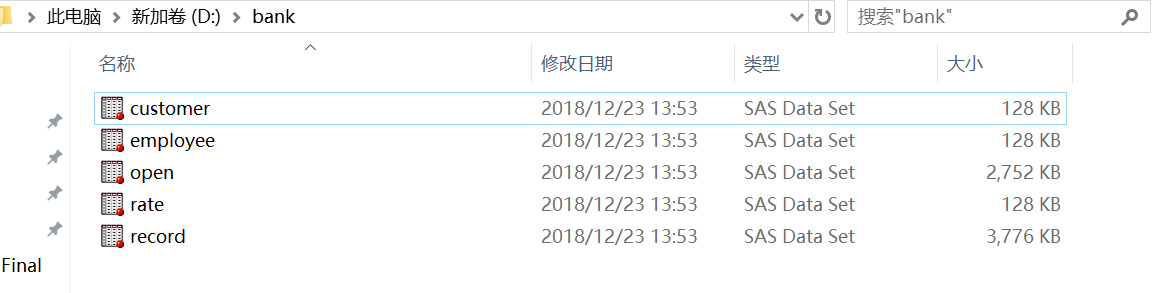
**PROC** **DATASETS**;

COPY IN = banksql OUT =bankloc;

**RUN**;

**QUIT**;





**五、创建带有当时利率的每笔交易数据的数据集trans\_r，再创建带有利息和余额(包括利息)的交易数据的数据集trans\_b，并输出cd15100001账号1998年及以前的交易信息**

**5.1 创建带有当时利率的每笔交易数据的数据集trans\_r**

**5.1.1 方法一**

**5.1.1.1 解题思路**

本题的核心是在新建trans\_r时，选出交易当天适用的利率。那么对于原本trans表格中的每一个交易日，只需要到crate的表格中选出这一天以前最大的一个日期值所对应的利率即可。

**5.1.1.2 代码**

OPTIONS USER=bankloc;

**PROC** **SQL**;

CREATE TABLE trans\_r AS

SELECT r2.tno, r2.account, r2.amount, r2.balance, r2.teno, r2.tdate, crate

FROM record r2,rate

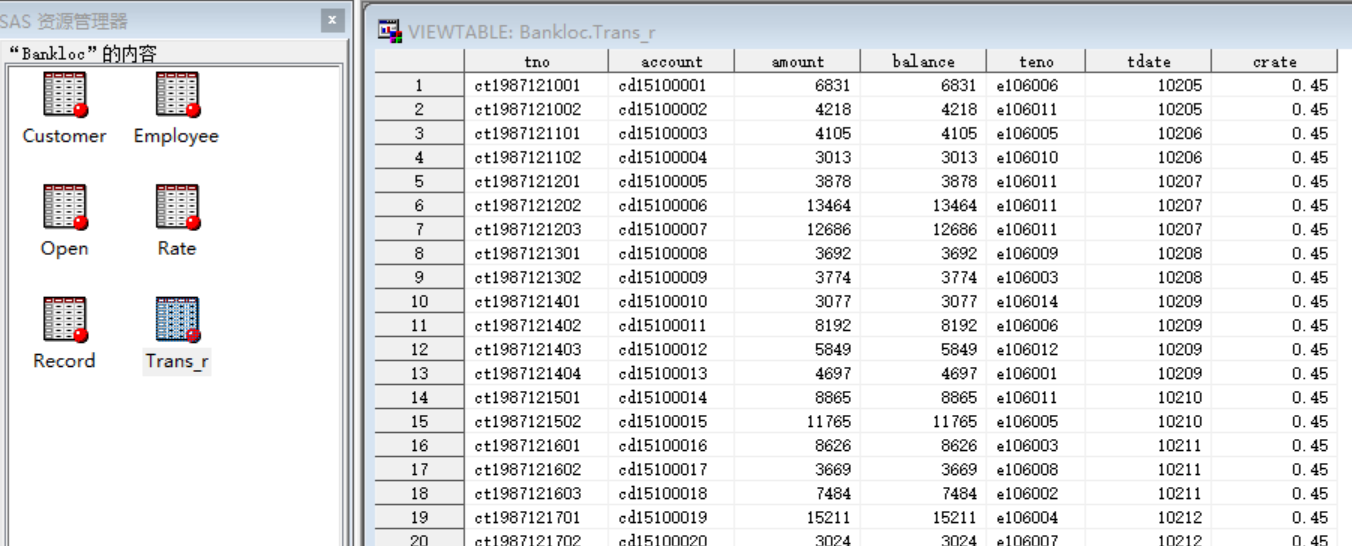
WHERE cdate=(

SELECT MAX(cdate)

FROM rate

WHERE cdate<=r2.tdate);

**QUIT**;



**5.1.2 方法二**

**5.1.2.1 解题思路**

先把record按tdate排序，然后把rate里的cdate重命名为tdate，作连接；然后把两张表merge起来，相当于是把rate里的数据按照tdate的大小插空填进去。因为当天的利率取的是最近一次改变利率的数值，就用了retain上一行的利率。最后把tno没有值（也就是从rate里合并来的数据）的行删除。

**PROC** **SORT** DATA=record;

BY tdate;

**RUN**;

**DATA** trate;

SET rate(

RENAME=(

cdate=tdate));

**RUN**;

**DATA** trans\_r2;

MERGE record trate;

BY tdate;

**RUN**;

**DATA** trans\_r2;

SET trans\_r2;

RETAIN r;

IF missing(crate)=**0** THEN r=crate;

**run**;

**DATA** trans\_r2;

SET trans\_r2;

DROP crate;

IF missing(tno)=**0**;

**run**;

**5.2 创建带有利息和余额(包括利息)的交易数据的数据集trans\_b**

**5.2.1 解题思路**

先把trans\_r中的数据按照account和tno排序，即把每个account的交易信息按发生先后顺序排序，新的数据存在sortedtrans\_r里。再在trans\_r里加一列编号和interest存在n\_sort里。接下来是算balance和interest，因为只有刚开户的时候balance有值（非null），新的一笔交易的balance=前一次交易的balance+前一次的balance产生的interest+这一次交易的amount，所以采用了retain上一次的balance和crate和tdate。这里我们用了两种方法计算balance和interest。

**5.2.1.1 排序**

**PROC** **SQL**;

CREATE TABLE sortedtrans\_r AS

SELECT \* FROM trans\_r

ORDER BY account,tno;

**QUIT**;

**5.2.1.2 编号并添加一列interest，并初始化为0**

**DATA** n\_sort;

n=\_N\_;

SET sortedtrans\_r;

interest=**0**;

**RUN**;

**5.2.1.3计算balance和interest**

**方法一：使用SAS语言**

**DATA** n\_sort;

MODIFY n\_sort;

IF Balance=NUll THEN DO;

day=tdate-ttdate;

balance=prebalance+amount+**0.01**\*day/**365**\*inte\_rate\*prebalance;

balance=round(balance,**.01**);

interest=round(**0.01**\*day/**365**\*inte\_rate\*prebalance,**.01**);

END;

prebalance=balance;

ttdate=tdate;

inte\_rate=crate;

RETAIN prebalance;

RETAIN ttdate;

RETAIN inte\_rate;

**RUN**;

*/\*然后选出数据\*/*

**PROC** **SQL**;

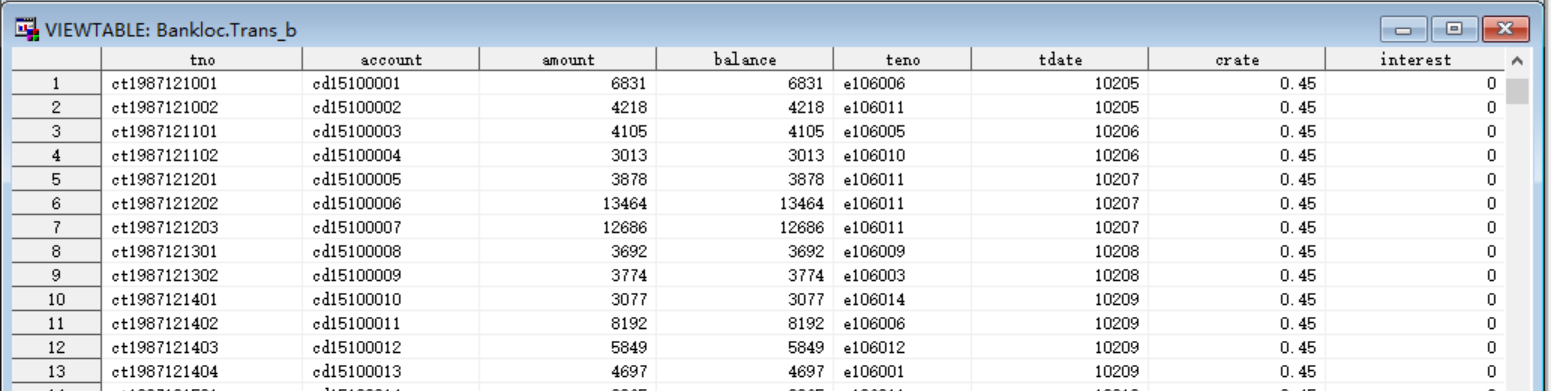
CREATE TABLE trans\_b AS

SELECT tno,account,amount,balance,teno,tdate,crate,interest

FROM n\_sort

ORDER BY tno;

**QUIT**;



**方法二：用sql过程**

CREATE PROCEDURE sp\_getbalance AS

BEGIN

DECLARE @balance real;

DECLARE @amount real;

DECLARE @day int;

DECLARE @tdate int;

DECLARE @ttdate int;

DECLARE @rate real;

DECLARE @prebalance real;

DECLARE @inte\_rate real;

DECLARE @interest real;

DECLARE @i int;

DECLARE @n int;

SET @i = 1;

SET @n=(select MAX(n) from n\_sort);

WHILE @i <= @n

BEGIN

SELECT @balance=balance,@amount=amount,@tdate=tdate,@rate=crate

FROM n\_sort

WHERE n=@i;

IF @balance is NULL

BEGIN

SET @day=@tdate-@ttdate;

SET @interest=0.01\*@day\*@inte\_rate\*@prebalance/365;

SET @balance=@prebalance+@amount+@interest;

UPDATE n\_sort

SET balance=round(@balance,2),interest=@interest

WHERE n=@i;

END

SET @ttdate=@tdate;

SET @prebalance=@balance;

SET @inte\_rate=@rate;

SET @i = @i +1;

END

END

**5.3输出cd15100001账号1998年及以前的交易信息**

**PROC** **SQL**;

CREATE TABLE info1 AS

SELECT \*

FROM trans\_b

WHERE account='cd15100001' AND year(tdate)<=**1998**;

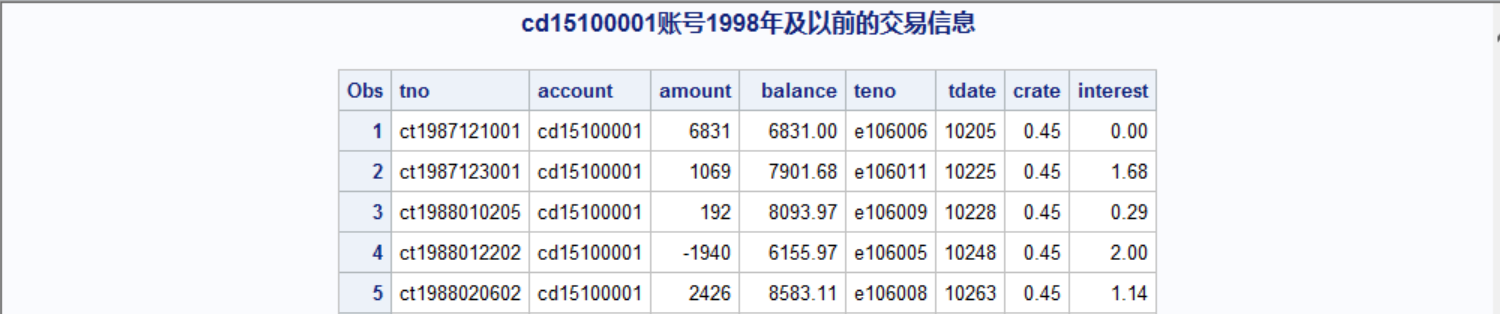
**QUIT**;

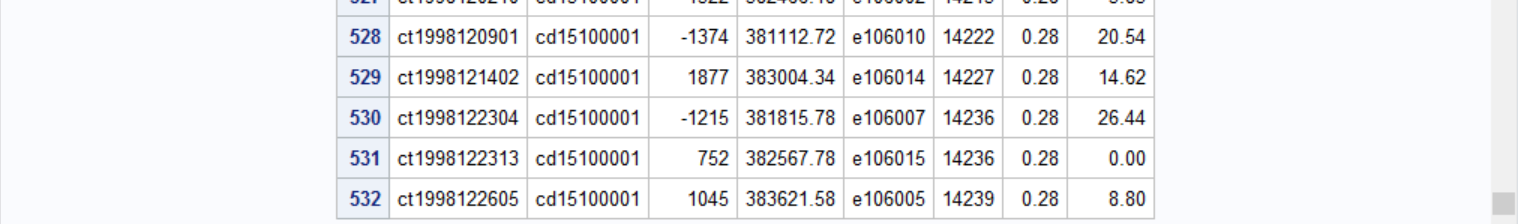
**PROC** **PRINT** DATA=info1;

TITLE "cd15100001账号1998年及以前的交易信息";

**RUN**;

**部分输出数据如下（共532条）**





**六、生存报表显示：客户姓名、客户账户数、客户总利息、客户总金额、客户的账号、账号交易总次数、账号总利息、账号总金额**

**6.1 解题思路**

第一步是产生一张cusinfo的表，存储每一个客户对应的账号及每个账号发生的交易号，但在这个过程中可能会把不同表中的重复信息导入，故NODUPKEY将其去掉。

第二步是cusinfo和customer连接产生cusinfo2，把客户姓名加入。

第三步是找出每一个account对应的最后一个交易记录，产生account\_info。

第四步是找出每一个tno对应的balance，产生tno\_info。

第五步合并tno\_account和account\_info生成account1。

第六步筛选出符合条件的表，也就是含有account信息的表，产生account2。

第七步产生一张含有客户基本信息的表customerinfo。

第八步对客户及帐户的信息做出报表分析。

**6.1 代码**

**PROC** **SQL**;

CREATE TABLE cusinfo AS

SELECT tno, trans\_b.account, cno ,balance,interest,tdate

FROM trans\_b,open

WHERE trans\_b.account=open.account;

**QUIT**;

**PROC** **SORT** DATA = cusinfo NODUPKEY;

BY account tno;

**RUN**;

/\*第一步完成\*/

**PROC** **SORT** DATA = cusinfo;

BY cno;

**DATA** cusinfo2;

MERGE cusinfo customer;

BY cno;

**RUN**;

/\*第二步完成\*/

**PROC** **SQL**;

CREATE TABLE account\_info AS

SELECT account, MAX(tno) AS max\_tno, COUNT(tno) AS times, round(SUM(interest), **2**) AS interests

FROM cusinfo2

GROUP BY account;

**QUIT**;

/\*第三步完成\*/

**PROC** **SQL**;

CREATE TABLE tno\_info AS

SELECT cname, account, tno, balance

FROM cusinfo2;

**QUIT**;

**PROC** **SORT** DATA = tno\_info;

BY account;

**RUN**;

/\*第四步完成\*/

**DATA** account1;

MERGE account\_info tno\_info;

BY account;

**RUN**;

/\*第五步完成\*/

**PROC** **SQL**;

CREATE TABLE Account2 AS

SELECT cname, account, times, interests, balance FROM account1 WHERE max\_tno = tno ORDER BY cname;

**QUIT**;

/\*第六步完成\*/

**PROC** **SQL**;

CREATE TABLE customerinfo AS

SELECT cname, count(account) AS total\_account, sum(interests) AS total\_interests, sum(balance) AS total\_balance

FROM account2

GROUP BY cname;

**QUIT**;

/\*第七步完成\*/

**DATA** cus\_final;

MERGE account2 customerinfo;

BY cname;

**RUN**;

**PROC** **REPORT** DATA = cus\_final NOWINDOWS HEADLINE;

COLUMN cname total\_account total\_interests total\_balance account times interests balance;

DEFINE cname/'客户姓名'WIDTH=**8**;

DEFINE total\_account/'客户账户数'WIDTH=**10**;

DEFINE total\_interests/'客户总利息'WIDTH=**10**;

DEFINE total\_balance/'客户总金额'WIDTH=**10**;

DEFINE account/'客户账号';

DEFINE times/'账号交易总次数'WIDTH=**14**;

DEFINE interests/'账号总利息'WIDTH=**10**;

DEFINE balance/'账号总金额'WIDTH=**10**;

define cname/order;

define total\_account/order;

define total\_interests/order;

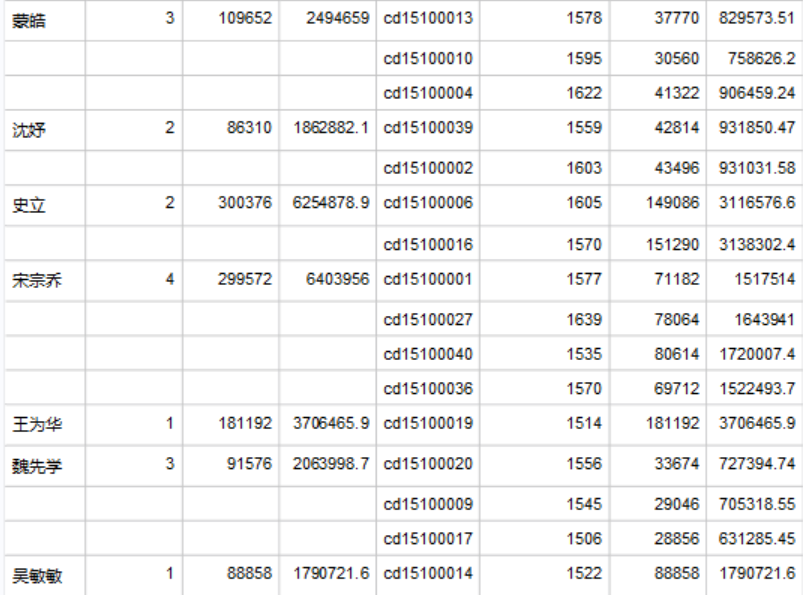
define total\_balance/order;

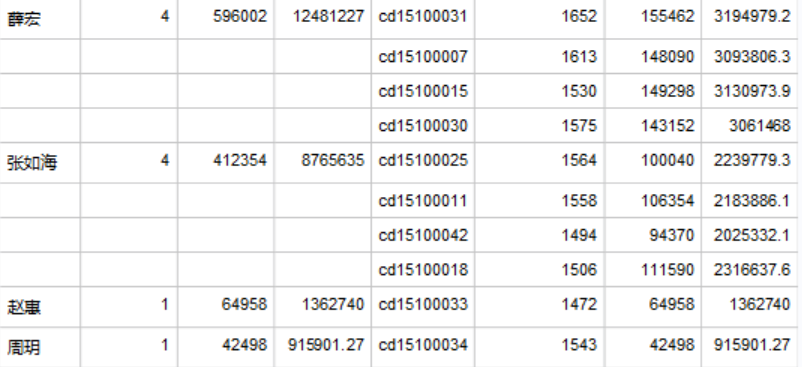
TITLE'客户信息';

**RUN**;

（输出结果见下页）







**七、输出交易量的统计矩(statistical moments)并画出带有正态分布拟合曲线的直方图**

**7.1 代码**

**PROC** **UNIVARIATE** DATA=trans\_b;

VAR amount;

HISTOGRAM amount / NORMAL (color=red l=**21**)

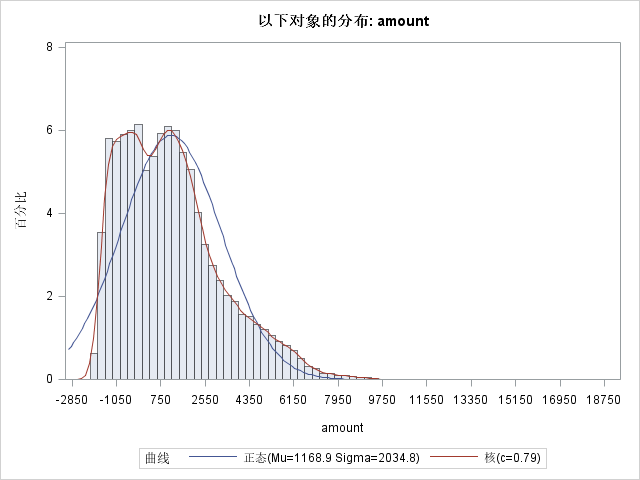
KERNEL(color=blue l=**21**) cfill=greyCC;

**RUN**;

**QUIT**;

输出结果如下：





**八、画出姓名为刘明书、国于伟、吴敏敏3个客户按4个季度统计的总利息立方体图(block chart)**

**8.1解题思路**

先筛选出李明书、国于伟、吴敏敏的信息，然后在Cusinfo2表的基础上，增加一列quarter，存储交易日期所属的季度。画出总利息立方体图。

**DATA** quarter\_interest;

SET cusinfo2;

quarter=QTR(tdate);

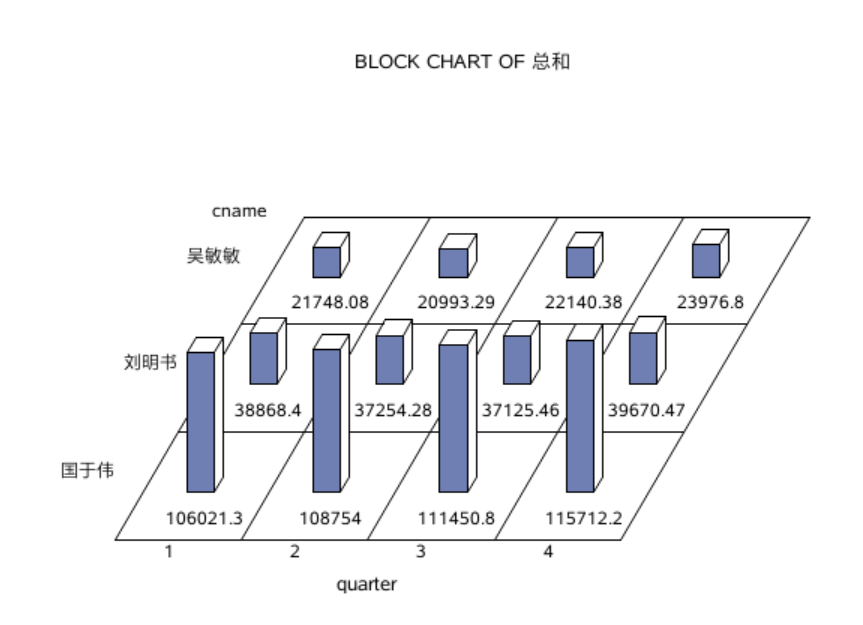
IF cname='刘明书' OR cname='国于伟' OR cname= '吴敏敏' ;

**PROC** **GCHART** DATA=quarter\_interest;

BLOCK quarter /GROUP=cname SUMVAR=interest TYPE=SUM DISCRETE;

title 'block chart of quarter\_interest';

**RUN**;



**九、画出按年(1988年-2017年)统计的银行存款总金额折线图，建立适当的回归模型，估计模型参数并预测2018年-2023年每年存款总金额**

**9.1 作图的数据准备**

**9.1.1 解题思路**

建立表fd找出每个account在每年的最后一个交易记录tno，再在add\_balance找出对应的balance，然后在sum\_balance里求和得到每年的银行存款总金额。

**9.1.2 代码**

**PROC** **SQL**;

CREATE TABLE fd AS

SELECT account,year(tdate) AS years,MAX(tdate) AS final\_date,MAX(tno) AS fianl\_tno

FROM trans\_b

GROUP BY account,year(tdate)

HAVING tdate=MAX(tdate) and tno=MAX(tno);

**QUIT**;

**PROC** **SQL**;

CREATE TABLE add\_balance AS

SELECT year(trans\_b.tdate) AS years,tdate,trans\_b.account,balance

FROM fd,trans\_b

WHERE fd.account=trans\_b.account AND fd.final\_date=trans\_b.tdate AND fd.fianl\_tno=trans\_b.tno ;

**Quit**;

**PROC** **SQL**;

CREATE TABLE Sum\_balance AS

SELECT years,sum(balance)AS yearlybalance

FROM Add\_balance

GROUP BY years

HAVING years>=**1988** AND years<=**2017**

ORDER BY years;

**QUIT**;

**9.2 画出折线图**

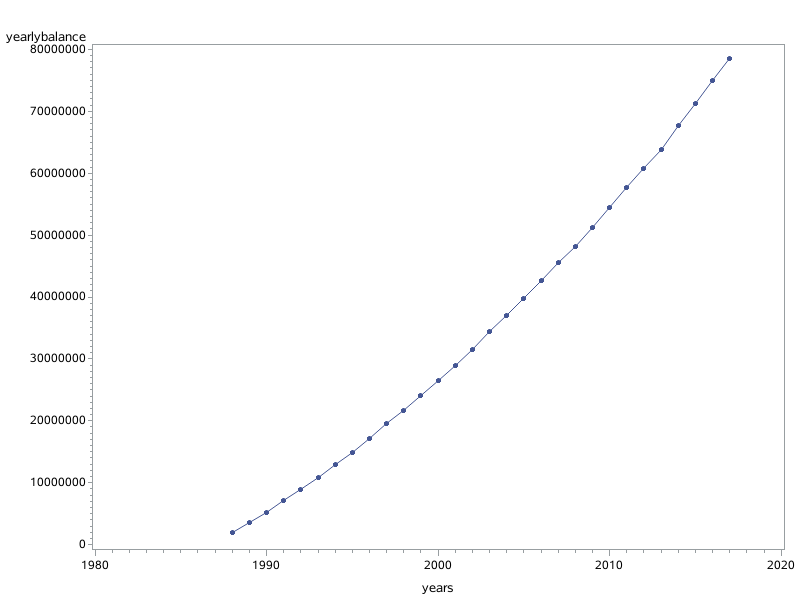
**PROC** **GPLOT** DATA=Sum\_balance;

SYMBOL VALUE=dot INTERPOL=join;

PLOT yearlybalance\*years;

**RUN**;

**QUIT**;



**9.3 建立模型并进行预测**

**9.3.1 解题思路**

首先利用几个可能的模型，去和折线图拟合，得到各个模型误差平方和，选定模型为误差平方和最小的模型，用估计参数得到预测公式，再进行预测。

**9.3.2 代码**

**（一）线性回归：**

**PROC** **REG** DATA=Bankloc.Sum\_balance;

MODEL yearlybalance=years/p r;

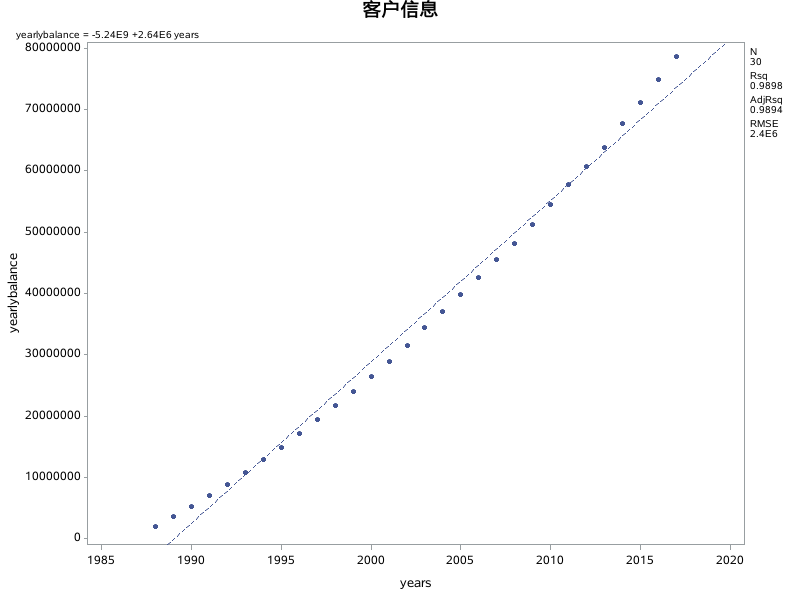
PLOT residual.\*predicted.

yearlybalance\*years;

**RUN**;

**QUIT**;





估计参数：Yearlybalance=-**5243682508**+**2636257**\*years

**（二）指数回归：**

**PROC** **NLIN**

DATA=Bankloc.Sum\_balance;

PARMS a=**0** b=**0**;

MODEL yearlybalance=exp(a+b\*years);

OUTPUT OUT=expout2 P=ygs;

**RUN**;

**PROC** **GPLOT** DATA=expout2;

PLOT yearlybalance\*years ygs\*years/HAXIS=axis1 VAXIS=axis2 OVERLAY;

AXIS1 ORDER=**1980** TO **2020** BY **10**;

AXIS2 ORDER=**0** TO **80000000** BY **10000000**;

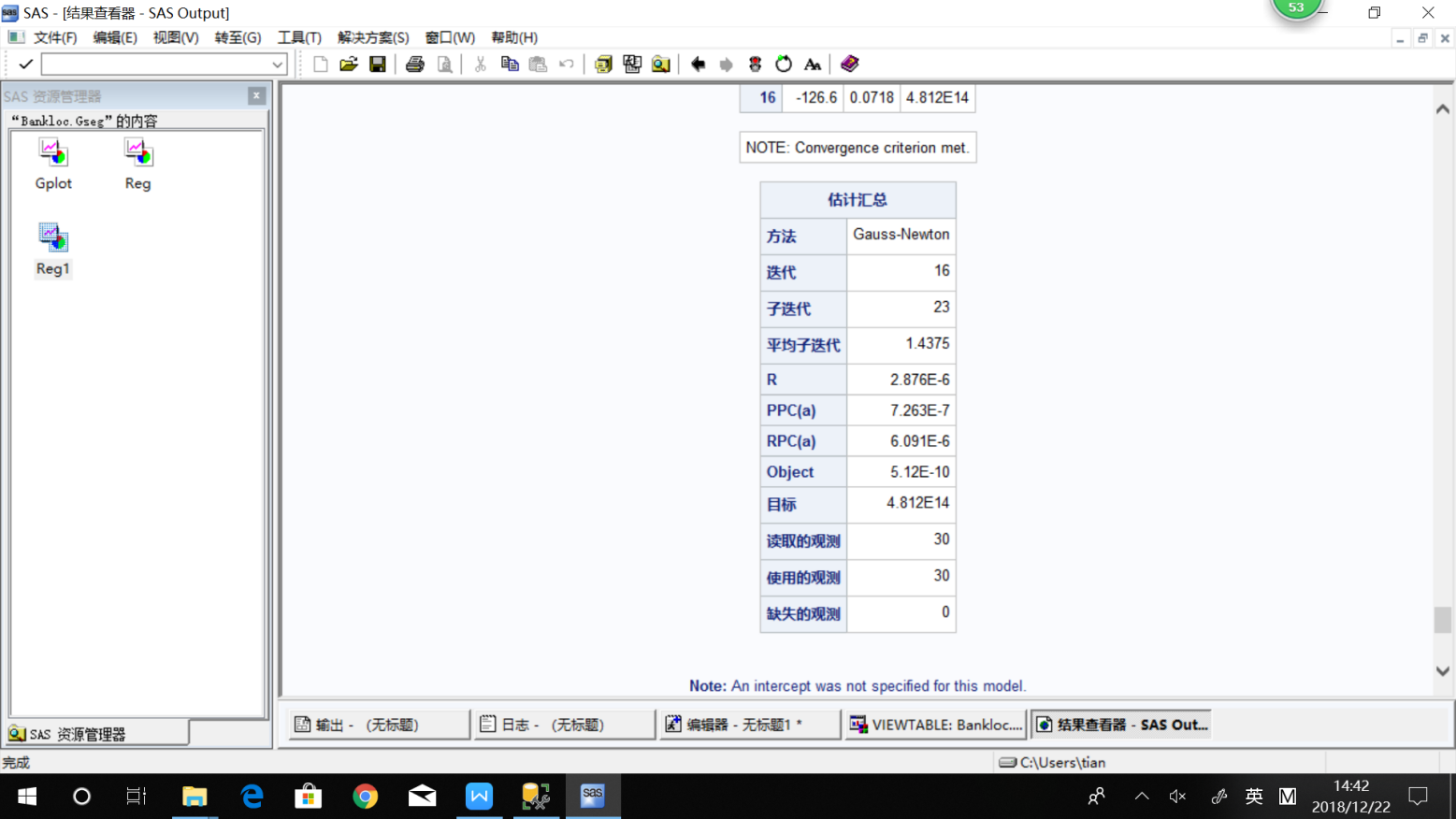
SYMBOL1 I=NONE V=plus CV=red H=**2.5** W=**2**;

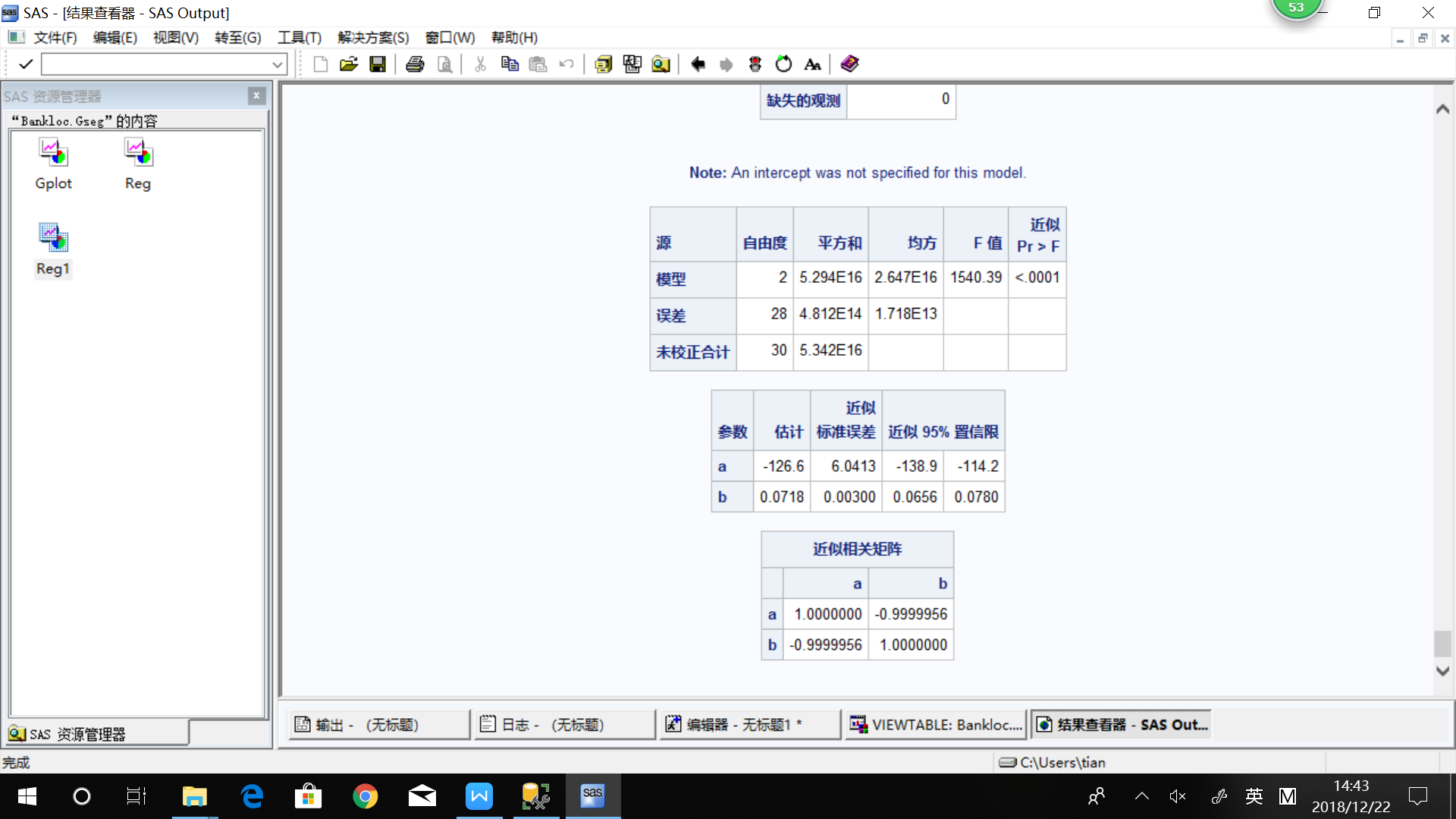
SYMBOL2 I=join V=NONE L=**1** H=**2.5** W=**2**;

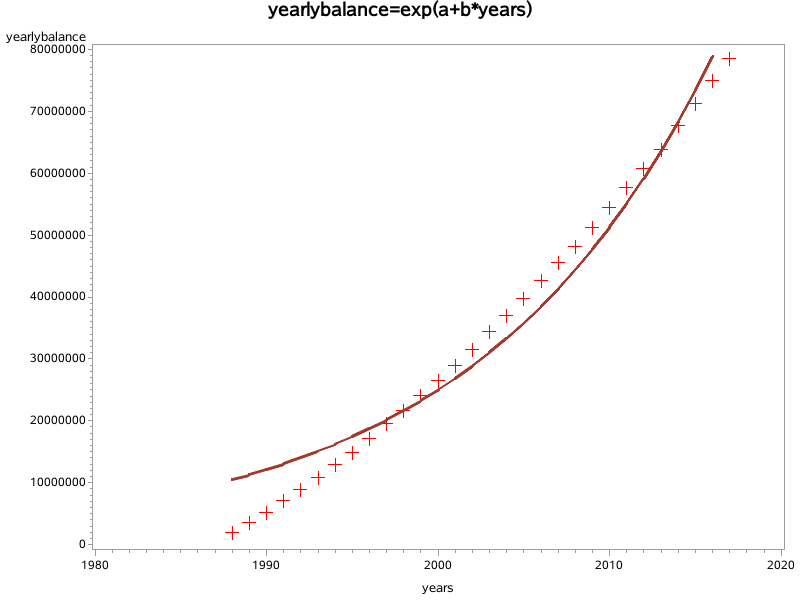
TITLE1 "yearlybalance=exp(a+b\*years)";

**RUN**;

**QUIT**;







估计参数：yearlybalance=EXP(-**126.6**+**0.0718**\*years)

**（三）二次回归:**

**PROC** **NLIN**

DATA=Bankloc.Sum\_balance;

PARMS a=**1** b=**1** c=**1**;

MODEL yearlybalance=a\*(years\*\***2**)+b\***1E7**\*years+c\***1E10**;

OUTPUT OUT=expout3 p=ygs;

**RUN**;

**PROC** **GPLOT** DATA=expout3;

PLOT yearlybalance\*years ygs\*years/HAXIS=axis1 VAXIS=axis2 OVERLAY;

AXIS1 ORDER=**1980** TO **2020** BY **10**;

AXIS2 ORDER=**0** TO **80000000** BY **10000000**;

SYMBOL1 I=NONE V=plus CV=red H=**2.5** W=**2**;

SYMBOL2 I=join V=NONE L=**1** H=**2.5** W=**2**;

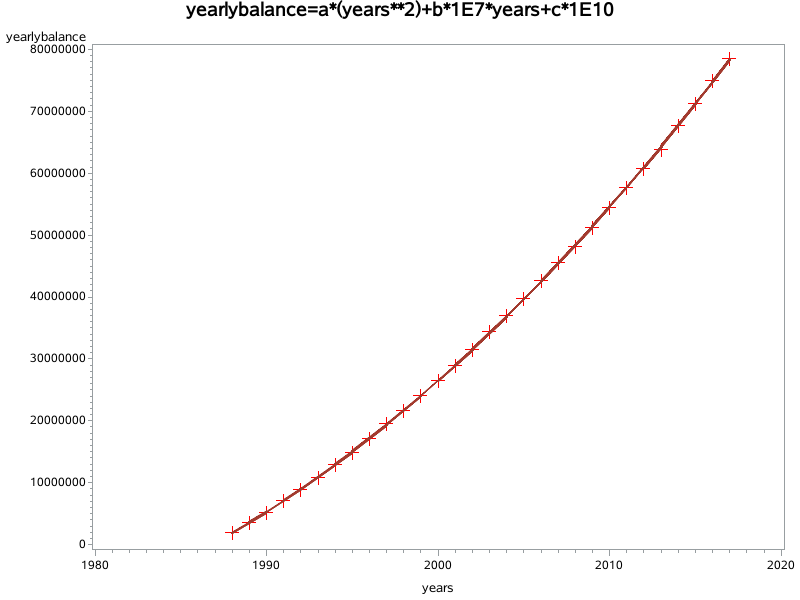
TITLE1 "a\*(years\*\*2)+b\*1E7\*years+c\*1E10";

**RUN**;

**QUIT**;







估计参数：yearlybalance=**34554.3**\*(years\*\***2**)-**13.5754E7**\*years+ **13.3317E10**;

**9.4 模型选择**

比较误差平方和，最小的拟合最好。

|  |  |
| --- | --- |
| 回归方式 | 误差平方和 |
| 线性回归 | 1.609702E14 |
| 指数回归 | 4.812E14 |
| 二次回归 | 6.745E11 |

二次回归误差平方和最小，为**6.745E11，**因此选择二次回归模型。

估计参数：yearlybalance=**34554.3**\*(years\*\***2**)-**13.5754E7**\*years+ **13.3317E10**

**9.5 计算2018-2023年的预测值**

**DATA** pre\_years;

DO years=**2018** TO **2023** BY **1**;

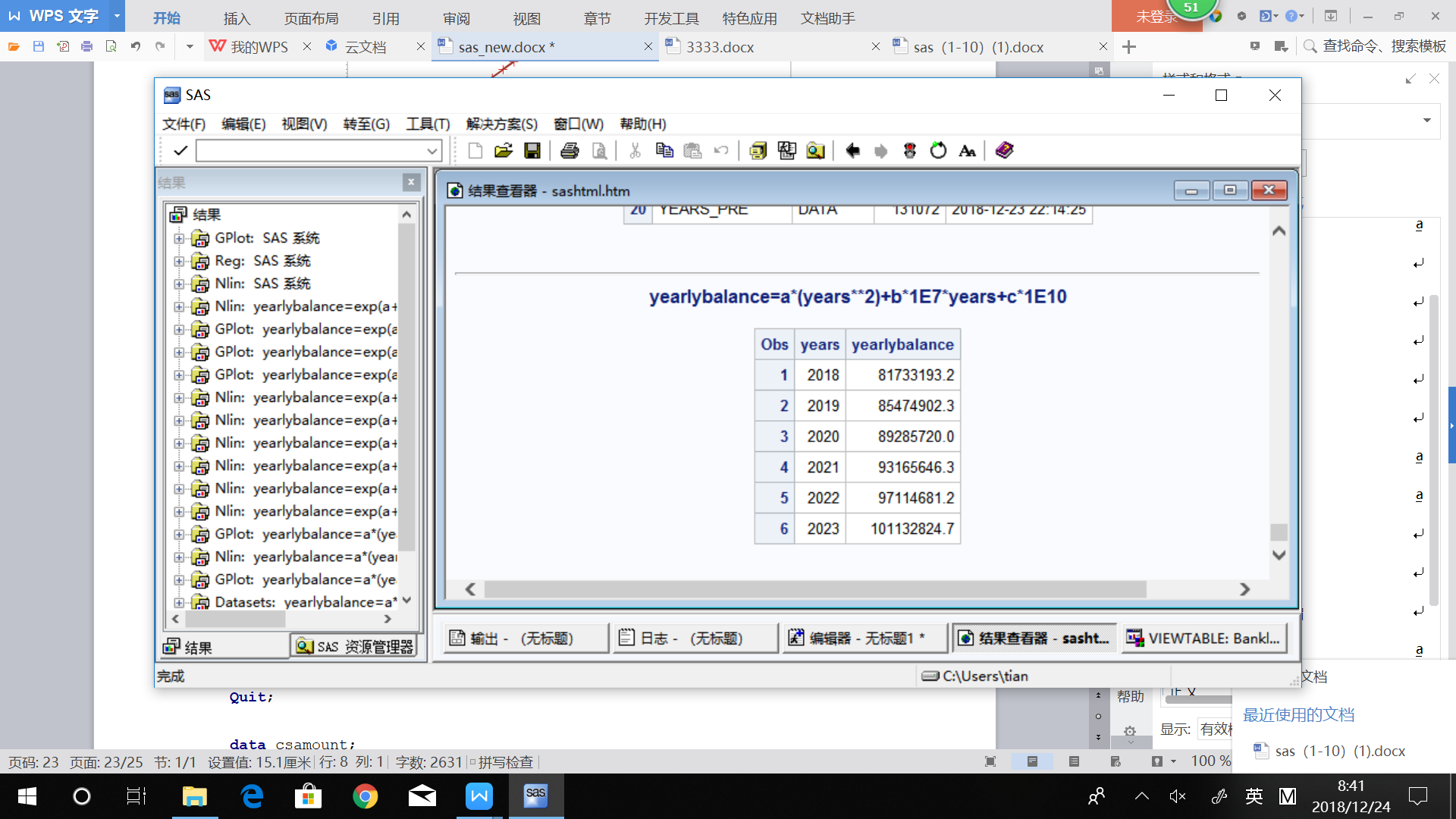
yearlybalance=**34554.3**\*(years\*\***2**)-**13.5754E7**\*years+ **13.3317E10**;

OUTPUT;

END;

**PROC** **PRINT** DATA=pre\_years;

**RUN**;



**十、创建带有观测行号、客户编号、客户姓名及每笔存钱金额的数据集csamount，建立广义回归模型(GLM)估计模型参数并分析模型参数**

**10.1 解题思路**

首先创建数据集csamount，数据筛选条件为当笔存款数大于0，并利用cusinfo2内的数据获取cno对应的cname。接下来，利用SAS中的自动变量\_n\_加入表格的行数，利用retain命令按照制定顺序排列，得到指定的数据集。由于cno和cname具有完全线性相关性（一一对应），因此制作新表csamount\_2，在这个表中cno和cname被合并，这样才满足GLM分析的假设条件。最后利用SAS的统计分析功能进行GLM估计。

**10.2 代码**

/\*创建csamount表\*/

**PROC** **SQL**;

CREATE TABLE GLM\_data AS

SELECT record.tno,cusinfo2.cno,cusinfo2.cname,record.amount FROM cusinfo2,record

WHERE cusinfo2.tno=record.tno AND record.amount>**0**;

**QUIT**;

**PROC** **SORT** DATA = Glm\_data;

BY tno;

**RUN**;

**DATA** csamount;

RETAIN index cno cname amount;

SET GLM\_data;

index=\_n\_;

**RUN**;

/\*将cno和cname合并，创建新表csamount\_2\*/

**PROC** **SQL**;

CREATE TABLE csamount\_2 AS

SELECT Index,Trim(csamount.cname)||'('||Trim(csamount.cno)||')' AS cname\_cno,amount

FROM csamount;

**QUIT**;

/\*GLM分析\*/

**PROC** **GLM** DATA=csamount\_2;

CLASS cname\_cno;

MODEL amount=cname\_cno index/SS3 SOLUTION;

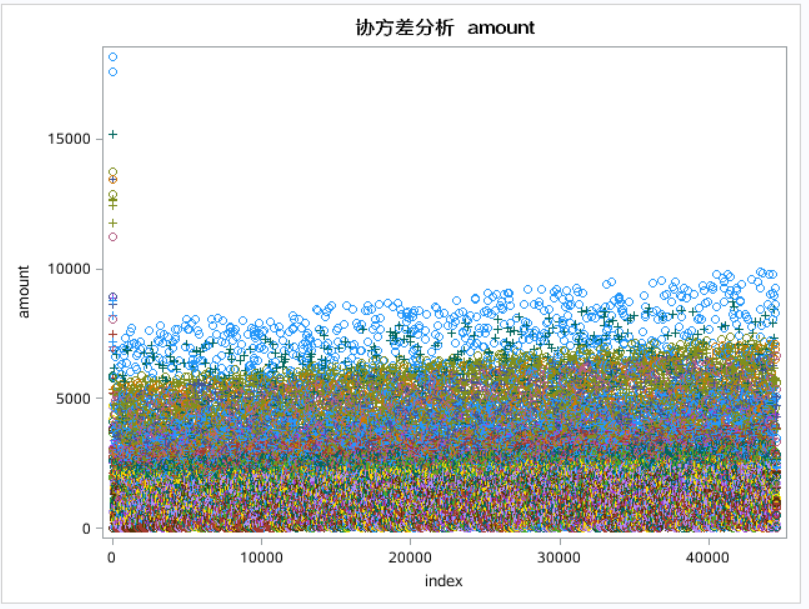
**RUN**;

**QUIT**;









**10.3 参数分析**

可以看出，自变量的F统计量为1361.55，其p-value小于0.01，因此这个模型的自变量具有联合显著性；数值变量index的F统计量为968.46，p-value小于0.01，因此数值变量具有统计显著性；类别变量F值为1380.89，p-value小于0.01，因此类别变量也具有联合显著性，但是沈妤的参数估计值p-value大于0.5，该单个变量不具有统计显著性。拟合优度不足40%，线性关系并不明显。