ORACLE SQL * FORMS 功能的扩展

张 淳 (上海市长宁区税务局)

摘要:在 *ORACLE7* 的 *SQL * FORMS3.0* 功能块上利用用户出口(*USER EXIT*)自定义屏幕输出打印函数,实现对 *SQL * FORMS3.0* 功能的扩展,以满足税收业务管理的要求。

一、问题提出的背景

ORACLE 数据库系统是由 ORACLE 公司研制与开发的一个关系数据库管理系统。SQL * FORMS 是 ORACLE 提供的众多工具软件之一,利用它,软件设计人员根据用户要求,不必编程,只需通过选择一些菜单和相应的功能键,就可完成应用程序的开发。FORMS3.0 提供了很强的屏幕操作功能,用户可以灵活方便地通过屏幕窗口直观地对数据库实施操作。上海市财税系统使用 U6000/65 小型机在 UNIX 操作系统下,用 ORACLE 数据库开发应用软件,在税收业务管理系统中,有很多任务是输出打印信息,以获得各类票据证卡。然而 FORMS3.0 没有提供现成的函数或命令可以实现 FORMS3.0 字段内容格式化输出打印。例如,在企业基本情况表的 FORMS 屏幕上,要及时从终端附联的打印机上格式化输出有关字段的内容,从而形成税务登记证(卡、表)等。为了实现税收业务管理的要求,必须编制能对 FORMS3.0 字段内容及时进行格式化输出打印的功能函数。另外,这样处理在实用上有较好的效果,由于屏幕显示的内容存在于内存中,所以输出速度快,使用户感到快速、直观。

二、解决问题的构思

我翻阅了有关资料,SQL * FORMS3.0 提供了一个名为 USER EXTT 的封装过程(PACKAGED PROCEDURES),它提供了 FORMS3.0 与 C、PASCAL、FORTRAN、PL/I 等第三代高级语言的接口,利用它可以实现对 FORMS3.0 功能的扩展。税收业务管理中,将有大量不同格式的输出打印,有格式不同的证卡表、不同格式的税票,而且中文打印机和中文终端的类型也不可能做到完全统一,不同的打印机其控制指令不可能完全相同,不同的中文终端也可能有不同的联机打印指令。如果把这些控制指令直接写入程序中,必然会形成每个应用要有与其相应的输出打印函数。这样必然增加程序的长度,增加调试程序的难度与工作量,因为每修改一次打印控制指令,都要重新编译一次整个 FORMS 的程序,而且函数功能单一,没有通用性,为了克服上述存在的问题,我把所有有关打印机控制指令、终端控制指令、输出字段的字段名,全部设计在参数文件里,这个参数文件形成该任务的描述部分,而需要设计的 FORMS3.0 扩展输出打印函数只需对所带参数所指定的参数文件进行逐个字段内容的解释,产生相应的输出动作。调用这样的功能函数,只要所带参数文件名不同,就可完成各种不同要求的输出打印,也可适用不同类型中文打印机和中文终端上的输出打印。因而这样的扩展的功能函具有一定的通用性。

三、对 SQL * FORMS3.0 功能扩展方法及其使用方法

SQL * FORMS3.0 功能扩展是通过自定义一个功能函数,然后与 FORMS 连结编译,用 USER EXIT 调用来实现其功能.SQL * FORMS3.0 允许在触发器中用 PL/SQL 块,因此在多数情况下可用的强有力的 PL/SQL 来代替 USER EXIT,或者可以在 PL/SQL 块中使用 USER EXIT 功能。USER EXIT 的编写技巧比 SQL,PL/SQL,SQL * FORMS 命令要难。因此,

它一般用于 SQL／PL／SQL 和 SQL＊FORMS 作用范围之外的处理过程。通常在以下情况下应用 USER EXIT:

1.需要处理速度快的应用;

2.控制实时设备和处理(控制打印机等);

3.数据的处理需要更加过程化;

4.特殊文件的 I／O 操作。

现在用一个例子来说明用 USER EXIT 来扩展一个名为"PRINT"的函数,它在操作的步骤规则上有严格要求,具体操作如下:

1.移动到正确的目录下

$ cd $ ORACLE_HOME／forms30／lib

把含有函数名 PRINT 的文件放入当前目录里。

2.为了建立 iabxtb 表,运行支持程序 genxtb(用户名 mana 口令 mana)

$ genxtb mana／mana。

3.用 generate30 程序生成一个名为 genxtb 的 FORM

$ generate30–TO genxtb mana／ mana

4.用 runforms30 运行 genxtb

$ runforms30 genxtb mana／mana

5.键入下列记录以使 user_exit 的 PRINT 能用

USER Exit Type Remarks

PRINT C Userexit with proccalls

6.提交记录

按 commit 键

7.用 genxib 支持程序生成一个 C 源代码文件

$ genxtb mana／mana iapxtb.c

8.现在建立一个新的 SQL＊FORMS3.0 版本,里面包含 user exit 的定义。

首先把包含"PRINT"函数文件的文件名改为 iaxpcc.pc 然后进行自动编译

$ make–f sqlforms30.mk sqlfurm S30X

编译后新版的 Forms 名为 sqlforms30x 和 rumform40x,这时可以使用 PRINT 函数了,它是通过在 Forms30 里调用封装过程 user exit 来使用。

这时,可以在所有能使用 user exit 的地方使用"PRINT",书写格式为:user exit('PRINT 参数文件名');

四、"PRINT"函数应用的设计

我们税收业务管理的税票有各式各样,而且在打印机上输出的各类帐表,不是把 Forms 里有关字段内容执行简单输出,而是要经过一定的处理后才输出打印。税票格式要求很高,首先要很好地控制打印机的打印位置,使输出的内容套印在印刷过的空白税票的适当位置上;其次,有些字需要放大,对 Forms 里的有关字段的内容进行处理,达到税票的特殊要求,例如要有中文大写金额,税款小写金额要右对齐,并且要有三位一撇,税目以及细目名要左对齐;输出的销票号时,要产生校验位;还有,有的输出内容不是一个字段的值,而是几个字段四则算后产生的内容;再次,换页走纸要精确,在实现连续走纸。分析了以上这些基本需求后,决定分二部分进行设计。

1.参数文件的设计.分析了各类输出打印需求,以及外设控制指令,归纳得出运用三种类型的描述字段,就可以达到目前所用到的所有需求。第一种字段以"N"作为起始符的字段,"N"后面跟上一个 0–255 的十进制数字,这种字段主要用来描述操作控制码,例如控制码"ESC"可用"N27,"来表示。第二种字段以"S"作为起始符的字段,"S"后面跟字符或字符串,这种字段主要用来输出字符或字符串,例如想要输出"销票号"这几个字,可以这样书写"S 销票号"第三种字段以":"作为起始符的字段,":" 后面是要求输出的 Forms 字段的字段名,例如有一个字段名为 name 的字段内容想输出打印,这样写":name",所有字段都用"。"表示字段的结束符。第三种字段使用复杂些,":"后可以跟代表特定字串处理的特殊符:

&表示把这字段的数值内容转换成相对应的中文大写金额。

@表示把这个字段的数值内容在最右边上产生一位校验位。

#表示删除字串的空格。

(N,M)表示取这个字段的 N 位起始到 M 位结束的区间内容。

"N"表示输出这个字段时向左靠齐,长度为 N,N 大于字段值长度时,字段值后充空格,反之字段值截位。

[N]表示输出此字段时向右靠齐,N 大于字段值长度

时,字段值后充空格,反之,字段值截位。

(N)表示输出此字段值时向右靠齐,数字值形成三位一撇,N 大于字段值长度时,字段值后充空格,反之截位。

以上这些特殊符号可组合在一起使用,优先级是从左到右。例如 salary 字段的内容为 2218732451,如下书写的字段:(2,8)″40″salary,其输出结果为:

″贰佰壹拾捌万柒仟叁佰贰拾肆元整

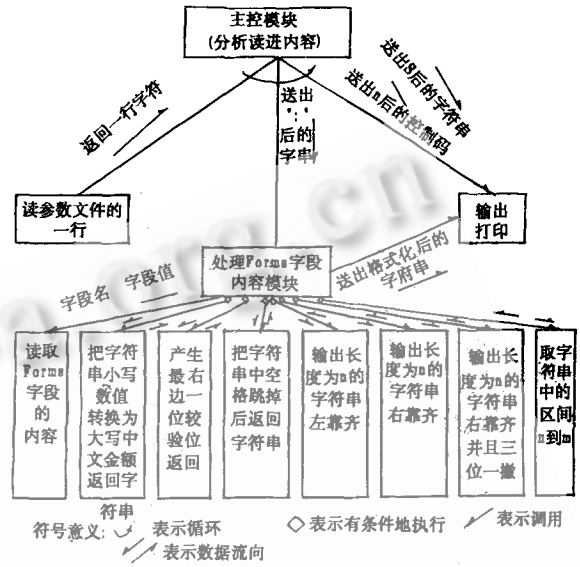
处理的过程是:首先取出字段名为 salary 的字段内容 2 到 8 位数字,然后把这 7 个数字翻译成中文口写金额,再左顶格输出 40 字节长的内容,不够部分用空格来补充。另外,字段之间、字段值和常数之间还可以包含四则运算符,例如,salary 为 20000, commition 为 8000, 用如下方式书写的字段,:(1, 11) # (15) salary +commition * 0.5,

其输出结果为:″14,000.00″

处理过程是,首先进行四则运算。由于有四则运算,所以需要有字符串类型数据转换到浮点型,浮点型转换到字符串的数值转换,按实用需要,浮点型精度定为 12 位,内含小数部分三位。上例中先把运算后的值取区间 1 到 11 位,把小数点后的第三位取掉,然后跳过前面无效的空格(″#″表示),以保证转换数值成三位一撇的正确格式;最后是向右靠齐输出 15 字节长的内容,多出长度充空格。以上二例中,为什么要定长向左输出呢?因为字段的值长度是不固定的,有长有短,如果不定长靠齐输出,那么打印出的内容参差不齐,很不美观,也不符合我们税票格式的要求。

利用以上三种字段格式,7 种特殊符及四则运算符就可以编写出描述各种输出功能参数文件。

2.对函数的设计.我们采用结构化设计思想来设计″PRINT″函数,整个函数是由若于个相对独立、功能单一的模块(子函数)所组成。整个″PRINT″函数实际上是由若于个子函数调用构成。每个子函数都可独立地进行编写、测试、排错和修改等,待一切通过后才可连结到主函数上。这样就可使复杂的研制工作得以简化,就象造机器先制造零部件,然后才总装成机器。此外,模块的相对独立性也能有效地减少编程错误在模块之间互相影响,因而提高程序的可靠性。在实际设计时,先把复杂的操作,逐层分解成一个个相对独立的功能模块,我们按下面的结构图分解 PRINT 函数的打印输出功能:



由上图可以看出每完成一个动作就是对子函数的一次调用。把功能分解到子函数,这样还容易扩充功能。如果还想扩充某种功能,则只要先独立地编制这种功能的子函数,待测试、排错通过后,连结到主函数上,主函数只要调用它,并给出调用参数,就可以实现扩充的功能。

五、应用的体会

这次采用参数文件来描述输出打印的方法,特别在程序调试过程中,受益非浅。我们使用的是 UNISYS U000/65 小型机,操作系统是 UNIX SVR4 版,数据库管理系统是 ORACLE 七版,在 ORACLE 的 Form 进行连结 menu 扩展编译时,需要先关闭数据库,让 ORACLE 数据库系统占用的内存释放出来,然后才能进行编译,一次编译所需时间等于编译所花时间加上关闭和打开 ORACLE 数据库所花时间,大约需要 5-6 分钟。我们使用爱华 HHX-22 型终端及 OKI 打印机,由于爱华 HHX-22 提供的手册有误,在既无其他资料可查阅,又无办法向他人询问的情况下,我进行了上百次的指令测试。由于控制指令包含在参数文件中,所以只要

修改参数文件,而不必重新编译 SQL * Forms3.0, 再运行一下已编译好的 SQL * Forms3.0 就可测试输出打印刚才的修改是否正确有效。就算测试了 200 次,就节省了大约(5×200=1,000)1,000 分钟的重复劳动时间。另外修改设备控制指令不用修改程序本身,这样就不会影响已调试准确的程序。在这次税收业务管理系统开发应用中,我编制了针对二种类型打印机的税票输出打印程序,一种是 OK15330,另一是 LQ1600,这二种打印机指令几乎完全不兼容,能在较短的时间里完成针对二种不兼容打印机税票输出打印程序,使用参数文件发挥了相当大的作用。

此外,我们还用此"PRINT"函数来控制爱华终端数字小键盘的功能状态切换。爱华终端键盘上的数字小键盘,有二种状态,即功能状态和数字状态。我们在使用 Forms 时,这两种奖态都要用到,然而使用手册上提供的状态切换操作,实际上不能实现有效的转换,于是我们用终端控制软指令来执行状态切换。由于我们的 Forms 已有了"PRINT"函数,所以实现起非常方便,把二个相互转换指令,分别写入三个参数文件中,在 Forms3.0 的 menu 顶目中增加二个调用 user exit 的过程项,一个写入 user exit('PRINT 数字转功能参数文件');另一个是 user exit('PRINT 功能转数字参数文件名)。

在这次编程中,我用 C 语言编程,感触最深的是,用 Case 语句来进行词意分析、用 CASE 语句来判断是否需字串处理,使编程结构非常清晰,程序功能易于扩展。

这次对 SQL * Forms3.0 功能扩展应用仅仅是初次试用,一些地方还需改进完善,例如需要编制一个对参数文件进行语法检查的程序,把书写规则错误预先排除在执行之前;各功能子函数中,应增加错误返回信息,以提示调用者。要提高这些子函数的容错性和通融性,使其能在不发生原则性错误时,尚能继续运行,不致于由于使用时传递进来的参数不规则而发生致命性错误