基于 Oracle * Forms 的开发及使用技巧

田凌 （成都铁路局电子计算技术中心 610082）

1. 前言

Oracle 数据库系统是美国 Oracle 公司提供的以分布式关系数据库为核心的一组软件产品。Oracle 的工具集中主要有三类产品：一类是决策支持工具，另一类是计算机辅助系统工具，第三类是 SQL * 系列产品，支持几种典型的应用开发，包括基于屏幕表格的应用生成、报表生成等，这是 Oracle 的最重要的一类产品。SQL * Forms 便是其中之一，用于快速开发基于表格的应用程序，以便更新维护数据。

然而，由于国内 Oracle 方面的资料很少，在遇到具体问题时更难于查找。SQL * Forms 中有众多的触发器和内部例程，在编程工作中往往要一边摸索一边总结，在此挑选出一些自己在编程过程中遇到的实用且重要的部分，介绍给大家了。

2. 示例

假设建立一个名为 TEST 的 Forms 表格 TEST.FMB 有如下结构：

```
- Trigger
  - WHEN - NEW - FORM - INSTANCE
- Alert
  - DELETE - Alert
- Block
  - MAIN
    - Item
      - CARD
        - Trigger
          - WHEN - MOUSE - CLICK
  - CXTJ
    - Item
      + TJ1
      + TJ2
      + OK
      + CANCEL
  - B-QUERY
    - Item
      + XM
      + GZRQ
  - B-BUTTON
    - Item
      - DELETE
        - WHEN - BUTTON - PRESSED
      + QUIT
```

·触发器 WHEN - NEW - FORM - INSTANCE 的工作是一旦成功进入这个程序即执行此触发器，因此在此触发器中可进行参数的设置、屏幕大小的设置、系统警告信息显示的级别设置等等初始化工作。如，Forms 信息级别分为 0、5、10、15、20、25 六个级别，在此触发器中加入以下示例可禁止 < = 20 级的信息，而只显示最严重的 25 级信息：

```
SYSTEM. MESSAGE-LEVEL: = 20;
```

在应用程序的设计中，为保持界面对用户的透明性及美观，在触发器 WHEN - NEW - FORM - INSTANCE 中屏蔽系统警告信息是非常重要的。

·对象 Alert（报警）：报警是显示的常见窗口，若要警告用户或把当前执行的操作信息报告给用户时必须使用它。在此例中给出了一个删除报警的例子，它的设计分为两部分：一部分在 SQL * Forms 的对象 Alert 中，另一部分在对象 Block 的 B-BUTTON 块的 DELETE 项中。

(1) 在对象 Alert 中，需添加名为 DELETE-Alert 的项，并修改其属性：

```
Display
  Title          删除警告
Functional
  Button 1       确认
  Button 2       取消
  Default Alert Button  是否确认删除此条记录?
```

(2) 在对象 Block 的 B-BUTTON 块中，DELETE 项是一个显示为“删除”的按钮，使用触发器 WHEN - BUTTON - PRESSED，当按钮按下时做如下工作：

```
DECLARE
```

```
    AL-BUTTON NUMBER;
  BEGIN
    AL-BUTTON: = SHOW-ALERT ( ' DELE-
ALERT');
    IF AL-BUTTON = ALERT-BUTTON1 THEN
      DELETE-RECORD;
      COMMIT-FORM;
    END IF;
  END;
```

两部分工作结合在一起，在按下“删除”按钮时便会出现报警信息，用户按下 1 号按钮“确认”则删除一条相应记录，否则按 2 号按钮“取消”。

·查询条件的设置：上例中，块 B-QUERY 是一个带基表的块，简单的查询可直接在 B-QUERY 的属性表中，修改 Database 的两项属性：

```
Database
  WHERE Clause
  ORDER BY Clause
```

依照 SQL 语言的格式填入。而复杂一些的组合条件查询便不能这样做了，必须将条件组合成一个字符串，如将上例 CXTJ 块的 TJ1 和 TJ2 组合成一个条件字符串，并赋给变量 QUERY-TJ，再通过以下调用进入 CXTJ 块，得到查询结果：

```
SET-BLOCK-PROPERTY ( ' B-QUERY ', DE-
FAULT-WHERE, QUERY-TJ);
GO-BLOCK('B-QUERY');
EXECUTE-QUERY;
```

·在生成带基表的块时，在 layout 卡片上有一选项 Button pallete，这是一个系统自动生成一些功能按钮的选项，生成的按钮将同基表块放在同一个块下，若选了此项，最好在块生成后将这些按钮移到另一个块中，以便管理。

·Property Classes，可定义属性类。如在目前流行的程序设计中，都普遍采用了卡片式界面，而 SQL * Forms 中无现成的卡片控件供程序员使用，因此可在 Property Classes 中加入 DISPLAY-ITEM 项：

```
- Property Classes
  + DISPLAY-ITEM
```

然后，在程序中便可以将文本项 CARD 设置成 DISPLAY-ITEM 类：

```
class          DISPLAY-ITEM
Type
Item Type      Display Item
Data
Default Value  查询条件
```

再配合 CARD 所带的触发器 WHEN - MOUSE - CLICK，当鼠标点在显示为“查询条件”的 CARD 项上时，可通过在触发器中加入 GO-BLOCK 语句来激活查询条件的输入界面。若为使卡片更加形象生动，可再配以适当的线条来加强效果。

·使用 SQL * Forms 制作程序封面：

从菜单中依次选择

```
Edit - > Import - > Image - > File
```

在 File 一栏中填入封面图象的文件名。

在调入图象后，有可能 WINDOWS 设置是 256 色，而调入 CANVAS 的图象却是 16 色，这种情况的出现是因为 Forms 菜单里还需要进行设置，设置的步骤如下：

```
Format - > Drawing options - > Image
  Image Dither
  ◉On
  ○OFF
```

将 Image Dither 设置成 On，图象显示便为 256 色了。

假设设计的封面在 CANVAS 中的名称是 C-FACE，则在 C-FACE 中必须得有一个项(Item)，否则在 Generate 生成执行文件后，若源文件没有块(Block)，生成的 FMX 文件在执行时不会显示封面，若源文件中有块(如 B-FACE)而无项，则会给出警告信息：

```
FRM - 30203: Warning! No items on block B-FACE.
```

FMX 执行文件仍然无法显示封面。因此，需在 C-FACE 中加入任意一个项 (Item)，例如设置一个按钮，按钮的位置在屏幕显示之内或之外都可，加入一个项后，便可顺利运行封面程序了。

3. 结论

本文介绍的一些 SQL * Forms 的开发及使用技巧为作者在编程过程中的一些心得体会，SQL * Forms 中还有很多的触发器和内部例程，要熟练地掌握它还需要不断地摸索和实践。

(来稿时间：1997 年 11 月)