ABAP开发编码规范

文档控制信息

文档信息

|  |  |
| --- | --- |
| 文档编号 |  |
| 文档名称 | ABAP开发\_编码规范 |

文档编辑历史

| **版本** | **日期** | **新增/修改** | **修订人** |
| --- | --- | --- | --- |
| <版本 #> | < yyyy-mm-dd > | <提供此版本的新增/修改细节> | <姓名> |
|  |  |  |  |
|  |  |  |  |

文档审阅/批准历史

| **日期** | **姓名** | **组织/职称** | **备注** |
| --- | --- | --- | --- |
| < yyyy-mm-dd > | <姓名> | <组织/职称> | <备注> |
|  |  |  |  |
|  |  |  |  |

文档最终交付

以下为最终版本的指定接受人：

| **姓名** | **签字** |
| --- | --- |
| <姓名> |  |
|  |  |
|  |  |

目录

[1 Development Guide 程序开发指导 5](#_Toc374110647)

[1.1 General Rule 一般规则 5](#_Toc374110648)

[1.2 Modularization 模块化 5](#_Toc374110649)

[1.3 Selection Screen选择屏幕 5](#_Toc374110650)

[1.4 Database Access数据库存取 5](#_Toc374110651)

[1.5 ‘WHERE’ Pattern WHERE语句 6](#_Toc374110652)

[1.6 Internal Table内表 6](#_Toc374110653)

[1.7 Module Pool模块池 6](#_Toc374110654)

[2 Internationalization(I18N) 多语言规范 8](#_Toc374110655)

[2.1 Text Elements 文本元素 8](#_Toc374110656)

[2.2 Error Messages 错误消息 8](#_Toc374110657)

[3 Performance 性能优化 9](#_Toc374110658)

[3.1 SQL 9](#_Toc374110659)

[3.2 SELECT 9](#_Toc374110660)

[3.3 JOIN 12](#_Toc374110661)

[3.4 UPDATE·INSERT 12](#_Toc374110662)

[3.5 LOCK OBJECT 13](#_Toc374110663)

[3.6 INTERNAL TABLE 13](#_Toc374110664)

[3.7 PROGRAM程序文件 13](#_Toc374110665)

[4 PROGRAM程序文件 15](#_Toc374110666)

[4.1 Program Header程序头 15](#_Toc374110667)

[5 Form 子程序 16](#_Toc374110668)

# Development Guide 程序开发指导

文档的该部分给出了ABAP编程的一般性指导

在组织管理模块中，系统的操作和管理都是针对“对象(Object)”进行的，并通过创建不同对象与对象之间的关系，搭建整个企业的组织结构。涉及的对象类型有：

## General Rule 一般规则

在创建新ABAP程序之前，应检查是否已经存在程序可以作为特定逻辑的模板。

尽量不要在程序中使用硬编码，而是定义常量作为比较运算。在输出Message消息时使用文本元素（Text elements），使用文本元素可以帮助同一个程序在多语言环境运行。要尽可能的使用系统值。

定义数据时尽可能使用LIKE语法，这有助于保持变量类型的一致性。

在程序传输到生产环境之前，确保所有硬断点和测试代码（例如附加的输出语句）都被删除。在程序运行时设置“软”断点来代替在程序中使用硬编码的断点。

必须为每个定义的变量提供有意义的描述，即变量名称本身应尽可能的表达出它的意义。

为了整体易读性，ABAP程序必须整齐的缩进。可以使用“PRETTY PRINTER”命令，它将为每个嵌套的命令结构缩进2个字符的位置。

限制每一行代码只有一个关键词来提高程序的可读性。

授权检查必须被加入到所有开发的代码中。授权检查的要求应该被规范化并且在开发规范中进行文档化只要有可能, 检查应该在选择屏幕层，在任意数据被取出之前进行。 (譬如为一个具体公司代码授权) 。

为了改进程序不应该直接删除原来的不需要的代码，而是应该使用块注释，然后再写新代码。

不要使用ABAP关键词作为变量；这将使程序变得不稳定。

## Modularization 模块化

使用子程序和Perform 的评价标准是：它是否可以提高程序的可读性和使得逻辑流更加清晰。

如果一块代码在ABAP中多出被调用，它就应该被创建成FORM。

PERFORM应该被使用在逻辑结构（如CASE, IF）中，与书写多条语句相比，可以提高程序的可读性。

## Selection Screen选择屏幕

参数和选择范围应该有一些默认值或必填字段，这样做的目的是防止以错误的方式运行时消耗掉SAP系统过多的资源。

尽量使用更多的选择范围来增加过滤条件，这样程序运行的会更快。为块标题和选择项指定具体的文本元素。

所有输入必须进行校验并输出合适的错误信息。

## Database Access数据库存取

在ABAP中严格禁止使用Native SQL语言(EXEC SQL … END-EXEC)。它将避开SAP数据字典的检查并打破系统的完整性。

出于性能的考虑，要限制使用逻辑数据库(LDB)。在逻辑数据库程序中大量的开销会对SAP运行时行为有负面的影响。

在每次数据表存取之后，都必须检查返回码来判断存取是否成功。如果没有数据记录被读取到，必须生成一个合适的消息通知用户。

SAP标准数据表仅能使用SAP提供的代码和方式来更新数据以保证数据的完整性。这可以通过使用已经存在的SAP程序或者通过BDC来实现。

一个外部文件的结构应该在数据字典中定义成一个结构，这将简化维护过程，特别是该外部文件在多个程序中被使用时。

## ‘WHERE’ Pattern WHERE语句

对于透明表或Pool表，SELECT语句的Where选项中尽可能的使用完全约束条件，包括那些不是关键字段的数据字段。

对于簇表，在WHERE选项中仅约束关键字段中连续的字段。使用CHECK命令进行检查来删除其他不需要的数据记录。

WHERE语句中约束字段的顺序与表中关键字段顺序应一致。

SELECT语句中WHERE条件的比较字段它们必须有相同的数据类型。

在任何可能的情况下，尽可能的使用EQ来代替使用 “=” 。

在任何可能的情况下，SELECT语句中应避免使用非逻辑(negative logic)。

尽可能的使用主键，指定完整的表关键字段并且使用SELECT SINGLE。

在其他可能的情况下使用索引。

## Internal Table内表

为了便于代码阅读，重复的命令应该写在冒号后的一行中，例如MOVE 或者 WRITE语句。各相关的对象应该在命令语句后采用缩进排版。

当内表用来保存数据库中取出的数据时，应该使用LIKE来定义内表或内表字段，以确保它们和数据库表拥有相同的特征。

在任何可能的情况下，使用CASE语句而不是重复的IF语句，可以提高代码的可读性。

在程序的LOOP循环中，不应该使用STOP和CHECK命令，而是使用EXIT跳出循环。CHECK语句可以使用在数据库读取的SELECT语句中。

READ TABLE语句必须同时使用WITH KEY并且使用BINARY SEARCH可以得到更好的性能。当然在使用READ命令之前，内表必须是按照KEY指定的列排序。

使用CLEAR语句来清空内表的表头工作区。

使用SORT <itab> ORDER BY而不是 SORT ITAB排序内表。排序内表要指明排序列。

使用REFRESH语句来删除内表的所有数据记录并释放已经被占用的空间。REFRESH不清除内表的表头数据。

如果一个内表在程序中不再需要了，使用FREE语句来删除内表所有记录并释放它所占用的全部系统资源。

## Module Pool模块池

使用SE80来创建和维护所有的MODULE POOL程序。这将确保程序的结构被系统正确的创建，所有全局变量的声明都应该包在“TOP”包含文件中。

模块池的初始屏幕号必须是1000，其他屏幕的编号应该以10或者100为单位在1000的基础上递增，如果是复制的系统标准程序，屏幕号应该在9000-9999范围内。

屏幕上除了RADIO-BUTTONS, Push-Button, 和Check-Boxes以外的所有字段都需要定义一个数据字典字段的参考，这确保字段帮助可用。如果可能的化，不要在屏幕上使用push-buttons，而是使用menu painter来代替。

在屏幕上输出文本时，将所有单词用下划线来连接，否则，每个单词会成为一个单独的文本域。

允许使用弹出式窗口。但是，如果开发的代码是用来进行批量数据处理的，那么应该提供能够避免窗口弹出的功能。

# Internationalization(I18N) 多语言规范

## Text Elements 文本元素

为不可变的文本使用文本元素，如：屏幕参数文本、屏幕选择范围文本、 其他屏幕文本, WRITE语句和应用程序的特定错误消息等。

## Error Messages 错误消息

使用错误信息表来存储错误信息文本，针对这些信息可以输入多种语言。显示给用户的信息依赖于用户登录时用户选择的语言。

# Performance 性能优化

运行性能是衡量程序好坏的重要指标之一，应该避免使用导致性能低下的语句。.

## SQL

关于SAP系统标准表的插入、更新和删除操作，原则上不通过SQL语句进行操作。

关于SQL语句，仅使用Open SQL，不使用Native SQL。

## SELECT

尽量在最开始先把要处理数据通过SELECT语句保存到内部表，然后对该内部表进行逻辑操作，在数据量不大的情况下，不推荐使用Select/EndSelect语句，非特殊情况，不允许对Select/EndSelect语句进行嵌套使用。

| 推荐 |
| --- |
| SELECT \* From KNA1  INTO TABLE GT\_KNA1  WHERE KTOKD = P\_KTOKD.  Loop at GT\_KNA1 into GS\_KNA1.  ……  EndLoop. |
| 不推荐 |
| SELECT \* FROM KNA1 WHERE KTOKD = P\_KTOKD.  ……  ENDSELECT. |
| 不允许使用 |
| SELECT \* FROM KNA1 WHERE KTOKD = P\_KTOKD.  SELECT \* FROM XXXX  WHERE XXXX = KNA1-KTOKD.  ENDSELECT.  ENDSELECT. |

在处理单条数据时，尽量使用Select Single语句。

在WHERE语句的条件指定key项目的时候，必须按照在字典设定的顺序(事务代码 SE11），当检索条件不是key时，首先查看该数据库表是否存在其他索引，如果不存在相关索引并且该开发的使用频度较高，则可以申请创建对字段的索引。

| 推荐 |
| --- |
| SELECT AUGDT BSCHL WRBTR FROM BSEG  INTO TABLE ITAB\_BSEG  WHERE BUKRS = P\_BUKRS  AND BELNR = P\_BELNR  AND GJAHR = P\_GJAHR.  ....... |
| 不推荐 |
| SELECT AUGDT BSCHL WRBTR FROM BSEG  INTO TABLE ITAB\_BSEG  WHERE BELNR = P\_BELNR  AND GJAHR = P\_GJAHR  AND BUKRS = P\_BUKRS.  ....... |

LIKE, NOT, <> 的WHERE语句因为性能差，所以请尽可能避免使用。

| 不推荐 |
| --- |
| SELECT KUNNR NAME1 TELF1 FROM KNA1  INTO TABLE ITAB\_KNA1  WHERE KUNNR LIKE CNS\_PRT.  ....... |

当的确需要使用For All Entries in语句时，必须在Select语句前增加对内表的检查，确保该内表不为空，否则Select语句将会获取数据库表中所有的记录，严重影响性能。

| 推荐 |
| --- |
| CHECK LINES(GT\_TEST) GT 0.(此处可以使用CHECK或者IF语句加以判断)  SELECT AUGDT BSCHL WRBTR FROM BSEG  INTO TABLE ITAB\_BSEG  FOR ALL ENTRIES IN GT\_TEST  WHERE BUKRS = GT\_TEST-BUKRS  AND BELNR = GT\_TEST-BELNR  AND GJAHR = GT\_TEST-GJAHR.  ....... |
| 不推荐 |
| SELECT AUGDT BSCHL WRBTR FROM BSEG  INTO TABLE ITAB\_BSEG  FOR ALL ENTRIES IN GT\_TEST  WHERE BUKRS = GT\_TEST-BUKRS  AND BELNR = GT\_TEST-BELNR  AND GJAHR = GT\_TEST-GJAHR.  ....... |

通过SELECT SINGLE取得的纪录一个一个追加到内部表的时候，使用SELECT SINGLE .. APPENDING TABLE或APPEND。

| 推荐 |
| --- |
| SELECT SINGLE <field> FROM KNA1 APPENDING TABLE ITAB\_KNA1  WHERE KUNNR = P\_KUNNR. |
| 不推荐 |
| SELECT SINGLE \* FROM KNA1 WHERE KUNNR = P\_KUNNR.  MOVE-CORRESPONDING KNA1 TO ITAB\_KNA1.  APPEND ITAB\_KNA1. |

透明表和POOL表进行SELECT的时候，要把包含key项目的条件的所有的条件写到WHERE语句。

| 推荐 |
| --- |
| SELECT SINGLE <field> FROM ZZLT2  WHERE RLDNR = V\_LDGR  AND RRCTY = CNS\_PRT1  AND RVERS = CNS\_PRT2  AND RYEAR = V\_YR  AND BUKRS = V\_CMPNY  AND RACCT = V\_ACCT  AND RCNTR = V\_CNTR. |
| 不推荐 |
| SELECT \* FROM ZZLT2  WHERE RLDNR = V\_LDGR  AND RRCTY = '0'  AND RVERS = '001'  AND RYEAR = V\_YR.  CHECK ZZLT2-BUKRS = V\_CMPNY.  CHECK ZZLT2-RACCT = V\_ACCT.  CHECK ZZLT2-RCNTR = V\_CNTR.  .......  ENDSELECT. |

CLUSTER表进行SELECT的时候，如果需求明确，则尽量使用其组成表进行读取，而不直接使用Cluster表，例如：

FI凭证行项目表BSEG由6张表组成，如果需要供应商未清项表，则只需要Select表BSIK就可以了，而不需要直接使用BSEG表。

不能对Cluster表使用Inner Join。

求最高值、最低值、合计、平均的时候，使用MAX、MIN、SUN、AVG函数。

| 推荐 |
| --- |
| SELECT MAX(MSGNR) FROM T100  INTO V\_MAX\_MSGNR. |
| 不推荐 |
| G\_MAX\_MSGNR = '000'.  SELECT \* FROM T100.  IF T100-MSGNR > V\_MAX\_MSGNR.  V\_MAX\_MSGNR = T100-MSGNR.  ENDIF.  ENDSELECT. |

## JOIN

仅允许对3张表进行Join操作

例：

Select \* From VBAK as A

Inner join VBAP as B

On A~VBELN EQ B~VBELN

Inner join VBUK as C

On A~VBELN EQ C~VBELN.

## UPDATE·INSERT

要对表的项目一起更新的时候，使用SET句会提高性能。但需要一个一个地进行Error处理的情况，不使用这个方法。

| 表的项目列一并更新 |
| --- |
| UPDATE VERI\_CLNT  SET FUNCTION = FUNCTION + 1.  IF SY-SUBRC <> 0.  <Error处理>  ENDIF. |
| 对于各个纪录进行Error处理的情况 |
| SELECT … FROM VERI\_CLNT.  VERI\_CLNT-FUNCTION = VERI\_CLNT-FUNCTION + 1.  UPDATE VERI\_CLNT.  IF SY-SUBRC <> 0.  <Error处理>  ENDIF.  ENDSELECT. |

要根据内部表的数据一并更新表的时候，使用UPDATE .. USING TABLE会提高性能。但需要一个一个地进行Error处理的情况，不使用这个方法。

| 通过内部表进行的表的一并更新 |
| --- |
| UPDATE ZOBJ USING TABLE ITAB\_OBJECTS  IF SY-SUBRC <> 0.  <Error处理>  ENDIF. |
| 对于各个纪录进行Error处理的情况 |
| LOOP AT ITAB\_OBJECTS.  UPDATE ZOBJ  SET TADIR = ITAB\_OBJECTS-TADIR  WHERE PGMID = ITAB\_OBJECTS-PGMID  AND OBJECT = ITAB\_OBJECTS-OBJECT  AND OJB\_NAME = ITAB\_OBJECTS-OBJ\_NAME.  IF SY-SUBRC <> 0.  <Error处理>  ENDIF.  ENDLOOP. |

要根据内部表的数据一并插入更新表的时候，使用INSERT .. FROM TABLE会提高性能。但需要一个一个地进行Error处理的情况，不使用这个方法。

| 通过内部表进行的表的一并更新 |
| --- |
| INSERT KNA1 FROM TABLE ITAB\_KNA1.  IF SY-SUBRC <> 0.  <Error处理>  ENDIF. |
| 对于各个纪录进行Error处理的情况 |
| LOOP AT ITAB\_KNA1 INTO V\_KNA1.  INSERT INTO KNA1 VALUES V\_KNA1.  IF SY-SUBRC <> 0.  <Error处理>  ENDIF.  ENDLOOP. |

## LOCK OBJECT

在使用OpenSQL更新数据库时必须使用Lock Object进行加锁和解锁

## INTERNAL TABLE

不允许使用带Header Line的Internal Table，必须额外声明Work Area来对Internal Table进行操作。

## PROGRAM程序文件

所有的程序文件都必须文档化到一个合理的水平，以促进可读性和可维护性。程序中的所有注释语句必须用通俗的语言来描述代码的目的和功能，而不是重述编程逻辑。另外，为了提高模块化水平，代码必须被分成更小的可工作的逻辑单元，并且在各个层次都加上适当的注释，在Form和程序块的头部，文档必须要用2条注释线来进行分割。

# PROGRAM程序文件

所有的程序文件都必须文档化到一个合理的水平，以促进可读性和可维护性。程序中的所有注释语句必须用通俗的语言来描述代码的目的和功能，而不是重述编程逻辑。另外，为了提高模块化水平，代码必须被分成更小的可工作的逻辑单元，并且在各个层次都加上适当的注释，在Form和程序块的头部，文档必须要用2条注释线来进行分割。

## Program Header程序头

统一使用Pattern：ZMAIN及ZDEFINE。

对程序源代码的修改，应该在源代码中做出对应的版本号，修改者，修改日期，修改描述的记录。

可以用Pattern：ZMODBEGIN/ZMODEND来生成修改注释模板。

| 范例 |
| --- |
| *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** *\*  Report <Report Name>* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** *\** *\*  作者:* *\*  完成日期:        YYYY/MM/DD* *\*  描述:* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** *\*  版本号 日期   作者   修改描述 功能更改说明书* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** *\*  1.  YYYY/MM/DD author  Read dataset from app. Server @001* *\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**  REPORT ZFICFD01 NO STANDARD PAGE HEADING LINE-SIZE 255.  \* Begin of @001  PARAMETERS: p\_ch TYPE c OBLIGATORY DEFAULT ','.  \* End of @001 |

除了描述Description部分为可选项外，其他部分都必须填写完整。

版本号（ Vx.xx ）：其中最后一位为修改bug、优化或完善程序时使用，前面的两位必须与技术说明书的版本保持一致。

程序的修改历史记录 （Change History ） 在程序发布后进行任何修改都必须维护完整。

代码编写宽度不要超过72列。

# Form 子程序

所有程序必须利用程序模板中的Form Header的模板。这个Form头是在该Form被创建之前，通过双击 PERFORM <name>语句来创建的。