

Sumario:

Este artigo descreve a utilização de alguns comandos do OPENSQl do ABAP/4.

Sobre o Autor:

Uderson Luis Fermino, formado em Ciências da Computação pela Faculdade de Pesquisa e Ensino IPEP, atua no mercado a 2 anos como desenvolvedor Java nas plataformas: (J2SE, J2EE e J2ME), com participação em grandes projetos envolvendo estas tecnologias. É consultor ABAP com experiências em REPORT, ALV (GRID, LIST, BLOCK, OO, TREE, HIERARQUICK), IDOC, ALE, ONLINE, SAPSCRIPT, SMARTFORM, NETWEAVER (JCO, BSP, WebDynpro).

Email:

Uderson@gmail.com

Selects

Os select são instruções SQL (Structure Query Language/ Linguagem de Consulta Estruturada), utilizadas para fazer seleções de dados, de um Banco de dados, os select dentro do SAP, são padronizado de acordo a SAP, é não seguem o Padrão SQL, o padrão de SQL SAP é chamado de OpenSql, que é um tipo de Linguagem diferenciada para, ser utilizada, dentro do Sistema SAP, podendo ter qualquer tipo de SGBD.

Select com inner join

Inner Join que dizer junta interna/junção interna. É a junção de duas ou mais tabelas, que tenham ligações entre si.

```
SELECT  tabela1~campoX  tabela2~campoY  tabela3~campoZ
        INTO TABLE ti_tabela
        FROM    tabela1
        INNER JOIN  tabela2
        ON      tabela1~primarykey      =  tabela2~primarykey
        INNER JOIN  tabela3
        ON      tabela2~primarykey      =  tabela3~primarykey.
```

Select com inner join com Where

```
SELECT  tabela1~campoX  tabela2~campoY  tabela3~campoZ
        INTO TABLE ti_tabela
        FROM    tabela1
        INNER JOIN  tabela2
        ON      tabela1~primarykey      =  tabela2~primarykey
        INNER JOIN  tabela3
        ON      tabela2~primarykey      =  tabela3~primarykey.
        WHERE   tabela1~campoX      =  ' '.
```

Exemplo

```
SELECT MAKT~SPRAS MARA~MATNR MAKT~MAKTX MARA~LVORM
        MARA~MEINS MARA~BRGEW MARA~NTGEW MARC~STEUC
        MARA~LAEDA MARA~AENAM MARA~MTART
INTO TABLE T_ZSTMM004
FROM MARA
INNER JOIN MAKT
ON MARA~MATNR = MAKT~MATNR
INNER JOIN MARC
ON MARC~MATNR = MAKT~MATNR
WHERE MARA~ERSDA >= '10102007'
AND MARA~MTART IN ('ROH','ERSA','HIBE','NLAG')
```

AND MAKT~SPRAS IN ('P','E').

Select simples

Os select simples são usados, para seleção de dados de apenas uma tabela.

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-interna  
  FROM tabela.
```

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-interna  
  FROM tabela.  
  WHERE campox = ''.
```

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-interna  
  FROM tabela.  
  WHERE campox = ''  
  AND CAMPO = ''.
```

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-interna  
  FROM tabela.  
  WHERE campox = ''  
  OR CAMPO = ''.
```

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-interna  
  FROM tabela.  
  WHERE campox IN ( '', '', '' ).
```

Exemplo

For all entrie é usado para seleção de dados de uma tabela, usando chaves de outra tabela interna, já preenchida por um select simples.

For all entries

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-internaX  
  FROM tabelaX.
```

```
SELECT campo1 campo2 ..... campoN  
  INTO TABLE ti_tabela-internaY  
  FROM tabelaY
```

```
FOR ALL ENTRIES IN ti_tabela-internaX
WHERE campo1 = ti_tabela-internaX-campo1
AND campo2 = ti_tabela-internaX-campo2.
```

Exemplo:

```
SELECT BUKRS LIFNR BELNR UMSKZ BLART XBLNR ZFB DT DMBTR
DMBE2 WRBTR BUZEI GJAHR BLDAT SHKZG ZBD1T
FROM BSIK
INTO TABLE T_BSIK
WHERE
BUKRS EQ P_BUKRS
AND
LIFNR IN S_LIFNR
AND
BELNR IN S_BELNR
AND
UMSKZ IN S_UMSKZ.
```

```
SELECT LIFNR NAME1
INTO TABLE T_LFA1
FROM LFA1
FOR ALL ENTRIES IN T_BSIK
WHERE
LIFNR EQ T_BSIK-LIFNR.
```

```
SELECT EBAN~BANFN EBAN~BNFPO EBAN~MATNR EBAN~TXZ01
EBAN~MENGE EBAN~MEINS EBAN~FRGKZ EBAN~ERNAM
EBAN~AFNAM EBAN~BADAT EBAN~WERKS EBAN~ERDAT
LFA1~STCD1
INTO TABLE T_ZSTMM0013
FROM EBAN
LEFT JOIN LFA1
ON EBAN~LIFNR EQ LFA1~LIFNR
WHERE EBAN~BSART EQ 'NB' "Tipo de documento da requisição de compra
AND EBAN~FRGKZ EQ '2' "Código de liberação
AND EBAN~BSTYP EQ 'B' "Categoria do documento de compras
AND EBAN~LOEKZ NE 'X' "Código de eliminação no documento de compras
AND EBAN~ERDAT GL IM_DATE.
```

Select Single

Select single usado para selecionar apenas um registro, em uma variável, ou u campo de uma determinada tabela.

```
SELECT SINGLE campo
FROM tabela
INTO ti_tabelainterna-campo.
```

```
SELECT SINGLE campo
      FROM tabela
      INTO  variavel.
```

```
SELECT SINGLE campo
      FROM tabela
      INTO  variavel.
      WHERE campo_da_tabela = ".
```

Selects sem inserir em tabelas internas

Selects sem inserir dados em tabelas internas, os dados ficam em memória, o desempenho fica maior, as desvantagens são:

- Não é possível verificar os dados em debug.
- Caso seja feito um append ou modify, será feito diretamente na tabela standart
- É necessário fazer select com *, isto causa uma sobrecarga caso a tabela tenha muitos campos.

Exemplo:

```
select single * from mara where matnr = '000000000000000059'.
```

```
select * from marc where matnr = mara-matnr.
endselect.
```

```
loop at mara.
  write: marc-matnr.
endloop.
```

Exemplo.

```
SELECT SINGLE name1 adrn j_1branch land1
      FROM t001w
      INTO (w_dados-name1, w_dados-adrn,w_dados-j_1branch, w_dados-land1)
      WHERE werks EQ w_ekpo-werks.
```

```
SELECT SINGLE street house_num1 city2 post_code1 city1 region tel_number
fax_number
      FROM adrc
```

```
INTO (w_dados-street, w_dados-house_num1, w_dados-city2,  
      w_dados-post_code1, w_dados-city1, w_dados-region,  
      w_dados-tel_number, w_dados-fax_number)  
WHERE addrnumber EQ w_dados-adrn.
```

```
SELECT stcd1 state_insc bukrs  
FROM j_1bbranch  
UP TO 1 ROWS  
INTO (w_dados-stcd1, w_dados-state_insc, w_dados-bukrs)  
WHERE bukrs EQ w_ekpo-werks  
AND branch EQ w_dados-j_1bbranch.  
ENDSELECT.
```

```
SELECT SINGLE waers  
FROM t001  
INTO w_dados-waers  
WHERE bukrs EQ w_dados-bukrs.
```

Exemplo de Select Single com Inner Join

```
SELECT SINGLE setlinet~descript  
INTO v_descript  
FROM setleaf  
INNER JOIN setlinet  
ON setlinet~setname EQ setleaf~setname  
AND setlinet~lineid EQ setleaf~lineid  
WHERE setleaf~setname EQ 'IVA'  
AND setleaf~valfrom EQ v_valfrom  
AND setlinet~langu EQ 'PT'.
```

Neste select está se selecionando um campo, e colocando dentro de uma tabela interna que já tem dados, porém este campo na tabela interna se encontra vazio, para preencher este campo em todas as tuplas, é necessário fazer uma loop nesta tabela, fazer o select single do campo a ser preenchido, e dar um modify, na tabela com o index Sy-tabix, que está guardando a posição da tupla atual que necessita ser preenchida, transportando o campo que é necessário ser preenchido.

```
Loop at T_BKPF  
SELECT SINGLE EBELN  
INTO T_BKPF-EBELN  
FROM RSEG  
WHERE BELNR EQ T_BKPF-AWKEY(10)
```

```
AND GJAHR EQ T_BKPF-AWKEY+10(4)
AND BUZEI EQ T_BSIK-BUZEI.
```

```
MODIFY T_BKPF INDEX SY-TABIX TRANSPORTING EBELN.
Endloop.
```

Inner Join com For All Entries

```
SELECT KNA1~KUNNR KNA1~NAME1 KNA1~NAME2
       KNA1~ORT01 KNA1~REGIO
       TD03~MATERIAL TD03~CPEMBAL
       TD03~CARREGAR TD03~UND_MEDIDA
INTO T_CLIMAIL
FROM KNA1
INNER JOIN ZOC0003_TD AS TD03
  ON KNA1~KUNNR EQ TD03~CLIENTE
  FOR ALL ENTRIES IN T_CLIENTES
WHERE KNA1~KUNNR EQ T_CLIENTES-CLIENTE.
```

READ TABLE

Toda tabela manipulada, dentro do sap é criada uma HeaderLine, que é um vetor de dado do tipo de uma tupla, de uma determinada tabela, a Header Line é uma copia dos dados do registro em que o ponteiro, está posicionado no momento, uma header line pode ser visualizada claramente dentro de um loop

HeaderLine da TabelaX

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima

LOOP AT TabelaX

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima
00002	14.12.2006	Álcool	Matéria Prima
00003	20.25.2000	Gasolina	Matéria Prima
00004	11.09.1997	Óleo Diesel	Matéria Prima
00005	13.08.2005	BioDiesel	Matéria Prima

ENDLOOP

HeaderLine da TabelaX

Matnr	Datnam	Nome	Descrição
00002	14.12.2006	Álcool	Matéria Prima

LOOP AT TabelaX

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima
00002	14.12.2006	Álcool	Matéria Prima
00003	20.25.2000	Gasolina	Matéria Prima
00004	11.09.1997	Óleo Diesel	Matéria Prima
00005	13.08.2005	BioDiesel	Matéria Prima

ENDLOOP

HeaderLine da TabelaX

Matnr	Datnam	Nome	Descrição
00003	20.25.2000	Gasolina	Matéria Prima

LOOP AT TabelaX

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima
00002	14.12.2006	Álcool	Matéria Prima
00003	20.25.2000	Gasolina	Matéria Prima
00004	11.09.1997	Óleo Diesel	Matéria Prima
00005	13.08.2005	BioDiesel	Matéria Prima

ENDLOOP

Observação:

LOOP AT

O Comando LOOP AT ENDLOOP.

Serve para andar em cada registro de uma tabela interna, podendo alterar, excluir, incluir dados nos campos deste registro, atual, o exemplo acima mostra o LOOP AT, em uma tabela interna, chamada de TabelaX, Cada iteração em um

registro, a HeaderLine é modificado para a posição atual, do LOOP AT, a variável SY-TABIX, é a variável, que armazena o numero atual da iteração.

Cont READ TABLE

O comando READ TABLE, é usado dentro de um LOOP AT, para fazer a leitura de um registro de uma determinada tabela, onde está tabela, geralmente deve ter campos de ligações com a tabela que está dentro do LOOP AT, em cada iteração do LOOP AT, deve-se chamar o comando HEAD TABLE verificando as condições de ligações de chaves Primarias e estrangeiras, quando o comando for chamado, a HeaderLine da tabela que está sendo chamada dentro do HEAD TABLE, terá o valor da condição de chaves.

Exemplos:

TabelaX

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima
00002	14.12.2006	Álcool	Matéria Prima
00003	20.12.2000	Gasolina	Matéria Prima
00004	11.09.1997	Óleo Diesel	Matéria Prima
00005	13.08.2005	BioDiesel	Matéria Prima

TabelaY

Matnr	Docnum	Docdat	Descrição
00001	10.10.2000	10.10.2000	Matéria Prima
00002	12.01.2006	19.12.1999	Matéria Prima
00003	24.12.2000	11.11.1996	Matéria Prima
00004	18.03.1997	09.09.1997	Matéria Prima
00005	11.07.2005	12.06.2000	Matéria Prima

TabelaZ

Matnr	Datnam	Nome	Descrição	Docnum	Docdat	Descrição

As primeiras iterações, as headerlines, ficariam destas formas.

LOOP AT TabelaX

HeaderLine da TabelaX

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima

Matnr	Datnam	Nome	Descrição
00001	10.10.2000	Cana de Açúcar	Matéria Prima
00002	14.12.2006	Álcool	Matéria Prima
00003	20.25.2000	Gasolina	Matéria Prima
00004	11.09.1997	Óleo Diesel	Matéria Prima
00005	13.08.2005	BioDiesel	Matéria Prima

READ TABLE TabelaY WITH KEY matnr = TabelaX -matnr.

Matnr	Docnum	Docdat	Descrição
00001	10.10.2000	10.10.2000	Matéria Prima

TabelaZ-Matnr = TabelaX-Matnr.
 TabelaZ-Datnam = TabelaX-Datnam.
 TabelaZ-Nome = TabelaX-Nome.
 TabelaZ-Descrição = TabelaX-Descrição.
 TabelaZ-Docnum = TabelaY-Docnum.
 TabelaZ- Docdat = TabelaY-Docdat.

ENDLOOP

Imprime:

00001 TabelaZ-Matnr
 10.10.2000 TabelaZ-Datnam
 Cana de Açúcar TabelaZ-Nome
 Matéria Prima TabelaZ-Descrição
 10.10.2000 TabelaZ-Docnum
 10.10.2000 TabelaZ- Docdat

Exemplo real em abap.

TABLES: MARA, MARC.
 DATA: BEGIN OF T_MARA OCCURS 0,
 MATNR LIKE MARA-MATNR,
 ERSDA LIKE MARA-ERSDA,
 ERNAM LIKE MARA-ERNAM,
 LAEDA LIKE MARA-LAEDA,
 AENAM LIKE MARA-AENAM,
 END OF T_MARA.

DATA: BEGIN OF T_MARC OCCURS 0,
 MATNR LIKE MARC-MATNR,
 PSTAT LIKE MARC-PSTAT,
 LVORM LIKE MARC-LVORM,
 BWTTY LIKE MARC-BWTTY,

```
XCHAR LIKE MARC-XCHAR,  
MMSTA LIKE MARC-MMSTA,  
END OF T_MARC.
```

```
DATA: BEGIN OF T_UNIDADOS OCCURS 0,  
      MATNR LIKE MARA-MATNR,  
      ERSDA LIKE MARA-ERSDA,  
      ERNAM LIKE MARA-ERNAM,  
      LAEDA LIKE MARA-LAEDA,  
      PSTAT LIKE MARA-PSTAT,  
      LVORM LIKE MARA-LVORM,  
      MTART LIKE MARA-MTART,  
      BWTTY LIKE MARC-BWTTY,  
      XCHAR LIKE MARC-XCHAR,  
      MMSTA LIKE MARC-MMSTA,  
END OF T_UNIDADOS.
```

```
SELECT MATNR ERSDA ERNAM LAEDA AENAM VPSTA PSTAT LVORM MTART  
FROM MARA  
INTO TABLE T_MARA  
WHERE MATNR > '000000000000000732'.
```

```
SELECT MATNR PSTAT LVORM BWTTY XCHAR MMSTA MMSTD MAABC KZKRI  
EKGRP AUSME  
INTO TABLE T_MARC  
FROM MARC  
FOR ALL ENTRIES IN T_MARA  
WHERE MATNR = T_MARA-MATNR.  
LOOP AT T_MARA.  
  READ TABLE T_MARC WITH KEY MATNR = T_MARA-MATNR.  
    T_UNIDADOS-MATNR = T_MARA-MATNR.  
    T_UNIDADOS-ERSDA = T_MARA-ERSDA.  
    T_UNIDADOS-ERNAM = T_MARA-ERNAM.  
    T_UNIDADOS-LAEDA = T_MARA-LAEDA.  
    T_UNIDADOS-AENAM = T_MARA-AENAM.  
    T_UNIDADOS-VPSTA = T_MARA-VPSTA.  
    T_UNIDADOS-PSTAT = T_MARA-PSTAT.  
    T_UNIDADOS-LVORM = T_MARA-LVORM.  
    T_UNIDADOS-MTART = T_MARA-MTART.  
    T_UNIDADOS-BWTTY = T_MARC-BWTTY.  
    T_UNIDADOS-XCHAR = T_MARC-XCHAR.  
    T_UNIDADOS-MMSTA = T_MARC-MMSTA.  
    T_UNIDADOS-MMSTD = T_MARC-MMSTD.  
    T_UNIDADOS-MAABC = T_MARC-MAABC.  
    T_UNIDADOS-KZKRI = T_MARC-KZKRI.  
    T_UNIDADOS-EKGRP = T_MARC-EKGRP.  
    T_UNIDADOS-AUSME = T_MARC-AUSME.  
  APPEND T_UNIDADOS.
```

```
ENDLOOP.
```

Clear, Refresh e Free

CLEAR: TABELA_INTERNA = Limpa a Header Line de uma tabela Interna

CLEAR: TABELA_INTERNA[] = Limpa a tabela interna sem limpar a Header Line

REFRESH TABELA_INTERNA = Limpa a tabela interna sem limpar a Header Line

FREE TABELA_INTERNA = Limpa a tabela interna sem limpar a Header Line

Exemplo:

Selecione os campos da tabela KNA1, insira dentro da tabela interna TABNKA1, com as condições de a chave primária = '0000002001'.

```
SELECT KUNNR NAME1 NAME2 ORT01 STRAS  
FROM KNA1  
INTO TABLE T_KNA1  
WHERE KUNNR < '0000002001'.
```

```
LOOP AT T_KNA1.  
WRITE: / T_KNA1-KUNNR,T_KNA1-NAME1,T_KNA1-NAME2, T_KNA1-ORT01,  
T_KNA1-STRAS.  
ENDLOOP.
```

Limpa a header e a tabela interna

`clear: T_KNA1, T_KNA1[].`

Comando SELECT

O comando select é usado para acessar e selecionar dados de tabelas internas do SAP. Por existirem diversas variações do mesmo comando, é fundamental que o programador saiba o mecanismo de funcionamento de cada uma delas pois só assim poderá dar ao programa uma performance satisfatória.

Variações:

1. **SELECT * FROM dbtab.**

```
...  
ENDSELECT.
```

Seleciona dados de uma tabela SAP num processo de “loop” que começa no select e termina no endselect. A cada passagem pelo “loop” temos um elemento lido e selecionado. É necessário que se coloque após o ENDESELCT uma condição de checagem de dados selecionados:

```
If sy-subrc ne 0.  
  Write: 'Nenhum dado foi selecionado'.  
Endif.
```

Se sy-subrc = 0 : pelo menos um dado foi selecionado

Se sy-subrc = 4 : nenhum dados foi lido

Exemplo:

```
SELECT * FROM BSEG.  
...  
ENDSELECT.  
If sy-subrc ne 0.  
  Write: 'Não tem dado na tabela BSEG'.  
Endif.
```

Adições :

1a) ... WHERE Condition

Seleciona apenas os dados que satisfazem a condição especificada.

Exemplo:

```
SELECT * FROM T001 WHERE BUKRS EQ '0001'.  
...
```

ENDSELECT.

- 1b) ... ORDER BY f'1...fn**
... ORDER BY PRIMARY KEY

Organiza os dados em ordem ascendente de acordo com os campos especificados (f1...fn).

Exemplo:

```
SELECT * FROM T001 ORDER BY MSGNR DESCENDING ARBGB ASCENDING.  
...  
ENDSELECT.
```

- 1c) ... UP TO n ROWS.**

Seleciona um número máximo de dados.

Exemplo:

```
SELECT * UP TO 100 ROWS FROM T001 WHERE...  
...  
ENDSELECT.
```

- 2) SELECT * FROM dbtab INTO TABEL itab.**

Os dados são selecionados e colocados na tabela interna itab de uma só vez. Não há mais o processo de loop e portanto não há mais ENDSELECT. Os dados novos da tabela interna são gravados por cima dos antigos.

É importante ressaltar que o * pode ser substituído pelos nomes dos campos da tabela, agilizando assim o processo e melhorando a performance.

Exemplo:

```
SELECT * FROM T001 INTO TABLE TAUX.
```

Adições:

- 2a) WHERE**
2b) ORDER BY
2c) UP TO n ROWS

3) SELECT * FROM dbtab APPENDING TABLE itab.

Mesmo processo do item 2, só que os dados novos são inseridos sem apagar os antigos.

Exemplo:

```
SELECT * FORM T100 APPENDING TABLE TAUX .
```

Adições:

- 3a) WHERE**
- 3b) ORDER BY**
- 3c) UP TO n ROWS**

4) SELECT SINGLE * FROM dbtab WHERE f1 = g1 AND... AND fn +

Seleciona apenas um único dado que satisfaça condições do where.
OBS.: Neste caso todas as chaves (índices da tabela) devem ser satisfeitos.

Exemplo:

```
SELECT SINGLE * FORM T100 WHERE BUKRS = '02'.
```

5) SELECT * FROM dbtab APPENDING CORRESPONDING FIELDS OF TABLE itab.

Mecanismo semelhante ao item 2, e deve ser usado quando a sintaxe do item 2 não puder ser usada.
Há diferenças de performance.

Exemplo:

```
SELECT * FORM T100 APPENDING CORRESPONDING FIELDS OF TABLE TAUX .
```

Onde TAUX recebeu a estrutura da tabela T100 (commando INCLUDE STRUCTURE).

6) SELECT * FROM dbtab FOR ALL ENTRIES in itab WHERE...

Usado quando selecionamos dados de uma tabela e precisamos de dados de outra tabela para compor as condições do where.

Exemplo:

```
SELECT * FROM BSEG FOR ALL ENTRIES IN T_BKPF
      WHERE BUKRS = T_BKPF-BUKRS AND
      BELNR = T_BKPF-BELNR .
```

Onde T_BKPF é uma tabela interna que recebeu a tabela BKPF. Este tipo de comando é utilizado entre tabelas internas.

TIPOS DE SELECT EXISTENTES E MAIS UTILIZADOS

1- SELECT * FROM ... <tabela>

Quando não se impõe nenhum tipo de restrição, ocorre uma varredura seqüencial dos registros da tabela. Quando se utiliza grandes tabelas, isso obviamente afeta o runtime. Performance: Select * seleciona todas as colunas de uma tabela. É melhor sempre especificar as colunas, pois em caso de tabelas com muitas colunas, prejudicará a performance.

2- SELECT * FROM <tabela> WHERE <campo> EQ <conteúdo>.

Lê todos os registros da tabela especificada onde o campo é igual ao conteúdo especificado. Performance: Select * Where seleciona todas as colunas de uma tabela de acordo com a condição de where. É melhor sempre especificar as colunas, pois em caso de tabelas com muitas colunas, prejudicará a performance.

3- SELECT * FROM <tabela> WHERE <table field> BETWEEN <field1> AND <field2>.

Exemplo: field1 = 100 e field2 = 500. pega inclusive 100 e 500. Você trabalha com o range.

4- SELECT * FROM <tabela> WHERE <table field> LIKE ... ‘_R%’.

 R a primeira letra não imprta o que virá.
 % a segunda deverá ser R (eu defini)
 não importa a seqüência de caracteres que virá

5- SELECT * FROM <tabela> WHERE <table field> IN (..... ,).

Exemplo: select * from <table> where campo1 in (123,1000) – podem ser valores ou literais.

 É como perguntar se campo1 é 123 ou 1000.

6- SELECT * FROM <tabela> WHERE <table field> IN <internal table>.

Exemplo:

DATA: begin of ITAB occurs 10,
 Sign(1), options(2), low like sflight-price, high like sflight-proce,
 End of ITAB.

Move: ‘l’ to itab-sign,
 ‘bt’ to itab-option,
 ‘500’ o itab-low,
 ‘1000’ to itab-high.

Append itab.

Move: ‘l’ to itab-sign,
 ‘bt’ to itab-option,
 ‘440’ to itab-low.

Append itab.

7- SELECT * FROM <tabela> ORDER BY <field1> <field2> ... PRIM ARY KEY.

Obs.: Classifica a tabela interna numa área auxiliar, sem afetar a tabela original. Evitar o uso de sorts dentro de um select. Consome mais tempo que descarregar os dados em uma tabela interna e classificá-los.

8- SELECT * FROM <tabela> BYPASSING BUFFER.

Usado para ler diretamente da tabela original e não do buffer.

OBS.: Select single * sempre com chave completa especificada.
particularmente do ABAP/4 Select * - procurar evitar. Informar as colunas que serão necessárias, apenas. Uso do comando extract (insert header, details) – para relatórios.

9- SELECT * FROM <tabela> APPENDING TABLE <internal table>.

Lê os registros e os Inclui – não sobrepõe – em uma tabela interna.

10- SELECT * FROM <tabela> INTO TABLE <internal table>.

A estrutura da tabela interna deve corresponder à estrutura da tabela que está sendo acessada. O sistema lê os registros em conjunto, não individualmente através de um LOOP e ir gravando os registros, uma a um.

11- SELECT ... INTO CORRESPONDING FIELDS OF TABLE <internal table>.

Neste caso a estrutura da tabela interna não precisa corresponder à estrutura da tabela que está sendo acessada. Movimentará os registros para as colunas definidas na tabela interna que possuam nome igual ao da tabela acessada.

Obs.: Corresponding ou appending não exigem o endselect.

12- SELECT * APPENDING COREESPONDING FIELDS OF TABLE <internal table>.

Lê e grava (não sobrepõe) os dados em uma tabela interna que possua nomes idênticos aos nomes da tabela que está sendo lida.

13- SELECT SINGLE * FROM <tabela> WHERE <campo> EQ <conteúdo>.

Toda vez que se usa select single * a chave primária completa deve ser especificada. Se a chave especificada não é qualificada, você receberá uma mensagem de warning e a performance ficará prejudicada. No caso de haver a necessidade de acessar um único registro via select, as opções são:

```
Select * ... seguido do comando EXIT  
ou  
select * ... UP TO 1 ROW.
```

Neste caso não é necessário especificar a chave completa.

14- SELECT <a1> <a2> ... INTO (<f1>, <f2>, ...) FROM ... <tabela> WHERE ...

Lê as colunas especificadas (a1, a2). Após INTO deverão ser especificadas as áreas de trabalho auxiliares (f1, f2). O número de colunas lidas deverá ser igual ao número de work-aeras especificadas.

**15- SELECT MAX (campo)
MIN (campo)
AVG (campo)
COUNT (*) FROM <tabela> INTO (... , ... , ... , ...)
WHERE**

AVG e SUM somente para campos numéricos.

Não se usa endselect.

Mais rápido fazer uma rotina “a mão” que utilizar esse comando.

17- SELECT * FROM (<tabela>) INTO <work area>.

Exemplo:

Data: begin of WA,
Line(1000,
end of WA.

Parameters: tablename(10) default 'SPFLI'.

*** Especificando o nome da tabela em tempo dinamicamente no comando select sempre consome mais tempo de CPU que especificando estatiamente no programa.

Select * from (tablename) into WA.
Write:
Endselect.

**18- SELECT * FROM <tabela> FOR ALL ENTRIES IN <tabela interna>
WHERE campo1 = conteúdo AND
WHERE campo2 = conteúdo .**

Defino uma tabela interna. Alimento os campos desta tabela interna. (move e append).

No meu select campo1 e campo2 serão os campos definidos e ealimentados na tabela interna.

**19- SELECT carrid MIN(price) MAX(price) INTO (carid, minimum, maximum)

FROM sflight
GROUP BY carrid.**

Todos os campos que eu quero que apareçam na minha lista eu preciso especificar após a cláusula GROUP BY. (carrid, maximum e minimum são campos auxiliares. Se o nome do database não é conhecido até o runtime, não se pode especificar a cláusula GROUP BY.