**Comandos de repetição**

**NOTA:** Em ABAP, **SY-INDEX** e **SY-TABIX** são variáveis do sistema que desempenham funções diferentes em relação a loops e tabelas internas. **SY-INDEX** rastreia o número de iterações em um loop, como DO ou WHILE, enquanto **SY-TABIX** indica o índice da linha atual em uma tabela interna.

**LOOP AT**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WRITE ‘LOOP AT’.

NEW-LINE.

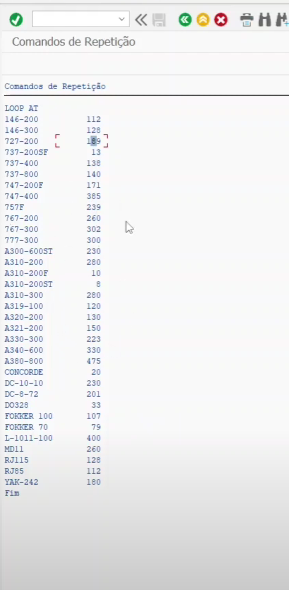
LOOP AT lt\_plane INTO ls\_plane. (este loop vai passar por toda a tabela e trazer os dados para a nossa estrutura)

WRITE: ls\_plane-planetype, ls\_plane-seatsmax. (em cada linha que passa, vai escrever na tela o tipo de avião e o número máximo de lugares, que são colunas existentes nestas tabelas já existentes no SAP.)

NEW-LINE.

ENDLOOP.

O resultado é o seguinte:



* A mesma coisa mas escrevendo na tela só os aviões que têm mais que 200 lugares máximos:

WRITE ‘LOOP AT’.

NEW-LINE.

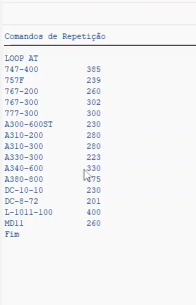
LOOP AT lt\_plane INTO ls\_plane WHERE seatsmax > 200.

WRITE: ls\_plane-planetype, ls\_plane-seatsmax.

NEW-LINE.

ENDLOOP.

O resultado é o seguinte:



Onde escrevemos o “WHERE”, podemos escrever AND, OR, planetype = ‘blabla’, ou qualquer outro filtro que queiramos usar.

**WHILE**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WRITE ‘WHILE’

NEW-LINE.

WHILE sy-index <= lines ( lt\_plane ). (o loop lê cada linha da tabela (usando SY-INDEX que aumenta 1 número de cada vez que passa lá), e enquanto o SY-INDEX for menor que o número de linhas existentes na tabela, vai escrever na tela o tipo de avião e o número de lugares.)

READ TABLE lt\_plane INTO ls\_plane INDEX sy-index.

WRITE ls\_plane-planetype, ls\_plane-seatsmax.

NEW-LINE.

ENDWHILE.

No WHILE não podemos escrever WHERE nem esse tipo de condições que escrevemos no LOOP AT porque a condição já é feita logo a seguir ao WHILE. É com esse tipo de condições que este loop trabalha.

**DO**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WRITE ‘DO x TIMES’

NEW-LINE.

DO 3 TIMES. (informamos quantas vezes queremos repetir o loop - neste caso são 3 vezes - e o loop repete este número de vezes. Neste caso vai mostrar só os 3 primeiros aviões porque depois de passar 3 vezes na tabela, termina o loop.)

READ TABLE lt\_plane INTO ls\_plane INDEX sy-index.

WRITE ls\_plane-planetype, ls\_plane-seatsmax.

NEW-LINE.

ENDDO.

**LOOP DENTRO DE LOOP**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WRITE ‘LOOP dentro de LOOP’

NEW-LINE.

LOOP AT lt\_scarr INTO ls\_scarr. (este loop vai pssar por todas as companhias aéreas existentes na tabela e escrever o nome delas)

WRITE ls\_scarr-carrname.

WRITE sy-uline. (este comando serve para criar uma linha de separação para ficar visualmente mais bonito)

LOOP AT lt\_spfli INTO ls\_spfli WHERE carrid = ls\_scarr-carrid (este loop vai escrever os voos existentes em cada uma das companhias aéreas nas quais o outro loop passou.)

WRITE: ls\_spfli-cityfrom, ls\_spfli-cityto.

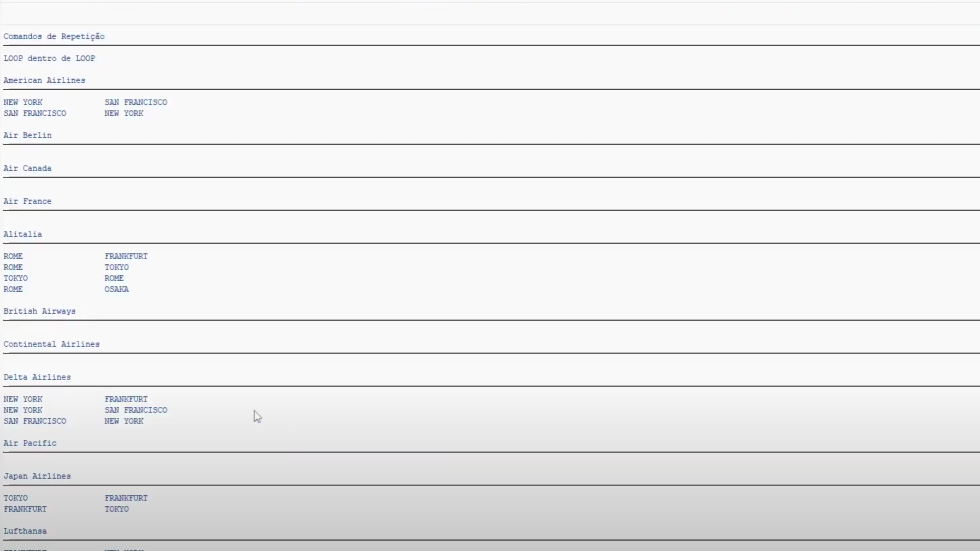
NEW-LINE.

ENDLOOP.

ENDLOOP.

O primeiro loop começa, encontra a primeira companhia aérea, passa para o segundo loop e vê que voos esta tem. Depois de encontrar todos os voos, sai deste loop e volta ao início. O primeiro loop procura a segunda companhia aérea. Depois de encontrar, passa para o segundo loop onde vai procurar que voos esta companhia tem. Depois de encontrar todos os voos disponíveis nesta companhia, sai do segundo loop e volta ao início....

O resultado é este:



**Parar um loop (EXIT)**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WRITE ‘Parar um loop’

NEW-LINE.

LOOP AT lt\_plane INTO ls\_plane.

WRITE: ls\_plane-planetype, ls\_plane-seatsmax.

IF ls-plane-seatsmax > 200.

EXIT.

ENDIF.

ENDLOOP.

(este loop para quando encontra um avião com mais de 200 lugares. Interrompe o loop e continua a ler o programa normalmente. O comando EXIT pode ser usado em qualqeur tipo de loop, este é apenas um exemplo de como podemos usá-lo.)

**Interromper a iteração atual (CONTINUE)**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WRITE ‘Interromper uma iteração’

NEW-LINE.

LOOP AT lt\_plane INTO ls\_plane.

IF ls-plane-seatsmax > 200.

CONTINUE.

ENDIF.

WRITE ls\_plane-planetype, ls\_plane-seatsmax.

ENDLOOP.

(usamos este loop quando numa linha específica não queremos usar o código, mas não queremos terminar o loop. Queremos apenas que ignore aquela linha e continue a ser executado.

Neste exemplo, sempre que o número de lugares for maior que 200, o loop vai terminar o seu processo para essa linha e não vai seguir o comando “WRITE” porque termina o seu trabalho logo depois do IF. Então volta ao início e procura a linha seguinte, se for menor que 200, faz o seu código até ao fim e segue o comando WRITE; se for maior que 200, termina novamente e volta ao início sem chegar a ler o comando WRITE)

**Loops infinitos (WHILE, DO)**

REPORT ZZADRI\_COMANDOS\_REPETIÇÃO

\* Declaração de tabela interna

DATA: lt\_plane TYPE STANDARD TABLE OF saplane.

DATA: ls\_plane TYPE saplane.

DATA: lt\_scarr TYPE STANDARD TABLE OF scarr.

DATA: ls\_scarr TYPE scarr.

DATA: lt\_spfli TYPE STANDARD TABLE OF spfli.

DATA: ls\_spfli TYPE spfli.

\* Carregar dados de cada tabela:

\* Tabela de aviões

SELECT \*

INSTO TABLE lt\_plane

FROM saplane.

\* Tabela de companhias aereas

SELECT \*

INSTO TABLE lt\_scarr

FROM scarr.

\* Tabela de voos

SELECT \*

INSTO TABLE lt\_spfil

FROM spfil.

WHILE 1 = 1.

IF sy-index > 5 .

EXIT.

ENDIF.

ENDWHILE.

Nos loops infinitos é muito importante ter sempre uma condição para ele parar, para não ficar a rodar infinitamente no programa.

Este tipo de loop é frequentemente usado para verificar processos assíncronos. Por exemplo, precisamos de ativar uma ação quando uma nota fiscal é lançada, então criamos um loop infinito para ir rodando sempre, verificando se a nota fiscal já foi lançada ou não.

O DO infinito é semelhante:

DO.

IF sy-index > 5 .

EXIT.

ENDIF.

ENDDO.