CREATE OR REPLACE PACKAGE BODY pck\_sfp IS

/\*

1º ->C processo criado

2º C->A mudou de criado para apto

3º A->E mudou de apto para executando

4º E->A mudou de executando para apto

5º E->B mudou de executando para bloqueado

6º E->D mudou de executando para destruido

\*/

v\_processos NUMBER;

v\_tempo NUMBER;

v\_total\_ciclos\_cpu NUMBER;

v\_criacao\_seq NUMBER;

--------------------------------------------------------------------------------------

--/set\_new\_processo

PROCEDURE set\_new\_processo(p\_ciclo\_cpu IN NUMBER) IS

v\_pid NUMBER;

v\_ciclos NUMBER;

BEGIN

--v\_pid := t\_criacao\_seq.nextval;

v\_criacao\_seq := v\_criacao\_seq + 1;

v\_pid := v\_criacao\_seq;

v\_ciclos := ROUND(dbms\_random.value(100, 300));

INSERT INTO t\_criacao

VALUES

(v\_pid

,'C'

,v\_ciclos

,0);

set\_processo\_hist(v\_pid, 'C', v\_ciclos, p\_ciclo\_cpu);

set\_log\_pid(p\_ciclo\_cpu, v\_pid, NULL, 'C', '1º ->C');

END;

--/

--/set\_fila\_apto

PROCEDURE set\_fila\_apto(p\_pid IN NUMBER) IS

BEGIN

INSERT INTO t\_apto

VALUES

(t\_apto\_seq.nextval

,p\_pid);

END;

--/

--/set\_fila\_apto\_esperando

PROCEDURE set\_fila\_apto\_esperando IS

BEGIN

UPDATE t\_criacao

SET esperando = esperando + 1

WHERE estado = 'A';

END;

--/

--/set\_fila\_executando

PROCEDURE set\_fila\_executando(p\_pid IN NUMBER) IS

BEGIN

INSERT INTO t\_executando

VALUES

(p\_pid);

END;

--/

--/set\_fila\_destruido

PROCEDURE set\_fila\_destruido(p\_pid IN NUMBER) IS

BEGIN

INSERT INTO t\_destruido

VALUES

(p\_pid);

END;

--/

--/set\_fila\_hd

PROCEDURE set\_fila\_hd(p\_pid IN NUMBER

,p\_ciclos IN NUMBER

,p\_ciclo\_cpu IN NUMBER) IS

v\_cod\_fila NUMBER := t\_hd\_seq.nextval;

BEGIN

INSERT INTO t\_hd

VALUES

(v\_cod\_fila

,p\_pid

,p\_ciclos);

set\_fila\_hd\_hist(v\_cod\_fila,p\_ciclo\_cpu,p\_pid, p\_ciclos);

END;

--/

--/set\_fila\_video

PROCEDURE set\_fila\_video(p\_pid IN NUMBER

,p\_ciclos IN NUMBER

,p\_ciclo\_cpu IN NUMBER) IS

v\_cod\_fila NUMBER := t\_video\_seq.nextval;

BEGIN

INSERT INTO t\_video

VALUES

(v\_cod\_fila

,p\_pid

,p\_ciclos);

set\_fila\_video\_hist(v\_cod\_fila,p\_ciclo\_cpu,p\_pid, p\_ciclos);

END;

--/

--/set\_fila\_impressora

PROCEDURE set\_fila\_impressora(p\_pid IN NUMBER

,p\_ciclos IN NUMBER

,p\_ciclo\_cpu IN NUMBER) IS

v\_cod\_fila NUMBER := t\_impressora\_seq.nextval;

BEGIN

INSERT INTO t\_impressora

VALUES

(v\_cod\_fila

,p\_pid

,p\_ciclos);

set\_fila\_impressora\_hist(v\_cod\_fila, p\_ciclo\_cpu, p\_pid, p\_ciclos);

END;

--/

--/set\_processo\_hist

PROCEDURE set\_processo\_hist(p\_pid IN NUMBER

,p\_estado IN VARCHAR2

,p\_ciclos IN NUMBER

,p\_ciclo\_cpu IN NUMBER) IS

BEGIN

INSERT INTO t\_criacao\_hist

VALUES

(p\_pid

,p\_estado

,p\_ciclos

,p\_ciclo\_cpu);

END;

--/

--/set\_fila\_hd\_hist

PROCEDURE set\_fila\_hd\_hist(p\_cod\_fila IN NUMBER

,p\_ciclo\_cpu IN NUMBER

,p\_pid IN NUMBER

,p\_ciclos IN NUMBER) IS

BEGIN

INSERT INTO t\_hd\_hist

VALUES

(p\_cod\_fila

,p\_ciclo\_cpu

,p\_pid

,p\_ciclos);

END;

--/

--/set\_fila\_video\_hist

PROCEDURE set\_fila\_video\_hist(p\_cod\_fila IN NUMBER

,p\_ciclo\_cpu IN NUMBER

,p\_pid IN NUMBER

,p\_ciclos IN NUMBER) IS

BEGIN

INSERT INTO t\_video\_hist

VALUES

(p\_cod\_fila

,p\_ciclo\_cpu

,p\_pid

,p\_ciclos);

END;

--/

--/set\_fila\_impressora\_hist

PROCEDURE set\_fila\_impressora\_hist(p\_cod\_fila IN NUMBER

,p\_ciclo\_cpu IN NUMBER

,p\_pid IN NUMBER

,p\_ciclos IN NUMBER) IS

BEGIN

INSERT INTO t\_impressora\_hist

VALUES

(p\_cod\_fila

,p\_ciclo\_cpu

,p\_pid

,p\_ciclos);

END;

--/

--/set\_estado\_criacao

PROCEDURE set\_estado\_criacao(p\_pid IN NUMBER

,p\_estado IN VARCHAR2) IS

BEGIN

UPDATE t\_criacao

SET estado = p\_estado

WHERE pid = p\_pid;

END;

--/

--/set\_ciclos\_criacao

PROCEDURE set\_ciclos\_criacao(p\_pid IN NUMBER) IS

BEGIN

UPDATE t\_criacao

SET ciclos = ciclos - 1

WHERE pid = p\_pid;

END;

--/

--/set\_ciclos\_hd

PROCEDURE set\_ciclos\_hd IS

v\_pid\_hd NUMBER;

v\_ciclos\_hd NUMBER;

BEGIN

SELECT pid

,ciclos

INTO v\_pid\_hd

,v\_ciclos\_hd

FROM t\_hd

WHERE fila = (SELECT MIN(fila)

FROM t\_hd);

UPDATE t\_hd

SET ciclos = ciclos - 1

WHERE pid = v\_pid\_hd;

IF v\_ciclos\_hd = 0

THEN

set\_fila\_apto(v\_pid\_hd);

set\_estado\_criacao(v\_pid\_hd, 'A');

set\_log\_pid(v\_total\_ciclos\_cpu, v\_pid\_hd, v\_ciclos\_hd, 'A', '5º B->A');

deleta\_pid\_hd(v\_pid\_hd);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

NULL;

END;

--/

--/set\_ciclos\_video

PROCEDURE set\_ciclos\_video IS

v\_pid\_video NUMBER;

v\_ciclos\_video NUMBER;

BEGIN

SELECT pid

,ciclos

INTO v\_pid\_video

,v\_ciclos\_video

FROM t\_video

WHERE fila = (SELECT MIN(fila)

FROM t\_video);

UPDATE t\_video

SET ciclos = ciclos - 1

WHERE pid = v\_pid\_video;

IF v\_ciclos\_video = 0

THEN

set\_fila\_apto(v\_pid\_video);

set\_estado\_criacao(v\_pid\_video, 'A');

set\_log\_pid(v\_total\_ciclos\_cpu, v\_pid\_video, v\_ciclos\_video, 'A', '5º B->A');

deleta\_pid\_video(v\_pid\_video);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

NULL;

END;

--/

--/set\_ciclos\_impressora

PROCEDURE set\_ciclos\_impressora IS

v\_pid\_impressora NUMBER;

v\_ciclos\_impressora NUMBER;

BEGIN

SELECT pid

,ciclos

INTO v\_pid\_impressora

,v\_ciclos\_impressora

FROM t\_impressora

WHERE fila = (SELECT MIN(fila)

FROM t\_impressora);

UPDATE t\_impressora

SET ciclos = ciclos - 1

WHERE pid = v\_pid\_impressora;

IF v\_ciclos\_impressora = 0

THEN

set\_fila\_apto(v\_pid\_impressora);

set\_estado\_criacao(v\_pid\_impressora, 'A');

set\_log\_pid(v\_total\_ciclos\_cpu, v\_pid\_impressora, v\_ciclos\_impressora, 'A', '5º B->A');

deleta\_pid\_impressora(v\_pid\_impressora);

END IF;

EXCEPTION

WHEN no\_data\_found THEN

NULL;

END;

--/

--/set\_log\_pid

PROCEDURE set\_log\_pid(p\_ciclo\_cpu IN NUMBER

,p\_pid IN NUMBER

,p\_ciclo\_pid IN NUMBER

,p\_estado IN VARCHAR2

,p\_acao IN VARCHAR2) IS

BEGIN

INSERT INTO t\_log\_pid

VALUES

(p\_ciclo\_cpu

,p\_pid

,NVL(p\_ciclo\_pid, 0)

,p\_estado

,p\_acao);

END;

--/

--/get\_estado

FUNCTION get\_estado(p\_pid IN NUMBER) RETURN VARCHAR2 IS

v\_estado t\_criacao.estado%TYPE;

BEGIN

SELECT estado

INTO v\_estado

FROM t\_criacao

WHERE pid = p\_pid;

RETURN v\_estado;

END;

--/

--/get\_ciclos\_criacao

FUNCTION get\_ciclos\_criacao(p\_pid IN NUMBER) RETURN NUMBER IS

v\_ciclos t\_criacao.ciclos%TYPE;

BEGIN

SELECT ciclos

INTO v\_ciclos

FROM t\_criacao

WHERE pid = p\_pid;

RETURN v\_ciclos;

END;

--/

--/get\_reg\_executando

FUNCTION get\_reg\_executando RETURN BOOLEAN IS

v\_existe NUMBER;

BEGIN

SELECT DISTINCT 1

INTO v\_existe

FROM t\_executando;

RETURN TRUE;

EXCEPTION

WHEN no\_data\_found THEN

RETURN FALSE;

END;

--/

--/existe\_reg\_hd

FUNCTION existe\_reg\_hd RETURN BOOLEAN IS

v\_existe NUMBER;

BEGIN

SELECT DISTINCT 1

INTO v\_existe

FROM t\_hd;

RETURN TRUE;

EXCEPTION

WHEN no\_data\_found THEN

RETURN FALSE;

END;

--/

--/existe\_reg\_video

FUNCTION existe\_reg\_video RETURN BOOLEAN IS

v\_existe NUMBER;

BEGIN

SELECT DISTINCT 1

INTO v\_existe

FROM t\_video;

RETURN TRUE;

EXCEPTION

WHEN no\_data\_found THEN

RETURN FALSE;

END;

--/

--/existe\_reg\_impressora

FUNCTION existe\_reg\_impressora RETURN BOOLEAN IS

v\_existe NUMBER;

BEGIN

SELECT DISTINCT 1

INTO v\_existe

FROM t\_impressora;

RETURN TRUE;

EXCEPTION

WHEN no\_data\_found THEN

RETURN FALSE;

END;

--/

--/get\_sum\_ciclos\_pid

FUNCTION get\_sum\_ciclos\_pid RETURN NUMBER IS

v\_retorno NUMBER;

BEGIN

SELECT NVL(SUM(ciclos), 1)

INTO v\_retorno

FROM t\_criacao;

RETURN v\_retorno;

END;

--/

--/get\_sum\_ciclos\_hd

FUNCTION get\_sum\_ciclos\_hd RETURN NUMBER IS

v\_retorno NUMBER;

BEGIN

SELECT SUM(ciclos)

INTO v\_retorno

FROM t\_hd;

RETURN v\_retorno;

END;

--/

--/get\_sum\_ciclos\_video

FUNCTION get\_sum\_ciclos\_video RETURN NUMBER IS

v\_retorno NUMBER;

BEGIN

SELECT SUM(ciclos)

INTO v\_retorno

FROM t\_video;

RETURN v\_retorno;

END;

--/

--/get\_sum\_ciclos\_impressora

FUNCTION get\_sum\_ciclos\_impressora RETURN NUMBER IS

v\_retorno NUMBER;

BEGIN

SELECT SUM(ciclos)

INTO v\_retorno

FROM t\_impressora;

RETURN v\_retorno;

END;

--/

--/deleta\_pid\_apto

PROCEDURE deleta\_pid\_apto(p\_pid NUMBER) IS

BEGIN

DELETE t\_apto

WHERE pid = p\_pid;

END;

--/

--/deleta\_pid\_executando

PROCEDURE deleta\_pid\_executando(p\_pid NUMBER) IS

BEGIN

DELETE t\_executando

WHERE pid = p\_pid;

END;

--/

--/deleta\_pid\_hd

PROCEDURE deleta\_pid\_hd(p\_pid NUMBER) IS

BEGIN

DELETE t\_hd

WHERE pid = p\_pid;

END;

--/

--/deleta\_pid\_video

PROCEDURE deleta\_pid\_video(p\_pid NUMBER) IS

BEGIN

DELETE t\_video

WHERE pid = p\_pid;

END;

--/

--/deleta\_pid\_impressora

PROCEDURE deleta\_pid\_impressora(p\_pid NUMBER) IS

BEGIN

DELETE t\_impressora

WHERE pid = p\_pid;

END;

--/

-------------------------------------------------------------------------------

--/inicializa

PROCEDURE inicializa(p\_processos IN NUMBER

,p\_tempo IN NUMBER) IS

BEGIN

DELETE t\_log\_pid;

DELETE t\_hd\_hist;

DELETE t\_video\_hist;

DELETE t\_impressora\_hist;

DELETE t\_criacao\_hist;

DELETE t\_hd;

DELETE t\_video;

DELETE t\_impressora;

DELETE t\_destruido;

DELETE t\_executando;

DELETE t\_apto;

DELETE t\_criacao;

IF p\_processos < 1000 AND

p\_processos > 0

THEN

v\_processos := p\_processos;

ELSE

raise\_application\_error(20001, 'Número de processos inválido! Digite um valor entre 1 e 999.');

END IF;

v\_tempo := p\_tempo;

v\_total\_ciclos\_cpu := 0;

v\_criacao\_seq := 0;

processar();

relatorio\_final(v\_total\_ciclos\_cpu);

END inicializa;

--/

--/processar

PROCEDURE processar IS

v\_processos\_criados NUMBER := 0;

v\_dispositivo NUMBER;

v\_total\_ciclos\_pid NUMBER;

v\_num\_sort NUMBER;

BEGIN

WHILE NOT (get\_sum\_ciclos\_pid = 0 AND --)

get\_sum\_ciclos\_hd IS NULL AND --

get\_sum\_ciclos\_video IS NULL AND --

get\_sum\_ciclos\_impressora IS NULL)

LOOP

IF v\_processos\_criados < v\_processos --TESTA SE EXISTE PROCESSO A SER CRIADO

THEN

IF ROUND(dbms\_random.value(1, 100)) < 11

THEN

set\_new\_processo(v\_total\_ciclos\_cpu);

dbms\_lock.sleep(v\_tempo);

v\_processos\_criados := v\_processos\_criados + 1;

END IF;

END IF;

FOR reg IN (SELECT tc.pid--MUDA O ESTADO DO CICLO PARA APTO APOS O PRIMEIRO CICLO DEPOIS DE CRIADO

,tc.estado

,tch.ciclo\_cpu

FROM t\_criacao tc

,t\_criacao\_hist tch

WHERE tc.pid = tch.pid

AND tc.estado = 'C'

AND tch.ciclo\_cpu != v\_total\_ciclos\_cpu)

LOOP

set\_fila\_apto(reg.pid);

set\_estado\_criacao(reg.pid, 'A');

set\_log\_pid(v\_total\_ciclos\_cpu, reg.pid, v\_total\_ciclos\_pid, 'A', '2º C->A');

END LOOP;

set\_fila\_apto\_esperando();--CONTADOR DE ESPERA NA FILA DE APTOS

IF NOT get\_reg\_executando() --SE NENHUM PID ESTA NO ESTADO EXECUTANDO, CHAMA O PROXIMO PID

THEN

FOR reg IN (SELECT pid

FROM t\_apto

WHERE fila = (SELECT MIN(fila)

FROM t\_apto))

LOOP

IF get\_ciclos\_criacao(reg.pid) > 0

THEN

set\_fila\_executando(reg.pid);

set\_estado\_criacao(reg.pid, 'E');

v\_total\_ciclos\_pid := 0;

set\_log\_pid(v\_total\_ciclos\_cpu, reg.pid, v\_total\_ciclos\_pid, 'E', '3º A->E');

ELSE

set\_fila\_destruido(reg.pid);

set\_estado\_criacao(reg.pid, 'D');

set\_log\_pid(v\_total\_ciclos\_cpu, reg.pid, v\_total\_ciclos\_pid, 'D', '6º E->D');

END IF;

deleta\_pid\_apto(reg.pid);

END LOOP;

ELSE --SE EXISTE PID NO ESTADO EXECUTANDO

FOR reg IN (SELECT pid

FROM t\_executando)

LOOP

set\_ciclos\_criacao(reg.pid);--DECREMENTA O CICLO NO ESTADO EXECUTANDO

IF get\_ciclos\_criacao(reg.pid) > 0

THEN

v\_total\_ciclos\_pid := v\_total\_ciclos\_pid + 1; --CONTADOR DE CICLOS DO PID DURANTE ESTADO EXECUTANDO

IF MOD(v\_total\_ciclos\_pid, 50) = 0 --TESTA SE O PID JA TEM PELO MENOS 1 CICLO E SE O CICLO ATUAL NÃO É MULTIPLO DE 50

THEN

set\_log\_pid(v\_total\_ciclos\_cpu, reg.pid, v\_total\_ciclos\_pid, 'A', '4º E->A');

set\_fila\_apto(reg.pid);

set\_estado\_criacao(reg.pid, 'A');

DELETE t\_executando;

ELSE --SORTEAR SE O PID VAI CHAMAR ALGUM RECURSO

v\_num\_sort := ROUND(dbms\_random.value(1, 100));

set\_log\_pid(v\_total\_ciclos\_cpu,reg.pid,v\_total\_ciclos\_pid,'E','NUM\_SERTEADO: ' || v\_num\_sort);--LOG DO MUMERO SORTEADO

IF v\_num\_sort = 1

THEN

set\_estado\_criacao(reg.pid, 'B');

v\_dispositivo := ROUND(dbms\_random.value(1, 3));--SORTEIO DO DISPOSITIVO A SER CHAMADO

set\_log\_pid(v\_total\_ciclos\_cpu, reg.pid, v\_total\_ciclos\_pid, 'B', '5º E->B');

DECLARE

v\_ciclos\_dispositivo NUMBER;

BEGIN

IF v\_dispositivo = 1 --VIDEO

THEN

v\_ciclos\_dispositivo := ROUND(dbms\_random.value(100, 200));

set\_fila\_video(reg.pid, v\_ciclos\_dispositivo,v\_total\_ciclos\_cpu);

ELSIF v\_dispositivo = 2 --HD

THEN

v\_ciclos\_dispositivo := ROUND(dbms\_random.value(200, 300));

set\_fila\_hd(reg.pid, v\_ciclos\_dispositivo,v\_total\_ciclos\_cpu);

ELSIF v\_dispositivo = 3 --IMPRESSORA

THEN

v\_ciclos\_dispositivo := ROUND(dbms\_random.value(500, 600));

set\_fila\_impressora(reg.pid, v\_ciclos\_dispositivo,v\_total\_ciclos\_cpu);

END IF;

END;

DELETE t\_executando --SE SORTEOU UM RECURSO REMOVE DA FILA DE EXECUTANDP

WHERE pid = reg.pid;

END IF;

END IF;

ELSE--SE CICLOS DO PID FOR IGUAL A ZERO PASSA PARA O ESTADO DESTRUIDO

set\_fila\_destruido(reg.pid);

set\_estado\_criacao(reg.pid, 'D');

set\_log\_pid(v\_total\_ciclos\_cpu, reg.pid, v\_total\_ciclos\_pid, 'D', '6º E->D');

deleta\_pid\_executando(reg.pid);

END IF;

END LOOP;

END IF;

--DECREMENTANDO CICLOS DOS PROCESSOS NO HD, VIDEO E IMPRESSORA

IF existe\_reg\_hd

THEN

set\_ciclos\_hd();

END IF;

IF existe\_reg\_video

THEN

set\_ciclos\_video();

END IF;

IF existe\_reg\_impressora

THEN

set\_ciclos\_impressora();

END IF;

--/

COMMIT;

v\_total\_ciclos\_cpu := v\_total\_ciclos\_cpu + 1; --CONTAR CICLOS GERAL DA CPU

END LOOP;

END;

--/

--/relatorio\_final

PROCEDURE relatorio\_final(p\_ciclos\_cpu IN NUMBER) IS

total\_ciclos\_pid NUMBER := 0;

media\_ciclos\_pid NUMBER := 0;

BEGIN

dbms\_output.put\_line('1) Total de processos: ' || v\_processos || CHR(10));

dbms\_output.put\_line('2) Total de ciclos CPU: ' || p\_ciclos\_cpu || CHR(10));

BEGIN

FOR reg IN (SELECT DISTINCT pid

,ciclos

FROM t\_criacao\_hist

WHERE estado = 'C')

LOOP

total\_ciclos\_pid := total\_ciclos\_pid + reg.ciclos;

END LOOP;

media\_ciclos\_pid := total\_ciclos\_pid / v\_processos;

dbms\_output.put\_line('3) Média de ciclos dos processos: ' || ROUND(media\_ciclos\_pid, 2) || CHR(10));

END;

BEGIN

dbms\_output.put\_line('4) Processos X Estado:');

FOR reg IN (SELECT estado

,COUNT(DISTINCT pid) qtd

FROM t\_log\_pid t

GROUP BY estado)

LOOP

dbms\_output.put\_line('Estado: ' || reg.estado || ' Qtd: ' || reg.qtd);

END LOOP;

END;

DECLARE

v\_qtd NUMBER;

BEGIN

SELECT ROUND(AVG(t.esperando)) qtd

INTO v\_qtd

FROM t\_criacao t;

dbms\_output.put\_line(CHR(10) || '5) Tempo médio de espera na fila de aptos: ' || v\_qtd || ' ciclos' || CHR(10));

SELECT SUM(COUNT(\*)) qtd

INTO v\_qtd

FROM t\_log\_pid t

WHERE acao = '4º E->A'

GROUP BY pid;

dbms\_output.put\_line('6) PID passou de Executando para Apto: ' || v\_qtd || ' vezes' ||CHR(10));

dbms\_output.put\_line('Quantas vezes cada recurso foi solicitado:');

SELECT COUNT(DISTINCT ciclos) qtd

INTO v\_qtd

FROM t\_hd\_hist;

dbms\_output.put\_line('HD: ' || v\_qtd);

SELECT COUNT(DISTINCT ciclos) qtd

INTO v\_qtd

FROM t\_video\_hist;

dbms\_output.put\_line('Video: ' || v\_qtd);

SELECT COUNT(DISTINCT ciclos) qtd

INTO v\_qtd

FROM t\_impressora\_hist;

dbms\_output.put\_line('Impressora: ' || v\_qtd || CHR(10));

END;

END;

--/

END pck\_sfp;