004.
Herramienta UML.
Oracle.
Matriculas
DTPL - CIMSICH

004.36B
004

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

MANUAL DEL PROGRAMADOR

"Sistema de matriculación para la modalidad presencial de la Universidad Técnica Particular de Loja"

Autores:

Omar Vinicio Hidalgo Valdiviezo

Jhison Enrique Romero Romero

Juan Carlos Sinche Quichimbo

Director:

Ing. Jorge Luis Jaramillo Pacheco

Loja – Ecuador

2004

ESQUEMA DE CONTENIDOS
MANUAL DE PROGRAMADOR

INGRESO AL SISTEMA

Verifica si el usuario existe

Verificación e ingreso al sistema de acuerdo a los usuarios

OPCIONES DEL ADMINISTRADOR

Creación de periodos

Ingreso de datos iniciales generales para el sistema

Migración de información de profesionales en formación

Migración de información de record académico

Migración de información de horarios

Fechas de entrega de notas

Datos iniciales de asignaturas

Datos iniciales de estudiantes

Creación de usuarios del sistema y asignación del nivel de operación

Creación de niveles del sistema

Cierre de periodo

Gestión curricular

Asignación de fechas de límites de pagos

Creación y modificación de tipos de pago

OPCIONES DE SECRETARIAS

Ingreso y modificación de horarios

Ingreso y modificación de paralelos

Modificación de datos del profesional en formación

Matriculación NBC

Matriculación carrera

Matriculación carrera modalidad créditos

Anulación de ficha de inscripción

Registro modificación de asistencias

Registro modificación de calificaciones

Equiparación de asignaturas

Cambio de paralelo

Cambio de carrera

OPCIONES DE INTERNET

Ingreso a la opción de Internet

OPCIONES TESORERA

Registro de pagos

FUNCIONES DE BASE

Procedimiento que inserta en la tabla record los datos de notas y asistencias

Función que devuelve el nombre del estado de la asignatura

Función que devuelve el estado de la asignatura en notas.

Función que devuelve el código de la carrera del estudiante que aprobó el NBC, determinado por la asignatura introducción a la carrera

Función que devuelve el número de horas de cruce de horario.

Función que determina el número de materias que tomo un estudiante en un periodo.

Función que determina el promedio final de una asignatura.

Función que determina los requisitos de una asignatura

Función que determina el promedio de un estudiante

Procedimiento posición ventana

LISTAS DE VALORES

Lista de valores de carrera

Lista de valores de niveles de usuario

Lista de estado del estudiante

Lista de valores colegios

Lista de estado civiles

Lista de valores de tipos de crédito

Lista de valores de datos de carreras

Lista de valores de tipos de asignatura

MANUAL DEL PROGRAMADOR

Ingreso al sistema

Verifica si el usuario existe

```
begin
        select nombre
        into :nombre
        from tgm_usuarioss
        where id_usuario  = :id_usuario;
        set_item_property('nombre', enabled, property_false);
exception when no_data_found then
        :system.message_level:=5;
        message('usuario no definido');
        :system.message_level:=10;
        raise form_trigger_failure;
        when too_many_rows then
        :system.message_level:=5;
        message('existen más de un usuario con el mismo código, revise');
        :system.message_level:=10;
        raise form_trigger_failure;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
```

Verificación e ingreso al sistema de acuerdo a los usuarios

```
declare
  usu_val number(1);
begin
  --verifica si el usuario que trata de ingresar al sistema existe y  además está activo
  select nvl(count(*),0)
  into usu_val
  from tgm_usuarioss
  where   id_usuario = :id_usuario and clave = :clave
            and estado = 'activo';
  --verifica si el usuario ha intentado ingresar al sistema más de 3 veces
  if :global.veces <= 2 then
        if usu_val = 0 then
              :system.message_level:=5;
              message('clave incorrecta');
              :system.message_level:=10;
              :global.veces:= :global.veces + 1;
              raise form_trigger_failure;
        else
              -- asigna valores iniciales para el sistema
              select id_nivel, id_carrera
              into :global.nivel, :global.carrera
```

```
                    -- nivel y        carrera
                    from tgm_usuarioss
                    where id_usuario = :id_usuario and
                          estado = 'activo';
                    -- verifica si es nivel de administrador
                    if :global.nivel = 1 then
                          call_form('c:\latesis\formas\menu_administrador.fmx' ,
                          hide,  do_replace);
                          exit_form(no_validate,full_rollback);
                    end if;
                    -- verifica si es nivel de secretarias
                    if :global.nivel = 2 then
                          call_form('c:\latesis\formas\menu_secretaria.fmx',
                          hide,  do_replace);
                          exit_form(no_validate,full_rollback);
                    end if;
                    -- verifica si es nivel de directores de àreas
                    if :global.nivel = 3 then
                          call_form('c:\latesis\formas\menu_director.fmx', hide,
                          no_replace);
                          exit_form(no_validate,full_rollback);
                    end if;
                    -- verifica si es nivel de recaudación
                    if :global.nivel = 4 then
                          call_form('c:\latesis\formas\tesor.fmx', hide,
                          no_replace);
                          exit_form(no_validate,full_rollback);
                    end if;
              end if;
  else
              :system.message_level:=5;
              message('no puede ingresar al sistema. ha provocado 3 intentos
              fallidos');
              :system.message_level:=10;
              raise form_trigger_failure;
  end if;
end;
```

OPCIÓNES DEL ADMINISTRADOR

Creación de periodos

```
declare Período varchar(6);
```

Verifica si el Período a generar existe

```
begin
        Período := :anio || :ciclo;
        select id_Período
        into Período
        from tgm_periodos
        where id_Período = Período;
        :system.message_level:=5;
```

```
        message('el período a insertar ya existe');
        :system.message_level:=10;
        raise form_trigger_failure;
exception when no_data_found then
```

Genera el código del período por año y por el ciclo

```
        :id_Período := Período;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
declare
        a_ini varchar2(4);
        a_fin varchar2(4);
begin
```

Verifica si la fecha de fin de período es menor que la fecha de inicio de este

```
if :fecha_fin < :fecha_inicio then
        :system.message_level:=5;
        message('fecha final debe ser mayor a la fecha inicial');
        :system.message_level:=10;
        raise form_trigger_failure;
else
```

Genera la descripción del período utilzada en reportes e informes

```
        a_fin := to_char(trunc(to_date(:fecha_fin)),'yyyy');
        a_ini := to_char(trunc(to_date(:fecha_inicio)),'yyyy');
        if :ciclo = '01' then
                :descripción := 'abril ' || a_ini || ' - ' || ' agosto '  || a_fin;
        else
                :descripción := 'octubre ' || a_ini || ' - ' || ' febrero '  || a_fin;
        end if;
end if;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Ingreso de datos iniciales generales para el sistema

```
declare
        Período varchar2(6);
```

Selecciónа y presenta los datos del Período correspondiente

```
begin
        select id_Período
        into Período
```

```
        from tgm_periodos
        where id_Período = :codigo_Período;
        message('el Período existe, presióne shift+avpag para agregar datos');
exception when no_data_found then
        :system.message_level:=5;
        message('Período no definido');
        :system.message_level:=10;
        raise form_trigger_failure;
        when too_many_rows then
        :system.message_level:=5;
        message('existen más de un Período con el mismo código, revise');
        :system.message_level:=10;
        raise form_trigger_failure;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Migración de información de profesiónales en formación

```
-- carga los datos del archivo texto en el bloque de datos

declare
  --variables necesarias para la captura del archivo, recorrido y posterior inserción en la bd
  iof_lee_archivo text_io.file_type;  -- nombre del archivo
  c_linebuf varchar2(10000);  -- asignación del archivo a variable para el
                recorrido
  n_longitud number(3);  -- longitud de las lineas
  n_campo number(2);              -- número del campo
  n_cont number(4);               -- contador del los caracterers del  archivo
  -- variables de la tabla
   telefono  varchar2(10);
   dirección  varchar2(60);
   cedula_militar  varchar2(10);
   sexo  varchar2(1);
   id_carrera varchar2(3);
   nombres_est  varchar2(30);
   estado  varchar2(1);
   observaciónes  varchar2(50);
   id_pensum  varchar2(5);
   apellidos_est varchar2(30);
   estado_civil  varchar2(1);
   id_estudiante  varchar2(10);
   fecha_ingreso  varchar2(10);
```

```
   tipo_sangre  varchar2(5);
   colegio  varchar2(3);
   naciónalidad  varchar2(30);
   ciudad_origen  varchar2(20);
   fecha_nac   varchar2(10);
begin
--recorrido del archivo texto
if :archivo is not null then
  c_linebuf:='hor';
  iof_lee_archivo:= text_io.fopen(:archivo,'r');
  go_block('tgm_estudiantess');
  while c_linebuf is not null loop
       begin
            text_io.get_line(iof_lee_archivo, c_linebuf);
            n_longitud := length(c_linebuf);
            n_campo := 1;
            n_cont:=1;
            while n_cont != n_longitud loop
                 if substr(c_linebuf,n_cont,1) != chr(9) then
                      if n_campo = 1 then
                           id_estudiante := id_estudiante
                           ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 2 then
                           apellidos_est := apellidos_est
                           ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 3 then
                           nombres_est :=  nombres_est ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 4 then
                           cedula_militar := cedula_militar
                           ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 5 then
                           dirección := dirección ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 6 then
                           telefono := telefono ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 7 then
                           sexo := sexo ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 8 then
                           estado := estado ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 9 then
                           id_carrera := id_carrera ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 10 then
                           id_pensum :=  id_pensum ||substr(c_linebuf,n_cont,1);
                           n_cont:=n_cont + 1;
                      elsif n_campo = 11 then
```

```
                estado_civil := estado_civil ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            elsif n_campo = 12 then
                observaciónes := observaciónes
                ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            elsif n_campo = 13 then
                fecha_ingreso := fecha_ingreso
                ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            elsif n_campo  = 14 then
                tipo_sangre := tipo_sangre ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            elsif n_campo  = 15 then
                 colegio := colegio ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            elsif n_campo = 16 then
                naciónalidad :=  naciónalidad
                ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            elsif n_campo = 17 then
                ciudad_origen := ciudad_origen
                ||substr(c_linebuf,n_cont,1);
                n_cont:=n_cont + 1;
            end if;
        else
                n_campo  := n_campo + 1;
                n_cont:= n_cont + 1;
                if  n_campo = 18 then
                        fecha_nac := fecha_nac  ||substr
                        (c_linebuf,n_cont,10);
                        n_cont:=n_longitud;
                end if;
        end if;
    end loop;
-- carga los datos de un horario al bloque da datos
:telefono      := telefono;
:dirección     := dirección;
:cedula_militar := cedula_militar;
:sexo          := sexo;
-- controla cuando el codigo de la carrera comienza con '0'
if length(id_carrera) = '1' then
        :id_carrera  := '0' ||  id_carrera;
else
        :id_carrera  := id_carrera;
end if;
:nombres_est    := nombres_est;
:estado         := estado;
:observaciónes       := observaciónes;
-- controla cuando el codigo del pensum comienza con '0'
if length(id_pensum) = '4' then
        :id_pensum  := '0' ||  id_pensum;
else
```

```
                :id_pensum   := id_pensum;
        end if;
        :apellidos_est  := apellidos_est;
        :estado_civil   := estado_civil;
        :id_estudiante  := id_estudiante;
        :fecha_ingreso  := to_date(replace(fecha_ingreso,'/',''),'ddmmyyyy');
        :tipo_sangre    := tipo_sangre;
        :colegio        := colegio;
        :naciónalidad   := naciónalidad;
        :ciudad_origen  := ciudad_origen;
        :fecha_nac      := to_date(replace(fecha_nac,'/',''),'ddmmyyyy');
        next_record;
        exception when no_data_found then
        c_linebuf := null;
        first_record;
        end;
   --reiniciar variables a null
         telefono := null ;
         dirección  := null ;
         cedula_militar := null ;
         sexo  := null ;
         id_carrera := null ;
         nombres_est := null ;
         estado  := null ;
         observaciónes   := null ;
         id_pensum  := null ;
         apellidos_est := null ;
         estado_civil := null ;
         id_estudiante := null ;
         fecha_ingreso := null ;
         tipo_sangre := null ;
         colegio  := null ;
         naciónalidad  := null ;
         ciudad_origen  := null ;
         fecha_nac := null ;
   end loop;
   text_io.fclose(iof_lee_archivo );
   set_item_property('cargar',enabled,property_false);
else
   :system.message_level:=5;
   message('no ha ingresado nombre de archivo');
   raise form_trigger_failure;
   :system.message_level:=10;
end if;
end;
```

Almacena los datos del bloque de datos en la tabla

```
begin
          :system.message_level:=5;
          commit;
          message('los datos se grabaron correctamente');
          :system.message_level:=10;
```

```
exception when others then
        :system.message_level:=5;
        message (sqlerrm);
        :system.message_level:=10;
end;
```

Migración de información de record académico

Carga los datos del archivo texto en el bloque de datos

```
declare
  --variables necesarias para la captura del archivo, recorrido y posterior inserción en la bd
  iof_lee_archivo text_io.file_type;  -- nombre del archivo
  c_linebuf varchar2(1000);  -- asignación del archivo a variable para el
                recorrido
  n_longitud number(3);  -- longitud de las lineas
  n_campo number(2);     -- número del campo
  n_cont number(2);      -- contador del los caracterers del archivo
  -- variables de la tabla
   id_estudiante  varchar2(10);
   id_asignatura varchar2(8);
   id_Período  varchar2(6);
   promedio  varchar2(6);
   h_asistidas  varchar2(6);
   h_dictadas  varchar2(6);
   creditos  varchar2(6);
   estado  varchar2(6);
begin
--recorrido del archivo texto
if :archivo is not null then
  c_linebuf:='rec';
  iof_lee_archivo:= text_io.fopen(:archivo,'r');
  go_block('tgm_record');
  while c_linebuf is not null loop
      begin
          text_io.get_line(iof_lee_archivo, c_linebuf);
          n_longitud := length(c_linebuf);
          n_campo := 1;
          n_cont:=1;
          while n_cont != n_longitud loop
              if substr(c_linebuf,n_cont,1) != chr(9) then
                  if n_campo = 1 then
                          id_estudiante := id_estudiante
                          ||substr(c_linebuf,n_cont,1);
                          n_cont:=n_cont + 1;
                  elsif n_campo = 2 then
                          id_asignatura  := id_asignatura
                          ||substr(c_linebuf,n_cont,1);
                          n_cont:=n_cont + 1;
                  elsif n_campo = 3 then
                          id_Período :=  id_Período ||substr(c_linebuf,n_cont,1);
                          n_cont:=n_cont + 1;
```

```
                elsif n_campo = 4 then
                        promedio := promedio ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 5 then
                        h_asistidas := h_asistidas ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 6 then
                        h_dictadas := h_dictadas ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 7 then
                        creditos := creditos ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                end if;
        else
                        n_campo         := n_campo + 1;
                        n_cont:= n_cont + 1;
                        if  n_campo = 8 then
                                estado :=       estado
                                ||substr(c_linebuf,n_cont,n_longitud-2);
                                n_cont:=n_longitud;
                        end if;
        end if;
    end loop;
  -- carga los datos de un horario al bloque da datos
   :id_estudiante := id_estudiante ;
   :id_asignatura := id_asignatura ;
   :id_Período  := id_Período ;
   :promedio :=  nvl(to_number(promedio),1);
   :h_asistidas := nvl(to_number(h_asistidas),1) ;
   :h_dictadas  := nvl(to_number(h_dictadas),1)  ;
   :creditos := nvl(to_number(creditos),1) ;
   :estado := estado ;
  next_record;
  exception when no_data_found then
  c_linebuf := null;
  first_record;
  end;
   --reiniciar variables a null
   id_estudiante := null ;
   id_asignatura := null ;
   id_Período  := null ;
   promedio := null ;
   h_asistidas := null ;
   h_dictadas  := null ;
   creditos := null ;
   estado :=null ;

  end loop;
  text_io.fclose(iof_lee_archivo );
  set_item_property('cargar',enabled,property_false);
else
  :system.message_level:=5;
  message('no ha ingresado nombre de archivo');
```

```
  raise form_trigger_failure;
  :system.message_level:=10;
end if;
end;
```

Almacena los datos del bloque de datos en la tabla

```
begin
        :system.message_level:=5;
        commit;
        message('los datos se grabaron correctamente');
        :system.message_level:=10;
exception when others then
        :system.message_level:=5;
        message (sqlerrm);
        :system.message_level:=10;
end;
```

Migración de información de horarios

Carga los datos del archivo texto en el bloque de datos

```
declare

  --variables necesarias para la captura del archivo, recorrido y posterior inserción en la bd
  iof_lee_archivo text_io.file_type;  -- nombre del archivo
  c_linebuf varchar2(1000);  -- asignación del archivo a variable para el recorrido
  n_longitud number(3);  -- longitud de las lineas
  n_campo number(2);              -- número del campo
  n_cont number(2);               -- contador del los caracterers del archivo

  -- variables de la tabla
   id_horario  varchar2(5);
   hora_fin  number(2);
   dia  number(1);
   id_asignatura   varchar2(8);
   hora_ini   number(2);
   id_Período   varchar2(6);
   docente   varchar2(40);
   id_paralelo   varchar2(1)       ;

begin
--recorrido del archivo texto
if :archivo is not null then
  c_linebuf:='hor';
  iof_lee_archivo:= text_io.fopen(:archivo,'r');
  go_block('tgm_horarioss');
  while c_linebuf is not null loop
      begin
          text_io.get_line(iof_lee_archivo, c_linebuf);
          n_longitud := length(c_linebuf);
          n_campo := 1;
          n_cont:=1;
```

```
        while n_cont != n_longitud loop
            if substr(c_linebuf,n_cont,1) != chr(9) then
                if n_campo = 1 then
                        id_horario := id_horario ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 2 then
                        id_asignatura  := id_asignatura
                        ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 3 then
                        id_paralelo :=  id_paralelo ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 4 then
                        docente := docente ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 5 then
                        dia := dia ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 6 then
                        hora_ini := hora_ini ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                elsif n_campo = 7 then
                        hora_fin := hora_fin ||substr(c_linebuf,n_cont,1);
                        n_cont:=n_cont + 1;
                end if;
            else
                        n_campo          := n_campo + 1;
                        n_cont:= n_cont + 1;
                        if  n_campo = 8 then
                                id_Período :=     id_Período
                                ||substr(c_linebuf,n_cont,n_longitud-2);
                                n_cont:=n_longitud;
                        end if;
            end if;
        end loop;
    -- carga los datos de un horario al bloque da datos
    :id_horario := id_horario ;
    :hora_fin  := hora_fin ;
    :dia    := dia;
    :id_asignatura := id_asignatura;
    :hora_ini := hora_ini  ;
    :id_Período := id_Período;
    :docente := docente;
    :id_paralelo := id_paralelo ;
    next_record;
    exception when no_data_found then
    c_linebuf := null;
    first_record;
    end;
--reinicia valores a null
id_horario  := null;
hora_fin := null;
dia  := null;
```

```
   id_asignatura := null;
   hora_ini  := null;
   id_Período := null;
   docente := null;
   id_paralelo  := null;
  end loop;
  text_io.fclose(iof_lee_archivo );
  set_item_property('cargar',enabled,property_false);
else
  :system.message_level:=5;
  message('no ha ingresado nombre de archivo');
  raise form_trigger_failure;
  :system.message_level:=10;
end if;
end;
```

Grabar datos en la tabla

```
begin
        :system.message_level:=5;
        commit;
        message('los datos se grabaron correctamente');
        :system.message_level:=10;
exception when others then
        :system.message_level:=5;
        message (sqlerrm);
        :system.message_level:=10;
end;
```

Fechas de entrega de notas

Verifica que las fechas ingresadas se encuentren dentro del rango de inicio y fin del Período respectivo

Fecha notas primer bimestre

```
declare fec_in date; fec_fin date;
begin
        select fecha_inicio, fecha_fin
        into fec_in, fec_fin
        from tgm_periodos
        where id_Período = :codigo_Período;
        if (:fecha_nota1 < fec_in) or (:fecha_nota1 > fec_fin) then
                message ('la fecha debe estar entre '|| fec_in || 'y' || fec_fin);
                raise form_trigger_failure;
        end if;
end;
```

Fecha notas segundo bimestre

```
declare fec_in date; fec_fin date;
begin
        select fecha_inicio, fecha_fin
        into fec_in, fec_fin
        from tgm_periodos
```

```
        where id_Período = :codigo_Período;

        if (:fecha_nota2 < fec_in) or (:fecha_nota2 > fec_fin) then
                message ('la fecha debe estar entre '|| fec_in || 'y' || fec_fin);
                raise form_trigger_failure;
        end if;
end;
```

Fecha notas supletorios

```
declare fec_in date; fec_fin date;
begin
        select fecha_inicio, fecha_fin
        into fec_in, fec_fin
        from tgm_periodos
        where id_Período = :codigo_Período;
        if (:fecha_nota3 < fec_in) or (:fecha_nota3 > fec_fin) then
                message ('la fecha debe estar entre '|| fec_in || 'y' || fec_fin);
                raise form_trigger_failure;
        end if;

end;

declare fec_in date; fec_fin date;
begin
        select fecha_inicio, fecha_fin
        into fec_in, fec_fin
        from tgm_periodos
        where id_Período = :codigo_Período;
        if (:fecha_nota4 < fec_in) or (:fecha_nota4 > fec_fin) then
                message ('la fecha debe estar entre '|| fec_in || 'y' || fec_fin);
                raise form_trigger_failure;
        end if;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Datos iniciales de asignaturas

Definir modalidades de asignatura

```
declare
        modalidad varchar2(1);
begin
        select modalidad
        into modalidad
        from tgm_modalidadesasigs
        where modalidad = :modalidad;
        :system.message_level:=5;
        message('la modalidad de la asignatura a crear ya existe');
        raise form_trigger_failure;
```

```
        :system.message_level:=10;
exception when no_data_found then
        null;
end;
```

Definir tipos de asignatura según la modalidad de créditos

```
declare
        tipo_cred varchar2(3);
begin
        select tipo_cred
        into tipo_cred
        from tgm_tipo_creditos
        where tipo_cred = :tipo_cred;
        :system.message_level:=5;
        message('el tipo de asignatura a crear ya existe');
        raise form_trigger_failure;
        :system.message_level:=10;
exception when no_data_found then
        null;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Datos iniciales de estudiantes

Definición de colegios

```
if message_code = 40350 then
        message('aún no se han definido colegios');
end if;

if error_code = 40100 then
         message('se encuentra en el primer registro...');
elsif error_code = 40508 then
        message('se ha violado una restricción. el colegio ya existe. consulte' );
end if;

declare
        colegio varchar2(3);
begin
        --verifica si el código a insertar existe
        select codigo
        into colegio
        from tgm_colegios
        where codigo = :codigo;
        :system.message_level:=5;
        message('el colegio a crear ya existe');
        raise form_trigger_failure;
        :system.message_level:=10;
exception when no_data_found then
```

```
        null;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Definición de estados de estudiantes

```
execute_query;
if message_code = 40350 then
        message('aún no se han definido estados del estudiante');
end if;
if :system.last_record = 'true' then
        :system.message_level:=5;
        message('se encuentra en el último registro');
        :system.message_level:=10;
else
        down;
end if;
if error_code = 40100 then
         message('se encuentra en el primer registro...');
elsif error_code = 40508 then
        message('se ha violado una restricción. el estado del estudiante ya existe. consulte' );
end if;

declare
        codigo varchar2(1);
begin
        select codigo
        into codigo
        from tgm_estadose
        where codigo = :codigo;
        :system.message_level:=5;
        message('el estado del estudiante a crear ya existe');
        raise form_trigger_failure;
        :system.message_level:=10;
exception when no_data_found then
        null;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Definición de estados civiles

```
execute_query;
if message_code = 40350 then
  message('aún no se han definido estados civiles');
```

```
end if;
if :system.last_record = 'true' then
        :system.message_level:=5;
        message('se encuentra en el último registro');
        :system.message_level:=10;
else
        down;
end if;
if error_code = 40100 then
         message('se encuentra en el primer registro...');
elsif error_code = 40508 then
        message('se ha violado una restricción. el estado civil ya existe. consulte' );
end if;
declare
        codigo varchar2(1);
begin
        select codigo
        into codigo
        from tgm_estadosciviles
        where codigo = :codigo;
        :system.message_level:=5;
        message('el estado civil a crear ya existe');
        raise form_trigger_failure;
        :system.message_level:=10;
exception when no_data_found then
        null;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('datos grabados correctamente');
:system.message_level:=10;
```

Seleción de carrera

Muestra el nombre de la carrera selecciónada

```
begin
  select nombre
  into :nombre_carrera
  from tgm_carrerass
  where id_carrera     = :codigo_carrera;
exception when no_data_found then
  :system.message_level:=5;
  message('carrera no definida');
  :system.message_level:=10;
  raise form_trigger_failure;
  when too_many_rows then
  :system.message_level:=5;
  message('existen más de una carrera con el mismo código, revise');
  :system.message_level:=10;
  raise form_trigger_failure;
```

```
  when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
end;
```

Asignación del código de la carrera a la variable :global.carrera

```
:global.carrera := :codigo_carrera;
message ('la carrera ' || :nombre_carrera || ' ha sido habilitada' );
```

Creación de usuarios del sistema y asignación del nivel de operación

Muestra los datos del usuario que existe y comprueba si el usuario a ingresar ya existe

```
declare
   id_usuario varchar2(8);
begin
  select id_usuario
  into id_usuario
  from tgm_usuarioss
  where id_usuario = :id_usuario;
  :system.message_level:=5;
  message('el usuario a crear ya existe');
  raise form_trigger_failure;
  :system.message_level:=10;
exception when no_data_found then
  null;
end;
```

Seleccióna el nivel al que va a pertenecer el usuario

```
begin
  select tgm_niveles.id_nivel, tgm_niveles.descripción
  into   :id_nivel,   :nombre_nivel
  from   tgm_niveles
  where tgm_niveles.id_nivel = :id_nivel;

exception when no_data_found then
  :system.message_level:=5;
  message('el nivel no existe. revise');
  :system.message_level:=10;
  raise form_trigger_failure;
when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
end;
```

Seleccióna la carrera que gestionará el usuario

```
declare
  carrera varchar2(3);
begin
```

```
  select tgm_carrerass.id_carrera
  into carrera
  from tgm_carrerass,     tgm_pensums
  where tgm_carrerass.id_carrera = :id_carrera
   and tgm_carrerass.id_carrera = tgm_pensums.id_carrera
       and tgm_pensums.fecha_final is null;
exception when no_data_found then
  :system.message_level:=5;
  message('carrera no definida');
  :system.message_level:=10;
  raise form_trigger_failure;
  when too_many_rows then
  :system.message_level:=5;
  message('existen más de una carrera con el mismo código, revise');
  :system.message_level:=10;
  raise form_trigger_failure;
  when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Creación de niveles del sistema

Muestra los datos del nivel que existe y comprueba si el nivel a ingresar ya existe

```
declare
  id_nivel varchar2(2);
begin
  select id_nivel
  into id_nivel
  from tgm_niveles
  where id_nivel = :id_nivel;
  :system.message_level:=5;
  message('el nivel a crear ya existe');
  raise form_trigger_failure;
  :system.message_level:=10;
exception when no_data_found then
  null;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Cierre de Período

Verifica si el período a cerrar existe

```
declare
  Período varchar2(6);
begin
  select id_Período
  into Período
  from tgm_periodos
  where id_Período = :codigo_Período;
exception when no_data_found then
  :system.message_level:=5;
  message('Período no definido');
  :system.message_level:=10;
  raise form_trigger_failure;
  when too_many_rows then
  :system.message_level:=5;
  message('existen más de un Período con el mismo código, revise');
  :system.message_level:=10;
  raise form_trigger_failure;
  when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
```

Opción que cierra el Período y llama al procedimiento inserta_record

```
begin
  inserta_record;
exception when others then
  :system.message_level:=5;
  message('se ha producido el error: '||sqlerrm);
  :system.message_level:=10;
end;
```

Gestión curricular

Creación de carreras

```
declare
        carrera varchar2(3);
begin
        select id_carrera
        into carrera
        from tgm_carrerass
        where id_carrera = :id_carrera;
        :system.message_level:=5;
        message('la carrera a crear ya exite');
        raise form_trigger_failure;
        :system.message_level:=10;

        exception when no_data_found then
        null;
```

```
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
clear_block ();
```

Modificación de carreras

```
begin

        if :codigo_carr = :global.carrera then
                select nombre, numero_total_creditos, titulación, id_conesup
                into :nombre,:total_creditos,:titulación,:estado
                from tgm_carrerass
                where id_carrera = :codigo_carr;
        else
                :system.message_level:=5;
                message('carrera no definida para este usuario');
                :system.message_level:=10;
                raise form_trigger_failure;
        end if;
        exception when no_data_found then
                :system.message_level:=5;
                message('carrera no definida');
                :system.message_level:=10;
                raise form_trigger_failure;

end;
```

Grabar datos en la tabla

```
if :codigo_carr = null then
        :system.message_level:= 5;
        message('ingrese código de carrera');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :nombre = null then
        :system.message_level:= 5;
        message('ingrese nombre de carrera');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :estado = null then
        :system.message_level:= 5;
        message('ingrese estado de carrera');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :total_creditos = null then
        :system.message_level:= 5;
        message('ingrese total crédito');
```

```
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :titulación = null then
        :system.message_level:= 5;
        message('ingrese titulación a obtener');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
update tgm_carrerass
set titulación = :titulación, nombre = :nombre, id_conesup = :estado,
numero_total_creditos =:total_creditos
where id_carrera = :codigo_carr;
:system.message_level:=5;
commit;
message('datos grabados correctamente');
clear_block;
:system.message_level:=10;
```

Creación modificación de planes de estudio

```
--muestra los datos de un pénsum existente
declare
        pensum varchar2(10);
        lacarrera varchar2(3);
begin
        select id_pensum , id_carrera
        into pensum , lacarrera
        from tgm_pensums
        where id_pensum  = :codigo;
        if lacarrera = :global.carrera then
                select id_carrera , fecha_vig , observación , fecha_final , num_opt , num_lib
                into :codigo_carrera, :inicio_vigencia, :descripción, :fin_vigencia , :num_opt , :num_lib
                from tgm_pensums
                where id_pensum  = :codigo;
                :global.est := 1;
                message('el plan de estudio a crear ya existe. se modificará');
                select nombre
                into :nombrecarrera
                from tgm_carrerass
                where id_carrera = :codigo_carrera;
        else
                :system.message_level:=5;
                message('el plan de estudios no está disponible para este usuario');
                raise form_trigger_failure;
                :system.message_level:=10;
        end if;
exception when no_data_found then
        :system.message_level:=5;
        message('el plan de estudio no existe. se creará');
        :global.est := 0;
        :system.message_level:=10;
        when too_many_rows then
```

```
        :system.message_level:=5;
        message('existe más de un plan con el mismo código');
        raise form_trigger_failure;
        :system.message_level:=10;
```

Grabar datos en la tabla

```
if :codigo = null then
        :system.message_level:= 5;
        message('ingrese código del pensum');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :descripción = null then
        :system.message_level:= 5;
        message('ingrese nombre del pensum');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :codigo_carrera = null then
        :system.message_level:= 5;
        message('ingrese código de carrera');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :inicio_vigencia = null then
        :system.message_level:= 5;
        message('ingrese fecha de inicio de vigencia de pensum');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :num_opt = null then
        :system.message_level:= 5;
        message('ingrese número de asignaturas optativas');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :num_lib = null then
        :system.message_level:= 5;
        message('ingrese número de asignaturas libre elección');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :global.est = 0 then
        insert into tgm_pensums values (:codigo_carrera, :inicio_vigencia, :codigo, :descripción, null, :num_opt
        , :num_lib);
else
        update tgm_pensums
        set id_carrera = :codigo_carrera, fecha_vig = :inicio_vigencia , observación = :descripción , fecha_final
        = :fin_vigencia, num_opt = :num_opt, num_lib = :num_lib
        where id_pensum = :codigo;
end if;
:system.message_level:=5;
commit;
```

```
message('datos grabados correctamente');
clear_block;
:system.message_level:=10;
```

Creación modificación de asignaturas

```
begin
        -- variabla que controla si la materia existió o no
        :global.existe := 0 ;
        -- verifica si la asignatura existe o no
select
        tgm_asignaturass.id_asignatura, horas_semana, tgm_asignaturass.nombre, numero_creditos,
        tipo_asig,nota_max,tipo_cred, modalidad, numero_alumnos, nota_min
into
        :codigo, :horas_semanales, :nombre, :creditos, :tipo_asignatura, :nota_maxima, :tipo_credito,
        :modalidad_asignatura, :estudiantes_permitidos, :nota_minima
from
        tgm_asignaturass,
        tgm_asignaruraspensums,
        tgm_pensums,
        tgm_carrerass
 where tgm_asignaturass.id_asignatura = :codigo
        and     tgm_asignaruraspensums.id_asignatura =
                tgm_asignaturass.id_asignatura
        and     tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
        and     fecha_final is null
        and     :global.carrera = tgm_carrerass.id_carrera
        and     :global.carrera  =  tgm_pensums.id_carrera
        and     tgm_asignaruraspensums.id_pensum= tgm_pensums.id_pensum;


        -- nos permite obtener los diferentes tipos de asignaturas que existen
        select descripción
        into :detalle_modalidad
        from tgm_modalidadesasigs
        where modalidad = :modalidad_asignatura;

        select descripción
        into :detalle_asignatura
        from tgm_tipo_asigs
        where tipo_asig = :tipo_asignatura;

        select descripción
        into :detalle_credito
        from tgm_tipo_creditos
        where tipo_cred = :tipo_credito;

        message ('la asignatura ya existe solo se puede modificar');
                :global.existe := 1;
        exception when no_data_found then
                :system.message_level:=5;
                message('asignatura no existe. se creará');
                :system.message_level:=10;
                when too_many_rows then
```

```
                :system.message_level:=5;
                message('existen más de una asignatura con el mismo código, revise');
                :system.message_level:=10;
                when others then
                :system.message_level:=5;
                message('se ha presentado el error: ' || sqlerrm);
                raise form_trigger_failure;
end;
begin
        select descripción
        into :detalle_asignatura
        from tgm_tipo_asigs
        where tipo_asig = :tipo_asignatura;
exception when no_data_found then
        :system.message_level:=5;
        message('tipo de asignatura no definida');
        :system.message_level:=10;
        raise form_trigger_failure;
        when too_many_rows then
        :system.message_level:=5;
        message('existe más de un tipo de asignatura con el mismo código, revise');
        :system.message_level:=10;
        raise form_trigger_failure;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
begin
        select descripción
        into :detalle_modalidad
        from tgm_modalidadesasigs
        where modalidad = :modalidad_asignatura;
        if :modalidad_asignatura = '1' then
                :nota_minima := 28;
                :nota_maxima := 40;
                set_item_property('nota_minima', enabled, property_false);
                set_item_property('nota_maxima', enabled, property_false);
        end if;
        if :modalidad_asignatura = '2' then
                :nota_minima := null;
                :nota_maxima := null;
                set_item_property('nota_minima', enabled, property_true);
                set_item_property('nota_maxima', enabled, property_true);
        end if;
        if :modalidad_asignatura = '3' then
                :nota_minima := null;
                :nota_maxima := null;
                set_item_property('nota_minima', enabled, property_true);
                set_item_property('nota_maxima', enabled, property_true);
        end if;
exception when no_data_found then
```

```
        :system.message_level:=5;
        message('modalidad de asignatura no definida');
        :system.message_level:=10;
        raise form_trigger_failure;
        when too_many_rows then
        :system.message_level:=5;
        message('existe más de una modalidad de asignatura con el mismo código, revise');
        :system.message_level:=10;
        raise form_trigger_failure;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;

begin
        select descripción
        into :detalle_credito
        from tgm_tipo_creditos
        where tipo_cred     = :tipo_credito;
exception when no_data_found then
        :system.message_level:=5;
        message('tipo de asignatura no definida');
        :system.message_level:=10;
        raise form_trigger_failure;
        when too_many_rows then
        :system.message_level:=5;
        message('existen más de un tipo de crédito con el mismo código, revise');
        :system.message_level:=10;
        raise form_trigger_failure;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
declare
        cant number(1);
        asig varchar2(6);

begin
        --verifica, antes de almacenar, que todos los datos de los campos de la asignatura no estén vacios
        if :codigo is null then
                :system.message_level:= 5;
                message('ingrese código de asignatura');
                :system.message_level:= 10;
                raise form_trigger_failure;
        end if;
        if :nombre is null then
                :system.message_level:= 5;
                message('ingrese nombre de asignatura');
                :system.message_level:= 10;
```

```
        raise form_trigger_failure;
end if;
if :tipo_asignatura is null then
        :system.message_level:= 5;
        message('ingrese tipo de asignatura');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;

if :modalidad_asignatura is null then
        :system.message_level:= 5;
        message('ingrese modalidad de la asignatura');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :tipo_credito is null then
        :system.message_level:= 5;
        message('ingrese el tipo de crédito');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :nota_minima is null then
        :system.message_level:= 5;
        message('el valor de nota mímima no se ha ingresado');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :nota_maxima is null then
        :system.message_level:= 5;
        message('el valor de nota máxima no se ha ingresado');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :estudiantes_permitidos is null then
        :system.message_level:= 5;
        message('ingrese el número de estudiantes');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :creditos is null then
        :system.message_level:= 5;
        message('ingrese el número de créditos que equivale la asignatura');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :horas_semanales is null then
        :system.message_level:= 5;
        message('ingrese el número de horas por semana');
        :system.message_level:= 10;
        raise form_trigger_failure;
end if;
if :ciclo is null then
        :system.message_level:= 5;
```

```
                message('ingrese ciclo de asignatura');
                :system.message_level:= 10;
                raise form_trigger_failure;
        end if;
        -- fin verificación de campos vacios
begin
        -- verifica si la asignatura ya existe para sola modificarla
        if :global.existe = 0 then
        --inserción de asignatura
                insert into tgm_asignaturass  values (:codigo, :horas_semanales, :nombre, :creditos,
                :tipo_asignatura, :nota_maxima, :tipo_credito, :modalidad_asignatura,
                estudiantes_permitidos, :nota_minima);
                if sql%notfound then
                        message('no pudo agregarse. error: ' || sqlerrm);
                        raise form_trigger_failure;
                end if;
                        :system.message_level:= 5;
                        message('la asignatura se agregó correctamente');
                        :system.message_level:= 10;
                else
```

Modificación de asignatura

```
                update tgm_asignaturass set horas_semana = :horas_semanales, nombre = :nombre,
                numero_creditos = :creditos, tipo_asig = :tipo_asignatura, nota_max =:nota_maxima, tipo_cred
                =:tipo_credito, modalidad =:modalidad_asignatura,
                numero_alumnos = :estudiantes_permitidos,
                nota_min = :nota_minima
                where id_asignatura = :codigo;
                if sql%notfound then
                        message('no pudo modificarse. error: ' || sqlerrm);
                        raise form_trigger_failure;
                end if;
                        :system.message_level:= 5;
                        message('la asignatura se modificó correctamente');
                        :system.message_level:= 10;
                end if;

end;
        begin
        --verifica si existe asignatura para poderla asociar a un pensum
        if :codigo is null or :codigo_carrera is null or :codigo_pensum is null then
                :system.message_level:= 5;
                message('la asignatura no se peude asociar. carrera no existe');
                :system.message_level:= 10;
                else
                --verifica si la asignatura ya está asociada a un pensum
                select count(*)
                into cant
                from tgm_asignaruraspensums
                where id_asignatura = :codigo and id_pensum = :codigo_pensum;
                if cant = 0 then
```

Asocia la asignatura a un pensum

```
                        insert into tgm_asignaruraspensums values (:ciclo, :codigo_pensum, :codigo);
```

```
                    :system.message_level:= 5;
                    message('la asignatura se asoció a la carrera');
                    system.message_level:= 10;
            else
                    :system.message_level:= 5;
                    message('la asignatura ya esta asociada a la carrera');
                    :system.message_level:= 10;
                    end if;
            end if;
    end;
```

Grabar los datos en la tabla

```
    :system.message_level:=5;
    commit;
    clear_block;
    :system.message_level:=10;
```

Ingreso modificación de alias

```
begin
        select nombre
        into :nombre_asignatura
        from tgm_asignaturass
        where id_asignatura = :codigo_asignatura;
        select id_pensum
        into :codigo_pensum
        from tgm_asignaturaspensums
        where id_asignatura = :codigo_asignatura;
exception when no_data_found then
        :system.message_level:=5;
        message('la asignatura no existe. revise');
        :system.message_level:=10;
        raise form_trigger_failure;
when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
begin
        select
                tgm_carrerass.nombre,
                tgm_asignaturass.nombre
        into
                :nombre_carrera,
                :nombre_alias
        from
                tgm_asignaturass,
                tgm_asignaturaspensums,
                tgm_pensums,
                tgm_carrerass
        where tgm_asignaturass.id_asignatura = :alias
                and tgm_asignaturaspensums.id_asignatura = tgm_asignaturass.id_asignatura
                and tgm_asignaturaspensums.id_pensum = tgm_pensums.id_pensum
```

```
                and fecha_final is null
                and tgm_carrerass.id_carrera = tgm_pensums.id_carrera
                and tgm_asignaturass.id_asignatura <> :codigo_asignatura
                and tgm_asignaturaspensums.id_pensum <>:codigo_pensum;
        exception when no_data_found then
                :system.message_level:=5;
                message('la asignatura no existe. revise');
                :system.message_level:=10;
                raise form_trigger_failure;
        when others then
                :system.message_level:=5;
                message('se ha presentado el error: '||sqlerrm);
                :system.message_level:=10;
                raise form_trigger_failure;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Ingreso modificación de requisitos

```
begin
        select nombre
        into :nombre_asignatura
        from tgm_asignaturass
        where id_asignatura = :codigo_asignatura;
        select id_pensum
        into :codigo_pensum
        from tgm_asignaturaspensums
        where id_asignatura = :codigo_asignatura;
        exception when no_data_found then
                :system.message_level:=5;
                message('la asignatura no existe. revise');
                :system.message_level:=10;
                raise form_trigger_failure;
        when others then
                :system.message_level:=5;
                message('se ha presentado el error: '||sqlerrm);
                :system.message_level:=10;
                raise form_trigger_failure;
end;
begin
        select
                tgm_carrerass.nombre,
                tgm_asignaturass.nombre
        into
                :nombre_carrera,
                :nombre_requisito
        from
                tgm_asignaturass,
                tgm_asignaturaspensums,
```

```
                tgm_pensums,
                tgm_carrerass
        where tgm_asignaturass.id_asignatura = :requisito
        and     tgm_asignaruraspensums.id_asignatura
                tgm_asignaturass.id_asignatura
        and     tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
        and     fecha_final is null
        and     tgm_carrerass.id_carrera = tgm_pensums.id_carrera
        and     tgm_asignaturass.id_asignatura <> :codigo_asignatura
        and     tgm_asignaruraspensums.id_pensum = :codigo_pensum;

exception when no_data_found then
        :system.message_level:=5;
        message('la asignatura no existe. revise');
        :system.message_level:=10;
        raise form_trigger_failure;
when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
```

Asignación de fechas de límites de pagos

Verifica que la fecha de límite de pago esté dentro del tiempo de duración del Período

```
declare
        fechainicio date;
        fechafin date;
begin
        select fecha_inicio, fecha_fin
        into fechainicio, fechafin
        from tgm_periodos
        where id_Período = :global.Período;

        if :fecha_limite > fechafin or  :fecha_limite  < fechainicio then
                :system.message_level:=5;
                message('la fecha debe estar entre' || fechainicio || ' y ' ||
                fechafin);
                raise form_trigger_failure;
            :system.message_level:=10;
        end if;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Creación y modificación de tipos de pago

Verifica si el tipo de pago a insertar ya existe

```
declare
  id_pago varchar2(2);
begin
  select id_pago
  into  id_pago
  from tgm_tiposs
  where id_pago = :id_pago;
  :system.message_level:=5;
  message('el tipo de pago a crear ya existe');
  raise form_trigger_failure;
  :system.message_level:=10;
exception when no_data_found then
  null;
end;
```

Verifica que el número de pagos este entre 1 y 9

```
if :numero_de_pagos < 1 then
      :system.message_level:=5;
      message('el número de pagos debe se entre 1 y 9');
      :system.message_level:=10;
      raise form_trigger_failure;
end if;
```

Grabar datos en la tipos de pago y actualiza la tabla fecha limites de pago

```
declare
      numeropagos number(1);
      codigopago varchar2(2);
begin
  go_block('tgm_tiposs');
  first_record;
  loop
      if :system.record_status = 'insert' then
          for i in 1..:numero_de_pagos loop
              begin
                  insert into tgm_fechalimitepagos values
                  (null,null,:id_pago, i, :codigo_Período);
              exception when others then
                  message(sqlerrm);
              end;
          end loop;
      end if;
  exit when :system.last_record = 'true';
          next_record;
  end loop;
  :system.message_level:=5;
  commit;
  message('datos grabados correctamente');
  :system.message_level:=10;
end;
```

OPCIÓNES DE LA SECRETARIA

Ingreso y modificación de horarios

Verifica y selecciónа la carrera a la que pertenece el horaio

```
begin
  select nombre
  into :nombre_carrera
  from tgm_carrerass,
                tgm_pensums
  where tgm_carrerass.id_carrera = :id_carrera
       and tgm_carrerass.id_carrera =       tgm_pensums.id_carrera
       and tgm_pensums.fecha_final is null
       and tgm_carrerass.id_carrera <> 0;
exception when no_data_found then
  :system.message_level:=5;
  message('carrera no definida');
  :system.message_level:=10;
  raise form_trigger_failure;
  when too_many_rows then
  :system.message_level:=5;
  message('existen más de una carrera con el mismo código, revise');
  :system.message_level:=10;
  raise form_trigger_failure;
  when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
end;
```

Verifica si la asignatura a la que se asigna el horario existe y pertence a la carrera de la secretaria

```
begin
select tgm_asignaturass.nombre
  into :nombre_asignatura
from
  tgm_asignaturass,
  tgm_asignaruraspensums,
  tgm_pensums,
  tgm_carrerass
 where tgm_asignaturass.id_asignatura = :id_asignatura
       and tgm_asignaturass.id_asignatura = tgm_asignaruraspensums.id_asignatura
   and tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
   and fecha_final is null
   and tgm_carrerass.id_carrera = tgm_pensums.id_carrera
  and  tgm_carrerass.id_carrera = :id_carrera;

exception when no_data_found then
```

```
  :system.message_level:=5;
  message('la asignatura no existe. revise');
  :system.message_level:=10;
  raise form_trigger_failure;
when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
end;
```

Valida la hora de inicio (7 – 20)

```
if :hora_ini < 7 or :hora_ini > 20 then
        :system.message_level:=5;
        message('la hora de inicio debe estar entre 7 y 20');
        :system.message_level:=10;
        raise form_trigger_failure;
end if;
```

Valida la hora de fin (8 – 21)

```
if (:hora_fin < 8 or :hora_fin > 21)  then
        :system.message_level:=5;
        message('la hora de fin debe estar entre 7 y 20');
        :system.message_level:=10;
        raise form_trigger_failure;
end if;
if  :hora_ini >= :hora_fin then
        :system.message_level:=5;
        message('la hora fin debe se mayor que la hora de inicio');
        :system.message_level:=10;
        raise form_trigger_failure;
end if;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Ingreso y modificación de paralelos

Verifica y selecciónа la carrera a la que pertenece el paralelo

```
begin
  select nombre
  into :nombre_carrera
  from tgm_carrerass,
                tgm_pensums
  where tgm_carrerass.id_carrera = :id_carrera
```

```
        and tgm_carrerass.id_carrera =        tgm_pensums.id_carrera
        and tgm_pensums.fecha_final is null
        and tgm_carrerass.id_carrera <> 0;
exception when no_data_found then
   :system.message_level:=5;
   message('carrera no definida');
   :system.message_level:=10;
   raise form_trigger_failure;
   when too_many_rows then
   :system.message_level:=5;
   message('existen más de una carrera con el mismo código, revise');
   :system.message_level:=10;
   raise form_trigger_failure;
   when others then
   :system.message_level:=5;
   message('se ha presentado el error: '||sqlerrm);
   :system.message_level:=10;
   raise form_trigger_failure;
end;
```

Verifica si la asignatura a la que se asigna el paralelo existe y pertence a la carrera de la secretaria

```
begin
select tgm_asignaturass.nombre
   into :nombre_asignatura
from
   tgm_asignaturass,
   tgm_asignaruraspensums,
   tgm_pensums,
   tgm_carrerass
 where tgm_asignaturass.id_asignatura = :id_asignatura
        and tgm_asignaturass.id_asignatura = tgm_asignaruraspensums.id_asignatura
   and tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
   and fecha_final is null
   and tgm_carrerass.id_carrera = tgm_pensums.id_carrera
  and  tgm_carrerass.id_carrera = :id_carrera;

exception when no_data_found then
   :system.message_level:=5;
   message('la asignatura no existe. revise');
   :system.message_level:=10;
   raise form_trigger_failure;
when others then
   :system.message_level:=5;
   message('se ha presentado el error: '||sqlerrm);
   :system.message_level:=10;
   raise form_trigger_failure;
end;
```

Asigna el horario a la asignatura y paralelo

```
begin
```

```
  select distinct id_horario
  into :id_horario
  from tgm_horarioss
  where id_asignatura = :id_asignatura
        and id_Período = :id_Período
        and id_paralelo = :id_paralelo;
exception when no_data_found then
  :system.message_level:=5;
  message('no hay horario disponible para esta asignatura. revise');
  :system.message_level:=10;
  raise form_trigger_failure;
when others then
  :system.message_level:=5;
  message('se ha presentado el error: '||sqlerrm);
  :system.message_level:=10;
  raise form_trigger_failure;
end;
```

Grabar datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

Modificación de datos del profesiónal en formación

Verifica si el profesiónal en formación existe y muestra los datos a modificarse

```
begin
      if   estudiante_existe(:codigo) > 0 then
              :system.message_level:=5;
              message('estudiante existe. se modificarán los datos');
              :system.message_level:=10;

              select fecha_ingreso, apellidos_est, nombres_est, sexo,
              tipo_sangre,
              dirección, naciónalidad, telefono, ciudad_origen,
              cedula_militar,
              estado_civil, colegio, estado, observaciónes, fecha_nac,
              id_carrera , id_pensum
              into
              :fecha_ingreso , :apellidos, :nombres , :sexo, :tipo_sangre,
              :dirección, :naciónalidad, :telefono, :ciudad, :cedula_militar,
              :estado_civil, :colegio, :estado_est, :observaciónes,
              :fecha_nac,
              :carrera , :pensum
              from
                      tgm_estudiantess
              where
                      id_estudiante = :codigo;

              select descripción
              into :descripción
```

```
                from tgm_estadosciviles
                where codigo = :estado_civil;

                select nombre
                into :nombre_colegio
                from tgm_colegios
                where codigo = :colegio;

                select descripción
                into :estado_estudiante
                from tgm_estadose
                where codigo = :estado_est;
        else
                :system.message_level:=5;
                message('estudiante no existe. no se pueden modificar los
                datos');
                :system.message_level:=10;
                raise form_trigger_failure;
        end if;
end;
```

Graba los datos en la tabla

```
declare
  error_jc exception;

begin
  update tgm_estudiantess
  set telefono = :telefono, dirección = :dirección,cedula_militar =
      :cedula_militar,sexo = :sexo,          id_carrera = :carrera,nombres_est =
      :nombres, estado = :estado_est, observaciónes =
      :observaciónes,id_pensum = :pensum, apellidos_est =:apellidos,
      estado_civil = :estado_civil,fecha_ingreso = :fecha_ingreso,
      tipo_sangre = :tipo_sangre, colegio = :colegio,naciónalidad =
      :naciónalidad, ciudad_origen = :ciudad, fecha_nac = :fecha_nac
  where id_estudiante = :codigo;

  if sql%notfound then
      raise error_jc;
  end if;

  :system.message_level:=5;
  commit;
  :system.message_level:=10;
  message('estudiante creado correctamente');

exception
  when error_jc then
          message('se ha presentado el error: '|| sqlerrm);
end;
```

Función que permite verificar si el estudiante a modificar los datos existe, devuele 0 si no existe y 1 si existe

```
function estudiante_existe(cedula char) return number is
nregistros number;
begin
        nregistros:=0;
        select count(*)
        into nregistros
        from tgm_estudiantess
        where id_estudiante = cedula;
 return nregistros;
end;
posición_ventana('Matriculación');
```

Matriculación nbc

Valores iniciales

```
--secuencia para la ficha de inscripción
select seq_ficha_inscripción.nextval
into :num_ficha
from dual;

set_item_property('crear_estudiante', visible, property_false);
set_item_property('introducción', visible, property_false);
set_item_property('talleres', visible, property_false);
:global.introducción:=0;
:global.talleres:
```

Verifica si el estudiante a matricular cumple con las normas

```
declare
  alerta integer;
  id_ficha number;
  id_elperiodo varchar2(6);
  n_materias_matriculo number;
  n_materias_record number;
  nregistros number;
  nrecords number;
begin
   -- para determinar si el estudiante ha obtenido ficha en el mismo Período
      select count(*)
      into nrecords
      from tgm_ficha_inscripciónes
      where id_estudiante = :codigo_estudiante
      and id_Período = :global.Período;

      if nrecords > 0 then
          :system.message_level:=5;
          message('el estudiante ya ha obtenido ficha de inscripción para
          el presente ciclo.');
          :system.message_level:=10;
          raise form_trigger_failure;
      end if;
      ----------
      -- para deerminar si el estudiante tiene valores pendientes de
```

```
        cancelación
    nregistros := 0;
    select count(*)
    into nregistros
    from tgm_pagoss
    where id_estudiante = :codigo_estudiante
    and status = 'p';

    if nregistros > 0 then
        :system.message_level:=5;
        message('el estudiante tienen valores pendientes de
        cancelación... revise');
        :system.message_level:=10;
        raise form_trigger_failure;
    end if;
    ----------

if estudiante_existe(:codigo_estudiante) = 0 then
    alerta:= show_alert('crea_estudiante');
    if alerta = 88 then
        set_item_property('crear_estudiante', visible, property_true);
        set_item_property('crear_estudiante', enabled, property_true);
    end if;
else
        -- para obtener el número de ficha y el período en la que aprobó
        begin
         select max(id_matricula), id_Período
            into id_ficha, id_elperiodo
            from tgm_ficha_inscripciónes
            where id_estudiante = :codigo_estudiante
            group by id_Período;
        exception when no_data_found then
            id_ficha:= 0;
            id_elperiodo:=0;
        end;

            -- para obtener el total de las materias en que se matrículó el
            último Período
            select count(*)
            into n_materias_matriculo
            from tgm_detalles
            where id_matricula = id_ficha;
            -- para obtener el total de las materias que aprobó el último
            período que se matriculó
            select count(*)
            into n_materias_record
            from tgm_record
            where id_estudiante = :codigo_estudiante
            and id_Período = id_elperiodo
            and estado = 'a';
            -- si el estudiante existe y tiene todas las materias aprobadas
            se va al else, cc reprobó y se debe matricular de nuevo
            if (n_materias_matriculo = 0 and n_materias_record = 0) or
```

```
        (n_materias_matriculo != n_materias_record) then
            select apellidos_est,     nombres_est,id_pensum,id_carrera
            into :apellidos1,:nombres1,:codigo_pensum,:codigo_carrera
            from tgm_estudiantess
            where id_estudiante = :codigo_estudiante and estado = 1;

            select nombre
            into :nombre_carrera
            from tgm_carrerass
            where id_carrera = :codigo_carrera;

            select          observación
            into:pensum
            from tgm_pensums
            where id_carrera = :codigo_carrera;

            set_item_property('crear_estudiante', visible,
            property_false);
            set_item_property('introducción', visible, property_true);
            set_item_property('introducción', enabled, property_true);
            set_item_property('talleres', visible, property_true);
            set_item_property('talleres', enabled, property_true);
    elsif (n_materias_matriculo > 0 and n_materias_record > 0) and
        (n_materias_matriculo = n_materias_record) then
        :system.message_level:=5;
        message('estudiante ya ha aprobado el nbc');
        :system.message_level:=10;
        raise form_trigger_failure;
    end if;
  end if;
end;
```

Crea al estudiante

```
declare
  error exception;
begin
  insert into tgm_estudiantess
  values  (:telefono,:dirección,:cedula_militar,:sexo,'15',:nombres,
:  estado_est,:observaciónes,'15001',:apellidos,:estado_civil,:codigo,:
  fecha_ingreso,:tipo_sangre,:colegio,:naciónalidad,:ciudad,:fecha_nac );

  if sql%notfound then
      raise error_jc;
  end if;

  :system.message_level:=5;
  commit;
  :system.message_level:=10;
  message('estudiante creado correctamente');

  hide_window('creación_estudiante');
  go_block('tgm_detalles');
```

```
  clear_block;
  :apellidos1 := :apellidos;
  :nombres1 := :nombres;

  select nombre
  into :nombre_carrera
  from tgm_carrerass
  where id_carrera     = 15;

 select  observación
  into :pensum
  from tgm_pensums
  where id_carrera = 15;

  :codigo_carrera:= 15;
  :codigo_pensum:= '15001';

  set_item_property('crear_estudiante', visible, property_false);
  set_item_property('introducción', visible, property_true);
  set_item_property('introducción', enabled, property_true);
  set_item_property('talleres', visible, property_true);
  set_item_property('talleres', enabled, property_true);

exception
  when error_jc then
          message('se ha presentado el error: '|| sqlerrm);
  end;
```

Selecióna las materia del nbc y las agrega en el bloque de datos

```
declare
  cant number(1);

  cursor materias_nbc is
      select tgm_asignaturass.id_asignatura a,
      tgm_asignaturass.nombre
      from tgm_asignaturass, tgm_asignaruraspensums, tgm_carrerass,
      tgm_pensums
      where tipo_cred = '2' and tgm_asignaturass.id_asignatura =
      tgm_asignaruraspensums.id_asignatura
          and tgm_pensums.fecha_final is null
          and  tgm_carrerass.id_carrera = tgm_pensums.id_carrera and
  tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
          and tgm_carrerass.id_carrera = 15;
          longitud number;
          contador number;
          eltaller varchar2(15);
begin
      --debe verificar si es primera vez o no
      clear_block(no_validate);
      select valor_arancel
      into :feutpl
      from tgm_aranceless
      where id_arancel = 'a21';
```

```
select valor_arancel
into :derecho_matricula
from tgm_aranceless
where id_arancel = 'a20';

:total.total_real:= :feutpl + :derecho_matricula;

set_item_property  ('tgm_detalles.veces',visible,property_false);
set_item_property  ('tgm_detalles.total',visible,property_false);
        for reg_nbc in materias_nbc loop
            :id_asignatura := reg_nbc.a;
            :nombre_asignatura := reg_nbc.nombre;
            :id_paralelo := 'a';
            :id_arancel := 'a20';
            :id_matricula := :num_ficha;
            next_record;
        end loop;

        if :global.introducción != 0 then
            :id_asignatura:= :global.introducción;
            select nombre
            into :nombre_asignatura
            from tgm_asignaturass
            where id_asignatura = :id_asignatura;
            :id_paralelo := 'a';
            :id_arancel := 'a20';
            :id_matricula := :num_ficha;
            next_record;
        end if;

        if :global.talleres != 0 then
        longitud:= length(:global.talleres);
        contador:= 1;
        eltaller:=null;
        loop
            if substr(:global.talleres,contador,1) != ',' then
            eltaller:= eltaller||substr(:global.talleres,contador,1);
            else
                    :id_asignatura:= eltaller;
                    select nombre
                    into :nombre_asignatura
                    from tgm_asignaturass
                    where id_asignatura = eltaller;
                    :id_paralelo := 'a';
                    :id_arancel := 'a20';
                    :id_matricula := :num_ficha;
                    next_record;
                    :id_asignatura:= substr(:global.talleres,contador+1,
                     longitud);
                    eltaller:= substr(:global.talleres,contador+1,
                    longitud);
                    select nombre
```

```
                    into :nombre_asignatura
                    from tgm_asignaturass
                    where id_asignatura = eltaller;
                    :id_paralelo := 'a';
                    :id_arancel := 'a20';
                    :id_matricula := :num_ficha;
                    next_record;
                    contador:=longitud;
                end if;
                exit when contador = longitud ;
                contador:= contador + 1;
            end loop;
            end if;
            first_record;
end;
```

Verifica que solo se marque una asignatura de introducción a la carrera

```
declare
  cont number;
begin
  cont:=0;
  first_record;
  loop
      if :chekea = 1 then
          cont:=cont + 1;
          :global.introducción:= :id_introducción;
      end if;
      exit when :system.last_record = 'true';
      next_record;
  end loop;

  if cont = 1 then
      hide_window('introducciónes');
      go_block('tgm_detalles');
  else
      :system.message_level:=5;
      message('debe selecciónar un sólo item');
      :system.message_level:=10;
      raise form_trigger_failure;
  end if;
end;
```

Verifica que solo se marque dos talleres a matricular

```
declare
  cont number;
  longitud number;
begin
  cont:=0;
  first_record;
  loop
      if :checa = 1 then
          cont:=cont + 1;
          :global.talleres:= :global.talleres||','||:id_taller;
```

```
        end if;
        exit when :system.last_record = 'true';
        next_record;
    end loop;

    if cont = 2 then
        longitud:= length(:global.talleres);
        :global.talleres:=substr(:global.talleres,3,longitud);
        hide_window('talleres');
        go_block('tgm_detalles');
    else
        :system.message_level:=5;
        message('debe selecciónar máximo dos item');
        :system.message_level:=10;
        raise form_trigger_failure;
    end if;
end;
```

Graba e imprime la ficha de inscripción

```
declare
    i number;
    total number;
    cuota1 number;
    cuota2 number;
begin
    begin
        insert into tgm_ficha_inscripciónes values ('p',
                                        :num_ficha,
                                        :radio_pago,
                                        trunc(sysdate),
                                        :global.Período,
                                        :codigo_estudiante);
                                        commit;
    exception when others then
        :system.message_level:=5;
        message('se ha producido el error: '||sqlerrm);
        :system.message_level:=10;
    end;
    if :radio_pago = 1 then
        begin
            insert into tgm_pagoss values (to_char(:radio_pago),
             :num_ficha,
             1,
             :total.total_real,
             'p',
             :codigo_estudiante);
        exception when others then
            :system.message_level:=5;
            message('se ha producido el error pagos: '||sqlerrm);
            :system.message_level:=10;
        end;
    elsif
            :radio_pago = 2 then
```

```
        i:=2;
        total:= :total_real - 20;
        cuota1:= total * 0.4 + 20;
        cuota2:= total * 0.3;
        begin
            insert into tgm_pagoss values (:radio_pago,
             :num_ficha,
             1,
             cuota1,
             'p',
            :codigo_estudiante);
        exception when others then
            :system.message_level:=5;
            message('se ha producido el error ..pagos credito1: '||sqlerrm);
            :system.message_level:=10;
        end;
        loop
        begin
            insert into tgm_pagoss values (:radio_pago,
             :num_ficha,
             i,
             cuota2,
             'p',
             :codigo_estudiante);
        exception when others  then
            :system.message_level:=5;
            message('se ha producido el error ..pagos credito 2,3: '||sqlerrm);
            :system.message_level:=10;
        end;
            exit when i = 3;
            i:= i + 1;
    end loop;
  end if;
  :system.message_level:= 5;
  commit;
  message ('datos grabados correctamente');
  :system.message_level:= 10;
  go_block('tgm_detalles');
  clear_block;
--imprime ficha
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\ficha_inscripción_nbc.rdf&userid=sistemagm/sistemagm@sistema&codigo_estudiante='||:codigo_estudian
te||'&ficha='|| :num_ficha || '&Período='||:global.Período );
end;
```

Matriculación carrera

Valores iniciales

```
--secuencia para la ficha de inscripción
select seq_ficha_inscripción.nextval
into :num_ficha
from dual;
```

```
:Período := :global.Período;
:fecha_emisión := sysdate;
-- porcentaje matrícula extraordinaria
:global.extranjera := 1;
-- porcentaje matrícula extranjero
:global.extraordinaria := 1;
--ficha adiciónal
:global.adiciónal := 0;
--:global.internet:=0;
  inte := :global.internet ;
  if inte = 1 then
       set_item_property ('sincadena',visible, property_false);
  else
       message ('matricula normal);
  end if;
end;
```

Verifica si el estudiante a matricular cumple con las normas

```
declare
  id_ficha number;
  id_elperiodo varchar2(6);
  n_materias_matriculo number;
  n_materias_record number;
  nregistros number;
  nrecords number;
  carrera varchar2(3);
  viene_nbc number;
begin
       :global.adiciónal := 0;
       -- para determinar si el estudiante tiene valores pendientes de cancelación
       nregistros := 0;
       select count(*)
       into nregistros
       from tgm_pagoss, tgm_ficha_inscripciónes
       where tgm_pagoss.id_estudiante = :codigo_estudiante
       and status = 'p'
       and tgm_pagoss.id_matricula =
       tgm_ficha_inscripciónes.id_matricula
       and id_Período < :Período;
       if nregistros > 0 then
            :system.message_level:=5;
            message('el estudiante tienen valores pendientes de
            cancelación... revise');
            :system.message_level:=10;
            raise form_trigger_failure;
       end if;
  if estudiante_existe(:codigo_estudiante) = 0 then
       :system.message_level:=5;
       message('estudiante no registrado, debe ser matriculado en el
       nbc.');
       :system.message_level:=10;
       raise form_trigger_failure;
  else
```

```
        viene_nbc:= estudiante_viene_nbc(:codigo_estudiante);
        if viene_nbc = '0' then
            :system.message_level:=5;
            message('estudiante ha reprobado el nbc. no puedo
            matricularlo... revise');
            :system.message_level:=10;
            raise form_trigger_failure;
        elsif viene_nbc = '1' then
            :system.message_level:=5;
            message('se ha producido un error durante la verificación de
            las cadenas');
            :system.message_level:=10;
            raise form_trigger_failure;
        end if;

        begin
            select apellidos_est, nombres_est
            into :apellidos1, :nombres1
            from tgm_estudiantess
            where id_estudiante = :codigo_estudiante;
        exception when no_data_found then
            :system.message_level:=5;
            message('estudiante no existe... revise');
            :system.message_level:=10;
            raise form_trigger_failure;
        end;
        :codigo_carrera:=que_carrera();
         select nombre ,id_pensum, observación
         into :nombre_carrera, :codigo_pensum, :pensum
      from tgm_carrerass, tgm_pensums
      where tgm_carrerass.id_carrera = :codigo_carrera
      and tgm_pensums.id_carrera = tgm_carrerass.id_carrera
      and fecha_final is null;
  end if;
end;
```

Selecciónа las materia que puede tomar el estudiante, dependiendo del estado de la caja de chequeo sin cadena y las agrega al bloque de datos

```
declare
  --cursor para seleccioanr las asignaturas del pensum menos las que  ya aprobo
  cursor materias_disponibles(lacarrera char, tipociclo char) is
 select id_asignatura
 from tgm_asignaruraspensums
 where decode(mod(ciclo,2),0,1,0) = to_number(tipociclo)
 and id_pensum = (select max(distinct id_pensum)
         from tgm_asignaruraspensums
         where substr(id_pensum,1,2) = lacarrera)
 minus

 select id_asignatura
 from tgm_record
 where id_estudiante = :codigo_estudiante
 and estado = 'a'
```

```
 minus

 select id_asignatura
  from tgm_detalles
  where id_matricula in (select id_matricula
            from tgm_ficha_inscripciónes
            where id_estudiante = :codigo_estudiante
            and id_Período = :global.Período);

  carrera varchar2(2);
  Período varchar2(2);
  cadena_ok varchar2(1);
  horasxsemana number(2);
  tipoasig varchar2(2);
  costomateria number(15,2);
  nregistros number;
  v_asignatura varchar2(10);

begin
      --debe verificar si es primera vez o no
      clear_block(no_validate);
      begin
          carrera:= que_carrera;
      exception when others then
          message('error, que carrera '||sqlerrm);
      end;

      Período:= substr(:global.Período,6,2);
      if Período = '1' then
          Período:='0';
      else
          Período:='1';
      end if;

      begin
      for reg in materias_disponibles(carrera, Período) loop
       :id_matricula:= :num_ficha;
          if :sincadena = 0 then
              begin
                  cadena_ok:= valida_cadena(reg.id_asignatura,
                  :codigo_estudiante);
              exception when others then
                  message('error, valida_cadena: '||sqlerrm);
              end;
          else
              cadena_ok:='s';
          end if;

          if cadena_ok = 's' then
              :id_asignatura:= reg.id_asignatura;
              begin
              select nombre, horas_semana, tipo_asig
```

```
into :nombre_asignatura, horasxsemana, tipoasig
from tgm_asignaturass
where id_asignatura = reg.id_asignatura;
exception when others then
    message('error..asignaturas' ||sqlerrm);
end;

nregistros:=0;
select count(*)
into nregistros
from tgm_record
where id_estudiante = :codigo_estudiante
and id_asignatura = reg.id_asignatura
and estado = 'r';

:id_paralelo:= 'a';
:veces:=1;
if nregistros = 1 then
    :veces:= 2;
elsif nregistros = 2 then
    :veces:= 3;
elsif nregistros > 2 then
    :system.message_level:=5;
    message('estudiante ha perdido más de 3 veces la
    asignatura: '||:nombre_asignatura);
    :system.message_level:=10;
    raise form_trigger_failure;
end if;
begin
if tipoasig = 1 and nregistros = 0 then
    costomateria:= costo_materia('a02');
    :tgm_detalles.total:= round(((horasxsemana *
    costomateria) + (((horasxsemana * costomateria)*
    extraordinaria)/100)) + (((horasxsemana * costomateria)
    + (((horasxsemana * costomateria)* extraordinaria)/100)) *
    extranjero(:codigo_estudiante)/100),2);
    :costo_unitario:= horasxsemana * costomateria;
elsif tipoasig = 2 and nregistros = 0 then
    costomateria:= costo_materia('a01');
    :tgm_detalles.total:= round(((horasxsemana *
    costomateria) + (((horasxsemana * costomateria)*
    extraordinaria)/100)) + (((horasxsemana * costomateria) +
     (((horasxsemana * costomateria)* extraordinaria)/100)) *
    extranjero(:codigo_estudiante)/100),2);
    :costo_unitario:= horasxsemana * costomateria;
elsif tipoasig = 1 and nregistros = 1 then
    costomateria:= costo_materia('a04');
    :tgm_detalles.total:= round(((horasxsemana *
    costomateria) + (((horasxsemana * costomateria)*
    extraordinaria)/100)) + (((horasxsemana * costomateria) +
     (((horasxsemana * costomateria)* extraordinaria)/100)) *
extranjero(:codigo_estudiante)/100),2);
    :costo_unitario:= horasxsemana * costomateria;
```

```
            elsif tipoasig = 2 and nregistros = 1 then
                costomateria:= costo_materia('a03');
                :tgm_detalles.total:= round(((horasxsemana *
                costomateria) + (((horasxsemana * costomateria)*
                extraordinaria)/100)) + (((horasxsemana * costomateria) +
                (((horasxsemana * costomateria)* extraordinaria)/100)) *
                extranjero(:codigo_estudiante)/100),2);
                :costo_unitario:= horasxsemana * costomateria;
            elsif tipoasig = 1 and nregistros = 2 then
                costomateria:= costo_materia('a06');
                :tgm_detalles.total:= round(((horasxsemana *
                costomateria) + (((horasxsemana * costomateria)*
                extraordinaria)/100)) + (((horasxsemana * costomateria) +
                 (((horasxsemana * costomateria)* extraordinaria)/100)) *
                extranjero(:codigo_estudiante)/100),2);
                :costo_unitario:= horasxsemana * costomateria;
            elsif tipoasig = 2 and nregistros = 2 then
                costomateria:= costo_materia('a05');
                :tgm_detalles.total:= round(((horasxsemana *
                costomateria) + (((horasxsemana * costomateria)*
                extraordinaria)/100)) + (((horasxsemana * costomateria) +
                (((horasxsemana * costomateria)* extraordinaria)/100)) *
                extranjero(:codigo_estudiante)/100),2);
                :costo_unitario:= horasxsemana * costomateria;
            end if;
            exception when others then
                message('error costo materias '||sqlerrm);
            end;
            next_record;
        end if;
    end loop;
    delete_record;
    exception when others then
        message('error, loop: '||sqlerrm);
    end;

  :derecho_matricula:= valor_aranceles;
  if :derecho_matricula = 0 then
        set_item_property ('credito', visible, property_false);
        :total.total_real:= :total_materias + :derecho_matricula;
        :global.adicionál :=1 ;
  else
      :total.total_real:= :total_materias + :derecho_matricula - 20;
  end if;
      first_record;
end;
```

Permite determinar si un estudiante puede tomar la asignatura selecciónada. verifica horarios y disponibilidad de paralelos

```
declare
 cursor c1(asignatura char, paralelo char) is
 select
   id_asignatura,
```

```
  dia,
  hora_ini,
  hora_fin
from
  tgm_horarioss
where id_asignatura = asignatura
and id_paralelo = paralelo
and id_Período = :Período;

cursor c2(asignatura char, paralelo char) is
select
  id_asignatura,
  dia,
  hora_ini,
  hora_fin
from
  tgm_horarioss
where id_asignatura = asignatura
and id_paralelo = paralelo
and id_Período = :Período;

horascruce number;
bandera varchar2(1);
laasignatura varchar2(8);
elparalelo varchar2(1);
total number(15,2);
nregistros number;
asignatura_ok varchar2(1);
maximo_alumnos number;
cantidad_alumnos number;

begin
  begin
      select
          numero_alumnos,
          numero_estudiantes
      into
          maximo_alumnos,
          cantidad_alumnos
      from
          tgm_asignaturass,
          tgm_paraleloss
      where
          tgm_paraleloss.id_asignatura = :id_asignatura
          and tgm_asignaturass.id_asignatura =
          tgm_paraleloss.id_asignatura
          and tgm_paraleloss.id_paralelo = :id_paralelo
          and tgm_paraleloss.id_Período = :global.Período;
  exception when no_data_found then
          :system.message_level:=5;
          message('paralelo no definido...');
          :system.message_level:=10;
          :checa:= 0;
```

```
        raise form_trigger_failure;
end;
    if maximo_alumnos < cantidad_alumnos then
        :system.message_level:=5;
        message('límite de estudiantes en ese paralelo excedió');
        :system.message_level:=10;
        :checa:= 0;
        raise form_trigger_failure;
    end if;

    select count(*)
    into nregistros
    from tgm_ficha_inscripciónes
    where id_estudiante = :codigo_estudiante
    and id_Período = :global.Período;

    if nregistros > 0 then
        asignatura_ok:= revisa_cruce_horario(:id_asignatura,
        :id_paralelo);
        if asignatura_ok = '1' then
            :system.message_level:=5;
            message('existe cruce de horario, con las asignaturas de
            fichas anteriores');
            :system.message_level:=10;
            :checa := 0;
            raise form_trigger_failure;
        end if;
    end if;
total:= :tgm_detalles.total;
if :checa = 1 then
    laasignatura:= :id_asignatura;
    elparalelo:=:id_paralelo;
    first_record;
    bandera:='0';
    horascruce:=0;
    loop
        if :checa = 1 then
            for reg in c1(laasignatura, elparalelo) loop
                for reg1 in c2(:id_asignatura, :id_paralelo) loop
                    if reg.dia = reg1.dia and reg.id_asignatura !=
                    reg1.id_asignatura then
                        horascruce:=horascruce +
                        verifica_horario(reg.hora_ini,reg.hora_fin,
                        reg1.hora_ini, reg1.hora_fin);
                        if horascruce > 2 then
                            bandera:= '1';
                        end if;
                    end if;
                end loop;
            end loop;
        end if;
    exit when :system.last_record = 'true';
        next_record;
```

```
        end loop;

        if bandera = '1' then
            :system.message_level:= 5;
            message('error, se le cruzan más de dos horas'||horascruce);
            :system.message_level:= 10;
            first_record;
            loop
                if :id_asignatura = laasignatura then
                    :checa:= 0;
                end if;
            exit when :system.last_record = 'true';
                next_record;
            end loop;
        else
            :total_materias:= :total_materias + total;

            if :global.adiciónal = 1 then
                        :total_real      := :total_materias;
            else
                        :total_real:= :total_materias +
                        :total.derecho_matricula - 20 ;
            end if;
        end if;
    elsif nvl(:checa,0) = 0 then
            :total_materias:= :total_materias - total;
            if :global.adiciónal = 1 then
                  :total_real          := :total_materias;
            else
                        :total_real:= :total_materias +
                        :total.derecho_matricula - 20;
            end if;
    end if;
    first_record;
end;
```

Graba e imprime la ficha de inscripción

```
declare
    i number;
    total number;
    cuota1 number;
    cuota2 number;
    v_contado number;
    v_credito number;
    total_aranceles number(15,2);
    forma_pago number(2);
begin
    -- para verificar si ha marcado al menos una materia
    if :total_materias = 0 then
        :system.message_level:=5;
        message('debe selecciónar al menos una materia');
        :system.message_level:=10;
        raise form_trigger_failure;
```

```
end if;

-- para verificar si es a crédito o a contado
if :credito = 1 then
    forma_pago:= 2;
else
    forma_pago:= 1;
end if;
-- inserta la ficha de inscripción
begin
    insert into tgm_ficha_inscripciónes values ('p',
                                    :num_ficha,
                                    forma_pago,
                                    trunc(sysdate),
                                    :global.Período,
                                    :codigo_estudiante);
exception when others then
    :system.message_level:=5;
    message('se ha producido el error: '||sqlerrm);
    :system.message_level:=10;
end;
-- para otener el descuento al contado y el recargo a crédito
begin
    select
        nvl(desc_contado,0) desc_contado,
        nvl(rec_credito,0) rec_credito
    into
        v_contado,
        v_credito
    from tgm_fechas
    where id_Período = :Período;
exception when others then
    message('se ha producido el error: '||sqlerrm);
end;

if forma_pago = 1 then
    begin
        -- para insertar los pagos
        insert into tgm_pagoss values (to_char(forma_pago),
         :num_ficha,
         1,
         :total.total_real,
         'p',
         :codigo_estudiante);
    exception when others then
        :system.message_level:=5;
        message('se ha producido el error pagos: '||sqlerrm);
        :system.message_level:=10;
    end;
    total_aranceles:= valor_aranceles;
    -- si ya ha obtenido anteriormente una ficha de inscripción
    if total_aranceles = 0 then
        begin
```

```
            insert into tgm_aux_totales values (:num_ficha,
            total_aranceles,
            :total_materias,
            0,
            0,
            :total_real);
        exception when others then
            message('insert tgm_aux_totales. se ha producido el error: '||sqlerrm);
        end;
    else
        begin
            insert into tgm_aux_totales values (:num_ficha,
            total_aranceles,
            :total_materias,
            v_contado,
            0,
            :total_real);
        exception when others then
            message('insert tgm_aux_totales. se ha producido el error: '||sqlerrm);
        end;
    end if;
elsif forma_pago = 2 then
        i:=2;
        total:= :total_real - 20;
        cuota1:= total * 0.4 + 20;
        cuota2:= total * 0.3;
        begin
            insert into tgm_pagoss values (forma_pago,
             :num_ficha,
                1,
                cuota1,
                'p',
                :codigo_estudiante);
        exception when others then
            :system.message_level:=5;
            message('se ha producido el error ..pagos credito1: '||sqlerrm);
            :system.message_level:=10;
        end;
    loop
            begin
            insert into tgm_pagoss values (forma_pago,
                :num_ficha,
                i,
                cuota2,
                'p',
                :codigo_estudiante);
        exception when others  then
            :system.message_level:=5;
            message('se ha producido el error ..pagos credito 2,3:
            '||sqlerrm);
            :system.message_level:=10;
        end;
            exit when i = 3;
```

```
            i:= i + 1;
    end loop;
    --inserta en la tgm_aux_totales
    begin
    insert into tgm_aux_totales values (:num_ficha,
            :derecho_matricula,
            :total_materias,
            0,
            v_credito,
            :total_real);
    exception when others then
        message('insert tgm_aux_totales. se ha producido el error:
        '||sqlerrm);
    end;
end if;

go_block('tgm_detalles');
first_record;
loop
    if nvl(:checa,0) = 0 then
        delete_record;
    end if;
exit when :system.last_record = 'true' ;
    if nvl(:checa,0) = 0 then
        delete_record;
    end if;
        next_record;
end loop;

if nvl(:checa,0) = 0 then
        delete_record;
end if;

go_block('tgm_detalles');
first_record;
loop
    --message(:id_asignatura);
    insert into tgm_detalles_aux values (:num_ficha,
            :id_asignatura,
            :id_paralelo,
            :veces,
            :costo_unitario,
            :tgm_detalles.total);
    update tgm_paraleloss
    set numero_estudiantes = numero_estudiantes + 1
    where id_asignatura = :id_asignatura
    and id_paralelo = :id_paralelo
    and id_Período = :global.Período;
exit when :system.last_record = 'true' ;
    next_record;
end loop;
:system.message_level:= 5;
commit;
```

```
  message ('datos grabados correctamente');
  :system.message_level:= 10;
  message (:block87.codigo_estudiante);
  web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\ficha_inscripción_carreras.rdf&userid=sistemagm/sistemagm@sistema&codigo_estudiante='||:block87.cod
igo_estudiante||'&ficha='|| :num_ficha || '&Período='||:global.Período );

  set_item_property ('credito', visible, property_true);
  set_item_property ('credito', enabled, property_true);
  go_block('tgm_detalles');
  clear_block;

  go_block('block87');
  clear_block;

  select seq_ficha_inscripción.nextval
  into :num_ficha
  from dual;

  :Período:= :global.Período;
  :fecha_emisión:= trunc(sysdate);
  :total.total_materias:= 0;
  :total.total_real:= :derecho_matricula - 20;
end;
```

Matriculación carrera modalidad créditos

Permite determinar si un estudiante puede tomar la asignatura selecciónada. verifica horarios, disponibilidad de paralelos y créditos que pueda tomar

```
declare
  cursor c1(asignatura char, paralelo char) is
select
  id_asignatura,
  dia,
  hora_ini,
  hora_fin
from
  tgm_horarioss
where id_asignatura = asignatura
and id_paralelo = paralelo
and id_Período = :Período;

cursor c2(asignatura char, paralelo char) is
select
  id_asignatura,
  dia,
  hora_ini,
  hora_fin
from
  tgm_horarioss
where id_asignatura = asignatura
and id_paralelo = paralelo
```

```
and id_Período = :Período;

horascruce number;
bandera varchar2(1);
laasignatura varchar2(8);
elparalelo varchar2(1);
total number(15,2);
begin
  if :checa = 1 then
      :total_creditos:= :total_creditos + :num_creditos;
      laasignatura:= :id_asignatura;
      elparalelo:=:id_paralelo;
      total:= :tgm_detalles.total;
      first_record;
      bandera:='0';
      horascruce:=0;
      loop
          if :checa = 1 then
              for reg in c1(laasignatura, elparalelo) loop
                  for reg1 in c2(:id_asignatura, :id_paralelo) loop
                      if reg.dia = reg1.dia and reg.id_asignatura !=
                      reg1.id_asignatura then
                          horascruce:=horascruce +
                          verifica_horario(reg.hora_ini,
                          reg.hora_fin, reg1.hora_ini, reg1.hora_fin);
                          if horascruce > 2 then
                              bandera:= '1';
                          end if;
                      end if;
                  end loop;
              end loop;
          end if;
      exit when :system.last_record = 'true';
          next_record;
      end loop;

      if bandera = '1' then
          :system.message_level:= 5;
          message('error, se le cruzan más de dos horas'||horascruce);
          :system.message_level:= 10;
          first_record;
          loop
              if :id_asignatura = laasignatura then
                  :checa:= 0;
              end if;
          exit when :system.last_record = 'true';
              next_record;
          end loop;
      else
          :total_materias:= :total_materias + total;
      end if;
  else
      :total_materias:= :total_materias - :tgm_detalles.total;
```

```
        :total_creditos:= :total_creditos - :num_creditos;
    end if;
    first_record;
end;
```

Verifica que solo se marque las materias optativas permitidas a su carrera

```
declare
    cont number;
begin
    cont:=0;
    go_block('optativas');
    first_record;
    loop
        if :chequea = 1 then
            cont:= cont + 1;
            :global.optativas:= :global.optativas||','||:ocodigo_asignatura;
        end if;
        exit when :system.last_record = 'true';
        next_record;
    end loop;
    if cont > 2 then
        :system.message_level:=5;
        message('solamente puede seleccionar hasta 2 asignaturas');
        :system.message_level:=10;
        raise form_trigger_failure;
    else
        :global.optativas:=substr(:global.optativas,3,  length(:global.optativas));
        hide_window('optativas');
        go_block('tgm_detalles');
    end if;
end;
```

Verifica que marque las materias de libre elección

```
declare
    cont number;
begin
    cont:=0;
    go_block('libres');
    first_record;
    loop
        if :chequealo = 1 then
            cont:= cont + 1;
            :global.libres:= :global.libres||','||:lcodigo_asignatura;
        end if;
        exit when :system.last_record = 'true';
        next_record;
    end loop;
    if cont > 3 then
        :system.message_level:=5;
        message('solamente puede seleccionar hasta 3 asignaturas');
        :system.message_level:=10;
        raise form_trigger_failure;
    else
```

```
      :global.libres:=substr(:global.libres,3, length(:global.libres));
  hide_window('libres');
      go_block('tgm_detalles');
  end if;
end;
```

Graba e imprime la ficha de inscripción

```
declare
  i number;
  total number;
  cuota1 number;
  cuota2 number;
  v_contado number;
  v_credito number;
  total_aranceles number(15,2);
  forma_pago number(2);
begin
  -- para verificar si ha marcado al menos una materia
  if :total_materias = 0 then
      :system.message_level:=5;
      message('debe seleccionar al menos una materia');
      :system.message_level:=10;
      raise form_trigger_failure;
  end if;

  -- para verificar si es a crédito o a contado
  if :credito = 1 then
      forma_pago:= 2;
  else
      forma_pago:= 1;
  end if;
  -- inserta la ficha de inscripción
  begin
      insert into tgm_ficha_inscripciónes values ('p',
                                        :num_ficha,
                                        forma_pago,
                                        trunc(sysdate),
                                        :global.Período,
                                        :codigo_estudiante);
  exception when others then
      :system.message_level:=5;
      message('se ha producido el error: '||sqlerrm);
      :system.message_level:=10;
  end;
  -- para otener el descuento al contado y el recargo a crédito
  begin
      select
          nvl(desc_contado,0) desc_contado,
          nvl(rec_credito,0) rec_credito
      into
          v_contado,
          v_credito
      from tgm_fechas
      where id_Período = :Período;
```

```
exception when others then
    message('se ha producido el error: '||sqlerrm);
end;

if forma_pago = 1 then
    begin
        -- para insertar los pagos
        insert into tgm_pagoss values (to_char(forma_pago),
         :num_ficha,
         1,
         :total.total_real,
         'p',
         :codigo_estudiante);
    exception when others then
        :system.message_level:=5;
        message('se ha producido el error pagos: '||sqlerrm);
        :system.message_level:=10;
    end;
    total_aranceles:= valor_aranceles;
    -- si ya ha obtenido anteriormente una ficha de inscripción
    if total_aranceles = 0 then
        begin
            insert into tgm_aux_totales values (:num_ficha,
            total_aranceles,
            :total_materias,
            0,
            0,
            :total_real);
        exception when others then
            message('insert tgm_aux_totales. se ha producido el error: '||sqlerrm);
        end;
    else
        begin
            insert into tgm_aux_totales values (:num_ficha,
            total_aranceles,
            :total_materias,
            v_contado,
            0,
            :total_real);
        exception when others then
            message('insert tgm_aux_totales. se ha producido el error: '||sqlerrm);
        end;
    end if;
elsif forma_pago = 2 then
        i:=2;
        total:= :total_real - 20;
        cuota1:= total * 0.4 + 20;
        cuota2:= total * 0.3;
        begin
            insert into tgm_pagoss values (forma_pago,
             :num_ficha,
                 1,
                 cuota1,
```

```
                    'p',
                    :codigo_estudiante);
            exception when others then
                :system.message_level:=5;
                message('se ha producido el error ..pagos credito1: '||sqlerrm);
                :system.message_level:=10;
            end;
        loop
                begin
                insert into tgm_pagoss values (forma_pago,
                    :num_ficha,
                    i,
                    cuota2,
                    'p',
                    :codigo_estudiante);
            exception when others  then
                :system.message_level:=5;
                message('se ha producido el error ..pagos credito 2,3:
                '||sqlerrm);
                :system.message_level:=10;
            end;
                exit when i = 3;
                i:= i + 1;
        end loop;
        --inserta en la tgm_aux_totales
        begin
        insert into tgm_aux_totales values (:num_ficha,
                :derecho_matricula,
                :total_materias,
                0,
                v_credito,
                :total_real);
        exception when others then
            message('insert tgm_aux_totales. se ha producido el error:
            '||sqlerrm);
        end;
end if;

go_block('tgm_detalles');
first_record;
loop
    if nvl(:checa,0) = 0 then
        delete_record;
    end if;
exit when :system.last_record = 'true' ;
    if nvl(:checa,0) = 0 then
        delete_record;
    end if;
        next_record;
end loop;

if nvl(:checa,0) = 0 then
        delete_record;
```

```
  end if;

  go_block('tgm_detalles');
  first_record;
  loop
      --message(:id_asignatura);
      insert into tgm_detalles_aux values (:num_ficha,
                  :id_asignatura,
                  :id_paralelo,
                  :veces,
                  :costo_unitario,
                  :tgm_detalles.total);
      update tgm_paraleloss
      set numero_estudiantes = numero_estudiantes + 1
      where id_asignatura = :id_asignatura
      and id_paralelo = :id_paralelo
      and id_Período = :global.Período;
  exit when :system.last_record = 'true' ;
      next_record;
  end loop;
  :system.message_level:= 5;
  commit;
  message ('datos grabados correctamente');
  :system.message_level:= 10;
  message (:block87.codigo_estudiante);

  web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\ficha_inscripción_carreras.rdf&userid=sistemagm/sistemagm@sistema&codigo_estudiante='||:block87.cod
igo_estudiante||'&ficha='|| :num_ficha || '&Período='||:global.Período );

  set_item_property ('credito', visible, property_true);
  set_item_property ('credito', enabled, property_true);
  go_block('tgm_detalles');
  clear_block;

  go_block('block87');
  clear_block;

  select seq_ficha_inscripción.nextval
  into :num_ficha
  from dual;

  :Período:= :global.Período;
  :fecha_emisión:= trunc(sysdate);
  :total.total_materias:= 0;
  :total.total_real:= :derecho_matricula - 20;
end;
```

Anulación de ficha de inscripción

Verifica si la ficha a anular existe y que este pendiente

```
begin
```

```
  select  fecha_emisión ,id_Período ,id_estudiante
  into :fecha_emisión, :Período , :codigo_estudiante
  from tgm_ficha_inscripciónes
  where id_matricula = :num_ficha and estado_cancelación = 'p' and id_Período = :global.Período;
  select nombres_est, apellidos_est
  into :nombres1 , :apellidos1
  from tgm_estudiantess
  where id_estudiante = :codigo_estudiante;
exception when no_data_found then
  :system.message_level:=5;
  message('la ficha a anular no existe, o ya está cancelada');
  raise form_trigger_failure;
  :system.message_level:=10;
end;
```

Graba los datos de la ficha anuluda

```
declare
      res integer;
begin
  res:= show_alert('pregunta');
  if res = '89' then
      message('ficha no anulada'); --no
  else
      update tgm_ficha_inscripciónes
      set estado_cancelación = 'a'
      where id_matricula = :num_ficha;
      update tgm_pagoss
      set status = 'a'
      where id_matricula = :num_ficha;
      :system.message_level:=5;
      commit;
      message('ficha anulada correctamente');
      :system.message_level:=10;
  end if;
end;
```

Función que devuleve el valor del arancel, se envía como parametro el codigo del arancel

```
function costo_materia(arancel char) return number is
valorarancel number(15,2);
begin
       select valor_arancel
      into valorarancel
      from tgm_aranceless
      where id_arancel = arancel;

      return valorarancel;
end;
```

Función que devuelve el porcentaje de recargo si el estudiante es extranjero, se envía como parametro el código del estudiante

```
function extranjero(cedula char)  return number is
nación varchar2(25);
```

```
retorna number(15,2);
begin
  select naciónalidad
  into nación
  from tgm_estudiantess
  where id_estudiante = cedula;
  if nación = 'ecuatoriana' then
       retorna:=0;
  else
       select    recargo_extranjero
       into retorna
       from      tgm_fechas
       where id_Período = :Período;
  end if;
       return retorna;
end;
```

Función que devuelve el porcentaje de recargo por matrícula extraordinaria

```
function extraordinaria return number is
lafecha date;
valor_retorna number(15,2);
begin
       select    fec_mat_or, recargo_ext
       into lafecha,valor_retorna
       from      tgm_fechas
       where id_Período = :Período;
       if trunc(sysdate) <= trunc(lafecha) then
            valor_retorna:= 0;
       end if;
       return valor_retorna;
end;
```

Función que devuelve la carrera a la que se tiene que matricular el estudiante cuando ha aprobado el nbc

```
function que_carrera return char is
asignatura varchar2(8);
lacarrera varchar2(2);
begin
  lacarrera:='xx';
  select id_asignatura
  into asignatura
  from tgm_record
  where id_estudiante = :codigo_estudiante
  and id_asignatura between '150011' and '150030'
  and estado = 'a';
  return substr(asignatura,5,2);
exception when no_data_found then ---para estudiantes de otras u's
  select
       id_carrera
  into
   lacarrera
  from
       tgm_estudiantess
```

```
  where id_estudiante = :codigo_estudiante;
  return lacarrera;
end;
```

Función que devuelve un estado que determina si existe cruce de horarios, se envía como parametro el código de la asignatura y el paralelo

```
function revisa_cruce_horario(laasignatura char, elparalelo char) return char is
  cursor c1 is
  select id_asignatura
  from tgm_detalles
  where id_matricula in (select id_matricula
  from tgm_ficha_inscripciónes
  where id_estudiante = :codigo_estudiante
  and id_Período = :global.Período);
cursor c2(asignatura char, paralelo char) is
select
  id_asignatura,
  dia,
  hora_ini,
  hora_fin
from
  tgm_horarioss
where id_asignatura = asignatura
and id_paralelo = paralelo
and id_Período = :Período;
cursor c3(asignatura char, paralelo char) is
select
  id_asignatura,
  dia,
  hora_ini,
  hora_fin
from
  tgm_horarioss
where id_asignatura = asignatura
and id_paralelo = paralelo
and id_Período = :Período;
labandera varchar2(1);
horascruce number;
begin
labandera:='0';
for reg in c1 loop
  horascruce:=0;
  for reg1 in c2 (reg.id_asignatura, elparalelo) loop
      for reg2 in c3(laasignatura, elparalelo) loop
          if reg1.dia = reg2.dia then
              horascruce:=horascruce +
              verifica_horario(reg1.hora_ini,reg1.hora_fin, reg2.hora_ini,
              reg2.hora_fin);
              if horascruce > 2 then
                  labandera:= '1';
              end if;
          end if;
      end loop;
```

```
  end loop;
 end loop;

 if labandera = '1' then
   return '1';
 else
   return '0';
 end if;
end;
```

Función que devuelve el valor del derecho de matrtícula, verificando si es una ficha adiciónal

```
function valor_aranceles return number is
nregistros number;
retorna number;
begin
   nregistros:=0;
   select count(*)
   into nregistros
   from tgm_ficha_inscripciónes
   where id_estudiante = :codigo_estudiante
   and id_Período =:Período;

   retorna:=0;
   if nregistros = 0 then
        begin
        select valor_arancel
             into retorna
             from tgm_aranceless
             where id_carrera = 2
             and id_arancel = 'a07';
        exception when others then
             message('error, aranceles '||sqlerrm);
        end;
   end if;
   return retorna;
end;
```

Registro modificación de asistencias

```
declare
        minimo number;
        totald number;

begin
```

Recupera las horas dictadas ingresadas anteriormente y los datos del paralelo

```
        select distinct docente
        into :docente
        from tgm_horarioss
        where   id_asignatura = :codigo_asignatura and
                        id_paralelo = :asignatura.paralelo and
                        id_Período = :asignatura.Período;
```

```
        begin
                select
                        hd_mes1, hd_mes2, hd_mes3, hd_mes4, hd_mes5
                into
                        :vha_mes1, :vha_mes2, :vha_mes3, :vha_mes4, :vha_mes5
                from
                        tgm_valoress
                where
                        id_Período = :Período and
                        id_asignatura = :codigo_asignatura and
                        paralelo  = :asignatura.paralelo;
                        totald := nvl(to_number(:vha_mes1),0) +
                        nvl(to_number(:vha_mes2),0) + nvl(to_number(:vha_mes3),0) +
                        nvl(to_number(:vha_mes4),0) + nvl(to_number(:vha_mes5),0)  ;
                        :totald := totald;
                        minimo := round(nvl((to_number(:totald)* 0.7),0),0);
                        :totalm := minimo;
                        :global.modifica := 1;
                exception when no_data_found then
                        :system.message_level:=5;
                        message('los valores asistencias no están ingresados.');
                        :global.modifica := 0;
                        :system.message_level:=10;
        end ;
        exception when no_data_found then
                :system.message_level:=5;
                message('la asignatura no existe. revise');
                :system.message_level:=10;
                raise form_trigger_failure;
end ;
```

Ingreso de los valores de asistencias. cálculo de sumatoria y mínimo

```
declare
        minimo number;
        totald number;
begin
        :vha_mes1 := nvl(:vha_mes1, '0');
        totald := nvl(to_number(:vha_mes1),0) + nvl(to_number(:vha_mes2),0) + nvl(to_number(:vha_mes3),0)
        + nvl(to_number(:vha_mes4),0) + nvl(to_number(:vha_mes5),0)  ;
        :totald := totald;
        minimo := round(nvl((to_number(:totald)* 0.7),0),0);
        :totalm := minimo;
exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totald number;
begin
        :vha_mes2 := nvl(:vha_mes2, '0');
        totald := nvl(to_number(:vha_mes1),0) + nvl(to_number(:vha_mes2),0) + nvl(to_number(:vha_mes3),0)
        + nvl(to_number(:vha_mes4),0) + nvl(to_number(:vha_mes5),0)  ;
```

```
        :totald := totald;
        minimo := round(nvl((to_number(:totald)* 0.7),0),0);
        :totalm := minimo;
exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totald number;
begin
        :vha_mes3 := nvl(:vha_mes3, '0');
        totald := nvl(to_number(:vha_mes1),0) + nvl(to_number(:vha_mes2),0) + nvl(to_number(:vha_mes3),0)
        + nvl(to_number(:vha_mes4),0) + nvl(to_number(:vha_mes5),0)  ;
        :totald := totald;
        minimo := round(nvl((to_number(:totald)* 0.7),0),0);
        :totalm := minimo;
exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totald number;
begin
        :vha_mes4 := nvl(:vha_mes4, '0');
        totald := nvl(to_number(:vha_mes1),0) + nvl(to_number(:vha_mes2),0) + nvl(to_number(:vha_mes3),0)
        + nvl(to_number(:vha_mes4),0) + nvl(to_number(:vha_mes5),0)  ;
        :totald := totald;
        minimo := round(nvl((to_number(:totald)* 0.7),0),0);
        :totalm := minimo;
exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totald number;
begin
        :vha_mes5 := nvl(:vha_mes5, '0');
        totald := nvl(to_number(:vha_mes1),0) + nvl(to_number(:vha_mes2),0) + nvl(to_number(:vha_mes3),0)
        + nvl(to_number(:vha_mes4),0) + nvl(to_number(:vha_mes5),0)  ;
        :totald := totald;
        minimo := round(nvl((to_number(:totald)* 0.7),0),0);
        :totalm := minimo;
exception when others then
        message (sqlerrm);
end;

--      selecciona los apellidos y nombres de los estudiantes pertenecientes al paralelo
select apellidos_est || ' ' ||  nombres_est
into :nombre
from tgm_estudiantess
```

```
where id_estudiante = :id_estudiante;
```

Recuperación e ingreso de los valores correspondientes a las horas asistidas por el estudiante

```
declare
        minimo number;
        totala number;
begin

        :ha_mes1 := nvl(:ha_mes1, '0');
        totala := nvl(to_number(:ha_mes1),0) + nvl(to_number(:ha_mes2),0) + nvl(to_number(:ha_mes3),0) +
        nvl(to_number(:ha_mes4),0) + nvl(to_number(:ha_mes5),0)  ;
        :totala := totala;
        :est := porcentajeasistencias(nvl(to_number(:totald),0), totala);

exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totala number;
begin
        :ha_mes2 := nvl(:ha_mes2, '0');
        totala := nvl(to_number(:ha_mes1),0) + nvl(to_number(:ha_mes2),0) + nvl(to_number(:ha_mes3),0) +
        nvl(to_number(:ha_mes4),0) + nvl(to_number(:ha_mes5),0)  ;
        :totala := totala;
        :est := porcentajeasistencias(nvl(to_number(:totald),0), totala);
exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totala number;
begin
        :ha_mes3 := nvl(:ha_mes3, '0');
         totala := nvl(to_number(:ha_mes1),0) + nvl(to_number(:ha_mes2),0) + nvl(to_number(:ha_mes3),0) +
        nvl(to_number(:ha_mes4),0) + nvl(to_number(:ha_mes5),0)  ;
        :totala := totala;
        :est := porcentajeasistencias(nvl(to_number(:totald),0), totala);

exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totala number;
begin
        :ha_mes4 := nvl(:ha_mes4, '0');
        totala := nvl(to_number(:ha_mes1),0) + nvl(to_number(:ha_mes2),0) + nvl(to_number(:ha_mes3),0) +
        nvl(to_number(:ha_mes4),0) + nvl(to_number(:ha_mes5),0)  ;
```

```
        :totala := totala;
        :est := porcentajeasistencias(nvl(to_number(:totald),0), totala);

exception when others then
        message (sqlerrm);
end;

declare
        minimo number;
        totala number;
begin
        :ha_mes5 := nvl(:ha_mes5, '0');
        totala := nvl(to_number(:ha_mes1),0) + nvl(to_number(:ha_mes2),0) + nvl(to_number(:ha_mes3),0) +
        nvl(to_number(:ha_mes4),0) + nvl(to_number(:ha_mes5),0)  ;
        :totala := totala;
        :est := porcentajeasistencias(nvl(to_number(:totald),0), totala);

exception when others then
        message (sqlerrm);
end;
```

Grabar datos en la tabla

```
declare
        es varchar2(1);
        estadonotas varchar2(1);
        estadorecord varchar2(1);
        creditosest number;
        dictadas number;

begin
        go_block('tgm_asistenciass');
        first_record;
        loop
                begin
                        update tgm_asistenciass
                        set ha_mes1 = :ha_mes1, ha_mes2 = :ha_mes2 , ha_mes3 =
                        :ha_mes3 , ha_mes4 = :ha_mes4 , ha_mes5 = :ha_mes5, estado = :est
                        where id_estudiante = :id_estudiante and id_Período =
                        :id_Período and paralelo = :tgm_asistenciass.paralelo and id_asignatura =
                        :id_asignatura;
                        exception when others then
                                message(sqlerrm);
                end;
                exit when :system.last_record = 'true';
                        next_record;
        end loop;


        --      verifica si los valores de asistencias estan llenos
        if :global.modifica = 1 then
        --      solo modifica los valores de asistencias
                update tgm_valoress
                set hd_mes1 = :vha_mes1, hd_mes2 = :vha_mes2 , hd_mes3 =
```

```
            :vha_mes3 , hd_mes4 = :vha_mes4 , hd_mes5 = :vha_mes5
            where id_Período = :id_Período and  paralelo = :tgm_asistenciass.paralelo and  id_asignatura =
            :id_asignatura;
      else
            insert into tgm_valoress
            values  (:vha_mes5  ,  :vha_mes3  ,  :vha_mes4  ,  :vha_mes1  ,  :vha_mes2  ,  :Período  ,
            :codigo_asignatura , :asignatura.paralelo);
      end if;

      --    actualiza los datos en la tabla tgm_record cuando esta en otro Período

       if :global.nuevoperiodo = 1 then
            go_block('tgm_asistenciass');
            first_record;
            loop
                  begin
                        select estado
                        into estadonotas
                        from tgm_notass
                        where id_estudiante = :id_estudiante and
                        id_Período  =  :id_Período  and   paralelo  =  :tgm_asistenciass.paralelo  and
                        id_asignatura = :id_asignatura;
                        if :est = 'a' and estadonotas = 'a' then
                              estadorecord = 'a' ;
                        else
                              estadorecord := 'r' ;
                        end if;

--    obtiene los creditos de la asignatura
      creditosest := obtienecreditos (:id_asignatura, estadorecord);
--    actualiza el record del estudiante
      update tgm_record
      set h_asistidas = :totala , h_dictadas = :totald , estado = estadorecord, creditos = creditosest
      where id_estudiante = :id_estudiante and  id_Período = :id_Período
      and id_asignatura = :id_asignatura;
      exception when others then
            message(sqlerrm);
      end;
            exit when :system.last_record = 'true';
                  next_record;
            end loop;

      end if;
            -- fin inserta tabla record

      :system.message_level:=5;
      commit;
      message('datos grabados correctamente');
      :system.message_level:=10;
end;
```

Registro modificación de calificaciónes

Recuperación de los datos de la asignatura, paralelo y docente

```
begin
        select
                tgm_asignaturass.nombre
        into    :nombre_asignatura
        from
                tgm_asignaturass,
                tgm_asignaruraspensums,
                tgm_pensums,
                tgm_carrerass
         where
                tgm_asignaturass.id_asignatura = :codigo_asignatura
        and     tgm_asignaturass.id_asignatura = tgm_asignaruraspensums.id_asignatura
        and     tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
        and     fecha_final is null
        and     tgm_carrerass.id_carrera = tgm_pensums.id_carrera
        and      tgm_carrerass.id_carrera = :global.carrera;


        exception when no_data_found then
                :system.message_level:=5;
                message('la asignatura no existe. revise');
                :system.message_level:=10;
                raise form_trigger_failure;
        when others then
                :system.message_level:=5;
                message('se ha presentado el error: '||sqlerrm);
                :system.message_level:=10;
                raise form_trigger_failure;
end;

begin
        select distinct docente
        into :docente
        from tgm_horarioss
        where id_asignatura = :codigo_asignatura and
                id_paralelo = :asignatura.paralelo and
                id_Período = :asignatura.Período;
        exception when no_data_found then
                :system.message_level:=5;
                message('la asignatura no existe. revise');
                :system.message_level:=10;
                raise form_trigger_failure;
        when others then
                :system.message_level:=5;
                message('se ha presentado el error: '||sqlerrm);
                :system.message_level:=10;
                raise form_trigger_failure;
end;
```

Ingreso de los valores de calificaciónes. calculo del total y estado de la asignatura

```
:nota1 := nvl(:nota1 , '0');
if :nota1 > 20 and :nota1  < 0 then
```

```
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
                :est := 'r';
        end if;
end if;

:nota2 := nvl(:nota2 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
                :est := 'r';
        end if;
end if;

:nota3 := nvl(:nota3 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
                :est := 'r';
        end if;

:nota4 := nvl(:nota4 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
```

```
                :est := 'r';
        end if;
end if;
```

Grabar datos en la tabla

```
declare
        es varchar2(1);
        estadoasistencias varchar2(1);
        estadorecord varchar2(1);
        creditosest number;

begin
        -- inserta o actualiza las notas en la tabla tgm_notass
        go_block('tgm_notass');
        first_record;
        loop
                begin
                        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
                                if :total >= 28 then
                                        es := 'a';
                                else
                                        es := 'r';
                                end if;
                                update tgm_notass
                                set nota1 = :nota1, nota2 = :nota2 , nota3 =
                                :nota3 , nota4 = :nota4, estado = es
                                where id_estudiante = :id_estudiante and  id_Período = :id_Período and
                                paralelo = :tgm_notass.paralelo and  id_asignatura = :id_asignatura;
                        exception when others then
                                message(sqlerrm);

                end;
                exit when :system.last_record = 'true';
                next_record;
        end loop;
        -- fin inserción

        -- actualiza los datos en la tabla tgm_record cuando esta en otro Período

        if :global.nuevoperiodo = 1 then
                go_block('tgm_notass');
                first_record;
                loop
                        begin
                                select estado
                                into estadoasistencias
                                from tgm_asistenciass
                                where id_estudiante = :id_estudiante and
                                id_Período = :id_Período and  paralelo = :tgm_notass.paralelo and
                                id_asignatura = :id_asignatura;

                                        if :est = 'a' and estadoasistencias = 'a' then
                                                estadorecord := 'a' ;
```

```
                                        else
                                                estadorecord := 'r' ;
                                        end if;

        -- obtiene los creditos de la asignatura
                        creditosest := obtienecreditos (:id_asignatura, estadorecord);
        -- actualiza el record del estudiante
                        update tgm_record
                                set promedio = :total, estado = estadorecord,
                                creditos = creditosest
                                where id_estudiante = :id_estudiante and
                                id_Período = :id_Período and id_asignatura = :id_asignatura;

                                exception when others then
                                        message(sqlerrm);
                                end;
                        exit when :system.last_record = 'true';
                                next_record;
                end loop;

        end if;
        -- fin inserta tabla record

        :system.message_level:=5;
        commit;
        message('datos grabados correctamente');
        --message(:id_estudiante || ',' ||  estadoasistencias);
        :system.message_level:=10;
end;
```

Equiparación de asignaturas

Recuperación de datos de estudiantes

```
declare
        est varchar2(10);

begin
        begin
        select apellidos_est || ' ' ||nombres_est
        into :nombre_estudiante
        from tgm_estudiantess
        where id_estudiante = :codigo_estudiante
                and estado = 1
                and id_carrera = :global.carrera;
        exception when no_data_found then
                        :system.message_level := 5;
                        message ('el estudiante no existe');
                        :system.message_level := 5;
                        raise form_trigger_failure;
        end;
        begin
                select id_carrera , nombre
                into :codigo_carrera1 , :nombre_carrera1
```

```
                from tgm_carrerass
                where id_carrera = :global.carrera;
        exception when no_data_found then
                :system.message_level := 5;
                message ('la carrera no existe');
                :system.message_level := 5;
                raise form_trigger_failure;
        end;
end;
```

Recuperación de los datos de la asignatura a equiparar

```
begin
select  tgm_asignaturass.nombre
                into :nombre_asignatura
from
        tgm_asignaturass,
        tgm_asignaruraspensums,
         tgm_pensums,
        tgm_carrerass
 where  tgm_asignaturass.id_asignatura = :codigo_asignatura
        and     tgm_asignaturass.id_asignatura = tgm_asignaruraspensums.id_asignatura
        and     tgm_asignaruraspensums.id_pensum = tgm_pensums.id_pensum
        and     fecha_final is null
        and     tgm_carrerass.id_carrera = tgm_pensums.id_carrera
        and     tgm_carrerass.id_carrera = :global.carrera;

exception when no_data_found then
        :system.message_level:=5;
        message('la asignatura no existe. revise');
        :system.message_level:=10;
        raise form_trigger_failure;
when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
```

Ingreso y recuperación de notas y asistencias para la asignatura a equiparar. obtener estado de la asignatura

Notas

```
:nota1 := nvl(:nota1 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
```

```
        else
                :est := 'r';
        end if;
end if;

:nota2 := nvl(:nota2 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
                :est := 'r';
        end if;
end if;

:nota3 := nvl(:nota3 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
                :est := 'r';
        end if;

:nota4 := nvl(:nota4 , '0');
if :nota1 > 20 and :nota1  < 0 then
        :system.message_level :=5;
        message('debe ingrear notas entre 1 y 20');
        raise form_trigger_failure;
        :system.message_level :=10;
else
        :total := obtienepromedio(:nota1,:nota2,:nota3,:nota4);
        if :total >= 28 then
                :est := 'a';
        else
                :est := 'r';
        end if;
end if;
```

Asistencias

```
declare
        minimo number;
        totala number;
```

```
begin
        :ha_mes1 := nvl(:ha_mes1, '0');
        totala := nvl(to_number(:ha_mes1),0) + nvl(to_number(:ha_mes2),0) + nvl(to_number(:ha_mes3),0) +
        nvl(to_number(:ha_mes4),0) + nvl(to_number(:ha_mes5),0)  ;
        :totala := totala;
        :estadoa := porcentajeasistencias(nvl(to_number(:totald),0), totala);
        exception when others then
                message (sqlerrm);
end;
```

Grabar datos en las tablas

```
declare
        total_creditos number;
begin
        if :estadon = 'a' and :estadoa = 'a' then
        begin
                 insert into tgm_notass values ( :nota4, :codigo_estudiante, :nota1,:nota3, 'a', :estadon,
                :global.Período, :nota2, :codigo_asignatura);
                exception when others then
                        :system.message_level:=5;
                        message('se ha producido el error: '||sqlerrm);
                        :system.message_level:=10;
        end;
        begin
                insert into tgm_asistenciass values (:ha_mes5,ha_mes4,:ha_mes1,'a',
                :estadoa,:ha_mes2,:codigo_asignatura,:global.Período,:ha_mes3,:codigo_estudiante);
                exception when others then
                        :system.message_level:=5;
                        message('se ha producido el error: '||sqlerrm);
                        :system.message_level:=10;
        end;

        begin
                 select numero_creditos
                 into total_creditos
                 from tgm_asignaturass
                 where id_asignatura = :codigo_asignatura;
                 exception when no_data_found then
                        :system.message_level:=5;
                        message('materia definida más de unas vez en la tabla asignaturas');
                        :system.message_level:=10;
                        raise form_trigger_failure;
        end;

        begin
                 insert into tgm_record values (:codigo_estudiante,
        :codigo_asignatura,:global.Período,:total,:totala,:totald,
                total_creditos, :estadoa);
                exception when others then
                        :system.message_level:=5;
                        message('se ha producido el error: '||sqlerrm);
                        :system.message_level:=10;
        end;
```

```
                :system.message_level:=5;
                commit;
                message('datos grabados correctamente');
                :system.message_level:=10;
        else
                :system.message_level:=5;
                message('está reprobado... revise');
                :system.message_level:=10;
        end if;
end;
```

Cambio de paralelo

Recupera los datos de la asignatura

```
begin
        select
                nombre
        into
                :nombre_asignatura
        from
                tgm_asignaturass,
                tgm_paraleloss
        where
                        tgm_paraleloss.id_asignatura = :cod_asignatura
                and     tgm_asignaturass.id_asignatura = tgm_paraleloss.id_asignatura
                and     tgm_paraleloss.id_carrera = :global.carrera
                and     tgm_paraleloss.id_Período = :global.Período
                and     tgm_paraleloss.id_paralelo = 'a';
                exception when no_data_found then
                        :system.message_level:=5;
                        message('carrera o materia no existe...');
                        :system.message_level:=10;
                        raise form_trigger_failure;
                when too_many_rows then
                        :system.message_level:=5;
                        message('existen definidos más de dos paralelos. revise');
                        :system.message_level:=10;
                        raise form_trigger_failure;
end;
```

Guardar datos en la tabla

```
declare
        maximo_alumnos number;
        cantidad_alumnos number;
begin
        go_block('block16');
        first_record;
        loop
                if :paralelo_ingresa != :paralelo then
                        begin
                                select
                                        numero_alumnos,
```

```
                numero_estudiantes
        into
                maximo_alumnos,
                cantidad_alumnos
        from
                tgm_asignaturass,
                tgm_paraleloss
        where
                tgm_paraleloss.id_asignatura = :cod_asignatura
                and tgm_asignaturass.id_asignatura = tgm_paraleloss.id_asignatura
                and tgm_paraleloss.id_paralelo = :paralelo
                and tgm_paraleloss.id_Período = :global.Período;
exception when no_data_found then
                :system.message_level:=5;
                message('paralelo no definido...');
                :system.message_level:=10;
                raise form_trigger_failure;
end;

if maximo_alumnos < cantidad_alumnos then
        :system.message_level:=5;
        message('no existe cupo para el paralelo: '||:paralelo);
        :system.message_level:=10;
        raise form_trigger_failure;
end if;

update tgm_asistenciass
set paralelo = :paralelo
where id_estudiante = :id_estudiante
and id_asignatura = :cod_asignatura
and id_Período = :global.Período
and paralelo = :paralelo_ingresa;

update tgm_notass
set paralelo = :paralelo
where id_estudiante = :id_estudiante
and id_asignatura = :cod_asignatura
and id_Período = :global.Período
and paralelo = :paralelo_ingresa;

update tgm_paraleloss
set numero_estudiantes = numero_estudiantes + 1
where id_asignatura = :cod_asignatura
and id_Período = :global.Período
and id_paralelo = :paralelo
and id_carrera = :global.carrera;

update tgm_paraleloss
set numero_estudiantes = numero_estudiantes - 1
where id_asignatura = :cod_asignatura
and id_Período = :global.Período
and id_paralelo = :paralelo_ingresa
and id_carrera = :global.carrera;
```

```
                    :system.message_level:=5;
                    commit;
                    :system.message_level:=10;
                end if;
        exit when :system.last_record = 'true';
                next_record;
end loop;
:system.message_level:=5;
message('datos grabados correctamente');
:system.message_level:=10;
end;
```

Cambio de carrera

```
declare
        pensum varchar2(5);
        carrera varchar2(3);

begin
        begin
                Comprobación de la existencia del estudiante
                select id_estudiante, nombres_est, apellidos_est, id_pensum, id_carrera
                into :id_estudiante, :nombre , :apellido, pensum , carrera
                from tgm_estudiantess
                where   id_estudiante = :id_estudiante
                                and estado = 1
                                and id_carrera = :global.carrera;
                exception when no_data_found then
                        :system.message_level:=5;
                        message('estudiante no existe');
                        raise form_trigger_failure;
                        :system.message_level:=10;
        end;

        begin
                Selecciónа  de la carrera del estudiante
                select id_carrera , nombre
                into :id_carrera , :nombre_carrera
                from tgm_carrerass
                where id_carrera = carrera;
                exception when no_data_found then
                        :system.message_level:=5;
                        message('carrera no existe');
                        raise form_trigger_failure;
                        :system.message_level:=10;
        end;
        begin


                Selección del pensum correspondiente a la carrera
select id_pensum , observación
                into :codigo_pensum , :pensum
                from tgm_pensums
```

```
                where id_pensum = pensum;
                exception when no_data_found then
                        :system.message_level:=5;
                        message('pensum no existe');
                        raise form_trigger_failure;
                        :system.message_level:=10;
        end;
end;

begin
        -- extrae el nombre de la carrera a la que pertenece el estudiante
        select nombre
        into :nombre_carrera
        from tgm_carrerass,
                tgm_pensums
        where tgm_carrerass.id_carrera = :id_carrera
                and tgm_carrerass.id_carrera =      tgm_pensums.id_carrera
                and tgm_pensums.fecha_final is null
                and tgm_carrerass.id_carrera <> 0;

        exception when no_data_found then
                :system.message_level:=5;
                message('carrera no definida');
                :system.message_level:=10;
                raise form_trigger_failure;
                when too_many_rows then
                :system.message_level:=5;
                message('existen más de una carrera con el mismo código, revise');
                :system.message_level:=10;
                raise form_trigger_failure;
                when others then
                :system.message_level:=5;
                message('se ha presentado el error: '||sqlerrm);
                :system.message_level:=10;
                raise form_trigger_failure;
end;
```

Grabar datos en la tabla

```
declare
        cursor c1 is
        select
                id_asignatura
        from
                tgm_record
        where
                id_estudiante = :id_estudiante
                and estado = 'a'
                and id_asignatura like :id_carrera||'%';asignatura varchar2(10);
                laasignatura varchar2(10);
begin
        asignatura:= :id_carrera_c||'%';
        for reg in c1 loop
                -- obtiene el alias de las asignaturas de la nueva carrera
```

```
                begin
                        select
                                alias
                        into
                                laasignatura
                        from
                                tgm_alias
                        where
                                id_asignatura = reg.id_asignatura
                                and alias like :id_carrera_c||'%';

                                message(laasignatura);
                                -- actualizo la tabla record
                                update tgm_record
                                set id_asignatura = laasignatura
                                where id_estudiante = :id_estudiante
                                and id_asignatura = reg.id_asignatura;

                                update tgm_estudiantess
                                set id_carrera = :id_carrera_c , id_pensum = :codigo_pensum_c
                                where id_estudiante = :id_estudiante;

                exception when too_many_rows then
                        :system.message_level:=5;
                        message('asignatura: '||reg.id_asignatura||' tiene más de un alias... revise');
                        :system.message_level:=10;
                        raise form_trigger_failure;
                when no_data_found then
                        null;
                end;
        end loop;
        :system.message_level:=5;
        commit;
        message('datos grabados correctamente...');
        :system.message_level:=10;
end;
```

Asignación de créditos adiciónales para los estudiantes

```
declare
        carrera varchar2(3);
begin
```

Verifica si el estudiante existe para asignarle créditos adiciónales

```
        select apellidos_est,  nombres_est,id_carrera
        into :apellidos, :nombres, carrera
        from tgm_estudiantess
        where id_estudiante = :codigo_estudiante
                        and estado = '1'
                        and id_carrera = :global.carrera;
        -- selecciónа la carrera a la que pertenece
        :codigo_carrera := carrera;
        select nombre
        into :nombre_carrera
        from tgm_carrerass
```

```
        where id_carrera = carrera;
exception when no_data_found then
                :system.message_level :=5;
                message ('el estudiante no existe. consulte');
                raise form_trigger_failure;
                :system.message_level :=5;
end;

begin
        -- extrae el nombre de la carrera y el pensum activo del estudiante a asignarle el crédito
        select nombre
        into :nombre_carrera
        from tgm_carrerass,        tgm_pensums
        where tgm_carrerass.id_carrera = :codigo_carrera
                and tgm_carrerass.id_carrera =      tgm_pensums.id_carrera
                and tgm_pensums.fecha_final is null
                and tgm_carrerass.id_carrera <> 0;

exception when no_data_found then
        :system.message_level:=5;
        message('carrera no definida');
        :system.message_level:=10;
        raise form_trigger_failure;
        when too_many_rows then
        :system.message_level:=5;
        message('existen más de una carrera con el mismo código, revise');
        :system.message_level:=10;
        raise form_trigger_failure;
        when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;

begin
        --verifica si el crédito a agregar al estudiante existe y además si pertenece a su carrera
         select id_credito
         into :nombre_credito
         from tgm_creditosadiciónales
         where   id_credito = :id_credito
                        and id_carrera = :codigo_carrera;

exception when no_data_found then
        :system.message_level:=5;
        message('el crédito a insertar no existe. revise');
        :system.message_level:=10;
        raise form_trigger_failure;
when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
```

```
end;

declare
        per varchar(6);
begin
        -- permite seleccionar el período en el cual se asigna el crédito al estudiante
        select id_Período
        into per
        from tgm_periodos
        where id_Período = :id_Período;
exception when no_data_found then
        :system.message_level:=5;
        message('el período a insertar no existe. revise');
        :system.message_level:=10;
        raise form_trigger_failure;
when others then
        :system.message_level:=5;
        message('se ha presentado el error: '||sqlerrm);
        :system.message_level:=10;
        raise form_trigger_failure;
end;
```

Grabar los datos en la tabla

```
:system.message_level:=5;
commit;
message('los datos se grabaron correctamente');
:system.message_level:=10;
```

OPCIÓNES DE INTERNET

Ingreso a la opción de Internet

```
declare
  lacarrera varchar2(3);
  nregistros number;

begin

  :global.carrera := :codigo_estudiante;
  select id_carrera
  into lacarrera
  from tgm_estudiantess
  where id_estudiante = :codigo_estudiante;
  :global.carrera := lacarrera;
  if lacarrera = :codigo_carrera then
          if lacarrera = 15 then
              :global.nbc := 1;
          else
              :global.nbc := 0;
          end if;
          :global.estudiante := :codigo_estudiante;
          hide_window('ingreso_internet');
```

```
            call_form('c:\latesis\formas\internet.fmx', hide,  do_replace);
            exit_form(no_validate,full_rollback);
   else
            :system.message_level:=5;
            message('el estudiante no pertenece a la carrera ingresada');
            raise form_trigger_failure;
            :system.message_level:=10;
   end if;
exception when no_data_found then
   :system.message_level:=5;
   message('el estudiante a consultar no existe o está inactivo. comuniquese con la UTPL');
   raise form_trigger_failure;
   :system.message_level:=10;
end;
```

Ingreso a la opción Ficha de inscripción desde el Internet

```
declare
   viene_nbc number;
begin
:global.internet := 1;
if :global.nbc = 1 then
            viene_nbc:= estudiante_viene_nbc(:global.estudiante);
            if viene_nbc = '0' then
                :system.message_level:=5;
                message('estudiante ha reprobado el nbc. no puedo matricularlo... revise');
                :system.message_level:=10;
                raise form_trigger_failure;
            elsif viene_nbc = '1' then
                :system.message_level:=5;
                message('se ha producido un error durante la verificación de las cadenas');
                :system.message_level:=10;
                raise form_trigger_failure;
            end if;
            hide_window('internet');
            call_form('c:\latesis\formas\matricula_carrera.fmx', hide,  do_replace);
else
            hide_window('internet');
            call_form('c:\latesis\formas\matricula_carrera.fmx', hide,  do_replace);
end if;
end;
```

OPCIÓNES TESORERA

Registro de pagos

Recuperación de datos personales del estudiante, carrera y fichas pendientes de cancelación

```
declare
        carrera varchar2(3);
begin
        begin
```

```
        -- verifica si el estudiante existe
        select apellidos_est, nombres_est,id_carrera
        into :apellidos, :nombres, carrera
        from tgm_estudiantess
        where id_estudiante = :codigo_estudiante
        and estado = '1';
        :codigo_carrera := carrera;

        select nombre
        into :nombre_carrera
        from tgm_carrerass
        where id_carrera = carrera;

        exception when no_data_found then
                :system.message_level :=5;
                message ('el estudiante no existe. consulte');
                raise form_trigger_failure;
                :system.message_level :=5;

end;

begin
        select id_matricula
        into :ficha
        from tgm_ficha_inscripciónes
        where id_estudiante = :codigo_estudiante
                and id_Período = :global.Período
                and estado_cancelación = 'p';

        exception when no_data_found then
                :system.message_level :=5;
                message ('no tiene ficha por regristrar el pago');
                raise form_trigger_failure;
                :system.message_level :=5;

        when too_many_rows then
                :system.message_level :=5;
                message ('tiene dos fichas a reguistrar. ctrl+l para selecciónа la ficha');
                :system.message_level :=5;

end;
end;
```

Actualización del estado de cancelación

```
if :checa = 1 then
        :status := 'c';
else
        :status := 'p';
end if;
```

Grabar datos en la tabla e imprimir factura

```
declare
        cursor c1 is
```

```
        select *
        from  tgm_detalles
        where id_matricula = :id_matricula;
        aux number;
        pago number;
begin
        go_block('tgm_pagoss');
        first_record;
        pago:= :pago;
        last_record;
        aux:=0;
        loop
                if :checa = 1 then
                        aux:= 1;

                else
                        if aux = 1 then
                                :system.message_level:=5;
                                message('debe seleccionar la cuota que le toca');
                                :system.message_level:=10;
                                raise form_trigger_failure;
                                aux:=2;
                        end if;
                end if;
                exit when :pago = pago;
                previous_record;
        end loop;

        if aux = 1 then
                        if :pago = 1 then
                                for reg in c1 loop
                                begin
                                        insert into tgm_notass values (null, :codigo_estudiante, null, null,
                                        reg.id_paralelo, 'm', :global.Período,null,reg.id_asignatura);
                                        exception when others then
                                        :system.message_level:=5;
                                                message('se ha producido el error notas: '||sqlerrm);
                                                :system.message_level:=10;
                                                raise form_trigger_failure;
                                        end;
                                        begin
                                                insert into tgm_asistenciass values(null,null,null,reg.id_paralelo,
                                                'm', null,reg.id_asignatura,:global.Período,null,
                                                :codigo_estudiante);
                                                exception when others then
                                                        :system.message_level:=5;
                                                        message('se ha producido el error asistencias: '||sqlerrm);
                                                        :system.message_level:=10;
                                                        raise form_trigger_failure;
                                                end;
                                end loop;
                end if;
```

```
                -- modificar los datos del estado en la ficha
                update tgm_ficha_inscripciónes
                set estado_cancelación = 'c'
                where id_matricula = :ficha;
                :system.message_level:=5;
                commit;
                message('los datos se grabaron correctamente');
                :system.message_level:=10;
        if :codigo_carrera          = '15' then
                web.show_document
                ('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&re
                port=c:\latesis\reportes\factura_nbc.rdf&userid=sistemagm/sistemagm@sistema&codigo_estud
                iante='||:codigo_estudiante||'&ficha='||:id_matricula||'&Período='||:global.Período);
        else
                web.show_document
                ('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&re
                port=c:\latesis\reportes\factura_carrera.rdf&userid=sistemagm/sistemagm@sistema&codigo_e
                studiante='||:codigo_estudiante||'&ficha='||:id_matricula||'&Período='||:global.Período);
        end if;
        end if;
end;
```

FUNCIÓNES DE BASE

Procedimiento que inserta en la tabla record los datos de notas y asistencias

```
create or replace procedure inserta_record is
cursor c1 is
select
  id_estudiante,
  id_asignatura,
  id_Período,
  paralelo,
  nvl(nota1,0) nota1,
  nvl(nota2,0) nota2,
  nvl(nota3,0) nota3,
  nvl(nota4,0) nota4
from tgm_notass
where id_Período = :global.Período;
porcentaje_asistencia  number;
nota_minima number;
promedio number;
horas_recibidas number;
horas_dictadas number;
creditos_materia number;
porcentaje_horas number;
nota_min number;
begin
  for reg in c1 loop
      if reg.nota3 = 0 and reg.nota4 = 0 then
          promedio:= reg.nota1 + reg.nota2;
      else
          if reg.nota3 != 0 and reg.nota4 != 0 then
```

```
            promedio:= reg.nota3 + reg.nota4;
        else
            if reg.nota3 = 0 then
                promedio:= reg.nota1 + reg.nota4;
            else
                promedio:= reg.nota3 + reg.nota2;
            end if;
        end if;
end if;
-- para obtener el total de horas recibidas
select sum(nvl(ha_mes1,0)+
        nvl(ha_mes2,0)+
        nvl(ha_mes3,0)+
        nvl(ha_mes4,0)+
        nvl(ha_mes5,0))
into horas_recibidas
from tgm_asistenciass
where id_asignatura = reg.id_asignatura
and id_Período = reg.id_Período
and id_estudiante = reg.id_estudiante
and paralelo = reg.paralelo;
-- para obtener el total de horas dictadas
select nvl(sum(nvl(hd_mes5,0)+
        nvl(hd_mes3,0)+
        nvl(hd_mes4,0)+
        nvl(hd_mes1,0)+
        nvl(hd_mes2,0)),0)
into horas_dictadas
from tgm_valoress
where id_asignatura =reg.id_asignatura
and id_Período = reg.id_Período
--and id_estudiante = reg.id_estudiante
and paralelo = reg.paralelo;
select
        porcentaje_asist,
        nota_min
into
        porcentaje_asistencia,
        nota_minima
from tgm_fechas
where id_Período = reg.id_Período;
porcentaje_horas:= (horas_dictadas * porcentaje_asistencia)/100;
if horas_recibidas >= porcentaje_horas and promedio >= nota_min then
        select numero_creditos
        into creditos_materia
        from tgm_asignaturass
        where id_asignatura = reg.id_asignatura;
        insert into tgm_record  values (reg.id_estudiante,
                reg.id_asignatura,
                reg.id_Período,
                promedio,
                horas_recibidas,
                horas_dictadas,
```

```
                creditos_materia,
                'a');
        else
            insert into tgm_record  values (reg.id_estudiante,
                reg.id_asignatura,
                reg.id_Período,
                promedio,
                horas_recibidas,
                horas_dictadas,
                0,
                'r');
        end if;
    end loop;
end;
/
```

Función que devuelve el nombre del estado de la asignatura

```
create or replace function estados_asig(est char) return char is
begin
   if est = 'a'  then
        return 'aprobada';
   elsif est = 'r'  then
        return 'reprobada';
   elsif est = 'e'  then
        return 'equiparada';
   end if;
end;
```

Función que devuelve el estado de la asignatura en notas. como parametro recibe el promedio

```
create or replace function estadonotas(promedio number, est_asis char) return char is
 estfin char;
 begin
   if (promedio >= 28) and (est_asis = 'a') then
        return ('aprobada');
   else
        return ('reprobada');
   end if;
 end;
```

Función que devuelve el codigo de la carrera del estudiante que aprobo el nbc, determinado por la asignatura introducción a la carrera

```
create or replace function estudiante_viene_nbc(elestudiante char) return char is
carrera varchar2(3);
id_ficha number;
id_elperiodo varchar2(6);
asignatura varchar2(8);
cursor materias_nbc(laficha number) is
 select *
 from tgm_detalles
 where id_matricula = laficha;
 retorna varchar2(1);
begin
   -- selecciónо la carrera
```

```
select id_carrera
into carrera
from  tgm_estudiantesss
where id_estudiante = elestudiante;
retorna := 2;
if carrera = '15' then
      begin
       select max(id_matricula), id_Período
          into id_ficha, id_elperiodo
          from tgm_ficha_inscripciónes
          where id_estudiante = elestudiante
          group by id_Período;
       exception when no_data_found then
          id_ficha:= 0;
          id_elperiodo:=0;
          retorna:= 0;
      end;
      for reg in materias_nbc(id_ficha) loop
       begin
          select id_asignatura
          into asignatura
          from tgm_record
          where id_estudiante = elestudiante
          and id_asignatura = reg.id_asignatura
          and estado = 'a';
       exception when no_data_found then
          retorna:= 0;
       when others then
          retorna:= 1;
       end;
      end loop;
  else
      retorna:= 3;
  end if;
return retorna;
end;
```

Función que devuelve el numero de horas de cruce de horario. recibe como parametro la hora de inicio y fin de la asignatura 1 y la hora de inicio y fin de la asignatura 2

```
create or replace function verifica_horario(hi1 number, hs1 number, hi2 number, hs2 number) return number is
diferencia1 number;
diferencia2 number;
menor_i number;
mayor_s number;
begin
  diferencia1:=  abs(hi1 - hi2) + abs(hs1 - hs2);
  if hi1 < hi2 then
      menor_i:= hi1;
  else
      menor_i:= hi2;
  end if;
  if hs1 > hs2 then
      mayor_s:= hs1;
```

```
  else
      mayor_s:= hs2;
  end if;
  diferencia2:= mayor_s - menor_i;
  return diferencia2 - diferencia1;
end;
```

Función que determina el numero de materias que tomo un estudiante en un Período. recibe como parametro el codigo del estudiante

```
create or replace function contarmaterias(estudiante char) return number
is
numero number;
begin
  select count (*)
  into numero
  from tgm_notass
  where id_estudiante = estudiante;
  return numero;
end;
```

Función que determina el promedio final de una asignatura. recibe como parametro las notas (b1, b2, s1, s2)

```
create or replace function obtienepromedio(nota1 number, nota2 number, nota3 number, nota4 number) return
number is
promedio number;
begin
      if nota3 = 0 and nota4 = 0 then
          promedio:= nota1 + nota2;
      else
          if nota3 != 0 and nota4 != 0 then
              promedio:= nota3 + nota4;
          else
              if nota3 = 0 then
                  promedio:= nota1 + nota4;
              else
                  promedio:= nota3 + nota2;
              end if;
          end if;
      end if;
  return promedio;
end;
```

Función que determina los requsitos de una asignatura

```
create or replace function requisitos (laasignatura char) return char is
cursor c1 is
 select requisito
 from tgm_requisitos
 where id_asignatura =  laasignatura;
 losrequisitos varchar2(1000);
begin
 for reg in c1 loop
   losrequisitos:= losrequisitos||','||reg.requisito;
```

```
 end loop;
 losrequisitos:= substr(losrequisitos,2,length(losrequisitos));
 return losrequisitos;
end;
```

Función que determina el promedio de un estudiante

```
create or replace function promedio_total(cedula char) return number is
   promedio number(4,2);
cursor c1 is
select
   promedio
from tgm_record
where tgm_record.id_estudiante = cedula and tgm_record.estado = 'a';
cuenta number;
   parcial number(4,2);
   total number;
begin
   total := 0;
   cuenta :=0;
   for reg in c1 loop
        total := total + reg.promedio
        cuenta := cuenta + 1;
   end loop;
   promedio := total / cuenta;
   return promedio;
end;
```

Procedimiento posición ventana

```
procedure posición_ventana(nombre varchar2) is
          vappx number(4);
          vappy number(4);
          vwinx number(4);
          vwiny number(4);
begin
          vappx := 670;
          vappy := 345;
          vwinx := get_window_property(nombre,width);
          vwiny :=  get_window_property(nombre,height);
          show_window(nombre, (vappx-vwinx)/2, (vappy-vwiny)/2);
end;
```

LISTAS DE VALORES

Lista de valores de carrera

```
select   tgm_carrerass.id_carrera, nombre
from   tgm_carrerass,   tgm_pensums
where   tgm_carrerass.id_carrera = tgm_pensums.id_carrera and
             tgm_pensums.fecha_final is null
```

Lista de valores de niveles de usuario

```
select *
```

```
from tgm_niveles
```

Lista de estado del estudiante

```
select *
from  tgm_estadose
```

Lista de valores colegios

```
select *
from tgm_colegios
```

Lista de estado civiles

```
select *
from tgm_estadosciviles
```

Lista de valores de tipos de crédito

```
select *
from tgm_tipo_creditos
```

Lista de valores de datos de carreras

```
select id_carrera, nombre
from tgm_carrerass
```

Lista de valores de tipos de asignatura

```
select *
from tgm_tipo_asigs
```

Lista de valores de modalidades de asignatura

```
select *
from tgm_modalidadesasigs
```

Lista de valores de datos de asignaturas asociadas a un plan de estudios

```
select
        tgm_asignaturass.nombre,
        tgm_asignaturass.id_asignatura,
        tgm_carrerass.nombre
from
        tgm_asignaturass,
        tgm_asignaturaspensums,
        tgm_pensums,
        tgm_carrerass
where
                tgm_asignaturass.id_asignatura = tgm_asignaturaspensums.id_asignatura
        and     tgm_asignaturaspensums.id_pensum = tgm_pensums.id_pensum
        and     fecha_final is null
        and     tgm_carrerass.id_carrera = tgm_pensums.id_carrera
        and     tgm_asignaturass.id_asignatura <> :codigo_asignatura
        and     tgm_asignaturaspensums.id_pensum <>:codigo_pensum
```

Lista de valores de periodos

```
select
        id_Período, descripción
from
```

```
        tgm_periodos
order by id_Período
```

Lista de valores de estudiantes con estado activo matriculados en una carrera determinada

```
select    apellidos_est || ' ' || nombres_est nombres
                , id_estudiante
 from     tgm_estudiantess
 where    estado = 1
                and id_carrera = :global.carrera
order by apellidos_est
```

Lista de valores de paralelos con su asignatura respectiva

```
select
        tgm_asignaturass.id_asignatura,
        nombre
from
        tgm_asignaturass,
        tgm_paraleloss
where
                tgm_asignaturass.id_asignatura = tgm_paraleloss.id_asignatura
        and     tgm_paraleloss.id_carrera = :global.carrera
        and     tgm_paraleloss.id_Período = :global.Período
```

Lista de valores para obtener todas las carreras excepto la que esta tomando el estudiante

```
select
        tgm_carrerass.id_carrera, tgm_carrerass.nombre
from
        tgm_carrerass, tgm_pensums
where
         tgm_carrerass.id_carrera = tgm_pensums.id_carrera
        and fecha_final is null
        and tgm_carrerass.id_carrera <> '0'
minus
select
        tgm_carrerass.id_carrera, tgm_carrerass.nombre
from
        tgm_carrerass, tgm_pensums
where
        tgm_carrerass.id_carrera = tgm_pensums.id_carrera
        and fecha_final is null
        and tgm_carrerass.id_carrera =  :id_carrera
```

Lista de valores de estudiantes con fichas de inscripción pendientes de cancelación

```
select    apellidos_est || ' ' || nombres_est nombres,
                id_matricula,
                tgm_estudiantess.id_estudiante ,
                fecha_emisión ,
                decode (id_pago, 1 , 'contado', 'crédito') tipo_pago

from tgm_estudiantess, tgm_ficha_inscripciónes

where           tgm_estudiantess.id_estudiante = tgm_ficha_inscripciónes.id_estudiante
```

and estado_cancelación = 'p' and id_Período = :global.Período

order by apellidos_est , fecha_emisión , id_matricula

REPORTES

ADMINISTRADOR

Reporte de usuarios disponibles en el sistema

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\usuarios.rdf&userid=sistemagm/sistemagm@sistema');

Reporte de todas las carreras existentes

Recibe como parámetro el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\carreras.rdf&userid=sistemagm/sistemagm@sistema&c_Período='|| :global.Período);

Reporte de las asignaturas y sus requisitos que pertenecen a un plan de estudios determinado

Recibe como parámetro el código del pensum de una carrera

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\planes_de_estudio.rdf&userid=sistemagm/sistemagm@sistema&p_pensum='||:pensum);

SECRETARIA

Ficha de inscripción

Recibe como parámetro el código del estudiante , el número de la ficha y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\ficha_inscripción_carreras.rdf&userid=sistemagm/sistemagm@sistema&codigo_estudiante='||:block87.codigo_estudiante||'&ficha='|| :num_ficha || '&Período='||:global.Período);

Horarios de clases de una carrera

Recibe como parámetro el código de la carrera y el período del cual se quiere obtener el horario

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\horarios_por_ciclo_carrera.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera ||'&c_Período='|| :global.Período);

Reporte de créditos por estudiante

Recibe como parámetro el código del estudiante del cual se quiere obtener los créditos acumulados en .la carrera

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\creditos_asignaturas.rdf&userid=sistemagm/sistemagm@sistema&cedula='||:global.estudiante);

Reporte de pagos pendientes

Recibe como parámetro el código de la carrera, el código del estudiante y el código del Período a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\pagos_pendientes_individuales.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera||'&cedula='||:global.estudiante || '&Período=' || :global.Período);

Reporte que muestra los requisitos de una asignatura

Recibe como parámetro el código de la asignatura a obtener los requisitos

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\requisitos_asignatura.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatura);

Reporte que muestra el horario de las asignaturas en las que se matriculó el estudiante

Recibe como parámetro el código de la carrera, el código del estudiante y el período a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\horarios_alumno.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:codigo_carrera ||
'&cedula='||:codigo_estudiante ||'&Período='|| :Período_reporte);

Reporte de matriculados por sexo en una carrera

Recibe como parámetro el código de la carrera, el período a consultar y el sexo a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estudiantes_por_carrera_sexo.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:codigo_carrera ||'&c_Período='|| :Período_reporte || '&sexo='||:estado);

Reporte de matriculados por estado civil en una carrera

Recibe como parámetro el código de la carrera, el período y el estado civil a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estudiantes_por_carrera_estadocivil.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:codigo_carrera ||'&c_Período='|| :Período_reporte || '&est_civil='|| :estado);

Reporte de matriculados por estado del estudiante en una carrera

Recibe como parámetro el código de la carrera, el período y el estado del estudiante a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estado_estudiante.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:codigo_carrera ||'&c_Período='|| :Período || '&estado='||:estado);

Reporte de matriculados por carrera

Recibe como parámetro el código de la carrera y el período a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estudiantes_por_carrera.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:global.carrera ||
'&c_Período=' || :global.Período);

Listado de estudiantes por asignatura y paralelo

Recibe como parámetro el código de la asignatura, el código de la carrera, el paralelo y el período.

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep

ortes\listado_paralelos.rdf&userid=sistemagm/sistemagm@sistema&c_asignatura='||:codigo_asignatura ||'&c_carrera='|| :codigo_carrera || '&c_paralelo='||:paralelo ||'&c_Período='|| :Período);

Listado de estudiantes por estado de asignatura

Recibe como parámetro el código de la asignatura, el estado de la asignatura y el período a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estado_asignatura_alumnos1.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatura ||'&estado='|| :estado || '&Período='||:Período);

Listado de notas y asistencias del estudiante en las asignaturas que se matriculó

Recibe como parámetro el código del estudiante y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\notas_asistencias_alumnos.rdf&userid=sistemagm/sistemagm@sistema&codigo_est='||:codigo_estudiante || '&Período=' || :global.Período);

Listado de notas y asistencias de los estudiantes matriculados en una carrera con todas las asignaturas tomadas

Recibe como parámetro el código de la carrera y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_notas_asistencias_matriculados_ciclo.rdf&userid=sistemagm/sistemagm@sistema&carrera=' || :global.carrera || '&Período='||:global.Período);

Listado de notas y asistencias por asignatura y paralelo

Recibe como parámetro el código de la asignatura, el paralelo, y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\asignatura_notas_asistencias_alumnos.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatura || '&paralelo='|| :paralelo ||'&Período='|| :global.Período);

Listado para registrar notas de estudiantes por asignatura y paralelo

Recibe como parámetro el código de la asignatura, el paralelo, la carrera y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\listado_registra_notas_paralelos.rdf&userid=sistemagm/sistemagm@sistema&c_asignatura='||:codigo_asignatura ||'&c_carrera='|| :codigo_carrera || '&c_paralelo='||:paralelo ||'&c_Período='|| :global.Período);

Listado para registrar asistencias de estudiantes por asignatura y paralelo

Recibe como parámetro el código de la asignatura, el paralelo, la carrera y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\listado_registra_asistencias_paralelos.rdf&userid=sistemagm/sistemagm@sistema&c_asignatura='||:codigo_asignatura ||'&c_carrera='|| :codigo_carrera || '&c_paralelo='||:paralelo ||'&c_Período='|| :global.Período);

Promedios por carrera

Recibe como parámetro el código de la carrera

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\promedio_general.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera);

Promedio por estudiante
Recibe como parámetro el código del estudiante
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\promedio_general_asignaturas_aprobadas.rdf&userid=sistemagm/sistemagm@sistema&c_estudiante='||:codigo_estudiante);

Listado de notas por asignatura y paralelo
Recibe como parámetro el código de la asignatura, el paralelo, y el Período actual
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\notas_asignatura_Período.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatura || '&paralelo='|| :paralelo ||'&Período='|| :Período);

Certificado de asignaturas tomadas en un ciclo
Recibe como parámetro el código de la carrera, el Período a consultar, el codigo del estudiante, y el Período actual
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_materias_ciclo_anterior.rdf&userid=sistemagm/sistemagm@sistema&carrera1='||:codigo_carrera ||'&c_ciclo_aprobado='|| :Período_reporte || '&c_estudiante='||:codigo_estudiante ||'&c_Período_actual='|| :Período);

Récord del estudiante de todas las asignaturas
Recibe como parámetro el codigo del estudiante, y el Período actual
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_general_todas_las_asignaturas.rdf&userid=sistemagm/sistemagm@sistema&c_estudiante='||:codigo_estudiante || '&c_Período='||:Período);

Récord del estudiante de todas las asignaturas aprobadas
Recibe como parámetro el codigo del estudiante, y el Período actual
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_general_asignaturas_aprobadas.rdf&userid=sistemagm/sistemagm@sistema&c_estudiante='||:codigo_estudiante || '&c_Período='||:Período);

Certificado de matricula y asistencia a clase
Recibe como parámetro el codigo del estudiante, y el Período a consultar
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\certficado_matricula_est.rdf&userid=sistemagm/sistemagm@sistema&c_Período='||:Período||'&c_estudiante='||:codigo_estudiante);

Certificado de asignaturas aprobadas en un ciclo
Recibe como parámetro el Período a consultar y el codigo del estudiante

```
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\rep_gen_asig_todos.rdf&userid=sistemagm/sistemagm@sistema&c_estudiante='||:codigo_estudiante
||'&c_Período='|| :Período_reporte);
```

Certificado de las asignaturas aprobadas y equiparadas en la carrera

Recibe como parámetro el Período actual y el codigo del estudiante

```
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\reporte_general_asignaturas_aprobadas.rdf&userid=sistemagm/sistemagm@sistema&c_estudiante='||:codi
go_estudiante ||'&c_Período='|| :Período_reporte);
```

TESORERA

Pagos pendientes por estudiante

Recibe como parámetros el Período actual, carrera y la cédula del estudiante.

```
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\pagos_pendientes_individuales.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:codigo_carrera||'&
cedula='||:codigo_estudiante || '&Período=' || :Período );
```

Pagos pendientes por carrera

Recibe como parámetros la carrera y el Período actual

```
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\pagos_pendientes.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera ||'&Período='||
:global.Período);
```

DIRECTORES DE ESCUELA

Obtener la carrera del director que ingresa

```
select nombre
        into :carrera
from tgm_carrerass
where id_carrera = :global.carrera ;
```

Profesiónales en formación

Por estado de asignatura

Recibe como parámetro el código de la asignatura, el estado de la asignatura y el período a consultar

```
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\estado_asignatura_alumnos1.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatur
a ||'&estado='|| :estado || '&Período='||:Período);
```

Por sexo

Recibe como parámetro el código de la carrera, el período a consultar y el sexo a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estudiantes_por_carrera_sexo.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:codigo_carrera ||'&c_Período='|| :Período_reporte || '&sexo='||:estado);

Por estado civil

Recibe como parámetro el código de la carrera, el período y el estado civil a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estudiantes_por_carrera_estadocivil.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:codigo_carrera ||'&c_Período='|| :Período_reporte || '&est_civil='|| :estado);

Preinscritos y matriculados

Recibe como parámetros la carrera y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_directora.rdf&userid=sistemagm/sistemagm@sistema&carrera=' || :global.carrera || '&Período='||:global.Período);

Por carrera

Recibe como parámetro el código de la carrera y el período a consultar

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\estudiantes_por_carrera.rdf&userid=sistemagm/sistemagm@sistema&c_carrera='||:global.carrera || '&c_Período=' || :global.Período);

Deudas pendientes

Recibe como parámetros la carrera y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\pagos_pendientes.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera ||'&Período='|| :global.Período);

Inscritos por asignatura

Recibe como parámetros la carrera y el Período actual

web.show_document('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_directora.rdf&userid=sistemagm/sistemagm@sistema&carrera=' || :global.carrera || '&Período='||:global.Período);

Asignaturas

Asignaturas por plan de estudios

Recibe como parámetro el código del pensum de una carrera

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\planes_de_estudio.rdf&userid=sistemagm/sistemagm@sistema&p_pensum='||:pensum);

Requisitos por asignatura

Recibe como parámetro el código de la asignatura a obtener los requisitos

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\requisitos_asignatura.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatura);

Horarios de la carrera

Recibe como parámetros la carrera y el Período actual

web.show_document('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\horarios_por_ciclo_carrera.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera ||'&c_Período='|| :global.Período);

Paralelos, notas y asistencias

Por estudiante

Recibe como parámetro el código del estudiante y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\notas_asistencias_alumnos.rdf&userid=sistemagm/sistemagm@sistema&codigo_est='||:codigo_estudiante || '&Período=' || :global.Período);

Por matriculados por Período

Recibe como parámetro el código de la asignatura, el código de la carrera, el paralelo y el período.

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\listado_paralelos.rdf&userid=sistemagm/sistemagm@sistema&c_asignatura='||:codigo_asignatura ||'&c_carrera='|| :codigo_carrera || '&c_paralelo='||:paralelo ||'&c_Período='|| :Período);

Reporte general por asignaturas

Recibe como parámetro el código de la carrera y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\reporte_notas_asistencias_matriculados_ciclo.rdf&userid=sistemagm/sistemagm@sistema&carrera=' || :global.carrera || '&Período='||:global.Período);

Notas por paralelos

Recibe como parámetro el código de la asignatura, el paralelo, y el Período actual

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\asignatura_notas_asistencias_alumnos.rdf&userid=sistemagm/sistemagm@sistema&asignatura='||:codigo_asignatura || '&paralelo='|| :paralelo ||'&Período='|| :global.Período);

Promedios por carrera

Recibe como parámetro el código de la carrera

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\promedio_general.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera);

Promedios de estudiantes

Recibe como parámetro el código del estudiante

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep

ortes\promedio_general_asignaturas_aprobadas.rdf&userid=sistemagm/sistemagm@sistema&c_estudiante='||:codigo_estudiante);

INTERNET

Horarios por Período

Recibe como parámetros la carrera y el Período actual

Web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=C:\latesis\Reportes\Horarios_por_ciclo_carrera.RDF&userid=sistemagm/sistemagm@sistema&carrera='||:global.carrera ||'&c_Período='|| :global.Período);

Notas y Asistencias

Record Academico

Recibe como parámetros la cédula del estudiante y el Período actual

Web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=C:\latesis\Reportes\Reporte_todas_las_asignaturas_alumno.RDF&userid=sistemagm/sistemagm@sistema&c_estudiante='|| :global.estudiante ||'&c_Período='|| :global.Período);

Materias Aprobadas

Recibe como parámetros la cédula del estudiante y el Período actual

Web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=C:\latesis\Reportes\Repor_Materias_Aprobadas_alumno_vida.RDF&userid=sistemagm/sistemagm@sistema&c_estudiante='|| :global.estudiante ||'&c_Período='|| :global.Período);

Por Período Académico

Recibe como parámetros la carrera, el Período del que se desee obtener el reporte, la cédula del estudiante y el Período actual

Web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=C:\latesis\Reportes\Reporte_asignaturas_ciclo_internet.RDF&userid=sistemagm/sistemagm@sistema&carrera1='||:codigo_carrera ||'&c_ciclo_aprobado='|| :Período_reporte || '&c_estudiante='||:codigo_estudiante ||'&c_Período_actual='|| :Período);

Asignaturas Matriculadas

Recibe como parámetros la cédula del estudiante y el Período actual

Web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=C:\latesis\Reportes\notas_Asistencias_alumnos.RDF&userid=sistemagm/sistemagm@sistema&codigo_est='||:global.estudiante || '&Período=' || :global.Período);

Créditos

Recibe como parámetros el código del estudiante del cual se quiere obtener los créditos acumulados en la carrera

web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\reportes\creditos_asignaturas.rdf&userid=sistemagm/sistemagm@sistema&cedula='||:global.estudiante);

Pagos Pendientes

Recibe como parámetros el Período actual, carrera y la cédula del estudiante.

```
web.show_document
('http://home:8888//reports/rwservlet?destype=cache&paramform=no&desformat=htmlcss&report=c:\latesis\rep
ortes\pagos_pendientes_individuales.rdf&userid=sistemagm/sistemagm@sistema&carrera='||:codigo_carrera||'&
cedula='||:codigo_estudiante || '&Período=' || :Período );
```

ÍNDICE

Índice

Índice