# Universidad Ricardo Palma

**FACULTAD DE INGENIERÍA**

**ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA INFORMÁTICA**



**SISTEMA DE REGISTROS Y ASISTENCIAS DE TUTORIAS EN LA FACULTAD DE INGENIERIA**

**Curso:** Base de Datos I

**Profesor:** Prof. Carpio Pedro

**Integrantes:**

* Miralles Delgado, José André *miralles.jose.18@gmail.com*
* Caja Rosas, Jhairo *jhairo49@gmail.com*
* Zurita Torvisco, Julio *julioffx@gmail.com*

**2017 – II**

**INTRODUCCIÓN**

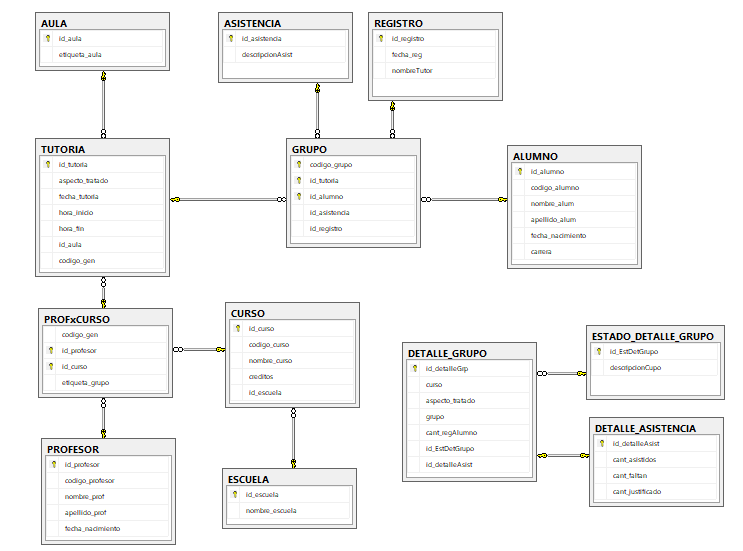
El presente trabajo se refiere en cuanto al registro y asistencia de tutorías entre los alumnos de la facultad de ingeniería, el desarrollo de este proyecto fue para en los casos de que los alumnos se registren con mayor facilidad a las tutorías; y de igual manera, los profesores abriendo tutorías en relación a los cursos que dictan.

Existen tres pasos del que se constituye el flujo principal:

1. El profesor asignado tiene la disponibilidad de abrir tutorías según el aspecto o tema a tratar. Una vez abra la tutoría, el sistema iniciara el estado de aquella clase como “Cupo abierto” hasta que alcance la cantidad máxima de alumnos; una vez se llene, el sistema cambiara de estado “Cupo abierto” a “Cupo cerrado”.
2. El alumno podrá registrar libremente la tutoría que desee asistir. Debido al apoyo lógico que brinda el sistema, el alumno podrá registrar uno o más tutorías donde cada uno ya tendrá predeterminadamente un horario fijo.
3. Existe ciertas tablas no relacionales, en los que contendrán datos estadísticos como la cantidad de alumnos registrados por cada tutoría y la cantidad de alumnos que asistieron, están en falta o justificaron sus ausencias.

El uso de los disparadores y de los procedimientos almacenados fueron de gran ayuda para soportar lógicamente todo los procesos, las entradas y salidas de los datos, y la estructura MER de la base de datos; así mismo, mencionando adicionalmente a los índices para mejorar la estructura de las tablas y las vistas para facilitar visualmente al programador.

**DIAGRAMA DE LA BASE DE DATOS**



La tabla “Tutoría” se compone entre las aulas asignadas, el aspecto por tratar, la fecha que se realizara, los horarios específicos, y el profesor por curso que haya registrado.

Al referirse como grupo es entre la tutoría y el alumno que se haya registrado por aquella tutoría, seguidamente el estado de asistencia: Presente, Falta, y Justificado; y el código único del registro (la tabla “Registro” es en relación al documento físico que se le da al docente en la escuela, por lo que por medio de esta tabla se generaría el documento PDF).

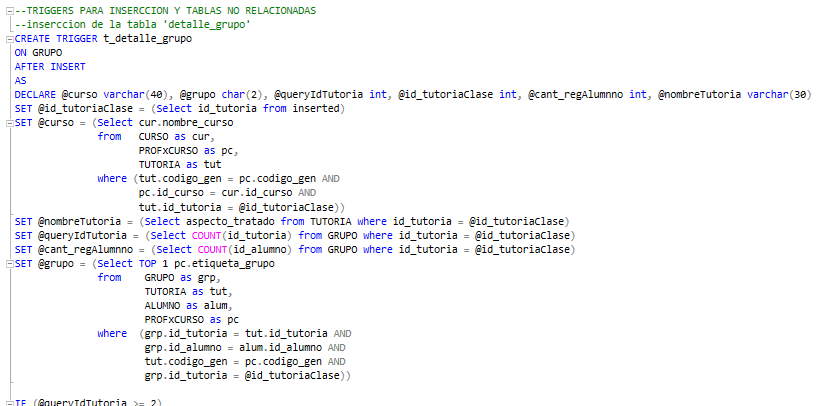
En la tabla “Detalle\_Grupo” se mostraría los datos estadísticos de cada tutoría registrada por el docente respectivamente, entre sus campos resultantes como: Cantidad de los alumnos registrados y, cantidad de los alumnos tomados por asistencia. También se sabe que esta tabla está en función exclusivamente entre cada grupo, por lo que si estuviera relacionado directamente con la tabla “Grupo” o “ProfxCurso”, no se mostraría correctamente los datos más relevantes para el usuario.

**DESCRIPCIÓN DE LOS OBJETOS/INSTRUCCIONES**

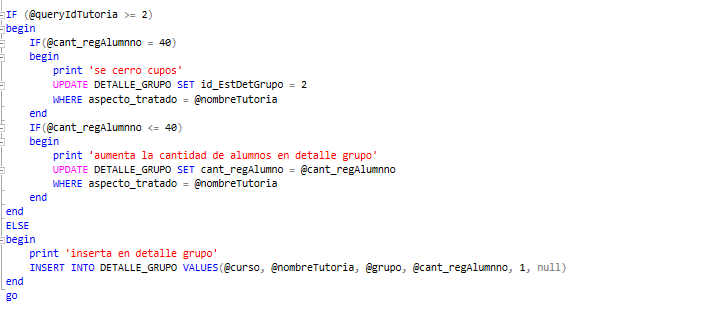
1. **DISPARADORES**

Los objetos que se verá a continuación tienen, como fin, soportar cualquier inconveniente en cuanto a la relación entre tablas o multiplicidad de datos al momento de retornar un valor de algún campo por medio de consultas e instrucciones.

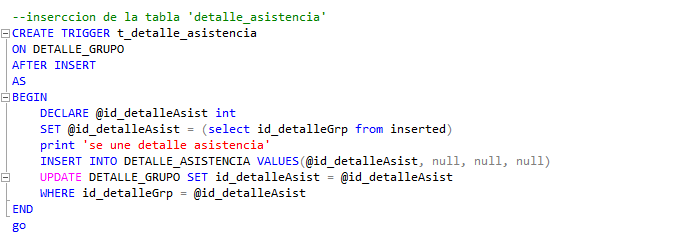
1. *T\_DETALLE\_GRUPO*

Lo primero es declarar variables que reciban como valor dichas consultas, donde su condición esté relacionado con otras variables, por lo que cada uno puede recibir como resultado una cadena o entero. Este método es de mucha utilidad.

La variable @queryIdTutoria obtiene la cantidad de filas en relación al id. de la tutoría de la tabla “Tutoría”. Si es menor que 2, entonces insertara aquel tutoría recién registrado; de lo contrario, procede en solo actualizar y contar el número de alumnos que vayan a registrar dentro de dicha tutoría (por cada alumno que registre tal tutoría, va sumando en el total de alumnos registrados de la clase). Además, mientras el número de alumnos registrados sea menor de 40, entonces el estado de tal grupo estará como “ABIERTO”; si sobrepasa el límite establecido, entonces cambia de “ABIERTO” a “CERRADO”.



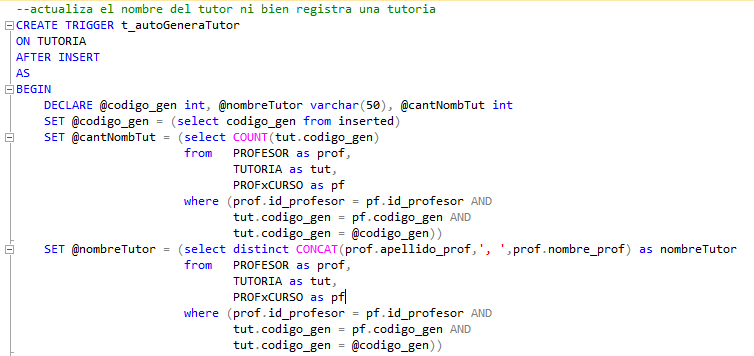
1. *T\_DETALLE\_ASISTENCIA*

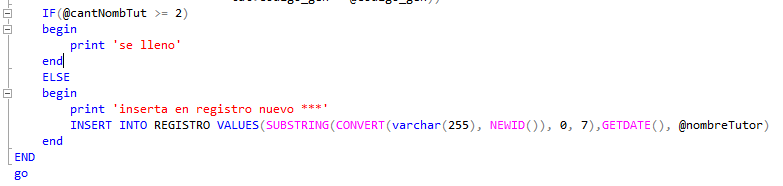
Este trigger se trata en relacionar automáticamente un grupo de la tabla “Detalle\_Grupo” a una fila nueva insertada de la tabla “Detalle\_Asistencia”, por lo que cumple exitosamente la relación 1:1 entre estas dos tablas.

1. *T\_AUTOGENERATUTOR*

Se trata en que el docente, ni bien inaugure una tutoria en especifico, se añade su documento de registro de tutorias en la tabla “Registro” bajo su nombre y apellido, y la fecha que comenzó. Para ser mas exactos, esta tabla se retroalimenta recursivamente de datos en funcion a la tabla “Tutoria”.

Se declara la variable @codigo\_gen para obtener ese campo en la tabla “Tutoria” despues de una insercción; con esta variable relacional podremos obtener varias consultas en funcion a la variable @codigo\_gen. Ademas, la variable @cantNombTut obtiene la cantidad de filas en relacion al codigo unico de la tabla “ProfxCurso”, donde podemos evaluar si hay mas de una fila en que un docente habra uno o mas tutorias.

Tambien se declara la variable @nombreTutor para obtener como resultado el apellido y el nombre concatenado, todo en relacion a la valor obtenido de @codigo\_gen.

Cuando @cantNombTut tenga como resultado 0 filas encontradas, entonces el sistema procede en insertar en la tabla “Registro”: Codigo alfanumero aleatorio de 6 caracteres, la fecha en el que recien registra su primera tutoria, y el valor obtenido de la variable @nombreTutor. Por otro lado, si la variable @cantNombTut supera el limite de la cantidad de filas encontradas, entonces el trigger solo mostrará como mensaje “se lleno”.

1. *T\_REGULAREGISTRO*

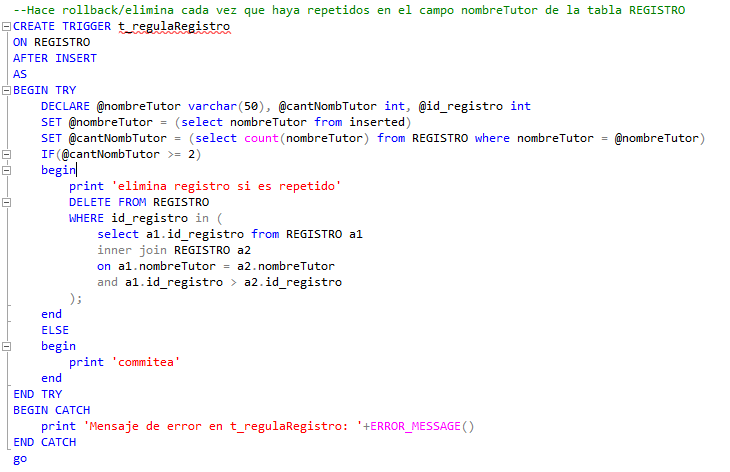
Una vez funcionando el trigger T\_AUTOGENERATUTOR, provocaba un error del que no nos habíamos percatado, y el problema surgía en base a la siguiente lógica:

*Cuando un Profesor es dedicado* ***a un curso*** *– abre 1 o más tutorías:*

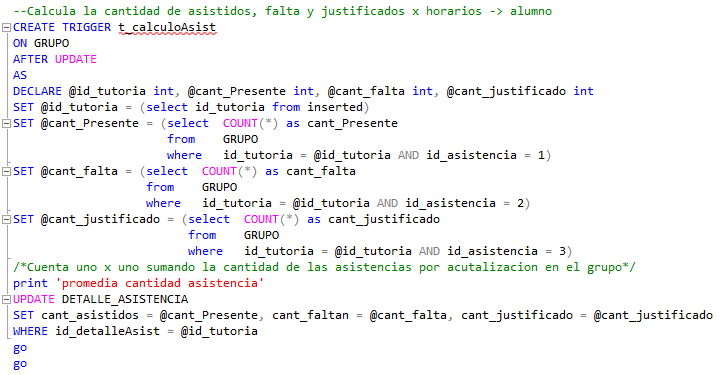
* *Resultado: Existirá una sola fila única del documento del tutor.*

*Cuando un Profesor es dedicado* ***a uno o más cursos*** *– abre 1 o más tutorías:*

* *Resultado: Error porque existirá dos o más filas del mismo tutor.*

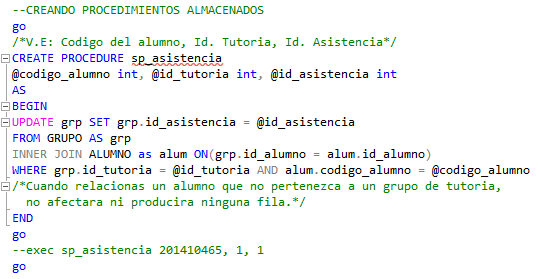
Por lo que era obligatoriamente necesario eliminar duplicados en la tabla “Registro” con el fin de que se pueda cumplir exitosamente el propósito de relacionar 1:m entre las tablas “Grupo” y “Registro”. La variable @nombreTutor obtiene el nombre del tutor registrado en la tabla “Registro”, la otra variable @cantNombTutor se usa para obtener la cantidad de filas en relación al valor de @nombreTutor.

1. *T\_CALCULOASIST*

En este trigger solamente se calcula la cantidad de alumnos que están registrados en un grupo. Se requiere de las tres variables presentes en la imagen, y con estas, por consultas, obtener la cantidad de filas en relación al valor del campo id\_asistencia.

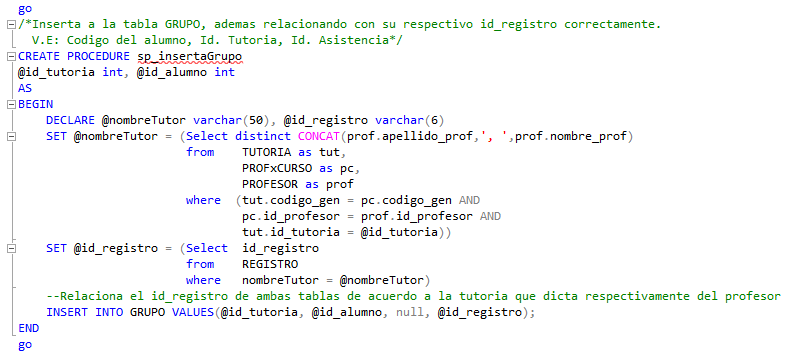
Consiguientemente, actualizar en la tabla “Detalle\_Asistencia” los valores obtenidos, obviamente todo en relación al valor registrado de la variable @id\_tutoria.

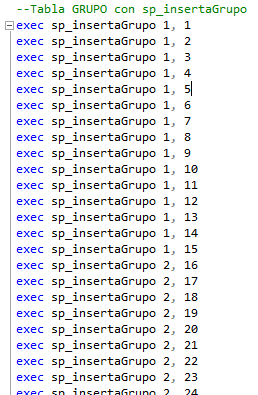
1. **PROCEDIMIENTOS ALMACENADOS**

* Se crea 3 variables de entrada, entre ellas: Código del alumno, id. de la tutoría y la id. de la asistencia (1🡪Presente, 2🡪Falta, 3🡪Justificado). Se trata cuando el tutor toma la asistencia a los alumnos al iniciar o acabando la clase, por lo que este SP es muy útil y reutilizable.
* Al principio, el problema surgía en como relacionar predeterminadamente un grupo insertado y a su vez, al campo id\_registro (en relación a la tabla “Registro”); una solución más que eficiente es crear un procedimiento almacenado en el que se usarían como variables de entrada: Id. de la tutoría y la id. del alumno.

Luego declarar la variable @nombreTutor que obtiene concatenado el apellido y nombre del docente en relación al valor obtenido de la variable de entrada @id\_tutoria, y la variable @id\_registro donde obtiene la id primaria de la tabla Registro” en relación al nombre del tutor.

Dicho esto, finalmente, se inserta con los valores que obtuvimos, en orden, en la tabla “Grupo”.

Este procedimiento almacenado es, a su vez, la solución más óptima para relacionar por cada documento que tenga un tutor, uno o más grupos registrados en esta.



1. **VISTAS**

Son todas las vistas creadas con el fin de facilitar visualmente al programador y a los usuarios que vean mediante el responsive del web.

/\*CREANDO VIEWS\*/

CREATE VIEW [dbo.ESCUELA] AS SELECT \* FROM ESCUELA

go

CREATE VIEW [dbo.CURSO] AS SELECT \* FROM CURSO

go

CREATE VIEW [dbo.PROFESOR] AS SELECT \* FROM PROFESOR

go

CREATE VIEW [dbo.PROFxCURSO] AS SELECT \* FROM PROFxCURSO

go

CREATE VIEW [dbo.TUTORIA] AS SELECT \* FROM TUTORIA

go

CREATE VIEW [dbo.ALUMNO] AS SELECT \* FROM ALUMNO

go

CREATE VIEW [dbo.GRUPO] AS SELECT \* FROM GRUPO

go

CREATE VIEW [dbo.REGISTRO] AS SELECT \* FROM REGISTRO

go

CREATE VIEW [dbo.AULA] AS SELECT \* FROM AULA

go

CREATE VIEW [v.DETALLExGRUPO]

AS

SELECT dg.curso, dg.aspecto\_tratado, dg.grupo, dg.cant\_regAlumno, edg.descripcionCupo,

da.cant\_asistidos, da.cant\_faltan, da.cant\_justificado

FROM DETALLE\_GRUPO as dg,

ESTADO\_DETALLE\_GRUPO as edg,

DETALLE\_ASISTENCIA as da

WHERE (dg.id\_EstDetGrupo = edg.id\_EstDetGrupo AND

dg.id\_detalleAsist = da.id\_detalleAsist)

go

CREATE VIEW [v.GRUPO]

AS

SELECT grp.codigo\_grupo, tut.aspecto\_tratado, tut.fecha\_tutoria,alum.codigo\_alumno,

CONCAT(alum.apellido\_alum,', ',alum.nombre\_alum) as alumno, pc.etiqueta\_grupo as grupo, grp.id\_asistencia, reg.nombreTutor, reg.fecha\_reg

FROM GRUPO as grp,

TUTORIA as tut,

ALUMNO as alum,

REGISTRO AS reg,

PROFxCURSO as pc

WHERE (grp.id\_tutoria = tut.id\_tutoria AND

grp.id\_alumno = alum.id\_alumno AND

grp.id\_registro = reg.id\_registro AND

tut.codigo\_gen = pc.codigo\_gen)

go

CREATE VIEW [v.TUTORIAxPROFESOR]

AS

SELECT tut.id\_tutoria, tut.fecha\_tutoria, esc.nombre\_escuela, tut.aspecto\_tratado,

CONCAT(pro.apellido\_prof,', ',pro.nombre\_prof) as profesor, cur.nombre\_curso, pc.etiqueta\_grupo,

CONCAT(tut.hora\_inicio,' - ',tut.hora\_fin) as horario, aula.etiqueta\_aula

FROM ESCUELA as esc,

CURSO as cur,

PROFESOR as pro,

PROFxCURSO as pc,

TUTORIA as tut,

AULA as aula

WHERE (tut.codigo\_gen = pc.codigo\_gen AND

pc.id\_profesor = pro.id\_profesor AND

pc.id\_curso = cur.id\_curso AND

cur.id\_escuela = esc.id\_escuela AND

tut.id\_aula = aula.id\_aula)

go

CREATE VIEW [v.REGISTRO]

AS

SELECT reg.id\_registro, reg.nombreTutor, reg.fecha\_reg, grp.codigo\_grupo, tut.fecha\_tutoria, esc.nombre\_escuela,

alum.codigo\_alumno, cur.nombre\_curso, pc.etiqueta\_grupo, tut.aspecto\_tratado,

CONCAT(alum.apellido\_alum,', ',alum.nombre\_alum) as estudiante

FROM REGISTRO as reg,

GRUPO as grp,

TUTORIA as tut,

ALUMNO as alum,

PROFxCURSO as pc,

PROFESOR as pro,

CURSO as cur,

ESCUELA as esc

WHERE (reg.id\_registro = grp.id\_registro AND

grp.id\_tutoria = tut.id\_tutoria AND

grp.id\_alumno = alum.id\_alumno AND

tut.codigo\_gen = pc.codigo\_gen AND

pc.id\_profesor = pro.id\_profesor AND

pc.id\_curso = cur.id\_curso AND

cur.id\_escuela = esc.id\_escuela)

go

go

1. **CONCLUSIÓN**

El sistema puede soportar cualquier inconveniente como: Multidatos, errores de sintaxis, errores mediante una instrucción SQL, errores al retornar más de un valor, y entre otros. Por lo que la base de datos cumple bajo cualquier expectativa en su uso, de igual manera, es exacto los cálculos que se hacen tanto para los datos estadísticos de la tabla “Detalle\_Grupo”, como también para la retroalimentación de la tabla “Registro”.