Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Laboratorio Sistemas de Bases de Datos 1 A
Ing. Luis Fernando Espino
Aux. Edin Emanuel Montenegro Vásquez

Enunciado

PROYECTO #2

Introducción

La integridad de los datos es un término usado para referirse a la exactitud y fiabilidad que estos adquieren. Los datos deben estar completos, sin variaciones o compromisos del original, que se considera confiable y exacto.

Al crear bases de datos, se debe prestar atención a la integridad de los datos y a cómo mantenerlos. Una buena base de datos hará cumplir la integridad de los datos siempre que sea posible.

Como parte del diseño es necesario crear procedimientos que permitan acceder de una manera más confiable a la base de datos, así como también garantizar que las transacciones se realicen correctamente.

Toda función, procedimiento almacenado y triggers, nos permiten poder administrar de una mejor manera cómo debe de comportarse una tabla de acuerdo con cada inserción, validación y disparador al momento de que ocurra cualquier evento o gestionemos nuestra base de datos con información nueva.

Objetivos

General:

- Adquirir nuevos conocimientos y destrezas en la gestión de una base de datos relacional para el uso profesional.

Específicos:

- Que el estudiante diseñe un sistema basado en un caso real. ○ Aprender a utilizar procedimientos almacenados, funciones y triggers.
- Aplicar las primeras formas normales para la elaboración de un modelo relacional más optimizado. ○ Asegurar la integridad relacional y aplicar prácticas de seguridad.

Descripción del problema

La facultad de Ingeniería ha decidido rediseñar su sistema de información que contiene al módulo de estudiantes, docentes, cursos y actas finales. De igual forma desea tener un mejor control en el almacenamiento de las actas de cada curso al finalizar un semestre o escuela de vacaciones.

Se le requiere a usted, como estudiante del curso de Sistemas de Bases de Datos 1, para brindar una solución inicial en fase beta, a nivel de base de datos para que usted sea el encargado de proponer, diseñar e implementar todo el flujo de los datos asegurando la persistencia e integridad de toda la información que se registre.

Se le dará la libertad de crear sus propios procedimientos, funciones y disparadores automatizados para el buen funcionamiento de la base de

datos de la facultad; pero respetando las principales funcionalidades que se le requieren.

Requerimientos del proyecto

Diseño:

Se debe de realizar todo el proceso de modelado de la base de datos necesario para su creación. Lo cual consistirá en:

1. Modelo conceptual
2. Modelo lógico
3. Modelo relacional (Diagrama ER)

Funcionalidades:

La administración de la facultad ha brindado todos los requerimientos funcionales de manera detallada para que el estudiante los pueda implementar correctamente. Los cuales se detallan a continuación:

1. Registrar estudiante

Se debe de almacenar los datos del estudiante, la carrera que cursa (se omitirán los casos de carreras simultáneas) y la cantidad de créditos que posee (al momento de crearlo por defecto es 0). También es necesario guardar la fecha exacta de su creación.

registrarEstudiante

Parámetro	Tipo de dato	Observación
Carnet	Bigint	
Nombres	String	
Apellidos	String	
Fecha de nacimiento	Fecha	
Correo	String	*Validar que sea un formato válido

Teléfono	Numérico	*Obviar código de área
Dirección	String	
Número de DPI	Bigint	
Carrera	Numérico	*Se refiere al identificador de la carrera

2. Crear carrera

Se debe de poder crear una carrera en caso de que no exista. El identificador primario de la carrera es de tipo numérico autoincremental.

crearCarrera

Parámetro	**Tipo de dato**	**Observación**
Nombre	String	*Validar que sólo sean letras

3. Registrar docente

Sirve para agregar un nuevo docente a la base de datos con su respectiva información. También se debe de guardar la fecha inicial en que es agregado al sistema. **Se debe validar que el docente no se haya creado.**

registrarDocente

Parámetro	**Tipo de dato**	**Observación**
Nombres	String	
Apellidos	String	
Fecha de nacimiento	Fecha	
Correo	String	*Validar que sea un formato válido
Teléfono	Numérico	*Obviar código de área
Dirección	String	
Número de DPI	Bigint	
Registro SIIF	Numérico	*Es el código de personal

4. Crear curso

Se debe de poder crear un curso en caso de que no exista.

crearCurso

Parámetro	Tipo de dato	Observación
Código	Numérico	*Se especifica el código del curso al momento de crearlo
Nombre	String	
Créditos necesarios	Numérico	*Validar que sea 0, o un entero positivo
Créditos que otorga	Numérico	*Validar que sea un entero positivo
Carrera a la que pertenece	Numérico	*Se refiere al identificador de la carrera, se coloca 0 si es área común.
Es obligatorio	Booleano	*0=no, 1=sí

5. Habilitar curso para asignación

Se utilizará para crear una nueva disponibilidad de un curso para que los estudiantes puedan asignarse. Se debe validar que la sección no se repita. Al momento de realizar la habilitación, se guarda automáticamente un atributo que indica el año actual y la cantidad de estudiantes asignados al curso que por defecto será 0.

Se debe generar un identificador autoincremental de los cursos habilitados.

habilitarCurso

Parámetro	Tipo de dato	Observación
Código del curso	Numérico	*Se debe validar que el curso exista
Ciclo	String	*Solamente puede aceptar los siguientes valores: '**1S**', '**2S**', '**VJ**', '**VD**'
Docente	Numérico	*Identificador de quien impartirá
Cupo máximo	Numérico	*Validar que sea un número entero positivo
Sección	Caracter	*Una letra y guardarla en mayúscula

6. Agregar un horario de curso habilitado

Sirve para agregar los distintos días y horarios en que se impartirá el curso que ya se encuentra habilitado. Si no se encuentra el id debe retornar error.

agregarHorario

Parámetro	Tipo de dato	Observación
Id de curso habilitado	Numérico	*Es el id generado cuando se habilitó un curso
Día	Numérico	*Debe ser dentro del dominio [1,7] correspondiente al día de la semana.
Horario	String	*Por ejemplo: "9:00-10:40"

7. Asignación de curso

Se realiza la asignación de un estudiante a determinado curso. Se debe validar que no se encuentre ya asignado a la misma u otra sección, que cuente con los créditos necesarios y que pertenezca a un curso correspondiente a su carrera o área común, también validar que la sección que elige el estudiante sí existe y que no ha alcanzado el cupo máximo, de lo contrario mostrar una explicación del error.

asignarCurso

Parámetro	Tipo de dato	Observación
Código de curso	Numérico	*Es el código como tal del curso (no del habilitado) Se debe hacer match con la relación de curso habilitado por medio del año actual, ciclo y sección.
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Sección	Caracter	
Carnet	Bigint	*Validar que el carnet exista

8. Desasignación de curso

Sirve para desasignar al estudiante del curso, por lo que es necesario validar que el estudiante ya se encontraba asignado a esa sección, se debe de llevar un control de cada desasignación y asegurarse de que el cupo no se siga viendo reducido puesto que habría un cupo más para otro estudiante.

desasignarCurso

Parámetro	Tipo de dato	Observación
Código de curso	Numérico	*Es el código como tal del curso (no del habilitado) Se debe hacer match con la relación de curso habilitado por medio del año actual, ciclo y sección.
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Sección	Caracter	
Carnet	Bigint	*Validar que el carnet exista

9. Ingresar notas

Se utiliza cuando el docente ingresa notas finales por cada estudiante del curso al finalizar el semestre o escuela de vacaciones. Debería existir una entidad Notas que almacene los atributos necesarios, el año se toma del año actual. La nota puede venir con decimales, pero se debe aplicar un redondeo al entero más próximo. Si el estudiante aprobó el curso (Nota >= 61) entonces se suma la cantidad de créditos que posee el estudiante con la cantidad de créditos que otorga el curso aprobado.

ingresarNota

Parámetro	Tipo de dato	Observación
Código de curso	Numérico	*Es el código como tal del curso.
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Sección	Caracter	
Carnet	Bigint	*Validar que el carnet exista
Nota	Numérico	*Validar que sea positivo

10. Generar acta

Al momento de que el docente termina de ingresar notas se genera un acta, por lo que es necesario hacer la validación de que ya ingresó las notas de todos los estudiantes asignados, de lo contrario mostrar un error. Se debe de almacenar la fecha y hora exacta en que se generó el acta.

generarActa

Parámetro	Tipo de dato	Observación
Código de curso	Numérico	*Es el código como tal del curso.
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Sección	Caracter	

Procesamiento de datos:

La administración de la facultad solicita que se extraiga la información necesaria a través de los datos almacenados con el fin de poder generar reportes y constancias necesarias para los distintos procesos administrativos y de los estudiantes.

Para ello es necesario elaborar los siguientes procedimientos:

1. Consultar pensum

Debe retornar un listado de todos los cursos pertenecientes a una carrera (incluir los de área común y área profesional).

consultarPensum

Parámetro	Tipo de dato	Observación
Código de carrera	Numérico	*Es el id

Resultado esperado:

- ✓ Código del curso
- ✓ Nombre del curso
- ✓ Es obligatorio (sí o no)
- ✓ Créditos necesarios

2. Consultar estudiante

Debe retornar la información de un estudiante según su carnet.

consultarEstudiante

Parámetro	Tipo de dato	Observación
Carnet	Bigint	*Si no existe mostrar error

Resultado esperado:

- ✓ Carnet
- ✓ Nombre completo (concatenado)
- ✓ Fecha de nacimiento
- ✓ Correo
- ✓ Teléfono
- ✓ Dirección
- ✓ Número de DPI
- ✓ Carrera
- ✓ Créditos que posee

3. Consultar docente

Debe retornar la información de un docente según su código de personal.

consultarDocente

Parámetro	Tipo de dato	Observación
Registro SIIF	Bigint	*Si no existe mostrar error

Resultado esperado:

- ✓ Registro SIIF
- ✓ Nombre completo
- ✓ Fecha de nacimiento
- ✓ Correo
- ✓ Teléfono
- ✓ Dirección
- ✓ Número de DPI

4. Consultar estudiantes asignados

Debe retornar el listado de estudiantes asignados al curso y sección. Si no se encuentra debe mostrar un error.

consultarAsignados

Parámetro	**Tipo de dato**	**Observación**
Código de curso	Numérico	*Si no existe mostrar error
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Año	Numérico	*Si es un año que no posee registros no se retorna nada
Sección	Caracter	*En mayúscula

Resultado esperado:

- ✓ Carnet
- ✓ Nombre completo
- ✓ Créditos que posee

5. Consultar aprobaciones

Debe retornar el listado con el carnet del estudiante, y si aprobó o reprobó el curso (nota de aprobación = 61).

consultarAprobacion

Parámetro	**Tipo de dato**	**Observación**
Código de curso	Numérico	*Si no existe mostrar error
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Año	Numérico	*Si es un año que no posee registros no se retorna nada
Sección	Caracter	*En mayúscula

Resultado esperado:

- ✓ Código de curso (se repite en cada fila)
- ✓ Carnet
- ✓ Nombre completo
- ✓ "APROBADO" / "DESAPROBADO"

6. Consultar actas

Debe retornar el listado de actas de un determinado curso y ordenarlo por fecha y hora de generado.

consultarActas

Parámetro	Tipo de dato	Observación
Código de curso	Numérico	*Si no existe mostrar error

Resultado esperado:

- ✓ Código de curso (se repite en cada fila)
- ✓ Sección
- ✓ Ciclo, se debe de traducir según sea el caso: "PRIMER SEMESTRE" / "SEGUNDO SEMESTRE" / "VACACIONES DE JUNIO" / "VACACIONES DE
  - DICIEMBRE"
- ✓ Año
- ✓ Cantidad de estudiantes que llevaron el curso (o cantidad de notas que fueron ingresadas)
- ✓ Fecha y hora de generado

7. Consultar tasa de desasignación

Debe retornar el porcentaje de desasignación de un curso y sección. Esto sirve para conocer qué curso y sección está teniendo más problemas.

consultarDesasignacion

Parámetro	Tipo de dato	Observación
Código de curso	Numérico	*Si no existe mostrar error
Ciclo	String	*Puede ser '**1S**', '**2S**', '**VJ**', '**VD**'.
Año	Numérico	*Si es un año que no posee registros no se retorna nada
Sección	Caracter	*En mayúscula

Resultado esperado:

- ✓ Código de curso
- ✓ Sección
- ✓ Ciclo, se debe de traducir según sea el caso: "PRIMER SEMESTRE" / "SEGUNDO SEMESTRE" / "VACACIONES DE JUNIO" / "VACACIONES DE
    - DICIEMBRE"
- ✓ Año
- ✓ Cantidad de estudiantes que llevaron el curso ❼ Cantidad de estudiantes que se desasignaron
- ✓ Porcentaje de desasignación

Historial de transacciones

Se debe llevar un control de transacciones por medio de una tabla adicional en donde se registrará automáticamente (por medio de triggers) cada vez que ocurra una inserción, modificación o eliminación en una tabla. Se debe de almacenar la fecha y hora exacta, y el nombre de la tabla que se vio afectada.

Tabla Ejemplo:

Fecha	Descripción	Tipo
YYYY-MM-DD hh:mm:ss	Se ha realizado una acción en la tabla "Nombre".	INSERT/UPDATE/DELETE

Entregables

- Modelos conceptual, lógico y relacional
- Script de la base de datos y datos de carga
- Funciones, procedimientos y triggers implementados
- Para poder calificarse el estudiante debe cargar datos de ejemplo en sus tablas, al menos deben ser:
  - 4 carreras
  - 5 docentes
  - 10 estudiantes
  - 5 cursos por cada carrera y 5 de área común

Restricciones

- El gestor de base de datos a utilizar puede ser **MySQL**, **Oracle** o **SQL Server**.
- Use el nombre dado para cada función o procedimiento con la misma notación que se tiene "camelCase"
- El proyecto es individual.
- Sistema operativo libre.
- Pueden utilizar Docker o una instancia local para su base de datos.
- Todo puede ser presentado desde su gestor nativo o IDE.
- El código y todo lo relacionado a su proyecto debe alojarse en un repositorio de **GitHub** agregando a su auxiliar como colaborador. No se permiten ediciones o modificaciones después de la entrega.
- En caso de copias totales o parciales será anulado y reportado a escuela.

Entrega

- Fecha límite: domingo 29 de octubre de 2023 hasta las 23:59 hrs.
- La entrega se hará vía UEDi por medio del enlace a su repositorio.