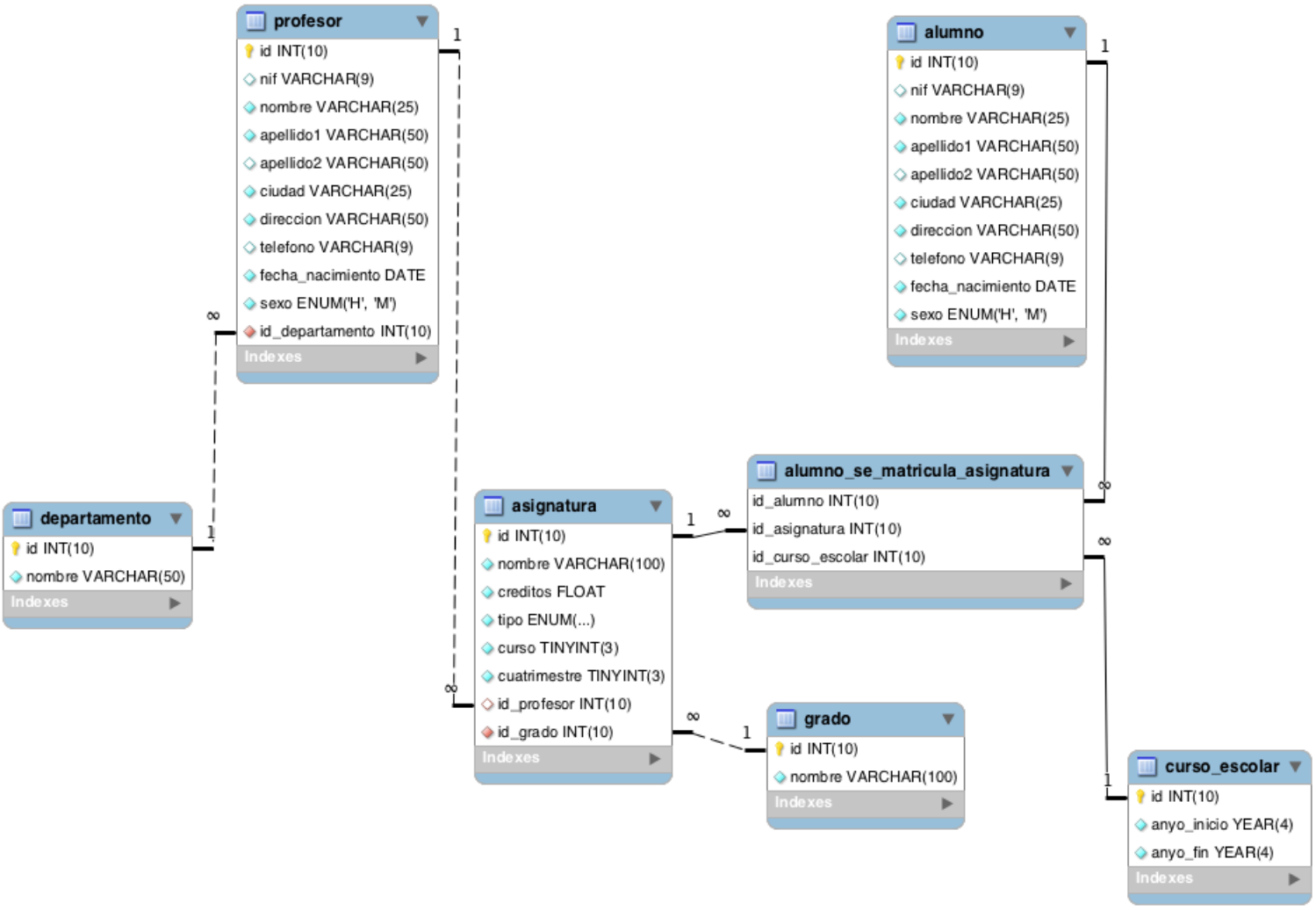
Práctico de SQL SERVER

Notas:

a. Todo se encontrará dentro del repositorio de GitHub (aquí).
b. Formato de entrega: Libre. No me interesa si es un .txt o un .sql o un powerpoint, el tema es que esté ordenado. Tienen que considerar que es una entrega de la cual, si se me facilita el código, se suma practicidad.
c. Se aprecia el orden e identación en el código.
d. Si creen que deben agregar comentarios, por mi mejor. Los comentarios son parte del código, ya que en un futuro no serán los únicos en leer un código.
e. Considerar buenas prácticas. Controles de errores, o aperturas y finalización de transacciones. Queda a su criterio.
f. Leer todo el práctico antes de comenzar.

Parte 1

1. Crear una base de datos para poder alojar la siguiente estructura. El nombre será a elección de cada uno en base a la interpretación del modelo. NO CREAR TABLAS AÚN.



2. Crear 3 usuarios/logins con las siguientes características:

Usuario	Login	Contraseña	Esquema	Rol/Permisos
User_01	User_01	A criterio propio	Dbo Esquema_01	Heredar el rol de administrador.
User_02	User_02	A criterio propio	Esquema_02	Heredar rol de Lectura y Escritura
User_03	User_03	A criterio propio	Esquema_01 Esquema_03	Dar permisos de creación de vistas y lectura a todas las tablas.

3. Crear todas las tablas en el esquema **Esquema_01**. Tener en cuenta qué usuario se usará.
4. Insertar los datos desde el archivo *Data_SQLServer.sql* en la carpeta *Parte 1*.
5. Insertar al menos 10 alumnos a la tabla **alumno**.
6. Actualizar los campos Varchar en la tabla alumno, quitando los acentos. Ejemplo: Nicolás debería de ser Nicolas.

Parte 2

1. Devuelve todos los datos del alumno más joven.
2. Devuelve un listado con los profesores que no están asociados a un departamento.
3. Devuelve un listado con los departamentos que no tienen profesores asociados.
4. Devuelve un listado con los profesores que tienen un departamento asociado y que no imparten ninguna asignatura.
5. Devuelve un listado con las asignaturas que no tienen un profesor asignado.
6. Devuelve un listado con todos los departamentos que no han impartido asignaturas en ningún curso escolar.
7. Devuelve un listado con los nombres de todos los profesores y los departamentos que tienen vinculados. El listado también debe mostrar aquellos profesores que no tienen ningún departamento asociado. El listado debe devolver cuatro columnas, nombre del departamento, primer apellido, segundo apellido y nombre del profesor. El resultado estará ordenado alfabéticamente de menor a mayor por el nombre del departamento, apellidos y el nombre.
8. Devuelve un listado con todos los departamentos que tienen alguna asignatura que no se haya impartido en ningún curso escolar. El resultado debe mostrar el nombre del departamento y el nombre de la asignatura que no se haya impartido nunca.
9. Eliminar aquellos profesores que hayan nacido en 1979. Tener en cuenta las relaciones.
10. Eliminar a la alumna “Sonia Gea Ruiz”. Tener en cuenta las relaciones.

Parte 3

1. Crear una vista para que un usuario X (no es necesario crearlo) pueda obtener una lista de todos los alumnos que cursaron la asignatura “Álgegra lineal y matemática discreta”.
2. Crear una tabla en el esquema de dbo que sea idéntica a la tabla alumno (sin FKs), e insertar los datos de *Data_SQLServer.sql*.
3. Aplicar un método que, actualice esta “nueva tabla” cuando el nombre, los apellidos, la ciudad hayan cambiado, inserte cuando el registro sea nuevo, y borre aquellos que no estén en la tabla **alumno** (original).
4. Colocar índices personalizados dentro las tablas: **alumno**, **profesor** y **asignatura**. Esto es para manejo propio del criterio de cada uno.
5. Crear una tabla donde se encontrarán las notas finales. Debe contener: 3 notas principales (pueden ser aleatorias), el id de los alumnos, la asignatura para la cual fue tomada y el año escolar. La nota final será un promedio de las 3 notas principales.

6. Crear un proceso que cargue la tabla anterior con cada alumno, cada asignatura, y cada año escolar. Se puede usar cursores.
7. Crear un proceso que actualice la nota de un alumno.
8. Agregar a la tabla en donde se encontrarán las notas finales, una columna de nota del coloquio. Esta columna puede ser nula. Esto último dependerá del criterio de quien haga la columna.
9. Crear una vista de alumnos desaprobados. Usen el criterio que les guste >= 4 o >= 6 aprobados.
10. Realizar un proceso que reciba por parámetro la nota de coloquio y el alumno, y que sea insertado sobre la misma tabla.

Parte 4 - ¿Qué medidas tomarán?

Esta sección está para atender **problemas de usuarios**. Deben escribir qué medidas tomarán frente a problemas que tengan usuarios y en algunos casos colocar la query SQL.
Nota de real importancia: no a todos los usuarios les interesan los IDs. Si deben devolver algo, ver de entregarlo lo más similar a un reporte, pero por salida de SQL.

1. Como usuario 3, quiero ver un listado de todos los profesores y sus asignaturas.
2. Como usuario 2, quiero poder crear una tabla en la base de datos actual que me tenga todos los alumnos de la promoción 2010. Este usuario solo sabe consultar a la base de datos, y su conocimiento técnico de la misma, es mínimo.
3. Entra el usuario Pepito y necesita poder consular sobre la base de datos actual.
4. El gerente de la empresa le da los privilegios a Pepito y ahora puede ejecutar store procedures dentro de la base de datos.
5. Se le han denegado los permisos a Pepito y se necesita eliminar este usuario de la base de datos. Ordenes directivas.
6. Entra alguien del ministerio de educación y le tenemos que dar un usuario que pueda ver en una sola consulta los alumnos aprobados, el año, y las asignaturas.
7. El usuario se queja porque las consultas sobre las tablas son lentas.
8. El usuario 4 (no está en la tabla de arriba), se queja porque no puede ver el resultado de la consulta a los alumnos del modelo, y no hay problemas en la consulta SQL.