Laboratorio 3: Consultas Avanzadas en SQL

Objetivo: Escribir consultas avanzadas en SQL.

Guía de Trabajo

A. Abra la aplicación SQL Server Management Studio. En la ventana de conexión debe seleccionar las opciones 'Database Engine' y 'SQL Authentication' y escribir la dirección IP 172.16.202.209 en el campo 'Nombre del servidor'.

B. Busque la base de datos llamada BD_Universidad, dé click derecho en ella y seleccione la opción 'New Query' del menú contextual. Esto les abre una nueva ventana lista para que escriban consultas sobre esta base de datos. Recuerden escribir al inicio de la consulta "`use <Nombre de la BD>;`".

C. Escriba las siguientes consultas en SQL, tomando como referencia las Figuras 1 y 2 de esta guía. En todas estas consultas es necesario unir la información de más de dos tablas. En MS SQL Server, un *join* de tres tablas tiene la siguiente forma general:

```
SELECT * FROM tabla1 t1 JOIN tabla2 t2 ON t1.a = t2.b JOIN tabla3 t3 ON t3.c = t2.c
```

1. Recupere el nombre y primer apellido de los asistentes, y el nombre de los cursos que han asistido. Si un asistente ha asistido varias veces un mismo curso, el curso sólo debe aparecer una vez en el resultado.
2. Para el estudiante llamado "Gabriel Sánchez", liste el expediente académico (incluyendo sigla del curso, número de grupo, semestre, año y nota) de los cursos que ha matriculado, y el nivel del plan de estudios en el que está cada uno de los cursos aprobados. El listado debe ordenarse por nivel del plan de estudios, y luego por sigla.

D. Escriba las siguientes consultas en SQL, tomando como referencia las Figuras 1 y 2 de esta guía. Para estas consultas es necesario agrupar la información de la(s) tabla(s). En MS SQL Server, para agrupar datos se usa la cláusula GROUP BY. Algunas veces se requiere filtrar sobre los grupos creados, para lo cual se usa la cláusula HAVING (funciona de forma similar al WHERE, pero en lugar de filtrar tuplas/registros, el HAVING filtra grupos después de aplicar el GROUP BY). También es usual combinar el agrupamiento de datos con funciones de agregación, tales como max, count, sum, etc.

3. Recupere la cantidad de profesores por grado académico (título). Ordene el resultado por cantidad de profesores, de mayor a menor.
4. Calcule el promedio de notas de cada estudiante. El reporte debe listar la cédula del estudiante en la primera columna y su promedio en la segunda columna, ordenado por cédula.

5. Para cada proyecto de investigación, obtenga el total de carga asignada al proyecto (la suma de la carga de cada uno de sus participantes).
6. Para aquellos cursos que pertenecen a más de 2 carreras, recupere la sigla del curso y la cantidad de carreras que tienen ese curso en su plan de estudios.
7. Para todas las facultades, recupere el nombre de la facultad y la cantidad de carreras que posee. Si hay facultades que no poseen escuelas o carreras, deben salir en el listado con cero en la cantidad de carreras. Ordene descendentemente por cantidad de carreras (de la facultad que tiene más carreras la que tiene menos).
8. Liste la cantidad de estudiantes matriculados en cada grupo de cursos de computación (sigla inicia con el prefijo “CI”). Se debe mostrar el número de grupo, la sigla de curso, el semestre y el año de cada grupo, además de la cantidad de estudiantes matriculados en él. Si hay grupos que no tienen estudiantes matriculados, deben salir en el listado con cero en la cantidad de estudiantes. Ordene por año, luego por semestre, y finalmente por sigla y grupo.
9. Liste los grupos (identificados por sigla de curso, número de grupo, semestre y año) donde la nota mínima obtenida por los estudiantes fue mayor o igual a 70 (es decir, todos los estudiantes aprobaron). Muestre también la nota mínima, máxima, y promedio de cada grupo en el resultado. Ordene el resultado descendentemente por el promedio de notas del grupo.

E. Envíen su trabajo a través de la plataforma vitual del curso. Para el reporte deben subir el archivo *sql*. Indiquen claramente, mediante comentarios, el número de consulta a la que corresponde cada comando SQL. Verifique que el script se ejecute sin errores.

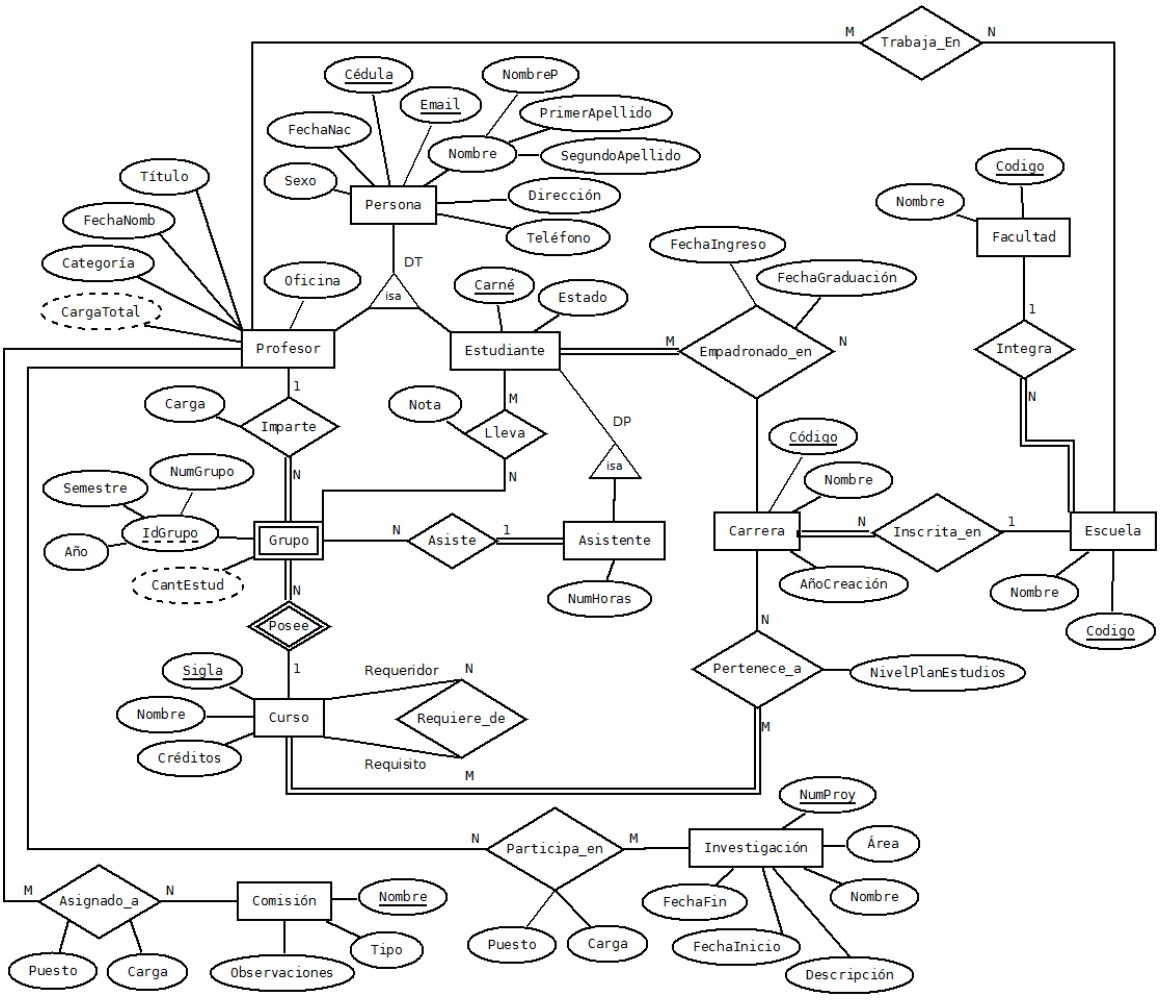

Figura 1. Diagrama ER de la base de datos **Universidad**.

1. **ESTUDIANTE** (Cédula, Email, NombreP, Apellido1, Apellido2, Sexo, FechaNac, Dirección, Teléfono, Carné, Estado)
2. **PROFESOR** (Cédula, Email, NombreP, Apellido1, Apellido2, Sexo, FechaNac, Dirección, Teléfono, Categoría, FechaNomb, Título, Oficina)
3. **ASISTENTE** (Cédula, NumHoras)
   FK(Estudiante)
4. **CURSO** (Sigla, Nombre, Créditos)
5. **GRUPO** (SiglaCurso, NumGrupo, Semestre, Año, CedProf, Carga, CedAsist)
   FK(Curso) FK(Profesor) FK(Asistente)
6. **LLEVA** (CedEstudiante, SiglaCurso, NumGrupo, Semestre, Año, Nota)
   FK(Estudiante) FK(Grupo)
7. **REQUIERE_DE** (SiglaCursoRequeridor, SiglaCursoRequisito)
   FK(Curso) FK(Curso)
8. **FACULTAD** (Código, Nombre)
9. **ESCUELA** (Código, Nombre, CodFacultad)
   FK(Facultad)
10. **CARRERA** (Código, Nombre, AñoCreación, CodEscuela)
    FK(Escuela)
11. **PERTENECE_A** (SiglaCurso, CodCarrera, NivelPlanEstudios)
    FK(Curso) FK(Carrera)
12. **EMPADRONADO_EN** (CedEstudiante, CodCarrera, FechaIngreso, FechaGraduación)
    FK(Estudiante) FK(Carrera)
13. **INVESTIGACION** (NumProy, Nombre, Área, Descripción, FechaInicio, FechaFin)
14. **COMISION** (Nombre, Tipo, Observ)
15. **PARTICIPA_EN** (CedProf , NumProy, Puesto, Carga)
    FK(Profesor) FK(Investigacion)
16. **ASIGNADO_A** (CedProf , NombComision, Puesto, Carga)
    FK(Profesor) FK(Comision)
17. **TRABAJA_EN** (CedProf , CodEscuela)
    FK(Profesor) FK(Escuela)

Figura 2. Esquema relacional de la BD Universidad.