Universidad Nacional Autónoma de México
Facultad de Ciencias

Fundamentos de Bases de Datos

Práctica 4

Consultas Avanzadas y Joins

Autor: Arroyo Martínez Erick Daniel

Introducción

En esta práctica, los estudiantes se enfocarán en la realización de consultas avanzadas utilizando `joins` para combinar datos de múltiples tablas en una base de datos relacional. Las consultas requerirán el uso de funciones de agregación, alias, órdenes de clasificación, concatenación de campos, subconsultas y operadores lógicos. El propósito de esta práctica es fortalecer la capacidad de los estudiantes para trabajar con esquemas relacionales complejos y manipular datos de manera eficiente mediante SQL.

Objetivos

1. Desarrollar la habilidad para realizar consultas avanzadas que involucren la combinación de datos de múltiples tablas utilizando `joins`.
2. Aplicar funciones de agregación, alias, órdenes de clasificación, y concatenación de campos en consultas SQL.
3. Manejar subconsultas y operadores lógicos para realizar consultas más complejas y precisas.
4. Abstraer y representar adecuadamente las relaciones, atributos y entidades de un esquema relacional basado en una descripción funcional.

Especificaciones de Desarrollo

El esquema que deberán implementar y consultar los estudiantes está basado en un sistema de gestión de proyectos en una empresa. La base de datos debe manejar la información de empleados, departamentos, proyectos, y asignaciones de empleados a proyectos. A continuación se describen los requisitos funcionales del esquema:

- La base de datos debe incluir una tabla que almacene la información de los **Empleados**, incluyendo atributos como el `employee_id`, `first_name`, `last_name`, `email`, `hire_date`, y `salary`. Cada empleado debe estar asignado a un **Departamento**.
- Los **Departamentos** deben ser gestionados en una tabla separada, con atributos como `department_id`, `department_name`, y `manager_id`. Cada departamento tiene un gerente, que es un empleado, y puede tener múltiples empleados asignados.
- Los **Proyectos** en la empresa deben ser gestionados en otra tabla, que incluye atributos como `project_id`, `project_name`, `start_date`, `end_date`, y `budget`. Cada proyecto puede estar asociado a múltiples empleados, y cada empleado puede estar asignado a múltiples proyectos.

- Las **Asignaciones de Proyectos** deben ser gestionadas en una tabla que relacione a los empleados con los proyectos a los que están asignados. Esta tabla debe incluir atributos como `assignment_id`, `employee_id`, `project_id`, `role`, y `hours_worked`.
- Se deben implementar relaciones de clave foránea entre las tablas, como la relación entre `employee_id` en la tabla de `Empleados` y `manager_id` en la tabla de `Departamentos`, y entre `department_id` en la tabla de `Empleados` y la tabla de `Departamentos`. También debe existir una relación entre `employee_id` y `project_id` en la tabla de `Asignaciones de Proyectos`.
- El esquema debe permitir la extracción de información agregada, como el salario promedio de los empleados en cada departamento, el número total de horas trabajadas por empleado en cada proyecto, y el presupuesto total asignado a proyectos activos.
- El sistema debe permitir el manejo de **Proveedores**, que son entidades externas involucradas en los proyectos. Los proveedores tienen atributos como `supplier_id`, `supplier_name` y `contact_info`. Cada proveedor puede estar relacionado con múltiples proyectos, pero cada proyecto solo tiene un provedor.
- Se debe manejar la relación entre **Clientes** y proyectos. Cada cliente, almacenado en una tabla con atributos como `customer_id`, `customer_name`, `contact_info`, y `project_id`, puede estar relacionado con uno o más proyectos, y cada proyecto puede estar relacionado con múltiples clientes.

Con base en esta descripción, los estudiantes deben:

- Abstraer las entidades (Identificadores en inglés), relaciones y cardinalidades.
- Identificar los atributos y las claves primarias y foráneas.
- Crear el esquema relacional completo en SQL.
- Ejecutar consultas avanzadas que involucren múltiples tablas y apliquen los conceptos de `joins`, funciones de agregación, alias, y operadores lógicos.

Consultas Avanzadas

A continuación, se presentan doce consultas avanzadas que deben ser implementadas sobre el esquema relacional desarrollado en la práctica. Cada consulta está diseñada para reforzar los conceptos de `joins`, funciones de agregación, alias, y otros aspectos complejos de SQL.

1. **Consulta 1**: Obtener el nombre completo (`first_name`, `last_name`) y el salario de los empleados junto con el nombre del departamento al que pertenecen. Ordenar los resultados por el nombre del departamento y luego por el apellido del empleado.
2. **Consulta 2**: Calcular el salario promedio, el salario mínimo y el salario máximo de los empleados en cada departamento. Mostrar el nombre del departamento y los valores calculados, usando alias para las columnas agregadas.
3. **Consulta 3**: Listar todos los proyectos activos (aquellos cuyo `end_date` es posterior a la fecha actual) junto con el número total de empleados asignados a cada proyecto. Utilizar `joins` y una subconsulta para filtrar los proyectos activos.
4. **Consulta 4**: Obtener el nombre de los empleados que trabajan en más de un proyecto, junto con la cantidad de proyectos en los que están involucrados. Los resultados deben estar ordenados por el número de proyectos en orden descendente.

5. **Consulta 5**: Mostrar el nombre del gerente (`manager_id`) y el nombre del departamento que gestiona, junto con el número total de empleados en cada departamento. Considerar solo aquellos departamentos que tienen más de 5 empleados.

6. **Consulta 6**: Obtener una lista de los proyectos que tienen un presupuesto superior al promedio de todos los proyectos en la empresa. Mostrar el `project_name`, `budget`, y el `start_date`.

7. **Consulta 7**: Concatenar el `first_name` y `last_name` de los empleados en un solo campo, junto con su `employee_id` y el nombre del departamento al que pertenecen. Los resultados deben estar ordenados por el nombre completo del empleado.

8. **Consulta 8**: Listar los empleados que no tienen asignado ningún proyecto actualmente. Mostrar el `employee_id`, `first_name`, `last_name`, y el nombre del departamento. Usar una subconsulta para identificar a los empleados sin proyectos.

9. **Consulta 9**: Obtener el nombre y el contacto de los proveedores (`suppliers`) que están asociados con proyectos en los que trabaja al menos un empleado del departamento de `IT`. Usar un `join` para combinar la información de proveedores, proyectos, y empleados.

10. **Consulta 10**: Listar los nombres de los clientes que están asociados con más de un proyecto. Mostrar el `customer_name`, el número de proyectos asociados y el `contact_info` del cliente.

11. **Consulta 11**: Mostrar los nombres de los proyectos que tienen más de 100 horas trabajadas en total, junto con el nombre del cliente asociado y el número total de horas trabajadas. Usar `joins` y funciones de agregación.

12. **Consulta 12**: Obtener una lista de todos los empleados cuyo salario es superior al salario promedio de su departamento. Mostrar el `employee_id`, `first_name`, `last_name`, `salary`, y el nombre del departamento. Ordenar los resultados por el salario en orden descendente.

Entregables

Para completar esta práctica, los estudiantes deben entregar los siguientes elementos:

1. Un archivo `.sql` que contenga:
   - La implementación de cada una de las doce consultas avanzadas solicitadas.
   - Comentarios que expliquen brevemente la lógica detrás de cada consulta.
2. Un archivo `.sql` adicional que contenga el esquema relacional creado, incluyendo las instrucciones para crear las tablas y las relaciones entre ellas. Este archivo debe contener únicamente las instrucciones de creación de esquema, sin datos insertados.
3. Todo el material debe ser entregado en un archivo comprimido (`.zip` o `.tar.gz`) que incluya los archivos mencionados arriba, nombrado con el formato `Equipo_N_Practica4.zip`.

Rúbrica de Evaluación

La evaluación de la práctica se realizará de acuerdo con los siguientes criterios:

1. **Implementación correcta de las consultas avanzadas** (40 %):
   - Cada consulta correctamente implementada recibirá una puntuación proporcional a la dificultad y la precisión requerida.

- Se evaluará la correcta utilización de `joins`, funciones de agregación, alias, subconsultas y operadores lógicos.

2. **Documentación y comentarios en el código SQL** (20 %):
   - Se evaluará la claridad y la pertinencia de los comentarios en el archivo `.sql`.
   - La justificación escrita en el `.pdf` debe ser clara, concisa y reflejar un entendimiento sólido de las consultas.
3. **Esquema relacional en `.sql`** (40 %):
   - Se evaluará la correcta implementación del esquema relacional, incluyendo la creación de tablas y relaciones.
   - Se considerará la inclusión de restricciones como `NOT NULL`, `AUTO INCREMENT`, `CONSTRAINTS`, y verificaciones necesarias.

Recursos

- Live SQL
- SQL Language Reference
- SQL Tutorial
- mockaroo