Pontificia Universidad Católica Madre y Maestra

Campus: Santo Tomás de Aquino

Departamento de Ingeniería

Base de Datos I

Asignación 1

Javier Falcón (2016-5265)

Raúl Herrera (2016-5536)

Manuel Molina (2016-5468)

Santo Domingo, 09/10/2018

Tipo de asignación: Lectura e Investigación

Tema: Capitulo 4

4.1 Introducción

Análisis critico

SQL (Structured Query Language) surge de la evolución del originalmente denominado Sequel, implementado por IBM a principios de 1970. El lenguaje SQL está formado de varios componentes: lenguaje de definición de datos (LDD), lenguaje interactivo de manipulación de datos (LMD), definición de visitas, control de transacciones, SQL incorporado y dinámico, integridad y autorización.

Creatividad

Imagínate que coleccionas rocas. En tu colección tienes varias rocas de diferentes tipos tamaños y colores. Si empiezas a guardar las rocas sin ninguna consideración cuando quieras encontrar alguna roca especifica no tendrás como encontrarla. Por esta razón es bueno tener algún tipo de sistema que te permita eficientemente almacenar e identificar cosas, en este caso rocas. Para esto es que usamos SQL en informática.

4.2 Estructura Básica

Análisis critico

Las bases de datos relacional están formadas de un conjunto de relaciones con nombres únicos. Cada expresión SQL consiste en tres grupos: select, from y where. Select tiene como función listar atributos deseados de la consulta. From, lista las relaciones o grupos que deben ser tomados en cuenta en la evaluación. Where, abarca los atributos de las relaciones que aparecen de la cláusula from.

4.2.1 La cláusula select

Los lenguajes formales de consulta tienen base matemática, una relación siendo un conjunto. SQL permite duplicados en las relaciones por lo que para evitar esto usamos la palabra clave “distinct” luego de select., si se desean presentar los duplicados se usa “all”. Select puede contener expresiones aritméticas con operadores sobre constantes u atributos de otras tuplas.

4.2.2 La cláusula where

Where permite operandos lógicos como <, <= ,> ,>= y = y operadores lógicos como and, or y not. Se permite el uso de operadores de comparación de cadenas y expresiones aritméticas. Se incluye el operador de comparación “between”

4.2.3 La cláusula from

La cláusula from es un producto cartesiano de las relaciones que se encuentran en la cláusula. Este funciona como una selección y una proyección.

4.2.4 La operación renombramiento

En SQL podemos renombrar relación y atributos, para esto se utiliza la cláusula as. Esta cláusula puede usarse tanto con select como con from.

4.2.5 Variables tupla

En SQL una variable tupla debe asociarse con una relación concreta. Se puede definir una variable tupla mediante la cláusula from usando la cláusula as. Las variables tupla pueden llegar a ser de gran utilidad al momento de comparar dos tuplas en la misma relación.

4.2.6 Operaciones sobre cadenas

En SQL para especificar una cadena se debe encerrar entre comillas. La operación más usada sobre cadenas es el operador “like” y para la descripción de patrones se utilizan el carácter “%” que encaja con cualquier subcadena y el carácter “_” que encaja con cualquier carácter.

4.2.7 Orden en la presentación de las tuplas

En SQL tenemos control sobre el orden en el que se presentan las tuplas en una relación, para obtener este orden utilizamos la cláusula “order by”, esta cláusula hace que las tuplas de una consulta salgan en un orden.

4.2.8 Duplicados

Para obtener duplicados de una consulta SQL se utilizan operadores relacionales para multiconjuntos. En SQL los duplicados pueden ser de uso para definir las tuplas que están en el resultado de una consulta y para el número de copias de cada una de las tuplas que aparece en el resultado.

Creatividad

Supongamos que vas de compras y estás buscando vegetales, específicamente zanahorias. En este caso podemos decir que el "select" seria vegetal, en otras palabras, estas especificando que lo que quieres es un vegetal. El "from" seria supermercado. Aquí estas especificando de donde lo quieres. Y finalmente el "where" seria zanahoria. Con esto, estas especificando que el vegetal que quieres es una zanahoria y no otro tipo de vegetal.

4.3 Operaciones sobre conjuntos

Análisis critico

En SQL las operaciones unión, intersect y except son equivalentes a las operaciones algebraicas relacional unión, intersección y diferencia. En SQL estas operaciones deben tener el mismo conjunto de atributos a diferencia de una contraparte algebraica.

4.3.1 La operación unión

La operación unión elimina duplicados. Si se desean conservar los duplicados se debe utilizar unión all. La operación unión a diferencia de la cláusula select no conserva duplicados.

4.3.2 La operación intersección

La operación intersect elimina duplicados. Para conservar los duplicados sustituimos intersect por intersect all.

4.3.3 La operación excepto

Except elimina automáticamente todos los duplicados. Si se quieren conserva los duplicados al utilizar except, debe ponerse except all en vez de except.

Creatividad

Imaginemos que un amigo nos regaló su colección de monedas y ahora quieres juntarla con la tuya. Con un "union" todas tus monedas y las de tu amigo quedarían todas en una sola colección. Pero antes de esto, quieres ver cual monedas tiene tenía tu amigo que ya tienes tu. Para esto podemos usar un "intersect". Ahora supongamos que solo te gustan las monedas viejas así que antes de unirlas podemos hacer un "except" para excluir las monedas que se hicieron del 2000 en adelante.

4.4 Función de agregación

Análisis critico

En SQL existen funciones de agregación, estas toman un grupo de valores como entrada y dan un solo valor como salida. Existen cinco funciones de agregación: avg (media), min (minimo), max (máximo), sum (total) y count (cuenta). Algunas de estas funciones pueden solo operar sobre datos numéricos a diferencia de otras que pueden operar tanto con datos numéricos como con no numéricos, tales como las cadenas. La entrada sum y avg solo operan sobre datos numéricos, min, max y count pueden operar datos numéricos y no numéricos.

Creatividad

Si en una escuela queremos sacar el promedio de la nota de los estudiantes, esto sería "avg". Si quisiéramos sacar la nota más baja, esto sería "min" y si quisiéramos sacar la nota más alta sería el "max". Si quisiéramos sacar cuanto se dinero obtiene anual del pago de los estudiantes se podría hacer un "sum". Finalmente, si quisiéramos saber cuántos estudiantes hay en la escuela, esto se podría hacer con un "count".

4.5 Valores Nulos

Análisis critico

Los valores nulos indican la ausencia de información del valor de un atributo. Se coloca la palabra null en el predicado para verificar si el valor es nulo. Para buscar los valores con ausencia de valores null se utiliza is not null. El resultado de una expresión aritmética da resultado nulo si cualquiera de los valores de entrada es nulo.

Creatividad

Los valores nulos son muy usados en el día a día. Por ejemplo, supongamos que eres un profesor y quieres saber cuáles estudiantes están ausentes. Para esto podemos seleccionar todos los estudiantes (select * from estudiantes) donde la asistencia este nula. (where absence is null).

4.6 Subconsultas Anidadas

Análisis critico

En SQL tenemos la herramienta de consultas anidadas. Una expresión anidada es una expresión (select, from y/o where) que esta anidada o dentro de otra consulta. El uso más común de subconjuntos realizar comprobaciones sobre pertenecías de conjuntos, de conjuntos y cardinalidad de conjuntos.

4.6.1

La conectiva in verifica la pertenencia a un conjunto, siendo este conjunto el grupo de valores resultados de una cláusula select. El uso más común de subconsultas es hacer comprobaciones sobre pertenencia de conjuntos, esto se puede hacer con in y not in, estos comprueban la pertenencia y la no pertenecía a un conjunto.

4.6.2

SQL nos permite realizar comparaciones a nivel e subconsulta utilizando <some, <= some, >= some, = some y <> some.

4.6.3

En SQL tenemos las herramientas para comprobar si una subconsulta no produce ninguna tupla como resultado, para esto utilizamos exist, esta nos devuelve cierto si la subconsulta no es vacia. También podemos utilizar not edist para comprobar la inexistencia de tuplas en el resultado de la subconsulta.

4.6.4

Podemos comprobar en SQL si una subconsulta da como resultado tuplas duplicadas utilizando unique que nos devuelve cierto si la subconsulta que se le pasa no nos da como resultado tuplas duplicadas. Tenemos la posibilidad de utilizar not unique para comprobar la existencia de subcosnultas con tuplas duplicadas.

Creatividad

Imaginemos nos que acabas de abrir un nuevo restaurante. Tenías otro en otro lugar, pero te moviste de local y quieres saber cuántos de tus viejos clientes vinieron a la ceremonia de inauguración. Para esto puedes tomar la lista de los clientes que asistieron a la inauguración y seleccionar todos “in” la otra lista que tienes de tus viejos clientes. Y ahora para saber cuántos de los clientes eran nuevos puedes seleccionar todos “not in” la lista que tienes de tus viejos clientes.

4.7. Vistas

Análisis crítico

Siendo una vista definida como una relación virtual que abstrae al usuario del modelo lógico, debemos ejecutar el comando de **create view** *my_view* **as** <*consulta*> para definir su creación. Se dice que estas vistas estarán descritas en cualquier lugar donde una relación pueda describirse.

Creatividad

En el tema 4.7 nos enseñan como crear un “view”. Un “view” es importante porque nos deja mostrar solo la información que creamos importantes. Por ejemplo, si estas vendiendo un producto, cuando vayas a presentarlo, no te interesa presentar datos no relevantes como las dimensiones o el material del producto. Solo quieres presentar la funcionalidad del producto. Un “view” nos permite hacer esto con las tablas en SQL.

4.8. Consultas Complejas

Análisis Crítico

En la práctica, la realización de consultas complejas puede ser un dolor de cabeza cuando se intentan escribir en un solo bloque o en varias operaciones de conjuntos sobre dichos bloques, por lo que se describen dos maneras para dividir estas consultas en varios bloques: relaciones derivadas y la cláusula **with.**

4.8.1. Relaciones derivadas

Las relaciones derivadas consisten en realizar una subconsulta dentro de la cláusula **from**, la cual debe ser nombrada para lograr su manipulación utilizando la cláusula **as.** El nombre de los atributos también puede ser modificado para fines de manipulación.

4.8.2 Cláusula **with**

Esta cláusula nos ofrece la creación de una vista temporal, la cual existe únicamente dentro de los ámbitos de la consulta.

Creatividad

Supongamos que estamos buscando una roca muy específica en nuestra colección de rocas. Sabes que la roca es sedimentaria y también sabes que es roja. Así seleccionas las rocas sedimentarias “select”. Tu eres muy organizado así que tienes tus rocas categorizadas en base a color también. Así que de donde selecciones va a ser de las rocas rojas [select from(select * rocas rojas)].

4.9. Modificación de la base de datos

Análisis crítico

4.9.1 Borrado

El borrado no puede eliminar valores de un atributo, solo elimina tuplas. El comando **delete from** *r* lo empleamos para realizar la consulta de borrado, donde *r* es una relación cualquiera válida. Existirá una orden **delete** por cada relación.

4.9.2 Inserción

La inserción de datos en un atributo puede hacerse tupla por tupla o por el resultado originado por una consulta. De igual manera, permite la inserción de valores nulos en campos donde esté permitido.

4.9.3 Actualizaciones

A través de la instrucción **update** podemos actualizar valores de una tupla elegida. Si utilizamos la cláusula **where** con la instrucción, limitamos la modificación de los datos.

4.9.4 Actualización de vistas

Las operaciones de **insert, delete y update** en una vista solo serán válidas si la vista se define en cuestión de la base de datos real.

4.9.5 Transacciones

Una transacción es una serie de consultas y actualizaciones. A través de 2 comandos, podemos manipular dichas transacciones. **Commit work** nos deja registrar permanentemente los cambios en la base de datos y **rollback work** restaura la base al estado previo en el que se encontraba antes de la transacción.

Creatividad

El borrado es una funcionalidad muy importante que nos deja modificar nuestros datos. Por ejemplos, imaginemos que tenemos una tienda de juegos y se dejó de fabricar un cierto juguete. Ahora que este juguete ya no se vende, queremos eliminarlo del nuestro listado inventario. Para esto podemos usar el “delete”.

4.10 Reunión de relaciones

Análisis crítico

SQL nos permite reunir relaciones a través de relaciones condicionales y reuniones naturales.

4.10.1 Ejemplos

Inner join: el resultado tendrá los atributos que estén en el lado izquierdo y en el lado derecho de las relaciones.

4.10.2 Tipos y condiciones de reunión

Hay tres condiciones de reunión: **natural, on** y **using.** Las condiciones de reunión son obligatorias para las reuniones externas y opcionales en las reuniones internas.

Right outer join: las tuplas del lado derecho que no existan en el lado izquierdo se tomarán como nulas. El **left outer join** se comporta de la misma manera, pero con el lado izquierdo.

Full outer join: reúne la combinación del **right outer join** con **left outer join.**

Creatividad

El “inner join” se puede ver como una unión con parámetros. Supongamos que tenemos 2 cajas de medias. Nosotros queremos sacar queremos sacar todas las medias iguales y meterlas en otra caja, pero solamente las que sean rojas. Para esto podemos hacer un “inner join” de las 2 cajas de medias y con el parámetro de que sean rojas.

4.11 Lenguaje de definición datos

Análisis crítico

El LDD de SQL nos permite definir el esquema de cada relación, los dominios de valores de los atributos, las restricciones, los índices de cada relación, protocolos de seguridad y la estructura del almacenamiento físico.

4.11.1 Tipos de dominio de SQL

Tenemos los siguientes tipos de datos: char (longitud definida), varchar (longitud variable), int, smallint, numeric (punto flotante), real, double precisión, float, date, time, timestamp.

4.11.2 Definición de esquemas en SQL

Los esquemas se crean utilizando **create table**, donde pueden tener restricciones de tipo **primary key** y **check(p)**, el cual evalúa la condición **p.**

Para borrar tablas utilizamos **drop table** *t*. Adicionalmente, la orden **alter table** se utiliza para agregar algún atributo en una tabla ya creada.

Creatividad

Toda información almacenada en una computadora es una combinación de 0s y 1s. Es la interpretación que le damos a esa combinación de 0s y 1s que determina que son esos datos, por esto se usan los tipos de dominios en SQL.

4.12 SQL incorporado

Análisis crítico

Debido a que no todas las consultas pueden ser expresadas en SQL, se vio la necesidad de integrar este lenguaje en otros lenguajes de propósito general como Java o C/C++. La introducción de consultas SQL en un lenguaje convierten a éste en el lenguaje anfitrión, por lo que los programas escritos en dicho lenguaje podrán actualizar datos dentro de la base de datos.

Creatividad

Hay veces cuando quieres hacer algún trabajo, pero resulta mejor hacerlo a través de algún otro servicio ya sea porque sea más eficiente o el servicio te deja hacer otras cosas que también quieras hacer.

4.13 SQL dinámico

A diferencia de SQL incorporado, SQL dinámico tiene la capacidad de realizar y construir consultas en tiempo de ejecución. Dichas consultas se hacen a través de sesiones: el usuario inicia sesión y consulta y, al finalizar, se desconecta.

4.13.1 ODBC (Open Database Connectivity)

Es un API que permite a un programa de aplicación comunicarse con el servidor de la base de datos.

4.13.2

Es el mismo concepto que ODBC, sólo que es un API exclusivo del lenguaje Java.

Cratividad

ODBC le permite a un programa conectarse a un servidor de base de datos. Para ilustrar esto podemos tomar como ejemplo cuando usamos el internet. Cuando no sabemos algo o queremos obtener información nosotros nos "conectamos" al internet a través de un celular o computadora.

4.14 Otras características de SQL

Análisis crítico

4.14.1 Esquemas, catálogos y entornos

SQL ofrece un sistema de jerarquía de tres niveles para ordenar las relaciones, siendo los catálogos el nivel superior. Los catálogos pueden agruparse en esquemas. Éstos, al mismo tiempo, se definen dentro de un entorno SQL.

4.14.2 Extensiones procedimentales y procedimientos almacenados.

SQL permite agrupar varios procedimientos SQL en módulos y cada uno de estos puede ser ejecutado con la instrucción **call.**

Creatividad

SQL permite procedimientos almacenados. While, if, then-else, etc. Estos son muy útiles. Por ejemplo, si estas moviendo escombros de un lugar a otro, puede hacer un "while" haya escombros, mover los escombros.

Bibliografía

Silberschatz, A., F. Korth, H. and Sudarshan, S. (2002). *FUNDAMENTOS DE BASES DE DATOS*. 4th ed. Madrid: McGraw-Hill.