**GUÍA 16**

Jessica Estefania Castillo Mejia

408

Programación de software

2016

**3.1 ¿Cuál sería la importancia de realizar consultas entre dos o más tablas que retornen registros específicos?**

**RTA:** Son importantes porque generalmente las utilizamos en nuestro sistema de información para cumplir con los requerimientos solicitados por los usuarios y nos permiten desglosar una consulta en una serie de pasos lógicos.

**Anidar:** Se puede anidar de forma que una subconsulta aparezca en la cláusula WHERE de otra subconsulta que a su vez forma parte de otra consulta principal.

**Lista de selección:** Indican las columnas que se tienen que visualizar en el resultado de la consulta.

**Cláusula FROM:** Es una lista separada por comas de nombres de tablas, nombres de vistas y cláusulas JOIN

**Referencia externa:** Es un nombre de columna que estando en la subconsulta, no se refiere a ninguna columna de las tablas designadas.

**Group by:** Se usa para generar valores de agregado para cada fila del conjunto de resultados.

**Order by:** Ordena los datos devueltos por una consulta en **SQL** Server.

**Clásula having:** Especifica una condición de búsqueda para un grupo o agregado. Solo se puede utilizar con la instrucción SELECT

**Consulta:** Es aquella que envía una solicitud de datos almacenados en SQL Server.

**Cláusula where:** Especifica los criterios que deben cumplir los valores de campo para los registros que contienen los valores que se van a incluir en los resultados de la consulta.

**Select:** Nos permite consultar los datos almacenados en una tabla de la base de datos.

**Paréntesis:** Identifica el orden de evaluación de las expresiones, tienen la prioridad de evaluación más alta y se evalúan desde dentro hacia fuera.

**Clave primaria:** Se utiliza para identificar en forma única cada línea en la tabla. Puede consistir en uno o más datos en una tabla.

**Restricciones:** Definen reglas relativas a los valores permitidos en las columnas y constituyen el mecanismo estándar para exigir la integridad.

**Operadores de comparación:** Comprueban si dos expresiones son iguales. Se pueden utilizar en todas las expresiones excepto en las de los tipos de datos text, ntext o image.

**On:** Es una combinación entre tablas, las vinculaciones entre tablas se realizan mediante la cláusula INNER

**Unión externa:** Es una reunión interna donde no es necesario que el registro hijo tenga informada la clave foránea para ser mostrado,

**Producto cartesiano:** Es un tipo de composición de tablas, aplicando el producto cartesiano a dos tablas se obtiene una tabla con las columnas de las 2 tablas

**Unión interna:** Es la que reúne registros de varias tablas, en ella intervienen las claves primarias y foráneas,

**Composición interna:** Es aquella en la que los valores de las columnas que se están combinando se comparan mediante un operador de comparación.

**Composición de tablas:** Consiste en obtener a partir de dos tablas una nueva tabla fusionando las filas de una con la otra.

**Unión de tablas:** Es combinar los resultados de dos consultas juntas y se necesitan que sean del mismo tipo de datos.

**CONSULTA MULTITABLA**

Es una consulta que obtiene datos de varias tablas por lo que deberá contener en la zona de tablas de la ventana Diseño las distintas tablas de donde obtiene esos datos.

**SUBCONSULTAS**

**2. \* Crear una vista que me consulte el cargo del empleado, el nombre y la fecha de la entrevista para calcular su salario.**

CREATE VIEW SALARIO\_EMPLEADO AS SELECT EMPLEADO.COD\_EMP, EMPLEADO.NOM\_EMP, CARGO.COD\_CAR, CARGO.NOM\_CAR, ENTREVISTA.COD\_ENT, ENTREVISTA.FEC\_ENT

FROM SALARIO INNER JOIN ENTREVISTA INNER JOIN CARGO ON ENTREVISTA.COD\_ENT = CARGO.COD\_CAR INNER JOIN EMPLEADO ON CARGO.COD\_CAR = EMPLEADO.COD\_EMP ON SALARIO = ENTREVISTA.COD\_ENT

CROSS JOIN ENTREVISTA CROSS JOIN SALARIO CROSS JOIN EMPLEADO INNER JOIN CARGO ON EMPLEADO.COD\_EMP = CARGO.COD\_CAR

SELECT \* FROM SALARIO\_EMPLEADO;

\* FROM SALARIO INNER JOIN ENTREVISTA INNER JOIN CARGO ON ENTREVISTA.COD\_ENT = CARGO.COD\_CAR INNER JOIN EMPLEADO ON CARGO.COD\_CAR = EMPLEADO.COD\_EMP ON SALARIO = ENTREVISTA.COD\_ENT

\*

CREATE VIEW EJEMPLO AS SELECT EMPLEADO.COD\_EMP, EMPLEADO.NOM\_EMP, FACTURA\_COMPRA.COD\_FAC, FACTURA\_COMPRA.FEC\_FAC, DET\_FAC\_COMPRA.CAN\_PRO, EJ\_PRODUCTO.VAL\_PRO

FROM PRV INNER JOIN EJ\_PRODUCTO INNER JOIN DET\_FAC\_COMPRA ON EJ\_PRODUCTO.COD\_PRO = DET\_FAC\_COMPRA.COD\_PRO INNER JOIN FACTURA\_COMPRA ON DET\_FAC\_COMPRA.COD\_FAC = FACTURA\_COMPRA.COD\_FAC ON PRV.COD\_PRO = EJ\_PRODUCTO.COD\_PRO

CROSS JOIN DET\_FAC\_VEN CROSS JOIN FAC\_VENTA CROSS JOIN EMPLEADO INNER JOIN CARGO ON EMPLEADO.COD\_EMP = CARGO.COD\_CAR

SELECT \* FROM EJEMPLO;

**3.4**

**-** Seleccionar las tablas de mi base de datos que se requieren para realizar una consulta de todos los atributos que se encuentran en ella.

- Usar la cláusula where para identificar atributos específicos en la base de datos.

- Consultar mis tablas en la base de datos de SQL Server.

- Utilizar la cláusula FROM para ordenar los nombres de vistas y cláusulas JOIN en una lista separada por comas.

- Realizar una unión interna entre mis atributos, ya sean claves primarias o foráneas.