FUNDAMENTOS DE BASE DE DATOS.

Tema Nº9:Lenguaje para la manipulación de datos DML II.

Indicador de logro Nº9:Aplica sentencias optimizadas y funciones, en casos prácticos.

**TEMA 01 Teoría de los**

Imagen que contiene Icono

Descripción generada automáticamente

**TEMA Nº9:**

Lenguaje para la manipulación de datos DML II.

**Subtema 9.1:**

Empleo de funciones para el manejo de fechas.

DAY (), MONTH (), YEAR (), DATEPART (), DATENAME (), DATEADD (), DATEDIFF ().

http://www.w3bai.com/es/sql/func\_datediff.html

|  |  |
| --- | --- |
| **Función** | **Descripción** |
| **GETDATE()** | Devuelve la fecha del sistema.  Ejemplo:  **PRINT**(**GETDATE**()) |
| **DAY()** | Devuelve el día de la fecha.  Ejemplo:  **PRINT**(**DAY**(**GETDATE**()) |
| **YEAR()** | Devuelve el año de la fecha.  Ejemplo:  **PRINT**(**YEAR**(**GETDATE**()) |
| **DATEPART()** | Devuelve un entero que representa la parte de la fecha especificada de la fecha indicada  Sintaxis: **DATEPART** (Abreviatura, expresión de fecha) |
| **DATENAME()** | Devuelve una cadena de caracteres que representa la parte de la fecha especificada de la fecha especificada  Sintaxis: **DATENAME** (Abreviatura, Expresión de Fecha) |
| **DATEADD()** | Añade un intervalo de tiempo especificado en una fecha  Sintaxis: DATEADD (Parte Fecha, Numero, Fecha) |
| **DATEDIFF()** | Devuelve parte de la fecha entre dos fechas  Sintaxis: **DATEDIFF** (parte, fecha1, fecha2) |

**DATENAME:**

Devuelve una cadena de caracteres que representa la parte de la fecha especificada de la fecha especificada

Sintaxis: **DATENAME** (Abreviatura, Expresión de Fecha)

|  |  |
| --- | --- |
| **PARTE DE LA FECHA** | **ABREVIATURA** |
| **Year** | yy, yyyy (Numero del años en 2 o 4 digitos) |
| **Quarter** | qq, q (número de trimestres transcurridos según fecha actual) |
| **Month** | mm, m (Nombre del mes (Enero - Diciembre) Español o Inglés según versión del programa. |
| **Dayofyear** | dy, y (Numero del día del año 1 - 365) |
| **Day** | dd, d (Numero del día del mes 1 - 31) |
| **Week** | wk, ww (semana del año) |
| **Weekday** | Dw (nombre del día de la semana “Lunes” – “Viernes”) |
| **Hour** | hh (devuelve el número de la hora) |
| **Minute** | mi, n (devuelve el minuto 1 – 59) |
| **Second** | ss, s (devuelve el segundo 1 – 59) |
| **Millisecond** | ms (devuelve el milisegundo) |

**DATEPART:**

Devuelve un entero que representa la parte de la fecha especificada de la fecha indicada

Sintaxis: **DATEPART** (Abreviatura, expresión de fecha)

|  |  |
| --- | --- |
| **PARTE DE LA FECHA** | **ABREVIATURA** |
| **Year** | yy, yyyy (Numero del años en 2 o 4 dígitos) |
| **Quarter** | qq, q (número de trimestres transcurridos según fecha actual) |
| **Month** | mm, m (Numero de mes 1 - 12) |
| **Dayofyear** | dy, y (Numero del día del año 1 - 365) |
| **Day** | dd, d (Numero del día del mes 1 - 31) |
| **Week** | wk, ww (semana del año) |
| **Weekday** | Dw (número del día de la semana 1 - 7) |
| **Hour** | hh (devuelve el número de la hora) |
| **Minute** | mi, n (devuelve el minuto 1 – 59) |
| **Second** | ss, s (devuelve el segundo 1 – 59) |
| **Millisecond** | ms (devuelve el milisegundo) |

**DATEADD:**

Suma o resta un intervalo de tiempo especificado a partir de una fecha

Sintaxis: DATEADD (Parte Fecha, Numero, Fecha)

|  |  |
| --- | --- |
| **PARTE DE LA FECHA** | **ABREVIATURA** |
| **Year** | yy, yyyy (Numero del años en 2 o 4 dígitos) |
| **Quarter** | qq, q (número de trimestres transcurridos según fecha actual) |
| **Month** | mm, m (Numero de mes 1 - 12) |
| **Dayofyear** | dy, y (Numero del día del año 1 - 365) |
| **Day** | dd, d (Numero del día del mes 1 - 31) |
| **Week** | wk, ww (semana del año) |
| **Weekday** | Dw (número del día de la semana 1 - 7) |
| **Hour** | hh (devuelve el número de la hora) |
| **Minute** | mi, n (devuelve el minuto 1 – 59) |
| **Second** | ss, s (devuelve el segundo 1 – 59) |
| **Millisecond** | ms (devuelve el milisegundo) |

**DATEDIFF:**

Devuelve parte de la fecha entre dos fechas

Sintaxis: **DATEDIFF** (parte de la fecha, fecha1, fecha2)

|  |  |
| --- | --- |
| **PARTE DE LA FECHA** | **ABREVIATURA** |
| **Year** | yy, yyyy (Numero del años en 2 o 4 dígitos) |
| **Quarter** | qq, q (número de trimestres transcurridos según fecha actual) |
| **Month** | mm, m (Numero de mes 1 - 12) |
| **Dayofyear** | dy, y (Numero del día del año 1 - 365) |
| **Day** | dd, d (Numero del día del mes 1 - 31) |
| **Week** | wk, ww (semana del año) |
| **Weekday** | Dw (número del día de la semana 1 - 7) |
| **Hour** | hh (devuelve el número de la hora) |
| **Minute** | mi, n (devuelve el minuto 1 – 59) |
| **Second** | ss, s (devuelve el segundo 1 – 59) |
| **Millisecond** | ms (devuelve el milisegundo) |

**CONVERT** (Nuevo Tipo, Valor Tipo)

Convierte un Tipo de Dato, a un tipo de dato deseado, estas conversiones se podrán usar en otros procesos.

|  |  |
| --- | --- |
| **MODO DE CONVERSION** | **EXPLICACION** |
| CONVERT (datetime, expresión) | Convierte la expresión a tipo datetime |
| CONVERT (varchar, expresión) | Convierte la expresión a tipo Varchar |
| CONVERT (char, expresión) | Convierte la expresión a tipo Char |
| CONVERT (Int, expresión) | Convierte la expresión a tipo Int (tipo entero numérico) |
| CONVERT (Real, expresión) | Convierte la expresión a tipo Real (tipo entero numérico) |
| CONVERT (Numeric(x,y), expresión) | Convierte la expresión a tipo Numeric con o sin decimales |

EJEMPLOS:

Se tiene la siguiente tabla de datos:



**Ejemplo 1**:

Obtener la fecha actual.



**Ejemplo 2:**

De la fecha actual obtener el día. Si la fecha actual es 05 de julio del 2021 entonces:



**Ejemplo 3:**

Obtener de la fecha actual el año. Si la fecha actual es 05 de julio del 2021 entonces:



**Ejemplo 4:**

Obtener de la fecha actual el mes. Si la fecha actual es 05 de julio del 2021 entonces:



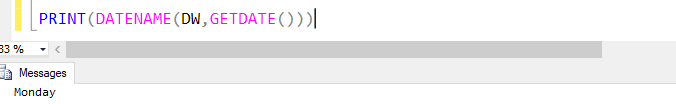
**Ejemplo 5:**

Obtener de la fecha actual parte de la fecha día.



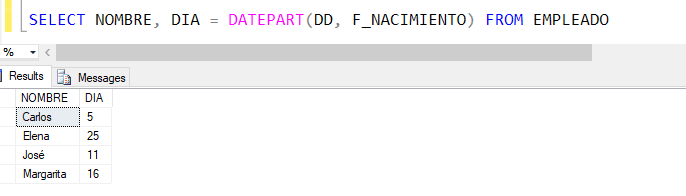
**Ejemplo 6:**

Obtener el día de la semana (en texto) de la fecha actual.



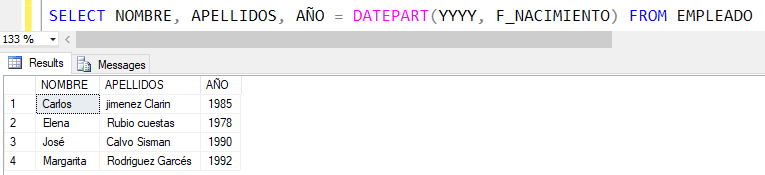
**Ejemplo 7:**

De la tabla EMPLEADO, se pide mostrar el nombre y el numero día de su fecha de nacimiento.



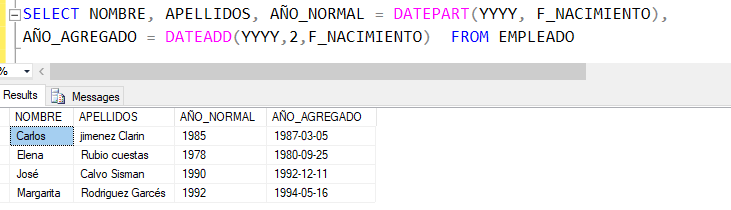
**Ejemplo 8:**

De la tabla EMPLEADO, se pide mostrar el nombre y apellido, el año de su fecha de nacimiento.



**Ejemplo 9:**

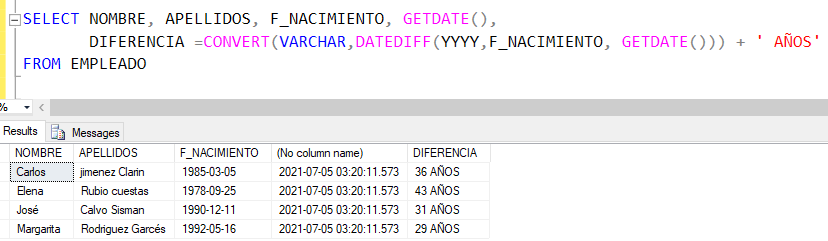
De la tabla EMPLEADO, se pide mostrar el nombre y apellido, el año de su fecha de nacimiento agregado 2 años.



.

**Ejemplo 10:**

De la tabla EMPLEADO, se pide mostrar el nombre y apellido, fecha de nacimiento, fecha actual y la diferencia en años entre la fecha de nacimiento y la fecha actual.



**Subtema 9.2:**

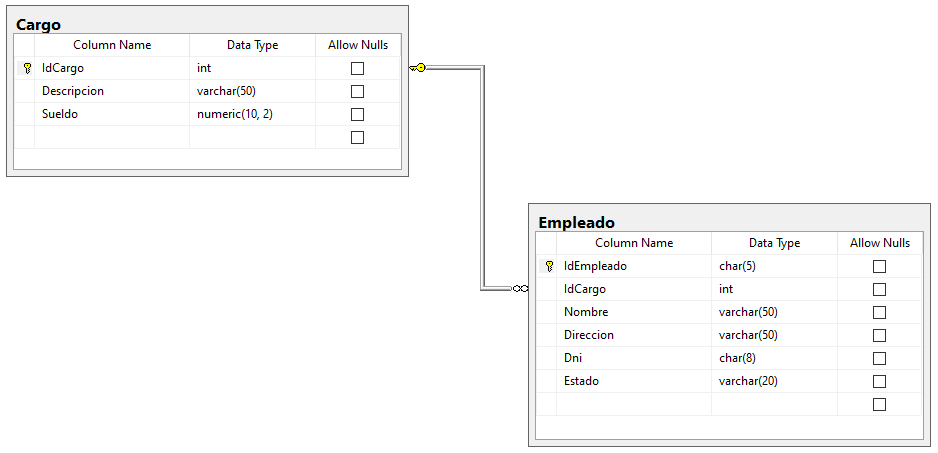
Consulta Multi tablas, Alias de tablas, Regla de Relación entre llaves PK - FK usando la cláusula WHERE.

**CONSULTAS DE TIPO MULTI – TABLA.**

Es habitual que queramos acceder a datos que se encuentran en más de una tabla y mostrar información mezclada de todas ellas como resultado de una consulta. Para ello tendremos que hacer combinaciones de columnas de tablas diferentes.

Este último punto es muy importante, ya que si seleccionamos varias tablas y no hacemos coincidir los valores de las columnas relacionadas, obtendremos una gran duplicidad de filas, realizándose el [producto cartesiano](https://es.wikipedia.org/wiki/Producto_cartesiano) entre las filas de las diferentes tablas seleccionadas.

En este ejemplo tomaremos en cuenta el siguiente modelo relacional, donde se enfrentan las tablas **EMPLEADO** y **CARGO**. Y estos llevan una relación por el **IdCargo**.

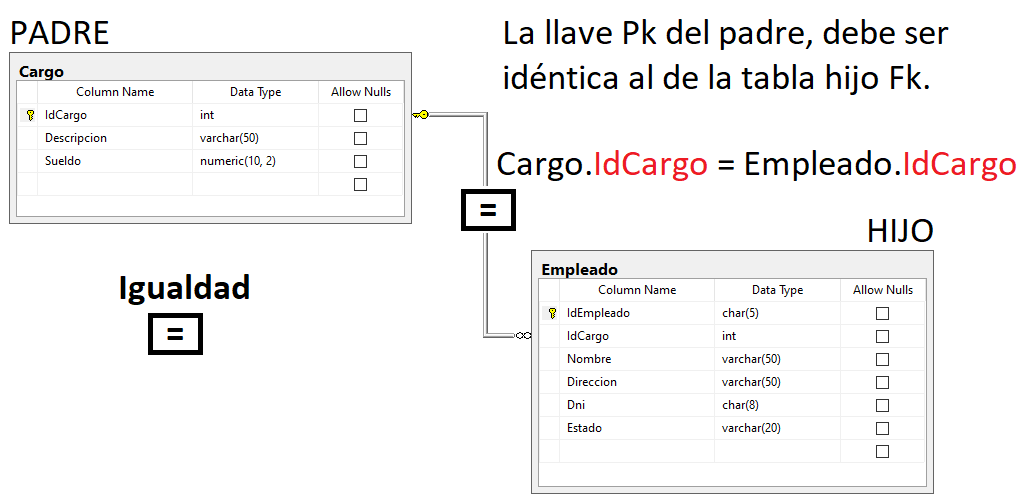


**SE DEBE RESPETAR SIEMPRE UNA REGLA Y ESTA ES:**

LA LLAVE DE LA TABLA PADRE (**PK**), DEBE COINCIDIR CON LA LLAVE (**FK**) DE LA TABLA HIJO. Es decir, deben ser idénticas en contenido y estructura.

En este ejemplo la tabla padre es CARGO y la tabla hijo es EMPLEADO.

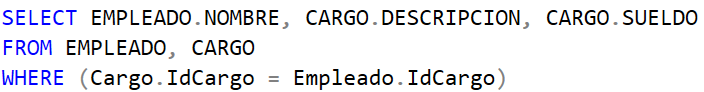
Si se desea hacer valer esta regla, entonces la gráfica del modelo relacional debe tener el siguiente cambio.



**Ahora si se puede hacer una consulta de datos usando varias tablas, por ejemplo:**

**Ejemplo 1:**

Se dese mostrar en una consulta multi tabla, al Nombre del Empleado, Cargo y Sueldo de todos los empleados.

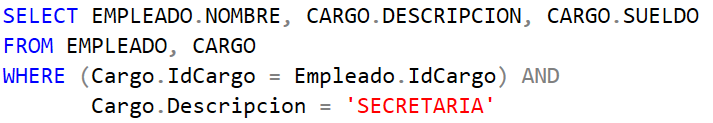


**Nota:**

Siempre se debe respetar la regla de las llaves **PK** y **FK**. Entre las tablas padres e hijos. La clausula WHERE se encarga de comparar dichas llaves.

**Ejemplo 2:**

Se dese mostrar en una consulta multi tabla, al Nombre del Empleado, Cargo y Sueldo de todos los empleados en especial los que ocupan el cargo de SECRETARIA.



**Nota:**

Sin importar que clase de consulta se solicite, siemprese ha de respetar la regla de las llaves **PK** y **FK**. Entre las tablas padres e hijos.

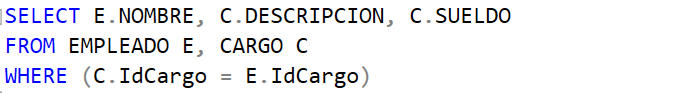
**ESTO SE PUEDE MEJORAR UN POCO, USANDO LOS ALIAS EN LAS TABLAS.**

**¿Que son los Alias?**

Elementos de base de datos, capaces de reemplazar el nombre de una tabla, por un sobre nombre, observe:

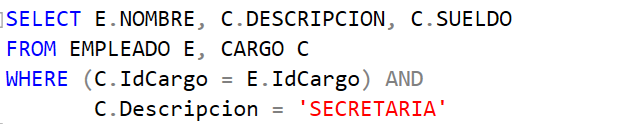
**Ejemplo 1:**

Se dese mostrar en una consulta multi tabla, al Nombre del Empleado, Cargo y Sueldo de todos los empleados.



**Ejemplo 2:**

Se dese mostrar en una consulta multi tabla, al Nombre del Empleado, Cargo y Sueldo de todos los empleados en especial los que ocupan el cargo de SECRETARIA.



**Nota:**

Como observa en ambos ejemplos, de manera explicita ya no se usan los nombre de tablas en las columnas, por el contrario usamos un alias, este puede ser una palabra, letra o frase que Ud. Mismo puede inventar.

En la clausula FROM es donde se definen los alias de cada tabla por ejemplo: **FROM EMPLEADO E, CARGO C**. esto quiere decir que la tabla empleado sera reconocida con el alias **E** y la tabla cargo será reconocida con el alias **C**.

La clausula **WHERE** se encarga de comparar dichas llaves.

**Subtema 9.3:**

Sub consultas y combinación de tablas.

**Internas INNER JOIN**, **externas** **LEFT OUTER JOIN** & **RIGHT OUTER JOIN.**

**INTRODUCCIÓN A SUB CONSULTAS:**

Una subconsulta es una consulta que aparece dentro de otra consulta o subconsultas, en la lista de selección o en la cláusula WHERE o HAVING, originalmente no se podían incluir en la lista de selección.

Una subconsulta se denomina también consulta o selección interna, mientras que la instrucción que contiene la subconsulta es conocida como consulta o selección externa.

Aparece siempre encerrada entre paréntesis y tiene la misma sintaxis que una sentencia SELECT normal con alguna limitación:

No puede incluir una cláusula COMPUTE o FOR BROWSE y sólo puede incluir una cláusula ORDER BY cuando se especifica también una cláusula TOP.

Una subconsulta puede anidarse en la cláusula WHERE o HAVING de una instrucción externa SELECT, INSERT, UPDATE o DELETE, o bien en otra subconsulta. Se puede disponer de hasta 32 niveles de anidamiento, aunque el límite varía dependiendo de la memoria disponible y de la complejidad del resto de las expresiones de la consulta. Hay que tener en cuenta que para cada fila de la consulta externa, se calcula la subconsulta, si anidamos varias consultas, el número de veces que se ejecutarán las subconsultas ¡puede dispararse!

La sintaxis de una subconsulta en SQL cuando el subquery o subconsulta forma parte de la condición WHERE es la siguiente:

SELECT "column\_name1"

FROM "table\_name1"

WHERE "column\_name2" [Operador de Comparación]

(SELECT "column\_name3"

FROM "table\_name2"

WHERE "condición")

**[Operador de comparación]** podría ser operadores de igualdad como =,>, <,> =, <=. También puede ser un operador de texto como "**LIKE**". La parte entre paréntesis se considera como la "**consulta interna, subconsulta o subquery**", mientras que la parte que no está entre paréntesis se considera como la "**consulta externa**".

**IMPORTANTE:**

Cuando la subconsulta aparece en la lista de selección de otra consulta, deberá devolver un solo valor, de lo contrario provocará un error.

**Ejemplo de subconsulta:** Listar los empleados cuya cuota no supere el importe vendido por el empleado.

**SELECT nombre**

**FROM empleados**

**WHERE cuota** <= **(SELECT SUM(importe)**

**FROM pedidos**

**WHERE rep = numemp);**

*numemp* es una referencia externa, no es una columna del origen de datos de la subconsulta (*pedidos*), es una columna del origen de la consulta externa (*empleados*).  
Hay que tener en cuenta de cómo se ejecuta la consulta; por cada fila de la consulta externa se calcula el resultado de la subconsulta y se evalúa la comparación.  
En el ejemplo, se coge el primer empleado (numemp= 101, por ejemplo) y se calcula la subconsulta sustituyendo numemp por el valor 101, se calcula la suma de los pedidos del rep = 101, y el resultado se compara con la cuota de ese empleado, y así se repite el proceso con todas las filas de empleados.

El nombre de una columna dentro de la subconsulta se presupone del origen de datos de la subconsulta y, sólo si no se encuentra en ese origen, la considera como columna externa y la busca en el origen de la consulta externa.

**Por ejemplo:**

**SELECT oficina, ciudad**

**FROM oficinas**

**WHERE objetivo > (SELECT SUM(ventas)**

**FROM empleados WHERE oficina = oficina);**

La columna *oficina* se encuentra en los dos orígenes (***oficinas*** y ***empleados***) pero esta consulta no dará error (no se nos pedirá cualificar los nombres como pasaría en una composición de tablas), dentro de la subconsulta se considera oficina el campo de la tabla empleados. Con lo que compararía la oficina del empleado con la misma oficina del empleado y eso no es lo que queremos, queremos comparar la oficina del empleado con la oficina de oficinas, lo escribiremos pues así para forzar a que busque la columna en la tabla oficinas.

**SELECT oficina, ciudad**

**FROM oficinas**

**WHERE objetivo >****(SELECT SUM(ventas)**

**FROM empleados**

**WHERE oficina = oficinas.oficina)**

**Tenemos tres tipos de subconsultas:**

* Las que devuelven un solo valor, aparecen en la lista de selección de la consulta externa o con un operador de comparación sin modificar.
* Las que generan una columna de valores, aparecen con el operador IN o con un  operador de comparación modificado con  ANY, SOME o ALL.
* Las que pueden generar cualquier número de columnas y filas, son utilizadas en pruebas de existencia especificadas con EXISTS.

**COMBINACIÓN DE TABLAS:**

Recuperación avanzada de datos.

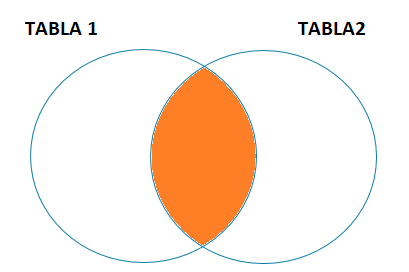
* Técnicas de consultas más avanzadas.
* Una de estas técnicas es la de combinar contenidos de una o más tablas para producir un conjunto de resultados que incorpore filas y columnas de cada tabla.
* Otra técnica es la de agrupar los datos para obtener, desde un conjunto de filas, datos agrupados

**INSTRUCCIÓN JOIN**

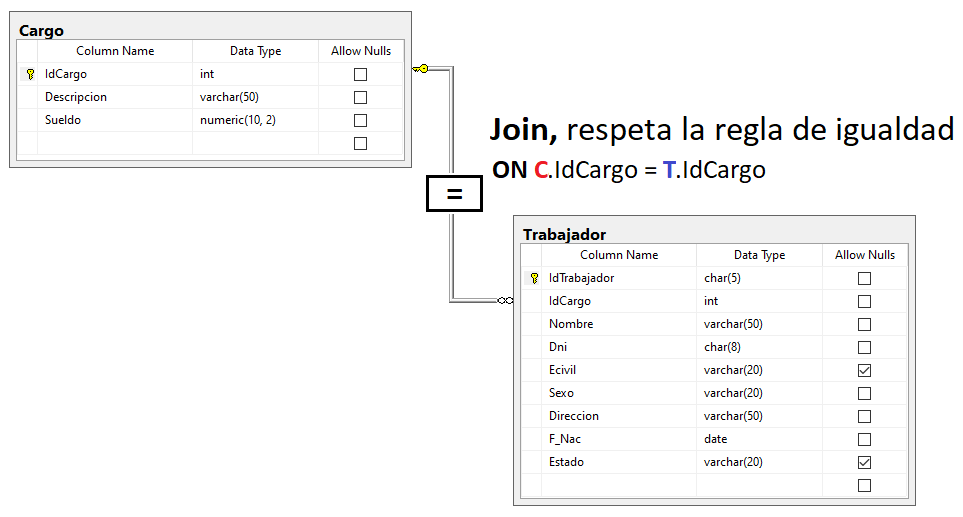
* JOIN en el lenguaje de consulta, permite combinar registros de dos o más tablas en una base de datos relacional.
* La sentencia JOIN se pueden especificar en las cláusulas FROM o WHERE, aunque se recomienda que se especifiquen en la cláusula **FROM**.

**COMBINACIÓN INTERNA:**

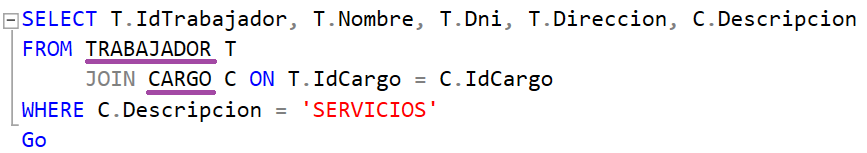
* El operador INNER JOIN permite la combinación de los registros de las tablas, comparando los valores de la columna específica en ambas tablas.
* Cuando no existe esta correspondencia, el registro no se muestra.



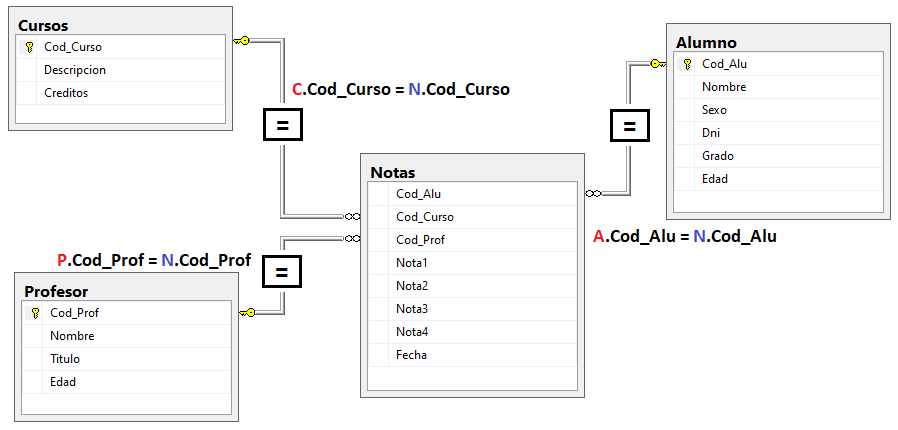
**EJEMPLO DE COMBINACIÓN DE 2 TABLAS:**



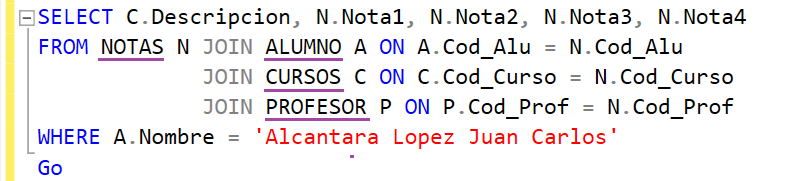
Se desea mostrar los datos de los empleados cuyo cargo sea “**SERVICIOS**’.



**EJEMPLO DE COMBINACIÓN DE 3 a MAS TABLAS:**



Se desea mostrar las asignaturas y calificaciones del alumno: ‘**Alcantara Lopez Juan Carlos’.**



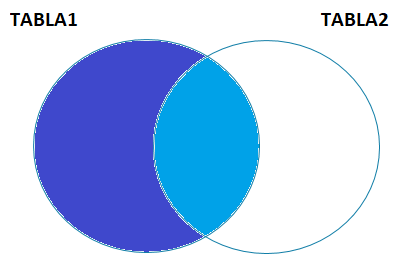
**COMBINACION EXTERNA**

Mediante esta operación no se requiere que cada registro en las tablas a tratar tenga un registro equivalente en la otra tabla. El registro es mantenido en la tabla combinada si no existe otro registro que le corresponda

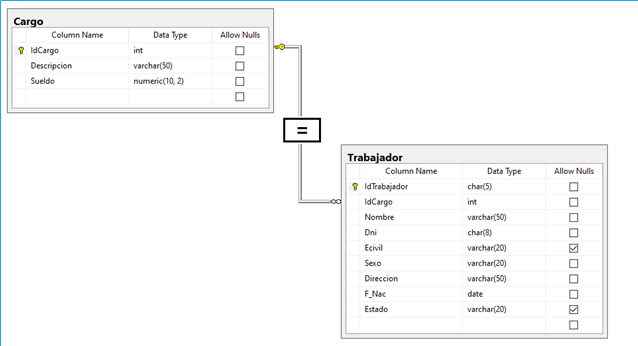
* Las combinaciones externas especificadas en una cláusula FROM:
  + LEFT OUTER JOIN o LEFT JOIN
  + RIGHT OUTER JOIN o RIGHT JOIN
  + FULL OUTER JOIN o FULL JOIN

**Combinación Externa LEFT JOIN**

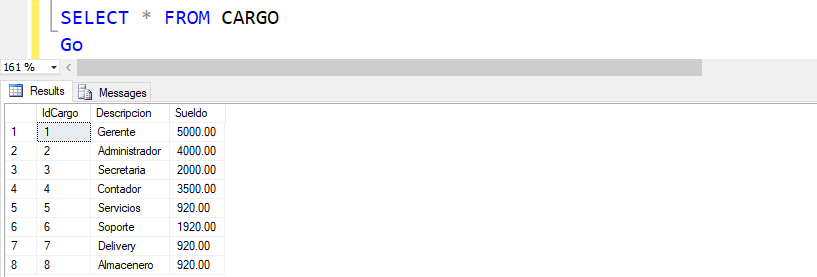
* El operador de combinación LEFT JOIN retorna la pareja de todos los valores de la izquierda con los valores de la tabla de la derecha correspondientes, o retorna un valor nulo NULL en caso de no correspondencia.

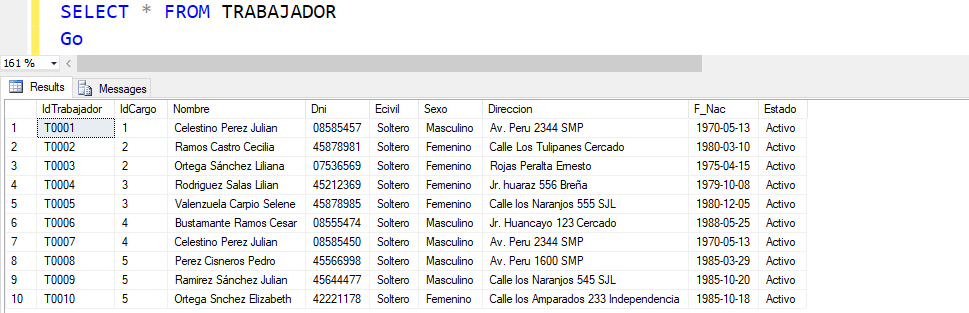


Para probar la combinación externa LEFT JOIN, debemos contar con el sgiguiente modelo relacional:



Cuyos datos son los siguientes:

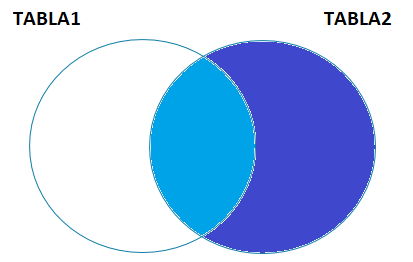


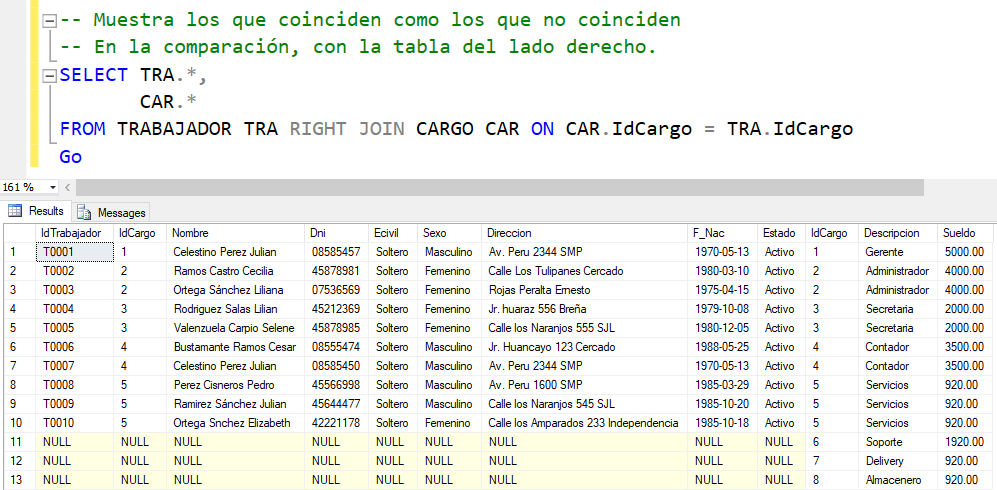




**Combinación Externa RIGHT JOIN**

* El operador de combinación RIGHT JOIN retorna la pareja de todos los valores de la derecha con los valores de la tabla de la izquierda correspondientes, o retorna un valor nulo NULL en caso de no correspondencia.





**CONCLUSIONES Y RECOMENDACIONES DE LA EXPERIENCIA**

* Comprende el Lenguaje de Manipulación de Datos (DML) utilizando el lenguaje de manipulación de datos INNER JOIN, LEFT JOIN y RIGHT JOIN.

**ACTIVIDAD VIRTUAL**

Analizar y revisar los siguientes enlaces; luego responde las preguntas propuestas.

<https://www.youtube.com/watch?v=2LtcWYdVx_I>

<https://www.youtube.com/watch?v=eNikUI0Y8y8>

* ¿Cuál es la finalidad de INNER JOIN?
* ¿Cuál es la finalidad de LEFT JOIN?
* ¿Cuál es la finalidad de RIGHT JOIN?
* ¿Cuál es la diferencia en hacer un LEFT JOIN con un RIGHT JOIN?
* ¿Cuál es la diferencia de Unión externa e interna?