Universidad Autónoma de Chile
Facultad de Ingeniería
Ingeniería en Informática

Guía Nº 04

Base de Datos II

Docente: Sergio Antonio Baltierra Valenzuela

21 **de abril de** 2020

Operadores SQL

Requisitos de Software

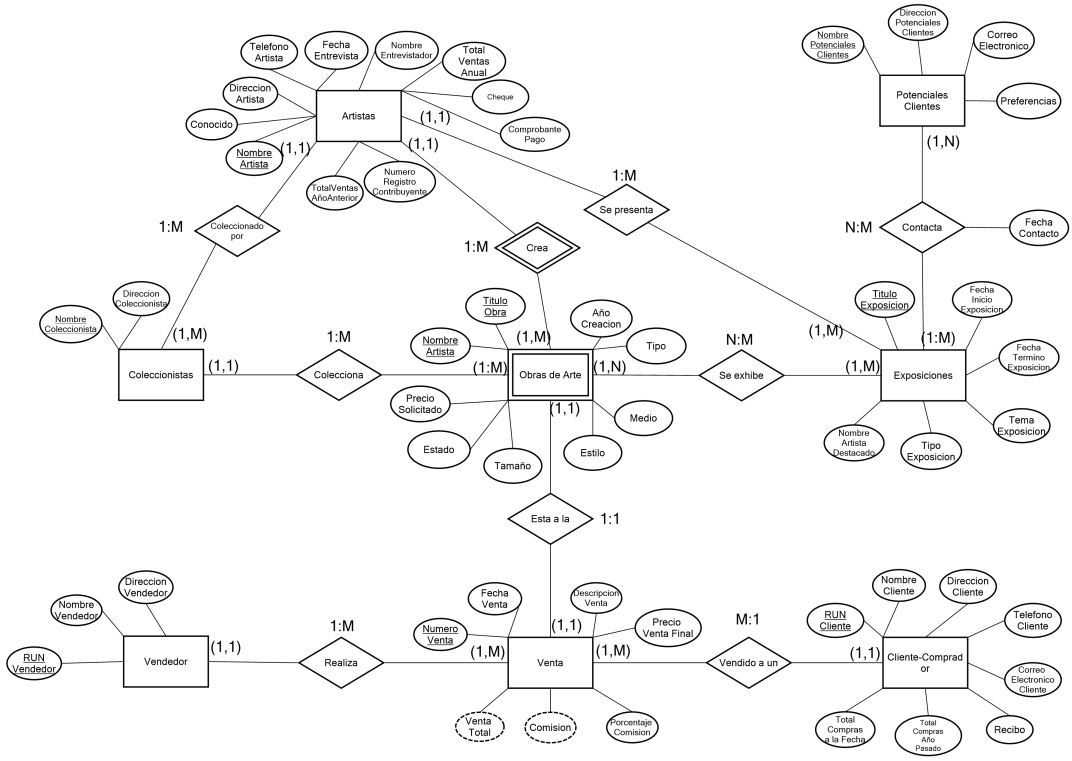
A continuación para el desarrollo de esta guía debe tener instalado en su equipo los siguientes softwares:

- Editor de código como: Sublime Text 3, Visual Studio Code o similares.

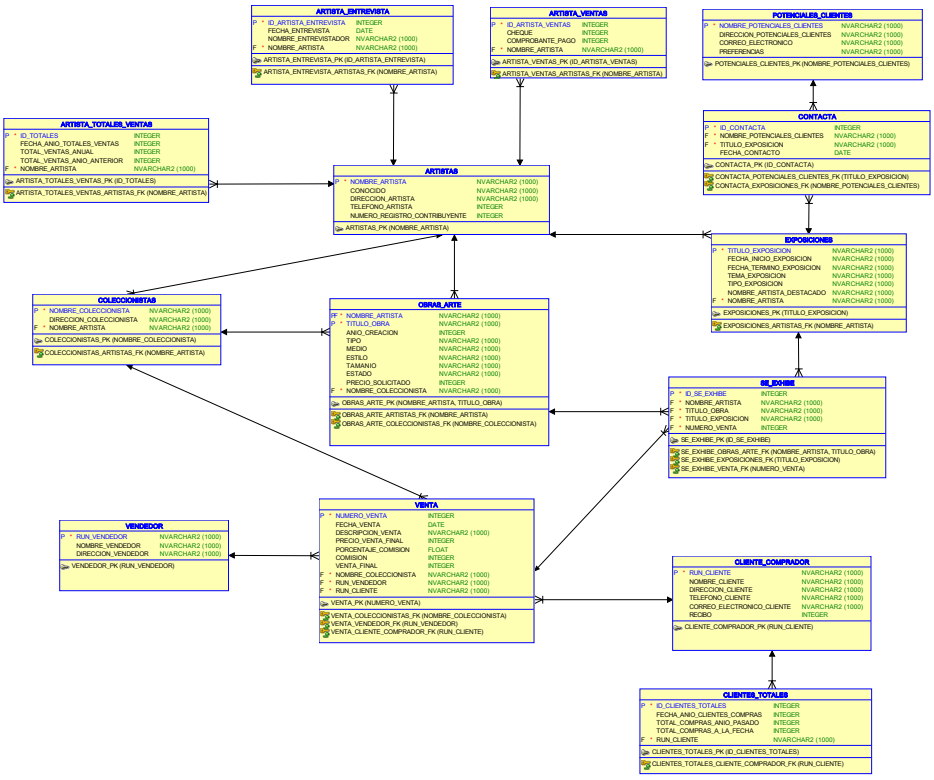
Enunciado

A continuación se presentan los modelos MER y MR:

Modelo MER


Artistas
Telefono Artista
Fecha Entrevista
Nombre Entrevistador
Total Ventas Anual
Cheque
Comprobante Pago
Direccion Artista
Conocido
Nombre Artista
TotalVentas AñoAnterior
Numero Registro Contribuyente
(1,1)
(1,1)
(1,1)
(1,1)
Potenciales Clientes
Nombre Potenciales Clientes
Direccion Potenciales Clientes
Correo Electronico
Preferencias
(1,N)
Contacta
N:M
Fecha Contacto
Se presenta
1:M
Coleccionado por
1:M
Crea
1:M
Coleccionistas
Nombre Coleccionista
Direccion Coleccionista
(1,M)
(1,1)
Colecciona
1:M
Obras de Arte
Titulo Obra
Nombre Artista
Año Creacion
Tipo
(1:M)
(1,M)
(1,N)
(1,1)
Precio Solicitado
Estado
Tamaño
Estilo
Medio
Se exhibe
N:M
Exposiciones
Titulo Exposicion
Fecha Inicio Exposicion
Fecha Termino Exposicion
Tema Exposicion
Tipo Exposicion
Nombre Artista Destacado
(1,M)
(1:M)
(1,M)
Esta a la
1:1
Vendedor
Nombre Vendedor
Direccion Vendedor
RUN Vendedor
(1,1)
1:M
Realiza
Venta
Fecha Venta
Descripcion Venta
Numero Venta
Precio Venta Final
Venta Total
Comision
Porcentaje Comision
(1,M)
(1,1)
(1,M)
M:1
Vendido a un
Cliente-Comprador
Nombre Cliente
Direccion Cliente
RUN Cliente
Telefono Cliente
Correo Electronico Cliente
Total Compras a la Fecha
Total Compras Año Pasado
Recibo
(1,1)

Modelo MR



Se pide

Dado el modelo, usted debe:

- Usar los siguientes operadores en consultas de lectura (SELECT):
  - MIN y MAX
  - AVG
  - CASE
  - BETWEEN
  - IN y NOT IN
  - ANY y ALL
  - EXISTS y NOT EXISTS
  - NULL y NOT NULL

Solución

MIN y MAX

MIN devuelve el valor más pequeño de la columna seleccionada.

Ejemplo: Calcular el mínimo valor del campo TOTAL_VENTAS_ANIO_ANTERIOR de la tabla ARTISTA_TOTALES_VENTAS.

```
SELECT
    MIN(TOTAL_VENTAS_ANIO_ANTERIOR)
        MIN_TOTAL_VENTAS_ANIO_ANTERIOR
FROM 'ARTISTA_TOTALES_VENTAS';
```

MAX devuelve el valor más grande de la columna seleccionada.

Ejemplo: Calcular el máximo valor del campo TOTAL_VENTAS_ANIO_ANTERIOR de la tabla ARTISTA_TOTALES_VENTAS.

```
SELECT
    MAX(TOTAL_VENTAS_ANIO_ANTERIOR)
        MIN_TOTAL_VENTAS_ANIO_ANTERIOR
FROM 'ARTISTA_TOTALES_VENTAS';
```

Observación: si queremos calcular MIN o MAX en una columna seleccionada de varias, debe agregarse el operador GROUP BY con las columnas que no usadas para el calculo de MIN o MAX.

AVG

AVG devuelve el valor promedio de una columna numérica.

Ejemplo: Calcular el promedio del campo TOTAL_VENTAS_ANIO_ANTERIOR de la tabla ARTISTA_TOTALES_VENTAS.

```
SELECT
    AVG(TOTAL_VENTAS_ANIO_ANTERIOR)
        MIN_TOTAL_VENTAS_ANIO_ANTERIOR
FROM 'ARTISTA_TOTALES_VENTAS'
```

Observación: si queremos calcular AVG en una columna seleccionada de varias, debe agregarse el operador GROUP BY con las columnas que no usadas para el calculo de AVG.

CASE

CASE es un tipo de sentencia parecido a IF-THEN-ELSE, el cual se puede evaluar una o varias condiciones, y la coincidencia devuelve un valor.

Entonces, una vez que una condición es verdadera, dejará de leer y devolverá el resultado. Si no hay condiciones verdaderas, devuelve el valor en la cláusula ELSE (última condición).

Si no hay otra parte y no hay condiciones verdaderas, devuelve NULL.

Ejemplo: Dado los campos NOMBRE_ARTISTA, TOTAL_VENTAS_ANUAL y TOTAL_VENTAS_ANIO_ANTERIOR de la tabla ARTISTA_TOTALES_VENTAS, realizar una comparación con las siguiente condiciones y valor a entregar: si TOTAL_VENTAS_ANUAL >TOTAL_VENTAS_ANIO_ANTERIOR, el resultado es: VENTAS_ALTAS; si TOTAL_VENTAS_ANUAL <TOTAL_VENTAS_ANIO_ANTERIOR, el resultado es: VENTAS_BAJAS, de lo contrario VENTAS_IGUALES. El resultado de la query debe estar ordenado por NOMBRE_ARTISTA.

```
SELECT
    NOMBRE_ARTISTA,
    TOTAL_VENTAS_ANUAL,
    TOTAL_VENTAS_ANIO_ANTERIOR,
    CASE
        WHEN TOTAL_VENTAS_ANUAL >
            TOTAL_VENTAS_ANIO_ANTERIOR THEN 'VENTAS
            ALTAS'
        WHEN TOTAL_VENTAS_ANUAL <
            TOTAL_VENTAS_ANIO_ANTERIOR THEN 'VENTAS
            BAJAS'
        ELSE 'VENTAS IGUALES'
    END AS 'COMPARACION VENTAS'
FROM `ARTISTA_TOTALES_VENTAS`
ORDER BY NOMBRE_ARTISTA;
```

BETWEEN

BETWEEN selecciona valores dentro de un rango dado. Los valores pueden ser números, texto o fechas. Este último valor es el más usado por BETWEEN.

BETWEEN es inclusivo: se incluyen valores iniciales y finales.

Un ejemplo: Mostrar los campos RUN_VENDEDOR, FECHA_VENTA y VENTA_FINAL de la tabla VENTA, el campo NOMBRE_VENDEDOR de la tabla VENDEDOR, el campo NOMBRE_CLIENTE de la tabla CLIENTE_COMPRADOR y el campo TOTAL_COMPRAS_A_LA_FECHA de la tabla CLIENTES_TOTALES. Donde el rango de FECHA_VENTA es entre el 20 de abril del 2020 con 00:00:00 horas y 20 de abril del 2020 con 23:59:59 horas.

```
SELECT
    V.RUN_VENDEDOR,
    V.FECHA_VENTA,
    V.VENTA_FINAL,
    VE.NOMBRE_VENDEDOR,
    CC.NOMBRE_CLIENTE,
    CT.TOTAL_COMPRAS_A_LA_FECHA
FROM `VENTA` V
JOIN VENDEDOR VE ON V.RUN_VENDEDOR = VE.RUN_VENDEDOR
JOIN CLIENTE_COMPRADOR CC ON CC.RUN_CLIENTE = V.
    RUN_CLIENTE
JOIN CLIENTES_TOTALES CT ON CT.RUN_CLIENTE = CC.
    RUN_CLIENTE
WHERE V.FECHA_VENTA BETWEEN '2020-04-20 00:00:00' AND '
    2020-04-20 23:59:59'
```

IN y NOT IN

IN le permite especificar múltiples valores en una cláusula WHERE.

IN es una abreviatura para múltiples condiciones OR.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA de la tabla ARTISTAS, los campos TITULO_EXPOSICION y TEMA_EXPOSICION de la tabla EXPOSICIONES. Donde los NOMBRES_ARTISTA a mostrar son PABLO PICASSO y LEONARDO DA VINCI.

```
SELECT
    A.NOMBRE_ARTISTA,
    E.TITULO_EXPOSICION,
    E.TEMA_EXPOSICION
```

```
FROM `ARTISTAS` A
JOIN EXPOSICIONES E ON E.NOMBRE_ARTISTA = A.
    NOMBRE_ARTISTA
WHERE A.NOMBRE_ARTISTA IN ('PABLO PICASSO','LEONARDO DA
     VINCI');
```

NOT IN se usa cuando no queremos o queremos excluir de un listado algunos valores.

NOT IN tiene la misma sintaxis que IN.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA de la tabla ARTISTAS, los campos TITULO_EXPOSICION y TEMA_EXPOSICION de la tabla EXPOSICIONES. Donde los NOMBRES_ARTISTA a mostrar no son PABLO PICASSO y LEONARDO DA VINCI.

```
SELECT
     A.NOMBRE_ARTISTA,
     E.TITULO_EXPOSICION,
     E.TEMA_EXPOSICION
FROM `ARTISTAS` A
JOIN EXPOSICIONES E ON E.NOMBRE_ARTISTA = A.
    NOMBRE_ARTISTA
WHERE A.NOMBRE_ARTISTA NOT IN ('PABLO PICASSO','
    LEONARDO DA VINCI');
```

ANY y ALL

ANY y ALL se usan con una cláusula WHERE.

ANY devuelve verdadero si alguno de los valores de subconsulta cumple la condición.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA y CONOCIDO de la tabla ARTISTAS, y los campos TOTAL_VENTAS_ANUAL y TOTAL_VENTAS_ANIO_ANTERIOR de la tabla ARTISTA_TOTALES_VENTAS. Donde algunos de los NOMBRE_ARTISTA hayan realizado una EXPOSICION con FECHA_INICIO_EXPOSICION entre 19 de abril del 2018 y 19 de abril del 2020.

```
SELECT
          A.NOMBRE_ARTISTA,
```

```
        A.CONOCIDO,
        ATV.TOTAL_VENTAS_ANUAL,
        ATV.TOTAL_VENTAS_ANIO_ANTERIOR
FROM ARTISTAS A
JOIN ARTISTA_TOTALES_VENTAS ATV ON ATV.NOMBRE_ARTISTA =
    A.NOMBRE_ARTISTA
WHERE A.NOMBRE_ARTISTA = ANY
(
        SELECT E.NOMBRE_ARTISTA
        FROM EXPOSICIONES E
        WHERE E.FECHA_INICIO_EXPOSICION BETWEEN '
            19-04-2018' AND '19-04-2020'
        GROUP BY E.NOMBRE_ARTISTA
);
```

ALL devuelve verdadero si todos los valores de subconsulta cumplen la condición.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA y CONOCIDO de la tabla ARTISTAS, y los campos TOTAL_VENTAS_ANUAL y TOTAL_VENTAS_ANIO_ANTERIOR de la tabla ARTISTA_TOTALES_VENTAS. Donde todos los NOMBRE_ARTISTA hayan realizado una EXPOSICION con FECHA_INICIO_EXPOSICION entre 19 de abril del 2018 y 19 de abril del 2020.

```
SELECT
        A.NOMBRE_ARTISTA,
        A.CONOCIDO,
        ATV.TOTAL_VENTAS_ANUAL,
        ATV.TOTAL_VENTAS_ANIO_ANTERIOR
FROM ARTISTAS A
JOIN ARTISTA_TOTALES_VENTAS ATV ON ATV.NOMBRE_ARTISTA =
    A.NOMBRE_ARTISTA
WHERE A.NOMBRE_ARTISTA = ALL
(
        SELECT E.NOMBRE_ARTISTA
        FROM EXPOSICIONES E
        WHERE E.FECHA_INICIO_EXPOSICION BETWEEN '
```

```
            19-04-2018' AND '19-04-2020'
        GROUP BY E.NOMBRE_ARTISTA
);
```

EXISTS y NOT EXISTS

EXISTS se usa para probar la existencia de cualquier registro en una subconsulta.

EXISTS devuelve verdadero si la subconsulta devuelve uno o más registros.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA, TITULO_OBRA y ANIO_CREACION de la tabla OBRAS_ARTE, y el campo TITULO_EXPOSICION de la tabla SE_EXHIBE. Donde existan un NUMERO_VENTA de VENTA entre las FECHA_VENTA desde 1 de enero del 2020 al 31 de diciembre del 2020.

```
SELECT
        OA.NOMBRE_ARTISTA,
        OA.TITULO_OBRA,
        OA.ANIO_CREACION,
        SE.TITULO_EXPOSICION
FROM OBRAS_ARTE OA
LEFT JOIN SE_EXHIBE SE ON SE.NOMBRE_ARTISTA = OA.
    NOMBRE_ARTISTA AND SE.TITULO_OBRA = OA.TITULO_OBRA
WHERE EXISTS
(
        SELECT V.NUMERO_VENTA
        FROM `VENTA` V
        WHERE V.FECHA_VENTA BETWEEN '2020-01-01' AND '
            2020-12-31'
        GROUP BY V.NUMERO_VENTA
);
```

NOT EXISTS es lo opuesto a EXISTS, donde se prueba la no existencia de cualquier registro en una subconsulta.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA, TITULO_OBRA y ANIO_CREACION de la tabla OBRAS_ARTE, y el campo TITULO_EXPOSICION

de la tabla SE_EXHIBE. Donde no existan un NUMERO_VENTA de VENTA entre las FECHA_VENTA desde 1 de enero del 2020 al 31 de diciembre del 2020.

```
SELECT
        OA.NOMBRE_ARTISTA,
        OA.TITULO_OBRA,
        OA.ANIO_CREACION,
        SE.TITULO_EXPOSICION
FROM OBRAS_ARTE OA
LEFT JOIN SE_EXHIBE SE ON SE.NOMBRE_ARTISTA = OA.
    NOMBRE_ARTISTA AND SE.TITULO_OBRA = OA.TITULO_OBRA
WHERE NOT EXISTS
(
        SELECT V.NUMERO_VENTA
        FROM `VENTA` V
        WHERE V.FECHA_VENTA BETWEEN '2020-01-01' AND '
            2020-12-31'
        GROUP BY V.NUMERO_VENTA
);
```

NULL y NOT NULL

Un campo con un valor NULL es un campo sin valor.

Si un campo en una tabla es opcional, es posible insertar un nuevo registro o actualizar un registro sin agregar un valor a este campo. Luego, el campo se guardará con un valor NULL.

Es importante saber que un valor NULL es diferente de un valor cero o un campo que contiene espacios. Un campo con un valor NULL es uno que se ha dejado sin un valor durante la creación del registro.

No es posible probar valores NULL con operadores de comparación, como =, <, >o <>.

Tendremos que usar los operadores IS NULL y IS NOT NULL en su lugar.

IS NULL busca campos con valores nulos, se usa en el operador WHERE.

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA, NUMERO_REGISTRO_CONTRIB
TELEFONO_ARTISTA de la tabla ARTISTAS. Donde el TELEFONO_ARTISTA

sean nulos.

```
SELECT
    NOMBRE_ARTISTA,
    NUMERO_REGISTRO_CONTRIBUYENTE,
    TELEFONO_ARTISTA
FROM ARTISTAS
WHERE TELEFONO_ARTISTA IS NULL;
```

IS NOT NULL se usa para probar valores no vacíos (valores NOT NULL).

Un ejemplo: Mostrar los campos NOMBRE_ARTISTA, NUMERO_REGISTRO_CONTRIB
TELEFONO_ARTISTA de la tabla ARTISTAS. Donde el TELEFONO_ARTISTA
no sean nulos.

```
SELECT
    NOMBRE_ARTISTA,
    NUMERO_REGISTRO_CONTRIBUYENTE,
    TELEFONO_ARTISTA
FROM ARTISTAS
WHERE TELEFONO_ARTISTA IS NOT NULL;
```