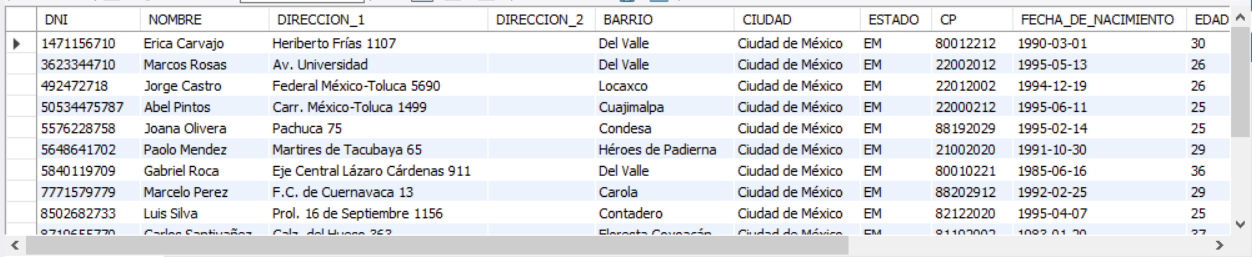
7) Conociendo cómo es nuestra base, podemos hacer nuestras consultas. Selecciona un nuevo script de SQL, con la base de datos seleccionada, y digita:

USE jugos\_ventas;

SELECT DNI, NOMBRE, DIRECCION\_1, DIRECCION\_2, BARRIO, CIUDAD, ESTADO,

CP, FECHA\_DE\_NACIMIENTO, EDAD, SEXO, LIMITE\_DE\_CREDITO, VOLUMEN\_DE\_COMPRA,

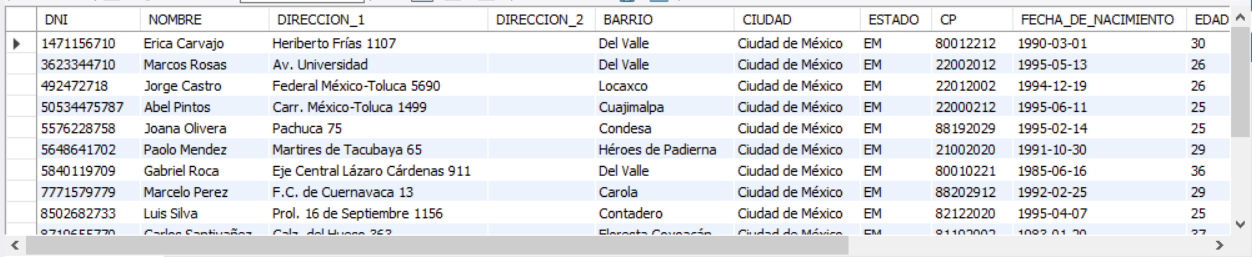
PRIMERA\_COMPRA FROM tabla\_de\_clientes;



Aquí veremos todos los campos de la tabla **tabla\_de\_clientes**. Esto porque los campos fueron seleccionados uno a uno.

8) Digita a continuación:

SELECT \* FROM tabla\_de\_clientes;



Este resultado fue igual al de la consulta anterior. Ello porque al colocar \* estamos seleccionando todos los campos.

9) Digita:

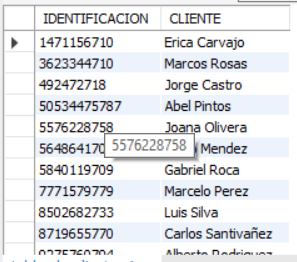
SELECT DNI, NOMBRE FROM tabla\_de\_clientes;



Ahora podemos ver que no es necesario seleccionar todos los campos de una tabla. Basta destacar los campos que serán vistos.

10) Digita:

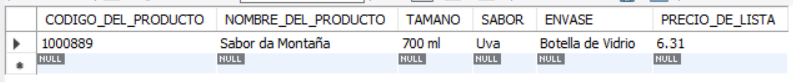
SELECT DNI AS IDENTIFICACION, NOMBRE AS CLIENTE FROM tabla\_de\_clientes;



No siempre el nombre original de la columna es el nombre que queremos que sea retornado por la consulta. Por ello, podemos crear Alias (Sobrenombres) para los campos escribiendo algo después del comando AS.

11) Podemos filtrar nuestra consulta. Digita:

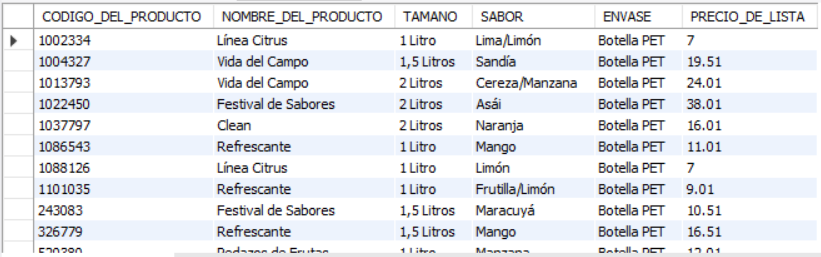
SELECT \* FROM tabla\_de\_productos WHERE SABOR = 'Uva';



SELECT \* FROM tabla\_de\_productos WHERE SABOR = 'Mango';

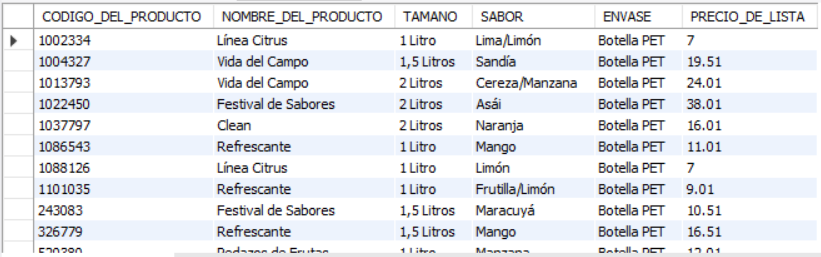


SELECT \* FROM tabla\_de\_productos WHERE ENVASE = 'Botella PET';



El resultado es el mismo si se escribe en mayúscula o en minúscula ya que MySQL no es case sensitive:

SELECT \* FROM tabla\_de\_productos WHERE ENVASE = 'botella pet';



Los filtros usados retornan todos los registros de la tabla donde se cumple lo especificado por la cláusula WHERE. Podemos usar cualquier columna como criterio.

12) Existen comandos de filtro aplicados a valores:

SELECT \* FROM tabla\_de\_productos WHERE PRECIO\_DE\_LISTA > 16;



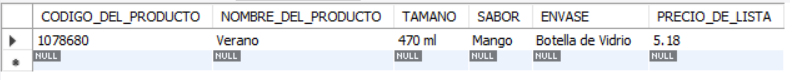
SELECT \* FROM tabla\_de\_productos WHERE PRECIO\_DE\_LISTA BETWEEN 16 AND 16.02;



En este caso podemos usar >, >=, <, <=, =, <> y BETWEEN. Así, podemos aplicar filtros sobre los valores para obtener diversos resultados.

13) Es posible aplicar consultas condicionales usando los operadores AND y OR. El output va a depender del significado de AND y OR en una estructura lógica. Digita:

SELECT \* FROM tabla\_de\_productos WHERE SABOR='mango' AND TAMANO='470 ml';



A causa del operador AND, el output solamente ocurrirá cuando las dos condiciones se cumplan en el mismo registro de la tabla.

14) Digita:

SELECT \* FROM tabla\_de\_productos WHERE SABOR='mango' OR TAMANO='470 ml';



Aquí obtuvimos un filtro (Sabor = Mango) u otro ( Tamaño = 470 ml). Esto porque usamos el operador OR.

15) Podemos usar parte de un texto para ser empleado como criterio de localización de registros de la tabla. Digita a continuación:

SELECT \* FROM tabla\_de\_productos WHERE SABOR LIKE '%manzana';



Aquí buscaremos todos los registros cuyo sabor contenga la palabra Manzana, pero solamente al final del texto, ya que el signo % precede el texto Manzana.

16) Podemos mezclar condiciones LIKE con otras. Digita:

SELECT \* FROM tabla\_de\_productos WHERE SABOR LIKE '%manzana'

AND ENVASE = 'Botella PET';



Finalmente, obtuvimos el resultado de la consulta del texto “Manzana” tan solo para envases de tipo “Botella PET”.

//

¿Cuál (o cuáles) fue (fueron) la(s) mayor(es) venta(s) del producto “Refrescante, 1 Litro, Frutilla/Limón”, en cantidad? (Obtenga este resultado utilizando 2 comandos SQL).

VER OPINIÓN DEL INSTRUCTORpinión del instructor

Las consultas que debemos realizar son las siguientes:

SELECT CODIGO\_DEL\_PRODUCTO FROM tabla\_de\_productos

WHERE NOMBRE\_DEL\_PRODUCTO = "Refrescante" AND TAMANO = "1 Litro"

AND SABOR = "Frutilla/Limón";

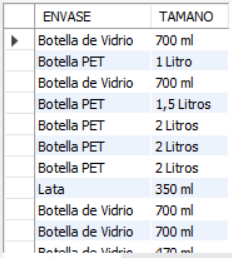
Haga lo que hicimos en aula

[**PRÓXIMA ACTIVIDAD**](https://app.aluracursos.com/course/consultas-sql-mysql/task/76390/next)

Llegó la hora de que sigas todos los pasos realizados por mí durante esta aula. En caso de que ya lo hayas hecho, excelente. Si aún no lo hiciste, es importante que ejecutes lo que fue visto en los videos para poder continuar con la siguiente aula.

1) De nuevo en Workbench, vamos a ver formas diferentes de exhibir los resultados. Digita:

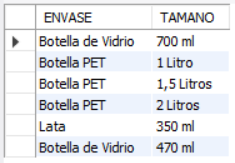
SELECT ENVASE, TAMANO FROM tabla\_de\_productos;



Observa que tenemos registros donde el conjunto ENVASE / TAMAÑO se repite.

2) Ahora digita el comando:

SELECT DISTINCT ENVASE, TAMANO FROM tabla\_de\_productos;

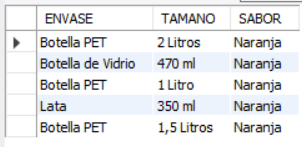


El simple hecho de incluir la cláusula DISTINCT hace que los registros no se repitan.

3) Podemos aplicar filtros a la selección con DISTINCT e incluso añadir más campos.

SELECT DISTINCT ENVASE, TAMANO, SABOR FROM tabla\_de\_productos

WHERE SABOR = 'Naranja';



4) Podemos limitar el número de registros exibidos en el output. Digita:

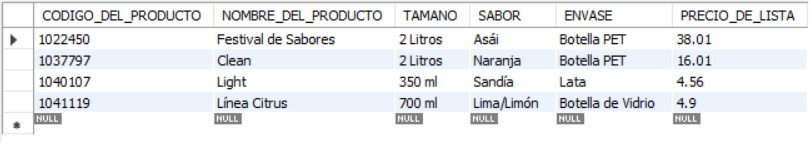
SELECT \* FROM tabla\_de\_productos LIMIT 5;



El output está limitado a los primeros 5 registros.

5) Podemos exhibir los registros dentro de un intervalo de filas. Digita:

SELECT \* FROM tabla\_de\_productos LIMIT 5,4;



6) El output de un comando SELECT puede ser presentado de forma ordenada. Observa:

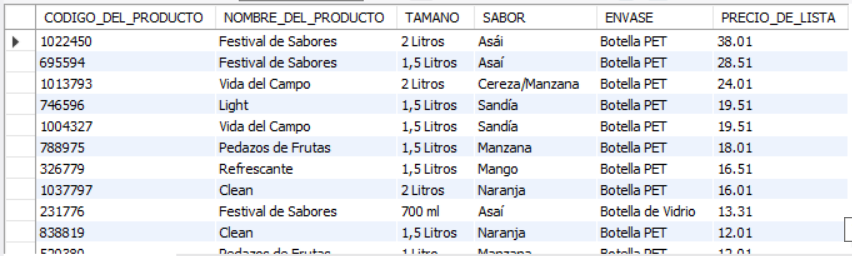
SELECT \* FROM tabla\_de\_productos ORDER BY PRECIO\_DE\_LISTA;



Tenemos los valores ordenados por precio de lista, de menor a mayor.

7) Podemos cambiar este orden. Digita:

SELECT \* FROM tabla\_de\_productos ORDER BY PRECIO\_DE\_LISTA DESC;



8) Los valores pueden ser ordenados alfabéticamente cuando incluimos un campo de texto en el criterio de ordenamiento. Digita:

SELECT \* FROM tabla\_de\_productos ORDER BY NOMBRE\_DEL\_PRODUCTO;



9) También, en el criterio de ordenamiento del tipo texto, podemos cambiar el orden de mayor a menor. Digita:

SELECT \* FROM tabla\_de\_productos ORDER BY NOMBRE\_DEL\_PRODUCTO DESC;



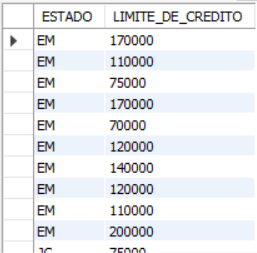
10) El criterio de ordenamiento puede ser diferente para cada tipo. Observa el ejemplo a continuación donde usamos dos campos como criterio de ordenamiento y un orden diferente para cada uno de ellos:

SELECT \* FROM tabla\_de\_productos ORDER BY ENVASE DESC, NOMBRE\_DEL\_PRODUCTO ASC;



11) Los datos pueden ser agrupados. Cuando esto sucede, tenemos que aplicar um criterio de agrupamiento para los campos numéricos. Podemos emplear SUM, AVG, MAX, MIN, entre otros. Digita el comando siguiente:

SELECT ESTADO, LIMITE\_DE\_CREDITO FROM tabla\_de\_clientes;

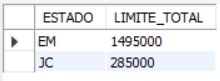


Puedes notar que tenemos varias líneas para EM y JC. ¿Cómo hacemos para sumar todos los límites de crédito para EM y JC?

12) La solución está en el siguiente comando:

SELECT ESTADO, SUM(LIMITE\_DE\_CREDITO) AS LIMITE\_TOTAL

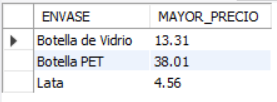
FROM tabla\_de\_clientes GROUP BY ESTADO;



13) Podemos emplear otros criterios como el valor máximo.

SELECT ENVASE, MAX(PRECIO\_DE\_LISTA) AS MAYOR\_PRECIO

FROM tabla\_de\_productos GROUP BY ENVASE;COPIA EL CÓDIGO

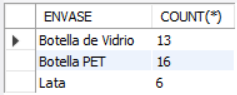


Aquí vemos el mayor precio de lista para cada tipo de envase.

14) El comando COUNT cuenta el número de ocurrencias en la tabla. Digita:

SELECT ENVASE, COUNT(\*) FROM tabla\_de\_productos

GROUP BY ENVASE;

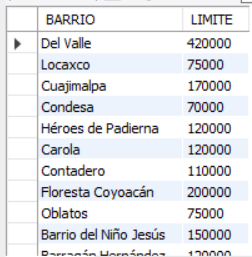


Tenemos el número de productos cuyo envase es botella PET, botella de vidrio y Lata.

15) El filtro puede ser aplicado sobre el agrupamiento, como una consulta cualquiera. Digita:

SELECT BARRIO, SUM(LIMITE\_DE\_CREDITO) AS LIMITE

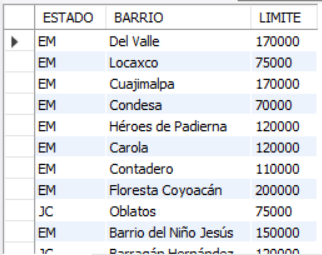
FROM tabla\_de\_clientes GROUP BY BARRIO;



16) Adicionalmente, el agrupamiento puede ser realizado en más de un campo. Digita:

SELECT ESTADO, BARRIO, MAX(LIMITE\_DE\_CREDITO) AS LIMITE

FROM tabla\_de\_clientes GROUP BY ESTADO, BARRIO;



17) Podemos mezclar agrupamiento com ordenamiento. Digita:

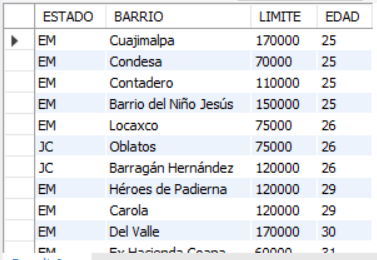
SELECT ESTADO, BARRIO, MAX(LIMITE\_DE\_CREDITO) AS LIMITE,

EDAD FROM tabla\_de\_clientes

WHERE EDAD >=20

GROUP BY ESTADO, BARRIO

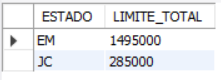
ORDER BY EDAD;



18) Observa la consulta a continuación:

SELECT ESTADO, SUM(LIMITE\_DE\_CREDITO) AS LIMITE\_TOTAL

FROM tabla\_de\_clientes GROUP BY ESTADO;



19) Queremos aplicar un filtro sobre el resultado de esta consulta. Entonces, digita:

SELECT ESTADO, SUM(LIMITE\_DE\_CREDITO) AS LIMITE\_TOTAL

FROM tabla\_de\_clientes WHERE LIMITE\_TOTAL > 300000

GROUP BY ESTADO;COPIA EL CÓDIGO

Nota que la consulta anterior generó un error.

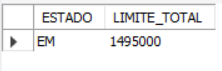
20) Usamos el comando HAVING para filtrar el output de una consulta usando como criterio el valor agrupado. Digita:

SELECT ESTADO, SUM(LIMITE\_DE\_CREDITO) AS LIMITE\_TOTAL

FROM tabla\_de\_clientes

GROUP BY ESTADO

HAVING LIMITE\_TOTAL > 300000;

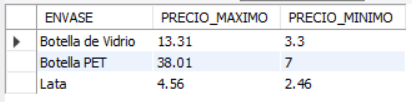


21) El criterio usado con el comando HAVING no necesita ser el mismo usado en el filtro. Observa el siguiente comando:

SELECT ENVASE, MAX(PRECIO\_DE\_LISTA) AS PRECIO\_MAXIMO,

MIN(PRECIO\_DE\_LISTA) AS PRECIO\_MINIMO

FROM tabla\_de\_productos GROUP BY ENVASE;



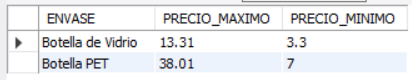
Utiliza el MIN para agrupamiento.

22) Pero, en la siguiente consulta, el criterio del comando HAVING pide la suma. Digita:

SELECT ENVASE, MAX(PRECIO\_DE\_LISTA) AS PRECIO\_MAXIMO,

MIN(PRECIO\_DE\_LISTA) AS PRECIO\_MINIMO

FROM tabla\_de\_productos GROUP BY ENVASE;



23) Al utilizar HAVING podemos usar más de un criterio empleando AND u OR.

SELECT ENVASE, MAX(PRECIO\_DE\_LISTA) AS PRECIO\_MAXIMO,

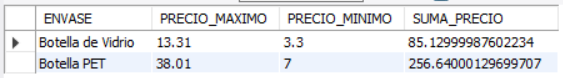
MIN(PRECIO\_DE\_LISTA) AS PRECIO\_MINIMO,

SUM(PRECIO\_DE\_LISTA) AS SUMA\_PRECIO

FROM tabla\_de\_productos GROUP BY ENVASE

HAVING SUM(PRECIO\_DE\_LISTA) >= 80

AND MAX(PRECIO\_DE\_LISTA) >= 5;



24) El comando CASE permite la clasificación de cada registro de la tabla. Digita el comando siguiente:

SELECT NOMBRE\_DEL\_PRODUCTO, PRECIO\_DE\_LISTA,

CASE

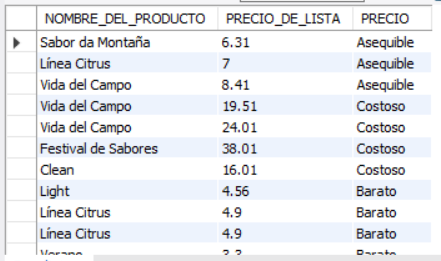
WHEN PRECIO\_DE\_LISTA >= 12 THEN 'Costoso'

WHEN PRECIO\_DE\_LISTA >= 5 AND PRECIO\_DE\_LISTA < 12 THEN 'Asequible'

ELSE 'Barato'

END AS PRECIO

FROM tabla\_de\_productos;



Con CASE fue posible clasificar los productos como Costoso, Barato o Asequible conforme al valor de su precio de lista.

25) Podemos usar el comando CASE como criterio de agrupamiento, Digita el siguiente comando:

SELECT ENVASE, SABOR,

CASE

WHEN PRECIO\_DE\_LISTA >= 12 THEN 'Costoso'

WHEN PRECIO\_DE\_LISTA >= 5 AND PRECIO\_DE\_LISTA < 12 THEN 'Asequible'

ELSE 'Barato'

END AS PRECIO, MIN(PRECIO\_DE\_LISTA) AS PRECIO\_MINIMO

FROM tabla\_de\_productos

WHERE TAMANO = '700 ml'

GROUP BY ENVASE,

CASE

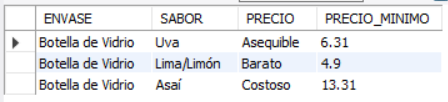
WHEN PRECIO\_DE\_LISTA >= 12 THEN 'Costoso'

WHEN PRECIO\_DE\_LISTA >= 5 AND PRECIO\_DE\_LISTA < 12 THEN 'Asequible'

ELSE 'Barato'

END

ORDER BY ENVASE;



//

 Haga lo que hicimos en aula

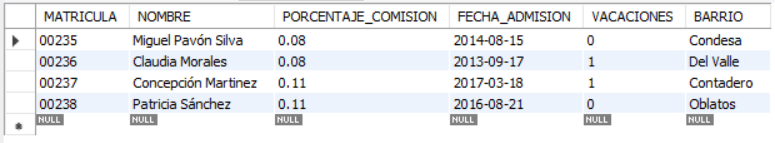
[**PRÓXIMA ACTIVIDAD**](https://app.aluracursos.com/course/consultas-sql-mysql/task/76391/next)

Llegó la hora de que sigas todos los pasos realizados por mí durante esta aula. En caso de que ya lo hayas hecho, excelente. Si aún no lo hiciste, es importante que ejecutes lo que fue visto en los videos para poder continuar con la siguiente aula.

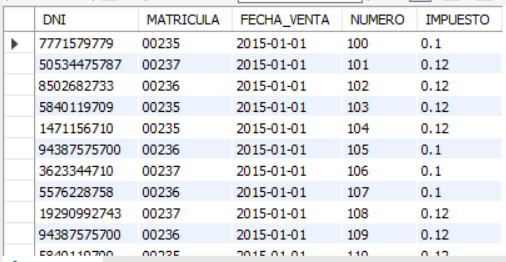
1) Aquí veremos cómo conectar las consultas de tablas diferentes. Esta unión se conoce como JOIN.

2) Observa el contenido de dos tablas digitando los siguientes comandos:

SELECT \* FROM tabla\_de\_vendedores;



SELECT \* FROM facturas;



3) Podemos conectar estas dos tablas por un campo en común (MATRICULA). Digita:

SELECT \* FROM tabla\_de\_vendedores A

INNER JOIN

facturas B

ON A.MATRICULA = B.MATRICULA;



4) Podemos aplicar agrupamientos al resultado de la consulta que conecta una o más tablas:

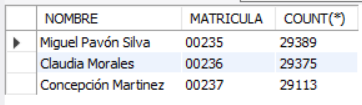
SELECT A.NOMBRE, B.MATRICULA, COUNT(\*) FROM tabla\_de\_vendedores A

INNER JOIN

facturas B

ON A.MATRICULA = B.MATRICULA

GROUP BY A.NOMBRE, B.MATRICULA;



5) No siempre todos los registros pueden ser conectados. Existen otros tipos de JOIN que nos permiten identificar lo que no puede ser conectado. Observa la siguiente consulta:

SELECT count(\*) FROM tabla\_de\_clientes;



Ella muestra que tenemos 15 clientes.

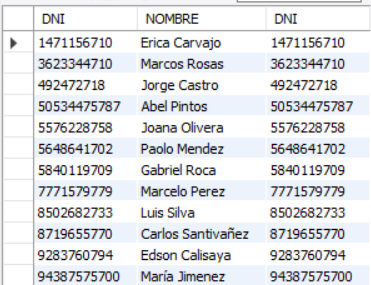
6) Vamos a realizar un JOIN con la tabla de facturas para ver cuántos clientes poseen facturas emitidas. Digita:

SELECT DISTINCT A.DNI, A.NOMBRE, B.DNI FROM tabla\_de\_clientes A

INNER JOIN

facturas B

ON A.DNI = B.DNI;



Si cuentas a los clientes verás que, en la consulta encima, tenemos 12 registros. Existen tres clientes que están registrados pero nunca se les emitió facturas.

7) Podemos usar un LEFT JOIN. Digita:

SELECT DISTINCT A.DNI, A.NOMBRE, A.CIUDAD, B.DNI FROM tabla\_de\_clientes A

LEFT JOIN

facturas B

ON A.DNI = B.DNI;



El cliente que posee el DNI proveniente de la tabla de facturas con el valor nulo, es el cliente a quien nunca se le emitió facturas.

8) La selección correcta sería:

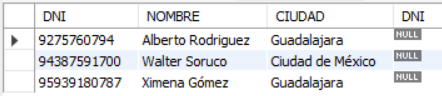
SELECT DISTINCT A.DNI, A.NOMBRE, A.CIUDAD, B.DNI FROM tabla\_de\_clientes A

LEFT JOIN

facturas B

ON A.DNI = B.DNI

WHERE B.DNI IS NULL;

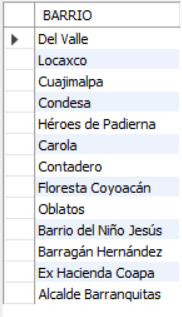


9) Podemos juntar dos o más consultas, Desde que los campos seleccionados sean los mismos. Digita:

SELECT DISTINCT BARRIO FROM tabla\_de\_clientes

UNION

SELECT DISTINCT BARRIO FROM tabla\_de\_vendedores;

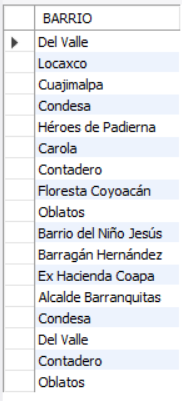


10) El comando UNION ALL no realiza la selección con un DISTINCT. Los registros se repiten si existen en ambas tablas. Digita:

SELECT DISTINCT BARRIO FROM tabla\_de\_clientes

UNION ALL

SELECT DISTINCT BARRIO FROM tabla\_de\_vendedores;



Observa que Del Valle aparece dos veces, al igual que Condesa, Contadero y Oblatos. Una proveniente de la tabla de clientes y la otra proveniente de la tabla de productos.

11) Podemos simular el FULL JOIN, que no es soportado por MySQL, usando el LEFT JOIN y el RIGHT JOIN con UNION. Digita:

SELECT tabla\_de\_clientes.NOMBRE, tabla\_de\_clientes.CIUDAD, tabla\_de\_clientes.BARRIO,

tabla\_de\_vendedores.NOMBRE, VACACIONES

FROM tabla\_de\_clientes

LEFT JOIN

tabla\_de\_vendedores

ON tabla\_de\_clientes.BARRIO = tabla\_de\_vendedores.BARRIO

UNION

SELECT tabla\_de\_clientes.NOMBRE, tabla\_de\_clientes.CIUDAD, tabla\_de\_clientes.BARRIO,

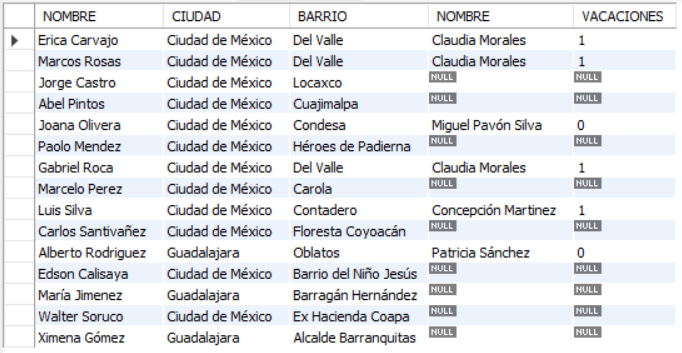
tabla\_de\_vendedores.NOMBRE, VACACIONES

FROM tabla\_de\_clientes

RIGHT JOIN

tabla\_de\_vendedores

ON tabla\_de\_clientes.BARRIO = tabla\_de\_vendedores.BARRIO;



12) Las subconsultas permiten realizar selecciones usando como criterios otras selecciones. Digita:

SELECT \* FROM tabla\_de\_clientes

WHERE BARRIO IN (SELECT DISTINCT BARRIO FROM tabla\_de\_vendedores);



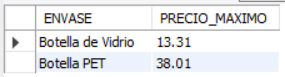
13) Podemos aplicar una consulta sobre otra consulta directamente. Digita:

SELECT X.ENVASE, X.PRECIO\_MAXIMO FROM

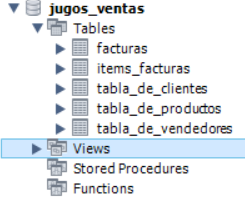
(SELECT ENVASE, MAX(PRECIO\_DE\_LISTA)

AS PRECIO\_MAXIMO FROM tabla\_de\_productos GROUP BY ENVASE) X

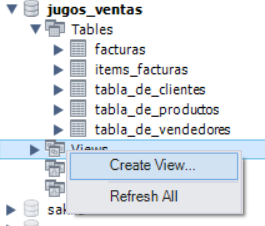
WHERE X.PRECIO\_MAXIMO >=10;



14) Podemos transformar una consulta en una vista (View) que después puede ser usada en otras consultas como una tabla. Crea la vista. Para ello, expande el árbol de la esquina superior izquierda, donde tenemos el nombre de la base, y vaya a Views.



15) Haz clic con el botón derecho del mouse y selecciona la opción Create View…



16) Digita el siguiente comando:

CREATE VIEW ‘vw\_envases\_grandes’

AS SELECT ENVASE, MAX(PRECIO\_DE\_LISTA)

AS PRECIO\_MAXIMO FROM tabla\_de\_productos GROUP BY ENVASE;

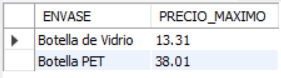
17) Haz clic en Apply y sigue los pasos hasta crear la vista.

18) Podemos manipular la vista como una tabla. Digita:

SELECT X.ENVASE, X. PRECIO\_MAXIMO FROM

vw\_envases\_grandes X

WHERE PRECIO\_MAXIMO >=10;



19) Podemos crear JOINs de tablas con views:

SELECT A.NOMBRE\_DEL\_PRODUCTO, A.ENVASE, A.PRECIO\_DE\_LISTA,

B.PRECIO\_MAXIMO FROM tabla\_de\_productos A

INNER JOIN

vw\_envases\_grandes B

ON A.ENVASE = B.ENVASE;



 Haga lo que hicimos en aula

[**PRÓXIMA ACTIVIDAD**](https://app.aluracursos.com/course/consultas-sql-mysql/task/76392/next)

Llegó la hora de que sigas todos los pasos realizados por mí durante esta aula. En caso de que ya lo hayas hecho, excelente. Si aún no lo hiciste, es importante que ejecutes lo que fue visto en los videos para poder continuar con la siguiente aula.

1) En esta aula veremos ejemplos de funciones.

2) Primero vimos las funciones de tipo texto. Observa algunos ejemplos con sus respectivos outputs:

SELECT LTRIM(" MySQL es fácil") AS RESULTADO;

SELECT RTRIM("MySQL es fácil ") AS RESULTADO;



SELECT TRIM(" MySQL es fácil ") AS RESULTADO;



SELECT CONCAT("MySQL es fácil,", " no lo crees?") AS RESULTADO;

SELECT UPPER("mysql es una base de datos interesante.") AS RESULTADO;

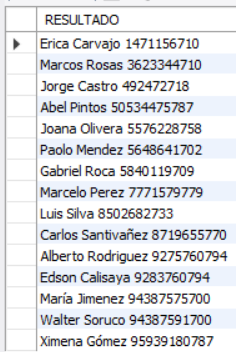


SELECT LOWER("MYSQL ES UNA BASE DE DATOS INTERESANTE.") AS RESULTADO;



SELECT SUBSTRING("mysql es una base de datos interesante.", 14,4) AS RESULTADO;



SELECT CONCAT(NOMBRE, " ", DNI) AS RESULTADO FROM tabla\_de\_clientes;

3) Tenemos las funciones de fechas. Ejecuta los siguientes comandos:

SELECT CURDATE();



SELECT CURRENT\_TIMESTAMP();



SELECT YEAR(CURRENT\_TIMESTAMP());

SELECT MONTH(CURRENT\_TIMESTAMP());

COPIA EL CÓDIGO



SELECT DAY(CURRENT\_TIMESTAMP());



SELECT MONTHNAME(CURRENT\_TIMESTAMP());

SELECT DAYNAME(CURRENT\_TIMESTAMP());



SELECT DATEDIFF(CURRENT\_TIMESTAMP(), '2021-01-01') AS RESULTADO;



SELECT DATEDIFF(CURRENT\_TIMESTAMP(), '1984-06-20') AS RESULTADO;



SELECT current\_timestamp() AS DIA\_HOY,

DATE\_SUB(current\_timestamp(), INTERVAL 1 MONTH) AS RESULTADO;

SELECT DISTINCT FECHA\_VENTA,

DAYNAME(FECHA\_VENTA) AS DIA, MONTHNAME(FECHA\_VENTA) AS MES,

YEAR(FECHA\_VENTA) AS AÑO FROM facturas;

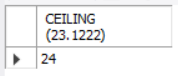


4) Tenemos algunos ejemplos de funciones matemáticas:

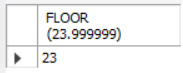
SELECT (34+346-67)/15 \* 29 AS RESULTADO;



SELECT CEILING (23.1222);



SELECT FLOOR (23.999999);



SELECT RAND() AS RESULTADO;



SELECT ROUND(254.545,2);



SELECT ROUND(254.545,1);



SELECT NUMERO, CANTIDAD, PRECIO,

CANTIDAD \* PRECIO AS FACTURACIÓN FROM items\_facturas;



SELECT NUMERO, CANTIDAD, PRECIO,

ROUND(CANTIDAD \* PRECIO,2) AS FACTURACIÓN FROM items\_facturas;



5) Tenemos también funciones de conversión. Ejecuta los siguientes ejemplos:

SELECT CURRENT\_TIMESTAMP() AS RESULTADO;



SELECT CONCAT("La fecha y la hora de hoy son: ", CURRENT\_TIMESTAMP()) AS RESULTADO;



SELECT CONCAT("La fecha y el horario son: ",

DATE\_FORMAT(CURRENT\_TIMESTAMP(),"%W, %d/%m/%Y a las %T" )) AS RESULTADO;



SELECT CONVERT(23.45, CHAR) AS RESULTADO;



SELECT SUBSTRING(CONVERT(23.45, CHAR),3,1) AS RESULTADO;



En esta aula construimos un informe que presentó a los clientes que tenían ventas inválidas. Complementa este informe listando solamente a los que tuvieron ventas inválidas y calcula la diferencia entre el límite de venta máximo y la cantidad vendida en porcentuales.

Tips: Utiliza el comando SQL empleado al final del video.

* Filtra solamente las líneas donde: (A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA) < 0
* Lista el campo X.CANTIDAD\_LIMITE
* Crea nuevo campo ejecutando la fórmula: (1 - (X.CANTIDAD\_LIMITE/X.CANTIDAD\_VENDAS)) \* 100.

VER OPINIÓN DEL INSTRUCTOR

Este desafío puede tener diversas respuestas. A continuación te muestro una de ellas:

SELECT A.DNI, A.NOMBRE, A.MES\_AÑO,

A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA AS DIFERENCIA,

CASE

WHEN (A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA) <= 0 THEN 'Venta Válida'

ELSE 'Venta Inválida'

END AS STATUS\_VENTA, ROUND((1 - (A.CANTIDAD\_MAXIMA/A.CANTIDAD\_VENDIDA)) \* 100,2) AS PORCENTAJE

FROM(

SELECT F.DNI, TC.NOMBRE, DATE\_FORMAT(F.FECHA\_VENTA, "%m - %Y") AS MES\_AÑO,

SUM(IFa.CANTIDAD) AS CANTIDAD\_VENDIDA,

MAX(VOLUMEN\_DE\_COMPRA)/10 AS CANTIDAD\_MAXIMA

FROM facturas F

INNER JOIN

items\_facturas IFa

ON F.NUMERO = IFa.NUMERO

INNER JOIN

tabla\_de\_clientes TC

ON TC.DNI = F.DNI

GROUP BY

F.DNI, TC.NOMBRE, DATE\_FORMAT(F.FECHA\_VENTA, "%m - %Y"))A

WHERE (A.CANTIDAD\_MAXIMA - A.CANTIDAD\_VENDIDA) < 0;

En esta aula construimos un informe que presentó a los clientes que tenían ventas inválidas. Ahora lista solamente a los que tuvieron ventas inválidas en el año 2018 excediendo más del 50% de su límite permitido por mes. Calcula la diferencia entre el límite de venta máximo y la cantidad vendida, en porcentuales.

Tips:

Te puedes apoyar en el código que realizaste para el desafío anterior.

VER OPINIÓN DEL INSTRUCTOROel instructor

Este desafío puede tener diversas respuestas. A continuación te muestro una de ellas:

SELECT A.DNI, A.NOMBRE, A.MES\_AÑO,

A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA AS DIFERENCIA,

CASE

WHEN (A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA) <= 0 THEN 'Venta Válida'

ELSE 'Venta Inválida'

END AS STATUS\_VENTA, ROUND((1 - (A.CANTIDAD\_MAXIMA/A.CANTIDAD\_VENDIDA)) \* 100,2) AS PORCENTAJE

FROM(

SELECT F.DNI, TC.NOMBRE, DATE\_FORMAT(F.FECHA\_VENTA, "%m - %Y") AS MES\_AÑO,

SUM(IFa.CANTIDAD) AS CANTIDAD\_VENDIDA,

MAX(VOLUMEN\_DE\_COMPRA)/10 AS CANTIDAD\_MAXIMA

FROM facturas F

INNER JOIN

items\_facturas IFa

ON F.NUMERO = IFa.NUMERO

INNER JOIN

tabla\_de\_clientes TC

ON TC.DNI = F.DNI

GROUP BY

F.DNI, TC.NOMBRE, DATE\_FORMAT(F.FECHA\_VENTA, "%m - %Y"))A

WHERE (A.CANTIDAD\_MAXIMA - A.CANTIDAD\_VENDIDA) < 0 AND ROUND((1 - (A.CANTIDAD\_MAXIMA/A.CANTIDAD\_VENDIDA)) \* 100,2) > 50

AND A.MES\_AÑO LIKE "%2018";

Modifica el informe pero ahora para ver el ranking de las ventas por tamaño.

Tips:

Puede parecer difícil pero este es el ejercicio más fácil de resolver.

VER OPINIÓN DEL INSTRUCTOR**Opininstructor**

Lo único que debes hacer es tomar el informe, y en vez de utilizar el sabor como parámetro, utilizas tamaño. El restante de la consulta permanece igual:

SELECT VENTAS\_TAMANO.TAMANO, VENTAS\_TAMANO.AÑO, VENTAS\_TAMANO.CANTIDAD\_TOTAL,

ROUND((VENTAS\_TAMANO.CANTIDAD\_TOTAL/VENTA\_TOTAL.CANTIDAD\_TOTAL)\*100,2)

AS PORCENTAJE FROM (

SELECT P.TAMANO, SUM(IFa.CANTIDAD) AS CANTIDAD\_TOTAL,

YEAR(F.FECHA\_VENTA) AS AÑO FROM

tabla\_de\_productos P

INNER JOIN

items\_facturas IFa

ON P.CODIGO\_DEL\_PRODUCTO = IFa.

INNER JOIN

facturas F

ON F.NUMERO = IFa.NUMERO

WHERE YEAR(F.FECHA\_VENTA) = 2016

GROUP BY P.TAMANO, YEAR(F.FECHA\_VENTA)

ORDER BY SUM(IFa.CANTIDAD) DESC) VENTAS\_TAMANO

INNER JOIN (

SELECT SUM(IFa.CANTIDAD) AS CANTIDAD\_TOTAL,

YEAR(F.FECHA\_VENTA) AS AÑO FROM

tabla\_de\_productos P

INNER JOIN

items\_facturas IFa

ON P.CODIGO\_DEL\_PRODUCTO = IFa.

INNER JOIN

facturas F

ON F.NUMERO = IFa.NUMERO

WHERE YEAR(F.FECHA\_VENTA) = 2016

GROUP BY YEAR(F.FECHA\_VENTA)) VENTA\_TOTAL

ON VENTA\_TOTAL.AÑO = VENTAS\_TAMANO.AÑO

ORDER BY VENTAS\_TAMANO.CANTIDAD\_TOTAL DESC;

SELECT VENTAS\_TAMANO.TAMANO, VENTAS\_TAMANO.AÑO, VENTAS\_TAMANO.CANTIDAD\_TOTAL,

ROUND((VENTAS\_TAMANO.CANTIDAD\_TOTAL/VENTA\_TOTAL.CANTIDAD\_TOTAL)\*100,2)

AS PORCENTAJE FROM (

SELECT P.TAMANO, SUM(IFa.CANTIDAD) AS CANTIDAD\_TOTAL,

YEAR(F.FECHA\_VENTA) AS AÑO FROM

tabla\_de\_productos P

INNER JOIN

items\_facturas IFa

ON P.CODIGO\_DEL\_PRODUCTO = IFa.

INNER JOIN

facturas F

ON F.NUMERO = IFa.NUMERO

WHERE YEAR(F.FECHA\_VENTA) = 2016

GROUP BY P.TAMANO, YEAR(F.FECHA\_VENTA)

ORDER BY SUM(IFa.CANTIDAD) DESC) VENTAS\_TAMANO

INNER JOIN (

SELECT SUM(IFa.CANTIDAD) AS CANTIDAD\_TOTAL,

YEAR(F.FECHA\_VENTA) AS AÑO FROM

tabla\_de\_productos P

INNER JOIN

items\_facturas IFa

ON P.CODIGO\_DEL\_PRODUCTO = IFa.INNER JOIN

facturas F

ON F.NUMERO = IFa.NUMERO

WHERE YEAR(F.FECHA\_VENTA) = 2016

GROUP BY YEAR(F.FECHA\_VENTA)) VENTA\_TOTAL

ON VENTA\_TOTAL.AÑO = VENTAS\_TAMANO.AÑO

ORDER BY VENTAS\_TAMANO.CANTIDAD\_TOTAL DESC;

 Haga lo que hicimos en aula

[**PRÓXIMA ACTIVIDAD**](https://app.aluracursos.com/course/consultas-sql-mysql/task/76393/next)

Llegó la hora de que sigas todos los pasos realizados por mí durante esta aula. En caso de que ya lo hayas hecho, excelente. Si aún no lo hiciste, es importante que ejecutes lo que fue visto en los videos para poder continuar con la siguiente aula.

1) Vamos a poner en práctica nuestro conocimiento.

2) Primero, generamos una selección que determina si las ventas mensuales por cliente son válidas o no. Consideramos válidas las ventas por debajo de la cantidad límite e inválidas por encima de la cantidad límite existente en el registro del cliente. La consulta se muestra a continuación:

SELECT A.DNI, A.NOMBRE, A.MES\_AÑO,

A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA AS DIFERENCIA,

CASE

WHEN (A.CANTIDAD\_VENDIDA - A.CANTIDAD\_MAXIMA) <= 0 THEN 'Venta Válida'

ELSE 'Venta Inválida'

END AS STATUS\_VENTA

FROM(

SELECT F.DNI, TC.NOMBRE, DATE\_FORMAT(F.FECHA\_VENTA, "%m - %Y") AS MES\_AÑO,

SUM(IFa.CANTIDAD) AS CANTIDAD\_VENDIDA,

MAX(VOLUMEN\_DE\_COMPRA)/10 AS CANTIDAD\_MAXIMA

FROM facturas F

INNER JOIN

items\_facturas IFa

ON F.NUMERO = IFa.NUMERO

INNER JOIN

tabla\_de\_clientes TC

ON TC.DNI = F.DNI

GROUP BY

F.DNI, TC.NOMBRE, DATE\_FORMAT(F.FECHA\_VENTA, "%m - %Y"))A;



3) Otro ejemplo de informe es el que determina la venta por sabores, para el año de 2016, presentando el porcentaje de participación de cada uno de estos sabores, ordenados.

SELECT VENTAS\_SABOR.SABOR, VENTAS\_SABOR.AÑO, VENTAS\_SABOR.CANTIDAD\_TOTAL,

ROUND((VENTAS\_SABOR.CANTIDAD\_TOTAL/VENTA\_TOTAL.CANTIDAD\_TOTAL)\*100,2)

AS PORCENTAJE FROM (

SELECT P.SABOR, SUM(IFa.CANTIDAD) AS CANTIDAD\_TOTAL,

YEAR(F.FECHA\_VENTA) AS AÑO FROM

tabla\_de\_productos P

INNER JOIN

items\_facturas IFa

ON P.CODIGO\_DEL\_PRODUCTO = IFa.CODIGO\_DEL\_PRODUCTO

INNER JOIN

facturas F

ON F.NUMERO = IFa.NUMERO

WHERE YEAR(F.FECHA\_VENTA) = 2016

GROUP BY P.SABOR, YEAR(F.FECHA\_VENTA)

ORDER BY SUM(IFa.CANTIDAD) DESC) VENTAS\_SABOR

INNER JOIN (

SELECT SUM(IFa.CANTIDAD) AS CANTIDAD\_TOTAL,

YEAR(F.FECHA\_VENTA) AS AÑO FROM

tabla\_de\_productos P

INNER JOIN

items\_facturas IFa

ON P.CODIGO\_DEL\_PRODUCTO = IFa.

INNER JOIN

facturas F

ON F.NUMERO = IFa.NUMERO

WHERE YEAR(F.FECHA\_VENTA) = 2016

GROUP BY YEAR(F.FECHA\_VENTA)) VENTA\_TOTAL

ON VENTA\_TOTAL.AÑO = VENTAS\_SABOR.AÑO

ORDER BY VENTAS\_SABOR.CANTIDAD\_TOTAL DESC;

