Contenido

[Leetcode 2](#_Toc201839476)

[Select 2](#_Toc201839477)

[Ejercicio 1 Productos Reciclables y bajos en grasa 2](#_Toc201839478)

[Ejercicio 2 encontrar los Clientes recomendados 2](#_Toc201839479)

[Ejercicio 3 países grandes 3](#_Toc201839480)

[Ejercicio 5 twets no validos 4](#_Toc201839481)

[Uniones básicas 5](#_Toc201839482)

[Ejemplo 6 reemplaza la identificación del empleado con el identificador único 5](#_Toc201839483)

[Ejercicio 7 análisis de productos de ventas 8](#_Toc201839484)

[Ejercicio 9 Aumento de temperatura 11](#_Toc201839485)

[Ejemplo 11 Bono para empleados 12](#_Toc201839486)

[Ejercicio 12 estudiantes y exámenes 14](#_Toc201839487)

[Basic agregation functions 18](#_Toc201839488)

[Sorting and gouping 18](#_Toc201839489)

[Advanced select and joins 18](#_Toc201839490)

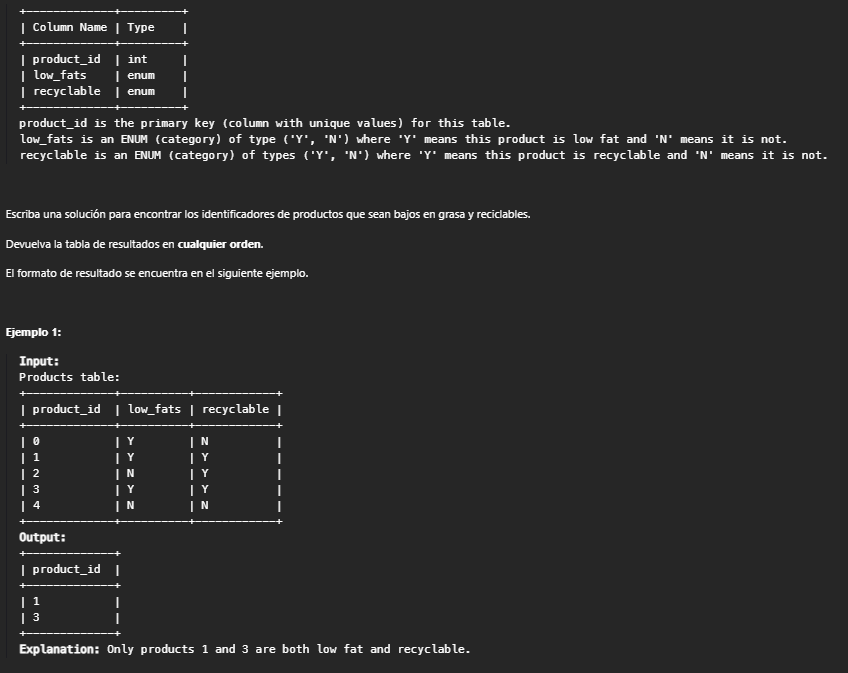
[Subqueries 19](#_Toc201839491)

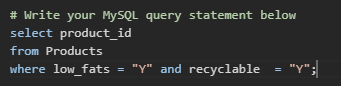
[Advanced String Functions / Regex / Clause 19](#_Toc201839492)

# Leetcode

## Select

### Ejercicio 1 Productos Reciclables y bajos en grasa



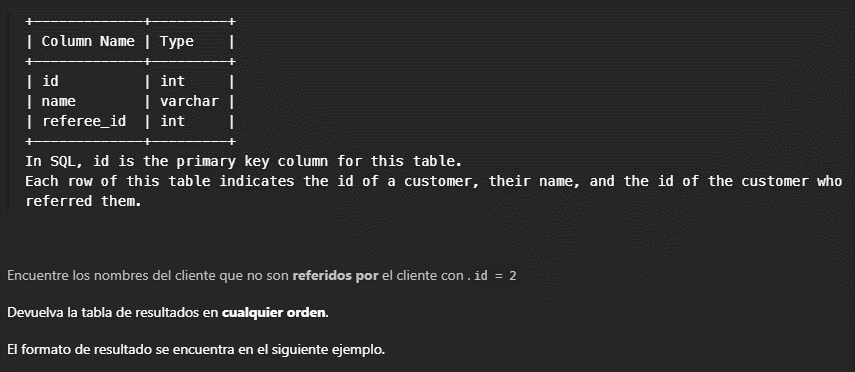


Esta consulta quiere decir selecciona las columnas producto id de la tabla products donde la columna low fats sea Y y la columna recyclable Y

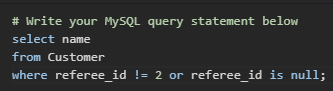
La versión de pandas:



### Ejercicio 2 encontrar los Clientes recomendados

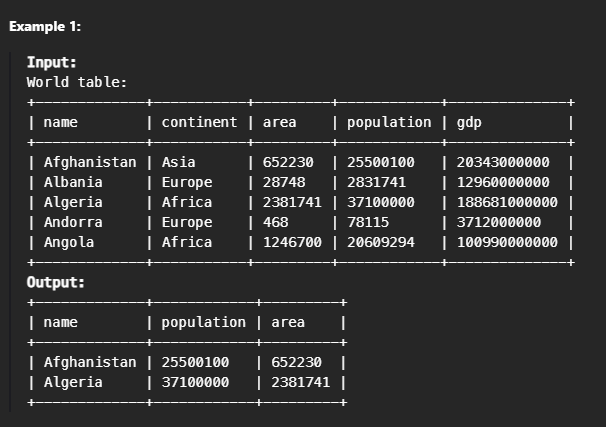


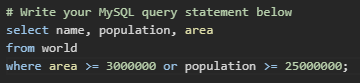




### Ejercicio 3 países grandes

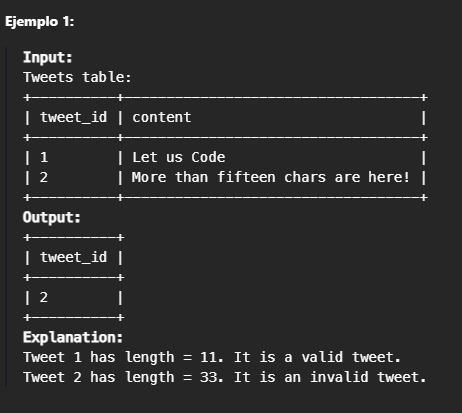






### Ejercicio 5 twets no validos





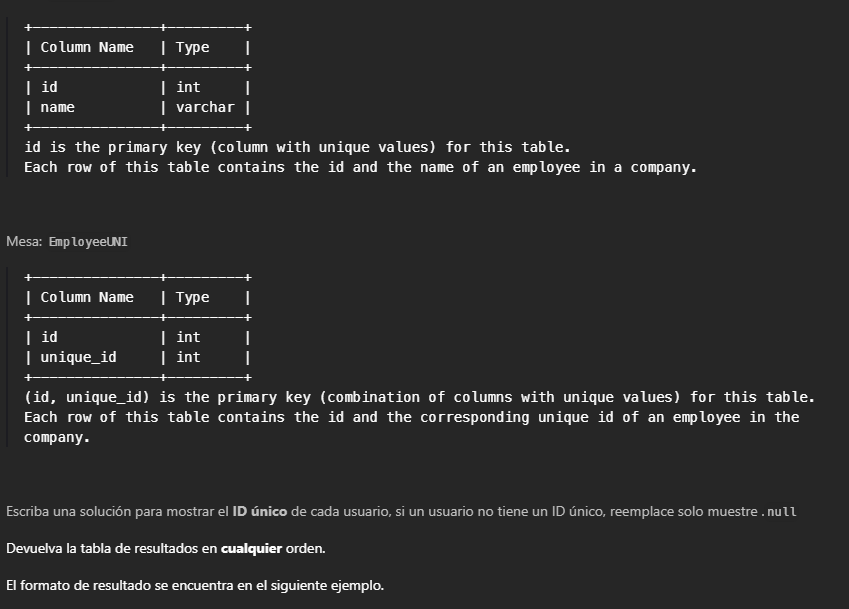
select tweet\_id

from tweets

where char\_length(content) > 15;

## Uniones básicas

### Ejemplo 6 reemplaza la identificación del empleado con el identificador único





SELECT eu.unique\_id, e.name

FROM Employees e

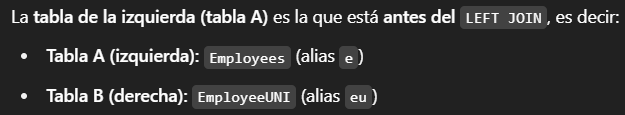
LEFT JOIN EmployeeUNI eu

ON e.id = eu.id;

Linea 1: la columna unique\_id de la tabla EmployeesUNI se renombra como eu, la columna name de la tabla employees se renombra como e

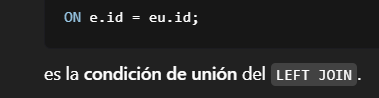
Linea 2: de la tabla employees y por sintaxis se debe poner el alias “e”

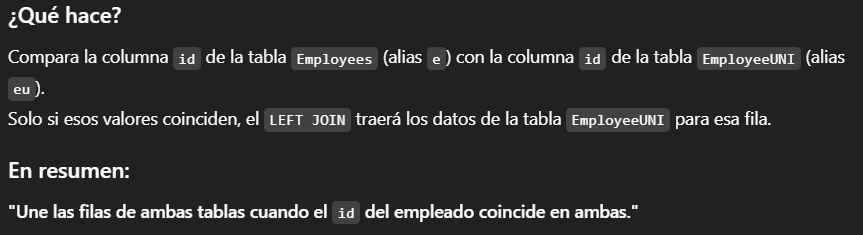
Linea 3: unión por la izquierda con la tabla employee “e” y la tabla employeeUNI “eu”. Es decir,



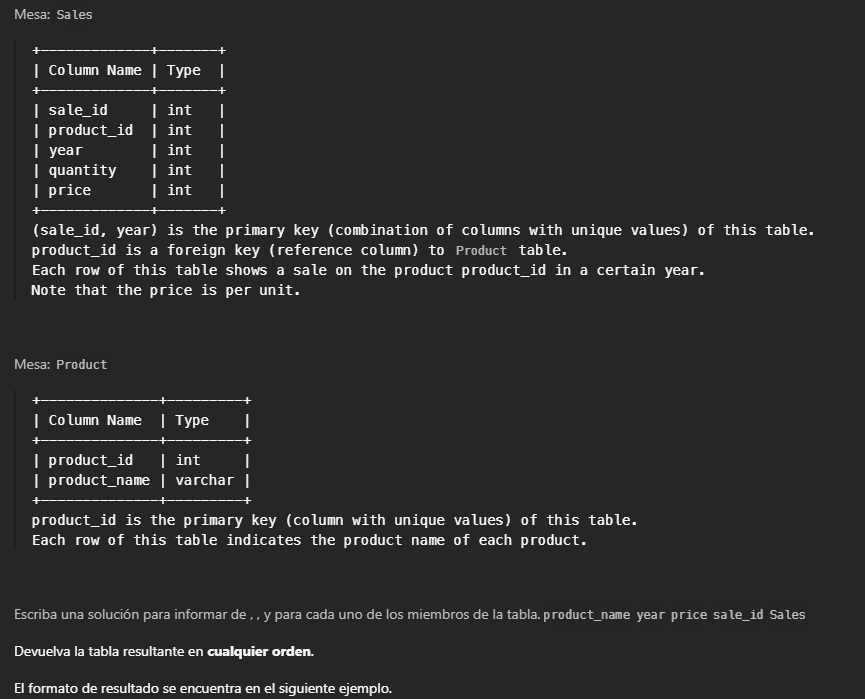
Es decir va a traer todos los registros de la tabla employees Y la intersección con la tabla employeesUNI

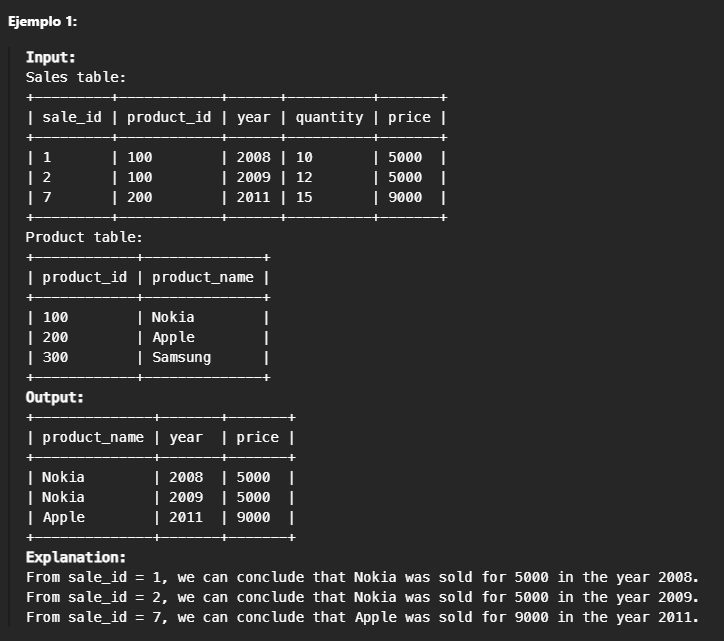
Linea 4:





### Ejercicio 7 análisis de productos de ventas





select product.product\_name, sales.year , sales.price

from sales

left join product

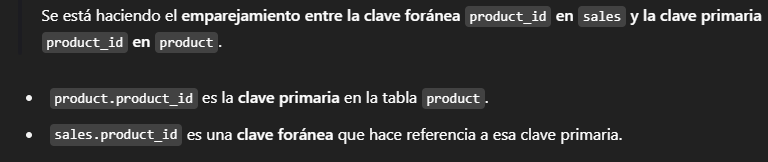
on sales.product\_id = product.product\_id;

linea 1 seleccionar la columna producto\_name de la tabla producto y seleccionar la columna year y Price de la tabla sales

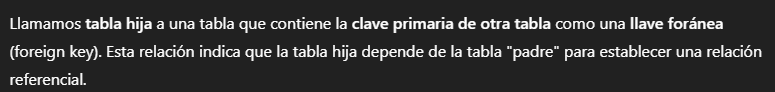
linea 2 traer información de la tabla sales

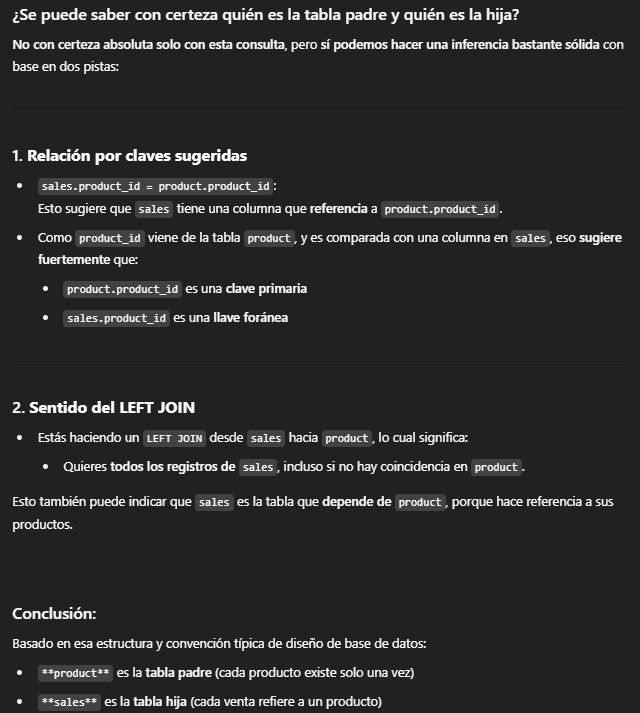
linea 3 Traer toda la información de la tabla sales, y en caso de que haya coincidencias en product, traer también esa información. Si no hay coincidencia, igualmente se incluye la fila de sales con valores NULL en las columnas de product.

linea 4

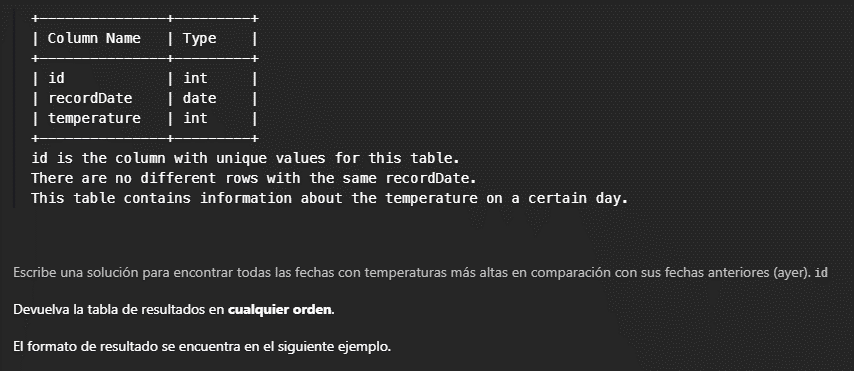


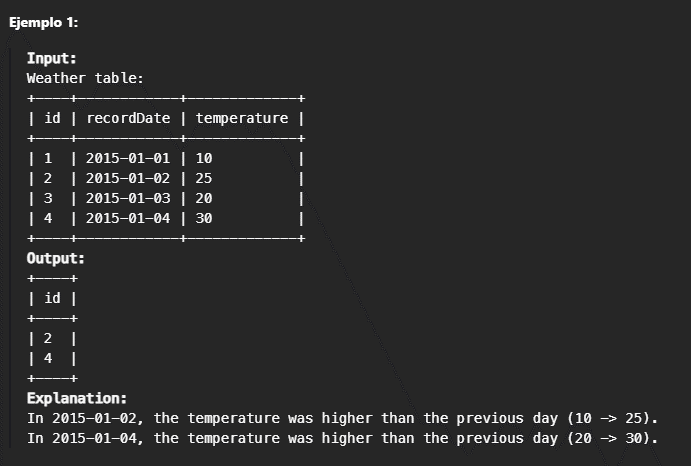
Notas:





### Ejercicio 9 Aumento de temperatura





SELECT w1.id

FROM Weather w1

JOIN Weather w2

  ON DATEDIFF(w1.recordDate, w2.recordDate) = 1

WHERE w1.temperature > w2.temperature;

Ahora estudiemos las líneas de Código

1 seleccionar la columna id de la tabla weather renombrada como w1

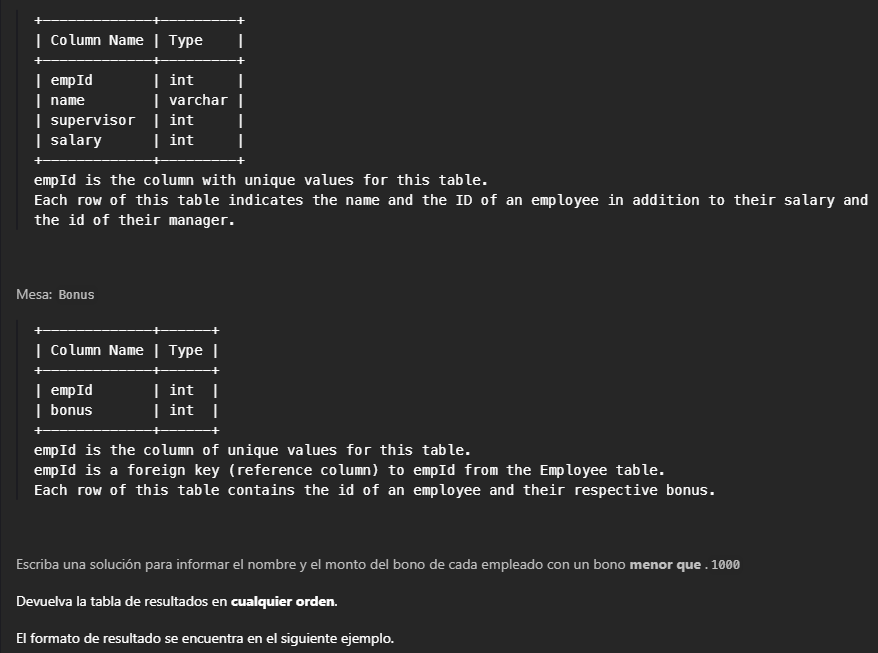
2de la tabla weather

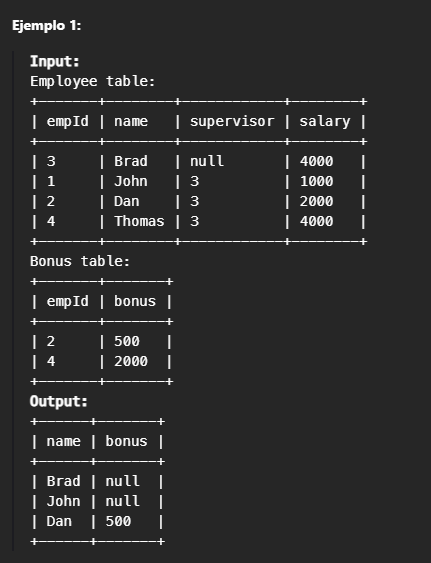
3 hacer un inner join (Recordemos que cuando sólo ponemos el join es como poner el inner Join, es la unión por defecto). Al hacer el inner join vamos a traer La tabla completa ya que la intersección de ambas tablas w1 y w2 Son todas sus filas

4 DATEDIFF(w1.recordDate, w2.recordDate) = 1 asegura que se compara solo con la fecha inmediatamente anterior. El orden del argumento de la función que estamos utilizando sí importa

5 Imponemos la condición de que los valores de la columna temperatura de la tabla 1 sean mayores que los valores de la columna temperatura de la tabla 2

### Ejemplo 11 Bono para empleados





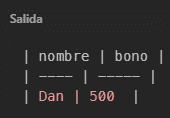
select e.name, b.bonus

from employee e

left join bonus b

    on e.empId = b.empId

where bonus < 1000;



Esta consulta está parcialmente bien pero hay que ver la forma de traer los valores nulos de aquellos que no tienen asignado un bono

Para corregirlo simplemente hay que agregar un operador lógico “or”

select e.name, b.bonus

from employee e

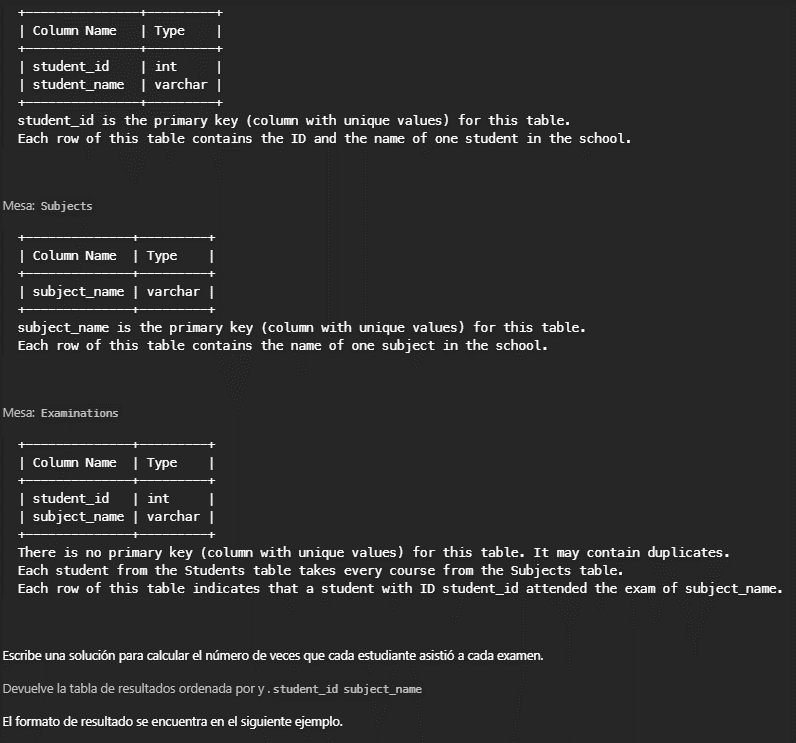
left join bonus b

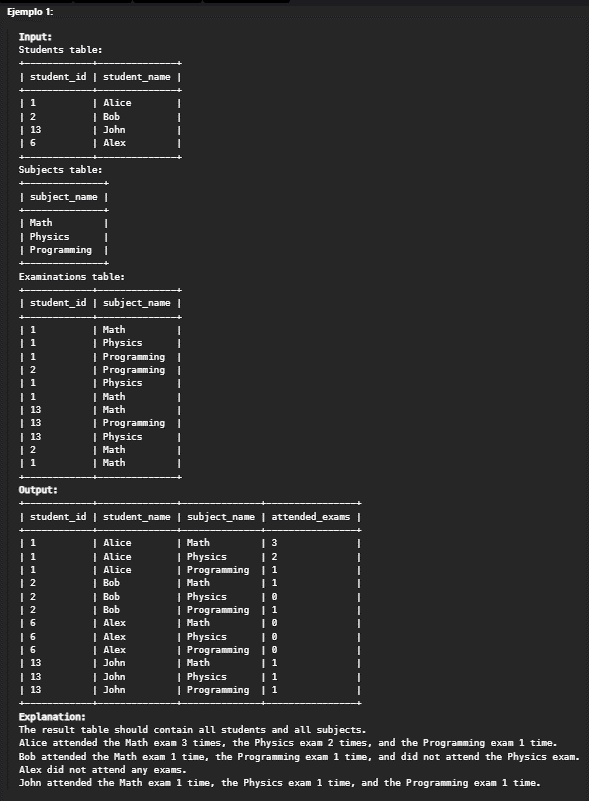
    on e.empId = b.empId

where bonus < 1000 OR b.bonus IS NULL;

Y esta consulta ya es correcta

### Ejercicio 12 estudiantes y exámenes





SELECT

S.student\_id

,S.student\_name

,SU.subject\_name

,COUNT(E.student\_id) attended\_exams

FROM Students S

CROSS JOIN Subjects SU

LEFT JOIN Examinations E

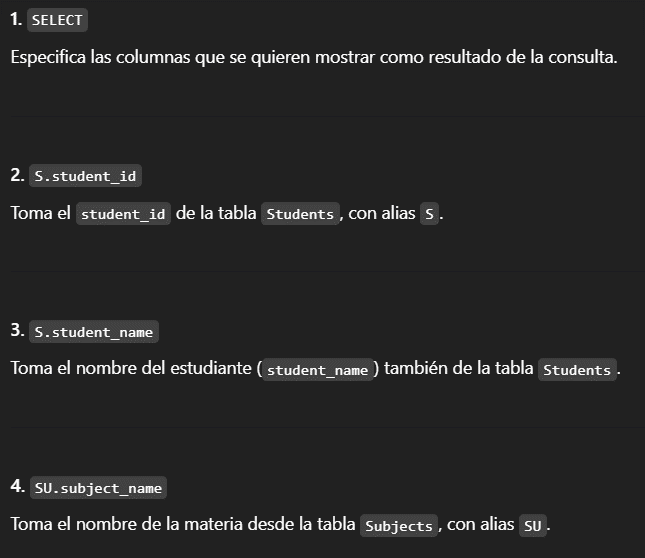
ON S.student\_id = E.student\_id

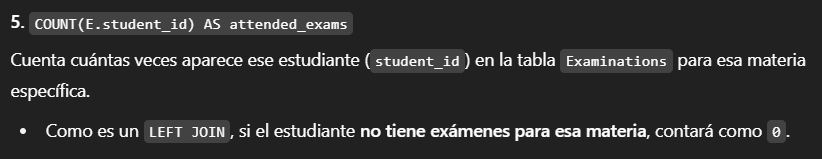
AND SU.subject\_name = E.subject\_name

GROUP BY S.student\_id, S.student\_name, SU.subject\_name

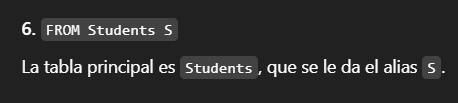
ORDER BY S.student\_id, S.student\_name, SU.subject\_name

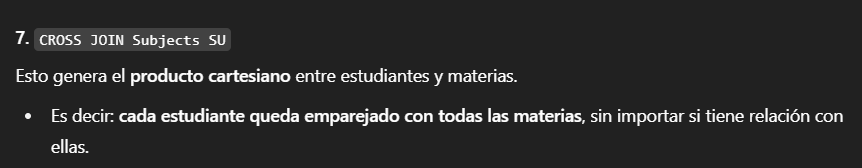
;



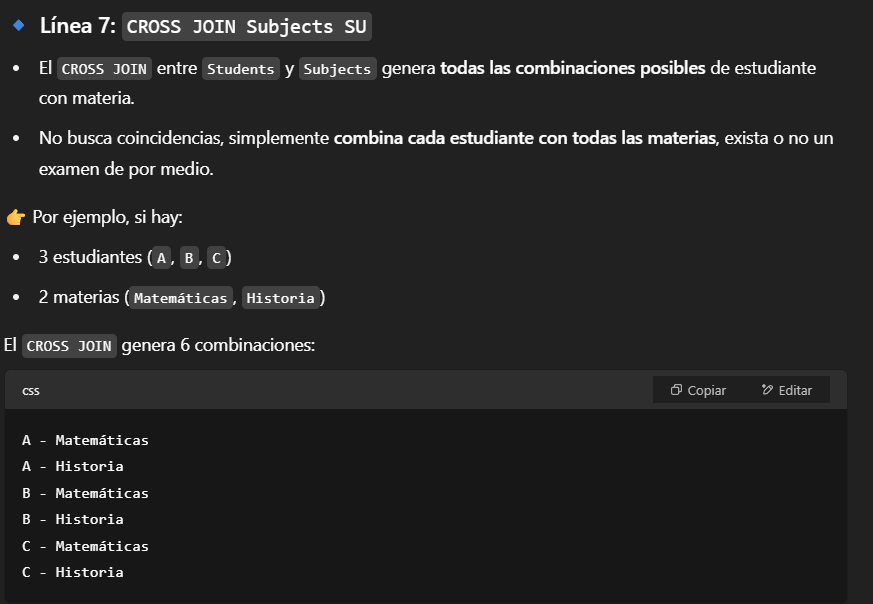


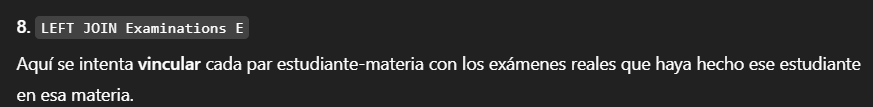
Aquí se creará una columna que contabiliza el número de repeticiones de cada categoría de la columna student\_id de la tabla examination



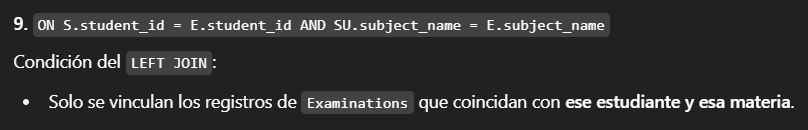


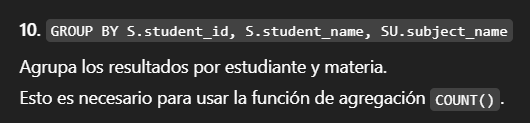
Producto cartesiano significa lo siguiente

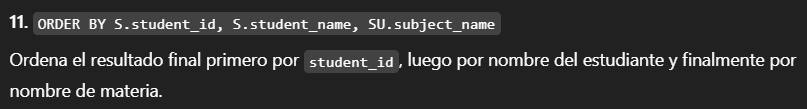




El left join se hace a partir del resultado del cross join. Es decir que la tabla A Sería el resultado del Cross join y la tabla B seria la tabla E examination







# Basic agregation functions

# Sorting and gouping

# Advanced select and joins

# Subqueries

# Advanced String Functions / Regex / Clause