

FUNDACIÓN UNIVERSITARIA COLOMBO INTERNACIONAL UNICOLOMBO

FACULTAD DE INGENIERÍA

ESTUDIANTE:

Melany Olivera Atencio

DOCENTE:

Jhon Carlos Arrieta Arrieta

ASIGNATURA:

Bases de Datos

NOMBRE:

Diseño e Implementación de Bases de Datos: Aplicación en PostreSQL

CARTAGENA

2025-1

*Tabla de Contenido*

1. Presentación
2. Introducción
3. Objetivos
4. Justificación
5. Desarrollo
6. Implementación del Sistema de Gestión de Base de Datos
7. Bibliografía
8. Herramientas utilizadas
9. Sitios web y documentación
10. Presentación

Este documento presenta el desarrollo de un sistema de gestión de evaluaciones académicas para la asignatura de Computación II en la Universidad de Chile. El trabajo incluye el análisis, modelado y diseño de la base de datos que permitirá almacenar y gestionar los controles escritos, prácticas y exámenes de los estudiantes, así como la asignación de profesores a las distintas evaluaciones.

1. Introducción

El presente trabajo tiene como propósito diseñar un sistema de gestión de evaluaciones académicas que permita almacenar y consultar los resultados de los controles escritos y prácticas realizados por los estudiantes de Computación II. Se ha considerado la necesidad de registrar las notas de cada estudiante en diferentes tipos de evaluaciones, incluyendo controles escritos y prácticas individuales o grupales, con el fin de mejorar la gestión académica.

Para ello, se ha utilizado un modelo de base de datos relacional, asegurando que todas las consultas requeridas puedan ser ejecutadas de manera eficiente. Se incluyen los aspectos clave del diseño, como entidades, relaciones y restricciones, con el objetivo de garantizar la integridad y coherencia de la información.

1. Objetivos

* Diseñar un modelo de base de datos que permita gestionar los resultados de evaluaciones académicas.
* Implementar relaciones entre alumnos, profesores, asignaturas y evaluaciones.
* Garantizar la correcta organización de los datos para facilitar consultas eficientes sobre los resultados de los estudiantes.
* Permitir la gestión de prácticas individuales y grupales.
* Asegurar que los profesores puedan registrar evaluaciones y colaborar en el diseño de prácticas.

1. Justificación

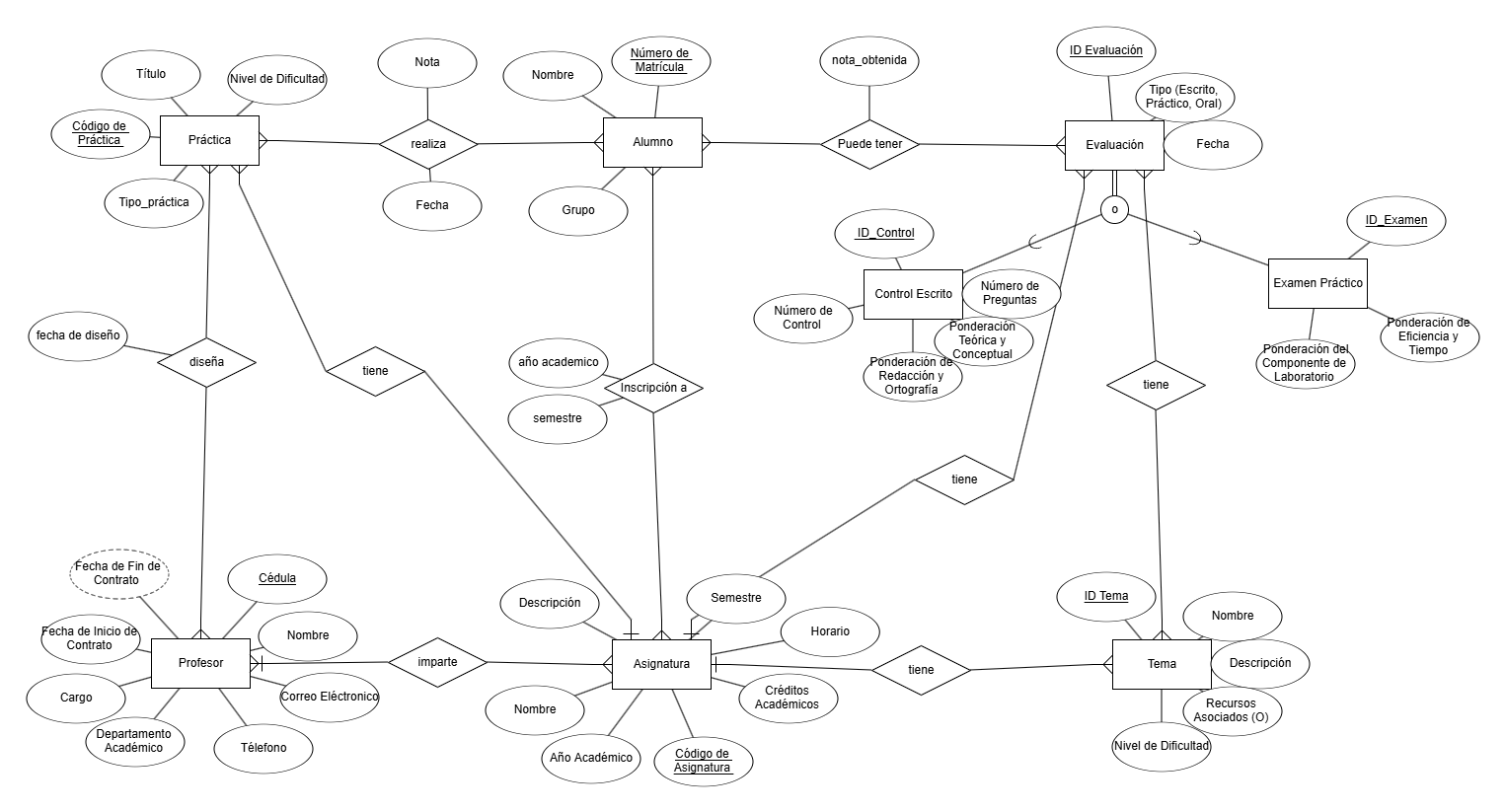
El desarrollo de este sistema es fundamental para mejorar la gestión académica de la asignatura de Computación II, ya que permite automatizar el registro y consulta de evaluaciones. Actualmente, la información de los exámenes y prácticas puede ser difícil de manejar sin una base de datos bien estructurada.

Además, este sistema permitirá a los docentes realizar un seguimiento más preciso del rendimiento de los estudiantes, identificar patrones de aprendizaje y optimizar los procesos de evaluación. Asimismo, facilitará la asignación de profesores a distintas prácticas y exámenes, promoviendo la colaboración en el diseño de actividades evaluativas.

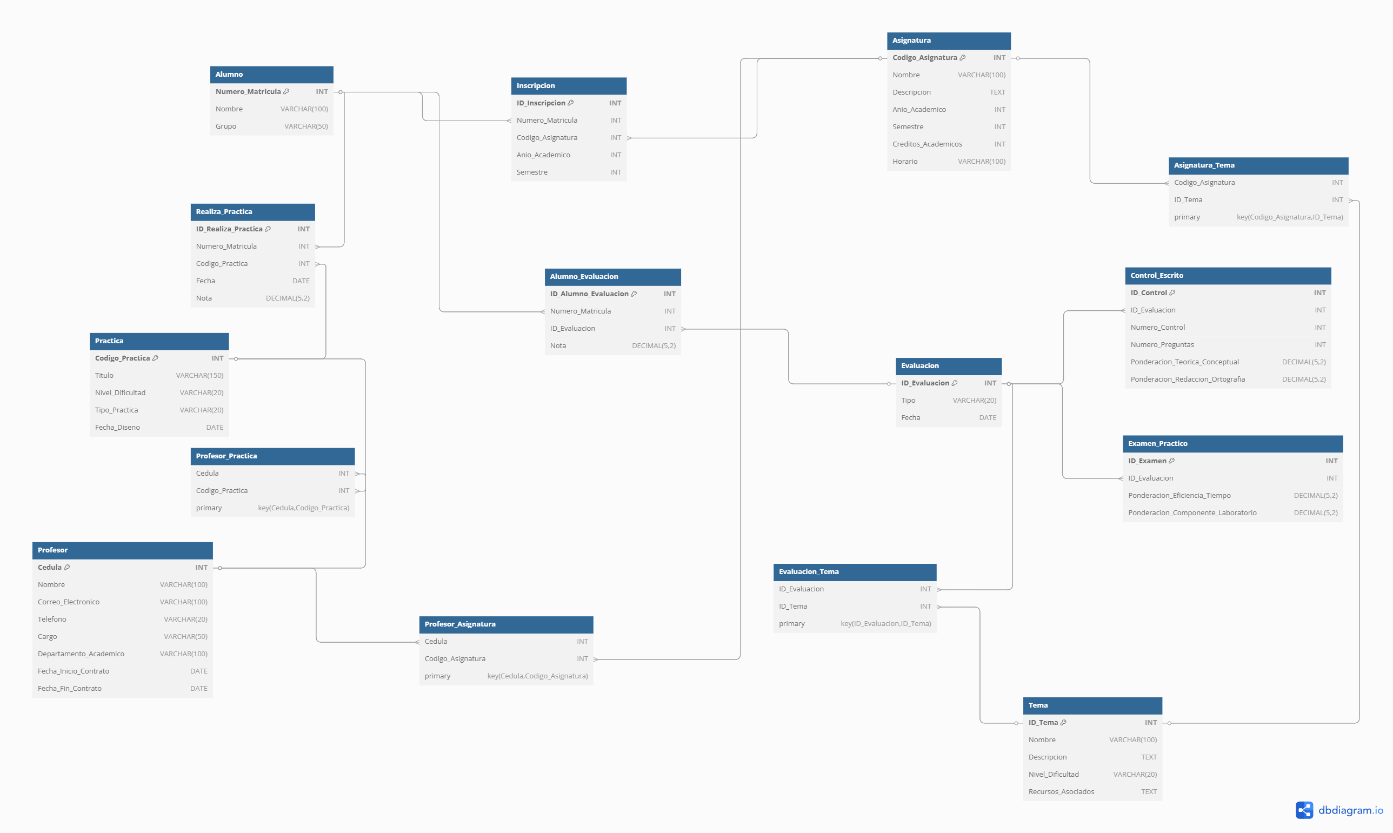
El diseño de la base de datos también asegura la posibilidad de escalar el sistema en el futuro, integrando nuevas funcionalidades como reportes estadísticos y análisis de desempeño estudiantil.

5. Desarrollo

**Modelo Entidad Relación**



**Modelo Relacional**



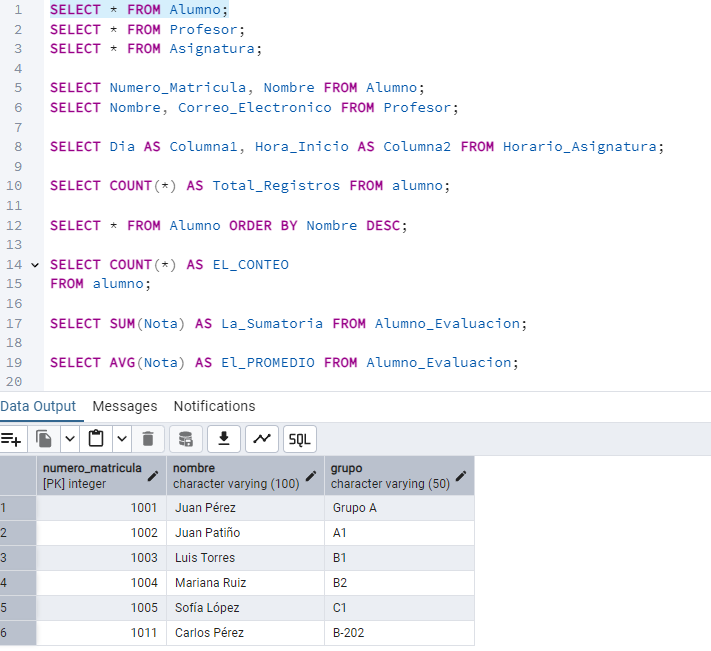
1. **Motor MYSQL**

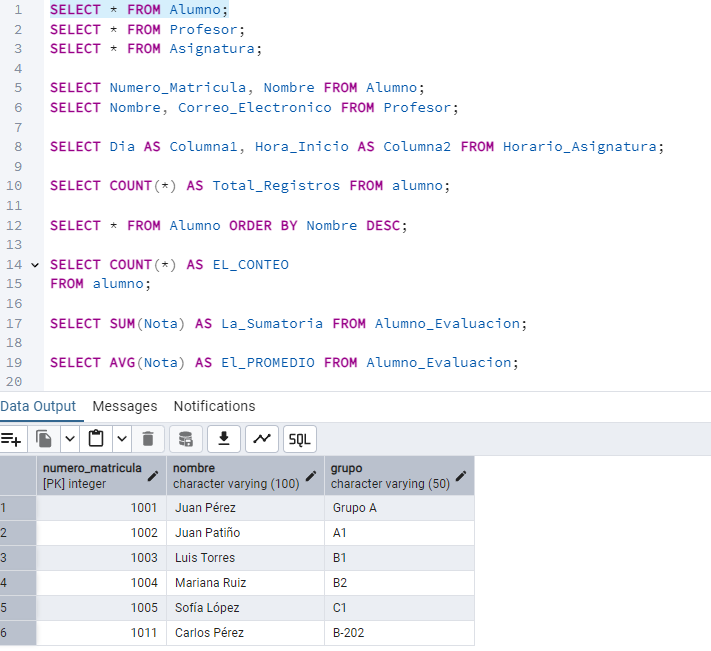
Tomar alguna de las tablas (TablaX, TablaY, TablaZ) de la base de datos bd\_grupo\_xyz y realizar las siguientes operaciones.

1. Mostrar **todos los registros** de la TablaX:

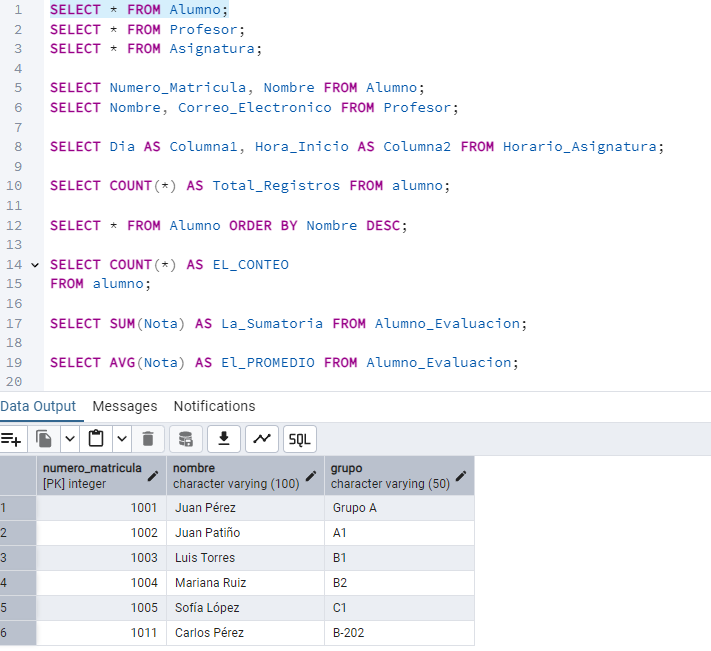
**INSTRUCCIÓN SQL**:

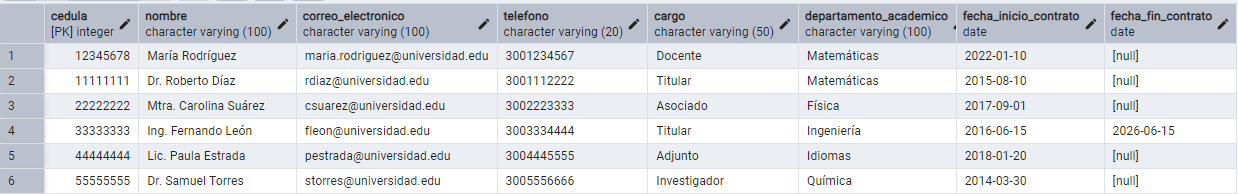
SELECT \* FROM Alumno;





SELECT \* FROM Profesor;



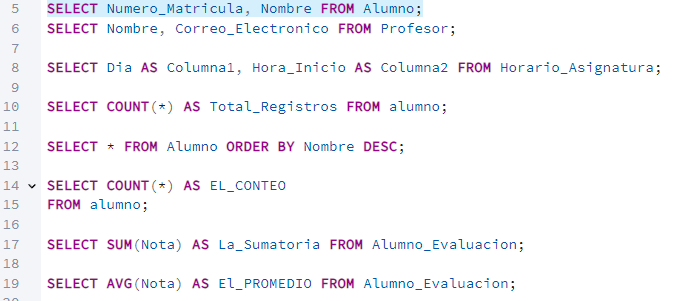


1. **Mostrar el valor solo de 2 columnas** de todos los registros de la TablaX

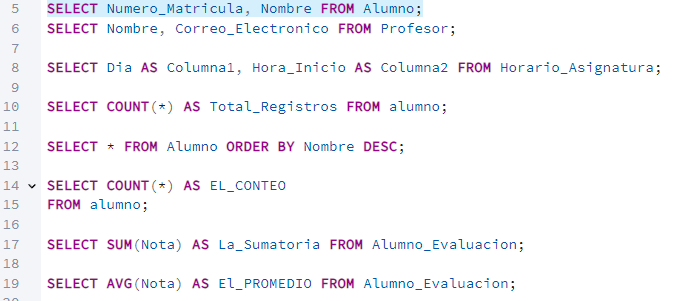
**INSTRUCCIÓN SQL**:

SELECT Numero\_Matricula, Nombre FROM Alumno;

SELECT Nombre, Correo\_Electronico FROM Profesor;





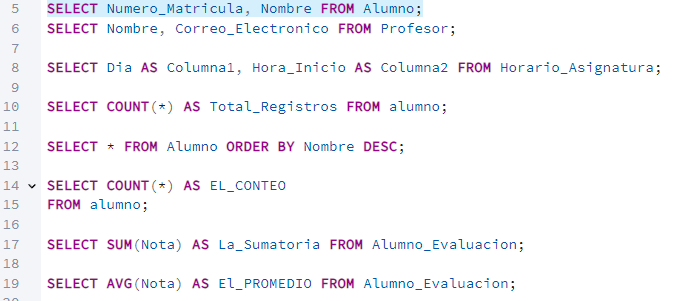


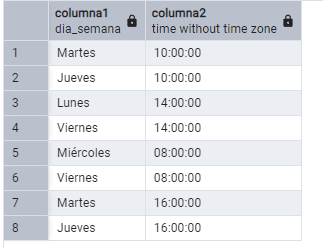


1. Mostrar el valor de 2 columnas de todos los registros de la TablaX, pero en el resultado de la consulta el **nombre de esas dos columnas debe ser** Columna1, Column2 respectivamente:

**INSTRUCCIÓN SQL**:

SELECT Dia AS Columna1, Hora\_Inicio AS Columna2 FROM Horario\_Asignatura;

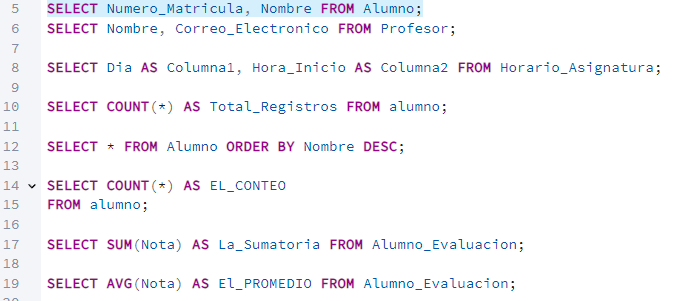




1. Mostrar el **número de registros** almacenados en la TablaX:

**INSTRUCCIÓN SQL**:

SELECT COUNT(\*) AS Total\_Registros FROM Horario\_Asignatura;

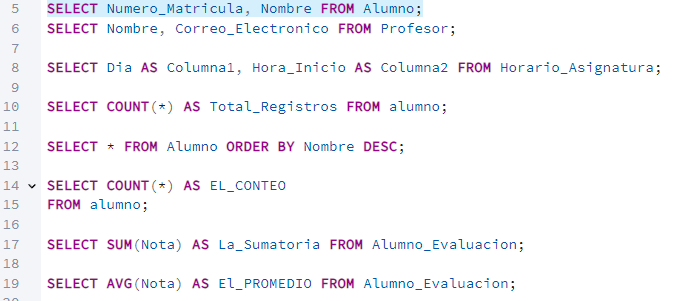




1. Mostrar todos los registros de la TablaX **ordenados descendentemente**:

**INSTRUCCIÓN SQL**:

SELECT \* FROM Alumno ORDER BY Nombre DESC;

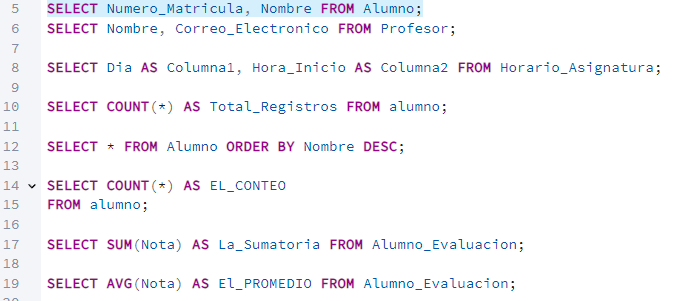




1. Mostrar el **número de registros** almacenados en la TablaX, pero el valor del resultado debe estar almacenado en una columna llamada EL\_CONTEO.

**INSTRUCCIÓN SQL**:

SELECT COUNT(\*) AS EL\_CONTEO FROM alumno;

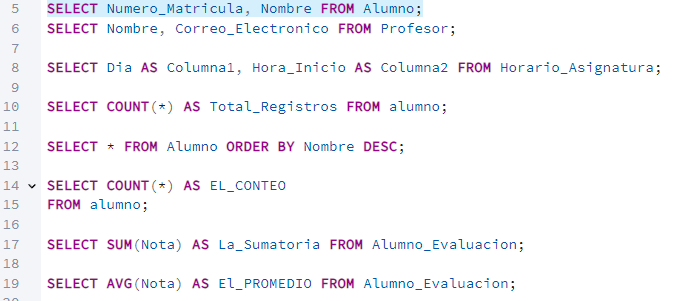


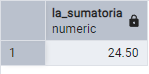


1. Realizar una consulta en la TabalX o la Tabla Y o la TablaZ en la que se pueda obtener una **sumatoria** de valores de alguna columna numérica, pero el valor del resultado debe estar almacenado en una columna llamada LA\_SUMATORIA.

**INSTRUCCIÓN SQL**:

SELECT SUM(Nota) AS La\_Sumatoria FROM Alumno\_Evaluacion;



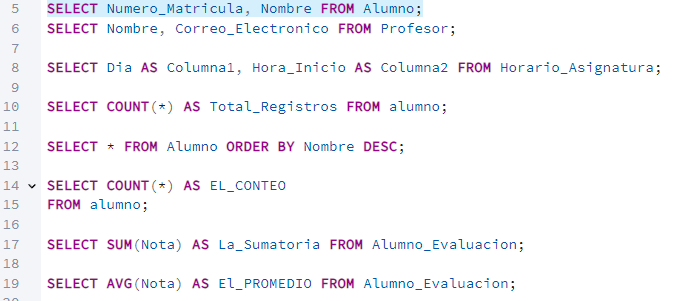


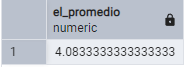
SUM(Nota) AS La\_Sumatoria: Suma todas las notas registradas en la tabla Alumno\_Evaluacion.

1. Realizar una consulta en la TablaX o la TablaY o la TablaZ en la que se pueda obtener el **promedio** de valores de alguna columna numérica, pero el valor del resultado debe estar almacenado en una columna llamada El\_PROMEDIO.

**INSTRUCCIÓN SQL**:

SELECT AVG(Nota) AS El\_PROMEDIO FROM Alumno\_Evaluacion;

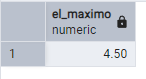
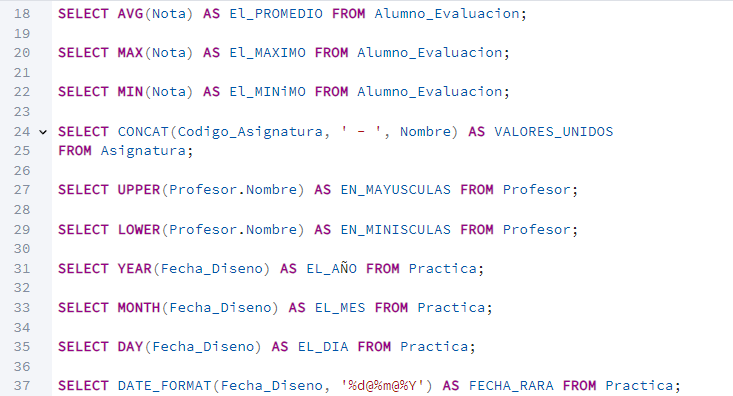




1. Realizar una consulta en la TablaX o la TablaY o la TablaZ en la que se pueda obtener el **el valor máximo** de alguna columna numérica, pero el valor del resultado debe estar almacenado en una columna llamada El\_MAXIMO.

**INSTRUCCIÓN SQL**:

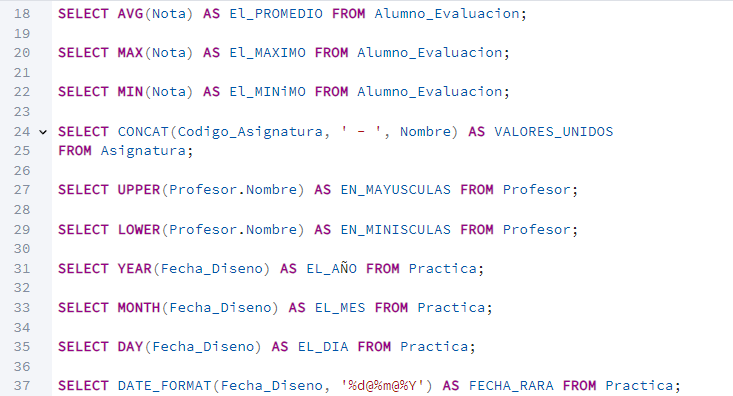
SELECT MAX(Nota) AS El\_MAXIMO FROM Alumno\_Evaluacion;



1. Realizar una consulta en la TablaX o la TablaY o la TablaZ en la que se pueda obtener el **el valor mínimo** de alguna columna numérica, pero el valor del resultado debe estar almacenado en una columna llamada El\_MINIMO.

**INSTRUCCIÓN SQL**:

SELECT MIN(Nota) AS El\_MINiMO FROM Alumno\_Evaluacion;

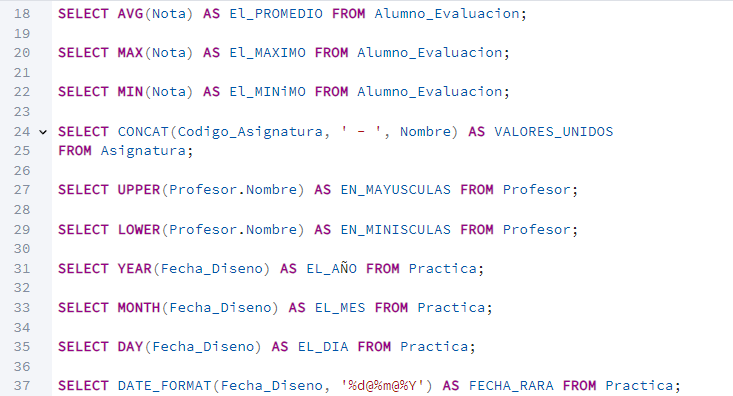


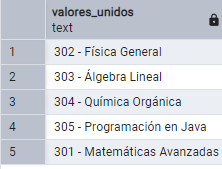
1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar los valores de dos columnas **concatenados** en una sola columna, pero el valor del resultado debe estar almacenado en una columna llamada VALORES\_UNIDOS.

**INSTRUCCIÓN SQL**:

SELECT CONCAT(Codigo\_Asignatura, ' - ', Nombre) AS VALORES\_UNIDOS

FROM Asignatura;

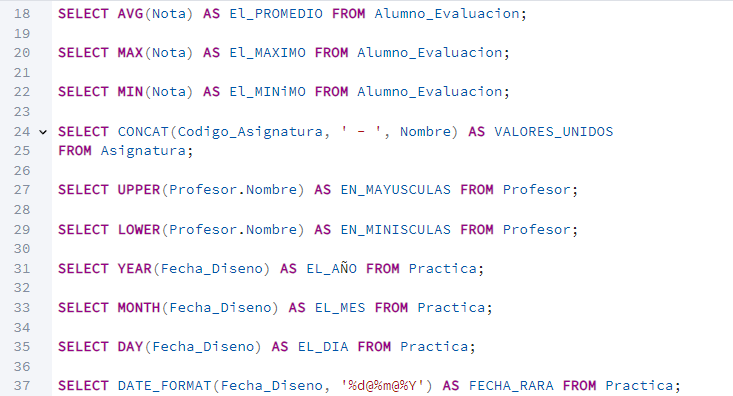




1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar el **MAYÚSCULAS** los valores de una columna, y en **Minúsculas** los valores de otra columna, pero en el resultado estas columnas deben llamarse EN\_MAYUSCULA, EN\_MINUSCULA respectivamente.

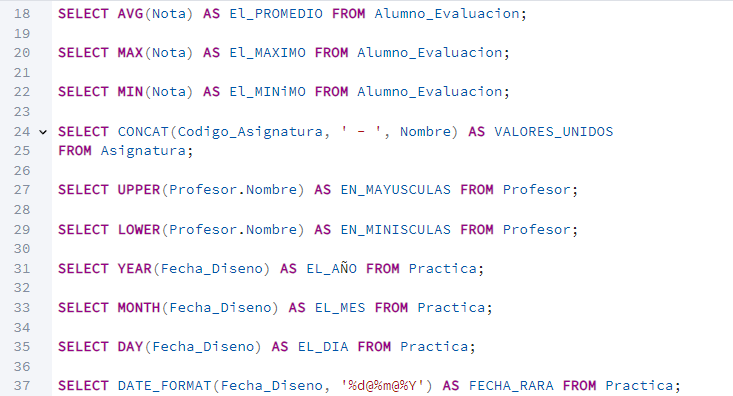
**INSTRUCCIÓN SQL**:

SELECT UPPER(Profesor.Nombre) AS EN\_MAYUSCULAS FROM Profesor;





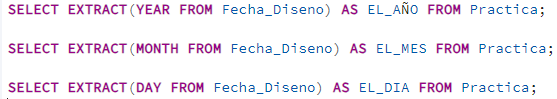
SELECT LOWER(Profesor.Nombre) AS EN\_MINISCULAS FROM Profesor;

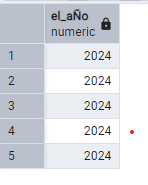




1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar sólo **el AÑO** de alguna columna Tipo Fecha, pero en el resultado esta columna debe llamarse EL\_AÑO.

**INSTRUCCIÓN SQL**:

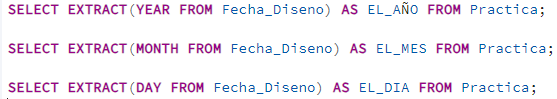
SELECT EXTRACT(YEAR FROM Fecha\_Diseno) AS EL\_AÑO FROM Practica;

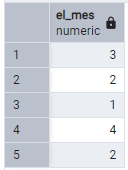


1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar sólo **el MES** de alguna columna Tipo Fecha, pero en el resultado esta columna debe llamarse EL\_MES:

**INSTRUCCIÓN SQL**:

SELECT EXTRACT(MONTH FROM Fecha\_Diseno) AS EL\_MES FROM Practica;

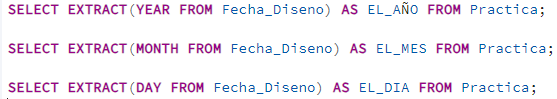


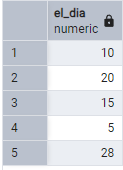


1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar sólo **el DIA** de alguna columna Tipo Fecha, pero en el resultado esta columna debe llamarse EL\_DIA..

**INSTRUCCIÓN SQL**:

SELECT EXTRACT(DAY FROM Fecha\_Diseno) AS EL\_DIA FROM Practica;

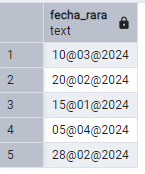




1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar sólo **la fecha en formato personalizado,** así **(dia@mes@año)** de alguna columna Tipo Fecha, pero en el resultado esta columna debe llamarse FECHA\_RARA.:

**INSTRUCCIÓN SQL**:

SELECT TO\_CHAR(Fecha\_Diseno, 'DD@MM@YYYY') AS FECHA\_RARA FROM Practica;



1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar sólo **la FECHA ACTUAL** en una nueva columna**,** pero en el resultado esta columna debe llamarse FECHA\_ACTUAL.

**INSTRUCCIÓN SQL**:

SELECT NOW() AS FECHA\_Y\_HORA\_ACTUAL;





1. Realizar una consulta en la TablaX o la TablaY o la TablaZ que permita mostrar en 3 nuevas columnas el resultado de **SUMAR**, **RESTAR** Y **MULTIPLICAR** los valores de 2 columnas Tipo numérico, pero en el resultado esta columna debe llamarse LA\_SUMA, LA\_RESTA, LA\_MULTIPLICACION respectivamente.

**INSTRUCCIÓN SQL**:

SELECT

Creditos\_Academicos,

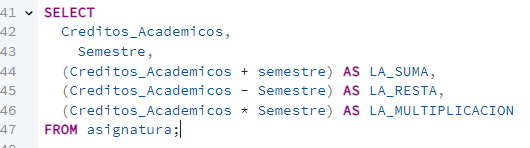
Semestre,

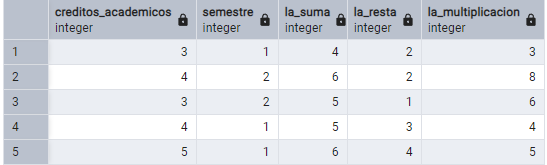
(Creditos\_Academicos + semestre) AS LA\_SUMA,

(Creditos\_Academicos - Semestre) AS LA\_RESTA,

(Creditos\_Academicos \* Semestre) AS LA\_MULTIPLICACION

FROM asignatura;0





1. Investigar cual es el objetivo de la Palabra DISTINCT de SQL

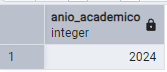
La palabra clave DISTINCT en SQL se utiliza para eliminar valores duplicados en los resultados de una consulta, mostrando únicamente registros únicos. Es útil cuando se desea conocer los diferentes valores que existen en una columna específica sin repeticiones.

1. Seleccionar alguna de las tablas TablaX o la TablaY o la TablaZ y realizar una consulta donde se pueda apreciar el uso de la palabra **DISTINCT**

**INSTRUCCIÓN SQL**:

SELECT DISTINCT Anio\_Academico FROM asignatura;



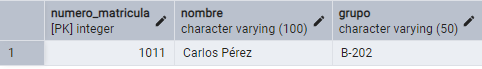


1. Realizar una consulta en la TablaX que permita mostrar los datos de un registro cuya columna llave primaria sea **igual** al valorY

**INSTRUCCIÓN SQL**:

SELECT \* FROM alumno WHERE Numero\_Matricula = 1011;

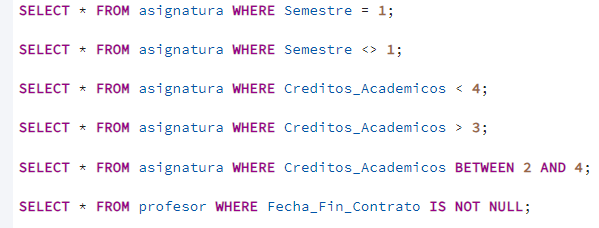


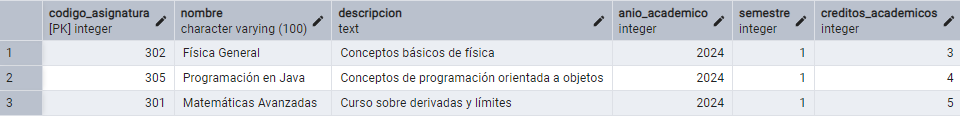


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la condición columnaY **igual** datoZ

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Semestre = 1;



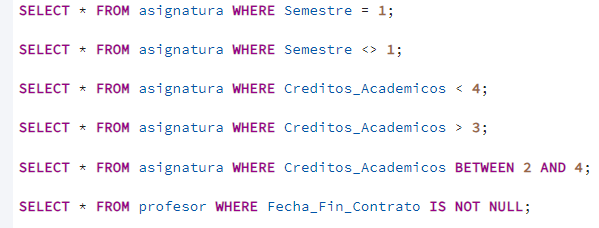


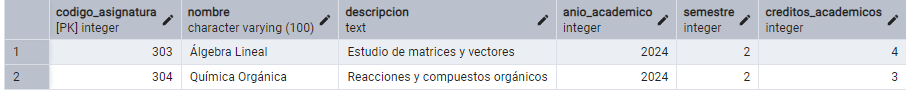
La consulta significa buscar en la tabla asignatura, todas las materias que se dan en el semestre 1.

1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **diferente**  datoZ

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Semestre <> 1;

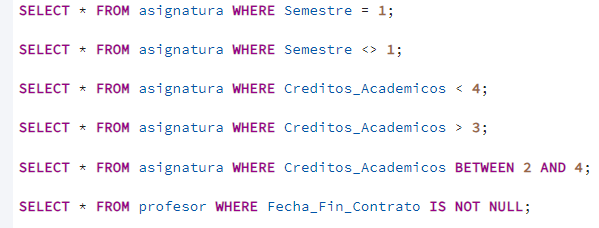


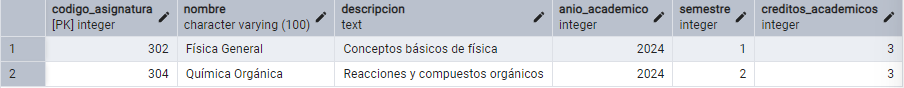


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **menor**  datoZ

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Creditos\_Academicos < 4;



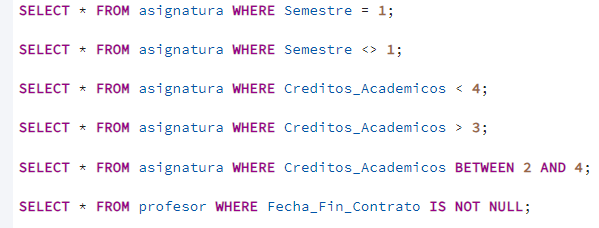


Mostrar todas las asignaturas con **menos de 4 créditos académicos**

1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **mayor** datoZ

**INSTRUCCIÓN SQL**:

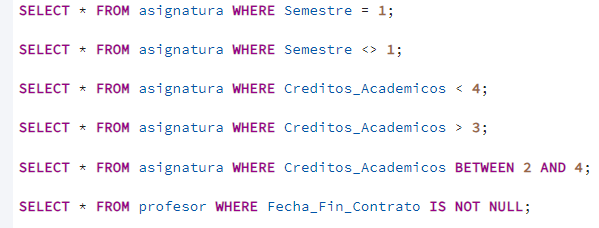
SELECT \* FROM asignatura WHERE Creditos\_Academicos > 3;

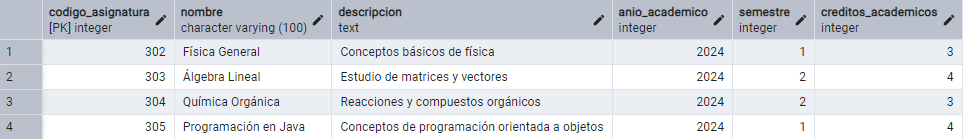


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **entre valor** datoX y valorZ

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Creditos\_Academicos BETWEEN 2 AND 4;

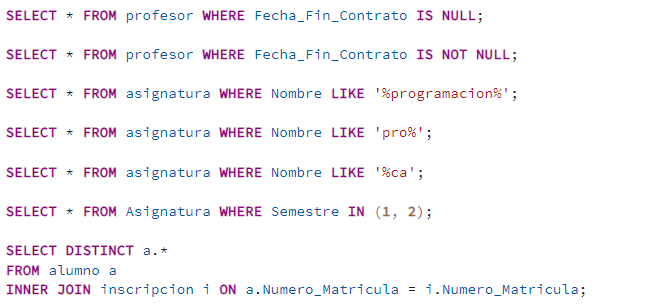


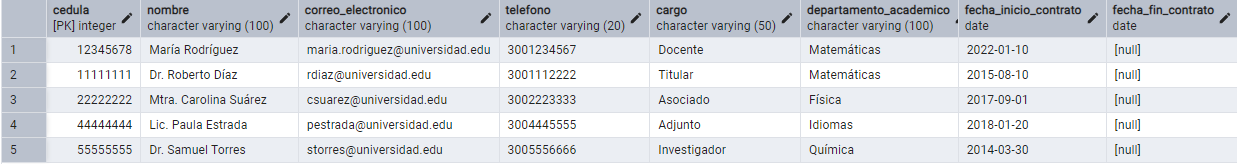


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **Sea NULO**

**INSTRUCCIÓN SQL**:

SELECT \* FROM profesor WHERE Fecha\_Fin\_Contrato IS NULL;

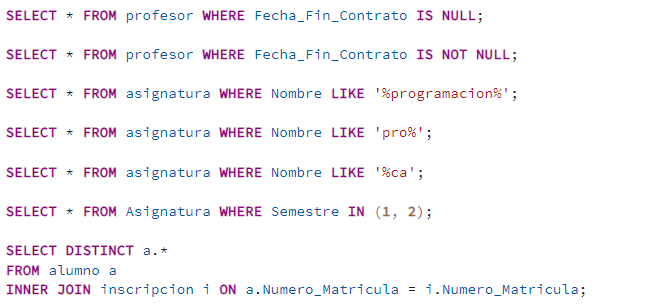


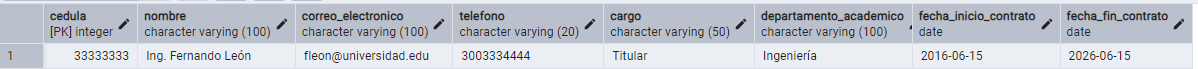


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **No Sea NULO**

**INSTRUCCIÓN SQL**:

SELECT \* FROM profesor WHERE Fecha\_Fin\_Contrato IS NOT NULL;



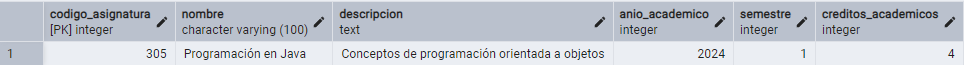


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **Contenga el valor** DatoX en su interior

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Nombre ILIKE '%programac%';

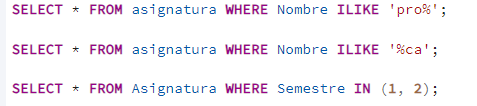


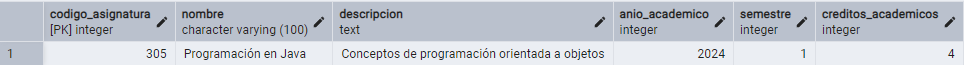


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **Inicie con el valor** DatoX

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Nombre LIKE 'pro%';

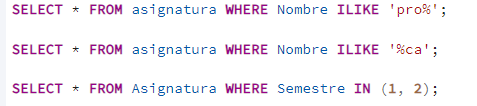


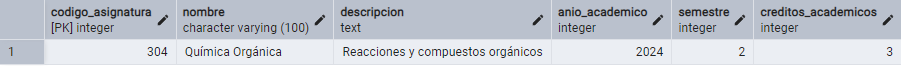


1. Realizar una consulta en la TablaX que permita mostrar los registros con que cumplan con la conducción columnaY **Termine con el valor** DatoX

**INSTRUCCIÓN SQL**:

SELECT \* FROM asignatura WHERE Nombre LIKE '%ca';





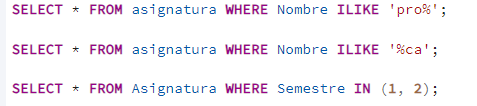
1. Investigar cual es el objetivo del comando IN.

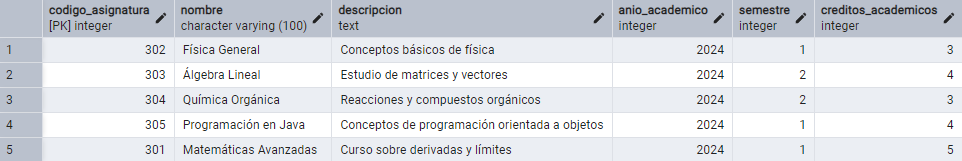
El comando IN en SQL se utiliza para filtrar registros que coincidan con uno o varios valores específicos dentro de una columna. Es útil para evitar escribir múltiples condiciones con OR.

1. Realizar una consulta en la TablaX que permita mostrar el uso y aplicación de la palabra **IN**

**INSTRUCCIÓN SQL**:

SELECT \* FROM Asignatura WHERE Semestre IN (1, 2);





Esta instrucción selecciona **todas las asignaturas que están en el semestre 1 o 2.**

Para realizar las siguientes consultas se asume que TablaA, TablaB, TablaC, …, TablaZ, representan relacionadas entre sí, mediante llaves foráneas, que se describen en El Modelo Relacional del ejercicio que han seleccionado.

Asumimos que TablaA y TablaB están relacionados así:

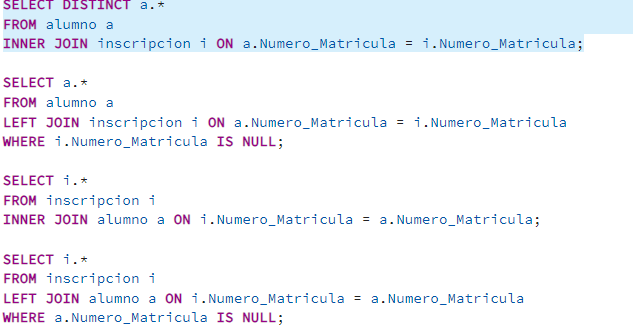
[**TablaA**] ->----<**R**>----|- [**TablaB**] ->----<**R**>----|- [**TablaC**]

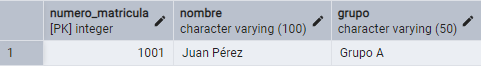
Es decir, [**TablaA**] tiene una llave foránea llamada por ejemplo **tablaB\_id**, la cual se conecta con la llave primaria **id** de la [**TablaB**], mientras que [**TablaB**] tiene una llave foránea llamada por ejemplo **tablaC\_id** la cual se conecta con la llave primaria **id** de la [**TablaC**],

1. Realizar una consulta que permitan mostrar los registros de la **TablaA que están relacionados** con la **TablaB**

**INSTRUCCIÓN SQL**:

SELECT DISTINCT a.\* FROM alumno a INNER JOIN inscripcion i ON a.Numero\_Matricula = i.Numero\_Matricula;



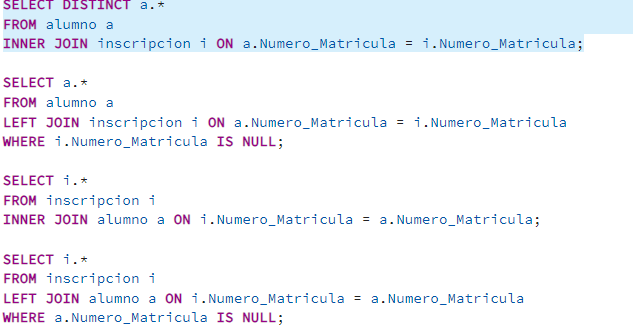


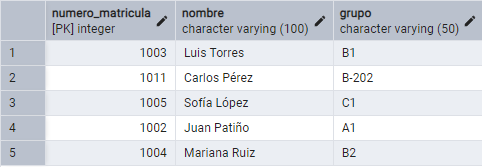
Muestra los **alumnos que tienen al menos una inscripción** registrada en la tabla inscripción.

1. Realizar una consulta que permitan mostrar los registros de la **TablaA** que **NO están relacionados** con la **TablaB**

**INSTRUCCIÓN SQL**:

SELECT a.\* FROM alumno a LEFT JOIN inscripcion i ON a.Numero\_Matricula = i.Numero\_Matricula WHERE i.Numero\_Matricula IS NULL;





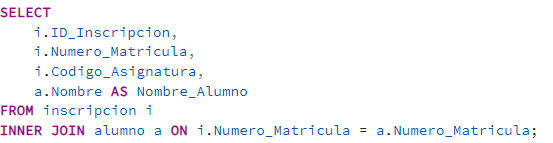
Muestra los **alumnos que NO tienen ninguna inscripción registrada.**

1. Realizar una consulta que permitan mostrar los registros de la **TablaB** que **están relacionados** con la **TablaA**

**INSTRUCCIÓN SQL**:

SELECT i.ID\_Inscripcion, i.Numero\_Matricula, i.Codigo\_Asignatura, a.Nombre AS Nombre\_Alumno FROM inscripcion i

INNER JOIN alumno a ON i.Numero\_Matricula = a.Numero\_Matricula;



  
Muestra solo las **inscripciones que pertenecen a un alumno existente**.  
Así evitamos mostrar inscripciones huérfanas (que no tienen alumno relacionado).

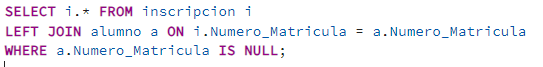
1. Realizar una consulta que permitan mostrar los registros de la **TablaB** que **NO están relacionados** con la **TablaA**

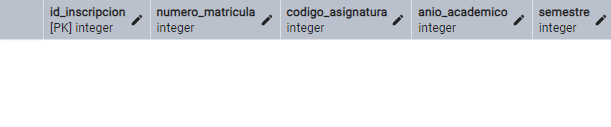
**INSTRUCCIÓN SQL**:

SELECT i.\* FROM inscripcion i

LEFT JOIN alumno a ON i.Numero\_Matricula = a.Numero\_Matricula

WHERE a.Numero\_Matricula IS NULL;





Muestra las **inscripciones que no tienen un alumno asociado**.  
Esto puede suceder si se borró un alumno pero quedó la inscripción guardada.

1. Realizar una consulta que permitan mostrar los registros de la [**TablaA] que están relacionados** con la **TablaC** mediante la [**TablaB**]

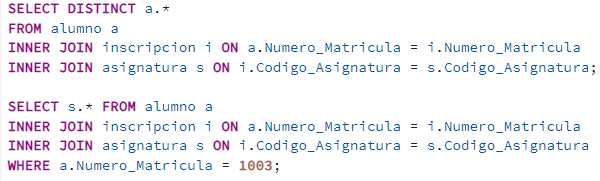
**INSTRUCCIÓN SQL**:

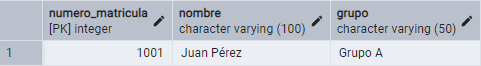
SELECT DISTINCT a.\*

FROM alumno a

INNER JOIN inscripcion i ON a.Numero\_Matricula = i.Numero\_Matricula

INNER JOIN asignatura s ON i.Codigo\_Asignatura = s.Codigo\_Asignatura;





Muestra los alumnos que están inscritos en al menos una asignatura, usando la relación que existe a través de la tabla inscripcion.

1. Supongamos que necesitamos buscar un registro en [**TablaA]** cuya columna **llave primaria** tiene el **valor X,** pero también necesitamos obtener los registros de [**TablaC**] que están relacionados con ese registro de [**TablaA**]. Escriba la consulta SQL necesaria para lograr obtener la información necesaria.

**INSTRUCCIÓN SQL:**

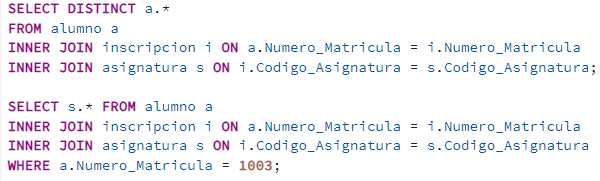
SELECT s.\*

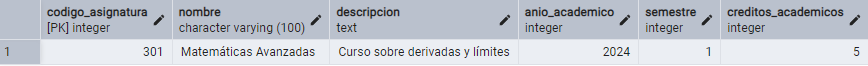
FROM alumno a

INNER JOIN inscripcion i ON a.Numero\_Matricula = i.Numero\_Matricula

INNER JOIN asignatura s ON i.Codigo\_Asignatura = s.Codigo\_Asignatura

WHERE a.Numero\_Matricula = 1003;





Obtiene las asignaturas relacionadas a través de esa inscripción

1. Bibliografía

Libros y recursos sobre bases de datos

Coronel, C., Morris, S., & Rob, P. (2017). Database Systems: Design, Implementation, & Management (12th ed.). Cengage Learning.

Elmasri, R., & Navathe, S. (2015). Fundamentals of Database Systems (7th ed.). Pearson.

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database System Concepts (7th ed.). McGraw-Hill.

1. Herramientas utilizadas

Oracle Corporation. (2024). MySQL 8.0 Reference Manual. Recuperado de https://dev.mysql.com/doc/

Oracle Corporation. (2024). MySQL Workbench. Recuperado de https://www.mysql.com/products/workbench/

Holistics Software. (2024). dbdiagram.io - Database Relationship Diagrams Design Tool. Recuperado de https://dbdiagram.io/

Microsoft Corporation. (2024). Microsoft Word. Recuperado de https://www.microsoft.com/

Google. (2024). Google Docs. Recuperado de https://docs.google.com/

1. Sitios web y documentación

Oracle Corporation. (2024). MySQL Documentation. Recuperado de https://dev.mysql.com/doc/

Holistics Software. (2024). dbdiagram.io Documentation. Recuperado de https://dbdiagram.io/docs

Stack Exchange Inc. (2024). Stack Overflow - SQL Queries and Optimization. Recuperado de https://stackoverflow.com/

W3Schools. (2024). SQL Tutorial. Recuperado de <https://www.w3schools.com/sql/>