

# **Prueba Técnica de SQL y Análisis de Datos**

**Desarrollo de lenguaje SQL, Implementación de lógica y administración de datos, así como seguridad y protección de la información.**

**En el siguiente documento encontrara un compendio de las pruebas impuestas por los reclutadores, están contestadas a mi manera y experiencia, dejo a su consideración mi evolución.**

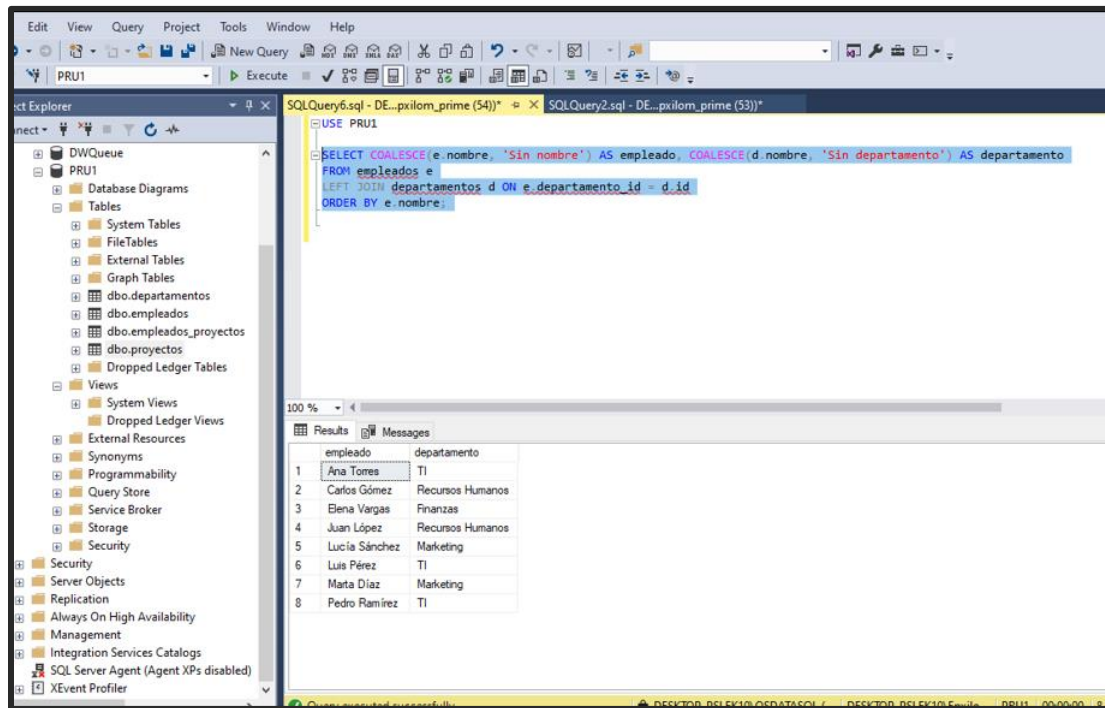
# Prueba Técnica de SQL y Análisis de Datos

## Instrucciones:

- Utiliza únicamente SQL para resolver las siguientes preguntas.
- Algunas preguntas requieren pensar en optimización o diseño de bases de datos.

## PREGUNTAS:

1. Lista el nombre de los empleados junto al nombre de su departamento.



The screenshot shows the Microsoft SQL Server Enterprise Manager interface. On the left, the 'Object Explorer' pane displays the database structure, including tables like 'empleados' and 'departamentos'. The main window shows a SQL query in the 'Query Editor' pane, which is a LEFT JOIN between 'empleados' and 'departamentos'. The 'Results' pane at the bottom displays the output of the query as a table with two columns: 'empleado' and 'departamento'.

```
USE PRU1

SELECT COALESCE(e.nombre, 'Sin nombre') AS empleado, COALESCE(d.nombre, 'Sin departamento') AS departamento
FROM empleados e
LEFT JOIN departamentos d ON e.departamento_id = d.id
ORDER BY e.nombre;
```

	empleado	departamento
1	Ana Torres	TI
2	Carlos Gómez	Recursos Humanos
3	Elena Vargas	Finanzas
4	Juan López	Recursos Humanos
5	Lucía Sánchez	Marketing
6	Luis Pérez	TI
7	Marta Díaz	Marketing
8	Pedro Ramírez	TI

Imagen. 1 pregunta 1.

2. ¿Cuántos empleados hay por departamento?

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the 'Server Objects' tree is expanded to 'Tables', showing tables like 'dbo.departamentos' and 'dbo.empleados'. The main pane displays two SQL queries. The first query is a LEFT JOIN between 'empleados' and 'departamentos' on 'departamento\_id', ordered by employee name. The second query is a LEFT JOIN between 'departamentos' and 'empleados' on 'departamento\_id', grouped by department name and ordered by department name. The 'Results' pane shows the output of the second query:

	departamento	total_empleados
1	Finanzas	1
2	Marketing	2
3	Recursos Humanos	2
4	TI	3

Imagen. 2 pregunta 2

3. Muestra todos los proyectos y cuántos empleados están asignados a cada uno.

The screenshot shows the SQL Server Enterprise Manager interface. The main pane displays a SQL query that is a LEFT JOIN between 'proyectos' and 'empleados\_proyectos' on 'proyecto\_id', grouped by project name and ordered by project name. The 'Results' pane shows the output of the query:

	proyecto	total_empleados
1	Auditoría Fiscal	1
2	Campaña RRHH	2
3	Rebranding	2
4	Rediseño Web	2
5	Sistema Interno	2

Imagen. 3 pregunta 3

4. ¿Cuáles empleados no están asignados a ningún proyecto?

The screenshot shows a SQL query in a text editor and its results in a database client. The query is:

```
SELECT COALESCE(e.nombre, 'Sin nombre') AS empleado
FROM empleados e
LEFT JOIN empleados_proyectos ep ON e.id = ep.empleado_id
WHERE ep.proyecto_id IS NULL
ORDER BY e.nombre;
```

The results pane shows a single row with the name 'Marta Díaz' under the column 'empleado'.

	empleado
1	Marta Díaz

Imagen. 4 pregunta 4

5. ¿Cuál es el salario promedio por departamento?

The screenshot shows a SQL query in a text editor and its results in a database client. The query is:

```
SELECT COALESCE(d.nombre, 'Sin nombre') AS departamento,
       COALESCE(ROUND(AVG(e.salario), 2), 0) AS salario_promedio
FROM departamentos d
LEFT JOIN empleados e ON e.departamento_id = d.id
GROUP BY d.nombre
ORDER BY d.nombre;
```

The results pane shows a table with two columns: 'departamento' and 'salario\_promedio'. The data is as follows:

	departamento	salario_promedio
1	Finanzas	31000.000000
2	Marketing	23500.000000
3	Recursos Humanos	21500.000000
4	TI	28166.670000

Imagen. 5 pregunta 5

6. ¿Qué empleados han participado en más de un proyecto?

The screenshot shows a SQL query in a text editor and its results in a table. The query is as follows:

```
ORDER BY d.nombre;  
  
SELECT e.nombre AS empleado, COUNT(ep.proyecto_id) AS total_proyectos  
FROM empleados e  
JOIN empleados_proyectos ep ON e.id = ep.empleado_id  
GROUP BY e.nombre  
HAVING COUNT(ep.proyecto_id) > 1  
ORDER BY e.nombre;
```

The results table shows two employees who have participated in more than one project:

	empleado	total_proyectos
1	Ana Torres	2
2	Luis Pérez	2

Imagen. 6 pregunta 6

7. ¿Cuál es el proyecto más largo (por días de duración)?

The screenshot shows a SQL query in a text editor and its results in a table. The query is as follows:

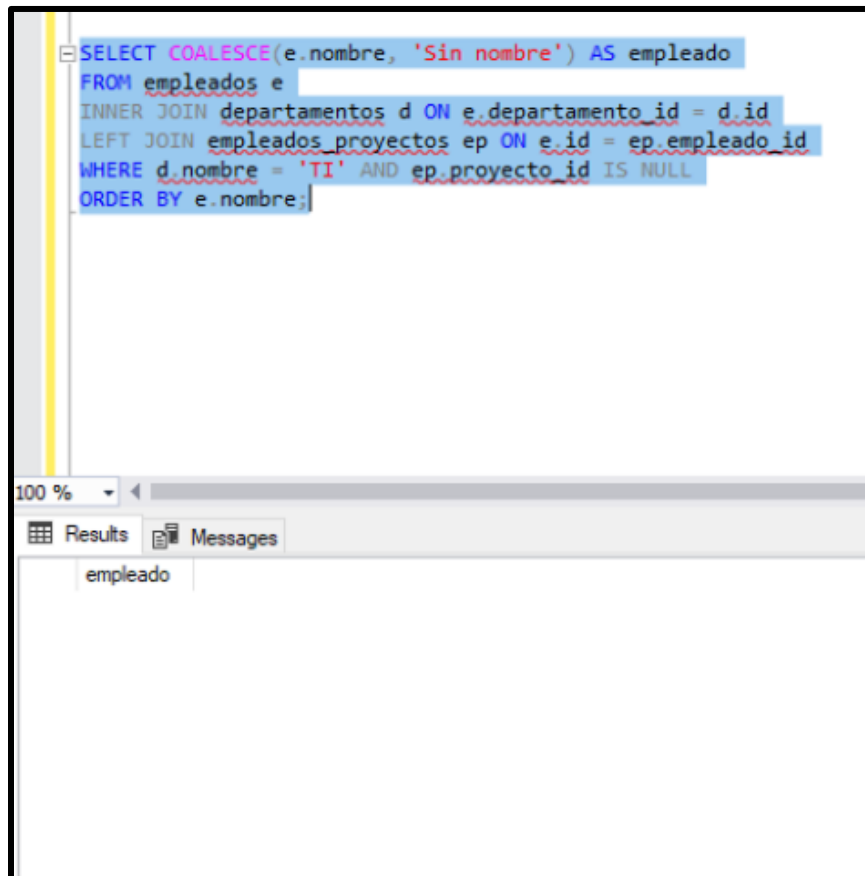
```
SELECT TOP 1  
nombre AS proyecto,  
DATEDIFF(DAY, fecha_inicio, ISNULL(fecha_fin, GETDATE())) AS duracion_dias  
FROM proyectos  
ORDER BY duracion_dias DESC;
```

The results table shows the longest project:

	proyecto	duracion_dias
1	Rebranding	627

Imagen. 7 pregunta 7

8. ¿Qué empleados del departamento de TI no están asignados a ningún proyecto?

A screenshot of a SQL query editor window. The query is written in a syntax-highlighted language. Below the editor, there is a toolbar with 'Results' and 'Messages' buttons. The 'Results' button is active, and a column header 'empleado' is visible in the results pane.

```
SELECT COALESCE(e.nombre, 'Sin nombre') AS empleado
FROM empleados e
INNER JOIN departamentos d ON e.departamento_id = d.id
LEFT JOIN empleados_proyectos ep ON e.id = ep.empleado_id
WHERE d.nombre = 'TI' AND ep.proyecto_id IS NULL
ORDER BY e.nombre;
```

Imagen. 8 pregunta 8

9. ¿Cuál es el top 3 de empleados con mayor salario por departamento?

```
WITH EmpleadosConRanking AS (  
    SELECT  
        COALESCE(e.nombre, 'Sin nombre') AS empleado,  
        COALESCE(d.nombre, 'Sin departamento') AS departamento,  
        e.salario,  
        ROW_NUMBER() OVER (PARTITION BY d.id ORDER BY e.salario DESC) AS posicion  
    FROM empleados e  
    INNER JOIN departamentos d ON e.departamento_id = d.id  
)  
SELECT empleado, departamento, salario  
FROM EmpleadosConRanking  
WHERE posicion <= 3  
ORDER BY departamento, salario DESC;
```

	empleado	departamento	salario
1	Elena Vargas	Finanzas	31000.00
2	Lucía Sánchez	Marketing	24000.00
3	Marta Díaz	Marketing	23000.00
4	Juan López	Recursos Humanos	22000.00
5	Carlos Gómez	Recursos Humanos	21000.00
6	Ana Torres	TI	29000.00
7	Luis Pérez	TI	28000.00
8	Pedro Ramírez	TI	27500.00

Imagen. 9 pregunta 9

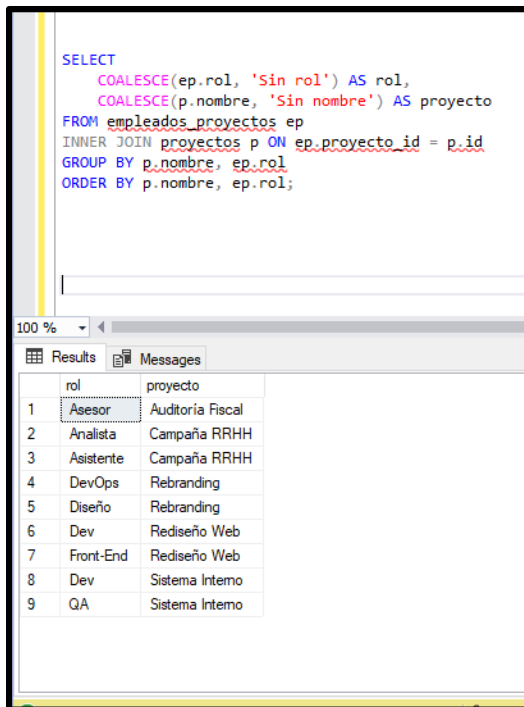
10. Lista los proyectos activos que empezaron en 2024 y que aún no tienen fecha de fin.

```
SELECT nombre AS proyecto, fecha_inicio  
FROM proyectos  
WHERE fecha_inicio >= '2024-01-01'  
    AND fecha_inicio < '2025-01-01'  
    AND fecha_fin IS NULL  
ORDER BY nombre;
```

	proyecto	fecha_inicio
1	Rediseño Web	2024-03-01

Imagen. 10 pregunta 10

## 11. ¿Qué roles únicos existen por proyecto?



The screenshot shows a SQL query in a text editor and its results in a table. The query is as follows:

```
SELECT
    COALESCE(ep.rol, 'Sin rol') AS rol,
    COALESCE(p.nombre, 'Sin nombre') AS proyecto
FROM empleados_proyectos ep
INNER JOIN proyectos p ON ep.proyecto_id = p.id
GROUP BY p.nombre, ep.rol
ORDER BY p.nombre, ep.rol;
```

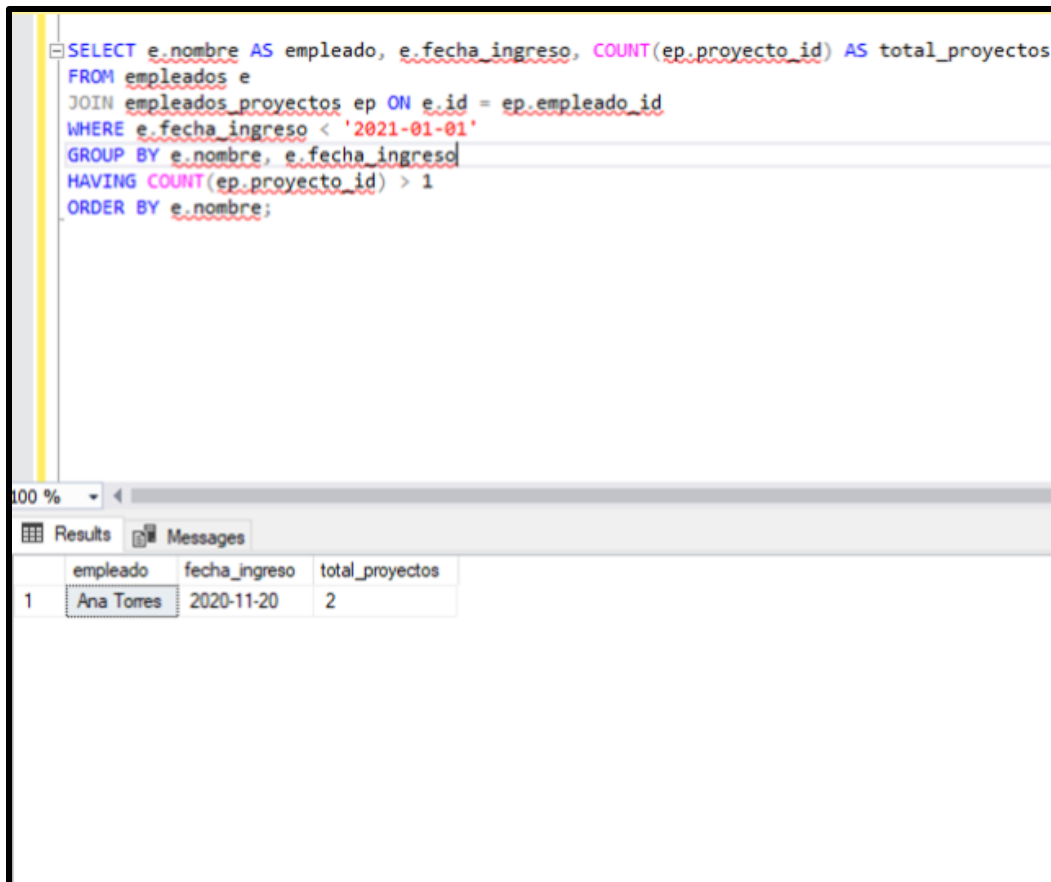
The results are displayed in a table with two columns: 'rol' and 'proyecto'. The table contains 9 rows of data, with the first row highlighted.

	rol	proyecto
1	Asesor	Auditoría Fiscal
2	Analista	Campaña RRHH
3	Asistente	Campaña RRHH
4	DevOps	Rebranding
5	Diseño	Rebranding
6	Dev	Rediseño Web
7	Front-End	Rediseño Web
8	Dev	Sistema Interno
9	QA	Sistema Interno

Imagen. 11 pregunta 11



12. ¿Qué empleados fueron contratados antes de 2021 y están asignados a más de un proyecto?



The screenshot shows a SQL query in the Enterprise Manager query window. The query is as follows:

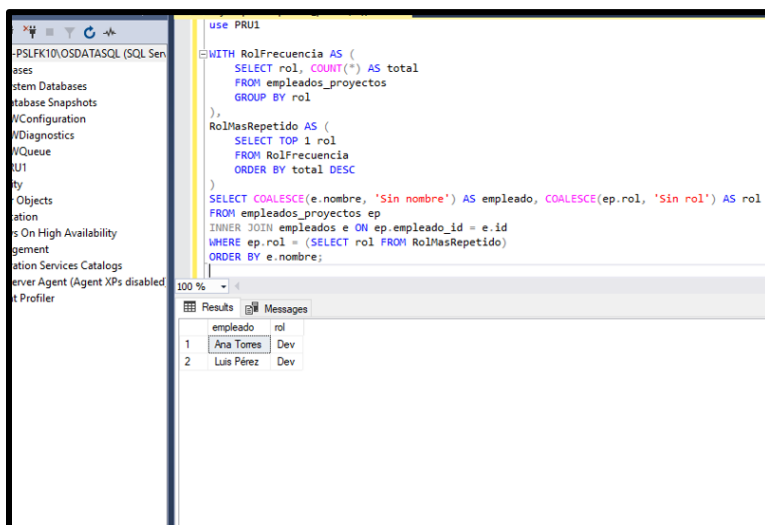
```
SELECT e.nombre AS empleado, e.fecha_ingreso, COUNT(ep.proyecto_id) AS total_proyectos
FROM empleados e
JOIN empleados_proyectos ep ON e.id = ep.empleado_id
WHERE e.fecha_ingreso < '2021-01-01'
GROUP BY e.nombre, e.fecha_ingreso
HAVING COUNT(ep.proyecto_id) > 1
ORDER BY e.nombre;
```

The Results pane shows the following data:

	empleado	fecha_ingreso	total_proyectos
1	Ana Torres	2020-11-20	2

Imagen. 12 pregunta 12

13. ¿Qué empleado tiene asignado el rol más repetido entre todos los proyectos?



The screenshot shows a SQL query in the Enterprise Manager query window. The query is as follows:

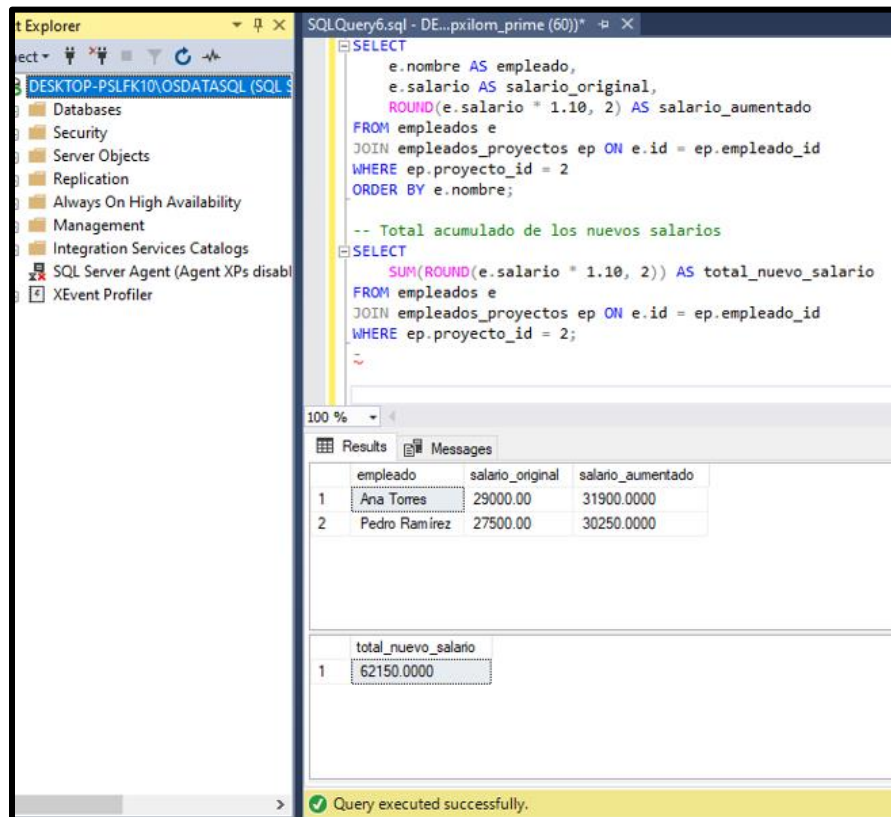
```
use PRU1
WITH RolFrecuencia AS (
    SELECT rol, COUNT(*) AS total
    FROM empleados_proyectos
    GROUP BY rol
),
RolMasRepetido AS (
    SELECT TOP 1 rol
    FROM RolFrecuencia
    ORDER BY total DESC
)
SELECT COALESCE(e.nombre, 'Sin nombre') AS empleado, COALESCE(ep.rol, 'Sin rol') AS rol
FROM empleados_proyectos ep
INNER JOIN empleados e ON ep.empleado_id = e.id
WHERE ep.rol = (SELECT rol FROM RolMasRepetido)
ORDER BY e.nombre;
```

The Results pane shows the following data:

	empleado	rol
1	Ana Torres	Dev
2	Luis Pérez	Dev

Imagen. 13 pregunta 13

14. Si decides aumentar un 10% el salario de todos los empleados asignados al proyecto 2, ¿cuál sería el nuevo salario total de esos empleados?



The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Server Explorer' with the 'Databases' folder expanded. The right pane shows a SQL query window with the following code:

```
SELECT
    e.nombre AS empleado,
    e.salario AS salario_original,
    ROUND(e.salario * 1.10, 2) AS salario_aumentado
FROM empleados e
JOIN empleados_proyectos ep ON e.id = ep.empleado_id
WHERE ep.proyecto_id = 2
ORDER BY e.nombre;

-- Total acumulado de los nuevos salarios
SELECT
    SUM(ROUND(e.salario * 1.10, 2)) AS total_nuevo_salario
FROM empleados e
JOIN empleados_proyectos ep ON e.id = ep.empleado_id
WHERE ep.proyecto_id = 2;
```

The 'Results' tab shows the output of the query. The first table lists individual employees and their salaries, and the second table shows the total new salary.

empleado	salario_original	salario_aumentado
1 Ana Torres	29000.00	31900.0000
2 Pedro Ramirez	27500.00	30250.0000

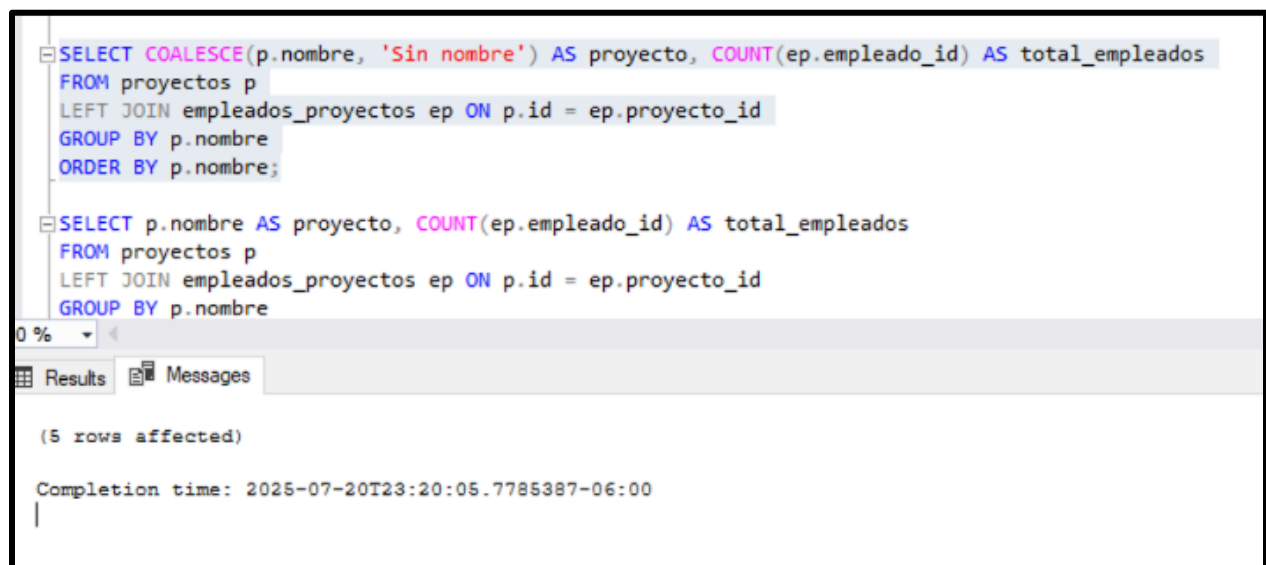
  

total_nuevo_salario
1 62150.0000

The status bar at the bottom indicates 'Query executed successfully.'

Imagen. 14 pregunta 14

15. ¿Qué índice agregarías para mejorar la consulta 3 si empleados\_proyectos tuviera 5 millones de filas?



The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Server Explorer' with the 'Databases' folder expanded. The right pane shows a SQL query window with the following code:

```
SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

SELECT p.nombre AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
```

The 'Results' tab shows the output of the query. The first table lists individual employees and their salaries, and the second table shows the total new salary.

empleado	salario_original	salario_aumentado
1 Ana Torres	29000.00	31900.0000
2 Pedro Ramirez	27500.00	30250.0000

total_nuevo_salario
1 62150.0000

The status bar at the bottom indicates 'Query executed successfully.'

Imagen. 15 pregunta 15, 1 de 6

```
--P3
SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

SELECT p.nombre AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
```

100 %

Results Messages

	proyecto	total_empleados
1	Auditoría Fiscal	1
2	Campaña RRHH	2
3	Rebranding	2
4	Rediseño Web	2
5	Sistema Interno	2

Imagen. 16 pregunta 15, 2 de 6

```
ORDER BY d.nombre;
ORDER BY d.nombre;

--P3
SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

SELECT p.nombre AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;
```

100 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.empleado\_id) AS total\_empleados FROM proyectos p LEFT JOIN empleados\_proyect...

```

graph LR
    SELECT[SELECT  
Cost: 0 %] --> CS1[Compute Scalar  
Cost: 0 %]
    CS1 --> CS2[Compute Scalar  
Cost: 0 %]
    CS2 --> SA[Stream Aggregate  
(Aggregate)  
Cost: 0 %]
    SA --> NL[Nested Loops  
(Left Outer Join)  
Cost: 1 %]
    NL --> S[Sort  
Cost: 61 %]
    S --> CIS1[Clustered Index Scan (Clustered)  
[proyectos].[PK_proyecto_3213E83F]  
Cost: 18 %]
    S --> CIS2[Clustered Index Scan (Clustered)  
[empleados_proyectos].[FK_empleado_...]  
Cost: 20 %]
  
```

Imagen. 17 pregunta 15, 3 de 6

```

-- Total acumulado de los nuevos salarios
SELECT
    SUM(ROUND(e.salario * 1.10, 2)) AS total_nuevo_salario
FROM empleados e
JOIN empleados_proyectos ep ON e.id = ep.empleado_id
WHERE ep.proyecto_id = 2;

--P15

CREATE INDEX IX_empleados_proyectos_proyecto_id
ON empleados_proyectos(proyecto_id);

```

100 %

Messages

Commands completed successfully.

Completion time: 2023-07-20T23:42:43.3677069-06:00

Imagen. 18 pregunta 15, 4 de 6

```

ORDER BY d.nombre;

--P3

SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

SELECT p.nombre AS proyecto, COUNT(ep.empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

```

0 %

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.empleado\_id) AS total\_empleados FROM proyectos p LEFT JOIN empleados\_proyectos ep ON p.id = ep.proyecto\_id GROUP BY p.nombre ORDER BY p.nombre;

Query executed successfully.

DESKTOP-PSLFK10\OSDATASQL (...)

DESKTOP-PSLFK10\Epxilo...

PRU1 00:00:00 0 rows

Imagen. 19 pregunta 15, 5 de 6

```
--P3
SELECT COALESCE(p.nombre, 'Sin nombre') AS proyecto, COUNT(ep.Empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

SELECT p.nombre AS proyecto, COUNT(ep.Empleado_id) AS total_empleados
FROM proyectos p
LEFT JOIN empleados_proyectos ep ON p.id = ep.proyecto_id
GROUP BY p.nombre
ORDER BY p.nombre;

SELECT COALESCE(e.nombre, 'Sin nombre') AS empleado
FROM empleados e
```

100 %

Results Messages

	proyecto	total_empleados
1	Auditoría Fiscal	1
2	Campaña RRHH	2
3	Rebranding	2
4	Rediseño Web	2
5	Sistema Interno	2

Imagen. 20 pregunta 15, 6 de 6

16. ¿Qué alternativa usarías a JOIN si quisieras optimizar la consulta 4 con subconsultas?

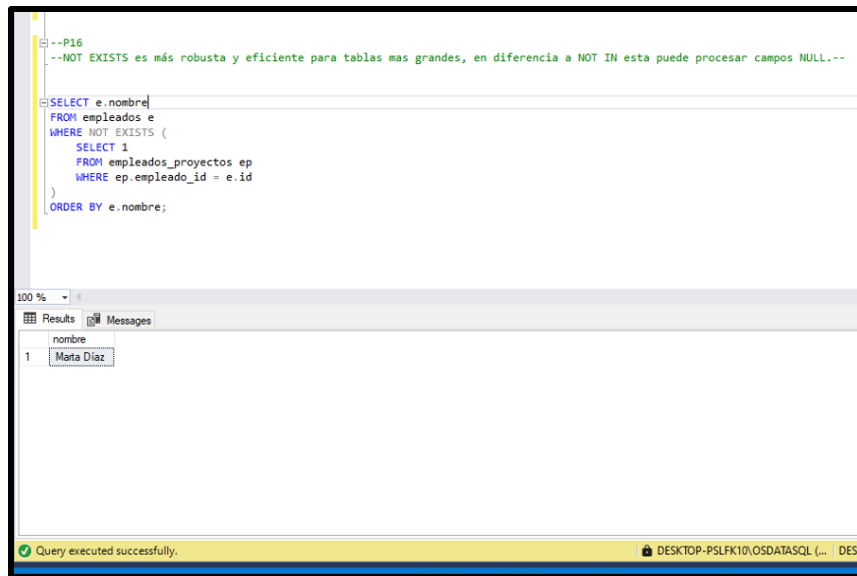


Imagen. 21 pregunta 16

17. ¿Cómo protegerías la información de salario y asignación si esta fuera una base de datos real de empleados con acceso público limitado?

```
USE PRU2;
GO
-- 1. CONTROL DE ACCESO CON ROLES Y VISTAS

--1 Creacion roles de seguridad
CREATE ROLE rol_publico;
CREATE ROLE rol_rh;
GO

-- 2 Revocar SELECT directo sobre tablas sensibles
DENY SELECT ON dbo.empleados TO rol_publico;
DENY SELECT ON dbo.empleados_proyectos TO rol_publico;
GO

--3 Crear vistas con control de acceso

-- Vista publica sin salarios ni asignaciones
CREATE VIEW vista_empleados_publica AS
SELECT id, nombre, departamento_id
FROM dbo.empleados;
GO

-- 4 Vista solo para RH
```

```

CREATE VIEW vista_empleados_rh AS
SELECT id, nombre, salario, departamento_id
FROM dbo.empleados
WHERE departamento_id = 2; -- Recursos Humanos
GO

-- 5 Conceder acceso solo a las vistas
GRANT SELECT ON vista_empleados_publica TO rol_publico;
GRANT SELECT ON vista_empleados_rh TO rol_rh;
GO

-- 6 Cifrado para columna con Always Encrypted, se debe configurar desde la
terminal PowerShell o SSMS.
-- script es referencial y solo se ejecuta definiendo las claves v lidas.

-- clave maestra y clave de cifrado (solo referencial aqu )
-- Estas acciones normalmente se hacen en el asistente de Always Encrypted

/*
CREATE COLUMN MASTER KEY CMK_Seguridad
WITH (
    KEY_STORE_PROVIDER_NAME = 'MSSQL_CERTIFICATE_STORE',
    KEY_PATH = 'CurrentUser/My/CN=ClaveAlwaysEncrypted'
);
GO

CREATE COLUMN ENCRYPTION KEY CEK_Seguridad
WITH VALUES (
    COLUMN_MASTER_KEY = CMK_Seguridad,
    ALGORITHM = 'RSA_OAEP',
    ENCRYPTED_VALUE = <valor_cifrado>);
GO

-- Aplicar cifrado a la columna salario
ALTER TABLE empleados
ALTER COLUMN salario
ADD ENCRYPTED WITH (
    ENCRYPTION_TYPE = RANDOMIZED,
    COLUMN_ENCRYPTION_KEY = CEK_Seguridad );
GO
*/

-- 7. Enmascaramiento de datos (DDM)

-- Verificar si ya existe enmascaramiento, y aplicarlo si no

```

```
-- IMPORTANTE: solo se puede aplicar si la tabla es nueva o no tiene  
restricciones de encriptado.
```

```
-- Aplicar máscara a salario  
-- *Solo si no tiene encriptación activa*
```

```
ALTER TABLE empleados  
ALTER COLUMN salario  
ADD MASKED WITH (FUNCTION = 'default()');  
GO
```

```
-- Prueba con usuario sin privilegios  
-- SELECT nombre, salario FROM empleados;
```

```
-- 8.Registro( AUDITORÍA DE ACCESOS).  
-- Crear auditoría (a nivel servidor)  
CREATE SERVER AUDIT Audit_Salarios  
TO FILE (FILEPATH = 'C:\AuditLogs\')  
WITH (ON_FAILURE = CONTINUE);  
GO
```

```
-- Activar auditoría  
ALTER SERVER AUDIT Audit_Salarios  
WITH (STATE = ON);  
GO
```

```
-- Crear especificación a nivel de base de datos  
CREATE DATABASE AUDIT SPECIFICATION AuditSelectSalario  
FOR SERVER AUDIT Audit_Salarios  
ADD (SELECT ON OBJECT::dbo.empleados BY PUBLIC)  
WITH (STATE = ON);  
GO
```



18. ¿Cómo modificarías la tabla empleados\_proyectos para guardar un historial temporal de asignaciones, con fechas de inicio y fin por cada asignación?

```
ALTER TABLE empleados_proyectos
ADD fecha_inicio DATE;

-- Luego agregamos 'fecha_fin'
ALTER TABLE empleados_proyectos
ADD fecha_fin DATE;

ALTER TABLE empleados_proyectos
ADD CONSTRAINT chk_fechas_validas CHECK (fecha_fin IS NULL OR fecha_fin >= fecha_inicio);

SP_HELP EMPLEADOS_PROYECTOS;
```

%

Results Messages

Name	Owner	Type	Created_datetime
empleados_proyectos	dbo	user table	2025-07-21 18:12:20.680

Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
empleado_id	int	no	4	10	0	no	(n/a)	(n/a)	NULL
proyecto_id	int	no	4	10	0	no	(n/a)	(n/a)	NULL
rol	varchar	no	100			yes	no	yes	Modern_Spanish_CI_AS
fecha_inicio	date	no	3	10	0	yes	(n/a)	(n/a)	NULL
fecha_fin	date	no	3	10	0	yes	(n/a)	(n/a)	NULL

Identity	Seed	Increment	Not For Replication
No identity column defined.	NULL	NULL	NULL

RowGuidCol
No rowguidcol column defined.

Query executed successfully. DESKTOP-PSLFK10\OSDATASQL (... | DESKTOP-PSLFK10\Epxilo... | PRU3

Imagen. 22 pregunta 18, 1 de 4

0 %

Results Messages

Name	Owner	Type	Created_datetime
empleados_proyectos	dbo	user table	2025-07-21 18:12:20.680

Column_name	Type	Computed	Length	Prec	Scale	Nullable	TrimTrailingBlanks	FixedLenNullInSource	Collation
empleado_id	int	no	4	10	0	no	(n/a)	(n/a)	NULL
proyecto_id	int	no	4	10	0	no	(n/a)	(n/a)	NULL
rol	varchar	no	100			yes	no	yes	Modern_Spanish_CI_AS
fecha_inicio	date	no	3	10	0	yes	(n/a)	(n/a)	NULL
fecha_fin	date	no	3	10	0	yes	(n/a)	(n/a)	NULL

Identity	Seed	Increment	Not For Replication
No identity column defined.	NULL	NULL	NULL

RowGuidCol
No rowguidcol column defined.

Data_located_on_filegroup
PRIMARY

index_name	index_description	index_keys
PK__empleado__65C8F638347039D8	clustered, unique, primary key located on PRIMARY	empleado_id, proyecto_id

constraint_type	constraint_name	delete_action	update_action	status_enabled	status_for_replication	constraint_keys
CHECK Table Level	chk_fechas_validas	(n/a)	(n/a)	Enabled	Is_For_Replication	([fecha_fin] IS NULL OR [fecha_fin]>=[fe
FOREIGN KEY	FK_empleados__emple__3E52440B	No Action	No Action	Enabled	Is_For_Replication	empleado_id
FOREIGN KEY	FK_empleados__proye__3F466844	No Action	No Action	Enabled	Is_For_Replication	REFERENCES PRU3.dbo.empleados (id
						proyecto_id
						REFERENCES PRU3.dbo.proyectos (id)
PRIMARY KEY (cl...	PK__empleado__65C8F638347039...	(n/a)	(n/a)	(n/a)	(n/a)	empleado_id, proyecto_id

Imagen. 23 pregunta 18, 2 de 4

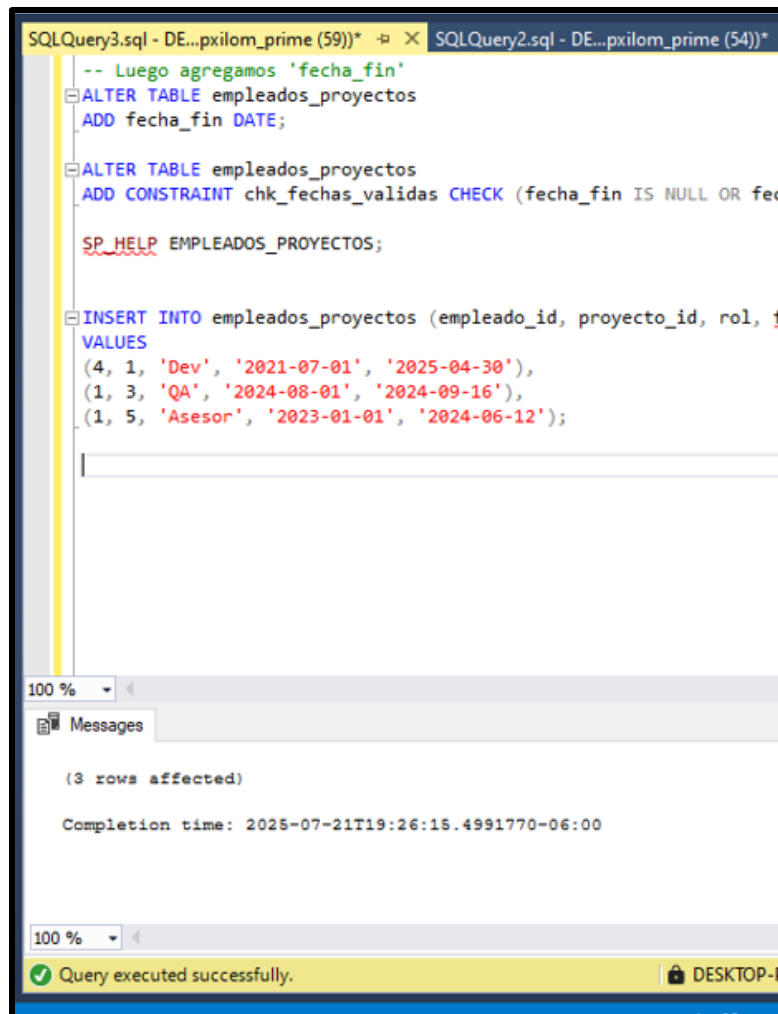


Imagen. 24 pregunta 18, 3 de 4

```

UPDATE ep
SET
    fecha_inicio = p.fecha_inicio,
    fecha_fin = p.fecha_fin
FROM
    empleados_proyectos ep
JOIN
    proyectos p ON ep.proyecto_id = p.id
WHERE
    ep.fecha_inicio IS NULL OR ep.fecha_fin IS NULL

```

Results Messages

empleado_id	proyecto_id	rol	fecha_inicio	fecha_fin
1	1	Dev	2024-01-01	2024-06-01
1	3	QA	2024-08-01	2024-09-16
1	4	DevOps	2023-11-01	NULL
1	5	Asesor	2023-01-01	2024-06-12
2	1	QA	2024-01-01	2024-06-01
2	2	Dev	2024-03-01	NULL
3	3	Analista	2023-10-15	2024-04-01
4	1	Dev	2021-07-01	2025-04-30
5	3	Asistente	2023-10-15	2024-04-01

Query executed successfully.

Imagen. 25 pregunta 18, 4 de 4

```

create database PRU3;
use PRU3

-- Tabla de departamentos
CREATE TABLE departamentos (
    id INT PRIMARY KEY,
    nombre VARCHAR(100)
);

-- Tabla de empleados
CREATE TABLE empleados (
    id INT PRIMARY KEY,
    nombre VARCHAR(100),
    departamento_id INT,
    salario DECIMAL(10, 2),
    fecha_ingreso DATE,
    FOREIGN KEY (departamento_id) REFERENCES departamentos(id)
);

```

```

-- Tabla de proyectos
CREATE TABLE proyectos (
    id INT PRIMARY KEY,
    nombre VARCHAR(100),
    fecha_inicio DATE,
    fecha_fin DATE
);

-- Relación muchos a muchos: empleados asignados a proyectos
CREATE TABLE empleados_proyectos (
    empleado_id INT,
    proyecto_id INT,
    rol VARCHAR(100),
    PRIMARY KEY (empleado_id, proyecto_id),
    FOREIGN KEY (empleado_id) REFERENCES empleados(id),
    FOREIGN KEY (proyecto_id) REFERENCES proyectos(id)
);

-- Inserts
INSERT INTO departamentos VALUES
(1, 'TI'), (2, 'Recursos Humanos'), (3, 'Marketing'), (4, 'Finanzas');

INSERT INTO empleados VALUES
(1, 'Luis Pérez', 1, 28000, '2021-05-10'),
(2, 'Ana Torres', 1, 29000, '2020-11-20'),
(3, 'Carlos Gómez', 2, 21000, '2023-01-01'),
(4, 'Marta Díaz', 3, 23000, '2022-03-15'),
(5, 'Juan López', 2, 22000, '2019-09-09'),
(6, 'Elena Vargas', 4, 31000, '2020-02-10'),
(7, 'Pedro Ramírez', 1, 27500, '2023-07-01'),
(8, 'Lucía Sánchez', 3, 24000, '2023-02-01');

INSERT INTO proyectos VALUES
(1, 'Sistema Interno', '2024-01-01', '2024-06-01'),
(2, 'Rediseño Web', '2024-03-01', NULL),
(3, 'Campaña RRHH', '2023-10-15', '2024-04-01'),
(4, 'Rebranding', '2023-11-01', NULL),
(5, 'Auditoría Fiscal', '2023-07-01', '2023-12-15');

INSERT INTO empleados_proyectos VALUES
(1, 1, 'Dev'),
(2, 1, 'QA'),
(2, 2, 'Dev'),
(3, 3, 'Analista'),
(5, 3, 'Asistente'),

```

```

(6, 5, 'Asesor'),(7, 2, 'Front-End'),
(1, 4, 'DevOps'),
(8, 4, 'Diseno');

ALTER TABLE empleados_proyectos
ADD fecha_inicio DATE;

-- Luego agregamos 'fecha_fin'
ALTER TABLE empleados_proyectos
ADD fecha_fin DATE;

ALTER TABLE empleados_proyectos
ADD CONSTRAINT chk_fechas_validas CHECK (fecha_fin IS NULL OR fecha_fin >=
fecha_inicio);

SP_HELP EMPLEADOS_PROYECTOS;

INSERT INTO empleados_proyectos (empleado_id, proyecto_id, rol, fecha_inicio,
fecha_fin)
VALUES
(4, 1, 'Dev', '2021-07-01', '2025-04-30'),
(1, 3, 'QA', '2024-08-01', '2024-09-16'),
(1, 5, 'Asesor', '2023-01-01', '2024-06-12');

SELECT * FROM empleados_proyectos;


INSERT INTO empleados_proyectos (empleado_id, proyecto_id, rol, fecha_inicio)
VALUES (2, 2, 'Analista', '2024-03-15');

SELECT * FROM empleados_proyectos;

UPDATE ep
SET
    fecha_inicio = p.fecha_inicio,
```

```

    fecha_fin = p.fecha_fin
FROM
    empleados_proyectos ep
JOIN
    proyectos p ON ep.proyecto_id = p.id
WHERE
    ep.fecha_inicio IS NULL OR ep.fecha_fin IS NULL;

```

## ADICIONAL:

Con las preguntas 1,2 y 5 generar algún frontend simple y mostrar esas preguntas en forma de tablas. No debe ser nada muy complejo necesariamente mientras permita visualizar los datos en un front simulando que sería lo que va a visualizar un usuario final.

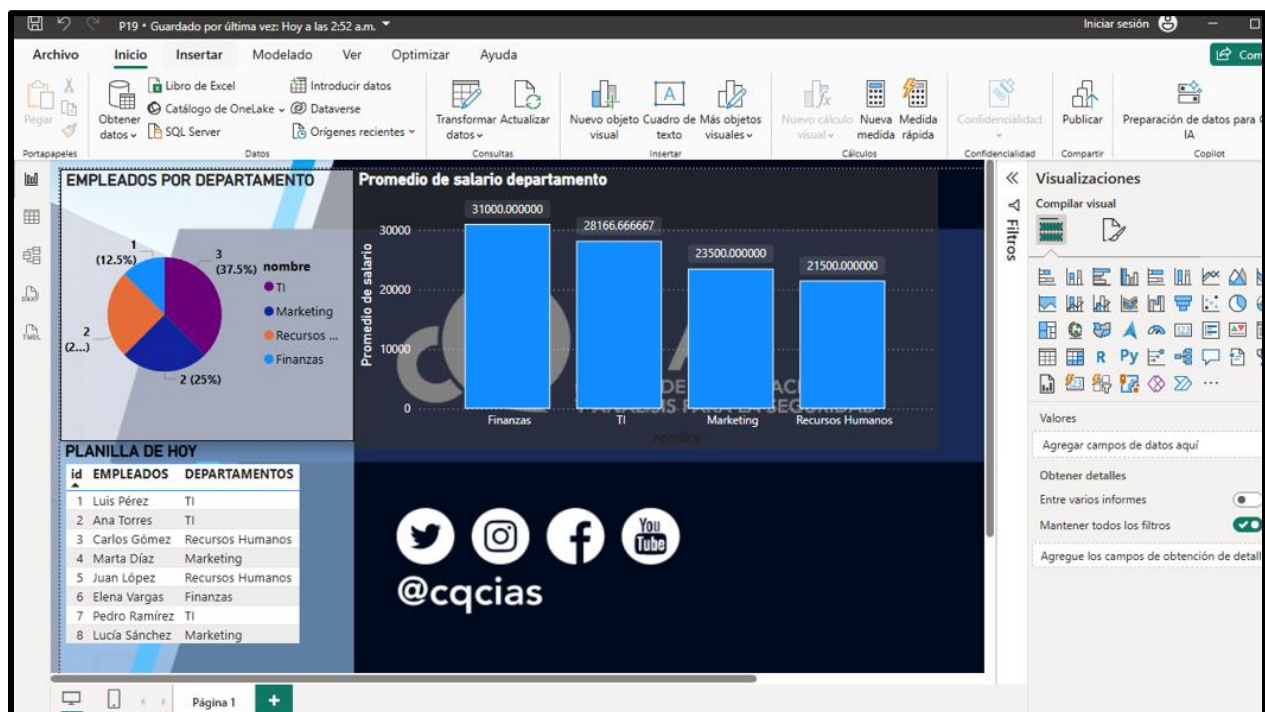


Imagen. 26 Adicional, 1 de 2

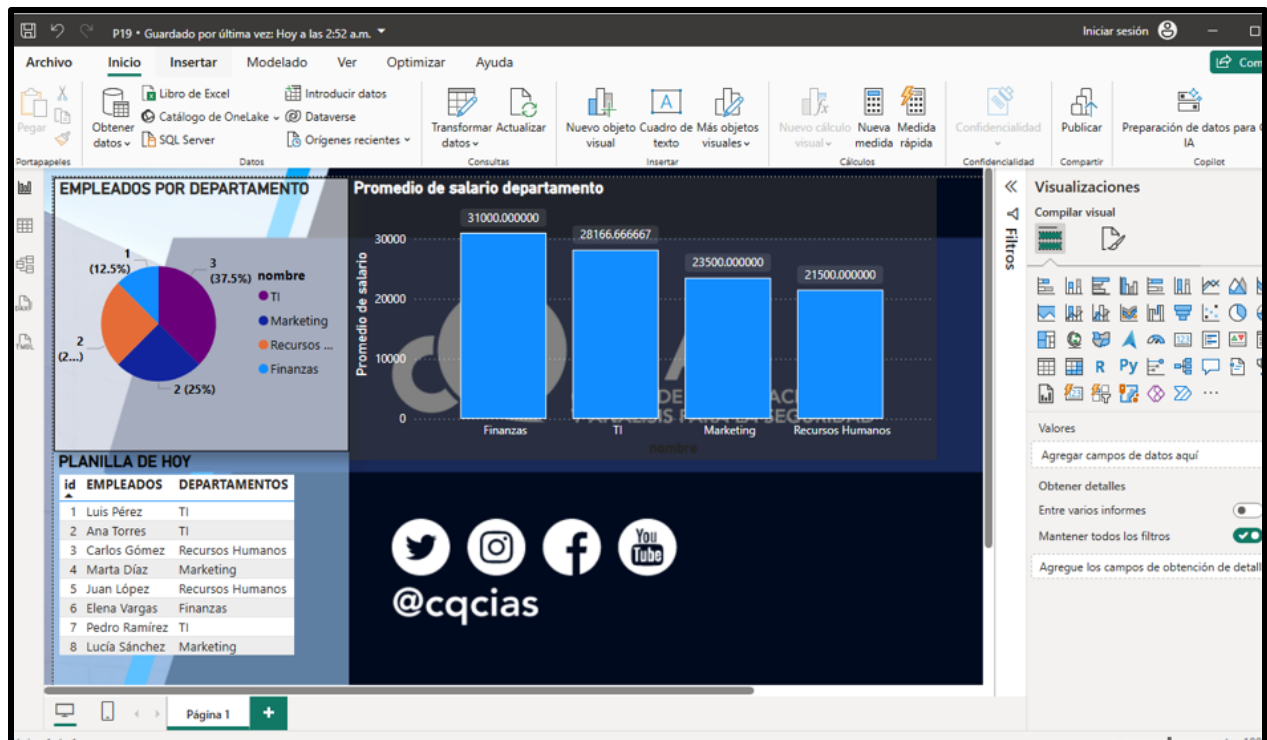


Imagen. 27 Adicional, 2 de 2