

# **PRUEBA TÉCNICA QA**

## Componente practico

Wilder Adolfo Perez Amador



RappiPay

# ★ contextualizacion

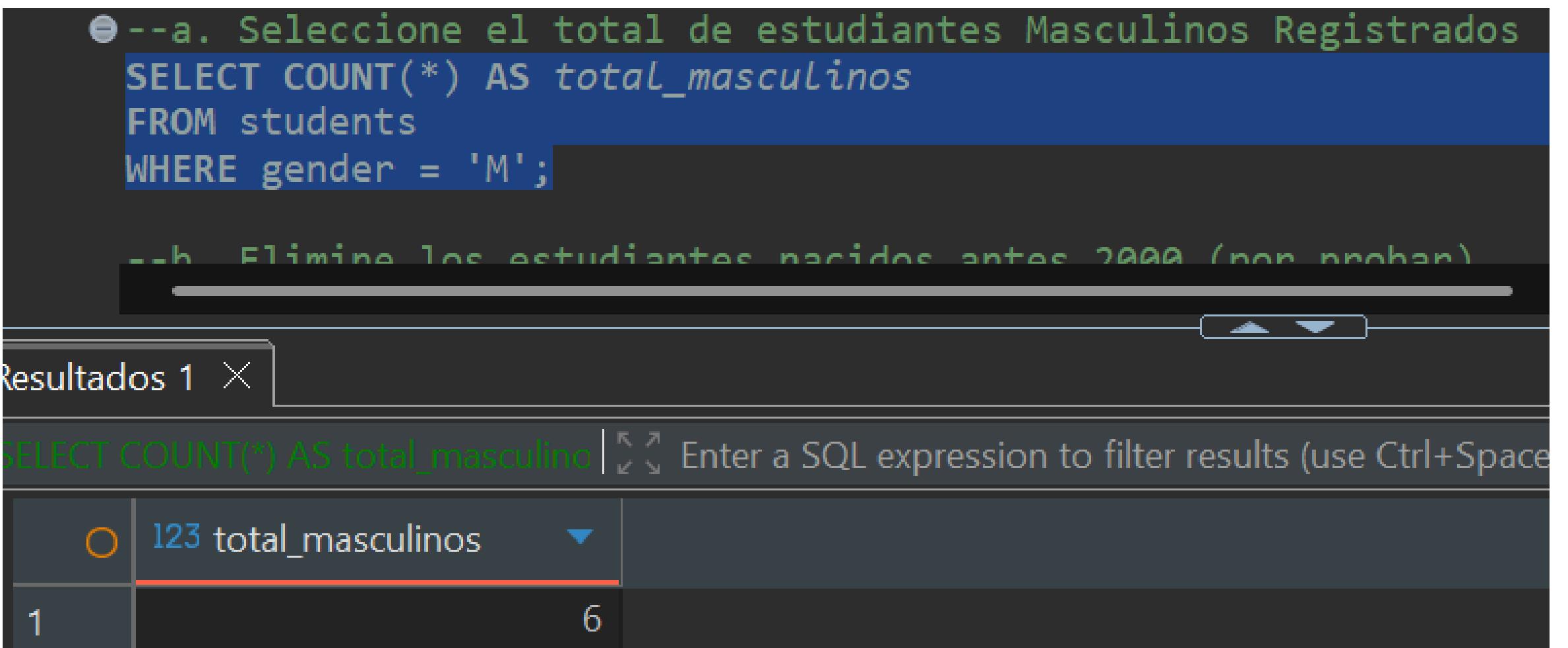
Para resolver el ejercicio primero se construyó una base de datos relacional con las tablas students, careers y career\_student, respetando claves primarias y foráneas.

Sobre esta base se ejecutaron consultas de lectura, actualización y eliminación, cuidando la integridad referencial y validando los resultados antes y después de cada operación.

de igual forma se adjunta un archivo llamado sql-exercises.sql el cual tiene los scripts de creacion de la bd y los ejercicios aca presentados.

# ★ Seleccione el total de estudiantes Masculinos Registrados

```
SELECT COUNT(*) AS total_masculinos  
FROM students  
WHERE gender = 'M';
```



The screenshot shows a SQL editor interface with the following components:

- Query Bar:** Displays the SQL query: `--a. Seleccione el total de estudiantes Masculinos Registrados  
SELECT COUNT(*) AS total_masculinos  
FROM students  
WHERE gender = 'M';`
- Status Bar:** Shows the status message: `--b Elimine los estudiantes nacidos antes 2000 (no proban)`.
- Results Tab:** Labeled "Resultados 1" with a close button.
- Result Grid:** A table with one row showing the result of the query.

	l23 total_masculinos
1	6
- Filter Bar:** An input field with placeholder text: `Enter a SQL expression to filter results (use Ctrl+Space)`.

# ★ Elimine los estudiantes nacidos antes de 2000

```
SELECT student_id, name, birthdate  
FROM students
```

```
WHERE birthdate < '2000-01-01'  
ORDER BY birthdate;
```

--2 eliminar estudiantes nacidos antes del 2000

```
DELETE FROM career_student
```

```
WHERE student_id IN (
```

```
SELECT student_id
```

```
FROM students
```

```
WHERE birthdate < '2000-01-01'
```

```
);
```

```
SELECT changes() AS carreras_borradas;
```

```
DELETE FROM students
```

```
WHERE birthdate < '2000-01-01';
```

```
SELECT changes() AS estudiantes_borrados;
```

The screenshot shows a MySQL command-line interface. At the top, a query is being typed:

```
SELECT student_id, name, birthdate  
FROM students  
WHERE birthdate < '2000-01-01'  
ORDER BY birthdate;
```

Below the query, a message indicates: "2 eliminar estudiantes nacidos antes del 2000". The results of the query are displayed in a table:

student_id	name	birthdate
9.000.189	ELENA CASTRO	1972-02-18
9.000.185	PEDRO GOMEZ	1985-10-30
9.000.194	MONICA FIGUEROA	1987-05-06
9.000.190	MATIAS PEREZ	1993-11-01
9.000.186	JUAN HERNANDEZ	1995-01-25
9.000.193	CAROLINA LUNA	1997-07-21

The screenshot shows a MySQL command-line interface. A variable is being set:

```
123 carreras_borradas
```

The value of the variable is displayed as 10.

The screenshot shows a MySQL command-line interface. A variable is being set:

```
123 estudiantes_borrados
```

The value of the variable is displayed as 8.

★ Actualice la descripción de la carrera con id '10026' de "Administración" a "Administración de Empresas"

```
UPDATE careers  
SET description = 'Administración de Empresas'  
WHERE career_id = 10026;
```

```
--C. Actualice la descripción de la carrera con id '10026' de "Administración" a "Administración de Empresas"  
UPDATE careers  
SET description = 'Administración de Empresas'  
WHERE career_id = 10026;  
  
Estadísticas 1 X  


|           | Value                                                                                       |
|-----------|---------------------------------------------------------------------------------------------|
| Rows      | 1                                                                                           |
| Time      | 0,005s                                                                                      |
| Date      | Mon Feb 02 21:03:54 COT 2026                                                                |
| Statement | Mon Feb 02 21:03:54 COT 2026                                                                |
| SQL       | UPDATE careers<br>SET description = 'Administración de Empresas'<br>WHERE career_id = 10026 |


```

# ★Muestre los nombres de los estudiantes que estén cursando dos o más materias

```
SELECT s.name  
FROM students s  
JOIN career_student cs ON cs.student_id =  
s.student_id  
GROUP BY s.student_id, s.name  
HAVING COUNT(*) >= 2;
```

The screenshot shows a SQL editor interface with the following details:

- Query:** A green-highlighted comment at the top reads: `--d. Muestre los nombres de los estudiantes que estén cursando dos o más materias`. Below it is the actual SQL code:

```
SELECT s.name  
FROM students s  
JOIN career_student cs ON cs.student_id = s.student_id  
GROUP BY s.student_id, s.name  
HAVING COUNT(*) >= 2;
```
- Results:** The results pane shows a table with one row, labeled "Students 1". The table has two columns: "name" and "description". The "name" column contains four entries: PEDRO GOMEZ, JUAN HERNANDEZ, MATIAS PEREZ, and SEBASTIAN VARGAS. The "description" column is partially visible as "Enter a SQL expression to filter results (use Ctrl+Space)".
- Filter:** A dropdown menu at the bottom left is set to "A-Z name".

# ★Calificación promedio de todos los estudiantes que estén cursando carreras

```
SELECT AVG(cs.calification) AS  
promedio_calificacion  
FROM career_student cs;
```

The screenshot shows a MySQL command-line interface. The top part displays the SQL query:

```
--e. Calificación promedio de todos los es  
SELECT AVG(cs.calification) AS promedio_c  
FROM career_student cs;
```

The bottom part shows the results of the query:

tados 1	
CT s.name FROM students s JOIN	Enter a SQL expression
O 123 promedio_calificacion	▼
	76,4615384615

★Muestra el nombre, carrera id, correos de los estudiantes que tienen una calificación mayor 70

```
SELECT s.name,  
       cs.career_id,  
       s.email,  
       cs.calification  
  FROM students s  
 JOIN career_student cs ON cs.student_id =  
       s.student_id  
 WHERE cs.calification > 70;
```

The screenshot shows a database query editor with the following details:

- Query:** A SQL SELECT statement is displayed in the top pane:

```
--f. Muestra el nombre, carrera id, correos de los estudiantes que tienen una calificación mayor 70  
SELECT s.name,  
       cs.career_id,  
       s.email,  
       cs.calification  
  FROM students s  
 JOIN career_student cs ON cs.student_id = s.student_id  
 WHERE cs.calification > 70;
```
- Results:** The bottom pane displays the results of the query in a table format. The table has four columns: name, career\_id, email, and calification. The data is as follows:

	A-Z name	123 career_id	A-Z email	123 calification
1	PEDRO GOMEZ	10.025	PEDRO.GOMEZ@GMAIL.COM	
2	JUAN HERNANDEZ	10.030	Juan.Hernandez@gmail.com	
3	SEBASTIAN VARGAS	10.028	SEBASTIAN.VARGAS@OUTLOC	
4	ELENA CASTRO	10.025	elena.casto@hotmail.com	
5	SEBASTIAN VARGAS	10.026	SEBASTIAN.VARGAS@OUTLOC	

# ★Cuántos estudiantes que están cursando 2 o más carreras

```
SELECT COUNT(*) AS
estudiantes_con_2_o_mas_carreras
FROM (
SELECT cs.student_id
FROM career_student cs
GROUP BY cs.student_id
HAVING COUNT(*) >= 2
) t;
```

The screenshot shows a SQL editor interface with the following content:

```
--g. Cuántos estudiantes que están cursando 2 o más carreras
SELECT COUNT(*) AS estudiantes_con_2_o_mas_carreras
FROM (
    SELECT cs.student_id
    FROM career_student cs
    GROUP BY cs.student_id
    HAVING COUNT(*) >= 2
) t;
```

The results pane displays the output of the query:

Resultados 1 ×
SELECT s.name, cs.career_id, s.email, c.career_name 123 estudiantes_con_2_o_mas_carreras

A status bar at the bottom right indicates "Enter a SQL expression to filter results (use Ctrl+Space)" and shows the number "4".