

<b>NOMBRE DEL TEMA:</b>	<b>Operadores Set, Joins y Subconsultas</b>			<b>No.</b>	<b>4</b>
<b>Asignatura:</b>	<b>TALLER DE BASE DE DATOS</b>	<b>Carrera:</b>	<b>ISIC</b>	<b>Duración de la práctica (Hrs)</b>	<b>5</b>

**Jocelin Reyes Rodriguez**

**I. Competencia(s) específica(s):**

**II. Lugar de realización de la práctica (laboratorio, taller, aula u otro): Aula**

**III. Material empleado:**

**Equipo de Cómputo**

**IV. Desarrollo de la práctica:**

## CONSULTAS Y SUBCONSULTAS EN MySQL

```

2 -- Calcular el promedio de la longitud de 'first_name' en la tabla 'actor'
3 SELECT AVG (LENGTH(first_name)) FROM actor;

```

AVG (LENGTH(first_name))
5.3050

```

5 -- Seleccionar actores con longitud de 'first_name' mayor al promedio
6 SELECT * FROM actor WHERE LENGTH(first_name) > (SELECT AVG(LENGTH(first_name)) FROM actor);
7

```

actor_id	first_name	last_name	last_update
1	PENELOPE	GUINNESS	2006-02-15 04:34:33
4	JENNIFER	DAVIS	2006-02-15 04:34:33
5	JOHNNY	LOLOBRIGIDA	2006-02-15 04:34:33
8	MATTHEW	JOHANSSON	2006-02-15 04:34:33
10	CHRISTIAN	GABLE	2006-02-15 04:34:33
14	VIVIEN	BERGEN	2006-02-15 04:34:33
20	LUCILLE	TRACY	2006-02-15 04:34:33
21	KIRSTEN	PALTROW	2006-02-15 04:34:33
23	SANDRA	KILMER	2006-02-15 04:34:33

Record 1 of 82    Ln 4, Col 1    Elapsed Time



```
8 -- Encuentra los actores que han participado en películas de la categoría  
comedy  
9 -- Hechos por separado  
10 SELECT category_id FROM category WHERE name = "Comedy";
```

Message Summary Result 1

Data Info

Cell Editor

Data Profiling

Export

category\_id

5

```
12 SELECT film_id FROM film_category WHERE category_id = 5;  
13
```

Message Summary Result 1

Data Info

Cell Editor

Data Profiling

Export

Pin

film\_id

7

28

99

119

+ - ✓ ✕

SELECT film\_id FROM film\_category WHERE category\_id = 5

Read Only

Record 1 of 58

Ln 11, Col 56

Elapsed Time: 0.346

```
14 -- Uniendo ambas QUERY  
15 SELECT film_id FROM film_category WHERE category_id = (SELECT category_id FROM  
category WHERE name = "Comedy");  
16
```

Message Summary Result 1

Data Info

Cell Editor

Data Profiling

Export

Pin

film\_id

7

28

99

119

+ - ✓ ✕

SELECT film\_id FROM film\_category WHERE category\_id = (SELECT category\_id FROM category WHERE name = "Comedy")

Read Only

Record 1 of 58

Ln 13, Col 3

Elapsed Time: 0.31

```
16 -- Para listas hacemos uso del IN y del NOT IN  
17 SELECT actor_id FROM film_actor WHERE film_id IN (7,28,99);  
18
```

Message Summary Result 1

Data Info

Cell Editor

Data Profiling

Export

Pin

actor\_id

99

133

162

170

+ - ✓ ✕

SELECT actor\_id FROM film\_actor WHERE film\_id IN (7,28,99)

Read Only

Record 1 of 9

Ln 15, Col 1

Elapsed Time: 0.31



```
19 -- Uniendo QUERY
20 SELECT actor_id FROM film_actor WHERE film_id IN (SELECT film_id FROM
    film_category WHERE category_id = (SELECT category_id FROM category WHERE name
    = "Comedy"));
21
```

Message Summary Result 1

Data Info

actor_id
99
133
162
170
185

Cell Editor Data Profiling Export Pin

-- Uniendo QUERY SELECT actor\_id FROM film\_actor WHERE film\_id IN (SELECT film\_id FROM film\_category WHERE category\_id = (SELECT category\_id FROM category WHERE name = "Comedy")); Read Only Record 1 of 286 Ln 18, Col 1 Elapsed Time: 0.3

```
22 -- Haciendo el último SELECT
23 SELECT first_name, last_name FROM actor WHERE actor_id IN (SELECT actor_id
    FROM film_actor WHERE film_id IN (SELECT film_id FROM film_category WHERE
    category_id = (SELECT category_id FROM category WHERE name = "Comedy")));
```

Message Summary Result 1

Data Info

first_name	last_name
JIM	MOSTEL
RICHARD	PENN
OPRAH	KILMER
MENA	HOPPER
MICHAEL	BOLGER

Cell Editor Data Profiling Export Pin

SELECT first\_name, last\_name FROM actor WHERE actor\_id IN (SELECT actor\_id FROM film\_actor WHERE film\_id IN (SELECT film\_id FROM film\_category WHERE category\_id = (SELECT category\_id FROM category WHERE name = "Comedy"))); Read Only Record 1 of 147 Ln 21, Col 1 Elapsed Time: 0.3

```
1 -- Encuentra los títulos de las películas que nunca han sido alquiladas
2 -- Averiguar que ID estan en la tabla
3 SELECT rental_id FROM rental WHERE rental_id ;
4
```

Message Summary Result 1 Explain 1

Data Info

rental_id
1
2
3
4
5
6

Cell Editor Data Profiling Export Pin



```
5 -- Inventario
6 SELECT film_id FROM inventory WHERE inventory_id IN ( SELECT inventory_id FROM
rental WHERE rental_id );
```

Message Summary Result 1 Explain 1

Data Info

film_id
1
1
1
1
1
1
1
1
2
2
2

SELECT film\_id FROM inventory WHERE inventory\_id IN ( SELECT inventory\_id FROM rental WHERE rental\_id ) Read Only Record 1 of 4580 Ln 5, Col 14 Elapsed Time: 0.34

```
8 -- FILM
9
10 SELECT title FROM film WHERE film_id NOT IN (SELECT film_id FROM inventory WHERE
inventory_id IN ( SELECT inventory_id FROM rental WHERE rental_id ));
```

Message Summary Result 1 Explain 1

Data Info

title
ALICE FANTASIA
APOLLO TEEN
ARGONAUTS TOWN
ARK RIDGEMONT

SELECT title FROM film WHERE film\_id NOT IN (SELECT film\_id FROM inventory WHERE inventory\_id IN ( SELECT inventory\_id FROM rental WHERE rental\_id )) Read Only Record 1 of 42 Ln 7, Col 1 Elapsed Time: 0.34

## JOINS EN MySQL

```
1 -- JOINS en MySQL
2 |-- INNER JOIN : cuando coinciden ambas tablas
3 SELECT f.title, c.name AS category_name
4 FROM film AS f
5 INNER JOIN film_category AS fc
6 ON f.film_id = fc.film_id
7 INNER JOIN category AS c
8 ON fc.category_id = c.category_id
9 WHERE c.name = "Comedy";
10
```

Message Summary Result 1

Data Info

title	category_name
AIRPLANE SIERRA	Comedy
ANTHEM LUKE	Comedy
BRINGING HYSTE	Comedy
CAPER MOTIONS	Comedy
CAT CONEHEADS	Comedy
CLOSER BANG	Comedy
CONNECTION MI	Comedy

SELECT f.title, c.name AS category\_name FROM film AS f INNER JOIN film\_category AS fc ON f.film\_id = fc.film\_id INNER JOIN category AS c ON fc.category\_id = c.category\_id WHERE c.name = "Comedy" Read Only Record 1 of 58 Ln 2, Col 3 Elapsed Time: 0.34



Objects | \* functions @sakila (mysql8-localhost) ... | peliculas @sakila (mysql8-localhost) - ... | JOIN @sakila (mysql8-localhost) - Query

Save | Query Builder | Beautify SQL | Code Snippet

mysql8-localhost | sakila | Run | Stop | Explain

```
10
11 -- Otro ejemplo de INNER JOIN
12 SELECT film.title, c.name FROM film
13 INNER JOIN film_category ON film.film_id = film_category.film_id
14 JOIN category AS c ON film_category.category_id = c.category_id;
15
```

Message | Summary | Result 1

Data | Info | Cell Editor | Data Profiling | Export | Pin

title	name
AMADEUS HOLY	Action
AMERICAN CIRC	Action
ANTITRUST TOM	Action
ARK RIDGEMONT	Action
BAREFOOT MAN	Action
BERETS AGENT	Action
BRIDE INTRIGUE	Action
BULL SHAWSHA	Action
CADDYSHACK JE	Action
CAMPUS REMEM	Action
CASUALTIES ENC	Action
CELEBRITY HORN	Action

+ - ✓ ✕

SELECT film.title, c.name FROM film INNER JOIN film\_category ON film.film\_id = film\_category.film\_id JOIN category AS c ON Read Only Record 1 of 1000 Ln 10, Col 3 Elapsed Time: 0.3

Objects | \* functions @sakila (mysql8-localhost) ... | peliculas @sakila (mysql8-localhost) - ... | JOIN @sakila (mysql8-localhost) - Query

Save | Query Builder | Beautify SQL | Code Snippet

mysql8-localhost | sakila | Run | Stop | Explain

```
16
17 -- LEFT JOIN Toma los elementos que se encuentran a la izq
18 -- Encuentra el titulo de las peliculas y la cantidad de veces que han sido
19 -- alquiladas
20 SELECT f.title, COUNT(r.rental_id) AS cantidad_rentas
21 FROM film AS f
22 LEFT JOIN inventory AS i ON f.film_id = i.film_id
23 LEFT JOIN rental AS r ON i.inventory_id= r.inventory_id
24 GROUP BY f.title;
```

Message | Summary | Result 1

Data | Info | Cell Editor | Data Profiling | Export | Pin

title	cantidad_rentas
ACADEMY DINO	23
ACE GOLDFINGER	7
ADAPTATION HC	12
AFFAIR PREJUDIC	23
AFRICAN EGG	12
AGENT TRUMAN	21
AIRPLANE SIERRA	15

+ - ✓ ✕

SELECT f.title, COUNT(r.rental\_id) AS cantidad\_rentas FROM film AS f LEFT JOIN inventory AS i ON f.film\_id = i.film\_id LEFT Read Only Record 1 of 1000 Ln 16, Col 3 Elapsed Time: 0.37



Objects | functions @sakila (mysql8-localhost) ... | peliculas @sakila (mysql8-localhost) - ... | JOIN @sakila (mysql8-localhost) - Qu...

Save | Query Builder | Beautify SQL | Code Snippet

mysql8-localhost | sakila | Run | Stop | Explain

```
25 -- Muestra los nombres de los actores y los títulos de las películas en las que han participado
26 -- FIRSTNAME, LASTNAME, TITLE
27
28 SELECT a.first_name, a.last_name, f.title
29 FROM actor AS a
30 LEFT JOIN film_actor AS fa ON a.actor_id = fa.actor_id
31 LEFT JOIN film AS f ON fa.film_id = f.film_id;
32
33 SELECT CONCAT(a.first_name, " ", a.last_name) AS full_name, f.title
34 FROM actor AS a
35 JOIN film_actor AS fa ON a.actor_id = fa.actor_id
36 JOIN film AS f ON fa.film_id = f.film_id;
```

Message Summary Result 1

Data Info | Cell Editor | Data Profiling | Export | Pin

first_name	last_name	title
PENELOPE	GUINESS	ACADEMY DINO!
PENELOPE	GUINESS	ANACONDA CON
PENELOPE	GUINESS	ANGELS LIFE
PENELOPE	GUINESS	BULWORTH COM
PENELOPE	GUINESS	CHEAPER CLYDE
PENELOPE	GUINESS	COLOR PHILADEI

SELECT a.first\_name, a.last\_name, f.title FROM actor AS a LEFT JOIN film\_actor AS fa ON a.actor\_id = fa.actor\_id LEFT JOIN film AS f ON ... Read Only | Record 1 of 5462 | Ln 27, Col 1 | Elapsed Time: 0.3

Objects | functions @sakila (mysql8-localhost) ... | peliculas @sakila (mysql8-localhost) - ... | JOIN @sakila (mysql8-localhost) - Qu...

Save | Query Builder | Beautify SQL | Code Snippet

mysql8-localhost | sakila | Run | Stop | Explain

```
25 -- Muestra los nombres de los actores y los títulos de las películas en las que han participado
26 -- FIRSTNAME, LASTNAME, TITLE
27
28 SELECT a.first_name, a.last_name, f.title
29 FROM actor AS a
30 LEFT JOIN film_actor AS fa ON a.actor_id = fa.actor_id
31 LEFT JOIN film AS f ON fa.film_id = f.film_id;
32
33 SELECT CONCAT(a.first_name, " ", a.last_name) AS full_name, f.title
34 FROM actor AS a
35 JOIN film_actor AS fa ON a.actor_id = fa.actor_id
36 JOIN film AS f ON fa.film_id = f.film_id;
```

Message Summary Result 1

Data Info | Cell Editor | Data Profiling | Export | Pin

full_name	title
PENELOPE GUINESS	ACADEMY DINO!
PENELOPE GUINESS	ANACONDA CON
PENELOPE GUINESS	ANGELS LIFE
PENELOPE GUINESS	BULWORTH COM
PENELOPE GUINESS	CHEAPER CLYDE
PENELOPE GUINESS	COLOR PHILADEI

SELECT CONCAT(a.first\_name, " ", a.last\_name) AS full\_name, f.title FROM actor AS a JOIN film\_actor AS fa ON a.actor\_id = fa.actor\_id JOIN film AS f ON fa.film\_id = f.film\_id; Read Only | Record 1 of 5462 | Ln 32, Col 1 | Elapsed Time: 0.3



## OPERACIONES SET EN MySQL

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
39 -- Operaciones SET
40 SELECT first_name FROM actor
41 UNION
42 SELECT first_name FROM customer;
43
```

The query is executed, and the results are displayed in the 'Result 1' tab. The results show a list of first names from the 'actor' and 'customer' tables, including PENELOPE, NICK, ED, JENNIFER, JOHNNY, BETTE, GRACE, MATTHEW, JOE, CHRISTIAN, ZERO, KARL, UMA, and VIVIEN.

Message Summary Result 1

Data Info

first\_name

PENELOPE

NICK

ED

JENNIFER

JOHNNY

BETTE

GRACE

MATTHEW

JOE

CHRISTIAN

ZERO

KARL

UMA

VIVIEN

SELECT first\_name FROM actor UNION SELECT first\_name FROM customer Read Only Record 1 of 647 Ln 40, Col 1 Elapsed Time: 0.37

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
44 -- Encuentra las películas que no han sido alquiladas
45 SELECT title FROM film
46 EXCEPT
47 SELECT f.title FROM film AS f
48 JOIN inventory AS i ON f.film_id = i.film_id
49 JOIN rental AS r ON i.inventory_id = r.inventory_id;
```

The query is executed, and the results are displayed in the 'Result 1' tab. The results show a list of movie titles that have not been rented, including ALICE FANTASIA, APOLLO TEEN, ARGONAUTS TOI, ARK RIDGEMONT, ARSENIC INDEPE, BOONDOCK BALI, BUTCH PANTHER, CATCH AMISTAD, CHINATOWN GL, CHOCOLATE DUC, COMMANDMEN, and CROSSING DIVOI.

Message Summary Result 1

Data Info

title

ALICE FANTASIA

APOLLO TEEN

ARGONAUTS TOI

ARK RIDGEMONT

ARSENIC INDEPE

BOONDOCK BALI

BUTCH PANTHER

CATCH AMISTAD

CHINATOWN GL

CHOCOLATE DUC

COMMANDMEN

CROSSING DIVOI

SELECT title FROM film EXCEPT SELECT f.title FROM film AS f JOIN inventory AS i ON f.film\_id = i.film\_id JOIN rental AS r ON i.inventory\_id = r.inventory\_id; Read Only Record 1 of 42 Ln 44, Col 1 Elapsed Time: 0.34



```
51 -- Con subconsultas y con JOINS
52 -- Devuelve las ciudades donde viven los clientes o empleados sin duplicados
53 SELECT DISTINCT city FROM city
54 JOIN address ON city.city_id = address.city_id
55 JOIN staff ON address.address_id = staff.address_id;
```

Message Summary Result 1

Data Info

city

Lethbridge

Woodridge

SELECT DISTINCT city FROM city JOIN address ON city.city\_id = address.city\_id JOIN staff ON address.address\_id = staff.address\_id Read Only Record 1 of 2 Ln 40, Col 1 Elapsed Time: 0.34

Objects \* functions @sakila (mysql8-localhost) ... peliculas @sakila (mysql8-localhost) - ... \* JOIN @sakila (mysql8-localhost) - Qu...

Save Query Builder Beautify SQL Code Snippet

mysql8-localhost sakila Run Stop Explain

```
51 -- Con subconsultas y con JOINS
52 -- Devuelve las ciudades donde viven los clientes o empleados sin duplicados
53 SELECT DISTINCT city FROM city
54 JOIN address ON city.city_id = address.city_id
55 JOIN staff ON address.address_id = staff.address_id;
56
57
58 SELECT city FROM city WHERE city_id IN(SELECT city_id FROM address WHERE address_id IN (SELECT
address_id FROM customer));
```

Message Summary Result 1

Data Info

city

A Coruña (La C...

Abha

Abu Dhabi

Acuña

Adana

Addis Abeba

Aden

Adoni

Ahmadnagar

SELECT city FROM city WHERE city\_id IN(SELECT city\_id FROM address WHERE address\_id IN (SELECT address\_id FROM customer)) Read Only Record 1 of 597 Ln 51, Col 17 Elapsed Time: 0.3

## V. CONCLUSIONES:

Las consultas en MySQL son una parte esencial de la gestión y manipulación de bases de datos. Al analizar el uso de las consultas, se pueden extraer diversas conclusiones sobre su funcionamiento, eficiencia y buenas prácticas.

Las consultas en MySQL son fundamentales para interactuar con bases de datos, y su correcta optimización y uso adecuado son claves para lograr aplicaciones eficientes y escalables. El dominio de buenas prácticas como el uso de índices, la selección precisa de columnas, la optimización de JOINS, el manejo de transacciones y el monitoreo del rendimiento son esenciales para mantener el rendimiento y la integridad de los datos en sistemas de bases de datos MySQL.



