

Sesion 1

```
USE classicmodels;
-- prueba
# comentario

use tienda;

show tables;

describe articulo;

# articulo

select * from empleado
where apellido_paterno = "Risom"
;

select * from empleado
where id_puesto in (100,200)

;
# ¿Cuál es el nombre de los empleados con el puesto 4?
select *
from empleado
where id_puesto = 4;

# Qué puestos tienen salarios mayor a 10000
select *
from puesto
where salario > 10000;

#¿Que articulos tienen precio mayor a $1000 y un iva mayor a 100
select *
from articulo
where precio > 1000
and iva > 100;

# ¿Qué ventas incluyen los articulos 135 o 963 y fueron hechas por los empleados 835 o
369
select *
from venta
where id_articulo in (135,963)
and id_empleado in (835,369);
```

Proyecto Sesión 1

<https://drive.google.com/file/d/1n811AqmnPjtkmocy1IO5bUfStB6ACGED/view?usp=sharing>

-- Dentro del mismo servidor de bases de datos, conéctate al esquema classicmodels.
use classicmodels;

-- Dentro de la tabla employees, obtén el apellido de todos los empleados.

```
select  
lastName  
from employees;
```

-- Dentro de la tabla employees, obtén el apellido, nombre y puesto de todos los empleados

```
select  
lastName,  
firstName,  
jobTitle  
from employees;
```

-- Dentro de la tabla employees, obtén todos los datos de cada empleado

```
select *  
from employees;
```

-- Dentro de la tabla employees, obtén el apellido, nombre y puesto de todos los empleados que tengan el puesto Sales Rep.

```
select  
lastName,  
firstName,  
jobTitle  
from employees  
where jobTitle = "Sales Rep";
```

-- Dentro de la tabla employees, obtén el apellido, nombre, puesto y código de oficina de todos los empleados que tengan el puesto Sales Rep y código de oficina 1.

```
select  
lastName,  
firstName,  
jobTitle,  
officeCode  
from employees  
where jobTitle = "Sales Rep"
```

```
and officeCode = 1;
```

```
-- Dentro de la tabla employees, obtén el apellido, nombre y código de oficina de todos los empleados que tenga código de oficina 1, 2 o 3.
```

```
select  
lastName,  
firstName,  
officeCode  
from employees  
where officeCode in (1,2,3);
```

```
-- Dentro de la tabla employees, obtén el apellido, nombre y puesto de todos los empleados que tengan un puesto distinto a Sales Rep.
```

```
select  
lastName,  
firstName,  
jobTitle  
from employees  
where jobTitle <> "Sales Rep";
```

```
-- Dentro de la tabla employees, obtén el apellido, nombre y código de oficina de todos los empleados cuyo código de oficina sea mayor a 5.
```

```
select  
lastName,  
firstName,  
officeCode  
from employees  
where officeCode > 5;
```

```
-- Dentro de la tabla employees, obtén el apellido, nombre y código de oficina de todos los empleados cuyo código de oficina sea menor o igual 4.
```

```
select  
lastName,  
firstName,  
officeCode  
from employees  
where officeCode <= 4;
```

```
--
```

```
select *  
from customers;
```

```
-- Dentro de la tabla customers, obtén el nombre, país y estado de todos los clientes cuyo país sea USA y cuyo estado sea CA.
```

```
select  
customerName,  
country,  
state  
from customers
```

```
where country = "USA"
AND state = "CA";
```

-- Dentro de la tabla customers, obtén el nombre, país, estado y límite de crédito de todos los clientes cuyo país sea USA, cuyo estado sea CA y cuyo límite de crédito sea mayor a 100000.

```
select
customerName,
country,
state,
creditLimit
from customers
where country = "USA"
AND state = "CA"
AND creditLimit > 100000;
```

-- Dentro de la tabla customers, obtén el nombre y país de todos los clientes cuyo país sea USA o France.

```
select
customerName,
country
from customers
where country in ( "USA" , "France")
;
```

-- Dentro de la tabla customers, obtén el nombre, pas y límite de crédito de todos los clientes cuyo país sea USA o France y cuyo límite de crédito sea mayor a 100000. Para este ejercicio ten cuidado con los paréntesis.

```
select
customerName,
country,
creditLimit
from customers
where country in ( "USA" , "France")
AND creditLimit > 100000;
```

-- Dentro de la tabla offices, obtén el código de la oficina, ciudad, teléfono y país de aquellas oficinas que se encuentren en USA o France.

```
select
officeCode,
city,
phone,
country
from offices;
```

-- Dentro de la tabla offices, obtén el código de la oficina, ciudad, teléfono y país de aquellas oficinas que no se encuentren en USA o France.

```
select
officeCode,
city,
phone,
country
from offices
where country <> "USA"
and country <> "France" ;
```

-- Dentro de la tabla orders, obtén el número de orden, número de cliente, estado y fecha de envío de todas las órdenes con el número 10165, 10287 o 10310.

```
select
orderNumber,
customerNumber,
'status',
shippedDate
from orders;
```

-- Dentro de la tabla customers, obtén el apellido y nombre de cada cliente y ordena los resultados por apellido de forma ascendente.

```
select
contactLastName,
contactFirstName
from customers
order by contactLastName asc;
```

-- Dentro de la tabla customers, obtén el apellido y nombre de cada cliente y ordena los resultados por apellido de forma descendente

```
select
contactLastName,
contactFirstName
from customers
order by contactLastName desc;
```

-- Dentro de la tabla customers, obtén el apellido y nombre de cada cliente y ordena los resultados por apellido de forma descendente y luego por nombre de forma ascendente

```
select
contactLastName,
contactFirstName
from customers
order by contactLastName desc, contactFirstName asc;
```

-- Dentro de la tabla customers, obtén el número de cliente, nombre de cliente y el límite de crédito de los cinco clientes con el límite de crédito más alto (top 5).

```
select  
customerNumber,  
customerName,  
creditlimit  
from customers  
order by creditlimit desc  
limit 5;
```

-- Dentro de la tabla customers, obtén el número de cliente, nombre de cliente y el límite de crédito de los cinco clientes con el límite de crédito más bajo

```
select  
customerNumber,  
customerName,  
creditlimit  
from customers  
order by creditlimit  
limit 5;
```

SESION 2

USE tienda;

```
select *  
from empleado
```

```
where nombre like 'M%';  
  
select *  
from empleado  
where nombre like '%a';  
  
select *  
from empleado  
where nombre like 'M%a';  
  
-- con condicion de largo  
select *  
from empleado  
where nombre like 'M_I_S_';  
  
select  
avg(precio) as Precio_promedio  
from articulo;  
  
select  
count(*) as conteo  
from empleado  
where nombre <> "Klara";  
  
select max(precio) from articulo;  
  
select min(precio) from articulo;
```

Reto 1

- ¿Qué artículos incluyen la palabra `Pasta` en su nombre?

```
22 •      select *  
23       from articulo  
24       where nombre like '%Pasta%';  
25  
26
```

Result Grid					
	id_articulo	nombre	precio	iva	cantidad
	91	Pasta - Cheese / Spinach Bauletti	5811.44	619.36	15
	134	Pasta - Orzo, Dry	6537.91	1113.99	906
	213	Pasta - Rotini, Colour, Dry	1830.13	373.98	309
	233	Pasta - Cannelloni, Sheets, Fresh	2316.37	605.55	307
	327	Pasta - Cappellini, Dry	6994.49	766.18	828
	361	Pasta - Penne, Rigate, Dry	2222.62	584.88	276
	426	Pasta - Cappellini, Dry	2417.66	1088.42	411
	431	Pasta - Orzo, Dry	806.33	495.92	142
	570	Pasta - Penne, Lisce, Dry	2141.06	809.94	88

articulo 8 ×

Output:

- ¿Qué artículos incluyen la palabra **Cannelloni** en su nombre?

```
select *  
from articulo  
where nombre like '%Cannelloni%';
```

```
27      -- ¿Qué artículos incluyen la palabra Cannelloni en su nombre?
28 •   select *
29     from articulo
30     where nombre like '%Cannelloni%';
31
32      --Qué nombres están separados por un guión (-) por ejemplo Puree -
33      Kiwi?
```

The screenshot shows a MySQL query being run in a database client. The query selects all columns from the 'articulo' table where the 'nombre' column contains the string 'Cannelloni'. The result grid displays one row with the following data:

	id_articulo	nombre	precio	iva	cantidad
▶	233	Pasta - Cannelloni, Sheets, Fresh	2316.37	605.55	307
*	NULL	NULL	NULL	NULL	NULL

- ¿Qué nombres están separados por un guión (-) por ejemplo Puree - Kiwi?

```
select *
from articulo
where nombre like '%-%';
```

```

32      -- ¿Qué nombres están separados por un guión (-) por ejemplo Puree - Kiwi?
33 •   select *
34     from articulo
35     where nombre like '%-%';
36

```

The screenshot shows a MySQL Workbench interface with a query editor and a results grid. The query editor contains the code above. The results grid has the following data:

id_articulo	nombre	precio	iva	cantidad
1	Chocolate - Feathers	2738.93	12.26	144
2	Pasta - Angel Hair	4391.73	959.51	503
3	Soup Campbells - Tomato Bisque	2991.35	587.59	604
4	Wine - Valpolicella Masi	2625.2	770.1	575
5	Mousse - Banana Chocolate	3701.62	893.46	248
6	Yeast Dry - Fleischman	923.18	524.08	818
7	Nantucket - Kiwi Berry Cktl.	5579.47	1012.33	527
8	Wine - Fontanafredda Barolo	2684.64	327.16	682
9	Lotus Rootlets - Canned	1996.46	324.72	636

Reto 2

-- ¿Cuál es el promedio de salario de los puestos?

```

select
    avg(salario) as Promedio_salario
from puesto;

```

```
51      -- ¿Cuál es el promedio de salario de los puestos?
52 •   select
53     avg(salario) as Promedio_salario
54   from puesto;
55
56      -- ¿Cuántos artículos incluyen la palabra Pasta en su nombre?
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

Promedio_salario
19595.051179999973

-- ¿Cuántos artículos incluyen la palabra Pasta en su nombre?

```
select count(*) as Articulos_con_pasta
```

```
from articulo
```

```
where nombre like '%Pasta%';
```

```
55
56      -- ¿Cuántos artículos incluyen la palabra Pasta en su nombre?
57 •   select count(*) as Articulos_con_pasta
58   from articulo
59   where nombre like '%Pasta%';
60
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

Articulos_con_pasta
17

-- ¿Cuál es el salario mínimo y máximo?

```
select
```

```
max(salario) as Salario_maximo,
```

```
min(salario) as Salario_minimo
```

```
from puesto;
```

The screenshot shows a MySQL query editor interface. The SQL code entered is:

```
60
61      -- ¿Cuál es el salario mínimo y máximo?
62 •  select
63      max(salario) as Salario_maximo,
64      min(salario) as Salario_minimo
65      from puesto;
```

The 'select' keyword is highlighted with a blue background. The result grid shows the following data:

	Salario_maximo	Salario_minimo
▶	29996.58	10013.44

-- ¿Cuál es la suma del salario de los últimos cinco puestos agregados?

```
select
sum(salario)
from puesto
where id_puesto >= 995;
```

```
71  
72      -- ¿Cuál es la suma del salario de los últimos cinco puestos agregados?  
73  
74      -- select sum(salario) from puesto order by  
75  
76 •   select  
77      sum(salario)  
78      from puesto  
79      where id_puesto >= 995;  
80
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
	sum(salario)			
▶	98919.69			

reto 3

```
-- ¿Cuántos registros hay por cada uno de los puestos?  
select count(*) As numero_puestos,  
nombre  
from puesto  
group by nombre;
```

```
99      -- ¿Cuántos registros hay por cada uno de los puestos?
100 •  select count(*) As numero_puestos,
101     nombre
102     from puesto
103     group by nombre;
104      -- ¿Cuánto dinero se paga en total por puesto?
105      -- ¿Cuál es el número total de ventas por vendedor?
106      -- ¿Cuál es el número total de ventas por artículo?
107
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	numero_puestos	nombre
4		Administrative Officer
10		Environmental Tech
2		Database Administrator II
2		Programmer III
4		Senior Sales Associate
15		Project Manager
11		Occupational Therapist
7		Information Systems Manager
3		Software Test Engineer III

Result 32 ×

```
-- ¿Cuánto dinero se paga en total por puesto?
select sum(salario) As Total_salario,
nombre
from puesto
group by nombre;
```

```
105 •      select sum(salario) As Total_salario,  
106      nombre  
107      from puesto  
108      group by nombre;  
109      -- ¿Cuál es el número total de ventas por vendedor?  
110      -- ¿Cuál es el número total de ventas por artículo?  
111  
112
```

The screenshot shows a MySQL query editor interface. At the top, there is a code editor window containing the provided SQL script. Below it is a results grid titled "Result Grid". The grid has two columns: "Total_salario" and "nombre". The data rows are as follows:

	Total_salario	nombre
▶	179310.180000000002	Analog Circuit Design manager
	156846.26	Junior Executive
	136630.69	Director of Sales
	157528.98	Staff Scientist
	92315.22	Desktop Support Technician
	70107.77	Budget/Accounting Analyst III
	78947.08	Accounting Assistant III
	35658.78	Programmer Analyst II
	296384.04	Nurse Practitioner

```
select count(id_venta) as Ventas,  
id_empleado  
from venta group by id_empleado
```

```
109      -- ¿Cuál es el número total de ventas por vendedor?  
110 •   select count(id_venta) as Ventas,  
111     id_empleado  
112   from venta group by id_empleado  
113  
114      -- Resultado del número total de ventas por vendedor
```

<

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Ventas	id_empleado
	1	75
	1	77
	2	83
	2	85
	3	86
	1	88
	1	89
	2	90
	1	91

Result 36 ×

```
select count(id_venta) as Ventas,  
       id_articulo  
  from venta group by id_articulo;
```

```
114      -- ¿Cuál es el número total de ventas por artículo?  
115 •   select count(id_venta) as Ventas,  
116     id_articulo  
117   from venta group by id_articulo;  
118
```

The screenshot shows a MySQL query results grid. The grid has two columns: 'Ventas' and 'id_articulo'. The data is as follows:

Ventas	id_articulo
1	2
1	3
2	4
1	8
1	10
1	11
1	12
2	13
1	15

Result 38 ×

Output

Reto 4

-- ¿Cuál es el nombre de los empleados cuyo sueldo es menor a \$10,000?

```
select nombre  
from empleado  
where id_puesto in  
(select id_puesto  
from puesto  
where salario<10000);
```

```
--  
142      -- ¿Cuál es el nombre de los empleados cuyo sueldo es menor a $10,000?  
143  •   select nombre  
144    from empleado  
145    where id_puesto in  
146      (select id_puesto  
147       from puesto  
148      where salario<10000);  
149  
150      -- ¿Cuál es la cantidad mínima y máxima de ventas de cada empleado?  
151      -- ¿Cuál es el nombre del puesto de cada empleado?  
<  
Result Grid | Filter Rows:  | Export:  | Wrap Cell Content:   


| nombre |
|--------|
|--------|


```

```
-- ¿Cuál es la cantidad mínima y máxima de ventas de cada empleado?  
select id_empleado,  
max(total_ventas),  
min(total_ventas)  
from  
(select clave, id_empleado, count(*) as total_ventas  
from venta  
group by clave, id_empleado  
) as A  
group by id_empleado;
```

```

149
150      -- ¿Cuál es la cantidad mínima y máxima de ventas de cada
151 •      select id_empleado,
152          max(total_ventas),
153          min(total_ventas)
154      from
155          (select clave, id_empleado, count(*) as total_ventas
156          from venta
157          group by clave, id_empleado
158      ) as A
159          group by id_empleado;
160      -- ¿Cuál es el nombre del puesto de cada empleado?
161

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	id_empleado	max(total_ventas)	min(total_ventas)
▶	569	1	1
	413	2	1
	765	1	1
	119	1	1
	90	1	1
	835	1	1
	369	1	1
	555	1	1
	187	1	1

selected Result 2 ×

Output:

Action Output

#	Time	Action	Message
1	21-10-10 10:40:40	select id_empleado, max(total_ventas), min(total_ventas) from (select clave, id_empleado, count(*) as total_ventas from venta group by clave, id_empleado) as A group by id_empleado;	Error Code: 1046 N

-- ¿Cuál es el nombre del puesto de cada empleado?

```

select
nombre,
apellido_paterno,
(select
nombre
from puesto p
where p.id_puesto = e.id_puesto) as puesto
from empleado as e;

```

```

161
162 •   select
163     nombre,
164     apellido_paterno,
165     (select
166         nombre
167         from puesto p
168         where p.id_puesto = e.id_puesto) as puesto
169     from empleado as e;
170
171
172
173

```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell

	nombre	apellido_paterno	puesto
▶	Enrichetta	Bodechon	Product Engineer
	Morey	Bowskill	Budget/Accounting Analyst IV
	Jeannette	Potes	Occupational Therapist
	Cassey	Womersley	Financial Advisor
	Gnni	Risom	Physical Therapy Assistant
	Lisle	Carlsson	Marketing Assistant
	Andre	Theurer	Tax Accountant
	Land	Locksley	Product Engineer
	Nikki	Fayerbrother	Sales Associate

Result 4 ×

Output

Action Output ▾

PROYECTO SESION 2

-- Dentro de la tabla employees, obtén el número de empleado, apellido y nombre de todos los empleados cuyo nombre empiece con a.

```

select
employeeNumber,
lastname,
firstname
from employees
where firstname like 'A%';

```

Result Grid | Filter Rows:

	employeeNumber	lastname	firstname
▶	1143	Bow	Anthony
*	1611	Fixter	Andy
*	NULL	NULL	NULL

-- Dentro de la tabla employees, obtén el número de empleado, apellido y nombre de todos los empleados cuyo nombre termina con on.

```
select
employeeNumber,
lastname,
firstname
from employees
where firstname like '%on';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

	employeeNumber	lastname	firstname
*	HULL	HULL	HULL

-- Dentro de la tabla employees, obtén el número de empleado, apellido y nombre de todos los empleados cuyo nombre incluye la cadena on.

```
select
employeeNumber,
lastname,
firstname
from employees
where firstname like '%on%';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

	employeeNumber	lastname	firstname
▶	1143	Bow	Anthony
*	1286	Tseng	Foon Yue
*	HULL	HULL	HULL

-- Dentro de la tabla employees, obtén el número de empleado, apellido y nombre de todos los empleados cuyos nombres tienen tres letras e inician con T y finalizan con m.

```
select
employeeNumber,
lastname,
firstname
from employees
where firstname like '%T_m%';
```

	employeeNumber	lastname	firstname
▶	1619	King	Tom
*	NULL	NULL	NULL

-- Dentro de la tabla employees, obtén el número de empleado, apellido y nombre de todos los empleados cuyo nombre no inicia con B.

```
select
employeeNumber,
lastname,
firstname
from employees
where firstname not like 'B%';
```

	employeeNumber	lastname	firstname
▶	1002	Murphy	Diane
	1056	Patterson	Mary
	1076	Firrelli	Jeff
	1088	Patterson	William
	1102	Bondur	Gerard
	1143	Bow	Anthony
	1165	Jennings	Leslie
	1166	Thompson	Leslie

-- Dentro de la tabla products, obtén el código de producto y nombre de los productos cuyo código incluye la cadena _20.

```
select
productCode,
productName
from
products
where productCode like'____20%'
```

	productCode	productName
▶	S10_2016	1996 Moto Guzzi 1100i
	S24_2000	1960 BSA Gold Star DBD34
	S24_2011	18th century schooner
	S24_2022	1938 Cadillac V-16 Presidential Limousine
*	NULL	NULL

-- Dentro de la tabla orderdetails, obtén el total de cada orden.

```
select  
sum(priceEach) as Total_orden,  
orderNumber  
from orderdetails  
group by orderNumber;
```

	Total_orden	orderNumber
	651.79	10163
	758.85	10164
	1794.94	10165
	285.66	10166
	1271.62	10167
	1472.50	10168
	1130.70	10169
	410.22	10170

Result 33 ×

-- Dentro de la tabla orders obtén el número de órdenes por año.

```
select  
count(*) as ordenes,  
year(orderDate) as 'year'  
from orders  
group by year(orderDate)
```

	ordenes	year
▶	111	2003
	151	2004
	64	2005

-- Obtén el apellido y nombre de los empleados cuya oficina está ubicada en USA.

```
select  
lastName,  
firstName  
from employees  
where officeCode in  
(select officeCode  
from offices  
where country = 'USA');
```

Result Grid | Filter Rows: []

	lastName	firstName
▶	Murphy	Diane
	Patterson	Mary
	Firrelli	Jeff
	Bow	Anthony
	Jennings	Leslie
	Thompson	Leslie
	Firrelli	Julie
...	Patterson	Steve

employees 41 ×

Output

Action Output ▾

-- Obten el número de cliente, número de cheque y cantidad del cliente que ha realizado el pago más alto.

```
select
customerNumber,
checkNumber,
amount
from payments
where amount in
(select
max(amount)
from payments);
```

	customerNumber	checkNumber	amount
▶	141	JE105477	120166.58
*	NULL	NULL	NULL

-- Obten el número de cliente, número de cheque y cantidad de aquellos clientes cuyo pago es más alto que el promedio.

```
select
customerNumber,
checkNumber,
amount
from payments
where amount >
(select
avg(amount)
from payments);
```

Result Grid | Filter Rows: ||

	customerNumber	checkNumber	amount
▶	112	HQ55022	32641.98
	112	ND748579	33347.88
	114	GG31455	45864.03
	114	MA765515	82261.22
	114	NR27552	44894.74
	119	LN373447	47924.19
	119	NG94694	49523.67
	121	DB889831	50218.95

payments 60 ×

-- Obten el nombre de aquellos clientes que no han hecho ninguna orden.

```
select
contactLastName,
contactFirstName
from customers
where customerNumber
not in
(select customerNumber
from orders
)
```

Result Grid | Filter Rows: || Export: | Wrap Cell Content: |

	contactLastName	contactFirstName
	Müller	Rita
	Feuer	Alexander
	Ottlieb	Sven
	Anton	Carmen
	Moos	Hanna
	Semenov	Alexander
	Altagar,G M	Raanan

customers 65 ×

Output:

-- Obten el máximo, mínimo y promedio del número de productos en las órdenes de venta.

```
select
productCode,
max(quantityOrdered) as Max_productos,
min(quantityOrdered) as Min_productos,
avg(quantityOrdered) as Promedio_productos
from orderdetails
group by productCode
```

productCode	Max_productos	Min_productos	Promedio_productos
S18_2870	55	20	34.2000
S18_2949	50	10	37.0714
S18_2957	53	20	35.1786
S18_3029	49	21	34.5000
S18_3136	49	20	32.3929
S18_3140	49	20	32.7037
S18_3232	60	20	34.1132
S18_3259	50	21	34.0000

Result 68 ×

Output

```

select
D.Estado,
Sum(D.ordenes)
from
(select
C.state as Estado,
(select count(*) as ordenes
from orders as B
where B.customerNumber = C.customerNumber) as ordenes
from customers as C
) as D
group by
D.Estado;

```

Estado	Sum(D.ordenes)
NULL	180
NV	3
Victoria	8
CA	45
NY	18
PA	9
CT	8
MA	23

Result 75 ×

Output

SESION 3

¿Cuál es el nombre de los empleados que realizaron cada venta?
Reto 1

```

select *

from venta v
left join
empleado e
on v.id_empleado = e.id_empleado;

```

The screenshot shows a MySQL command-line interface. At the top, there is a code editor window containing the following SQL query:

```

24 • select *
25
26 from venta v
27 left join
28 empleado e
29 on v.id_empleado = e.id_empleado;
30 -- ¿Cuál es el nombre de los artículos que se han vendido?
31 -- ¿Cuál es el total de cada venta?

```

Below the code editor is a results grid titled "Result Grid". The grid has the following columns:

	id_venta	id_articulo	id_empleado	dave	fecha	id_empleado	id_puesto	nombre	apellido_paterno	apellido_materno	rfc
▶	1	296	569	0228-3661	2019-11-15 00:00:00	569	871	Arlana	Fanstone	Brothers	TARG400578M92
	2	311	413	52125-277	2019-04-23 00:00:00	413	433	Julianna	Gecke	Gille	WVTI176350P68
	3	960	765	0049-0032	2019-08-31 00:00:00	765	427	Maryellen	Parkisson	Itzkovici	HYWO934761B24
	4	536	119	52125-277	2020-02-03 15:04:57	119	79	Cad	Sambedge	Goulborn	EAGM298123A66
	5	693	90	13107-062	2019-10-30 00:00:00	90	258	Michæline	Stoll	Hamer	NLKQ487540S39
	6	135	835	0049-0032	2020-02-03 15:05:27	835	965	Courbay	Harston	Bowland	SVMX753070E78
	7	963	369	47335-894	2019-06-08 00:00:00	369	339	Giuditta	Chicchetto	Helbeck	NMFW494417N54
	8	234	413	52125-277	2020-02-03 15:04:57	413	433	Julianna	Gecke	Gille	WVTI176350P68
	9	721	555	0049-0032	2020-02-03 15:05:27	555	458	Rosalinde	Livock	Paike	NSEY678377W35

The results grid has 9 rows of data. The last row is indicated by an ellipsis (...).

-- ¿Cuál es el nombre de los artículos que se han vendido?

```

SELECT
v.id_venta,
v.id_articulo,
a.nombre
FROM venta v
left join articulo a
on v.id_articulo = a.id_articulo;

```

```
31 •   SELECT
32     v.id_venta,
33     v.id_articulo,
34     a.nombre
35   FROM venta v
36   left join articulo a
37   on v.id_articulo = a.id_articulo;
38   -- ¿Cuál es el total de cada venta?
39
```

	id_venta	id_articulo	nombre
▶	919	2	Pasta - Angel Hair
	885	3	Soup Campbells - Tomato Bisque
	473	4	Wine - Valpolicella Masi
	504	4	Wine - Valpolicella Masi
	387	8	Wine - Fontanafredda Barolo
	629	10	Wine - Vovray Sec Domaine Huet
	845	11	Cake - Pancake
	288	12	Chocolate Liqueur - Godet White
	144	13	Appetizer - Southwestern

-- ¿Cuál es el total de cada venta?

```
select  
v.clave as venta,  
sum(a.precio)  
FROM venta v  
left join articulo a  
on v.id_articulo = a.id_articulo  
group by v.clave  
;
```

```
41
42 • select
43   v.clave as venta,
44   sum(a.precio)
45   FROM venta v
46   left join articulo a
47   on v.id_articulo = a.id_articulo
48   group by v.clave
49 
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	venta	sum(a.precio)
▶	0228-3661	3714.37
	52125-277	340582.6499999999
	0049-0032	321524.6099999999
	13107-062	249071.45999999993
	47335-894	223650.32000000007
	51655-951	190821.20999999996
	52380-1865	162361.13
	69128-001	174310.76
	52343-028	150970.75

Result 16 ×

Output :

Reto 2

```
create view SAUempleado as
(select
e.id_puesto,
concat (e.nombre, ' ', e.apellido_paterno, ' ', e.apellido_materno) as empleado,
p.nombre as puesto
from empleado e left join
puesto p on e.id_puesto = p.id_puesto);

select *
from SAUempleado;
```

```

67 •   create view SAUempleado as
68     (select
69       e.id_puesto,
70       concat (e.nombre, ' ', e.apellido_paterno, ' ', e.apellido_materno) as empleado,
71       p.nombre as puesto
72     from empleado e left join
73       puesto p on e.id_puesto = p.id_puesto);
74
75 •   select *

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	id_puesto	empleado	puesto
▶	235	Enrichetta Bodechon Ivkovic	Product Engineer
	659	Morey Bowskill Metham	Budget/Accounting Analyst IV
	745	Jeannette Potes Heisler	Occupational Therapist
	139	Cassey Womersley Chapell	Financial Advisor
	668	Gnni Risom Kalinowsky	Physical Therapy Assistant

SAUempleado 22 ×

-- Saber qué artículos ha vendido cada empleado.

```

create view SAUarticulo_ventas as
(select
v.id_venta,
v.id_articulo,
a.nombre as articulo,
concat (e.nombre, ' ', e.apellido_paterno, ' ', e.apellido_materno) as empleado
from
venta v
left join
articulo a
ON a.id_articulo = v.id_articulo
left join empleado e
ON v.id_empleado = e.id_empleado);

select
count(*) as ventas,
articulo,
empleado
from
SAUarticulo_ventas
group by
articulo,
empleado
order by empleado;

```

```

9 •   create view SAUarticulo_ventas as
0   (select
1     v.id_venta,
2     v.id_articulo,
3     a.nombre as articulo,
4     concat (e.nombre, ' ', e.apellido_paterno, ' ', e.apellido_materno) as empleado
5   from
6     venta v
7     left join
8       articulo a
9       ON a.id_articulo = v.id_articulo
0     left join empleado e
1     ON v.id_empleado = e.id_empleado);
2

```

Sult Grid | Filter Rows: | Export: | Wrap Cell Content: | [Full 28](#) | [X](#)

ventas	articulo	empleado
1	Foil - 4oz Custard Cup	Eric Coram Semple
1	Sunflower Seed Raw	Eric Coram Semple
1	Butter Sweet	Erinn Littlecote Woodcroft
1	Chocolate Liqueur - Godet W...	Ermanno Reye Lowle
1	Cilantro / Coriander - Fresh	Ermin Davidow Catonnet
1	Longos - Chicken Curried	Ermin Davidow Catonnet
1	Wine - Red, Gallo, Merlot	Ermin Davidow Catonnet

-- Saber qué puesto ha tenido más ventas

```

create view SAUventas as
(select
v.id_venta,
v.clave,
e.id_empleado,
p.nombre as puesto
from
venta v
left join empleado e
ON v.id_empleado = e.id_empleado
left join puesto p
on e.id_puesto = p.id_puesto) ;

```

```

Select
puesto,
count(*) as ventas
from
SAUventas
group by puesto
order by ventas desc
limit 1;

```

```
119      select  
120      puesto,  
121      count(*) as ventas  
122      from  
123      SAUventas  
124      group by puesto  
125      order by ventas desc  
126      limit 1;  
127  
128  
129  
130
```

	puesto	ventas
▶	Physical Therapy Assistant	23

PROYECTO SESION 3

```
-- Para estas consultas usa RIGHT JOIN  
-- Obten el código de producto, nombre de producto y descripción de todos los productos.  
select  
productCode,  
productName,  
productDescription  
from  
products;
```

```

6 •    select
7      productCode,
8      productName,
9      productDescription
10     from
11     products;
12
13
14     -- Obten el número de orden, estado y costo total de cada orden.
15
16
17     -- Obten el número de orden, fecha de orden, línea de orden, nombre del producto, ca

```

< [Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:]

	productCode	productName	productDescription
▶	S10_1678	1969 Harley Davidson Ultimate Chopper	This replica features working kickstand, front su...
	S10_1949	1952 Alpine Renault 1300	Turnable front wheels; steering function; detail...
	S10_2016	1996 Moto Guzzi 1100i	Official Moto Guzzi logos and insignias, saddle b...
	S10_4698	2003 Harley-Davidson Eagle Drag Bike	Model features, official Harley Davidson logos a...
	S10_4757	1972 Alfa Romeo GTA	Features include: Turnable front wheels; steeri...
...			

-- Obten el número de orden, estado y costo total de cada orden.

```

select
o.orderNumber,
o.status,
odd.productCode
from orders o
right join
orderdetails odd
on o.orderNumber = odd.orderNumber

```

```

13
14    -- Obten el número de orden, estado y costo total de cada orden.
15 • select
16     o.orderNumber,
17     o.status,
18     odd.productCode
19   from orders o
20   right join
21     orderdetails odd
22   on o.orderNumber = odd.orderNumber
23
24

```

Result Grid | Filter Rows: Export: Wrap Cell Content: Fetch rows:

	orderNumber	status	productCode
▶	10107	Shipped	S10_1678
	10121	Shipped	S10_1678
	10134	Shipped	S10_1678
	10145	Shipped	S10_1678
	10159	Shipped	S10_1678

Result 15 ×

-- Obten el número de orden, fecha de orden, línea de orden, nombre del producto, cantidad ordenada y precio de cada pieza que muestre los detalles de cada orden.

Select

```

O.orderNumber,
O.orderdate,
odd.orderLineNumber,
p.productName,
odd.quantityOrdered,
odd.priceEach
from
products p
right join
orderdetails odd
on odd.productCode = p.productCode
right join orders O
on odd.orderNumber = O.orderNumber;

```

```

24 -- Obtén el número de orden, fecha de orden, línea de orden, nombre del producto, cantidad ordenada y precio de cada pieza que muestre los detalles de cada orden.
25 • Select
26     O.orderNumber,
27     O.orderdate,
28     odd.orderLineNumber,
29     p.productName,
30     odd.quantityOrdered,
31     odd.priceEach
32     from
33     products p
34     right join
35     orderdetails odd
36     on odd.productCode = p.productCode
37     right join orders O
38     on odd.orderNumber = O.orderNumber;

```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content: [] | Fetch rows: []

orderNumber	orderdate	orderLineNumber	productName	quantityOrdered	priceEach
10100	2003-01-06	3	1917 Grand Touring Sedan	30	136.00
10100	2003-01-06	2	1911 Ford Town Car	50	55.09
10100	2003-01-06	4	1932 Alfa Romeo 8C2300 Spider Sport	22	75.46
10100	2003-01-06	1	1936 Mercedes Benz 500k Roadster	49	35.29
10101	2003-01-09	4	1932 Model A Ford J-Coupe	25	108.06

Result 28 × Read Only

-- Obtén el número de orden, nombre del producto, el precio sugerido de fábrica (msrp) y precio de cada pieza.

```

select
O.orderNumber,
p.productname,
p.MSRP,
O.priceEach
from
products p
right join
orderdetails O
on O.productCode = p.productCode

```

```

40
41 -- Obtén el número de orden, nombre del producto, el precio sugerido de fábrica (msrp) y precio de cada pieza.
42 • select
43     O.orderNumber,
44     p.productname,
45     p.MSRP,
46     O.priceEach
47     from
48     products p
49     right join
50     orderdetails O
51     on O.productCode = p.productCode
52
53
54 -- Para estas consultas usa LEFT JOIN

```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content: [] | Fetch rows: []

orderNumber	productname	MSRP	priceEach
10100	1917 Grand Touring Sedan	170.00	136.00
10100	1911 Ford Town Car	60.54	55.09
10100	1932 Alfa Romeo 8C2300 Spider Sport	92.03	75.46
10100	1936 Mercedes Benz 500k Roadster	41.03	35.29
10101	1932 Model A Ford J-Coupe	127.13	108.06

Result 33 × Output ::::::::::::

-- Obtén el número de cliente, nombre de cliente, número de orden y estado de cada cliente.

```

select

```

```

c.customerNumber,
concat (c.contactFirstName, ' ', contactLastName) as nombre_cliente,
o.orderNumber,
o.status
from
customers c
left join orders o
on c.customerNumber = o.customerNumber;

```

```

53    -- Para estas consultas usa LEFT JOIN
54    -- Obtén el número de cliente, nombre de cliente, número de orden y estado de cada cliente.
55 • select
56    c.customerNumber,
57    concat (c.contactFirstName, ' ', contactLastName) as nombre_cliente,
58    o.orderNumber,
59    o.status
60    from
61    customers c
62    left join orders o
63    on c.customerNumber = o.customerNumber;
64
65    -- Obtén los clientes que no tienen una orden asociada.
66
67    -- Obtén el apellido de empleado, nombre de empleado, nombre de cliente, número de cheque y total, es decir
68

```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	customerNumber	nombre_cliente	orderNumber	status
▶	103	Carine Schmitt	10123	Shipped
	103	Carine Schmitt	10298	Shipped
	103	Carine Schmitt	10345	Shipped
	112	Jean King	10124	Shipped
	112	Jean King	10278	Shipped

Result 38 ×

Output

```

-- Obtén los clientes que no tienen una orden asociada.
select
c.customerNumber,
concat (c.contactFirstName, ' ', contactLastName) as nombre_cliente,
o.orderNumber,
o.status
from
customers c
left join orders o
on c.customerNumber = o.customerNumber
where o.orderNumber is null;

```

```

64
65 -- Obtén los clientes que no tienen una orden asociada.
66 • select
67 c.customerNumber,
68 concat (c.contactFirstName, ' ', contactLastName) as nombre_cliente,
69 o.orderNumber,
70 o.status
71 from
72 customers c
73 left join orders o
74 on c.customerNumber = o.customerNumber
75 where o.orderNumber is null;

```

<

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content: []

	customerNumber	nombre_cliente	orderNumber	status
▶	125	Zbyszek Piestrzeniewicz	NULL	NULL
	168	Keith Franco	NULL	NULL
	169	Isabel de Castro	NULL	NULL
	206	Brydey Walker	NULL	NULL

-- Obtén el apellido de empleado, nombre de empleado, nombre de cliente, número de cheque y total, es decir, los clientes asociados a cada empleado.

```

select
e.lastName,
e.firstName,
c.customerNumber,
p.checkNumber,
p.amount
from customers c
left join
employees e
on c.salesRepEmployeeNumber = e.employeeNumber
left join
payments p
on c.customerNumber = p.customerNumber;

```

```

76      -- Obtén el apellido de empleado, nombre de empleado, nombre de cliente, número de cheque y total, es decir, los clientes asociados a cada empleado.
77
78 •   select
79     e.lastName,
80     e.firstname,
81     c.customerNumber,
82     p.checkNumber,
83     p.amount
84   from customers c
85   left join
86     employees e
87   on c.salesRepEmployeeNumber = e.employeeNumber
88   left join
89     payments p
90   on c.customerNumber = p.customerNumber;

```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content: []

lastName	firstname	customerNumber	checkNumber	amount
Hernandez	Gerard	141	IN446258	65071...
Hernandez	Gerard	141	JE105477	12016...
Hernandez	Gerard	141	JN355280	49539...
Hernandez	Gerard	141	JN722010	40206...
Hernandez	Gerard	141	KT52578	63843...

Result 57 ×

-- Para estas consultas usa RIGHT JOIN

-- Repite los ejercicios 5 a 7 usando RIGHT JOIN.

-- 5r. Obtén el número de cliente, nombre de cliente, número de orden y estado de cada cliente.

```

select
c.customerNumber,
concat (c.contactFirstName,' ',contactLastName) as nombre_cliente,
o.orderNumber,
o.status
from
orders o
right join customers c
on c.customerNumber = o.customerNumber;

```

```

92
93 -- Para estas consultas usa RIGHT JOIN
94 -- Repite los ejercicios 5 a 7 usando RIGHT JOIN.
95 -- 5r.Obtén el número de cliente, nombre de cliente, número de orden y estado de cada cliente.
96 • select
97 c.customerNumber,
98 concat(c.contactFirstName, ' ', contactLastName) as nombre_cliente,
99 o.orderNumber,
100 o.status
101 from
102 orders o
103 right join customers c
104 on c.customerNumber = o.customerNumber;
105
106 -- Escoge 3 consultas de los ejercicios anteriores, crea una vista y escribe una consulta para cada una.
<

```

Result Grid | Filter Rows: [] | Export: [] | Wrap Cell Content: []

	customerNumber	nombre_cliente	orderNumber	status
▶	103	Carine Schmitt	10123	Shipped
	103	Carine Schmitt	10298	Shipped
	103	Carine Schmitt	10345	Shipped
	112	Jean King	10124	Shipped
	112	Jean King	10278	Shipped

Result 58 x

Output

Action Output

Time Action

Message

-- 7.r.Obtén el apellido de empleado, nombre de empleado, nombre de cliente, número de cheque y total, es decir, los clientes asociados a cada empleado.

```

select
e.lastName,
e.firstName,
c.customerNumber,
p.checkNumber,
p.amount
from
payments p
right join
(employees e
right join
customers c
on c.salesRepEmployeeNumber = e.employeeNumber
)
on c.customerNumber = p.customerNumber;

```

```

107 •   select
108     e.lastName,
109     e.firstname,
110     c.customerNumber,
111     p.checkNumber,
112     p.amount
113   from
114     payments p
115   right join
116     (employees e
117      right join
118      customers c
119      on c.salesRepEmployeeNumber = e.employeeNumber
120    )
121   on c.customerNumber = p.customerNumber;
122

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	lastName	firstname	customerNumber	checkNumber	amount
▶	NULL	NULL	125	NULL	NULL
	NULL	NULL	169	NULL	NULL
	NULL	NULL	206	NULL	NULL
	NULL	NULL	223	NULL	NULL
	NULL	NULL	237	NULL	NULL

Result 59 ×

Output :::::

-- Escoge 3 consultas de los ejercicios anteriores, crea una vista y escribe una consulta para cada una.

```

create view SAUClientes_sinorden as (
select
c.customerNumber,
concat(c.contactFirstName, ' ', contactLastName) as nombre_cliente,
o.orderNumber,
o.status
from
customers c
left join orders o
on c.customerNumber = o.customerNumber
where o.orderNumber is null);

select count(*) as clientes_sin_orden from
SAUClientes_sinorden

```

```
136 •      select count(*) as clientes_sin_orden from  
137      SAUClientes_sinorden  
138
```

The screenshot shows a MySQL Workbench interface with a result grid. The grid has one column labeled 'count(*)' and one row containing the value '24'. The grid includes standard database navigation buttons (back, forward, first, last) and export options.

count(*)
24

```
create view SAUClientes_empleado as (  
select  
e.lastName,  
e.firstname,  
c.customerNumber,  
p.checkNumber,  
p.amount  
from customers c  
left join  
employees e  
on c.salesRepEmployeeNumber = e.employeeNumber  
left join  
payments p  
on c.customerNumber = p.customerNumber);
```

```
select  
concat(firstname,",",lastname) as Empleado,  
customerNumber as Cliente,  
count(*) as ventas  
from  
SAUClientes_empleado  
group by Empleado, Cliente
```

```
155 •   select
156     concat(firstname,' ',lastname) as Empleado,
157     customerNumber as Cliente,
158     count(*) as ventas
159   from
160     SAUClientes_empleado
161   group by Empleado, Cliente
162
163
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result 63 ×

	Empleado	Cliente	ventas
1	MartinGe...	298	2
2	MartinGe...	344	2
3	MartinGe...	376	1
4	MartinGe...	458	3
5	MartinGe...	484	2

```
create view SAUProducto_precio as (
select
O.orderNumber,
p.productname,
p.MSRP,
O.priceEach
from
products p
right join
orderdetails O
on O.productCode = p.productCode);

select
orderNumber,
sum(MSRP) as Total_MSRP,
max(MSRP) as max_MSRP
from SAUProducto_precio
group by orderNumber
```

```
176 •   select
177     orderNumber,
178     sum(MSRP) as Total_MSRP,
179     max(MSRP) as max_MSRP
180   from SAUProducto_precio
181   group by orderNumber
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	orderNumber	Total_MSRP	max_MSRP
▶	10100	363.60	170.00
	10101	373.87	168.75
	10102	156.65	102.74
	10103	1702.52	214.30
	10104	1371.02	169.34

Result 65 |

Output ::::::::::::

Action Output

MONGO

```
mongodb://root:BEdu_0583@ec2-35-166-232-75.us-west-
2.compute.amazonaws.com:27017/?authSource=admin&connectTimeoutMS=600000&s
ocketTimeoutMS=6000000
```

SESION 4

RETO 1

- Fecha, nombre y texto de cada comentario.

FILTER

PROJECT `{date: 1, name: 1, text: 1}`

SORT

COLLATION

```
_id: ObjectId("5a9427648b0beebeb69579cc")
name: "Andrea Le"
text: "Rem officiis eaque repellendus amet eos doloribus. Porro dolor volupta...""
date: 2012-03-26T23:20:16.000+00:00
```

```
_id: ObjectId("5a9427648b0beebeb69579cf")
name: "Greg Powell"
text: "Tenetur dolorum molestiae ea. Eligendi praesentium unde quod porro. Co...""
date: 1987-02-10T00:29:36.000+00:00
```

```
_id: ObjectId("5a9427648b0beebeb69579d0")
name: "Talisa Maegyr"
text: "Rem itaque ad sit rem voluptatibus. Ad fugiat maxime illum optio iure ..."
date: 1998-08-22T11:45:03.000+00:00
```

```
_id: ObjectId("5a9427648b0beebeb69579d3")
name: "Cameron Duran"
text: "Quasi dicta culpa asperiores quaerat perferendis neque. Est animi pari...""
date: 1983-04-27T20:39:15.000+00:00
```

```
_id: ObjectId("5a9427648b0beebeb69579d5")
name: "Petyr Baelish"
text: "Ovo deserunt insam insum. Tenetur eos nemo nam sint praesentium minus ..."
```

- Título, elenco y año de cada película.

The screenshot shows a MongoDB query interface with the following filters applied:

- PROJECT**: {title: 1, cast: 1, year: 1}
- SORT**
- COLLATION**

The interface includes standard navigation buttons: **VIEW**, **grid**, **{}**, and **list**.

Five movie documents are listed:

- _id: ObjectId("573a1390f29313caabcd4135")**
▶ **cast: Array**
 title: "Blacksmith Scene"
 year: 1893
- _id: ObjectId("573a1390f29313caabcd42e8")**
▶ **cast: Array**
 title: "The Great Train Robbery"
 year: 1903
- _id: ObjectId("573a1390f29313caabcd4323")**
▶ **cast: Array**
 title: "The Land Beyond the Sunset"
 year: 1912
- _id: ObjectId("573a1390f29313caabcd446f")**
▶ **cast: Array**
 title: "A Corner in Wheat"
 year: 1909
- _id: ObjectId("573a1390f29313caabcd4803")**
▶ **cast: Array**
 title: "Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving ..."

- Nombre y contraseña de cada usuario

The screenshot shows a MongoDB query interface with the following configuration:

- Filter:** {name: 1, password: 1, _id:0}
- Sort:** (not explicitly shown)
- Collation:** (not explicitly shown)

The interface includes a toolbar with icons for upload, view mode, list mode, object mode, and grid mode.

The results list contains the following documents:

- name: "Ned Stark"
password: "\$2b\$12\$UREFwsRUoyF0CRqGNK0Lz00HM/jLhgUCNNIJ9RJAqMUQ74cr1J1Vu"
- name: "Robert Baratheon"
password: "\$2b\$12\$yGqxLG9LZpXA2xVDhuPnSOZd.VURVkz7wgOLY3pn00s7u2S1Z032y"
- name: "Jaime Lannister"
password: "\$2b\$12\$6vz7wiw0.EI5Rilvq1zUc./9480gb1uPtXcahDxIadgyC3PS8XCUK"
- name: "Catelyn Stark"
password: "\$2b\$12\$fiaTH5Sh1zKNFX2i/FTErEWGjxoJxvmV7XL.qlfqCr8CwOxK.mZWS"
- name: "Cersei Lannister"
password: "\$2b\$12\$FExjgr7CLhNCa.oUsB9seub8mqCHzkJCFZ8heMc8CeIK0ZfeTKP8m"
- name: "Daenerys Targaryen"
password: "\$2b\$12\$NzpbWHdMytemLtTfFKduHenr2NZ.rvxIKuYM4AWLTFaUShxbJ.G3q"
- name: "Jorah Mormont"
password: (redacted)

RETO 2

- ¿Qué comentarios ha hecho Greg Powell?

FILTER {name: 'Greg Powell'} **OPTIONS** **FIND**
PROJECT {name: 1, text: 1}
SORT **MAXTITEMS** 5000
COLLATION **SKIP** 0 **LIMIT** 0

VIEW Displaying documents 1 - 20 of 290

```

_id: ObjectId("5a9427648b0beebeb69579cf")
name: "Greg Powell"
text: "Tenetur dolorum molestiae ea. Eligendi praesentium unde quod porro. Co..."
```

```

_id: ObjectId("5a9427648b0beebeb6957afe")
name: "Greg Powell"
text: "Rem nostrum nobis saepe eaque itaque nemo. Fugit dignissimos nisi sapi..."
```

```

_id: ObjectId("5a9427648b0beebeb6957b56")
name: "Greg Powell"
text: "Officia atque ullam esse doloribus laborum. Maiores dicta ratione rem ..."
```

```

_id: ObjectId("5a9427648b0beebeb6957dd2")
name: "Greg Powell"
text: "Quo fugiat iure dolor nam dignissimos architecto eum. Amet molestias s..."
```

```

_id: ObjectId("5a9427648b0beebeb6958081")
name: "Greg Powell"
text: "Facilis ut eius aliquid eaque dolore. Ut ea eos nesciunt ex sed quia. ..."
```

-
- ¿Qué comentarios han hecho Greg Powell o Mercedes Tyler?

sample_mflix.comments

Documents Aggregations Schema Explain Plan Indexes

FILTER `{$or: [{name: 'Greg Powell'}, {name: 'Mercedes Tyler'}]}` **FIND**

PROJECT `{name: 1, text: 1}`

SORT

COLLATION

MAXTITEMS 5000 **SKIP** 0 **LIMIT** 0

VIEW

Displaying documents 1 - 20 of 576 < >

```
_id: ObjectId("5a9427648b0beebeb69579cf")
name: "Greg Powell"
text: "Tenetur dolorum molestiae ea. Eligendi praesentium unde quod porro. Co..."
```

```
_id: ObjectId("5a9427648b0beebeb69579e7")
name: "Mercedes Tyler"
text: "Eius veritatis vero facilis quaerat fuga temporibus. Praesentium exped..."
```

```
_id: ObjectId("5a9427648b0beebeb6957a78")
name: "Mercedes Tyler"
text: "Voluptate odio minima pariatur recusandae. Architecto illum dicta repu..."
```

```
_id: ObjectId("5a9427648b0beebeb6957afe")
name: "Greg Powell"
text: "Rem nostrum nobis saepe eaque itaque nemo. Fugit dignissimos nisi sapi..."
```

```
_id: ObjectId("5a9427648b0beebeb6957b56")
name: "Greg Powell"
text: "Officia atque ullam esse doloribus laborum. Maiores dicta ratione rem ..."
```



- ¿Cuál es el máximo número de comentarios en una película?

The screenshot shows a MongoDB query interface with the following configuration:

- PROJECT**: `{title: 1, num_mflix_comments: 1}`
- SORT**: None specified.
- COLLATION**: None specified.
- OPTIONS**: `MAXTIMEMS 5000`, `SKIP 0`, `LIMIT 0`.
- FIND** button is visible.

The results section displays five documents:

- `_id: ObjectId("573a1390f29313caabcd4135")`
`num_mflix_comments: 1`
`title: "Blacksmith Scene"`
- `_id: ObjectId("573a1390f29313caabcd42e8")`
`title: "The Great Train Robbery"`
- `_id: ObjectId("573a1390f29313caabcd4323")`
`num_mflix_comments: 2`
`title: "The Land Beyond the Sunset"`
- `_id: ObjectId("573a1390f29313caabcd446f")`
`num_mflix_comments: 1`
`title: "A Corner in Wheat"`
- `_id: ObjectId("573a1390f29313caabcd4803")`
`num_mflix_comments: 1`
`title: "Winsor McCay, the Famous Cartoonist of the N.Y. Herald and His Moving ..."`

Below the results, there is a bullet point list:

-
- ¿Cuál es título de las cinco películas más comentadas?

sample_mflix.comments Documents

sample_mflix.movies Documents

sample_mflix.movies Documents

sample_mflix.movies

DOCUMENTS 23.5k TOTAL SIZE 35.9MB AVG. SIZE 1.6KB INDEXES 2 TOTAL SIZE 13.1MB AVG. SIZE 6.6MB

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER PROJECT {title: 1, num_mflix_comments: 1} SORT {num_mflix_comments: -1} MAXTLEMENTS 5000 COLLATION SKIP 0 LIMIT 5

VIEW DISPLAYING DOCUMENTS 1 - 5 OF 5 REFRESH

_id:ObjectId("573a1399f29313caabcee886")
title:"The Mask"
num_mflix_comments:456

_id:ObjectId("573a1399f29313caabcee578")
title:"Dumb & Dumber"
num_mflix_comments:450

_id:ObjectId("573a13bfff29313caabd6001f")
title:"The Unborn"
num_mflix_comments:447

_id:ObjectId("573a13a5f29313caabbd159a9")
title:"About a Boy"
num_mflix_comments:441

_id:ObjectId("573a13a7f29313caabbd1aa55")
title:"8 Mile"
num_mflix_comments:441

PROYECTO 3

- Obtén los datos de contacto de cada compañía.

```
{
  project: {
    name: 1,
    email_address: 1,
    phone_number: 1
  }
}
```

- Obtén la fuente de cada tweet.

```
{
}
```

```
project: {
```

```
    text: 1,
```

```
    source: 1
```

```
}
```

```
}
```

- Obtén el nombre de todas las compañías fundadas en octubre.

```
{
}
```

```
filter: {
```

```
    founded_month: 10
```

```
},
```

```
project: {
```

```
    name: 1,
```

```
    founded_month: 1
```

```
}
```

```
}
```

The screenshot shows the MongoDB Compass interface with the 'Documents' tab selected. A search query is displayed in the top-left corner:

```
FILTER {founded_month: 10}
PROJECT {name: 1, founded_month: 1}
```

Below the query, there are buttons for 'OPTIONS', 'FIND', 'RESET', and '...'. On the right, there are controls for 'MAXTIMEMS' (set to 5000), 'SKIP' (0), and 'LIMIT' (0). The status bar at the bottom indicates "Displaying documents 1 - 20 of 301". To the right of the main interface, a sidebar displays the raw query and its results:

```
Mon Jul 13 2020 20:59:59 GMT-0500 (ho...)
FILTER
{
  founded_month: 10
}

PROJECT
{
  name: 1,
  founded_month: 1
}
```

- Obtén el nombre de todas las compañías fundadas en 2008.
- {
- filter: {
- founded_year: 2008
- },
- project: {
- name: 1,
- founded_year: 1
- }
- }

The screenshot shows the MongoDB Compass interface with the 'Documents' tab selected. A search query is displayed in the top-left corner:

```
FILTER {founded_year: 2008}
PROJECT {name: 1, founded_year: 1}
```

Below the query, there are buttons for 'OPTIONS', 'FIND', 'RESET', and '...'. On the right, there are controls for 'MAXTIMEMS' (set to 5000), 'SKIP' (0), and 'LIMIT' (0). The status bar at the bottom indicates "Displaying documents 1 - 20 of 1224". To the right of the main interface, a sidebar displays the raw query and its results:

```
Mon Jul 13 2020 21:01:42 GMT-0500 (ho...)
FILTER
{
  founded_year: 2008
}

PROJECT
{
  name: 1,
  founded_year: 1
}
```

- Obtén todos los *post* del autor machine.

```
{
```

```
filter: {
```

```
  author: 'machine'
```

```
},
```

```
project: {
```

```
  post: 1,
```

```
  author: 1
```

```
}
```

}

The screenshot shows the MongoDB Compass interface. The top bar displays "sample_training.posts". Below it, the "Documents" tab is selected. The search bar contains the query: {author: 'machine'}. The results pane shows one document with the following fields:

```
_id: ObjectId("50ab0f0bbcf1bfe2536dc3f9")
author: "machine"
```

- Obtén todos los tweets provenientes de la web.

{

```
filter: {
  source: 'web'
},
project: {
  text: 1,
  source: 1
}
}
```

The screenshot shows the MongoDB Compass interface. The top bar displays "sample_training.posts". Below it, the "Documents" tab is selected. The search bar contains the query: {source: 'web'}. The results pane shows one document with the following fields:

```
_id: ObjectId("5c8eccb0caa187d17ca623f5")
text: "eu preciso de terminar de fazer a minha tabela, está muito foda **"
```

- Obtén todas las compañías fundadas en octubre del 2008.

{

```
filter: {
  founded_month: 10,
  founded_year: 2008
}
```

```

    },
    project: {
        name: 1,
        founded_month: 1,
        founded_year: 1
    }
}

```

The screenshot shows the MongoDB Compass interface. At the top, it displays the database and collection: `sample_training.companies`. Below this, the document count is **9.5k**, total size is **34.8MB**, and average size is **3.7KB**. There is also information about indexes: **INDEXES 1**, **TOTAL SIZE 100.0KB**, and **AVG. SIZE 100.0KB**.

The main area shows a search query being run:

```

FILTER: {founded_month: 10, founded_year: 2008}
PROJECT: {name: 1, founded_month: 1, founded_year: 1}
SORT: 
COLLATION: 
MAXTIMEMS: 5000
SKIP: 0
LIMIT: 0

```

Below the query, it says "Displaying documents 1 - 20 of 63". A result row is shown:

```

_id: ObjectId("52cdef7c4bab8bd6752985ca")
name: "tunesBag"
founded_year: 2008
founded_month: 10

```

On the right side, there is a sidebar titled "Past Queries" with two entries:

- Mon Jul 13 2020 21:08:12 GMT-0500 (ho...
FILTER: {founded_month: 10, founded_year: 2008}
- Mon Jul 13 2020 21:01:42 GMT-0500 (ho...
PROJECT: {name: 1, founded_month: 1, founded_year: 1}

- Obtén todas las compañías con más de 50 empleados.

```

{
    filter: {
        number_of_employees: {
            $gt: 50
        }
    },
    project: {
        name: 1,
        number_of_employees: 1
    }
}

```

The screenshot shows the MongoDB Compass interface. On the left, the database is set to "sample_training.companies" and the collection is "Companies". The "Documents" tab is selected. In the top right, it says "DOCUMENTS 9.5k" and "INDEXES 1". Below that, there are filters: "FILTER" ({"number_of_employees": {"\$gt": 50}}), "PROJECT" ({"name": 1, "number_of_employees": 1}), "SORT" ({"MAXITEMS": 5000}), and "COLLATION". At the bottom, it says "Displaying documents 1 - 20 of 793". The results pane shows one document:

```
_id: ObjectId("52cddef7c4bab8bd675297d8e")
name: "Facebook"
number_of_employees: 5299
```

On the right, there is a sidebar with "Past Queries", "RECENT", and "FAVORITES". It shows the query: "FILTER { number_of_employees: { \$gt: 50 } }" and "PROJECT { name: 1, number_of_employees: 1 }". The timestamp is "Mon Jul 13 2020 21:12:01 GMT-0500 (ho...)".

- Obtén las historias con número de comentarios entre 10 y 30.

```
{
```

```
filter: {
```

```
$and: [
```

```
{
```

```
comments: {
```

```
$gte: 10
```

```
}
```

```
},
```

```
{
```

```
comments: {
```

```
$lte: 30
```

```
}
```

```
}
```

```
]
```

```
},
```

```
project: {
```

```
title: 1,
```

```
comments: 1
```

```
}
```

}

_id	title	comments
_id: ObjectId("4ba267dc238d3ba3ca000006")	"11 Amazing Treehouses from Around the World"	15
_id: ObjectId("4ba267dc238d3ba3ca00000b")	"NASA - The Wizard Nebula "	14
_id: ObjectId("4ba267dc238d3ba3ca00000c")	"WISE Captures a Cosmic Rose"	12
_id: ObjectId("4ba267dc238d3ba3ca000012")	"Astronomers Find a New Type of Star"	10

- Obtén la empresa con el menor número de empleados.

{

filter: {

\$and: [

{

number_of_employees: {

\$ne: null

}

},

{

number_of_employees: {

\$ne: 0

}

}

]

```

},
project: {
  name: 1,
  number_of_employees: 1
},
sort: {
  number_of_employees: 1
},
limit: 1
}

```

The screenshot shows the MongoDB Compass interface with the following details:

- COLLECTIONS:** sample_training.companies
- Documents:** 9.5k DOCUMENTS, 34.8MB TOTAL SIZE, 3.7KB AVG. SIZE, 1 INDEXES, 100.0KB TOTAL SIZE, 100.0KB AVG. SIZE.
- FILTER:** {\$and: [{number_of_employees: {\$ne: null}}, {number_of_employees: {\$ne: 0}}]}
- PROJECT:** {name: 1, number_of_employees: 1}
- SORT:** {number_of_employees: 1}
- COLLATION:**
- OPTIONS:** MAXTIMEMS 5000, SKIP 0, LIMIT 1
- DISPLAY:** Displaying documents 1 - 1 of 1
- RESULT:** One document is shown:


```

_id: ObjectId("52cdef7c4bab8bd675297e68")
name: "Felote"
number_of_employees: 1
      
```

- Obtén la empresa con el mayor número de empleados.

```

{
filter: {
  $and: [
    {
      number_of_employees: {

```

```
$ne: null  
}  
,  
{  
number_of_employees: {  
$ne: 0  
}  
}  
]  
,  
project: {  
name: 1,  
number_of_employees: 1  
},  
sort: {  
number_of_employees: -1  
},  
limit: 1  
}
```

The screenshot shows the MongoDB Compass interface with three tabs open: 'sample_mflix.comments', 'sample_mflix.movies', and 'sample_training.companies'. The 'sample_training.companies' tab is active, displaying a single document. The document has the following fields:

```

{
  "_id": ObjectId("52cdef7c4bab8bd67529858a"),
  "name": "IBM",
  "number_of_employees": 388000
}

```

The interface includes a sidebar with a list of collections and a right-hand panel for viewing and modifying queries.

- Obtén la historia más comentada.

```
{
  filter: {
    $and: [
      {
        comments: {
          $ne: null
        }
      },
      {
        comments: {
          $ne: 0
        }
      }
    ]
  },
}
```

```

project: {

    title: 1,

    comments: 1

},

sort: {

    comments: -1

},

limit: 1

}

```

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with tabs for 'Documents', 'Aggregations', 'Schema', 'Explain Plan', 'Indexes', and 'Validation'. The 'Aggregations' tab is selected. In the main area, a query results window displays a single document:

```

_id:ObjectId("4ba27ea0238d3ba3ca002251")
title:"Republican Brown wins Massachusetts Senate seat!"
comments:1864

```

The query definition on the right is as follows:

```

{
  $and: [
    {
      comments: {
        $ne: null
      }
    },
    {
      comments: {
        $ne: 0
      }
    }
  ]
}

PROJECT
{
  title: 1,
  comments: 1
}

SORT
{
  comments: -1
}

LIMIT
1

```

- Obtén la historia menos comentada.

```

{

filter: {

$and: [

{

comments: {

```

```
$ne: null  
}  
,  
{  
comments: {  
$ne: 0  
}  
}  
]  
,  
project: {  
title: 1,  
comments: 1  
},  
sort: {  
comments: 1  
},  
limit: 1  
}
```

SESION 4

- Propiedades que no permitan fiestas.

- Propiedades que admitan mascotas.

The screenshot shows a MongoDB query interface with a green header bar. Below it is a toolbar with a 'FILTER' button containing the query: '\$or: [{house_rules: /pets/i}, {house_rules: {\$not: /no pets/i}}]'. The main area displays a JSON document for a property:

```

_id: "10006546"
listing_url: "https://www.airbnb.com/rooms/10006546"
name: "Ribeira Charming Duplex"
summary: "Fantastic duplex apartment with three bedrooms, located in the histori..."
space: "Privileged views of the Douro River and Ribeira square, our apartment ..."
description: "Fantastic duplex apartment with three bedrooms, located in the histori..."
neighborhood_overview: "In the neighborhood of the river, you can find several restaura...
notes: "Lose yourself in the narrow streets and staircases zone, have lunch in..."
transit: "Transport: • Metro station and S. Bento railway 5min; • Bus stop a 50 ..."
access: "We are always available to help guests. The house is fully available t..."
interaction: "Cot - 10 € / night Dog - € 7,5 / night"
house_rules: "Make the house your home..."
property_type: "House"
room_type: "Entire home/apt"
bed_type: "Real Bed"
minimum_nights: "2"
maximum_nights: "30"
cancellation_policy: "moderate"
last_scraped: 2019-02-16T05:00:00.000+00:00
calendar_last_scraped: 2019-02-16T05:00:00.000+00:00
first_review: 2016-01-03T05:00:00.000+00:00
last_review: 2019-01-20T05:00:00.000+00:00
accommodates: 8
bedrooms: 3
beds: 5

```

At the bottom, there is a button labeled 'SHOW 14 MORE FIELDS' with a downward arrow.

- Propiedades que no permitan fumadores.

FILTER {house_rules: /no smoking/i} **FIND** **RESET** ...

ADD DATA **VIEW**

Displaying documents 1 - 20 of 639 < > C REFRESH

```

_id: "1003530"
listing_url: "https://www.airbnb.com/rooms/1003530"
name: "New York City - Upper West Side Apt"
summary: ""
space: "Murphy bed, optional second bedroom available. Wifi available, Hulu, N..."
description: "Murphy bed, optional second bedroom available. Wifi available, Hulu, N..."
neighborhood_overview: "Great neighborhood - many terrific restaurants, bakeries, bagelries. W..."
notes: "My cat, Samantha, are in and out during the summer. The apt is layed ..."
transit: "Conveniently located near 1, 2, 3, B & C subway lines. Also buses on C..."
access: "New York City!"
interaction: ""
house_rules: "No smoking is permitted in the apartment. All towels that are used sho..."
property_type: "Apartment"
room_type: "Private room"
bed_type: "Real Bed"
minimum_nights: "12"
maximum_nights: "360"
cancellation_policy: "strict_14_with_grace_period"
last_scraped: 2019-03-07T05:00:00.000+00:00
calendar_last_scraped: 2019-03-07T05:00:00.000+00:00
first_review: 2013-04-29T04:00:00.000+00:00
last_review: 2018-08-12T04:00:00.000+00:00
accommodates: 2
bedrooms: 1
beds: 1

```

SHOW 14 MORE FIELDS

```

_id: "10083468"
listing_url: "https://www.airbnb.com/rooms/10083468"
name: "Be Happy in Porto"
summary: "Be Happy Apartment is an amazing space. Renovated and comfortable apar..."
space: "Be Happy Apartment is housed in a typical Porto building, where the ap..."

```

- Propiedades que no permitan fiestas ni fumadores.

FILTER {\$_and: [{house_rules: /no parties/i}, {house_rules:/no smoking/i}]} **FIND** **RESET** ...

ADD DATA **VIEW**

Displaying documents 1 - 20 of 124 < > C REFRESH

```

_id: "10392282"
listing_url: "https://www.airbnb.com/rooms/10392282"
name: "Banyan Bungalow"
summary: "The place to be on the north shore is where you can be steps from the ..."
space: "Big, open space with lots of natural light. The cottage is clean and ..."
description: "The place to be on the north shore is where you can be steps from the ..."
neighborhood_overview: "This desirable neighborhood is comprised of other vacation rentals, lo..."
notes: ""
transit: "While you might think it nice to arrive and relax there will be many s..."
access: "Private driveway to access the property, parking on site."
interaction: "While we live on the property and will try to greet you at your arriva..."
house_rules: "No smoking, no pets, no parties. You are welcome to have guests, but ..."
property_type: "Bungalow"
room_type: "Entire home/apt"
bed_type: "Real Bed"
minimum_nights: "2"
maximum_nights: "300"
cancellation_policy: "flexible"
last_scraped: 2019-03-06T05:00:00.000+00:00
calendar_last_scraped: 2019-03-06T05:00:00.000+00:00
first_review: 2016-01-29T05:00:00.000+00:00
last_review: 2019-02-19T05:00:00.000+00:00
accommodates: 2
bedrooms: 0
beds: 1

```

SHOW 14 MORE FIELDS

```

_id: "10423504"
listing_url: "https://www.airbnb.com/rooms/10423504"
name: "Bondi Beach Dreaming 3-Bed House"
summary: "This peaceful house in North Bondi is 200m to the beach and a minute's ...

```

```

{ price: { $lte: 100 }, "address.country": "Spain",
"review_scores.review_scores_rating": { $gte: 50 }, amenities: { $in: [ "Internet",
"Wifi" ] }, amenities: { $in: ["Elevator"] } }

```

Usando la colección `sample_airbnb.listingsAndReviews`, agrega un filtro que permita obtener todas las publicaciones que tengan 50 o más comentarios, que la valoración sea mayor o igual a 80, que cuenten con conexión a Internet vía cable y estén ubicada en Brazil.

```
{number_of_reviews:{$gte: 50}, "review_scores.review_scores_rating":{$gte: 80},  
amenities: {$in:[/Internet/]}, "address.country": "Brazil"}
```

The screenshot shows a MongoDB query interface with the following details:

- Filter:** `{number_of_reviews:{$gte: 50}, "review_scores.review_scores_rating":{$gte: 80}, amenities: {$in:[/Internet/]}, "address.country": "Brazil"}`
- Options:** Options button is visible.
- Find:** Find button is highlighted in green.
- Reset:** Reset button is visible.
- Displaying documents:** 1 - 20 of 35
- Buttons:** ADD DATA, VIEW, and Refresh.
- Document Preview:** A single document is expanded, showing fields like _id, listing_url, name, summary, space, description, neighborhood_overview, notes, transit, access, interaction, house_rules, property_type, room_type, bed_type, minimum_nights, maximum_nights, cancellation_policy, last_scraped, calendar_last_scraped, first_review, last_review, accommodates, bedrooms, beds, number_of_reviews, bathrooms, amenities, price, security_deposit, cleaning_fee, extra_people, guests_included, images, and host.

```
_id: "10038496"  
listing_url: "https://www.airbnb.com/rooms/10038496"  
name: "Copacabana Apartment Posto 6"  
summary: "The Apartment has a living room, toilet, bedroom (suite) and American ..."  
space: "The apartment has a living room, wash room, suite and an American kitc..."  
description: "The Apartment has a living room, toilet, bedroom (suite) and American ..."  
neighborhood_overview: "Copacabana in the South zone is the district that offers the greatest ..."  
notes: ""  
transit: "On the street there is plenty of transport and the subway station is n..."  
access: "todo o espaço."  
interaction: "Contact telephone numbers if needed: Valeria ((PHONE NUMBER HIDDEN) (U...)"  
house_rules: "Entreguem o imóvel conforme receberam e respeitem as regras do condomí..."  
property_type: "Apartment"  
room_type: "Entire home/apt"  
bed_type: "Real Bed"  
minimum_nights: "3"  
maximum_nights: "75"  
cancellation_policy: "strict_14_with_grace_period"  
last_scraped: 2019-02-11T05:00:00.000+00:00  
calendar_last_scraped: 2019-02-11T05:00:00.000+00:00  
first_review: 2016-01-18T05:00:00.000+00:00  
last_review: 2019-01-28T05:00:00.000+00:00  
accommodates: 4  
bedrooms: 1  
beds: 3  
number_of_reviews: 70  
bathrooms: 2.0  
> amenities: Array  
price: 119.00  
security_deposit: 600.00  
cleaning fee: 150.00  
extra_people: 40.00  
guests_included: 3  
> images: Object  
> host: Object
```

Usando la colección `sample_airbnb.listingsAndReviews`, mediante el uso de agregaciones, encontrar el número de publicaciones que tienen conexión a Internet, sea desde Wifi o desde cable (Ethernet)

The screenshot shows the MongoDB Aggregation Pipeline interface. At the top, it displays the collection `sample_airbnb.listingsAndReviews`, with metrics: DOCUMENTS 5.6k, 90.0MB, 16.6KB, and INDEXES 4, 476.0KB. Below the header are tabs: Documents, Aggregations (which is selected), Schema, Explain Plan, Indexes, and Validation. A green button labeled "SAVE" is visible. On the right, there's a toggle switch for "SAMPLE" and a preview window showing the results of the pipeline.

\$match Stage:

```

1 /**
2 * query: The query in MQL.
3 */
4 {
5   amenities: {$in: ["Wifi", "Ethernet"]}
6 }

```

Output after \$match stage (Sample of 20 documents):

```

_id: "10006546"
listing_url: "https://www.airbnb.com/rooms/10006546"
name: "Ribeira Charming Duplex"
summary: "Fantastic duplex apartment with three bedroom located in the histori..."
space: "Privileged views of the Douro River and Ribeira square, our apartment ..."
description: "Fantastic duplex apartment with three bed located in the histori..."

_id: "10009999"
listing_url: "https://www.airbnb.com/rooms/10009999"
name: "Horto flat with small garden"
summary: "One bedroom + sofa-bed in quiet and bucolic neighbourhood right next t..."
space: "Lovely one bedroom + sofa-bed in the living room perfect for two but ..."
description: "One bedroom + sofa-bed in quiet and bucolic neighbourhood right next t..."

```

\$count Stage:

```

1 /**
2 * Provide the field name for the count.
3 */
4 "_id" | 5

```

Output after \$count stage (Sample of 1 document):

```

_id: 5303

```

ADD STAGE

PROYECTO 4

Para este proyecto deberás practicar en el uso de agregaciones, pues serán usadas durante la siguiente sesión.

La base de datos y colección que debes usar es `sample_airbnb.listingsAndReviews`.

El proyecto consiste en obtener todas las publicaciones que tengan 50 o más comentarios, que la valoración sea mayor o igual a 80, que cuenten con conexión a Internet vía cable y estén ubicadas en Brazil.

[{

 \$match: {

 number_of_reviews: {

 \$gte: 50

 }

}

```
}, {  
    $match: {  
        "review_scores.review_scores_rating": {  
            $gte: 80  
        }  
    }  
}, {  
    $match: {  
        amenities: {  
            $in: [/Internet/]  
        }  
    }  
}, {  
    $match: {  
        "address.country": "Brazil"  
    }  
}, {  
    $count: '_id'  
}]
```

The screenshot shows the MongoDB Aggregation Pipeline interface. At the top, it displays "sample_airbnb.listingsAndReviews" with document statistics: DOCUMENTS 5.6k, 90.0MB, 16.6KB and INDEXES 4, 476.0KB, 119.0K. Below this are tabs for "Documents", "Aggregations" (which is selected), "Schema", "Explain Plan", "Indexes", and "Validation". There are also "SAVE" and "SAMPLE MODE" buttons.

The main area shows two stages:

- \$match**: A query stage with the following MQL code:


```
1 //**
2 * query: The query in MQL.
3 */
4 {
5   "address.country": "Brazil"
6 }
```
- \$count**: An aggregation stage with the following MQL code:


```
1 //**
2 * Provide the field name for the count.
3 */
4 '_id'
```

Below each stage is a preview of the output documents. The first stage's output is a sample of 20 documents, and the second stage's output is a sample of 1 document.

SESION 5

Reto 1

Con base en el ejemplo 1, modifica el agrupamiento para que muestre el costo promedio por habitación por país de las propiedades de tipo casa.

[{

```
$match: {
  property_type: "House",
  bedrooms: {
    $gte: 1
  }
}, {
  $addFields: {
    costo_recamara: {
      $divide: ['$price', '$bedrooms']
    }
}
```

```

        }
    }
}, {
    $group: {
        _id: "$address.country",
        recamaras: {
            $sum: 1
        },
        total: {
            $sum: "$costo_recamara"
        }
    }
},
{
    $addFields: {
        pais: "$_id",
        costo_pais: {
            $divide: ['$total', '$recamaras']
        }
    }
}

```

The screenshot shows the MongoDB aggregation pipeline interface. At the top, there are buttons for 'dropdown', '\$project' (which is selected), a toggle switch, and a '+' button. To the right, it says 'Output after \$project stage (Sample of 9 documents)'. Below this, the pipeline stage is displayed:

```

1 * /**
2  *  * specifications: The fields to
3  *  * include or exclude.
4  *  */
5 * {
6     _id:0,
7     pais: 1,
8     costo_pais:1
9 }

```

The output sample shows a document with the following fields:

```

pais: "Canada"
costo_pais: 114.9886759581881533101045296167247

```

Reto 2

```
[  
    $lookup: {  
        from: 'users',  
        localField: 'email',  
        foreignField: 'email',  
        as: 'usuario'  
    }, {  
        $addFields: {  
            usuario_objeto: {  
                $arrayElemAt: ["$usuario", 0]  
            }  
        }  
    }, {  
        $addFields: {  
            usuario_password: "$usuario_objeto.password"  
        }  
    }, {  
        $project: {  
            _id: 0,  
            name: 1,  
            email: 1,  
            usuario_password: 1  
        }  
    }]
```

sample_mflix.comments

DOCUMENTS	50.3k	TOTAL SIZE	13.6MB	AVG. SIZE	284B	INDEXES	1	TOTAL SIZE	480.0KB	AVG. SIZE	480.0KB
COLLATION password						SAMPLE MODE AUTO PREVIEW					
<pre> 1 /* Let's optional variables to use in the pipeline */ 2 /* 3 * newField: The new field name. 4 * expression: The new field expression. 5 */ 6 { 7 "from": "users", 8 "localField": "email", 9 "foreignField": "email", 10 "as": "usuario" 11 } 12 13 14 </pre>											
Output after \$addFields stage (Sample of 20 documents)											
<pre> _id: ObjectId("5a942764b80b") name: "Andrea Le" email: "andrea_le@fakegmail.com" movie_id: ObjectId("573a1390f29313caabcf418c") text: "Rem officiis eaque repellendus amet eos doloribus. Porro dolor volupta.." date: 2012-03-26T23:20:16.000+00:00 ▶ usuario:Array ▶ usuario_objeto:Object </pre>											
<pre> _id: ObjectId("5a942764b80b") name: "Greg Powell" email: "greg_powell@fakegma movie_id: ObjectId("573a139 text: "Tenetur dolorum mole praesentium unde quod date: 1987-02-10T00:29:36.0 ▶ usuario:Array ▶ usuario_objeto:Object </pre>											
Output after \$addFields stage (Sample of 20 documents)											
<pre> name: "Andrea Le" email: "andrea_le@fakegmail.com" movie_id: ObjectId("573a1390f29313caabcf418c") text: "Rem officiis eaque repellendus amet eos doloribus. Porro dolor volupta.." date: 2012-03-26T23:20:16.000+00:00 ▶ usuario:Array ▶ usuario_objeto:Object usuario_password: "\$2b\$12\$J587HwUL2y0P1E6kYrcbKOKx22.wsKEdLtS0F734/vKdhuduLM8v" </pre>											
<pre> _id: ObjectId("5a942764b80b") name: "Greg Powell" email: "greg_powell@fakegma movie_id: ObjectId("573a139 text: "Tenetur dolorum mole praesentium unde quod date: 1987-02-10T00:29:36.0 ▶ usuario:Array ▶ usuario_objeto:Object usuario_password: "\$2b\$12\$J587HwUL2y0P1E6kYrcbKOKx22.wsKEdLtS0F734/vKdhuduLM8v" </pre>											

Reto 3

FILTER

VIEW

OPTIONS FIND RESET

Displaying documents 1 - 20 of N/A

REFRESH

name: "Andrea Le" email: "andrea_le@fakegmail.com" password: "\$2b\$12\$J587HwUL2y0P1E6kYrcbKOKx22.wsKEdLtS0F734/vKdhuduLM8v"
name: "Greg Powell" email: "greg_powell@fakegma password: "\$2b\$12\$XpveUB6kIIU3zG5aABw260itIB7cDBb5UNJA24hDF4XXyNjN/np76"
name: "Talisa Maegyr" email: "oona_chapling@gameofthron.es" password: "\$2b\$12\$B30krkTVNDDg1oxClpcwOJvnKICUsLcrTxNFZIeoqbHIQURsNrXS"
name: "Cameron Duran" email: "cameron.duran@fakegmail.com" password: "\$2b\$12\$50wZj63ATGmhVh2rgdjv.w0d9TV0Jb9Xk/Anms0fxVsVgHf5hvvK"
name: "Petyr Baelish" email: "sidan_gillen@gameofthron.es" password: "\$2b\$12\$qHl.YvmiekyYY7p7phpK30icbRCdkN7EsWYAnG/o9YnfHC0lhkmbl"
name: "Taylor Hill" email: "taylor.hill@fakegmail.com" password: "\$2b\$12\$NR03TpZti62ZN2rSlxyOuurfMAUEQ4oIufraahcsSqGp4eKF4G1"

The screenshot shows the MongoDB Compass interface. On the left, there's a sidebar with a 'FAVORITE' section, 'HOST' (ec2-54-213-176-37.us-west-2.compute.amazonaws.com), 'CLUSTER' (Standalone), and 'EDITION' (MongoDB 4.2.6 Community). Below these are lists of databases ('JosueDeltaCruz', 'ROMELALEJANDROHRNNNDZ', 'SalvadorVizcaino', 'admin', 'config', 'local', 'miPrimeraBD') and collections ('sample_airbnb' containing '4.ROAR_2', '4AVHandybnb', '4SAU_COSTO_PAIS', '4SCG_airbnb', '4SCZ_airbnb', '4YCM_RET003', 'MMS_airbnb'). The main area is titled 'sample_airbnb.listingsAndReviews' and shows documents with fields like 'pais' and 'costo_pais'. A 'FILTER' button is at the top, and a search bar with 'FIND' and 'RESET' buttons is below it. The status bar at the bottom says 'Displaying documents 1 - 9 of 9'.

PROYECTO 5

[{

```
$unwind: {
```

```
    path: "$genres",
```

```
}
```

```
}, {
```

```
$unwind: {
```

```
    path: "$countries"
```

```
}
```

```
}, {
```

```
$group: {
```

```
    _id: {
```

```
        age: "$genres",
```

```

    pais: "$countries"
},
Peliculas: {
    $sum: 1
}
}
}]

```

The screenshot shows the MongoDB aggregation pipeline interface. On the left, the pipeline stages are defined:

```

1 /*
2  * _id: The id of the group.
3  * fieldN: The first field name.
4 */
5 {
6   _id: {age: "$genres", pais: "$countries"},
7   Peliculas: {
8     $sum: 1
9   }
10 }

```

On the right, the "Output after \$group stage" is shown as a sample of 20 documents. Two examples are visible:

- Object 1: `_id: Object {age: "Family", pais: "Czech Republic"} Peliculas: 9`
- Object 2: `_id: Object {age: "Music", pais: "Argentina"} Peliculas: 3`

SESION 6

Configure Import Settings

Detected file format: csv 

Encoding: utf-8

Columns:

<input checked="" type="checkbox"/> Source Column	Field Type
<input checked="" type="checkbox"/> UserID	int
<input checked="" type="checkbox"/> Gender	text
<input checked="" type="checkbox"/> Age	int
<input checked="" type="checkbox"/> Occupation	int
<input checked="" type="checkbox"/> Zip-code	text

UserID	Gender	Age	Occupation	Zip-code
1	F	1	10	48067
2	M	56	16	70072
3	M	25	15	55117
4	M	45	7	02460
5	M	25	20	55455

< Back

Next >

Cancel

Configure Import Settings

Detected file format: csv 

Encoding: 

Columns:

<input checked="" type="checkbox"/> Source Column	Field Type
<input checked="" type="checkbox"/> UserID	<input type="text" value="int"/> 
<input checked="" type="checkbox"/> MovieID	<input type="text" value="int"/> 
<input checked="" type="checkbox"/> Rating	<input type="text" value="int"/> 
<input checked="" type="checkbox"/> Timestamp	<input type="text" value="int"/> 

UserID	MovieID	Rating	Timestamp
1	1193	5	978300760
1	661	3	978302109
1	914	3	978301968
1	3408	4	978300275
1	2355	5	978824291

 Back  Next 

PROYECTO 6

 _id: ObjectId("5f18f71c1525944a648cb54e")
id: 4000
titulo: "Avengers: Endgame (2019)"
genres: "Fantasy|Sci-Fi"

 _id: ObjectId("5f18f71c1525944a648cb54f")
id: 4001
titulo: "Glass (2019)"
genres: "Drama|Fantasy"

The screenshot shows a MongoDB query interface with the following settings:

- Filter:** {id: {\$in: [4000, 4001]}}
- MaxTimeMS:** 5000
- Skip:** 0
- Limit:** 0
- Display:** Documents 1 - 2 of 2

The results are:

```

_id: ObjectId("5f18f71c1525944a648cb54e")
id: 4000
titulo: "Avengers: Endgame (2019)"
genres: "Fantasy|Sci-Fi"

_id: ObjectId("5f18f71c1525944a648cb54f")
id: 4001
titulo: "Glass (2019)"
genres: "Drama|Fantasy"
valoraciones: Array
  0: Object
    userid: "1563"
    movieid: "4001"
    rating: "4"
  1: Object
    userid: "434"
    movieid: "4001"
    rating: "5"

```

SESION 8

Reto 1

```

[{$match: {
  $and: [
    {Latitude:{$gte: 19.5}},
    {Latitude:{$lte: 19.5}},
    {Longitude:{$gte: -99.2}},
    {Longitude:{$lte: -99.2}}
  ]
}}]

```

The screenshot shows the MongoDB aggregation pipeline interface. At the top, it says "25600 Documents in the Collection". Below that, there's a preview of documents and a section for selecting operators to construct expressions. The main area shows a query stage:

```

1 /**
2  * query: The query in MQL.
3 */
4 {
5   $and: [
6     {Latitude:{$gte: 19.5}},
7     {Latitude:{$lte: 19.5}},
8     {Longitude:{$gte: -99.2}},
9     {Longitude:{$lte: -99.2}}
10 ]
11 }

```

Below this, it says "Output after \$match stage (Sample of 1 document)". It shows two documents:

Document 1:

```

_id: ObjectId("5f1f717cb2012a3c40029a54")
Brand: "Starbucks"
Store Number: "17351-181185"
Store Name: "El Rosario"
Ownership Type: "Licensed"
Street Address: "Av. Del Rosario No. 1025, Col El Rosario"
City: "Mexico City"
State/Province: "DIF"
Country: "MX"

```

Document 2:

```

_id: ObjectId("5f1f717cb2012a3c40029a54")
Brand: "Starbucks"
Store Number: "22331-212325"
Store Name: "Ajman Drive Thru"
Ownership Type: "Licensed"
Street Address: "1 Street 6"
City: "Ajman"
State/Province: "AJ"
Country: "AE"

```

At the bottom, there's an "ADD STAGE" button.

Reto 2

- ¿Cuál fue el país con mayor número de muertes?

```

[{$group: {
  _id: '$Country',
  Casos: {
    $sum: '$Cases'
  },
  Muertes: {$sum: '$Deaths'}
}}, {$sort: {
  Muertes: -1
}}, {$match: {
  _id: {$nin: [/Grand Total/]}}
}, {$limit: 1}]

```

```
_id: "Mexico"  
Casos: 142567  
Muertes: 2271
```

- ¿Cuál fue el país con menor número de muertes?

```
[$group: {  
  _id: '$Country',  
  Casos: {  
    $sum: '$Cases'  
  },  
  Muertes: {$sum: '$Deaths'}  
}, {$sort: {  
  Muertes: 1  
}}, {$match: {  
  _id: {$nin: [/Grand Total/]}}  
}, {$limit: 1}]
```

```
_id: "Cap Verde"  
Casos: 18  
Muertes: 0
```

- ¿Cuál fue el país con el mayor número de casos?

```
[$group: {
```

```
_id: '$Country',
Casos: {
  $sum: '$Cases'
},
Muertes: {$sum: '$Deaths'}
}}, {$sort: {
  Casos: -1
}}, {$match: {
  _id: {$nin: [/Grand Total/]}
}}, {$limit: 1}]
```

```
_id: "United States of America"
Casos: 340804
Muertes: 1150
```

- ¿Cuál fue el país con el menor número de casos?

```
[{$group: {
  _id: '$Country',
  Casos: {
    $sum: '$Cases'
  },
  Muertes: {$sum: '$Deaths'}
}}, {$sort: {
  Casos: 1
}}, {$match: {
  _id: {$nin: [/Grand Total/]}
}}, {$limit: 1}]
```

```
_id: "Libya"
Casos: 1
Muertes: 0
```

- ¿Cuál fue el número de muertes promedio?

```
[$group: {
  _id: '$Country',
  Casos: {
    $sum: '$Cases'
  },
  Muertes: {$sum: '$Deaths'}
}

},
{$match: {
  _id: {$nin: [/Grand Total/]}
}}, {$group: {
  _id: null,
  Casos: {
    $sum: '$Casos'
  },
  Muertes: {$sum: '$Muertes'},
  Paises: {$sum: 1}
}}, {$addFields: {
  Muertes_prom: {$divide: ['$Muertes', '$Paises']}
}}, {$project: {
  _id: 0,
  Muertes_prom: 1
}}]
```

Muertes_prom: 27.05298013245033

- ¿Cuál fue el número de casos promedio?

```
[$group: {  
    _id: '$Country',  
    Casos: {  
        $sum: '$Cases'  
    },  
    Muertes: {$sum: '$Deaths'}  
  
},  
{$match: {  
    _id: {$nin: [/Grand Total/]}}  
  
},  
{$group: {  
    _id: null,  
    Casos: {  
        $sum: '$Casos'  
    },  
    Muertes: {$sum: '$Muertes'},  
    Paises: {$sum: 1}  
}},  
{$addFields: {  
    Casos_prom: {$divide: ['$Casos', '$Paises']}}},  
{$project: {  
    _id: 0,  
    Casos_prom: 1  
}}]
```

Casos_prom: 5668.463576158941

- Top 5 de países con más muertes

```
[{$group: {  
    _id: '$Country',  
    Casos: {  
        $sum: '$Cases'  
    },  
    Muertes: {$sum: '$Deaths'}  
  
},  
}, {$match: {  
    _id: {$nin: [/Grand Total/]}  
  
}}, {$sort: {  
    Muertes: -1  
}}, {$limit: 5}]
```

_id: "Mexico"
Casos: 142567
Muertes: 2271

_id: "United States of Amer
Casos: 340804
Muertes: 1150

- Top 5 de países con menos muertes

```

[{$group: {
  _id: '$Country',
  Casos: {
    $sum: '$Cases'
  },
  Muertes: {$sum: '$Deaths'}

},
{$match: {
  _id: {$nin: [/Grand Total/]}
}

}, {$sort: {
  Muertes: 1
}}, {$limit: 5}]

```

`_id: "Slovakia"`
`Casos: 101`
`Muertes: 0`

`_id: "Sri Lanka"`
`Casos: 78`
`Muertes: 0`

RETO 3

- ¿Cuál es país con mayor número de casos?

```

[$group: {
  _id: '$Region',
  Casos: {
    $sum: '$Confirmed'
  },
  Muertes: {
    $sum: '$Deaths'
  }
}, {$sort: {
  Casos: -1
}}, {$limit: 1}]

```

```
_id: "Mainland China"  
Casos: 2369152  
Muertes: 65325
```

- ¿Cuál es el país con mayor número de muertes?

```
[$group: {  
    _id: '$Region',  
    Casos: {  
        $sum: '$Confirmed'  
    },  
    Muertes: {  
        $sum: '$Deaths'  
    }  
}, {$sort: {  
    Muertes: -1  
}}, {$limit: 1}]
```

```
_id: "Mainland China"  
Casos: 2369152  
Muertes: 65325
```

- Usando las coordenadas, encuentra el epicentro del virus.

```
[$group: {  
    _id: '$Province',  
    Lat: {
```

```
$max: '$Lat'  
},  
    Long: {  
        $max: '$Long'  
    }  
}, {$group: {  
    _id: null,  
    TLat: {  
        $sum: '$Lat'  
    },  
    TLong: {  
        $sum: '$Long'  
    }  
},  
    TP: {  
        $sum: 1  
    }  
}  
}, {$addFields: {  
    Prom_lat: {$divide: ['$TLat','$TP']},  
    Prom_Lon: {$divide: ['$TLong','$TP']}
```

}}, {\$project: {
 Prom_lat: 1,
 Prom_Lon: 1
}}]

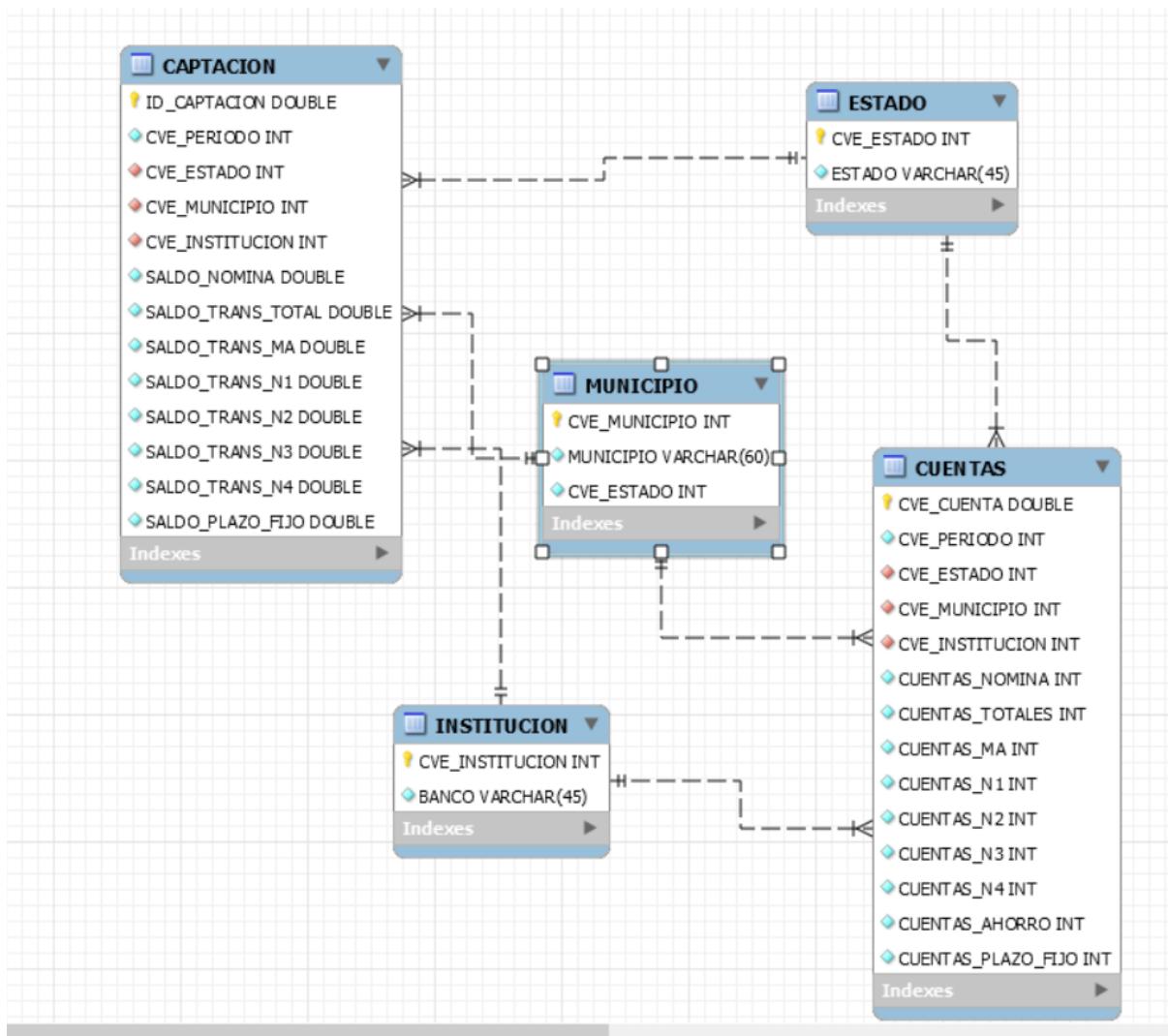
```
_id: null  
Prom_lat: 29.27755833333333  
Prom_Lon: 36.541709375
```

- Usando el epicentro, encuentra las 5 regiones más cercanas a dicho epicentro

POSTWORK

189.216.106.130/32

Diagrama



SQLWORKBENCH

1.1 Usando la base de datos SA, mostrar las tablas que contienen y la estructura de la tabla "cuenta"

```
USE SA;
SHOW TABLES;
DESCRIBE CUENTAS;
```

	Tables_in_sau
▶	captacion
	cuentas
	estado
	institucion
	municipio

Result Grid | Filter Rows: Export: Wrap Cell Content:

	Field	Type	Null	Key	Default	Extra
▶	CVE CUENTA	double	NO	PRI	NULL	
	CVE PERIODO	int	NO		NULL	
	CVE ESTADO	int	NO	MUL	NULL	
	CVE MUNICIPIO	int	NO	MUL	NULL	
	CVE INSTITUCION	int	NO	MUL	NULL	
	CUENTAS NOMINA	int	NO		NULL	
	CUENTAS TOTALES	int	NO		NULL	
	CUENTAS MA	int	NO		NULL	
	CUENTAS NIVEL1	int	NO		NULL	
	CUENTAS NIVEL2	int	NO		NULL	
	CUENTAS NIVEL3	int	NO		NULL	
	CUENTAS NIVEL4	int	NO		NULL	

Result 19 ×

Output Action Output

1.2 Cuáles son los municipios del estado 4

```
SELECT *
FROM MUNICIPIO
WHERE num_estado = 4
```

Result Grid | Filter Rows: Edit:

	CVE_MUNICIPIO	MUNICIPIO	NUM_ESTADO
▶	4001	Calkini	4
	4002	Campeche	4
	4003	Ciudad Del Carmen	4
	4004	Champoton	4
	4005	Hecelchakan	4
	4006	Hopelchen	4
	4007	Palizada	4
	4008	Tenabo	4
	4009	Escarcega	4
	4010	Calakmul	4
	4011	Candelaria	4
	NULL	NULL	NULL

MUNICIPIO 17 ×

Output

1.3 Cuáles son los banco y que saldo total de cuentas transaccionales que se tienen en mayo del 2020 (202005), que no tengan saldo de plazo fijo y que sean del municipio 2002

```
SELECT  
CVE_INSTITUCION,  
SALDO_TRANS_TOTAL  
from CAPTACION  
WHERE CVE_PERIODO = 202005  
AND CVE_MUNICIPIO = 2002  
AND SALDO_PLAZO_FIJO > 0
```

CVE_INSTITUCION	SALDO_TRANS_TOTAL
40002	7164755915
40012	10844727208
40014	4872665553
40021	4002187483
40030	555996400
40036	449849133
40044	2276675580
40058	1241892451
40072	7175954720
40112	125885538
40127	424413812
40178	371677

1.4 Cual es el top 9 de bancos que tienen más cuentas nivel 2 en abril y que sean del municipio 09014

```
SELECT  
CVE_INSTITUCION,  
CUENTAS_NIVEL2  
FROM CUENTAS  
WHERE CVE_MUNICIPIO = 9014  
AND CVE_PERIODO = 202004  
order by CUENTAS_NIVEL2 desc limit 9
```

	CVE_INSTITUCION	CUENTAS_NIVEL2
▶	40012	2374759
	40072	29932
	40002	15393
	40130	11550
	40044	7398
	40014	3455
	40030	1607
	40128	352
	40036	243

CUENTAS 15 ×

1.5 ¿Cuáles son los 20 municipios con menos saldo de plazo fijo del banco 40012 en enero del 2020?

```
select
CVE_MUNICIPIO,
SALDO_PLAZO_FIJO
from CAPTACION
WHERE CVE_INSTITUCION = 40012
AND CVE_PERIODO = 202001
AND SALDO_PLAZO_FIJO>0
ORDER BY SALDO_PLAZO_FIJO
limit 20
```

	CVE_MUNICIPIO	SALDO_PLAZO_FIJO
▶	12045	674000
	19018	1200000
	8062	1403971
	19041	3432294
	4007	3990588
	12040	4428884
	30085	5237457
	23009	6652807
	8052	6694567
	30116	7232000
	15023	8480232
	10031	9360000

CAPTACION 14 ×

Output :::::

Action Output

2.1 Sacar todos los municipios que empiezen con 'Ch'

```
SELECT MUNICIPIO
FROM MUNICIPIO
WHERE MUNICIPIO LIKE 'Ch%'
```

	MUNICIPIO
▶	Champoton
	Chalchihuitan
	Chamula
	Chanal
	Chapultenango
	Chenalho
	Chiapa De Corzo
	Chiapilla
	Chicoasen
	Chicomuselo
	Chilon
	Chilches
MUNICIPIO 13 ×	

2.2 Obtener el total de cuentas (Nomina, plazo fijo y transaccionales) en mayo del 2020 y el total general de estas

```
select
SUM(CUENTAS_TOTALES) AS CUENTAS_TRANSACCIONALES,
SUM(CUENTAS_NOMINA) AS NOMINA,
SUM(CUENTAS_PLAZOFIJO) AS PLAZO_FIJO,
SUM(CUENTAS_TOTALES) + SUM(CUENTAS_NOMINA) + SUM(CUENTAS_PLAZOFIJO)
AS TOTAL_GENERAL
FROM
CUENTAS
WHERE CVE_PERIODO = 202005
```

Result Grid Filter Rows: <input type="text"/> Export: Wrap Cell Content:				
	CUENTAS_TRANSACCIONALES	NOMINA	PLAZO_FIJO	TOTAL_GENERAL
▶	98686008	35205427	2854900	136746335

2.3 Obtener el total de saldo de plazo fijo que hay por institución en cada periodo y el numero de municipios en donde estan

Select
CVE_PERIODO,
CVE_INSTITUCION,

```

SUM(SALDO_PLAZO_FIJO) AS SALDO_PLAZO_FIJO,
COUNT(CVE_MUNICIPIO) AS MUNICIPIO
from CAPTACION
WHERE SALDO_PLAZO_FIJO > 0
group by
CVE_PERIODO,
CVE_INSTITUCION

```

Result Grid | Filter Rows: [] Export: [] Wrap Cell Content: []

	CVE_PERIODO	CVE_INSTITUCION	SALDO_PLAZO_FIJO	MUNICIPIO
▶	202005	40002	166023400366	509
	202005	40012	295956010928	572
	202005	40014	286377777868	340
	202005	40021	174759957555	310
	202005	40030	87123335232	120
	202005	40036	23167715496	178
	202005	40042	21959984896	32
	202005	40044	195069318602	196
	202005	40058	54369422715	60
	202005	40072	298491385936	331
	202005	40112	29060051148	36
	202005	40127	19549540854	771

Result 40 ×

Output: [] Action Output: []

2.4 Cual es el nombre las instituciones bancarias que no tienen saldo plazo fijo en el municipio 2002 o tienen cuenta nivel 3 en cualquier periodo

```

SELECT BANCO
FROM INSTITUCION
WHERE CVE_INSTITUCION IN
(select cve_institucion
FROM
CAPTACION
where SALDO_PLAZO_FIJO > 0
AND CVE_MUNICIPIO = 2002)
OR CVE_INSTITUCION IN
(select
cve_institucion
FROM CUENTAS
WHERE CUENTAS_NIVEL3 > 0)

```

BANCO
Banamex
BBVA Bancomer
Santander
HSBC
Banco del Bajío
Inbursa
Scotiabank
Banregio
Banorte
Monex
Banco Azteca
Autofin
INSTITUCION 10

2.5 ¿Cuál es el total de cuentas de Compartamos que se tienen por estado y en cuantos municipios tienen en mayo?

```

SELECT
CVE_ESTADO,
SUM(CUENTAS_TOTALES) AS CUENTAS,
COUNT(CVE_MUNICIPIO) AS MUNICIPIOS
FROM CUENTAS
WHERE CVE_PERIODO = 202005
AND CUENTAS_TOTALES > 0
AND CVE_INSTITUCION = (Select CVE_INSTITUCION
from INSTITUCION
WHERE BANCO LIKE '%compartamos%')
GROUP BY CVE_ESTADO

```

	CVE_ESTADO	CUENTAS	MUNICIPIOS
▶	1	532	1
	2	12703	5
	3	4425	3
	4	6311	5
	5	13997	12
	6	765	3
	7	20821	20
	8	4204	7
	9	22895	12
	10	11011	7
	11	7837	16
	12	19545	14

Result 59 ×

3.1 Cuáles son los 3 bancos con más saldo de nomina de febrero 2020

```

select
B.BANCO,
C.NOMINA AS SALDO
from
(
SELECT
CVE_INSTITUCION,
SUM(SALDO_NOMINA) AS NOMINA
FROM
CAPTACION
WHERE CVE_PERIODO = 202002
GROUP BY CVE_INSTITUCION
) AS C
LEFT JOIN INSTITUCION B
ON C.CVE_INSTITUCION = B.CVE_INSTITUCION
ORDER BY C.NOMINA DESC LIMIT 3

```

BANCO	SALDO
BBVA Bancomer	167539708646
Banorte	76432707031
Banamex	73524842186

3.2 Cuántos bancos tienen al menos una cuenta N4 por municipio y estado al cierre de mayo

```

SELECT
E.ESTADO,

```

```

M.MUNICIPIO,
BANCOS
FROM
(
SELECT
COUNT(CVE_INSTITUCION) AS BANCOS,
CVE_MUNICIPIO,
CVE_ESTADO
from CUENTAS
WHERE CUENTAS_TOTALES > 0
AND CVE_PERIODO = 202005
GROUP BY
CVE_MUNICIPIO,
CVE_ESTADO
) AS C
LEFT JOIN
MUNICIPIO AS M
ON M.CVE_MUNICIPIO = C.CVE_MUNICIPIO
LEFT JOIN
ESTADO AS E
ON E.CVE_ESTADO = C.CVE_ESTADO

```

	ESTADO	MUNICIPIO	BANCOS
▶	Aguascalientes	Aguascalientes	20
	Aguascalientes	Asientos	1
	Aguascalientes	Calvillo	5
	Aguascalientes	Cosio	1
	Aguascalientes	Jesus Maria (Aguascalientes)	4
	Aguascalientes	Pabellon De Arteaga	5
	Aguascalientes	Rincon De Romos	5
	Aguascalientes	San Jose De Gracia	1
	Aguascalientes	Tepezala	1
	Aguascalientes	El Llano	1
	Aguascalientes	San Francisco De Los Romo	2
	Baja California	Ensenada	19

Result 24 ×

Output :::::::::::::::::::::

3.3 Usando la consulta anterior obtener el saldo total que hay por cada municipio

```

SELECT
E.ESTADO,
M.MUNICIPIO,
BANCOS,
S.SALDO
FROM

```

```
(  
SELECT  
COUNT(CVE_INSTITUCION) AS BANCOS,  
CVE_MUNICIPIO,  
CVE_ESTADO  
from CUENTAS  
WHERE CUENTAS_TOTALES > 0  
AND CVE_PERIODO = 202005  
GROUP BY  
CVE_MUNICIPIO,  
CVE_ESTADO  
) AS C  
LEFT JOIN  
MUNICIPIO AS M  
ON M.CVE_MUNICIPIO = C.CVE_MUNICIPIO  
LEFT JOIN  
ESTADO AS E  
ON E.CVE_ESTADO = C.CVE_ESTADO  
LEFT JOIN  
(SELECT  
CVE_MUNICIPIO,  
SUM(SALDO_TRANS_TOTAL) AS SALDO  
from CAPTACION  
WHERE CVE_PERIODO = 202005  
GROUP BY  
CVE_MUNICIPIO,  
CVE_ESTADO  
) AS S  
ON S.CVE_MUNICIPIO = C.CVE_MUNICIPIO
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	ESTADO	MUNICIPIO	BANCOS	SALDO
▶	Aguascalientes	Aguascalientes	20	33207134998
	Aguascalientes	Asientos	1	605215
	Aguascalientes	Calvillo	5	653178835
	Aguascalientes	Cosio	1	71972
	Aguascalientes	Jesus Maria (Aguascalientes)	4	187072481
	Aguascalientes	Pabellon De Arteaga	5	466851427
	Aguascalientes	Rincon De Romos	5	600247625
	Aguascalientes	San Jose De Gracia	1	79014
	Aguascalientes	Tepezala	1	78923018
	Aguascalientes	El Llano	1	227337
	Aguascalientes	San Francisco De Los Romo	2	39855319
	Raia California	Fresanada	18	13040031500
Result 30 ×				
Output :::				
<input type="checkbox"/> Action Output				

3.4 Hacer la consulta anterior usando vistas

```
CREATE VIEW TBANCOS_MAYO AS
(
SELECT
COUNT(CVE_INSTITUCION) AS BANCOS,
CVE_MUNICIPIO,
CVE_ESTADO
from CUENTAS
WHERE CUENTAS_TOTALES > 0
AND CVE_PERIODO = 202005
GROUP BY
CVE_MUNICIPIO,
CVE_ESTADO
);
```

```
CREATE VIEW TSALDO_MAYO AS
(
SELECT
CVE_MUNICIPIO,
SUM(SALDO_TRANS_TOTAL) AS SALDO
from CAPTACION
WHERE CVE_PERIODO = 202005
GROUP BY
CVE_MUNICIPIO,
CVE_ESTADO
```

);

```
SELECT
E.ESTADO,
M.MUNICIPIO,
BANCOS,
S.SALDO
FROM TBANCOS_MAYO AS C
LEFT JOIN
MUNICIPIO AS M
ON M.CVE_MUNICIPIO = C.CVE_MUNICIPIO
LEFT JOIN
ESTADO AS E
ON E.CVE_ESTADO = C.CVE_ESTADO
LEFT JOIN
TSALDO_MAYO
AS S
ON S.CVE_MUNICIPIO = C.CVE_MUNICIPIO;
```

The screenshot shows a database query results grid titled 'Result Grid'. The grid has columns: ESTADO, MUNICIPIO, BANCOS, and SALDO. The data is as follows:

	ESTADO	MUNICIPIO	BANCOS	SALDO
▶	Aguascalientes	Aguascalientes	20	33207134998
	Aguascalientes	Asientos	1	605215
	Aguascalientes	Calvillo	5	653178835
	Aguascalientes	Cosio	1	71972
	Aguascalientes	Jesus Maria (Aguascalientes)	4	187072481
	Aguascalientes	Pabellon De Arteaga	5	466851427
	Aguascalientes	Rincon De Romos	5	600247625
	Aguascalientes	San Jose De Gracia	1	79014
	Aguascalientes	Tepezala	1	78923018
	Aguascalientes	El Llano	1	227337
	Aguascalientes	San Francisco De Los Romo	2	39855319
	Baja California	Ensenada	18	13060031500

3.5 Usando el ejercicio anterior, pero que se muestre el total de cuentas que había en mayo

```
CREATE VIEW TBANCOS_MAYO AS
(
SELECT
COUNT(CVE_INSTITUCION) AS BANCOS,
CVE_MUNICIPIO,
```

```
CVE_ESTADO  
from CUENTAS  
WHERE CUENTAS_TOTALES > 0  
AND CVE_PERIODO = 202005  
GROUP BY  
CVE_MUNICIPIO,  
CVE_ESTADO  
);
```

```
CREATE VIEW TSALDO_MAYO AS  
(  
SELECT  
CVE_MUNICIPIO,  
SUM(SALDO_TRANS_TOTAL) AS SALDO  
from CAPTACION  
WHERE CVE_PERIODO = 202005  
GROUP BY  
CVE_MUNICIPIO,  
CVE_ESTADO  
);
```

```
CREATE VIEW TCUENTAS_MAYO AS  
(  
SELECT  
CVE_MUNICIPIO,  
SUM(CUENTAS_TOTALES) AS CUENTAS  
from CUENTAS  
WHERE CVE_PERIODO = 202005  
GROUP BY  
CVE_MUNICIPIO,  
CVE_ESTADO  
);
```

```
SELECT  
E.ESTADO,  
M.MUNICIPIO,  
B.BANCOS,  
S.SALDO,  
C.CUENTAS  
FROM TBANCOS_MAYO AS B  
LEFT JOIN  
MUNICIPIO AS M  
ON M.CVE_MUNICIPIO = B.CVE_MUNICIPIO  
LEFT JOIN  
ESTADO AS E  
ON E.CVE_ESTADO = B.CVE_ESTADO  
LEFT JOIN
```

```

TSALDO_MAYO
AS S
ON S.CVE_MUNICIPIO = B.CVE_MUNICIPIO
LEFT JOIN
TCUENTAS_MAYO
AS C
ON C.CVE_MUNICIPIO = B.CVE_MUNICIPIO

```

The screenshot shows a MongoDB query result grid with the following columns: ESTADO, MUNICIPIO, BANCOS, SALDO, and CUENTAS. The data is as follows:

ESTADO	MUNICIPIO	BANCOS	SALDO	CUENTAS
Aguascalientes	Aguascalientes	20	33207134998	1038046
Aguascalientes	Asientos	1	605215	1322
Aguascalientes	Calvillo	5	653178835	19593
Aguascalientes	Cosio	1	71972	326
Aguascalientes	Jesus Maria (Aguascalientes)	4	187072481	17899
Aguascalientes	Pabellon De Arteaga	5	466851427	17837
Aguascalientes	Rincon De Romos	5	600247625	23695
Aguascalientes	San Jose De Gracia	1	79014	175
Aguascalientes	Tepezala	1	78923018	4127
Aguascalientes	El Llano	1	227337	420
Aguascalientes	San Francisco De Los Romo	2	39855319	4584
Baja California	Ensenada	18	13060031500	421517

Result 33 ×

Output

Action Output

MONGO

4.1 Mostrar los datos de cuentas totales y de instituciones

```
{
  CVE_INSTITUCION: 1,
  CUENTAS_TOTALES: 1
}
```

The screenshot shows a MongoDB query interface with the following configuration:

- FILTER:** {
 PROJECT: {
 CVE_INSTITUCION: 1,
 CUENTAS_TOTALES: 1
 }
}
- SORT:** {
 MAXTIMEMS: 5000
}
- COLLATION:** {
 SKIP: 0
 LIMIT: 0
}
- OPTIONS:** {
 OPTIONS: {
 MAXTIMEMS: 5000
 }
}
- Buttons:** FIND, RESET, ...
- Display:** Showing documents 1 - 20 of 35734.

The results list five documents, each containing the following fields:

- _id:** ObjectId("5f1dc06a26c880560c3cb854")
CVE_INSTITUCION: 40002
CUENTAS_TOTALES: 329701
- _id:** ObjectId("5f1dc06a26c880560c3cb855")
CVE_INSTITUCION: 40012
CUENTAS_TOTALES: 198784
- _id:** ObjectId("5f1dc06a26c880560c3cb856")
CVE_INSTITUCION: 40014
CUENTAS_TOTALES: 120681
- _id:** ObjectId("5f1dc06a26c880560c3cb857")
CVE_INSTITUCION: 40021
CUENTAS_TOTALES: 82892
- _id:** ObjectId("5f1dc06a26c880560c3cb858")
CVE_INSTITUCION: 40030
CUENTAS_TOTALES: 30847

On the right side, there are two panels showing the **PROJECT** definition:

- PROJECT:** {
 CVE_INSTITUCION: 1,
 CUENTAS_TOTALES: 1
}
- PROJECT:** {
 CVE_INSTITUCION: 1
}

Timestamps indicate the operations were performed on Sun Jul 26 2020 at 13:00:18 GMT-0500 and 12:59:22 GMT-0500.

4.2 ¿Cuáles son los municipios del estado 4?

FILTER

```
{
  CVE_ESTADO: 4
}
```

PROJECT

```
{
  _id: 0,
  MUNICIPIO: 1
}
```

The screenshot shows a MongoDB query interface with the following configuration:

- FILTER:** `{CVE_ESTADO: 4}`
- PROJECT:** `{_id: 0, MUNICIPIO: 1}`
- SORT:** None
- COLLATION:** None
- MAXTIMEMS:** 5000
- SKIP:** 0
- LIMIT:** 0

The results are displayed as follows:

- MUNICIPIO: "Calkini"
- MUNICIPIO: "Campeche"
- MUNICIPIO: "Ciudad Del Carmen"
- MUNICIPIO: "Champoton"
- MUNICIPIO: "Hecelchakan"
- MUNICIPIO: "Hopechen"
- MUNICIPIO: "Palizada"
- MUNICIPIO: "Tenabo"

On the right side, there are two timestamped log entries:

- Sun Jul 26 2020 13:06:09 GMT-0500 (ho...)**
- Sun Jul 26 2020 13:05:47 GMT-0500 (ho...)**

4.3 ¿Cuáles son los banco y que saldo total de cuentas transaccionales que se tienen en mayo del 2020 (202005), que no tengan saldo de plazo fijo y que sean del municipio 2002?

FILTER

```
{
  CVE_MUNICIPIO: 2002,
  SALDO_PLAZO_FIJO: {
    $gt: 0
  },
  CVE_PERIODO: 202005
}
```

PROJECT

```
{
  _id: 0,
  CVE_INSTITUCION: 1,
  SALDO_TRANS_TOTAL: 1
}
```

The screenshot shows the MongoDB Compass interface with a search query and its results.

Search Query (Left):

```
{
  FILTER {CVE_MUNICIPIO: 2002, SALDO_PLAZO_FIJO: {$gt: 0}, CVE_PERIODO: 202005}
  PROJECT {_id: 0, CVE_INSTITUCION: 1, SALDO_TRANS_TOTAL: 1}
  SORT
  COLLATION
  MAXTIMEMS 500
  SKIP 0
  LIMIT 0
}
```

Results (Left):

- CVE_INSTITUCION: 40002
SALDO_TRANS_TOTAL: 7164755915
- CVE_INSTITUCION: 40012
SALDO_TRANS_TOTAL: 10844727208
- CVE_INSTITUCION: 40014
SALDO_TRANS_TOTAL: 4872665553
- CVE_INSTITUCION: 40021
SALDO_TRANS_TOTAL: 4002187483
- CVE_INSTITUCION: 40030
SALDO_TRANS_TOTAL: 555996400
- CVE_INSTITUCION: 40036
SALDO_TRANS_TOTAL: 449849133

Right Panel (MongoDB Query Log):

```

FILTER
{
  CVE_MUNICIPIO: 2002,
  SALDO_PLAZO_FIJO: {
    $gt: 0
  },
  CVE_PERIODO: 202005
}

PROJECT
{
  _id: 0,
  CVE_INSTITUCION: 1,
  SALDO_TRANS_TOTAL: 1
}

Sun Jul 26 2020 13:16:25 GMT-0500 (ho..
```

4.4 ¿Cuál es el top 9 de bancos que tienen más cuentas nivel 2 en abril y que sean del municipio 09014?

FILTER

```
{
  CVE_PERIODO: 202004,
  CVE_MUNICIPIO: 9014
}
```

PROJECT

```
{
  _id: 0,
  CVE_INSTITUCION: 1
}
```

SORT

```
{
  CUENTAS_NIVEL2: -1
}
```

LIMIT

9

The screenshot shows a MongoDB query interface with the following details:

- Documents** tab selected.
- FILTER**: `{CVE_PERIODO: 202004, CVE_MUNICIPIO: 9014}`
- PROJECT**: `{_id:0,CVE_INSTITUCION: 1}`
- SORT**: `{CUENTAS_NIVEL2: -1}`
- OPTIONS**: `MAXTIMEMS 5000`
- SKIP**: 0
- LIMIT**: 9
- RESULTS**: Displays 9 documents, all with `CVE_INSTITUCION: 40012`.
- DISPLAY**: Shows the date `Sun Jul 26 2020 13:21:03 GMT-0500 (ho...)`.

4.5 ¿Cuáles son los 20 municipios con menos saldo de plazo fijo del banco 40012 en enero del 2020?

```

FILTER
{
  CVE_PERIODO: 202004,
  CVE_MUNICIPIO: 9014
}

PROJECT
{
  _id: 0,
  CVE_INSTITUCION: 1
}

SORT
{
  CUENTAS_NIVEL2: -1
}

LIMIT
9

```

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {CVE_INSTITUCION: 40012, CVE_PERIODO: 202001, SALDO_PLAZO_Fijo:{\$gt:0}}

PROJECT {_id:0,CVE_MUNICIPIO:1}

SORT {SALDO_PLAZO_Fijo:1} **MAXTIMEMS** 5000

COLLATION **SKIP** 0 **LIMIT** 0

VIEW REFRESH

Displaying documents 1 - 20 of 551

CVE_MUNICIPIO: 12045

CVE_MUNICIPIO: 19018

CVE_MUNICIPIO: 8062

CVE_MUNICIPIO: 19041

CVE_MUNICIPIO: 4007

CVE_MUNICIPIO: 12040

CVE_MUNICIPIO: 30085

CVE_MUNICIPIO: 23009



5.1 Sacar todos los municipios que empiezen con 'Ch'

FILTER

```
{
  MUNICIPIO: RegExp ('^Ch', i)
}
```

PROJECT

```
{
  _id: 0,
  MUNICIPIO: 1
}
```

The screenshot shows a MongoDB query interface with the following details:

- FILTER:** {MUNICIPIO: /*^Ch/i>}*
- PROJECT:** {_id: 0, MUNICIPIO: 1}
- SORT:** (not explicitly shown)
- COLLATION:** (not explicitly shown)
- OPTIONS:** MAXTIMEMS 5000
- Buttons:** FIND, RESET, ...
- Display:** Showing 1 - 20 of 78 documents.
- Results:** A list of Municipios starting with "Ch" (Champoton, Chalchihuitan, Chamula, Chanal, Chapultenango, Chenalho, Chiapa De Corzo).

On the right side, there are two timestamped sections of log output:

- Sun Jul 26 2020 13:56:50 GMT-0500 (ho...)**
 - FILTER:** {MUNICIPIO: /*^Ch/i>}*
 - PROJECT:** {_id: 0, MUNICIPIO: 1}
- Sun Jul 26 2020 13:56:23 GMT-0500 (ho...)**
 - FILTER:** {MUNICIPIO: /*^Ch/i>}*

5.2 Mostrar la institución, estado y municipio de aquellos bancos que tengan cuentas n1 o n2 en enero 2020

```
[{$match: {
  $or: [
    {CUNETAS_NIVEL1: {$gt: 0}}
  ,
    {CUENTAS_NIVEL2: {$gt: 0}}
  ]
,
  CVE_PERIODO: 202005
}},{$_project: {_id: 0,
  CVE_ESTADO: 1,
  CVE_MUNICIPIO: 1,
  CVE_INSTITUCION: 1
} }]
```

aggregation pipeline stages. [Learn more](#)

The screenshot shows the MongoDB aggregation pipeline interface. It consists of two main sections: the top section displays the pipeline stages and their results, and the bottom section shows the raw MQL code for each stage.

Output after \$match stage (Sample of 20 documents)

```

1 /**
2  * query: The query in MQL.
3 */
4 {
5   $or: [
6     {CUENTAS_NIVEL1: {$gt: 0}}
7     ,{CUENTAS_NIVEL2: {$gt: 0}}
8   ]
9   CVE_PERIODO: 202005
10 }
11
12
13 }
```

Output after \$project stage (Sample of 20 documents)

```

1 /**
2  * specifications: The fields to
3  * include or exclude.
4 */
5 {_id: 0,
6  CVE_ESTADO: 1,
7  CVE_MUNICIPIO: 1,
8  CVE_INSTITUCION: 1
9 }
```

NOMINA: {

CUENTAS_TRANS: {

CUENTAS_PLAZO_FIJO: {

CVE_MUNICIPIO: 1001

CVE_ESTADO: 1

CVE_INSTITUCION: 40012

CUENTAS_NOMINA: 107909

CUENTAS_PLAZOFIJO: 3093

CUENTAS_TOTALES: 198784

CVE_MUNICIPIO: 1001

CVE_ESTADO: 1

CVE_INSTITUCION: 40012

CUENTAS_NOMINA: 107909

CUENTAS_PLAZOFIJO: 3093

CUENTAS_TOTALES: 198784

CVE_MUNICIPIO: 1001

CVE_ESTADO: 1

CVE_INSTITUCION: 40012

CUENTAS_NOMINA: 107909

CUENTAS_PLAZOFIJO: 3093

CUENTAS_TOTALES: 198784

5.3 Obtener el total de cuentas (Nomina, plazo fijo y transaccionales) en mayo del 2020 y el total general de estas

```

[{$match: {
  CVE_PERIODO: 202005
}}, {$group: {
  _id: null,
  NOMINA: {
    $sum: '$CUENTAS_NOMINA'
  },
  CUENTAS_TRANS: {
    $sum: '$CUENTAS_TOTALES'
  },
  CUENTAS_PLAZO_FIJO: {
    $sum: '$CUENTAS_PLAZOFIJO'
  }
}, {$project: {_id:0,
  NOMINA: 1,
  CUENTAS_TRANS: 1,
  CUENTAS_PLAZO_FIJO: 1,
  TOTAL: {$add:['$NOMINA','$CUENTAS_TRANS','$CUENTAS_PLAZO_FIJO']}
}]]
```

The screenshot shows the MongoDB aggregation pipeline interface. On the left, there is a code editor with the following code:

```

1 /**
2  * specifications: The fields to
3  * include or exclude.
4 */
5 { _id:0,
6  NOMINA: 1,
7  CUENTAS_TRANS: 1,
8  CUENTAS_PLAZO_FIJO: 1,
9  TOTAL: {$add:[ '$NOMINA', '$CUENTAS_TRANS', '$CUENTAS_PLAZO_FIJO' ]},
10 }
11

```

On the right, the output after the \$project stage is shown in a box:

```

NOMINA: 35205427
CUENTAS_TRANS: 98686006
CUENTAS_PLAZO_FIJO: 2854900
TOTAL: 136746333

```

5.4 Obtener el total de saldo de plazo fijo que hay por institución en cada periodo y el número de municipios en donde están

```

[{$match: {
  SALDO_PLAZO_FIJO: {$gt:0}
}}, {$group: {
  _id: {PERIODO: '$CVE_PERIODO', BANCO: '$CVE_INSTITUCION'},
  PLAZO_FIJO: { $sum: '$SALDO_PLAZO_FIJO' },
  MUNICIPIO: {$sum: 1}
}}]

```

The screenshot shows the MongoDB aggregation pipeline interface. On the left, there is a code editor with the following code:

```

1 /**
2  * _id: The id of the group.
3  * fieldId: The first field name.
4 */
5 {
6  _id: {PERIODO: '$CVE_PERIODO', BANCO: '$CVE_INSTITUCION'},
7  PLAZO_FIJO: { $sum: '$SALDO_PLAZO_FIJO' },
8  MUNICIPIO: {$sum: 1}
9 }
10

```

On the right, the output after the \$group stage is shown in two boxes:

```

_id:Object
PLAZO_FIJO: 12649788
MUNICIPIO: 1

```

```

_id:Object
PLAZO_FIJO: 903220
MUNICIPIO: 1

```

5.5 Obtener el saldo de las cuentas N3 que hay por estado de cada institución en enero del 2020

```

[{
  $match: {
    CVE_PERIODO: 202001
  }
}, {
  $group: {
    _id: {
      ESTADO: '$CVE_ESTADO',
      INSTITUCION: '$CVE_INSTITUCION'
    }
  }
}]

```

```

        },
        SALDON4: {
            $sum: '$SALDO_TRANS_N4'
        }
    }
}]

```

The screenshot shows the MongoDB Aggregations interface. On the left, the sidebar lists databases (Local, 4 DBs) and collections (15 Collections). The current collection is 'SAU.Captacion'. The top right shows summary statistics: DOCUMENTS 35.7k, TOTAL SIZE 11.4MB, AVG. SIZE 334B, INDEXES 1, TOTAL SIZE 356.0KB, AVG. SIZE 356.0KB. Below the summary are tabs for Documents, Aggregations (selected), Schema, Explain Plan, Indexes, and Validation. The Aggregations tab displays a pipeline with three stages:

- \$group**: Groups documents by '_id' (with fields 'CVE_ESTADO' and 'CVE_INSTITUCION'), 'NOMINA' (calculated using '\$sum: '\$CUENTAS_NOMINA') and 'SALDON4'.
- \$addFields**: Adds a new field 'NOMINA' calculated as '\$sum: '\$CUENTAS_NOMINA' / '\$BANCOS'.
- \$project**: Projects fields '_id', 'ESTADO', 'INSTITUCION', 'SALDON4', and 'NOMINA'.

The results pane shows a sample of 20 documents with various values for these fields.

6.1 Obtener el promedio de cuentas por institución que hay por cada estado, en mayo

```

[{$match: {
    CVE_PERIODO: 202005,
    CUENTAS_NOMINA: {$gt:0}
}}, {$group: {
    _id: {ESTADO: '$CVE_ESTADO', INSTITUCION: '$CVE_INSTITUCION'},
    NOMINA: {
        $sum: '$CUENTAS_NOMINA'
    }
}}, {$addFields: {
    CVE_ESTADO: '$_id.ESTADO'
}}, {$group: {
    _id: '$CVE_ESTADO',
    NOMINA: {
        $sum: '$NOMINA'
    },
    BANCOS: {$sum: 1}
}}, {$addFields: {
    CUENTAS_NOM_PROM:{$divide: ['$NOMINA','$BANCOS']}
}}, {$project: {

```

```
CUENTAS_NOM_PROM:1
}}]
```

The screenshot shows the MongoDB aggregation pipeline interface. It consists of two main sections: "Output after \$addFields stage" and "Output after \$project stage".

\$addFields Stage:

```
1 /**
2  * newField: The new field name.
3  * expression: The new field expression.
4 */
5 {
6   CUENTAS_NOM_PROM:{$divide: ['$NOMINA','$BANCOS']}
7 }
```

Output after \$addFields stage (Sample of 20 documents):

_id	NOMINA	BANCOS	CUENTAS_NOM_PROM
32	287770	10	28777
6	186082	10	18608.2

\$project Stage:

```
1 /**
2  * specifications: The fields to
3  * include or exclude.
4 */
5 {
6   CUENTAS_NOM_PROM:1
7 }
```

Output after \$project stage (Sample of 20 documents):

_id	CUENTAS_NOM_PROM
28	99128.83333333333
5	95938.461

6.2 Obtener el saldo promedio de las instituciones hay por cada municipio de las cuentas de plazo fijo en febrero 2020, al final ordenar los municipios de mayor a menor por el saldo promedio

```
[$match: {
  SALDO_PLAZO_FIJO: {$gt: 0},
  CVE_PERIODO: 202002
}, {$group: {
  _id: '$CVE_MUNICIPIO',
  SALDO: {
    $sum: '$SALDO_PLAZO_FIJO'
  },
  BANCOS: {$sum: 1}
}}, {$addFields: {
  SALDO_PROMEDIO: {$divide: ['$SALDO','$BANCOS']}
}}, {$sort: {
  SALDO_PROMEDIO: -1
}}]
```

The screenshot shows the MongoDB aggregation pipeline interface with two stages displayed:

- \$addFields Stage:**
 - Code:

```
11 }  
1  /**  
2   * newField: The new field name.  
3   * expression: The new field expression.  
4   */  
5 {  
6   SALDO_PROMEDIO: {$divide: ['$SALDO', '$BANCOS']}
```
 - Output (Sample of 20 documents):
 - _id: 8026
SALDO: 39753652
BANCOS: 1
SALDO_PROMEDIO: 39753652
 - _id: 20318
SALDO: 575632755
BANCOS: 8
SALDO_PROMEDIO: 71954094.37
- \$sort Stage:**
 - Code:

```
1 /**  
2  * Provide any number of field/order pairs.  
3  */  
4 {  
5   SALDO_PROMEDIO: -1  
6 }
```
 - Output (Sample of 20 documents):
 - _id: 9010
SALDO: 163532905320
BANCOS: 19
SALDO_PROMEDIO: 8606995016.842106
 - _id: 9015
SALDO: 222299671982
BANCOS: 26
SALDO_PROMEDIO: 8549987383.

6.3 ¿Cuál es el total de cuentas totales (sin nomina ni plazo fijo) de Compartamos en mayo que se tienen por estado y en cuantos municipios esta?

```
[{$lookup: {
  from: 'Institucion',
  localField: 'CVE_INSTITUCION',
  foreignField: 'CVE_INSTITUCION',
  as: 'BANCO_A'
}}, {$addFields: {
  BANCO_OBJETO: {$arrayElemAt:["$BANCO_A",0]}
}}, {$addFields: {
  BANCO: '$BANCO_OBJETO.BANCO'
}}, {$match: {
  BANCO: /compartamos/i,
  CVE_PERIODO: 202005
}}, {$group: {
  _id: '$CVE_ESTADO',
  CUENTAS: {
    $sum: '$CUENTAS_TOTALES'
  },
  MUNICIPIO: {$sum: 1}
}]]
```

The screenshot shows the MongoDB aggregation pipeline interface. It consists of three main sections: a query editor, a stage editor, and an output viewer.

Query Editor:

```

1 /*
2   * query: The query in MQL.
3   */
4 {
5     BANCO: /compartamos/i,
6     CVE_PERIODO: 202005
7   }
  
```

Stage Editor:

Currently, the stage editor contains a single stage: `$group`.

```

1 /*
2   * _id: The id of the group.
3   * fieldN: The first field name.
4   */
5 {
6   _id: '$CVE_ESTADO',
7   CUENTAS: {
8     $sum: 'SCUENTAS_TOTALES'
9   },
10  MUNICIPIO: {$sum: 1}
11 }
  
```

Output Viewer:

The output shows the results of the aggregation stage. It includes a header for the output and two sample documents.

Header: Output after `$group` stage (Sample of 20 documents)

_id	CUENTAS	MUNICIPIO
11	7837	16
25	6415	6

Buttons at the bottom include `ADD STAGE`, `Run`, and `Cancel`.

6.4 Usando el ejercicio anterior, crear una vista y poner el nombre del estado

```

[{$lookup: {
  from: 'Estado',
  localField: '_id',
  foreignField: 'CVE_ESTADO',
  as: 'ARRE_ESTADO'
}}, {$addFields: {
  ESTADO_OBJETO: {$arrayElemAt: ['$ARRE_ESTADO',0]}
}}, {$addFields: {
  ESTADO: '$ESTADO_OBJETO.ESTADO'
}}, {$project: {_id:0,
  ESTADO: 1,
  CUENTAS: 1,
  MUNICIPIO: 1
}}]
  
```

The screenshot shows the MongoDB aggregation pipeline interface. It consists of two main sections: a left panel for defining stages and a right panel for viewing the output.

Left Panel (Stages):

- \$addFields Stage:** Contains the following stage definition:


```
1 /**
2  * newField: The new field name.
3  * expression: The new field expression.
4 */
5 {
6   ESTADO: '$ESTADO_OBJETO.ESTADO'
7 }
```
- \$project Stage:** Contains the following stage definition:


```
1 /**
2  * specifications: The fields to
3  * include or exclude.
4 */
5 {
6   _id: 1,
7   ESTADO: 1,
8   CUENTAS: 1,
9   MUNICIPIO: 1
10 }
```

Right Panel (Output):

Output after \$addFields stage (Sample of 20 documents):

```
_id: 10
CUENTAS: 11011
MUNICIPIO: 7
▶ ARRE_ESTADO: Array
▶ ESTADO_OBJETO: Object
ESTADO: "Durango"
```

```
_id: 18
CUENTAS: 1480
MUNICIPIO: 6
▶ ARRE_ESTADO: Array
▶ ESTADO_OBJETO: Object
ESTADO: "Nayarit"
```

Output after \$project stage (Sample of 20 documents):

```
CUENTAS: 13997
MUNICIPIO: 12
ESTADO: "Coahuila de Zaragoza"
```

```
CUENTAS: 66329
MUNICIPIO: 29
ESTADO: "Puebla"
```

Bottom Panel:

ADD STAGE

6.5 Hacer una vista del ejercicio 6.2 y poner el nombre del municipio y filtrar para mostrar los municipio que empiezan con ch

```
[{$lookup: {
  from: 'Municipio',
  localField: '_id',
  foreignField: 'CVE_MUNICIPIO',
  as: 'ARRE_ESTADO'
}}, {$addFields: {
  Municipo_objeto:{$arrayElemAt:['$ARRE_ESTADO',0]}
}}, {$addFields: {
  MUNICIPIO: '$Municipio_objeto.MUNICIPIO'
}}, {$match: {
  MUNICIPIO: /ch/i
}}, {$project: {
  SALDO: 1,
  BANCOS: 1,
  SALDO_PROMEDIO: 1,
  MUNICIPIO: 1
}}]
```

The screenshot shows a MongoDB aggregation pipeline editor interface. It displays two stages: `$match` and `$project`.

\$match Stage:

```
1 /**  
2 * query: The query in MQL.  
3 */  
4 {  
5   MUNICIPIO: /^ch/i  
6 }
```

Output after \$match stage (Sample of 20 documents):

```
_id: 8019  
SALDO: 18692362684  
BANCOS: 19  
SALDO_PROMEDIO: 9838808562.3157895  
► ARRE_ESTADO: Array  
► Municipio_objeto: Object  
MUNICIPIO: "Chihuahua"
```

```
_id: 12029  
SALDO: 3038844933  
BANCOS: 12  
SALDO_PROMEDIO: 253237077.7  
► ARRE_ESTADO: Array  
► Municipio_objeto: Object  
MUNICIPIO: "Chilpancingo De"
```

\$project Stage:

```
1 /**  
2 * specifications: The fields to  
3 * include or exclude.  
4 */  
5 {  
6   SALDO: 1,  
7   BANCOS: 1,  
8   SALDO_PROMEDIO: 1,  
9   MUNICIPIO: 1  
10 }
```

Output after \$project stage (Sample of 20 documents):

```
_id: 8019  
SALDO: 18692362684  
BANCOS: 19  
SALDO_PROMEDIO: 9838808562.3157895  
MUNICIPIO: "Chihuahua"
```

```
_id: 12029  
SALDO: 3038844933  
BANCOS: 12  
SALDO_PROMEDIO: 253237077.7  
MUNICIPIO: "Chilpancingo De"
```

Buttons:

- ADD STAGE