

Andrea Valdez Hernández

Sesión 1 - SQL	2
Código	2
Reto 1	2
Reto 2	4
Reto 3	5
Proyecto	5
Sesión 2	11
Código	11
Reto 1	12
Reto 2	13
Reto 3	14
Reto 4	16
Proyecto	17
Sesión 3	20
Código	20
Reto 1	21
Reto 2	22
Proyecto	23
Sesión 4 - Mongo DB	27
Reto 1	27
Reto 2	28
Proyecto	30
Sesión 5	38
Reto 1	38
Reto 2	40
Reto 3	40
Proyecto	41
Sesión 6	42
Reto 1	42
Reto 2	43
Reto 3	43
Proyecto	44

Sesión 1 - SQL

Código

```
#si no sabes que bases de datos contiene el documento
show databases;

#decir cual base de dato vas a utilizar
use tienda;

#ver las tablas
show tables;

#ver la estructura de la tabla
describe empleado;
#int son numeros enteros
#varchar() son textos en el parentesis se pone la longitud llega hasta 250 caracteres
#imágenes es mejor no guardarlos, mejor las ligas
```

Reto 1

1. Usando la base de datos tienda, muestra la descripción de las tablas articulo, puesto y venta. Por cada tipo de dato que encuentras llena la siguiente tabla, a mano. Usa la [Documentación de MySQL](#) como referencia.

```
describe articulo;
describe puesto;
describe venta;
13 • describe articulo;
14 • describe puesto;
15 • describe venta;
```

The screenshot shows a MySQL query results grid titled "Result Grid". It displays the structure of the "articulo" table. The columns are Field, Type, Null, Key, Default, and Extra. The rows show the following data:

Field	Type	Null	Key	Default	Extra
id_articulo	int	NO	PRI	NULL	
nombre	varchar(45)	NO		NULL	
precio	double	NO		NULL	
iva	double	NO		NULL	
cantidad	int	NO		0	

Result Grid | Filter Rows: Export: Wr

	Field	Type	Null	Key	Default	Extra
▶	id_articulo	int	NO	PRI	NULL	
	nombre	varchar(45)	NO		NULL	
	precio	double	NO		NULL	
	iva	double	NO		NULL	
	cantidad	int	NO		0	

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

	Field	Type	Null	Key	Default	Extra
▶	id_venta	int	NO	PRI	NULL	
	id_articulo	int	NO	MUL	NULL	
	id_empleado	int	NO	MUL	NULL	
	dave	varchar(45)	NO		NULL	
	fecha	timestamp	NO		CURRENT_TIMESTAMP	DEFAULT_GENERATED on update CURRENT_TIMESTAMP

#seleccionar datos de una tabla en especifico
 select nombre from empleado;

#para todos los campos de la tabla
 select * from empleado;

#se usa WHERE para filtrar
 #para que te de un resultado deseado
 select *
 from empleado
 where apellido_paterno = 'Risom';

#para que sea mayor a algo deseado
 select *
 from empleado
 where id_puesto > 100;

#se usa ORDER BY para ordenar
 SELECT *
 FROM empleado
 WHERE id_puesto >= 100
 AND id_puesto <= 200
 ORDER BY id_puesto ASC;

SELECT *
 FROM empleado
 WHERE id_puesto = 100
 OR id_puesto = 200;
 #otra forma de decirlo sería
 SELECT *
 FROM empleado
 WHERE id_puesto IN (100, 200, 300);

Reto 2

1. ¿Cuál es el nombre de los empleados con el puesto 4?

```
SELECT nombre  
FROM empleado  
WHERE id_puesto = 4;
```

Result Grid	
nombre	
Norrie	
Maxy	

2. ¿Qué puestos tienen un salario mayor a \$10,000?

```
SELECT *  
FROM puesto  
WHERE salario > 10000;
```

Result Grid	
id_puesto	nombre
1	Analog Circuit Design Manager
2	Junior Executive
3	Director of Sales
4	Researcher
5	Desktop Support Technician
6	Budget/Accounting Analyst III
7	Accounting Assistant III

Message
100 rows(s) returned

Action Output

Time	Action	Message
32 20:38:27	SELECT * FROM empleado WHERE id_puesto >= 100 AND id_puesto <= 200 ORDER BY id_puesto ASC	100 row(s) returned
33 20:40:19	SELECT * FROM empleado WHERE id_puesto = 100 OR id_puesto = 200	4 row(s) returned
34 20:42:17	SELECT * FROM empleado WHERE id_puesto IN (100, 200, 300)	6 row(s) returned
35 20:43:50	SELECT * FROM empleado	1000 row(s) returned
36 20:58:07	SELECT nombre FROM empleados WHERE id_puesto > 4	2 row(s) returned
37 20:59:19	SELECT * FROM puesto WHERE salario > 10000	1000 row(s) returned

3. ¿Qué artículos tienen un precio mayor a \$1,000 y un iva mayor a 100?

```
SELECT *  
FROM articulo  
WHERE precio > 1000  
AND iva > 100;
```

Result Grid	
id_articulo	nombre
2	Pasta - Angel Hair
3	Soup - French Onion
4	Wine - Valpolicella Med
5	Ham - Parma
7	Norwegian - Blueberry Oatmeal
8	Wine - Pinot Noir French Berries
9	Yogurt - Greek

Message
1000 row(s) returned

Action Output

Time	Action	Message
38 20:49:50	SELECT * FROM articulo	1000 row(s) returned
39 20:50:01	SELECT nombre FROM articulo WHERE id_articulo > 4	2 row(s) returned
40 20:50:19	SELECT * FROM articulo WHERE precio > 1000	1000 row(s) returned
41 20:50:22	SELECT * FROM articulo WHERE precio > 1000 AND iva > 100	Error Code: 1054. Unknown column 'precio' in 'where clause'
42 21:00:31	SELECT * FROM articulo WHERE precio > 1000 AND iva > 100	Error Code: 1054. Unknown column 'precio' in 'where clause'
43 21:01:12	SELECT * FROM articulo WHERE precio > 1000 AND iva > 100	70 row(s) returned

4. ¿Qué ventas incluyen los artículo 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);
```

Result Grid	
id_venta	id_articulo
7	963
6	135

Message
1000 row(s) returned

Action Output

Time	Action	Message
44 20:51:00	SELECT * FROM venta	1000 row(s) returned
45 20:51:12	SELECT * FROM venta WHERE id_articulo IN (135, 963)	2 row(s) returned
46 20:51:20	SELECT * FROM venta WHERE id_empleado IN (835, 369)	1000 row(s) returned
47 20:51:23	SELECT * FROM venta WHERE id_articulo IN (135, 963) AND id_empleado IN (835, 369)	Error Code: 1054. Unknown column 'id_articulo' in 'where clause'
48 20:51:31	SELECT * FROM venta WHERE id_articulo IN (135, 963) AND id_empleado IN (835, 369)	Error Code: 1054. Unknown column 'id_empleado' in 'where clause'
49 20:51:42	SELECT * FROM venta WHERE id_articulo IN (135, 963) AND id_empleado IN (835, 369)	7 row(s) returned

```
#para ordenar descendente
SELECT *
FROM puesto
ORDER BY salario DESC;

#para ordenar ascendente puede llevar o no ASC
```

```
SELECT *
FROM puesto
ORDER BY salario;
```

```
SELECT *
FROM puesto
ORDER BY nombre;
```

```
#es para limitar el numero de resultados
SELECT *
FROM empleado
LIMIT 5;
```

Reto 3

1. Usando la base de datos tienda, escribe una consulta que permita obtener el top 5 de puestos por salarios.

```
SELECT *
FROM puesto
ORDER BY salario DESC
LIMIT 5;
```

id_puesto	nombre	salario
494	Sales Representative	29996.58
18	Speech Pathologist	29967.17
487	Analog Circuit Design manager	29923.95
79	Junior Executive	29916.06
893	Technical Writer	29912.53
NULL	NULL	NULL

Proyecto

```
#proyecto
```

1. Dentro del mismo servidor de bases de datos, conéctate al esquema classicmodels.

```
USE classicmodels;
```

```
✓ 18 20:13:09 USE classicmodels 0 row(s) affected
```

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

```
SELECT lastName
FROM employees;
```

lastName
Murphy
Patterson
Firrelli
Patterson
Bondur
Bow
Jennings
Thompson
Firrelli
Patterson
Tseng
Vanauf
Bondur
Hernandez

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

SELECT lastName, firstName, jobTitle

FROM employees;

lastName	firstName	jobTitle
Murphy	Diane	President
Patterson	Mary	VP Sales
Firrelli	Jeff	VP Marketing
Patterson	William	Sales Manager (APAC)
Bondur	Gerard	Sale Manager (EMEA)
Bow	Anthony	Sales Manager (NA)
Jennings	Leslie	Sales Rep
Thompson	Leslie	Sales Rep
Firrelli	Julie	Sales Rep
Patterson	Steve	Sales Rep
Tseng	Foon Yue	Sales Rep
Vanauf	George	Sales Rep
Bondur	Loui	Sales Rep
Hernandez	Gerard	Sales Rep

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

SELECT *

FROM employees;

employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	jobTitle
1002	Murphy	Diane	x5800	dmurphy@classicmodelcars.com	1	NULL	President
1056	Patterson	Mary	x4611	mpatterso@classicmodelcars.com	1	1002	VP Sales
1076	Firrelli	Jeff	x9273	jfirrelli@classicmodelcars.com	1	1002	VP Marketing
1088	Patterson	William	x4671	wpaterson@classicmodelcars.com	6	1056	Sales Manager (APAC)
1102	Bondur	Gerard	x5408	gbondur@classicmodelcars.com	4	1056	Sale Manager (EMEA)
1143	Bow	Anthony	x5428	abow@classicmodelcars.com	1	1056	Sales Manager (NA)
1165	Jennings	Leslie	x3291	ljennings@classicmodelcars.com	1	1143	Sales Rep
1166	Thompson	Leslie	x4065	lthompson@classicmodelcars.com	1	1143	Sales Rep
1188	Firrelli	Julie	x2173	jfirrelli@classicmodelcars.com	2	1143	Sales Rep
1216	Patterson	Steve	x4334	spatterson@classicmodelcars.com	2	1143	Sales Rep
1286	Tseng	Foon Yue	x2248	ftsing@classicmodelcars.com	3	1143	Sales Rep
1323	Vanauf	George	x4102	gvanauf@classicmodelcars.com	3	1143	Sales Rep
1337	Bondur	Loui	x6493	lbondur@classicmodelcars.com	4	1102	Sales Rep
1370	Hernandez	Gerard	x2028	ghernande@classicmodelcars.com	4	1102	Sales Rep

5. 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';

lastName	firstName	jobTitle
Jennings	Leslie	Sales Rep
Thompson	Leslie	Sales Rep
Firrelli	Julie	Sales Rep
Patterson	Steve	Sales Rep
Tseng	Foon Yue	Sales Rep
Vanauf	George	Sales Rep
Bondur	Loui	Sales Rep
Hernandez	Gerard	Sales Rep
Castillo	Pamela	Sales Rep
Bott	Larry	Sales Rep
Jones	Barry	Sales Rep
Fixter	Andy	Sales Rep
Marsh	Peter	Sales Rep
King	Tom	Sales Rep

- 6. 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;
```

	lastName	firstName	jobTitle	officeCode
▶	Jennings	Leslie	Sales Rep	1
	Thompson	Leslie	Sales Rep	1

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

```
SELECT lastName, firstName, jobTitle, officeCode
FROM employees
WHERE jobTitle = 'Sales Rep'
    OR officeCode = 1;
```

	lastName	firstName	jobTitle	officeCode
▶	Murphy	Diane	President	1
	Patterson	Mary	VP Sales	1
	Firrelli	Jeff	VP Marketing	1
	Bow	Anthony	Sales Manager (NA)	1
	Jennings	Leslie	Sales Rep	1
	Thompson	Leslie	Sales Rep	1
	Firrelli	Julie	Sales Rep	2
	Patterson	Steve	Sales Rep	2
	Tseng	Foon Yue	Sales Rep	3
	Vanlauf	George	Sales Rep	3
	Bondur	Lou	Sales Rep	4
	Hernandez	Gerard	Sales Rep	4
	Castillo	Pamela	Sales Rep	4
	Bott	Larry	Sales Rep	7

- 8. 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);
```

	lastName	firstName	officeCode
▶	Murphy	Diane	1
	Patterson	Mary	1
	Firrelli	Jeff	1
	Bow	Anthony	1
	Jennings	Leslie	1
	Thompson	Leslie	1
	Firrelli	Julie	2
	Patterson	Steve	2
	Tseng	Foon Yue	3
	Vanlauf	George	3

- 9. 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';
```

	lastName	firstName	jobTitle
▶	Murphy	Diane	President
	Patterson	Mary	VP Sales
	Firrelli	Jeff	VP Marketing
	Patterson	William	Sales Manager (APAC)
	Bondur	Gerard	Sale Manager (EMEA)
	Bow	Anthony	Sales Manager (NA)

10. 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;
```

lastName	firstName	officeCode
Patterson	William	6
Botti	Larry	7
Jones	Barry	7
Fixter	Andy	6
Marsh	Peter	6
King	Tom	6

11. 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 a 4.

```
SELECT lastName, firstName, officeCode  
FROM employees  
WHERE officeCode <= 4;
```

lastName	firstName	officeCode
Murphy	Diane	1
Patterson	Mary	1
Firelli	Jeff	1
Bondu	Gerard	4
Bow	Anthony	1
Jennings	Leslie	1
Thompson	Leslie	1
Firelli	Julie	2
Patterson	Steve	2
Tseng	Foon Yue	3
Vanau	George	3
Bondu	Loui	4
Hernandez	Gerard	4
Castillo	Pamela	4

12. 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';
```

customerName	country	state
Mini Gifts Distributors Ltd.	USA	CA
Mini Wheels Co.	USA	CA
Technics Stores Inc.	USA	CA
Toys'N'GrownUps.com	USA	CA
Boards & Toys Co.	USA	CA
Collectable Mini Designs Co.	USA	CA
Corporate Gift Ideas Co.	USA	CA
Men 'R' US Retailers, Ltd.	USA	CA
The Sharp Gifts Warehouse	USA	CA
West Coast Collectables Co.	USA	CA
Signal Collectibles Ltd.	USA	CA

13. 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;
```

customerName	country	state	creditLimit
Mini Gifts Distributors Ltd.	USA	CA	210500.00
Collectable Mini Designs Co.	USA	CA	105000.00
Corporate Gift Ideas Co.	USA	CA	105000.00

14. 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');
```

customerName	country
Atelier graphique	France
Signal Gift Stores	USA
La Rochele Gifts	France
Mini Gifts Distributors Ltd.	USA
Mini Wheels Co.	USA
Land of Toys Inc.	USA
Savely & Henriot, Co.	France
Muscle Machine Inc	USA
Diecast Classics Inc.	USA
Technics Stores Inc.	USA
American Souvenirs Inc	USA
Daedalus Designs Imports	France
La Corne D'abondance, Co.	France
Cambridge Collectables Co.	USA

15. 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, state, creditLimit  
FROM customers  
WHERE country IN ('USA', 'France')  
AND creditLimit > 100000;
```

customerName	country	state	creditLimit
Mini Gifts Distributors Ltd.	USA	CA	210500.00
Collectable Mini Designs Co.	USA	CA	105000.00
Corporate Gift Ideas Co.	USA	CA	105000.00

16. 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  
WHERE country IN ('USA', 'France');
```

officeCode	city	phone	country
1	San Francisco	+1 650 219 4782	USA
2	Boston	+1 215 837 0825	USA
3	NYC	+1 212 555 3000	USA
4	Paris	+33 14 723 4404	France
NULL	NULL	NULL	NULL

17. 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 NOT IN ('USA', 'France');
```

officeCode	city	phone	country
5	Tokyo	+81 33 224 5000	Japan
6	Sydney	+61 2 9264 2451	Australia
7	London	+44 20 7877 2041	UK
NULL	NULL	NULL	NULL

18. 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  
WHERE orderNumber IN (10165, 10287, 10310);
```

orderNumber	customerNumber	status	shippedDate
10165	148	Shipped	2003-12-26
10287	298	Shipped	2004-09-01
10310	259	Shipped	2004-10-18
NULL	NULL	NULL	NULL

19. 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;
```

contactLastName	contactFirstName
Acconti	Pedro
Alagger,G M	Raman
Andersen	Mel
Anton	Carmen
Ashworth	Rachel
Barajas	Miguel
Banize	Violeta
Bennett	
Berglund	Helen
Bergulfsen	Christina
Bertrand	Jonas
Brown	Marie
Brown	Julie
Brown	Ann
Brown	William

20. 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;
```

contactLastName	contactFirstName
Young	Jeff
Young	Julie
Young	Mary
Young	Dorothy
Yodzido	Juri
Walker	Brydey
Victorino	Wendy
Urs	Braun
Tseng	Jerry
Tonini	Daniel
Thompson	Valarie
Thompson	Steve
Taylor	Leslie
Taylor	Sue

21. 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;
```

contactLastName	contactFirstName
Young	Dorothy
Young	Jeff
Young	Julie
Young	Mary
Young	Juri
Walker	Brydey
Victorino	Wendy
Urs	Braun
Tseng	Jerry
Tonini	Daniel
Thompson	Steve
Thompson	Valarie
Taylor	Leslie
Taylor	Sue

22. 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;
```

	customerNumber	customerName	creditLimit
▶	141	Euro+ Shopping Channel	227600.00
	124	Mini Gifts Distributors Ltd.	210500.00
	298	Vida Sport, Ltd	141300.00
	151	Muscle Machine Inc	138500.00
	187	AV Stores, Co.	136800.00
✳	NULL	NULL	NULL

23. 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;
```

	customerNumber	customerName	creditLimit
▶	223	Naturalich Auto	0.00
	168	American Souvenirs Inc	0.00
	169	Porto Imports Co.	0.00
	206	Asian Shopping Network, Co	0.00
	125	Havel & Zbyszek Co	0.00
✳	NULL	NULL	NULL

Sesión 2

Código

```
#empieza con M  
SELECT *  
FROM empleado  
WHERE nombre LIKE 'M%';
```

```
#termina con a  
SELECT *  
FROM empleado  
WHERE nombre LIKE '%A';
```

```
#empeiza con m y termina con a  
SELECT *  
FROM empleado  
WHERE nombre LIKE 'M%A';
```

```
#_ solo busca una letra
SELECT *
FROM empleado
WHERE nombre LIKE 'M_losa';
```

```
SELECT *
FROM empleado
WHERE nombre LIKE 'klara';
```

Reto 1

1. ¿Qué artículos incluyen la palabra Pasta en su nombre?

```
SELECT *
FROM articulo
WHERE nombre LIKE '%Pasta%';
```

id_articulo	nombre	precio	iva	cantidad
2	Pasta - Angel Hair	4391.73	959.51	503
27	Pasta - Elbows, Macaroni, Dry	3668.7	253.66	392
70	Pasta - Shells, Medium, Dry	801.74	773.8	206
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

2. ¿Qué artículos incluyen la palabra Cannelloni en su nombre?

```
SELECT *
FROM articulo
WHERE nombre LIKE '%Cannelloni%';
```

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

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

```
SELECT *
FROM articulo
WHERE nombre LIKE '%-%';
```

id_articulo	nombre	precio	iva	cantidad
1	Chocolate - Peathers	2729.93	12.26	144
2	Pasta - Angel Hair	4391.73	959.51	503
3	Soup Campbell's - Tomato Bisque	2991.35	587.59	604
4	Wine - Valpolicella Msd	2625.2	770.1	575
5	Mousse - Banana Chocolate	3701.62	893.46	248
6	Yeast,Dry - Feschner	923.48	524.08	918

Todos los resultados se han devuelto.

Última actualización: 10/03/2018 11:20:00

Action	Output
11. 20/03/03 11:20:00:00	SELECT * FROM articulo WHERE nombre LIKE "%Pasta%"
	17 row(s) returned
12. 20/03/06 11:20:00:00	SELECT * FROM articulo WHERE nombre LIKE "%Cannelloni%"
	0 row(s) returned
13. 20/03/19 11:20:00:00	SELECT * FROM articulo WHERE nombre LIKE "%-%"
	1 row(s) returned
14. 20/03/20 11:20:00:00	SELECT * FROM articulo WHERE nombre LIKE "%-%"
	76 row(s) returned

#agrupamiento

```
SELECT (1 + 7) * (10/ (6 -4));
```

```
SELECT AVG(a.precio) AS precio_promedio
```

```
FROM articulo a;
```

```
SELECT COUNT(*)
FROM empleado
WHERE nombre <> 'Klara';
```

```
SELECT MAX(precio)
FROM articulo;
```

```
SELECT MIN(precio)
FROM articulo;
```

```
SELECT SUM(precio)
FROM articulo;
```

Reto 2

1. ¿Cuál es el promedio de salario de los puestos?

```
SELECT AVG(salario)
FROM puesto;
```

AVG(salario)
19595.051179999973

2. ¿Cuántos artículos incluyen la palabra Pasta en su nombre?

```
SELECT COUNT(nombre)
FROM articulo
WHERE nombre LIKE '%pasta%';
```

COUNT(nombre)
17

3. ¿Cuál es el salario mínimo y máximo?

```
SELECT MAX(salario) AS salario_maximo, MIN(salario) AS salario_minimo
FROM puesto;
```

salario_maximo	salario_minimo
29996.58	10013.44

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

```
SELECT MAX(id_puesto) - 5
FROM puesto;
```

```
SELECT SUM(salario)
FROM puesto
WHERE id_puesto >= 995;
```

```
    SUM(salario)
    98919.69
```

#otra opcion

```
SELECT SUM(salario)
FROM puesto
WHERE id_puesto >= (SELECT MAX(id_puesto) - 5
FROM puesto);
```

```
SELECT nombre, SUM(precio) AS total
FROM articulo
GROUP BY nombre;
```

```
SELECT *
FROM articulo;
```

```
SELECT nombre, MIN(salario) AS menor, MAX(salario) AS maximo
FROM puesto
GROUP BY nombre;
```

```
SELECT nombre, COUNT(*) AS cantidad
FROM articulo
GROUP BY nombre
ORDER BY cantidad DESC;
```

Reto 3

1. ¿Cuántos registros hay por cada uno de los puestos?

```
SELECT nombre, COUNT(*)
FROM puesto
GROUP BY nombre;
```



nombre	COUNT(*)
Analyst Circuit Design manager	0
Junior Executive	8
Executive Leader	4
Staff Scientist	9
Dealership Support Technician	5
Marketing Quality Analyst (II)	4
Accounting Accountant (II)	3
Analyst	1
Analyst	1
Analyst	1

2. ¿Cuánto dinero se paga en total por puesto?

```
SELECT nombre, SUM(salario)
FROM puesto
GROUP BY nombre;
```

nombre	SUM(salario)
Analog Circuit Design manager	179310.18000000002
Junior Executive	156846.26
Director of Sales	136630.69
Staff Scientist	157528.98
Desktop Support Technician	92315.22
Budget/Accounting Analyst III	70107.77
Accounting Assistant III	78947.08

result 30 ×

Output:

Action Output

Time Action Message

37 20:56:06 SELECT nombre, MIN(salario) AS menor, MAX(salario) AS maximo FROM articulo GROUP BY... Error Code: 1054. Unknown column 'articulo' in 'group by'

38 20:56:24 SELECT nombre, MIN(salario) AS menor, MAX(salario) AS maximo FROM puesto GROUP BY... 181 row(s) returned

39 20:57:28 SELECT nombre, COUNT(*) FROM puesto GROUP BY nombre 181 row(s) returned

40 20:58:30 SELECT nombre, SUM(salario) FROM puesto GROUP BY nombre 181 row(s) returned

3. ¿Cuál es el número total de ventas por vendedor?

```
SELECT id_empleado, COUNT(*)
FROM venta
GROUP BY id_empleado;
```

id_empleado	COUNT(*)
2	2
3	1
4	1
5	1
6	2
12	5
15	4

result 31 ×

Output:

Action Output

Time Action Message

38 20:57:24 SELECT nombre, MIN(salario) AS menor, MAX(salario) AS maximo FROM puesto GROUP BY... 181 row(s) returned

39 20:57:28 SELECT nombre, COUNT(*) FROM puesto GROUP BY nombre 101 row(s) returned

40 20:58:30 SELECT nombre, SUM(salario) FROM puesto GROUP BY nombre 101 row(s) returned

41 21:00:55 SELECT id_empleado, COUNT(*) FROM venta GROUP BY id_empleado 642 row(s) returned

4. ¿Cuál es el número total de ventas por artículo?

```
SELECT id_articulo, COUNT(*)
FROM venta
GROUP BY id_articulo;
```

id_articulo	COUNT(*)
2	1
3	1
4	2
8	1
10	1
11	1
12	1

result 32 ×

Output:

Action Output

Time Action Message

40 20:58:30 SELECT nombre, MIN(salario) FROM puesto GROUP BY nombre 181 row(s) returned

41 21:00:55 SELECT id_empleado, COUNT(*) FROM venta GROUP BY id_empleado 642 row(s) returned

42 21:03:14 SELECT id_venta, COUNT(*) FROM venta GROUP BY id_articulo Error Code: 1055. Expression #1 of SELECT list contains an invalid expression

43 21:03:36 SELECT id_articulo, COUNT(*) FROM venta GROUP BY id_articulo 618 row(s) returned

```
SELECT *
FROM empleado where id_puesto IN (SELECT id_puesto
FROM puesto
WHERE nombre = 'Junior Executive');
```

```
SELECT clave, id_articulo, COUNT(*) AS cantidad
FROM venta
GROUP BY clave, id_articulo
ORDER BY clave;
```

```
SELECT id_articulo, MIN(cantidad), MAX(cantidad)
FROM (SELECT clave, id_articulo, COUNT(*) AS cantidad
FROM venta
GROUP BY clave, id_articulo
ORDER BY clave) AS subconsulta
GROUP BY id_articulo;
```

Reto 4

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

```
SELECT nombre, apellido_paterno, apellido_materno
FROM empleado
WHERE id_puesto IN (SELECT id_puesto
FROM puesto
WHERE salario < 20000);
```

nombre	apellido_paterno	apellido_materno
Normie	McGarrie	Hartopp
Maxxy	Udden	Kose
Della	Fulbrook	Harris
Katya	Banbridge	Fossett
Robyn	Hancock	Leibold
Hayyim	Verdon	Eastcott
Analiye	Beteriss	Tant

empleado 39 x

Output

Action Output

#	Time	Action	Message
47	21:19:13	SELECT id_articulo, MIN(cantidad), MAX(cantidad) FROM (SELECT clave, id_articulo, COUNT(*) AS cantidad FROM venta GROUP BY id_articulo ORDER BY cantidad DESC LIMIT 1) AS articulo	168 row(s) returned
48	21:23:40	SELECT nombre, apellido_paterno, apellido_materno FROM empleado WHERE id_puesto IN (SELECT id_puesto FROM puesto WHERE salario < 20000)	1000 row(s) returned
49	21:27:30	SELECT id_empleado, COUNT(*) AS cantidad FROM venta GROUP BY id_empleado ORDER BY cantidad DESC LIMIT 1	642 row(s) returned
50	21:28:50	SELECT nombre, apellido_paterno, apellido_materno FROM empleado WHERE id_puesto IN (SELECT id_puesto FROM puesto WHERE salario < 20000)	524 row(s) returned

2. ¿Cuál es la cantidad mínima y máxima de ventas de cada empleado?

```
SELECT id_empleado, MAX(cantidad), MIN(cantidad)
FROM (SELECT id_empleado, COUNT(*) AS cantidad
FROM venta
GROUP BY id_empleado, clave
ORDER BY clave) AS subconsulta
GROUP BY id_empleado;
```

id_empleado	MAX(cantidad)	MIN(cantidad)
228	1	1
447	1	1
577	1	1
390	1	1
657	1	1
838	1	1
586	1	1

Result 40 x

Output

Action Output

#	Time	Action	Message
48	21:23:40	SELECT nombre, apellido_paterno, apellido_materno FROM empleado WHERE id_puesto IN (SELECT id_puesto FROM puesto WHERE salario < 20000)	1000 row(s) returned
49	21:27:30	SELECT id_empleado, COUNT(*) AS cantidad FROM venta GROUP BY id_empleado ORDER BY cantidad DESC LIMIT 1	642 row(s) returned
50	21:28:50	SELECT nombre, apellido_paterno, apellido_materno FROM empleado WHERE id_puesto IN (SELECT id_puesto FROM puesto WHERE salario < 20000)	524 row(s) returned
51	21:30:03	SELECT id_empleado, MAX(cantidad), MIN(cantidad) FROM (SELECT id_empleado, COUNT(*) AS cantidad FROM venta GROUP BY id_empleado, clave ORDER BY clave) AS subconsulta GROUP BY id_empleado	642 row(s) returned

3. ¿Cuál es el nombre del puesto de cada empleado?

```
SELECT nombre, apellido_paterno, apellido_materno, (SELECT nombre
FROM puesto
WHERE id_puesto = e.id_puesto) AS Puesto
FROM empleado AS e;
```

nombre	apellido_paterno	apellido_materno	puesto
Enrichtta	Bodechon	Ivkovic	Product Engineer
Morey	Bowskill	Metham	Budget/Accounting Analyst IV
Jeanette	Potes	Heisler	Occupational Therapist
Casssey	Womersley	Chapell	Financial Advisor
Gnri	Risom	Kalinowsky	Physical Therapy Assistant
Lisle	Carlsson	Marquot	Marketing Assistant
Andre	Theurer	Craighill	Tax Accountant

Output:

```

Action Output
# Time Action Message
> 51 21:30:03 SELECT id_empleado, MAX(cantidad), MIN(cantidad) FROM (SELECT id_empleado, COUNT(*) AS cantidad FROM employees GROUP BY id_empleado ORDER BY cantidad DESC LIMIT 1) t1, (SELECT id_puesto, COUNT(*) AS cantidad FROM employees GROUP BY id_puesto ORDER BY cantidad DESC LIMIT 1) t2 WHERE t1.id_empleado = t2.id_puesto
> 52 21:35:55 SELECT nombre, apellido_paterno, apellido_materno, FROM (SELECT id_puesto FROM puesto ORDER BY id_puesto DESC LIMIT 1) t1, (SELECT nombre FROM employees WHERE id_empleado = t1.id_empleado) t2 WHERE t1.id_puesto = t2.id_puesto
> 53 21:40:13 SELECT nombre, apellido_paterno, apellido_materno, (SELECT nombre FROM puesto WHERE id_puesto = t1.id_puesto) AS puesto FROM (SELECT id_empleado, MAX(cantidad), MIN(cantidad) FROM (SELECT id_empleado, COUNT(*) AS cantidad FROM employees GROUP BY id_empleado ORDER BY cantidad DESC LIMIT 1) t1, (SELECT id_puesto, COUNT(*) AS cantidad FROM employees GROUP BY id_puesto ORDER BY cantidad DESC LIMIT 1) t2 WHERE t1.id_empleado = t2.id_puesto) t3, (SELECT nombre FROM employees WHERE id_empleado = t3.id_empleado) t4 WHERE t3.id_puesto = t4.id_puesto
> 54 21:40:50 SELECT nombre, apellido_paterno, apellido_materno, (SELECT nombre FROM puesto WHERE id_puesto = t1.id_puesto) AS puesto FROM (SELECT id_empleado, MAX(cantidad), MIN(cantidad) FROM (SELECT id_empleado, COUNT(*) AS cantidad FROM employees GROUP BY id_empleado ORDER BY cantidad DESC LIMIT 1) t1, (SELECT id_puesto, COUNT(*) AS cantidad FROM employees GROUP BY id_puesto ORDER BY cantidad DESC LIMIT 1) t2 WHERE t1.id_empleado = t2.id_puesto) t3, (SELECT nombre FROM employees WHERE id_empleado = t3.id_empleado) t4 WHERE t3.id_puesto = t4.id_puesto

```

Proyecto

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

```
SELECT firstName, lastName, employeeNumber
```

```
FROM employees
```

```
WHERE firstName LIKE 'A%';
```

firstName	lastName	employeeNumber
Anthony	Bow	1143
Andy	Fixter	1611
*	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 firstName, lastName, employeeNumber
```

```
FROM employees
```

```
WHERE firstName LIKE '%on';
```

firstName	lastName	employeeNumber
*	NULL	NULL
NULL	NULL	NULL

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

```
SELECT firstName, lastName, employeeNumber
```

```
FROM employees
```

```
WHERE firstName LIKE '%on%';
```

firstname	lastname	employeeNumber
Anthony	Bow	1143
Foon	Yue	1286
*	NULL	NULL

- 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 firstName, lastName, employeeNumber
```

```
FROM employees
```

WHERE firstName LIKE 'T_M';

	firstName	lastName	employeeNumber
▶	Tom	King	1619
*	NULL	NULL	NULL

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

```
SELECT firstName, lastName, employeeNumber  
FROM employees  
WHERE firstName NOT LIKE 'B%';
```

	firstName	lastName	employeeNumber
▶	Dane	Murphy	1002
	Mary	Patterson	1056
	Jeff	Smith	1070
	William	Patterson	1089
	Gerard	Bondu	1102
	Anthony	Bow	1143
	Ledie	Wingerts	1150
	Leslie	Thompson	1166

			employees 13 x

Output

Action Output

* Time Action Message

74 09:13:14 SELECT firstName, lastName, employeeNumber FROM employees WHERE firstName LIKE '...' 0 row(s) returned

75 09:13:41 SELECT firstName, lastName, employeeNumber FROM employees WHERE firstName NOT ... 22 row(s) returned

6. 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 Metro G-Lutz 1100
	S18_3320	1917 Maxwell Touring Car
	S24_2000	1960 BSA Gold Star DB614
	S24_2001	1960 BSA Gold Star
	S24_2022	1938 Cadillac V-16 Presidential Limousine
	S24_3420	1937 Horch 500V Limousine
	S24_4620	1961 Chevrolet Impala
	S32_2208	1980 Ducati 996 R

	products 19 x	

Output

Action Output

* Time Action Message

75 09:13:41 SELECT firstName, lastName, employeeNumber FROM employees WHERE firstName NOT ... 22 row(s) returned

76 09:14:01 SELECT productCode, productName FROM products WHERE productCode LIKE '%_20%' 10 row(s) returned

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

```
SELECT orderNumber, SUM(quantityOrdered * priceEach) AS total  
FROM orderdetails  
GROUP BY orderNumber;
```

	orderNumber	total
▶	O1000	6552.00
	O1001	5476.00
	O1002	5476.00
	O1003	5476.00
	O1004	5476.00
	O1005	5476.00
	O1006	5476.00
	O1007	22884.00
	O1008	22884.00
	O1009	22884.00
	O1010	22884.00
	orderNumber 10 x	

Output

Action Output

* Time Action Message

108 11:40:43 SELECT orderNumber, quantityOrdered * priceEach AS total FROM orderdetails 200 rows selected

108 11:40:43 SELECT orderNumber, SUM(quantityOrdered * priceEach) AS total FROM orderdetails GROUP BY orderNumber 10 rows selected

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

```
SELECT EXTRACT(YEAR FROM orderDate) AS año, COUNT(*) AS ordenes  
FROM orders  
GROUP BY año;
```

	año	ordenes
▶	2003	111
	2004	151
	2005	64

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

```
SELECT firstName, lastName, officeCode  
FROM employees  
WHERE officeCode IN (SELECT officeCode  
FROM offices  
WHERE country LIKE 'USA');
```

firstName	lastName	officeCode
Jeff	Pineill	1
Audrey	Pinell	1
Leslie	Jennings	1
Leddie	Thompson	1
Jade		2
Steve	Peterson	2
Foon Yue	Tsing	3
George	Yanouf	3

employees 39 x

Output

Action Details

Time Action Message

169 15:20:00 SELECT firstName, lastName, (SELECT... Error Code: 1242. Subquery returns more than 1 row

166 15:24:06 SELECT firstName, lastName, officeCod... 10 rows(1) returned

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

```
SELECT customerNumber, checkNumber, amount  
FROM payments  
ORDER BY amount DESC  
LIMIT 1;
```

customerNumber	checkNumber	amount
141	JE105477	120166.58
NULL	NULL	NULL

11. Obtén 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) AS promedio  
FROM payments);
```

customerNumber	checkNumber	amount
224	C4510306	45783.54
221	Q3500001	45559.79
214	LC240773	45332.47
124	HT141746	45094.26
144	HT202743	45074.24
181	CH957219	44400.00
149	CH202743	44380.29
166	DC970307	44460.92
174	PT114444	44460.92

payments 33 x

Output

Action Details

Time Action Message

166 15:10:16 SELECT AVG(amount) AS promedio FR... 1 rows(1) returned

157 15:10:16 SELECT customerNumber, checkNum... 134 rows(1) returned

12. Obtén el nombre de aquellos clientes que no han hecho ninguna orden.

```
SELECT customerName  
FROM customers  
WHERE customerNumber NOT IN (SELECT customerNumber  
FROM payments);
```

13. Obtén el máximo, mínimo y promedio del número de productos en las órdenes de venta.

```
SELECT orderNumber, MAX(quantityOrdered) AS maximo, MIN(quantityOrdered) AS minimo, AVG(quantityOrdered) AS promedio  
FROM orderdetails  
GROUP BY orderNumber;
```

orderNumber	maximo	minimo	promedio
10100	50	22	37.7500
10101	46	25	35.5000
10102	41	20	32.8000
10103	46	22	33.8125
10104	49	23	34.9769
10105	50	22	36.5333
10106	58	26	37.5000
10107	39	20	28.6250
...

14. Dentro de la tabla orders, obtén el número de órdenes que hay por cada estado.

```
SELECT status, COUNT(*)  
FROM orders  
GROUP BY status;
```

status	COUNT(*)
Shipped	303
Resolved	4
Cancelled	6
On Hold	4
Disputed	3
In Process	6

Sesión 3

Código

```
USE AVH;
```

```
#tambien puede ser AVH.movies y asi no pongo USE AVH;  
SELECT *  
FROM movies;
```

```
USE tienda;
```

```
SHOW KEYS  
FROM venta;
```

```
SELECT *  
FROM empleado AS e  
JOIN puesto AS p  
ON e.id_puesto = p.id_puesto;
```

#primer tabla es del FROM es el izquierdo y la segunda tabla es la derecha donde esta el JOIN

```
SELECT *
FROM puesto AS p
LEFT JOIN empleado AS e
    ON p.id_puesto = e.id_puesto;
```

```
SELECT *
FROM empleado AS e
RIGHT JOIN puesto AS p
    ON e.id_puesto = p.id_puesto;
```

Reto 1

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

```
SELECT clave, nombre, apellido_paterno, apellido_materno
FROM venta AS v
JOIN empleado AS e
    ON v.id_empleado = e.id_empleado
ORDER BY clave;
```

clave	nombre	apellido_paterno	apellido_materno
0002-8149	Lorentzos	Coley	Foucher
0002-8149	Ede	Billn	Waddell
0002-8149	Colter	Hedner	
0002-8149	Sydney	Hoyley	Jaggi
0002-8149	Agustín	Ríchez	Sturge
0002-8149	Sigrid	Teal	Pleasim
0002-8149	Rebecca	Rutherford	Sorrell
0002-8149	Quiana	Szalotis	
0002-8149	Luse	Lennard	Boanght
0002-8149	Gustie	Gryglewski	Gidley
0002-8149	Leanne	Hartman	Korson
0002-8149	Dotti	Mc Kellen	Dylke
0008-0833	Tan	Glossoco	HemFleet

Output

Action Output

Time Action Message

21 20:05:15 SELECT clave, nombre, apellido_paterno, apellido_materno ... Error Code: 1055. Expression #2 of SEL

22 20:05:39 SELECT clave, nombre, apellido_paterno, apellido_materno ... 1000 rows returned

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

```
SELECT clave, nombre
FROM venta AS v
JOIN articulo AS a
    ON v.id_articulo = a.id_articulo
ORDER BY clave;
```

clave	nombre
0002-8149	Sour Puss - Pea
0002-8149	Wine - Vint - Ing - 2002, Sauv
0002-8149	White Tomato Purée
0002-8149	Wine - Merlot - Roca Grana
0002-8149	Yakamein
0002-8149	Yakamein - Germmer, Peter, D.o.c.
0002-8149	Glace - Clear
0002-8149	Beans - Green
0002-8149	Go
0002-8149	Butter Sweet
0002-8149	Sour Puss - Tangerine
0002-8149	Ice Cream Bar - Chocolate
0002-8149	Juice - Orange - 1.89
0008-0833	Ice Cream Bar - Orange

Output

Action Output

22 20:05:39 SELECT clave, nombre, apellido_paterno, apellido_materno ... 1000 rows returned

23 20:08:00 SELECT clave, nombre FROM venta AS v JOIN articulo ... 1000 rows returned

3. ¿Cuál es el total de cada venta?

```
SELECT clave, SUM(precio) AS total
FROM venta AS v
JOIN articulo AS a
    ON v.id_articulo = a.id_articulo
GROUP BY clave
ORDER BY clave;
```

```
CREATE VIEW AVTickets AS  
(SELECT v.clave, v.fecha, a.nombre AS producto, a.precio,  
concat(e.nombre, ' ', e.apellido_paterno) AS empleado  
FROM venta AS v  
JOIN empleado AS e  
    ON v.id_empleado = e.id_empleado  
JOIN articulo AS a  
    ON v.id_articulo = a.id_articulo);
```

CREATE VIEW AVTickets AS

```
(SELECT v.clave, v.fecha, a.nombre AS producto, a.precio,  
concat(e.nombre, ' ', e.apellido_paterno) AS empleado  
FROM venta AS v  
JOIN empleado AS e  
    ON v.id_empleado = e.id_empleado  
JOIN articulo AS a  
    ON v.id_articulo = a.id_articulo);
```

Reto 2

1. Obtener el puesto de un empleado.

```
CREATE VIEW AVpuesto AS  
(SELECT p.nombre AS puesto, concat(e.nombre, ' ', e.apellido_paterno,'  
' ,e.apellido_materno) AS empleado  
FROM empleado AS e  
JOIN puesto AS p  
    ON e.id_puesto = p.id_puesto);
```

```
SELECT *  
FROM AVpuesto;
```

```
CREATE VIEW AVpuesto AS  
(SELECT p.nombre AS puesto, concat(e.nombre, ' ', e.apellido_paterno,'  
' ,e.apellido_materno) AS empleado  
FROM empleado AS e  
JOIN puesto AS p  
    ON e.id_puesto = p.id_puesto);
```

2. Saber qué artículos ha vendido cada empleado.

```
CREATE VIEW AVarticulosxempleado AS  
(SELECT v.clave, concat(e.nombre, ' ', e.apellido_paterno, ' ', e.apellido_materno) AS  
nombre, a.nombre AS articulo  
FROM venta AS v  
JOIN empleado AS e  
    ON v.id_empleado = e.id_empleado  
JOIN articulo AS a  
    ON v.id_articulo = a.id_articulo);
```

```
SELECT *  
FROM AVarticulosxempleado;
```

Spec	Product	Detail
00024149	Leontine Jankinson Briffield	Sprouts - Peas
00024149	Leslie Culvey Poucher	Wine - Saint - Brie 2002, Sauv.
00024149	Edie Bill Viadodd	Sauce Tomato Pouch
00024149	Wendy Gandy Parker	Wine - Chardonnay Royal Crémant
00024149	Sydney Wodway Jogg	Yakult
00024149	Aquilina Richards Surge	Marsala - Spumone, Fine, D...
00024149	Elspeth Gandy Parker	Ginger
00024149	Rebecca Ruthven Soll	Beans - Green
00024149	Nora C'Suean MacAskill	Winter, Tap
00024149	Elspeth Gandy Parker	Bread - Sweet
00024149	Gustie Orygvaldi Gilley	Sour Puss - Tangerine
00024149	Pete Skedgel Vassar	Bread - White, Sliced
00024149	Elspeth Gandy Parker	Juice - Orange 1.5L

1 rows(s) returned

Action	Output
Time	Action
55 21:12:34	Apply changes to Avventurasempleados
56 21:12:48	SELECT * FROM Avventurasempleados

3. Saber qué puesto ha tenido más ventas.

CREATE VIEW AVventas AS

(SELECT p.nombre AS puesto, COUNT(v.clave) AS ventas

FROM venta AS v

JOIN empleado AS e

ON v.id_empleado = e.id_empleado

JOIN puesto AS p

ON e.id_puesto = p.id_puesto

GROUP BY p.nombre);

```
SELECT *
FROM AVventas
ORDER BY ventas DESC
LIMIT 1;
```

puesto	ventas
Physical Therapy Assistant	23

Proyecto

Para estas consultas usa **RIGHT JOIN**

- Obtén el código de producto, nombre de producto y descripción de todos los productos.

```
SELECT productCode, productName, productDescription
FROM products;
```

productCode	productName	productDescription
S10_1678	1969 Harley Davidson Ultimate Chopper	This remarkable touring motorcycle has... The rear wheel is touring function detail... Official front wheel steering function detail...
S10_1910	1969 Harley Davidson 1300	Official front wheel steering function detail...
S10_2016	1996 Moto Guzzi 1300	Official Moto Guzzi logo and insignia, saddle b...
S10_4698	2003 Harley-Davidson Eagle Dragster	Model features, official Harley-Davidson logos a...
S10_4757	1972 Alfa Romeo GTA	Features include: Turnable front wheels; steerin...
S10_4900	1968 Ford Mustang GT	Power steering, turnable front wheels; steering f...
S12_1099	1968 Ford Mustang	Hood, doors and trunk all open to reveal highly ... Turnable front wheels; steering function detail...
S12_1108	2001 Ferrari Enzo	Hood, doors and trunk all open to reveal highly ... Turnable front wheels; steering function detail...
S12_1400	1969 Dodge Charger	Official logo and insignia, saddle bags included ...
S12_2823	2002 Suzuki VX800	Official logo and insignia, saddle bags included ...
S12_3148	1969 Cervar Monza	1:18 scale die-cast about 10" long doors open ...
S12_3380	1968 Dodge Charger	1:12 scale model of a 1968 Dodge Charger. No... Turnable front wheels; steering function detail...
S12_3990	1970 Plymouth Hemi Cuda	Very detailed 1970 Plymouth Cuda model in 1:18... products 5

Action	Output
Time	Action
5 21:40:19	SELECT o.orderNumber, o.status, od.priceEach FROM orders AS o RIGHT JOIN orderdetails od ON o.orderNumber = od.orderNumber...
6 21:40:48	SELECT productCode, productName, productDescription FROM products

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

```
SELECT o.orderNumber, o.status, od.priceEach
FROM orders o
RIGHT JOIN orderdetails od
ON o.orderNumber = od.orderNumber;
```

orderNumber	status	priceEach
10100	Shipped	126.00
10100	Shipped	55.09
10100	Shipped	25.40
10100	Shipped	35.29
10101	Shipped	108.06
10101	Shipped	32.53
10101	Shipped	25.40
10102	Shipped	95.95
10102	Shipped	43.13
10103	Shipped	120.20
10103	Shipped	119.67
10103	Shipped	121.64
10103	Shipped	24.93

sql > x

1) Action Output

2) Action

3) Action

4. 21 05:18 SELECT od.orderNumber, o.status, od.priceEach FROM orders AS o RIGHT JOIN orderdetails AS od ON od.orderNumber = o.orderNumber WHERE o.status = 'Shipped' ORDER BY od.orderLineNumber;

5. 21 05:18 SELECT o.orderNumber AS numerorden, o.orderDate AS fecha, od.orderLineNumber,

p.productName, od.quantityOrdered, od.priceEach

FROM orderdetails od
RIGHT JOIN products p
ON od.productCode = p.productCode
RIGHT JOIN orders o
ON od.orderNumber = o.orderNumber
ORDER BY numerorden, od.orderLineNumber;

numerorden	fecha	orderNumber	productName	quantityOrdered	priceEach
2003-01-06	2003-01-06	10000	Mercedes-Benz S500 Roadster	1	126.00
2003-01-06	2003-01-06	10000	1961 Ford Town Car	50	55.09
2003-01-06	2003-01-06	10000	1960 Lincoln Town Car	20	25.40
2003-01-06	2003-01-06	10000	1932 Alfa Romeo 8C2300 Spider Sport	22	35.29
2003-01-09	2003-01-09	10001	1938 Mercedes-Benz 50K	26	108.06
2003-01-09	2003-01-09	10001	1932 Lincoln Model A Standard Limousine	45	43.13
2003-01-09	2003-01-09	10001	1939 Chevrolet Deluxe Coupe	45	32.53
2003-01-09	2003-01-09	10001	1938 Lincoln Model K	25	35.29
2003-01-09	2003-01-09	10002	1938 Lincoln Model K	41	43.13
2003-01-09	2003-01-09	10002	1937 Lincoln Berline	39	95.95
2003-01-09	2003-01-09	10002	1938 Lincoln Model K	26	35.29
2003-01-09	2003-01-09	10003	1929 Ford Model A Engine	22	58.34
2003-01-09	2003-01-09	10003	1938 Lincoln Model K Express	21	58.34
2003-01-29	2003-01-29	10003	1962 Lincoln Sedan '39	42	119.67

sql > x

1) Action Output

2) Action

3) Action

4. 18 05:29 SELECT od.orderNumber AS numerorden, o.orderDate AS fecha

13 18 05:29 SELECT od.orderNumber AS numerorden, o.orderDate AS fecha

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

SELECT od.orderNumber, p.productName, p.MSRP, od.priceEach
FROM products p
RIGHT JOIN orderdetails od
ON od.productCode = p.productCode;

orderNumber	productName	MSRP	priceEach
10100	1932 Ford V8 Coupe	176.00	126.00
10100	1931 Ford V8 Coupe	176.00	55.09
10100	1932 Alfa Romeo 8C2300 Spider Sport	92.00	35.29
10100	1938 Mercedes-Benz 50K	120.00	108.06
10101	1932 Model A Ford 5 Window Coupe	127.13	29.98
10101	1938 Lincoln Model K	120.00	43.13
10101	1939 Chevrolet Deluxe Coupe	31.19	32.53
10101	1938 Lincoln Model K	120.00	35.29
10102	1937 Lincoln Berline	132.74	95.95
10102	1938 Lincoln Model K	120.00	35.29
10102	1938 Lincoln Model K Special Roadster	39.00	43.13
10103	1932 Alpine Renault 1300	214.90	214.90
10103	1938 Lincoln Model K	120.00	35.29
10103	1938 Seville Riva	126.17	121.64
10103	1940 Ford Pickup Truck	136.47	94.90

sql > x

1) Action Output

2) Action

3) Action

4. 22 05:02 SELECT od.orderNumber AS numerorden, p.productName AS productName, p.MSRP AS msrp, od.priceEach AS priceEach

12 22 05:02 SELECT od.orderNumber AS numerorden, p.productName AS productName, p.MSRP AS msrp, od.priceEach AS priceEach

Para estas consultas usa LEFT JOIN

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

SELECT o.customerNumber, c.customerName, o.orderNumber, c.state
FROM orders o
LEFT JOIN customers c
ON o.customerNumber = c.customerNumber;

CustomerNumber	CustomerName	OrderNumber	State
100	Atelier graphique	00123	CA
101	Atelier graphique	00248	CA
102	Atelier graphique	00709	CA
112	Signal Gif Stores	00124	NL
122	Signal Gif Stores	00125	NL
122	Signal Gif Stores	00396	NL
124	Australian Collectors, Co.	00126	Victoria
124	Australian Collectors, Co.	00323	Victoria
124	Australian Collectors, Co.	00324	Victoria
124	Australian Collectors, Co.	00347	Victoria
124	Australian Collectors, Co.	00350	Victoria
125	La Rochelle Gifts	00115	CA
126	La Rochelle Gifts	00175	CA

result 4 rows

sql

Action Detail

* Tree * Lines * Message

18 10:12:22 SELECT c.customerName, c.customerNumber, o.orderNumber, o.state FROM customers c LEFT JOIN orders o ON c.customerNumber = o.customerNumber WHERE o.customerNumber IS NULL;

18 10:12:41 SELECT c.customerName, c.customerNumber, o.orderNumber, o.state FROM customers c LEFT JOIN orders o ON c.customerNumber = o.customerNumber WHERE o.customerNumber IS NULL;

6. Obtén los clientes que no tienen una orden asociada.

```
SELECT c.customerName
FROM customers c
LEFT JOIN orders o
    ON c.customerNumber = o.customerNumber
WHERE o.customerNumber IS NULL;
```

7. 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.customerName, p.checkNumber, p.amount
FROM employees e
LEFT JOIN customers c
    ON e.employeeNumber = c.salesRepEmployeeNumber
LEFT JOIN payments p
    ON c.customerNumber = p.customerNumber;
```

lastName	firstName	customerName	checkNumber	amount
Murphy	Dawn			
Patterson	Mark			
Pirelli	Jeff			
Patterson	Mark			
Randall	Gerald			
Ross	Anthony			
Jennings	Lettie	Mev Gif Distributors Ltd.	H212453	12414.59
Jennings	Leesle	Mev Gif Distributors Ltd.	B235466	8542.87
Jennings	Leesle	Mev Gif Distributors Ltd.	CQ29767	1044.30
Jennings	Leesle	Mev Gif Distributors Ltd.	H235467	12414.59
Jennings	Leesle	Mev Gif Distributors Ltd.	H236474	47142.70
Jennings	Leesle	Mev Gif Distributors Ltd.	HR86578	55639.46
Jennings	Leesle	Mev Gif Distributors Ltd.	HR86579	55639.46
Jennings	Leesle	Mev Gif Distributors Ltd.	LS22759	43369.30
Jennings	Leesle	Mev Gif Distributors Ltd.	LS227599	43369.30

result 4 rows

sql

Action Detail

* Tree * Lines * Message

4 10:11:59 SELECT c.customerName FROM customers c LEFT JOIN orders o ON c.customerNumber = o.c...

5 10:40:53 SELECT e.lastName, e.firstName, c.customerName, o.checkNumber, p.amount FROM employees e...

Para estas consultas usa **RIGHT JOIN**

8. Repite los ejercicios 5 a 7 usando **RIGHT JOIN**.

```
SELECT c.customerNumber, c.customerName, o.orderNumber, c.state
FROM customers c
RIGHT JOIN orders o
    ON c.customerNumber = o.customerNumber;
```

customerNumber	customerName	orderNumber
201	Atelier graphique	00123
202	Atelier graphique	00298
203	Atelier graphique	00300
212	Signal Soft Services	00124
213	Signal Soft Services	00125
212	Signal Soft Services	00306
119	Australian Collectors, Co.	00126
120	Australian Collectors, Co.	00127
124	Australian Collectors, Co.	00223
124	Australian Collectors, Co.	00224
124	Australian Collectors, Co.	00247
124	Australian Collectors, Co.	00248
119	Le Rochelle Gifte	00315
119	Le Rochelle Gifte	00375

Result 13 x

Action Output

Time Action Message

27 13:43:31 SELECT c.customerName, c.customerNumber, o.orderNumber FROM orders o RIGHT JOIN customers c ON o.customerNumber = c.customerNumber WHERE o.customerNumber IS NULL; 297 rows returned

28 13:43:42 SELECT e.lastName, e.firstName, c.customerName, p.checkNumber, p.amount FROM payments p RIGHT JOIN customers c ON p.customerNumber = c.customerNumber RIGHT JOIN employees e ON c.salesRepEmployeeNumber = e.employeeNumber;

29 13:43:42 SELECT e.lastName, e.firstName, c.customerName, p.checkNumber, p.amount FROM payments p RIGHT JOIN customers c ON p.customerNumber = c.customerNumber WHERE c.state = 'VA'; 283 rows returned

```
SELECT c.customerName
FROM orders o
RIGHT JOIN customers c
    ON o.customerNumber = c.customerNumber
WHERE o.customerNumber IS NULL;
```

Customer	Order	Product	Quantity	Unit Price	Line Total
201 Atelier graphique	00123	100 - Red Cappuccino	1	10.50	10.50
202 Atelier graphique	00298	100 - Red Cappuccino	1	10.50	10.50
203 Atelier graphique	00300	100 - Red Cappuccino	1	10.50	10.50
212 Signal Soft Services	00124	100 - Red Cappuccino	1	10.50	10.50
213 Signal Soft Services	00125	100 - Red Cappuccino	1	10.50	10.50
212 Signal Soft Services	00306	100 - Red Cappuccino	1	10.50	10.50
119 Australian Collectors, Co.	00126	100 - Red Cappuccino	1	10.50	10.50
120 Australian Collectors, Co.	00127	100 - Red Cappuccino	1	10.50	10.50
124 Australian Collectors, Co.	00223	100 - Red Cappuccino	1	10.50	10.50
124 Australian Collectors, Co.	00224	100 - Red Cappuccino	1	10.50	10.50
124 Australian Collectors, Co.	00247	100 - Red Cappuccino	1	10.50	10.50
124 Australian Collectors, Co.	00248	100 - Red Cappuccino	1	10.50	10.50
119 Le Rochelle Gifte	00315	100 - Red Cappuccino	1	10.50	10.50
119 Le Rochelle Gifte	00375	100 - Red Cappuccino	1	10.50	10.50

Result 26 x

Action Output

Time Action Message

27 18:34:21 SELECT c.customerName FROM orders o RIGHT JOIN customers c ON o.customerNumber = c.customerNumber WHERE c.state = 'VA'; 297 row(s) returned

28 18:39:23 SELECT e.lastName, e.firstName, c.customerName, p.checkNumber, p.amount FROM payments p RIGHT JOIN customers c ON p.customerNumber = c.customerNumber WHERE c.state = 'VA'; 283 row(s) returned

```
SELECT e.lastName, e.firstName, c.customerName, p.checkNumber, p.amount
FROM payments p
RIGHT JOIN customers c
    ON p.customerNumber = c.customerNumber
RIGHT JOIN employees e
    ON c.salesRepEmployeeNumber = e.employeeNumber;
```

lastName	firstName	customerName	checkNumber	amount
Murphy	Diana	NULL	NULL	NULL
Patterson	Mary	NULL	NULL	NULL
Firelli	Jeff	NULL	NULL	NULL
Patterson	William	NULL	NULL	NULL
Bondu	Gerard	NULL	NULL	NULL
Bow	Anthony	NULL	NULL	NULL
Jennings	Leslie	Mini Gifts Distributors Ltd.	AE215433	101244.59
Jennings	Leslie	Mini Gifts Distributors Ltd.	BG259406	85410.87
Jennings	Leslie	Mini Gifts Distributors Ltd.	CQ287967	11044.30
Jennings	Leslie	Mini Gifts Distributors Ltd.	ET64396	83598.04
Jennings	Leslie	Mini Gifts Distributors Ltd.	H1366474	47142.70
Jennings	Leslie	Mini Gifts Distributors Ltd.	HR86578	55639.66
Jennings	Leslie	Mini Gifts Distributors Ltd.	K131716	111654.40
Jennings	Leslie	Mini Gifts Distributors Ltd.	LF217299	43369.30

Result 26 x

Action Output

Time Action Message

27 18:34:21 SELECT c.customerName FROM orders o RIGHT JOIN customers c ON o.customerNumber = c.customerNumber WHERE c.state = 'VA'; 297 row(s) returned

28 18:39:23 SELECT e.lastName, e.firstName, c.customerName, p.checkNumber, p.amount FROM payments p RIGHT JOIN customers c ON p.customerNumber = c.customerNumber WHERE c.state = 'VA'; 283 row(s) returned

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

```
CREATE VIEW AVco AS
(SELECT c.customerNumber, c.customerName, o.orderNumber, c.state
FROM customers c
RIGHT JOIN orders o
    ON c.customerNumber = o.customerNumber);
```

#cuantas órdenes han hecho los clientes

```
SELECT COUNT(*)
FROM AVco;
```

COUNT(*)
326

```
CREATE VIEW AVpce AS
```

```
(SELECT e.lastName, e.firstName, c.customerName, p.checkNumber, p.amount
FROM payments p
RIGHT JOIN customers c
    ON p.customerNumber = c.customerNumber
RIGHT JOIN employees e
    ON c.salesRepEmployeeNumber = e.employeeNumber);
```

#Cuantos empleados no venden

```
SELECT concat(firstName, ' ', lastName) AS name, SUM(amount)
FROM AVpce
GROUP BY name
HAVING SUM(amount) IS NULL;
```

name	SUM(amount)
Diane Murphy	NULL
Mary Patterson	NULL
Jeff Firrelli	NULL
William Patterson	NULL
Gerard Bondur	NULL
Anthony Bow	NULL
Tom King	NULL
Yoshimi Kato	NULL

```
CREATE VIEW AVo AS
(SELECT o.orderNumber, o.status, od.priceEach, od.quantityOrdered
FROM orders o
RIGHT JOIN orderdetails od
    ON o.orderNumber = od.orderNumber);
```

#Menciona las ordenes que se cancelaron y su precio

```
SELECT orderNumber, SUM(priceEach * quantityOrdered) AS total, status
FROM AVo
GROUP BY orderNumber
HAVING status LIKE 'Cancelled';
```

orderNumber	total	status
10167	44167.09	Cancelled
10179	22963.69	Cancelled
10248	41445.21	Cancelled
10253	45443.54	Cancelled
10260	37769.38	Cancelled
10262	47055.36	Cancelled

Sesión 4 - Mongo DB

Reto 1

1. Fecha, nombre y texto de cada comentario.

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

2. Título, elenco y año de cada película.

{ title:1, cast:1, year:1 }

3. Nombre y contraseña de cada usuario.

{ name:1, password:1 }

Reto 2

1. ¿Qué comentarios ha hecho Greg Powell?

Filter: { name: 'Greg Powell' }

```
_id: ObjectId("573a1399f29313caabceee886")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee886")
text: "Fugit dignissimos nisi sapi... Eligendi presentium unde quod parro. Co..."
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee887")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee887")
text: "Eius veritatis vero facilis occuerat. Fuga temporibus. Presentium expel... "
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee888")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee888")
text: "Voluptate odio scidens peratur. Reconsente. Architecto illum dicta rep... "
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee889")
name: "Mercedes Tyler"
email: "mercedes_tyler@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee889")
text: "Eius veritatis vero facilis occuerat. Fuga temporibus. Presentium expel... "
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee88a")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee88a")
text: "Fugit dignissimos nisi sapi... Eligendi presentium unde quod parro. Co..."
date: 2017-03-10T00:22:45.480+00:00
```

2. ¿Qué comentarios han hecho Greg Powell o Mercedes Tyler?

Filter: { \$or: [{ name: 'Greg Powell' }, { name: 'Mercedes Tyler' }] }

```
_id: ObjectId("573a1399f29313caabceee886")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee886")
text: "Fugit dignissimos nisi sapi... Eligendi presentium unde quod parro. Co..."
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee887")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee887")
text: "Eius veritatis vero facilis occuerat. Fuga temporibus. Presentium expel... "
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee888")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee888")
text: "Voluptate odio scidens peratur. Reconsente. Architecto illum dicta rep... "
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee889")
name: "Mercedes Tyler"
email: "mercedes_tyler@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee889")
text: "Eius veritatis vero facilis occuerat. Fuga temporibus. Presentium expel... "
date: 2017-03-10T00:22:45.480+00:00

_id: ObjectId("573a1399f29313caabceee88a")
name: "Greg Powell"
email: "greg_powell@gmail.com"
movie_id: ObjectId("573a1399f29313caabceee88a")
text: "Fugit dignissimos nisi sapi... Eligendi presentium unde quod parro. Co..."
date: 2017-03-10T00:22:45.480+00:00
```

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

Project: { title:1, num_mflix_comments: 1 }

Sort: { num_mflix_comments: -1 }

Limit: 1

```
FILTER
PROJECT { title:1, num_mflix_comments: 1 }
SORT { num_mflix_comments: -1 }
COLLATION
MAXTIMEMS 5000
SKIP 0
LIMIT 1
DISPLAYING DOCUMENTS 1 - 1 OF 1

_id: ObjectId("573a1399f29313caabceee886")
title: "The Mask"
num_mflix_comments: 456
```

4. ¿Cuál es título de las cinco películas más comentadas?

Project: { title:1, num_mflix_comments: 1 }

Sort: { num_mflix_comments: -1 }

Limit: 5

```

{
  "_id": "52cde7c4ab8bd675297d8e",
  "name": "WetPaint, Inc."
},
{
  "_id": "52cde7c4ab8bd675297d8f",
  "name": "Facebook"
},
{
  "_id": "52cde7c4ab8bd675297d90",
  "name": "OneDrive"
},
{
  "_id": "52cde7c4ab8bd675297d91",
  "name": "Twitter"
},
{
  "_id": "52cde7c4ab8bd675297d92",
  "name": "StumbleUpon"
}

```

Proyecto

1. Obtén los datos de contacto de cada compañía.

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

```

{
  "_id": "52cde7c4ab8bd675297d8e",
  "name": "WetPaint, Inc.",
  "email_address": "info@wetpaint.com",
  "phone_number": "+06.859.698",
  "homepage_url": "http://wetpaint.com"
},
{
  "_id": "52cde7c4ab8bd675297d8f",
  "name": "Facebook",
  "email_address": "",
  "phone_number": "",
  "homepage_url": "http://facebook.com"
},
{
  "_id": "52cde7c4ab8bd675297d90",
  "name": "OneDrive",
  "email_address": "info@onedrive.com",
  "phone_number": "+66.675.5002",
  "homepage_url": "http://www.onedrive.com"
},
{
  "_id": "52cde7c4ab8bd675297d91",
  "name": "Twitter",
  "email_address": "press@twitter.com",
  "phone_number": "",
  "homepage_url": "http://twitter.com"
},
{
  "_id": "52cde7c4ab8bd675297d92",
  "name": "StumbleUpon",
  "email_address": "",
  "phone_number": "",
  "homepage_url": "http://www.stumbleupon.com"
}

```

2. Obtén la fuente de cada tweet.

```
{
  project: {
    source: 1
  }
}
```

```

{
  "_id: ObjectId("522def7caababdd075297fda")
  source: "url"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  source: "url http://www.busteddeck.com" rel="nofollow" href="http://www.busteddeck.com"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  source: "url http://slimscraper.com/buster" rel="nofollow" href="http://slimscraper.com/buster" hreflang="en"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  source: "url http://www.electafon.com" rel="nofollow" href="http://www.electafon.com"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  source: "url http://t3logix.com/poweroftwitter" rel="nofollow" href="http://t3logix.com/poweroftwitter"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  source: "url"
}

```

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

```
{
  filter: {
    founded_month: 10
  },
  project: {
    name: 1
  }
}
```

```

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "unspoint"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "Powerset"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "Tiled"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "TechologyGuide"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "Technews"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "Togzoo"
}

{
  "_id: ObjectId("522def7caababdd075297fda")
  name: "Togzoo"
}

```

4. Obtén el nombre de todas las compañías fundadas en 2008.

```
{
  filter: {
    founded_year: 2008
  },
  project: {
    name: 1
  }
}
```

5. Obtén todos los post del autor machine.

```
{
  filter: {
    author: 'machine'
  }
}
```

6. Obtén todos los tweets provenientes de la web.

```
{
  filter: {
    source: 'web'
  }
}
```

```

_id: ObjectId("52c2d772e0ab0679298621")
text: {"text": "No pude de terminar de hacer x porque tabata, est\u00f3a mu\u00f1o foda ***", "reset": true}
createdAt: 2010-10-20T11:11:11+0000:2010
get: null
coordinates: null
coordinates_type: null
created_by: null
created_by_name: null
translated: false
sentiment: null
retweets: null
user: Object
user_id: null
user_id_str: null
is_reply_to_user_id: null
is_reply_to_user_id_str: null

_id: ObjectId("52c2d772e0ab0679298621")
text: {"text": "First year of school is over :)", "reset": true}
createdAt: 2010-10-20T11:11:11+0000:2010
get: null
coordinates: null
coordinates_type: null
created_by: null
created_by_name: null
translated: false
sentiment: null

```

7. Obtén todas las compa\u00f1ías fundadas en octubre del 2008.

```
{
filter: {
  $and: [
    {
      founded_year: 2008
    },
    {
      founded_month: 10
    }
  ]
}
}
```

```

_id: ObjectId("52c2d772e0ab0679298621")
permalink: "tunesbag"
crunchbase_url: "http://www.crunchbase.com/company/tunesbag"
blog_url: "http://tunesbag.tumblr.com/"
facebook_url: "http://www.facebook.com/tunesbag"
twitter_username: "tunesbag_tunesbag"
category_code: "game_alpha"
parent_id: null
founded_year: 2008
founded_month: 10
founded_day: 1
founded_hour: 0
founded_minute: null
founded_second: null
deactivated_year: null
deactivated_month: null
deactivated_day: null
deactivated_hour: null
deactivated_minute: null
deactivated_second: null
tag_list: "music, cloud, locker, mp3, music-streaming, streaming"
email_address: "official@tunesbag.com"
description: "Social Music Player"
created_at: "Thu Mar 26 11:41:48 UTC 2015"
overview: "playindia band tunesbag is a music player with social features on th..."

SHOW 17 MORE FIELDS

```

_id: ObjectId("52c2d772e0ab0679298621")

8. Obtén todas las compa\u00f1ías con m\u00e1s de 50 empleados.

```
{
filter: {
  number_of_employees: {
    $gt: 50
  }
}
```

```
{
}

{
    "_id": "ObjectId(\"52cdef7c48abbd675297d8e\")",
    "name": "Facebook",
    "parent_id": null,
    "crunchbase_url": "http://www.crunchbase.com/company/facebook",
    "homepage_url": "http://facebook.com",
    "blog_url": "http://www.facebook.com",
    "blog_feed_url": "http://www.facebook.com/atom.php",
    "twitter_username": "facebook",
    "category_code": "social",
    "number_of_employees": 5299,
    "founded_year": 2004,
    "founded_month": 2,
    "founded_day": 4,
    "deactivated": null,
    "deactivated_month": null,
    "deactivated_day": null,
    "deactivated_time": null,
    "tag_list": "facebook, college, students, profiles, network, online-communities, so...",
    "alias_list": "",
    "website": "http://facebook.com",
    "phone_number": "",
    "description": "Social network",
    "created_at": "Fri May 25 22:25:10 UTC 2007",
    "updated_at": "Thu May 21 18:48:55 UTC 2015",
    "overview": "<p>Facebook is the world's largest social network, with over <a href='https://www.facebook.com/about/size/'>2 billion</a> users. It was founded in 2004 by Mark Zuckerberg and is based in Menlo Park, California. Facebook has become one of the most popular social media platforms, connecting people across the globe through sharing news, photos, and other content. The company has expanded significantly over the years, adding features like messaging, groups, and marketplace. Facebook also offers advertising services to businesses and brands. The platform plays a major role in modern communication and has transformed the way we interact with each other online.</p>",
    "image": "https://www.crunchbase.com/resource/facebook/logo/52cdef7c48abbd675297d8e"
}

```

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

```
{
    filter: {
        $and: [
            {
                comments: {
                    $gte: 10
                }
            },
            {
                comments: {
                    $lte: 30
                }
            }
        ]
    }
}
```

The screenshot shows a MongoDB query interface with the following search parameters:

- \$and:** [{comments: { \$gt: 10 }}, {comments: { \$lt: 30 }}]
- OPTIONS:** MAXTIMEMS 5000, SKIP 0, LIMIT 0
- DISPLAY:** ADD DATA, VIEW, O, M
- RESULTS:** Displaying documents 1 - 20 of 1930

The results list two documents:

- Document 1:**
 - id:** ObjectId("4d9570120801baeab000001")
 - level:** "http://digg.com/travel_places/11_Amazing_Treehouses_from_around_the_Wo..."
 - title:** "11 Amazing Treehouses from Around the World"
 - container:** Object
 - type:** "http://digg.com/_type/120801baeab00001"
 - topic:** Object
 - promote_date:** 120801baeab00001
 - parent:** null
 - media:** "none"
 - status:** "popular"
 - whereurl:** Array
- Document 2:**
 - id:** ObjectId("4d9570120801baeab000002")
 - level:** "http://digg.com/science_ma_science/Ma_Wizard_Cluster"
 - title:** "MaWa - The Wizard Cluster"
 - container:** Object
 - type:** "http://digg.com/_type/120801baeab00002"
 - topic:** Object
 - promote_date:** 120801baeab00002
 - parent:** null
 - media:** "image"
 - status:** "popular"
 - whereurl:** Array

10. Obtén la empresa con el menor número de empleados.

```
{
  filter: {
    $and: [
      {
        number_of_employees: {
          $ne: null
        }
      },
      {
        number_of_employees: {
          $ne: 0
        }
      }
    ]
  },
  sort: {
    number_of_employees: 1
  },
  limit: 1
}
```

A screenshot of a MongoDB query interface. The top bar shows a filter for 'PROJECT' and a sort by 'number_of_employees: 1'. The results section displays one document with the following fields:

```

_id: ObjectId("52cdef7fc4ba88bd67529856a")
name: "Fevotte"
permalink: "fevotte"
crunchbase_url: "http://www.crunchbase.com/company/fevotte"
homepage_url: "http://www.fevotte.com"
blog_url: "http://blog.fevotte.com"
blog_feed_url: "http://blog.fevotte.com/feed/"
category_code: null
category_name: null
number_of_employees: 1
founded_year: null
founded_month: null
deactivated_year: null
deactivated_month: null
deactivated_day: null
deactivated_url: null
tag_list: null
email_address: ""
phone_number: ""
description: null
created_at: "Tue Sep 23 23:41 UTC 2008"
updated_at: "Tue Sep 23 23:44 UTC 2008"
overview: "opfevotte provides suggestion boards for companies and various subject..."

```

[SHOW 17 MORE FIELDS](#)

11. Obtén la empresa con el mayor número de empleados.

```
{
  sort: {
    number_of_employees: -1
  },
  limit: 1
}
```

A screenshot of a MongoDB query interface. The top bar shows a filter for 'PROJECT' and a sort by 'number_of_employees: -1'. The results section displays one document with the following fields:

```

_id: ObjectId("52cdef7fc4ba88bd67529856a")
name: "IBM"
permalink: "ibm"
crunchbase_url: "http://www.crunchbase.com/company/ibm"
homepage_url: "http://www.ibm.com"
blog_url: null
blog_feed_url: ""
twitter_username: "IBM"
category_code: "software"
number_of_employees: 388000
founded_year: 1911
founded_month: null
founded_day: null
deactivated_year: null
deactivated_month: null
deactivated_day: null
deactivated_url: ""
tag_list: ""
alias_name: ""
email_address: "env@us.ibm.com"
phone_number: "914-499-1900"
description: ""
created_at: "Fri Mar 14 21:55:52 UTC 2008"
updated_at: "Tue Sep 23 02:56:24 UTC 2014"
overview: "opibm acronym for International Business Machines, is a multinationa..."
image: Object
  properties: Array
  relationships: Array
  competitions: Array
  providerships: Array
  total money raised: "6b"

```

12. Obtén la historia más comentada.

```
{
  sort: {
    comments: -1
  },
  limit: 1
}
```

```

{
  "_id": "ObjectId(\"4ba27e0238d3ba3ca002251\")",
  "href": "http://digg.com/politics/Republican_Brown_wins_Massachusetts_Senate_seat",
  "title": "Republican Brown wins Massachusetts Senate seat!",
  "comments": 1864,
  "thumbnail": Object,
  "container": Object,
  "submit_date": 1263954202,
  "topic": Object,
  "promote_date": 1263956432,
  "id": "1862678",
  "media": "news",
  "diggs": 298,
  "description": "striving for an epic upset in liberal Massachusetts, Republican Scott ...",
  "link": "http://news.yahoo.com/s/ap/us_massachusetts_senate_ylt-AuEva3AHQxosR...",
  "user": Object,
  "status": "popular",
  "shorturl": Array
}

```

13. Obtén la historia menos comentada.

```
{
  filter: {
    comments: {
      $ne: 0
    }
  },
  sort: {
    comments: 1
  },
  limit: 1
}
```

```

{
  "_id": "ObjectId(\"4ba27e0238d3ba3ca001819\")",
  "href": "http://digg.com/general_sciences/Smallest_eel_loach_Fish_Discovered",
  "title": "Smallest eel-loach Fish Discovered",
  "comments": 2,
  "thumbnail": Object,
  "container": Object,
  "submit_date": 1265651817,
  "topic": Object,
  "promote_date": 1265703003,
  "id": "19106141",
  "media": "news",
  "diggs": 176,
  "description": "The world's smallest species of eel-loach fish has been discovered by ...",
  "link": "http://www.physorg.com/news184855470.html",
  "user": Object,
  "status": "popular",
  "shorturl": Array
}

```

Sesión 5

Reto 1

1. Propiedades que no permitan fiestas.

```
{  
  filter: {  
    house_rules: RegExp('.*No parties*', i)  
  }  
}
```

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

- Filter: { house_rules: /.*No parties*/i }
- Options: FIND, RESET
- Results: 20 of 243 documents

One document is displayed in the results table:

<code>_id: "581151"</code>	<code>link: "https://www.airbnb.com/rooms/193161"</code>
<code>name: "My Apt It's right in the middle of the best and hot spots in the heart..."</code>	<code>summary: "My Apt It's right in the middle of the best and hot spots in the heart..."</code>
<code>description: "My Apt It's right in the middle of the best and hot spots in the heart..."</code>	<code>neighborhood_overview: "Budapest Avenue is the neighborhood's main thoroughfare, with its colle..."</code>
<code>transit: "Yes, there is plenty of public transportation around. 3 metro lines! ..."</code>	<code>interaction: "As much as they need my help and guidance."</code>
<code>house_rules: "NO parties allowed inside the apt. NO parties are allowed. NO Kid..."</code>	<code>room_type: "Entire home/apt"</code>
<code>amenities: "Pet friendly"</code>	<code>bedrooms: 1</code>
<code>beds: 1</code>	

2. Propiedades que admitan mascotas.

```
{  
  filter: {  
    house_rules: RegExp('.*pet friendly*', i)  
  }  
}
```

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

- Filter: { house_rules: /.*pet friendly*/i }
- Options: FIND
- Results: 7 of 7 documents

One document is displayed in the results table:

<code>_id: "18944834"</code>	<code>link: "https://www.airbnb.com/rooms/18944834"</code>
<code>name: "Private bedroom in the heart of Manhattan"</code>	<code>summary: "spectacular private bedroom in UES! a lot of natural lights!"</code>
<code>space: "You can use kitchen and we will share a bathroom"</code>	<code>description: "spectacular private bedroom in UES! a lot of natural lights! You can use kitchen and we will share a bathroom, everything just arou..."</code>
<code>neighborhood_overview: "The place to be in the north shore is where you can sleep from the ..."</code>	<code>notes: "I have a wonderful Turkish angora cat, so please be pet friendly!"</code>
<code>transit: "5 min walk to Central Park, 5 min walk to East River, 2 blocks away ..."</code>	<code>interaction: "you will have your own private bedroom and we will share bathroom, kit..."</code>
<code>house_type: "Apartment"</code>	<code>access: "you will have your own private bedroom and we will share bathroom, kit..."</code>
<code>property_type: "Private Room"</code>	<code>amenities: "Pet friendly since i have a wonderful turkish angora cat"</code>
<code>room_type: "Real Bed"</code>	<code>bed_type: "Real Bed"</code>
<code>min_nights: 1</code>	<code>max_nights: "112"</code>
<code>cancellation_policy: "strict_14_with_grace_period"</code>	<code>maximum_nights: "112"</code>
<code>last_scraped: 2010-03-06T05:00:00.000+00:00</code>	<code>minimum_nights: "1"</code>
<code>calendar_last_scraped: 2019-03-06T05:00:00.000+00:00</code>	<code>beds: 1</code>
<code>first_review: 2017-06-22T04:00:00.000+00:00</code>	<code>bedrooms: 1</code>
<code>last_review: 2018-12-31T05:00:00.000+00:00</code>	

3. Propiedades que no permitan fumadores.

```
{
```

```

filter: {
  house_rules: RegExp('.*no smoking*', i)
}

```

The screenshot shows a search interface with a filter applied: `house_rules: /.*no smoking*/i`. The results page displays two property listings. The first listing is for a property with ID `_id: "10893448"`, located in New York City, Upper West Side. It has a summary: "Hurdy-hed, optional second bedroom available, wifis available, NYC, N...". The second listing is for a property with ID `_id: "10893448"`, located in New York City, Upper West Side. It has a summary: "In Happy Apartment is an amazing space. Renovated and comfortable apar...". Both listings include details like address, neighborhood, and availability.

4. Propiedades que no permitan fiestas ni fumadores.

```

{
filter: {
$and: [
{
  house_rules: RegExp('.*no smoking*', i)
},
{
  house_rules: RegExp('.*no parties*', i)
}
]
}

```

The screenshot shows a search interface with filters applied: `house_rules: /.*no smoking*/i` and `house_rules: /.*no parties*/i`. The results page displays two property listings. The first listing is for a property with ID `_id: "10893248"`, located in North Shore, Long Island. It has a summary: "The place to be on the North Shore is where you can be steps from the ...". The second listing is for a property with ID `_id: "108423504"`, located in North Bondi. It has a summary: "This peaceful house in North Bondi is 300m to the beach and a minute's...". Both listings include details like address, neighborhood, and availability.

Reto 2

1. 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.

```
{  
  filter: {  
    number_of_reviews: {  
      $gte: 50  
    },  
    'review_scores.review_scores_rating': {  
      $gte: 80  
    },  
    amenities: {  
      $in: [  
        RegExp('Ethernet', i)  
      ]  
    },  
    'address.country': 'Brazil'  
  }  
}
```

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

- Query: `review_scores.review_scores_rating: { $gte: 80 }, amenities: { $in: [/Ethernet/i] }, "address.country": "Brazil"`
- Results: 2 documents found (1-2 of 6).
- Document 1:
 - `listing_id: "4d1708044e4b4a3f80000001"`
 - `listing_url: "https://www.airbnb.com/rooms/104812"`
 - `name: "Cozying apartment,perfect location"`
 - `summary: "This is a bedroom and living room, cleaning, private, finely decorated,..."`
 - `space: "It is a bedroom and living room, cleaning, private, finely decorated,..."`
 - `neighborhood_overview: "There are a lot of bars, restaurants, supermarket, banks,...,..."`
 - `transit: "From the apartment you can walk to the bus stop or take a taxi very near to the building..."`
 - `amenities: "The full apartment is in some progress with my partner. It depends on my time and..."`
 - `house_rules: "2.5 apartments situated on a little residential. não é permitido fumar..."`
 - `room_type: "Entire home/apt"`
 - `host_since: "2013-01-01T00:00:00-00:00"`
 - `calendar_last_scraped: "2019-01-11T00:00:00-00:00-00:00"`
 - `last_review: "2019-01-01T00:00:00-00:00-00:00"`
 - `host_id: "104812"`
 - `bedrooms: 1`
- Document 2:
 - `listing_id: "4d1708044e4b4a3f80000002"`
 - `listing_url: "https://www.airbnb.com/rooms/104812"`
 - `name: "Stay in a quiet place with great location! Free wifi access throughout..."`
 - `summary: "Stay in a quiet place with great location! Free wifi access throughout..."`
 - `space: "Stay in a quiet place with great location! Free wifi access throughout..."`
 - `neighborhood_overview: "Stay in a quiet place with great location! Free wifi access throughout..."`
 - `transit: "Feel free to smoke in our outdoor area, but never in any room inside..."`
 - `amenities: "No smoking in the bedrooms, outside areas and common areas."`

Reto 3

1. 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).

```
[{$match: {  
  amenities: { $in: [/Ethernet/i, /Wifi/i]}  
}}, {$group: {  
  _id: null,  
  total: {  
    $sum: 1
```

```
}
```

```
}}]
```

The screenshot shows the MongoDB Compass interface with two stages of a pipeline:

- \$match Stage:** A sample of 20 documents is shown. One document is expanded to show its fields:

```
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..."
neighborhood_overview: "In the neighborhood of the rive
several restaurants as..."
```
- \$group Stage:** A sample of 1 document is shown. One document is expanded to show its fields:

```
_id: null
total: 5308
```

Proyecto

1. 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: { $in: [/Ethernet/i] }
}}, {$match: {
  "address.country": {$in: [/Brazil/i]}
}}, {$count: 'null'}]
```

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

Output after \$match stage (Sample of 6 documents):

```

1 /**
2 * query: The query in MQL.
3 */
4 {
5     "address.country": {"$in: [/Brazil/]"}
6 }

```

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

```

1 /**
2 * Provide the field name for the count.
3 */
4 'null': 6

```

At the bottom, there is a button labeled "ADD STAGE".

Sesión 6

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',
    propiedades: {
        $sum: 1
    },
    total: {
        $sum: "$costo_recamara"
    }
}}, {$addFields: {
    costo_promedio: {
        $divide: ["$total", "$propiedades"]
    }
}}]

```

The screenshot shows the MongoDB aggregation pipeline interface with three stages:

- Output after \$group stage (Sample of 6 documents):**

```

1 {
2     "_id": "BR",
3     "propiedades": 6,
4     "total": 1000000
5 }

```

- Output after \$project stage (Sample of 6 documents):**

```

1 {
2     "_id": "BR",
3     "propiedades": 6,
4     "total": 1000000
5     "costo_promedio": 166666.66666666666
6 }

```

Reto 2

Usando las colecciones comments y users, se requiere conocer el correo y contraseña de cada persona que realizó un comentario. Construye un pipeline que genere como resultado estos datos.

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

The screenshot shows the MongoDB Compass interface with two stages of a pipeline. Stage 1, titled '\$addFields', shows a document with a new field 'correo'. Stage 2, titled '\$project', shows the final output with fields '_id', 'email', and 'contraseña'.

Reto 3

Usando el *pipeline* que generaste en el Reto 2, genera la vista correspondiente.

The screenshot shows the MongoDB Compass interface displaying the results of the pipeline. It lists multiple documents, each containing an 'email' field and a 'contraseña' field.

The screenshot shows the MongoDB Compass interface with the database 'sample_airbnb' selected. The 'Documents' tab is active, displaying a list of documents. The documents represent countries with their counts, such as Australia (187.8), Brazil (348.7), Canada (115.8), China (393.8), Hong Kong (154.6), Portugal (48.8), Russia (47.3), and others. The interface includes a toolbar with 'View', 'Find', and 'RESE' buttons.

Proyecto

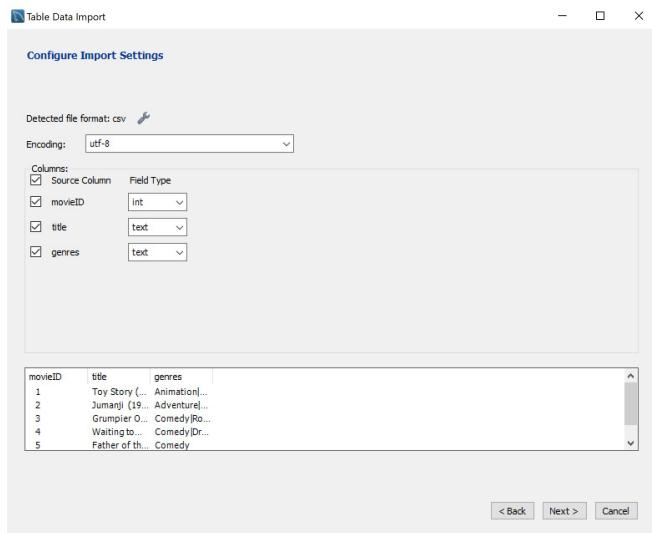
El proyecto consiste en obtener, por país, el número de películas que hay de cada género. Un ejemplo de salida en formato de tabla sería:

```
[$unwind: {
  path: "$genres"
}, {$unwind: {
  path: "$countries"
}}, {$group: {
  _id: {genero: "$genres", pais: "$countries"},
  peliculas: {
    $sum : 1
  }
}}, {$addFields: {
  País: "$_id.pais",
  Género: "$_id.genero"
}}, {$project: {
  _id:0
}}, {$sort: {
  peliculas: 1
}}]
```

The screenshot shows the MongoDB Compass interface with the database 'sample_mflix' selected. The 'Aggregations' tab is active, displaying an aggregation pipeline. The pipeline consists of four stages: a '\$project' stage, a '\$sort' stage, a '\$group' stage, and a '\$addFields' stage. The '\$project' stage output shows a sample of 20 documents with fields 'peliculas', 'País', and 'Género'. The '\$sort' stage output shows the same sample sorted by 'peliculas'. The '\$group' stage output shows the grouped documents with the '_id' field containing objects like {genero: "Comedy", pais: "Poland"} and {genero: "Thriller", pais: "Republic of Macedonia"}. The final '\$addFields' stage output adds the 'País' and 'Género' fields to the document. The interface shows statistics at the top: DOCUMENTS 23.5k TOTAL SIZE 35.9MB AVG. SIZE 1.6KB and INDEXES 2 TOTAL SIZE 13.1MB AVG. S 6.6B.

Sesión 7

Reto 1



Reto 2

Reto 3

Reto 4

Import To Collection AVH.ratings

Select File
C:\Users\andre\Documents\Data analysis\ml-1m\ratings.csv

Select Input File Type
 JSON CSV

Options
Select delimiter: COMMA
 Ignore empty strings
 Stop on errors

Specify Fields and Types

	userID	movieID	rating	timestamp
1	1	1193	5	978300760
2	1	661	3	9783002109
3	1	914	3	978301068
4	1	3408	4	9783002275
5	1	2355	5	9783024291
6	1	1197	3	9783002268
7	1	1287	5	9783002039
8	1	2804	5	9783000719
9	1	594	4	9783002268
10	1	919	4	978301368

Import completed 5,000 (100%)

Proyecto

Agregar los siguientes registros en formato CSV a la Colección movies

FILTER { movieID: { \$gte: "4000" } }

ADD DATA FIND RESET ...

Displaying documents 1 - 2 of 2

<pre>_id: ObjectId("5f18f4e4e62b391a133e7692") movieID: "4000" title: "Avengers: Endgame (2019)" genres: "Fantasy Sci-Fi"</pre>
<pre>_id: ObjectId("5f18f57fe62b391a133e7693") movieID: "4001" title: "Glass (2019)" genres: "Drama Fantasy"</pre>

Para recordar

id, titulo, genres
4000, Avengers: Endgame (2019), Fantasy|Sci-Fi
4001, Glass (2019), Drama|Fantasy

Y entonces el correspondiente formato JSON será:

```
{
  id: "4000",
  titulo: "Avengers: Endgame (2019)",
  genres: "Fantasy|Sci-Fi"
}
{
  id: "4001",
  titulo: "Glass (2019)",
  genres: "Drama|Fantasy"
```

```
}
```

Sesión 7

Reto 1-Starbucks

Usando la latitud y longitud de tu posición actual, encuentra el Starbucks más cercano a tu posición. Para conocer tu posición actual puedes usar Google Maps para, sólo debes copiar los datos de la URL.

```
[$match: {  
    $and:  
        [{Longitude: {$lte: "-103.99"}},  
         {Longitude: {$gte: "-103.40"}},  
         {Latitude: {$lte: "20.90"}},  
         {Latitude: {$gte: "20.70"}]}  
}]
```

The screenshot shows the MongoDB Compass interface with an aggregation pipeline. The top section displays the preview of documents in the collection, showing a sample document from Andorra with fields like City, State/Province, Country, Postcode, Phone Number, Timezone, Longitude, and Latitude. The bottom section shows the output after the \$match stage, displaying five documents from different locations: Ajman, Andorra la Vella, Valle Real Guadalajara, Technology Park, and Guadalajara. The left panel shows the full aggregation pipeline code, which includes the \$match stage defined above.

```
1 /*  
2  * query: The query in MQL.  
3  */  
4 {  
5     $and:  
6     [{Longitude: {$lte: "-103.99"}},  
7      {Longitude: {$gte: "-103.40"}},  
8      {Latitude: {$lte: "20.90"}},  
9      {Latitude: {$gte: "20.70"}]}  
10 }
```

Output after \$match stage (Sample of 5 documents)

_id	Brand	Store Number	Store Name	Ownership Type	Street Address	City	State/Province
ObjectId("5f1f76205a08422dc849cc2e")	Starbucks	30957-101702	Valle Real Guadalajara	Licensed	Prolongacion Avenida Santa Margarita, Subancia 3,	Guadalajara	JAL
ObjectId("5f1f76255a08422dc849cc2e")	Starbucks	47824-259926	Technology Par	Licensed	Carretera a	Guadalajara	JAL
ObjectId("5f1f76255a08422dc849cc2e")	Starbucks	22331-212325	Ajman Drive Th	Licensed	1 Street 6	Ajman	AJ
ObjectId("5f1f76255a08422dc849cc2e")	Starbucks	7	Andorra la Vella	AD	AD500	Andorra	AD

Reto 2-H1N1

1. ¿Cuál fue el país con mayor número de muertes?

```
{  
filter: {  
    Country: {  
        $ne: 'Grand Total'  
    }  
},  
sort: {
```

```

Deaths: -1
},
limit: 1
}

```

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

- Filter:** {Country: { \$ne: "Grand Total" }}
- Project:** (disabled)
- Sort:** {Deaths: -1}
- MaxItems:** 5000
- Collation:** (disabled)
- Skip:** 0
- Limit:** 1

The results table displays one document:

	Value
_id	ObjectId("5f1f71465a08422dc849a0a1")
Country	"United States of America"
Cases	33902
Deaths	170
Update Time	"7/6/2009 9:00"

Displaying documents 1 - 1 of 1

2. ¿Cuál fue el país con menor número de muertes?

```

[{$match: {
  Country: {
    $ne: 'Grand Total'
  },
  Deaths: {$ne: NaN}
}}, {$addFields: {
  date: {
    $dateFromString: {
      dateString: '$Update Time',
      format: '%m/%d/%Y %H:%M'
    }
  }
}}, {$sort: {
  date: -1
}}, {$sort: {
  Deaths: 1
}}, {$limit: 1}]

```

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

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

```

1 * [
  2   {
  3     Deaths: 1
  4   }
]

```

Two document samples are shown:

- Document 1: `_id: ObjectId("5f1f71465a08422dc849e02b")`, Country: "Barbados", Cases: 12, Deaths: 8, Update Time: "7/6/2009 9:00", date: 2009-07-06T09:00:00.000+00:00
- Document 2: `_id: ObjectId("5f1f71465a08422dc849e02d")`, Country: "Bermuda, UKOT", Cases: 1, Deaths: 0, Update Time: "7/6/2009 9:00", date: 2009-07-06T09:00:00.000+00:00

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

```

1 1

```

One document sample is shown:

- Document 1: `_id: ObjectId("5f1f71465a08422dc849e023")`, Country: "Algeria", Cases: 5, Deaths: 0, Update Time: "7/6/2009 9:00", date: 2009-07-06T09:00:00.000+00:00

3. ¿Cuál fue el país con el mayor número de casos?

```

[{$match: {
  Country: {
    $ne: 'Grand Total'
  }
}, {$addFields: {
  date: {
    $dateFromString: {
      dateString: '$Update Time',
      format: '%m/%d/%Y %H:%M'
    }
  }
}}, {$sort: {
  date: -1
}}, {$sort: {
  Cases: -1
}}, {$limit: 1}]

```

The screenshot shows two stages of a MongoDB aggregation pipeline:

- Output after \$sort stage (Sample of 20 documents):**

```
1 + [
  {
    Cases: -1
  }
]
```

Two sample documents are shown:

```
_id: ObjectId("5f1f71465a08422dc849a0a1")
Country: "United States of America"
Cases: 33902
Deaths: 170
Update Time: "7/6/2009 9:00"
date: 2009-07-06T09:00:00.000+00:00
```

```
_id: ObjectId("5f1f71465a08422dc849a121")
Country: "United States of America"
Cases: 33902
Deaths: 10
Update Time: "7/3/2009 9:00"
date: 2009-07-03T09:00:00.000+00:00
```
- Output after \$limit stage (Sample of 1 document):**

```
1 + 1
```

One sample document is shown:

```
_id: ObjectId("5f1f71465a08422dc849a0a1")
Country: "United States of America"
Cases: 33902
Deaths: 170
Update Time: "7/6/2009 9:00"
date: 2009-07-06T09:00:00.000+00:00
```

4. ¿Cuál fue el país con el menor número de casos?

```
[$match: {
  Country: {
    $ne: 'Grand Total'
  }
}, {$addFields: {
  date: {
    $dateFromString: {
      dateString: '$Update Time',
      format: '%m/%d/%Y %H:%M'
    }
  }
}}, {$sort: {
  date: -1
}}, {$sort: {
  Cases: 1
}}, {$limit: 1}]
```

The screenshot shows two stages of a MongoDB aggregation pipeline:

- Output after \$sort stage (Sample of 20 documents):**

```
1 + [
  {
    Cases: 1
  }
]
```

Two sample documents are shown:

```
_id: ObjectId("5f1f71465a08422dc849a069")
Country: "Libya"
Cases: 1
Deaths: 0
Update Time: "7/6/2009 9:00"
date: 2009-07-06T09:00:00.000+00:00
```

```
_id: ObjectId("5f1f71465a08422dc849a0e7")
Country: "Latvia"
Cases: 1
Deaths: 0
Update Time: "7/6/2009 9:00"
date: 2009-07-06T09:00:00.000+00:00
```
- Output after \$limit stage (Sample of 1 document):**

```
1 + 1
```

One sample document is shown:

```
_id: ObjectId("5f1f71465a08422dc849a02d")
Country: "Bermuda, UKOT"
Cases: 1
Deaths: 0
Update Time: "7/6/2009 9:00"
date: 2009-07-06T09:00:00.000+00:00
```

5. ¿Cuál fue el número de muertes promedio?

```

[{$match: {
  Country: {$ne: "Grand Total"}
}}, {$match: {
  Deaths: {$ne: NaN}
}}, {$group: {
  _id: null,
  avgDeaths: {
    $avg: "$Deaths"
  }
}}]

```

The screenshot shows the MongoDB aggregation pipeline interface. It consists of two main sections: the top section displays the pipeline stages, and the bottom section shows the output documents.

Top Section:

- \$match:** A dropdown menu is open, showing the query in MQL: `1 * /* query: The query in MQL.
2 * /
3 * {
4 * Deaths: {$ne: NaN}
5 * }`.
- Output after \$match stage:** (Sample of 20 documents) Shows two sample documents:
 - `_id: ObjectId("5f1f71465a08422dc849a023")
Country: "Algeria"
Cases: 5
Deaths: 0
Update Time: "7/6/2009 9:00"`
 - `_id: ObjectId("5f1f71465a08422dc849a024")
Country: "Antigua and Barbuda"
Cases: 2
Deaths: 0
Update Time: "7/6/2009 9:00"`

Bottom Section:

- \$group:** A dropdown menu is open, showing the group definition in MQL: `1 * /*
2 * _id: The id of the group.
3 * fieldN: The first field name.
4 * */
5 * {
6 * _id: null,
7 * avgDeaths: {
8 * $avg: "$Deaths"
9 * }
10 * }`.
- Output after \$group stage:** (Sample of 1 document) Shows one grouped document:
 - `_id: null
avgDeaths: 2.2694444444444444`

6. ¿Cuál fue el número de casos promedio?

```

[{$match: {
  Country: {$ne: "Grand Total"}
}}, {$group: {
  _id: null,
  avgCases: {
    $avg: "$Cases"
  }
}}]

```

The screenshot shows the MongoDB aggregation pipeline interface. At the top, there are tabs for 'COLLATION' and 'Untitled- Modified'. Below the tabs, there are two stages:

- \$match stage:** Shows the query as a code block and its output as a sample of 20 documents. The output documents include fields like _id, Country, Cases, Deaths, and Update Time.
- \$group stage:** Shows the grouping code and its output as a sample of 1 document. The output document includes _id and avgCases.

7. Top 5 de países con más muertes

```
{
  filter: {
    Country: {
      $ne: 'Grand Total'
    },
    Deaths: {
      $ne: NaN
    }
  },
  sort: {
    Deaths: -1
  },
  limit: 5
}
```

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

- Filter:** {Country: { \$ne: "Grand Total" }, Deaths: { \$ne: NaN }}
- Options:** MAXTIMEMS 5000
- Sort:** {Deaths: -1}
- Collation:**
- Find:** FIND
- Reset:** RESET
- Display:** ADD DATA, VIEW, LIST, GRID, DETAILS
- Results:** Displaying documents 1 - 5 of 5

The results are as follows:

- Document 1:** _id: ObjectId("5f1f71465a08422dc849a0e1")
Country: "United States of America"
Cases: 33902
Deaths: 170
Update Time: "7/6/2009 9:00"
- Document 2:** _id: ObjectId("5f1f71465a08422dc849a121")
Country: "United States of America"
Cases: 33902
Deaths: 170
Update Time: "7/3/2009 9:00"
- Document 3:** _id: ObjectId("5f1f71465a08422dc849a19a")
Country: "United States of America"
Cases: 27717
Deaths: 127
Update Time: "7/1/2009 9:00"
- Document 4:** _id: ObjectId("5f1f71475a08422dc849a20f")
Country: "United States of America"
Cases: 27717
Deaths: 127
Update Time: "6/29/2009 9:00"

8. Top 5 de países con menos muertes

```
{
  filter: {
    Country: {
      $ne: 'Grand Total'
    },
    Deaths: {
      $ne: NaN
    }
  },
  sort: {
    Deaths: 1
  },
  limit: 5
}
```

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

- Filter:** {Country: { \$ne: "Grand Total" }, Deaths: {\$ne: NaN}}
- Project:** (none)
- Sort:** {Deaths: 1}
- MaxTimeMS:** 5000
- Collation:** (none)
- Options:** Skip 0, Limit 5

The results display five documents:

- Algeria:** _id: ObjectId("5f1f71465a08422dc849a023"), Country: "Algeria", Cases: 5, Deaths: 0, Update Time: "7/6/2009 9:00"
- Antigua and Barbuda:** _id: ObjectId("5f1f71465a08422dc849a024"), Country: "Antigua and Barbuda", Cases: 2, Deaths: 0, Update Time: "7/6/2009 9:00"
- Austria:** _id: ObjectId("5f1f71465a08422dc849a027"), Country: "Austria", Cases: 19, Deaths: 0, Update Time: "7/6/2009 9:00"
- Bahamas:** _id: ObjectId("5f1f71465a08422dc849a028"), Country: "Bahamas", Cases: 7, Deaths: 0, Update Time: "7/6/2009 9:00"

Reto 3-Covid

1. ¿Cuál es país con mayor número de casos?

```
[{$addFields: {
  casos: {$convert: {input: "$Confirmed", to: 16}}}

}, {$group: {
  _id: "$Region",
  Cases: {
    $sum: "$casos"
  }
}}, {$sort: {
  Cases: -1
}}, {$limit: 1}]
```

The screenshot shows a MongoDB pipeline editor with two stages:

- \$sort:** Sorts by Cases in descending order.
- \$limit:** Limits the output to 1 document.

The output after the \$sort stage shows a sample of 20 documents, all from Mainland China with 2369152 cases. The output after the \$limit stage shows a single document from Mainland China with 2369152 cases.

2. ¿Cuál es el país con mayor número de muertes?

```
[$addFields: {  
    muertes: {$convert: {input: "$Deaths", to: 16}}  
}, {$group: {  
    _id: "$Region",  
    Deaths: {  
        $sum: "$muertes"  
    }  
}, {$sort: {  
    Deaths: -1  
}}, {$limit: 1}]
```

The screenshot shows the MongoDB Compass interface with an aggregation pipeline. The pipeline consists of four stages:

- \$addFields**: Adds a new field `muertes` by converting the `Deaths` field from a string to a double.
- \$group**: Groups documents by the `_id` field (`"$Region"`) and calculates the sum of `muertes` for each group.
- \$sort**: Sorts the grouped documents by the `Deaths` field in descending order.
- \$limit**: Limits the results to a single document.

The output of the pipeline is a single document:

```
_id: "Mainland China"  
Deaths: 65325
```

3. Usando las coordenadas, encuentra el epicentro del virus.

```
[$match: {  
    $and:  
        [{Lat: {$ne: ""}},  
         {Long: {$ne: ""}}]  
}, {$addFields: {  
    Lat: {$convert: {input: "$Lat", to: "double"}},  
    Long: {$convert: {input: "$Long", to: "double"}}  
}, {$group: {  
    _id: null,  
    meanLat: {  
        $avg: "$Lat"  
    },  
    meanLong: {  
        $avg: "$Long"  
    }  
}}]
```

```

$avg: "$Long"
}
}}]

```

The screenshot shows the MongoDB Compass interface with two stages displayed:

- \$addFields Stage:** Shows the code `{\$addFields: {Lat: {\$convert: {input: "\$Lat", to: "double"}}, Long: {\$convert: {input: "\$Long", to: "double"}}, ...}}` and its output, which includes fields like _id, Date, Province, Region, Last Update, Confirmed, Deaths, and Recovered.
- \$group Stage:** Shows the code `{\$group: {_id: null, meanLat: {\$avg: "\$Lat"}, meanLong: {\$avg: "\$Long"}, ...}}` and its output, which includes _id (null), meanLat (31.709998321342923), and meanLong (29.72109808153477).

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

```

[{$match: {
  $and:
  [{Lat: {$ne: ""}}, {Long: {$ne: ""}}]
}}, {$group: {
  _id: "$Region",
  Lat: {
    $max: "$Lat"
  },
  Long: {
    $max: "$Long"
  }
}}, {$addFields: {
  Lat: {$convert: {input: "$Lat", to: "double"}}
}}
]

```

```

    Long: {$convert: {input: "$Long", to: "double"}}
}, {$match: {
    $and: [{Lat: {$gte: 21}},
    {Lat: {$gte: 35}},
    {Long: {$gte: 30}},
    {Long: {$gte: 45}}]
}}, {$limit: 5}, {$project: {
    _id: 1
}}]

```

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

- \$limit Stage:**
 - Configuration: Shows a dropdown set to '\$limit' and a toggle switch.
 - Output: Shows a sample of 5 documents. The first document is '_id: "Mainland China", Lat: 47.862, Long: 95.9956'. The second document is '_id: "Japan", Lat: 36, Long: 138'.
- \$project Stage:**
 - Configuration: Shows a dropdown set to '\$project' and a toggle switch.
 - Output: Shows a sample of 5 documents. The first document is '_id: "Australia"'. The second document is '_id: "US"'.