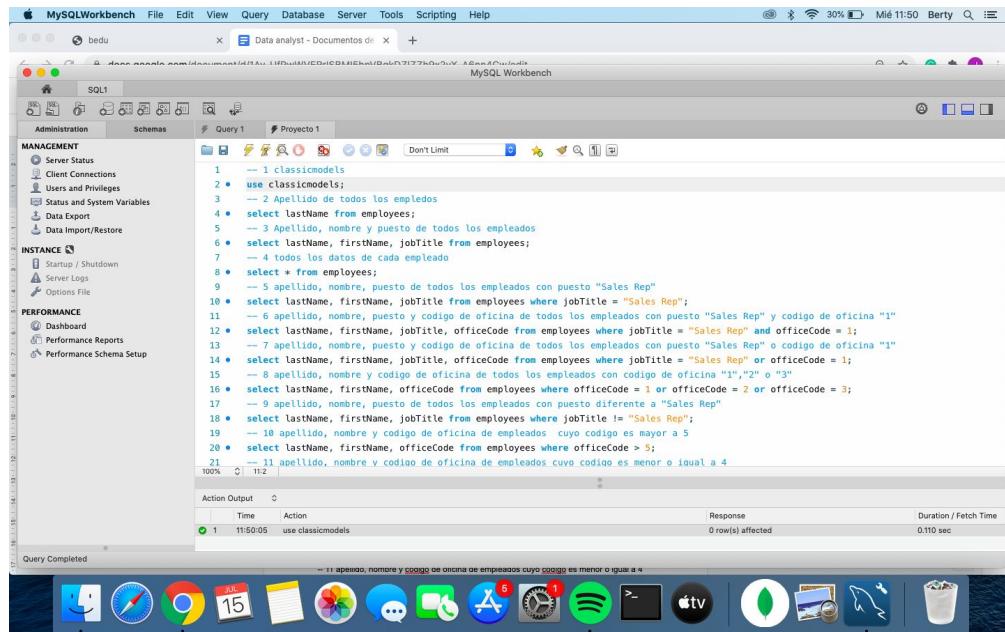


Sesión 01 Fundamentos SQL

Proyecto 1

-- 1 classicmodels

use classicmodels;

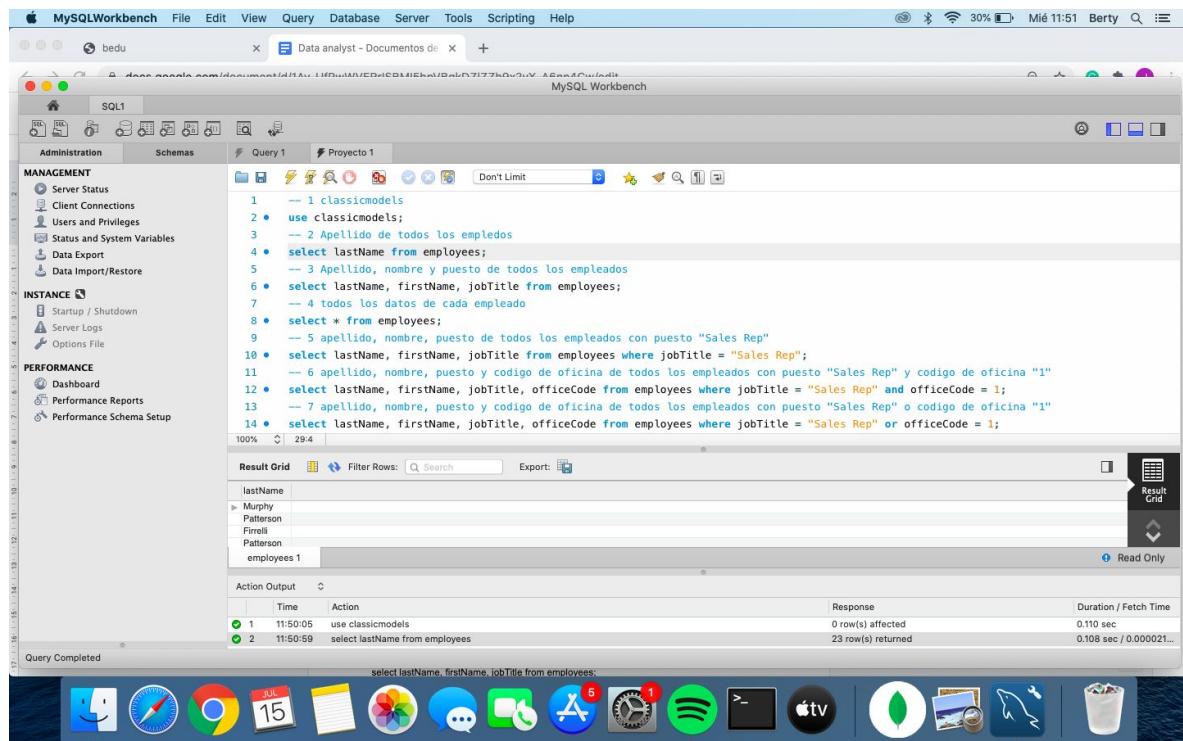


```
MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
bedu Data analyst - Documentos de + MySQL Workbench
SQL1 Query 1 Proyecto 1
ADMINISTRATION
MANAGEMENT
INSTANCE
PERFORMANCE
Schemas
Query 1
1 -- 1 classicmodels
2 • use classicmodels;
3 -- 2 Apellido de todos los empleados
4 • select lastName from employees;
5 -- 3 Apellido, nombre y puesto de todos los empleados
6 • select lastName, firstName, jobTitle from employees;
7 -- 4 todos los datos de cada empleado
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
13 -- 7 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" o codigo de oficina "1"
14 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1;
15 -- 8 apellido, nombre y codigo de oficina de todos los empleados con codigo de oficina "1","2" o "3"
16 • select lastName, firstName, officeCode from employees where officeCode = 1 or officeCode = 2 or officeCode = 3;
17 -- 9 apellido, nombre, puesto de todos los empleados con puesto diferente a "Sales Rep"
18 • select lastName, firstName, jobTitle from employees where jobTitle != "Sales Rep";
19 -- 10 apellido, nombre y codigo de oficina de empleados cuyo codigo es mayor a 5
20 • select lastName, firstName, officeCode from employees where officeCode > 5;
21 -- 11 apellido, nombre y codigo de oficina de empleados cuyo codigo es menor o igual a 4
100% ⌂ 11:2
Action Output
Time Action Response Duration / Fetch Time
1 11:50:05 use classicmodels 0 row(s) affected 0.10 sec
Query Completed
11 apellido, nombre y codigo de oficina de empleados cuyo codigo es menor o igual a 4

```

-- 2 Apellido de todos los empleados

select lastName from employees;



```
MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
bedu Data analyst - Documentos de + MySQL Workbench
SQL1 Query 1 Proyecto 1
ADMINISTRATION
MANAGEMENT
INSTANCE
PERFORMANCE
Schemas
Query 1
1 -- 1 classicmodels
2 • use classicmodels;
3 -- 2 Apellido de todos los empleados
4 • select lastName from employees;
5 -- 3 Apellido, nombre y puesto de todos los empleados
6 • select lastName, firstName, jobTitle from employees;
7 -- 4 todos los datos de cada empleado
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
13 -- 7 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" o codigo de oficina "1"
14 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1;
100% ⌂ 29:4
Result Grid Filter Rows: Q Search Export: E
lastName
Murphy
Patterson
Fjorrell
Patterson
employees 1
Read Only
Action Output
Time Action Response Duration / Fetch Time
1 11:50:05 use classicmodels 0 row(s) affected 0.10 sec
2 11:50:59 select lastName from employees 23 row(s) returned 0.108 sec / 0.000021...
Query Completed
select lastName, firstName, jobTitle from employees;

```

-- 3 Apellido, nombre y puesto de todos los empleados

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

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor window displays the following query and its execution results:

```
1 -- 1 classicmodels
2 • use classicmodels;
3 -- 2 Apellido de todos los empleados
4 • select lastName from employees;
5 -- 3 Apellido, nombre y puesto de todos los empleados
6 • select lastName, firstName, jobTitle from employees;
7 -- 4 todos los datos de cada empleado
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
```

The Result Grid shows the following data:

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

The Action Output table shows the execution details:

Action	Time	Response	Duration / Fetch Time
select lastName from employees	11:50:59	23 row(s) returned	0.108 sec / 0.000021...
select lastName, firstName, jobTitle from employees	11:54:13	23 row(s) returned	0.115 sec / 0.000025...

-- 4 todos los datos de cada empleado

```
select * from employees;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor window displays the following query and its execution results:

```
1 -- 1 classicmodels
2 • use classicmodels;
3 -- 2 Apellido de todos los empleados
4 • select lastName from employees;
5 -- 3 Apellido, nombre y puesto de todos los empleados
6 • select lastName, firstName, jobTitle from employees;
7 -- 4 todos los datos de cada empleado
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
```

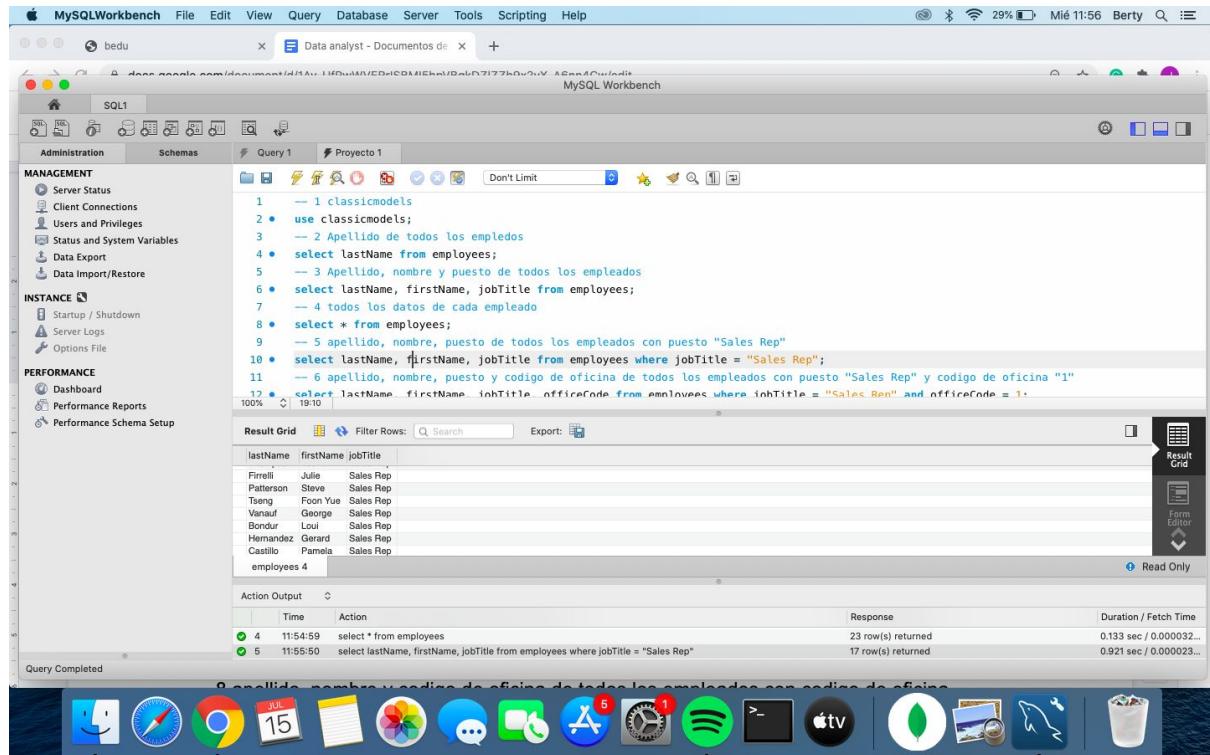
The Result Grid shows the following data:

employeeNumber	lastName	firstName	extension	email	officeCode	reportsTo	jobTitle
1076	Firelli	Jeff	x0273	jfirelli@classicmodelcars.com	1	1002	VP Marketing
1088	Patterson	William	x4871	wpatterson@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	Firelli	Julie	x2173	jfirelli@classicmodelcars.com	2	1143	Sales Rep

The Action Output table shows the execution details:

Action	Time	Response	Duration / Fetch Time
select lastName, firstName, jobTitle from employees	11:54:13	23 row(s) returned	0.115 sec / 0.000025...
select * from employees	11:54:59	23 row(s) returned	0.133 sec / 0.000032...

-- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
 select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";



The screenshot shows the MySQL Workbench interface with the following details:

- SQL Editor:** Contains the following SQL code:


```

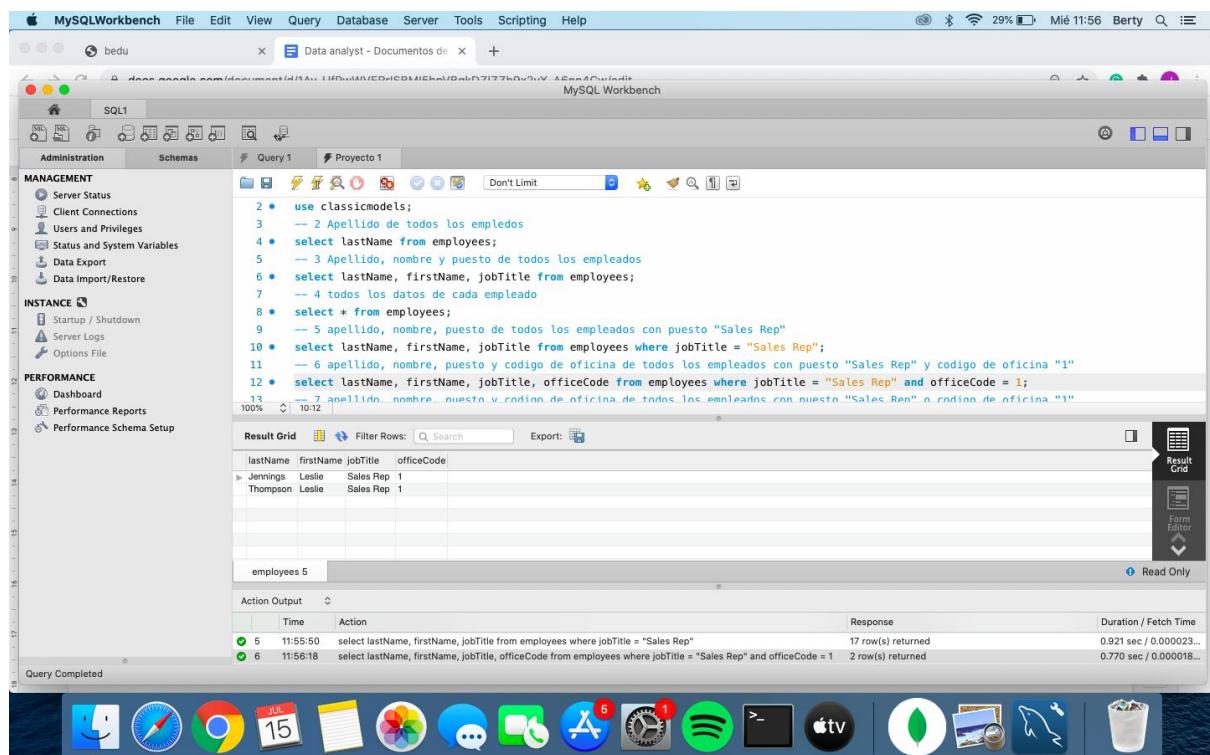
1 -- 1 classicmodels
2 • use classicmodels;
3 -- 2 Apellido de todos los empleados
4 • select lastName from employees;
5 -- 3 Apellido, nombre y puesto de todos los empleados
6 • select lastName, firstName, jobTitle from employees;
7 -- 4 todos los datos de cada empleado
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
        
```
- Result Grid:** Displays the results of the last query (line 12). The table has columns: lastName, firstName, jobTitle, and officeCode. The data is:

lastName	firstName	jobTitle	officeCode
Firelli	Julie	Sales Rep	
Patterson	Steve	Sales Rep	
Tseng	Foon Yue	Sales Rep	
Vaauw	George	Sales Rep	
Bondur	Loui	Sales Rep	
Hernandez	Gerard	Sales Rep	
Castillo	Pamela	Sales Rep	
- Action Output:** Shows two log entries:

Time	Action	Response	Duration / Fetch Time
11:54:59	select * from employees	23 row(s) returned	0.133 sec / 0.000032...
11:55:50	select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep"	17 row(s) returned	0.921 sec / 0.000023...

-- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"

select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;



The screenshot shows the MySQL Workbench interface with the following details:

- SQL Editor:** Contains the following SQL code:


```

2 • use classicmodels;
3 -- 2 Apellido de todos los empleados
4 • select lastName from employees;
5 -- 3 Apellido, nombre y puesto de todos los empleados
6 • select lastName, firstName, jobTitle from employees;
7 -- 4 todos los datos de cada empleado
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
        
```
- Result Grid:** Displays the results of the last query (line 12). The table has columns: lastName, firstName, jobTitle, and officeCode. The data is:

lastName	firstName	jobTitle	officeCode
Jennings	Leslie	Sales Rep	1
Thompson	Leslie	Sales Rep	1
- Action Output:** Shows two log entries:

Time	Action	Response	Duration / Fetch Time
11:55:50	select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep"	17 row(s) returned	0.921 sec / 0.000023...
11:56:18	select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1	2 row(s) returned	0.770 sec / 0.000018...

-- 7 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" o codigo de oficina "1"

```
select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The main window displays a SQL query in the Query Grid:

```
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
13 -- 7 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" o codigo de oficina "1"
14 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1;
15 -- 8 apellido, nombre y codigo de oficina de todos los empleados con codigo de oficina "1", "2" o "3"
16 • select lastName, firstName, officeCode from employees where officeCode = 1 or officeCode = 2 or officeCode = 3;
17 -- 9 apellido, nombre, puesto de todos los empleados con puesto diferente a "Sales Rep"
18 • select lastName, firstName, jobTitle from employees where jobTitle != "Sales Rep";
19 -- 10 apellido, nombre y codigo de oficina de empleados cuyo codigo es mayor a 3
```

The Result Grid shows the following data:

lastName	firstName	jobTitle	officeCode
Firelli	Jeff	VP Marketing	1
Bow	Anthony	Sales Manager (NA)	1
Jennings	Leslie	Sales Rep	1
Thompson	Leslie	Sales Rep	1
Firelli	Julie	Sales Rep	2
Patterson	Steve	Sales Rep	2
Tseng	Foon Yue	Sales Rep	3

The Action Output pane shows the execution details:

Action	Time	Action	Response	Duration / Fetch Time
6	11:56:18	select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1	2 row(s) returned	0.770 sec / 0.000018...
7	11:57:25	select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1	21 row(s) returned	0.114 sec / 0.000025...

-- 8 apellido, nombre y codigo de oficina de todos los empleados con codigo de oficina "1", "2" o "3"

```
select lastName, firstName, officeCode from employees where officeCode = 1 or officeCode = 2 or officeCode = 3;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The main window displays a SQL query in the Query Grid:

```
8 • select * from employees;
9 -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11 -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
13 -- 7 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" o codigo de oficina "1"
14 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1;
15 -- 8 apellido, nombre y codigo de oficina de todos los empleados con codigo de oficina "1", "2" o "3"
16 • select lastName, firstName, officeCode from employees where officeCode = 1 or officeCode = 2 or officeCode = 3;
17 -- 9 apellido, nombre, puesto de todos los empleados con puesto diferente a "Sales Rep"
18 • select lastName, firstName, jobTitle from employees where jobTitle != "Sales Rep";
19 -- 10 apellido, nombre y codigo de oficina de empleados cuyo codigo es mayor a 3
```

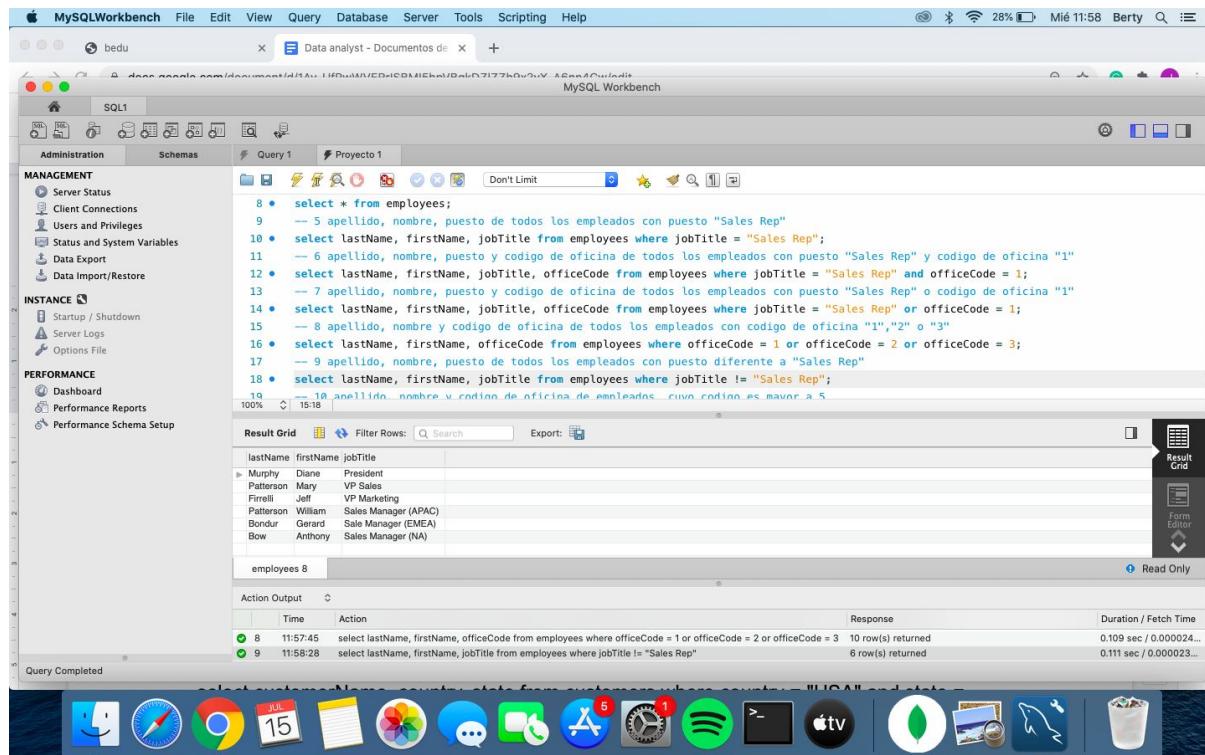
The Result Grid shows the following data:

lastName	firstName	officeCode
Firelli	Jeff	1
Bow	Anthony	1
Jennings	Leslie	1
Thompson	Leslie	1
Firelli	Julie	2
Patterson	Steve	2
Tseng	Foon Yue	3

The Action Output pane shows the execution details:

Action	Time	Action	Response	Duration / Fetch Time
7	11:57:25	select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1	21 row(s) returned	0.114 sec / 0.000025...
8	11:57:45	select lastName, firstName, officeCode from employees where officeCode = 1 or officeCode = 2 or officeCode = 3	10 row(s) returned	0.109 sec / 0.000024...

-- 9 apellido, nombre, puesto de todos los empleados con puesto diferente a "Sales Rep"
 select lastName, firstName, jobTitle from employees where jobTitle != "Sales Rep";



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The main window displays a query editor with the following SQL code:

```

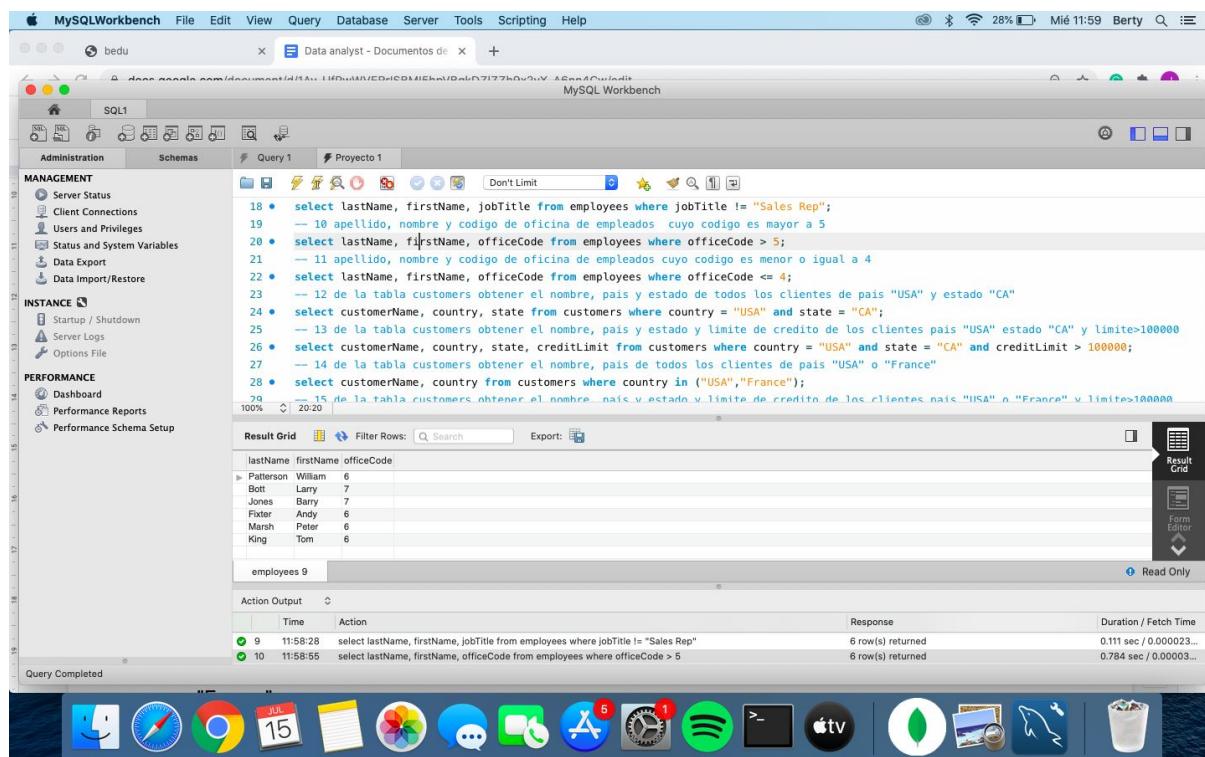
8 • select * from employees;
9   -- 5 apellido, nombre, puesto de todos los empleados con puesto "Sales Rep"
10 • select lastName, firstName, jobTitle from employees where jobTitle = "Sales Rep";
11   -- 6 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" y codigo de oficina "1"
12 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" and officeCode = 1;
13   -- 7 apellido, nombre, puesto y codigo de oficina de todos los empleados con puesto "Sales Rep" o codigo de oficina "1"
14 • select lastName, firstName, jobTitle, officeCode from employees where jobTitle = "Sales Rep" or officeCode = 1;
15   -- 8 apellido, nombre y codigo de oficina de todos los empleados con codigo de oficina "1", "2" o "3"
16 • select lastName, firstName, officeCode from employees where officeCode = 1 or officeCode = 2 or officeCode = 3;
17   -- 9 apellido, nombre, puesto de todos los empleados con puesto diferente a "Sales Rep"
18 • select lastName, firstName, jobTitle from employees where jobTitle != "Sales Rep";
19   -- 10 apellido, nombre y codigo de oficina de empleados cuyo codigo es mayor a 5
  
```

The result grid shows the following data:

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

The status bar at the bottom indicates "Query Completed".

-- 10 apellido, nombre y codigo de oficina de empleados cuyo codigo es mayor a 5
 select lastName, firstName, officeCode from employees where officeCode > 5;



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The main window displays a query editor with the following SQL code:

```

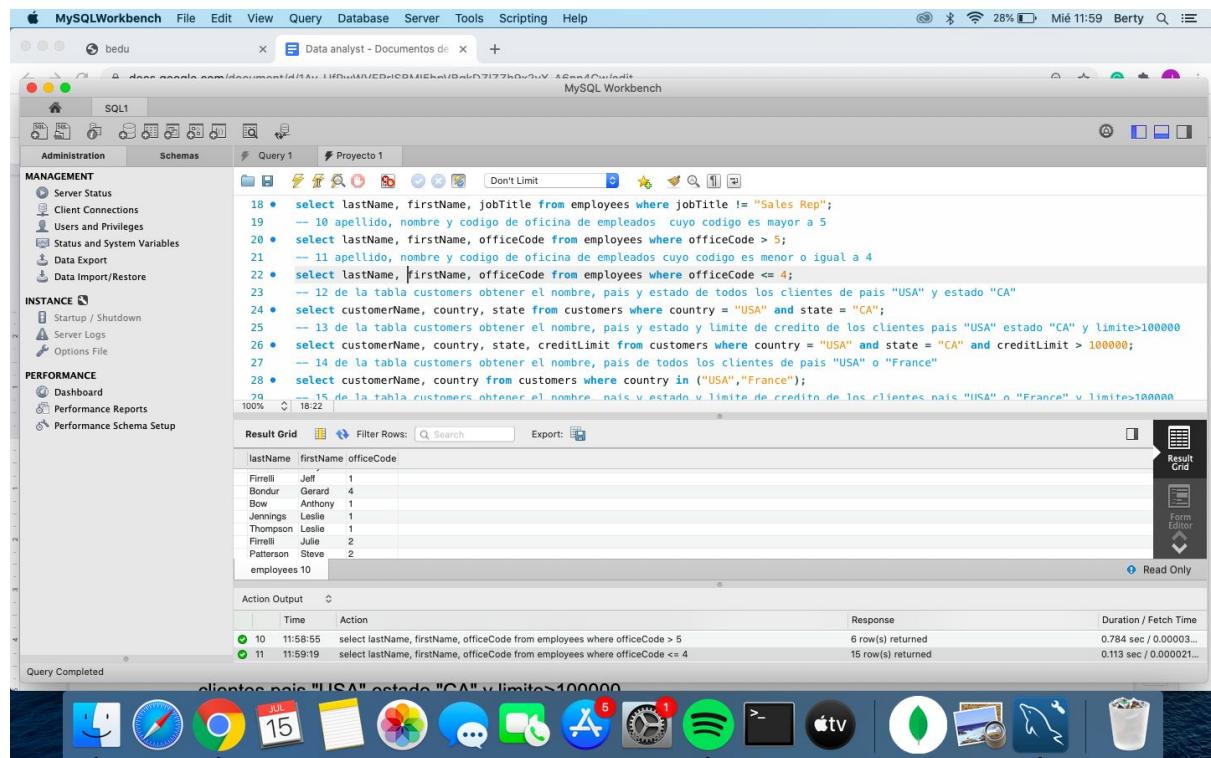
18 • select lastName, firstName, jobTitle from employees where jobTitle != "Sales Rep";
19   -- 10 apellido, nombre y codigo de oficina de empleados cuyo codigo es mayor a 5
20 • select lastName, firstName, officeCode from employees where officeCode > 5;
21   -- 11 apellido, nombre y codigo de oficina de empleados cuyo codigo es menor o igual a 4
22 • select lastName, firstName, officeCode from employees where officeCode <= 4;
23   -- 12 de la tabla customers obtener el nombre, pais y estado de todos los clientes de pais "USA" y estado "CA"
24 • select customerName, country, state from customers where country = "USA" and state = "CA";
25   -- 13 de la tabla customers obtener el nombre, pais y estado y limite de credito de los clientes pais "USA" estado "CA" y limite>100000
26 • select customerName, country, state, creditlimit from customers where country = "USA" and state = "CA" and creditlimit > 100000;
27   -- 14 de la tabla customers obtener el nombre, pais de todos los clientes de pais "USA" o "France"
28 • select customerName, country from customers where country in ("USA","France");
29   -- 15 de la tabla customers obtener el nombre, pais y estado y limite de credito de los clientes pais "USA" o "France" y limite>100000
  
```

The result grid shows the following data:

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

The status bar at the bottom indicates "Query Completed".

-- 11 apellido, nombre y codigo de oficina de empleados cuyo codigo es menor o igual a 4
 select lastName, firstName, officeCode from employees where officeCode <= 4;



```

MySQLWorkbench  File  Edit  View  Query  Database  Server  Tools  Scripting  Help
bedu          Data analyst - Documentos de +  Mié 11:59  Berty  S

SQL1
Administration  Schemas  Query 1  Proyecto 1

MANAGEMENT
    Server Status
    Client Connections
    Users and Privileges
    Status and System Variables
    Data Export
    Data Import/Restore

INSTANCE
    Startup / Shutdown
    Server Logs
    Options File

PERFORMANCE
    Dashboard
    Performance Reports
    Performance Schema Setup

Result Grid  Filter Rows: Q Search  Export: 
lastName  firstName  officeCode
Firelli  Jeff  1
Boudur  Gerard  4
Bow  Anthony  1
Jennings  Leslie  1
Thompson  Leslie  1
Firelli  Julie  2
Patterson  Steve  2
employees 10

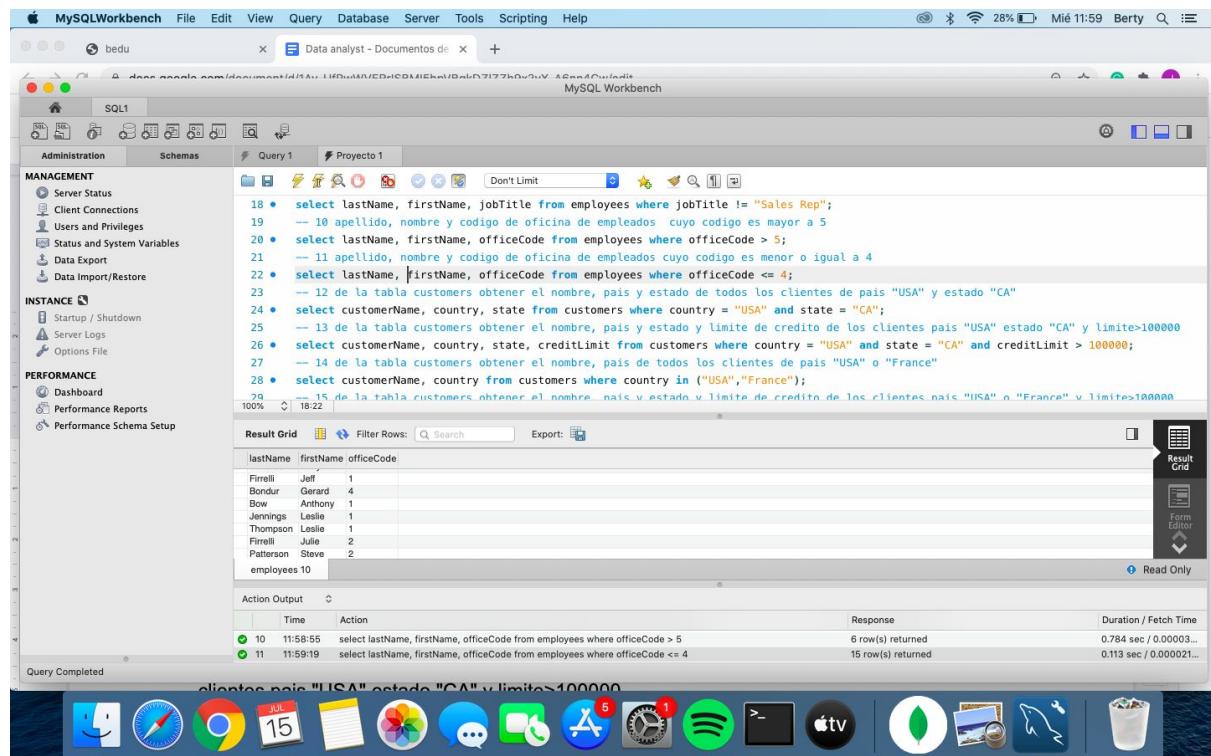
Action Output  C
Time  Action  Response  Duration / Fetch Time
10  11:58:55  select lastName, firstName, officeCode from employees where officeCode <= 4  15 row(s) returned  0.113 sec / 0.000021...
11  11:59:19  select lastName, firstName, officeCode from employees where officeCode <= 4  6 row(s) returned  0.784 sec / 0.00003...

Query Completed

```

-- 12 de la tabla customers obtener el nombre, pais y estado de todos los clientes de pais "USA" y estado "CA"

select customerName, country, state from customers where country = "USA" and state = "CA";



```

MySQLWorkbench  File  Edit  View  Query  Database  Server  Tools  Scripting  Help
bedu          Data analyst - Documentos de +  Mié 11:59  Berty  S

SQL1
Administration  Schemas  Query 1  Proyecto 1

MANAGEMENT
    Server Status
    Client Connections
    Users and Privileges
    Status and System Variables
    Data Export
    Data Import/Restore

INSTANCE
    Startup / Shutdown
    Server Logs
    Options File

PERFORMANCE
    Dashboard
    Performance Reports
    Performance Schema Setup

Result Grid  Filter Rows: Q Search  Export: 
lastName  firstName  officeCode
Firelli  Jeff  1
Boudur  Gerard  4
Bow  Anthony  1
Jennings  Leslie  1
Thompson  Leslie  1
Firelli  Julie  2
Patterson  Steve  2
employees 10

Action Output  C
Time  Action  Response  Duration / Fetch Time
10  11:58:55  select lastName, firstName, officeCode from employees where officeCode <= 4  15 row(s) returned  0.784 sec / 0.00003...
11  11:59:19  select lastName, firstName, officeCode from employees where officeCode <= 4  6 row(s) returned  0.113 sec / 0.000021...

Query Completed

```

-- 13 de la tabla customers obtener el nombre, pais y estado y limite de credito de los clientes pais "USA" estado "CA" y limite>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	106000.00
Corporate Gift Ideas Co.	USA	CA	105000.00

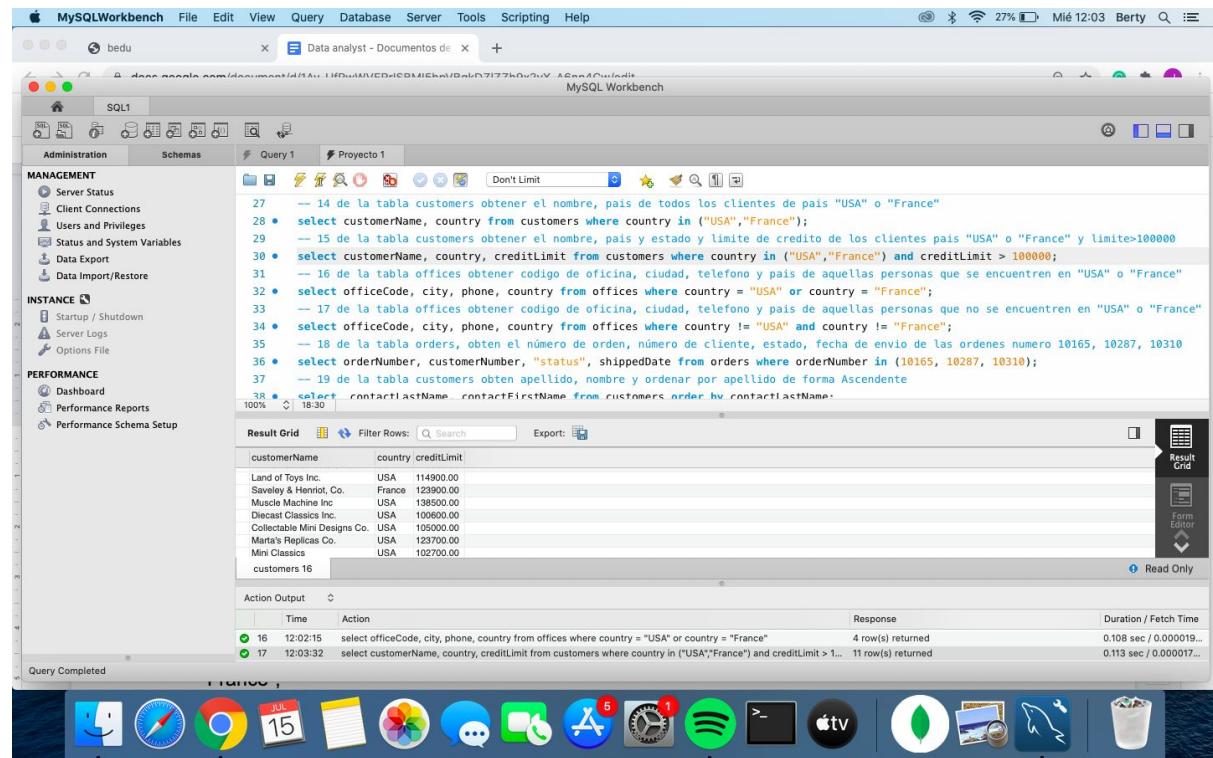
-- 14 de la tabla customers obtener el nombre, pais de todos los clientes de pais "USA" o "France"

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

customerName	country
La Rochelle Gifts	France
Mini Gifts Distributors Ltd.	USA
Mini World Co.	USA
Land of Toys Inc.	USA
Saviley & Henriot Co.	France
Muscle Machine Inc	USA
Diecast Classics Inc.	USA
Mini Gifts Distributors Ltd.	USA
Collectable Mini Designs Co.	USA
Corporate Gift Ideas Co.	USA
Mini Gifts Distributors Ltd.	USA
Collectable Mini Designs Co.	USA
Corporate Gift Ideas Co.	USA
Mini Gifts Distributors Ltd.	USA
Collectable Mini Designs Co.	USA
Corporate Gift Ideas Co.	USA

-- 15 de la tabla customers obtener el nombre, pais y estado y limite de credito de los clientes pais "USA" o "France" y limite>100000

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



```

MySQLWorkbench  File Edit View Query Database Server Tools Scripting Help
bedu          Data analyst - Documentos de + 
MySQL Workbench

SQL1
Query 1  Proyecto 1

MANAGEMENT
ADMINISTRATION
Schemas
Query Grid  Filter Rows: Q Search Export: 
Result Grid  Form Editor
customerName  country  creditLimit
Land of Toys Inc.  USA  114900.00
Saveley & Heniot, Co.  France  123900.00
Muscle Machine Inc.  USA  138500.00
Discast Classics Inc.  USA  109000.00
Collectors Mini Designs Co.  USA  109000.00
Marta's Replicas Co.  USA  123700.00
Min Classics  USA  102700.00
customers 16

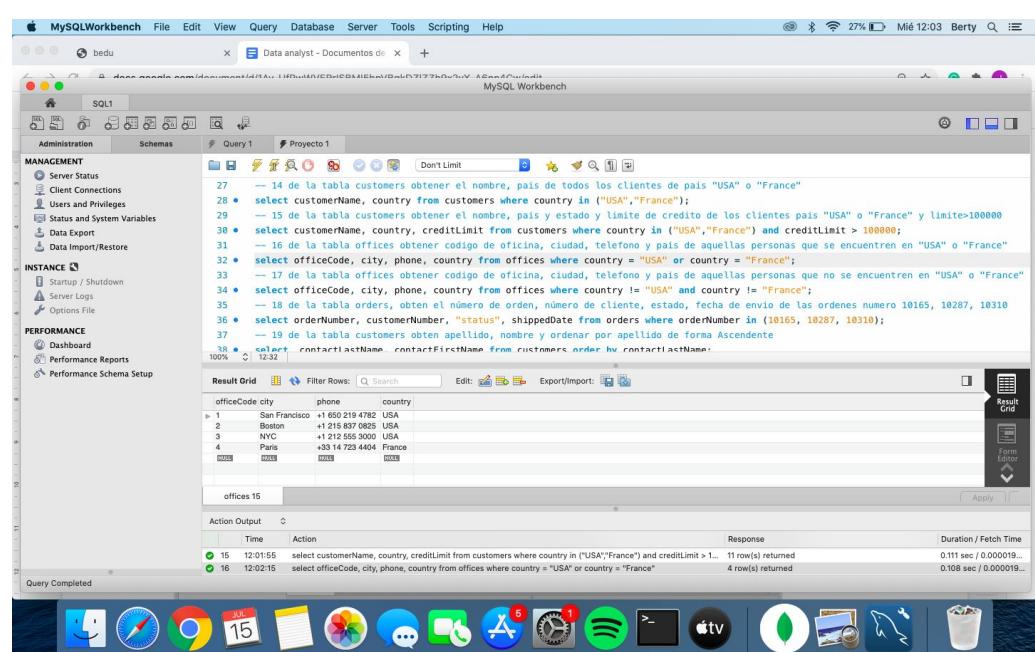
Action Output
Time Action Response Duration / Fetch Time
16 12:02:15 select officeCode, city, phone, country from offices where country = "USA" or country = "France" 4 row(s) returned 0.108 sec / 0.000019...
17 12:03:32 select customerName, country, creditLimit from customers where country in ("USA","France") and creditLimit > 100000; 11 row(s) returned 0.113 sec / 0.000017...
Read Only

Query Completed

```

-- 16 de la tabla offices obtener codigo de oficina, ciudad, telefono y pais de aquellas personas que se encuentren en "USA" o "France"

select officeCode, city, phone, country from offices where country = "USA" or country = "France";



```

MySQLWorkbench  File Edit View Query Database Server Tools Scripting Help
bedu          Data analyst - Documentos de + 
MySQL Workbench

SQL1
Query 1  Proyecto 1

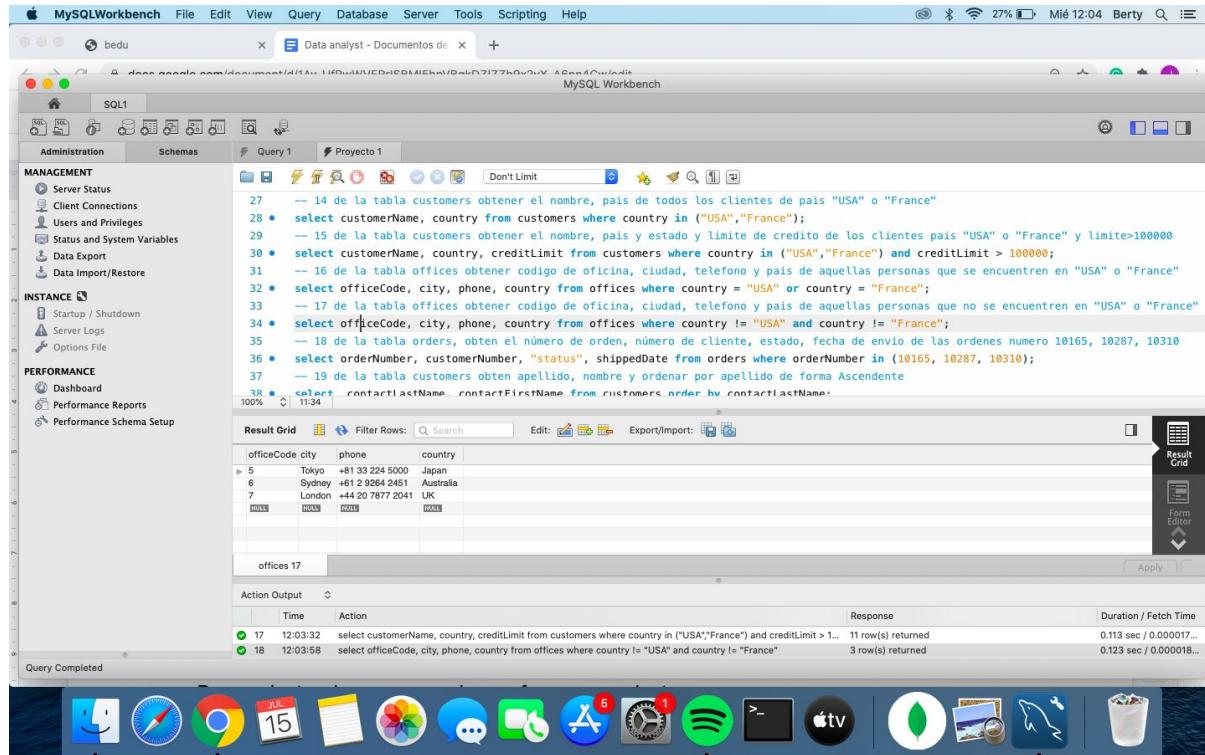
MANAGEMENT
ADMINISTRATION
Schemas
Query Grid  Filter Rows: Q Search Export: 
Result Grid  Form Editor
officeCode  city  phone  country
1 San Francisco  +1 650 510 1234  USA
2 Seattle  +1 206 553 3000  USA
3 NYC  +1 212 555 3000  USA
4 Paris  +33 14 723 4404  France
offices 15

Action Output
Time Action Response Duration / Fetch Time
15 12:01:55 select customerName, country, creditLimit from customers where country in ("USA","France") and creditLimit > 100000; 11 row(s) returned 0.111 sec / 0.000019...
16 12:02:15 select officeCode, city, phone, country from offices where country = "USA" or country = "France" 4 row(s) returned 0.108 sec / 0.000019...

```

-- 17 de la tabla offices obtener codigo de oficina, ciudad, telefono y pais de aquellas personas que no se encuentren en "USA" o "France"

select officeCode, city, phone, country from offices where country != "USA" and country != "France";



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor window displays the following query and its execution results:

```
27 -- 14 de la tabla customers obtener el nombre, pais de todos los clientes de pais "USA" o "France"
28 • select customerName, country from customers where country in ("USA","France");
29 -- 15 de la tabla customers obtener el nombre, pais y estado y limite de credito de los clientes pais "USA" o "France" y limite>100000
30 • select customerName, country, creditLimit from customers where country in ("USA","France") and creditLimit > 100000;
31 -- 16 de la tabla offices obtener codigo de oficina, ciudad, telefono y pais de aquellas personas que se encuentren en "USA" o "France"
32 • select officeCode, city, phone, country from offices where country = "USA" or country = "France";
33 -- 17 de la tabla offices obtener codigo de oficina, ciudad, telefono y pais de aquellas personas que no se encuentren en "USA" o "France"
34 • select officeCode, city, phone, country from offices where country != "USA" and country != "France";
35 -- 18 de la tabla orders, obtén el número de orden, número de cliente, estado, fecha de envío de las órdenes numero 10165, 10287, 10310
36 • select orderNumber, customerNumber, "status", shippedDate from orders where orderNumber in (10165, 10287, 10310);
37 -- 19 de la tabla customers obtener apellido, nombre y ordenar por apellido de forma Ascendente
38 • select contactLastName, contactFirstName from customers order by contactLastName;
```

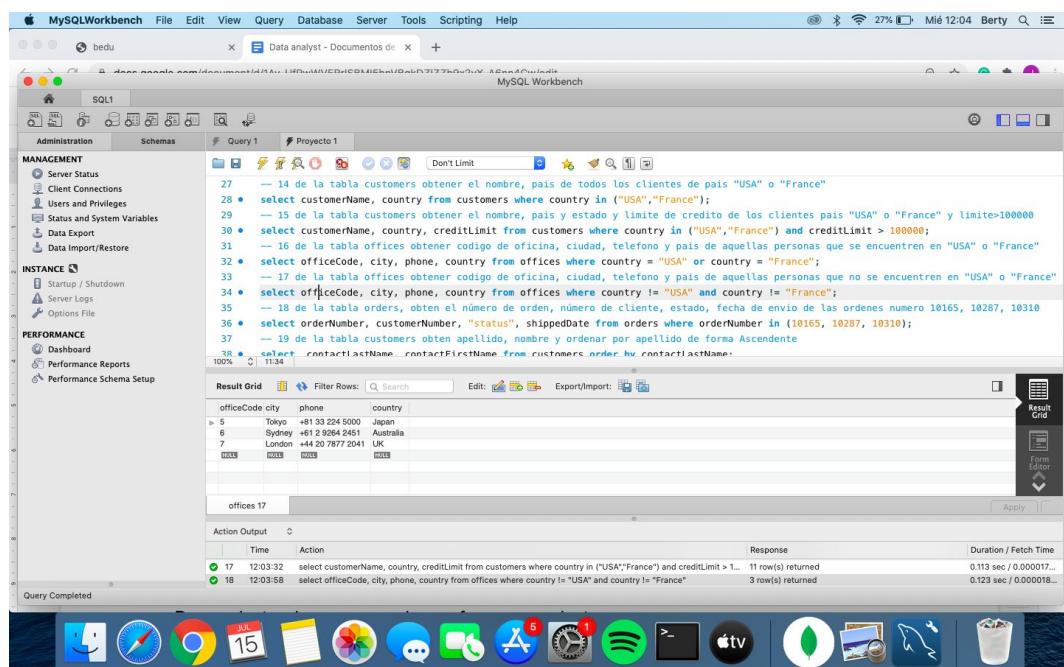
The results grid shows three rows of data:

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

The status bar at the bottom indicates "Query Completed".

-- 18 de la tabla orders, obtén el número de orden, número de cliente, estado, fecha de envío de las órdenes numero 10165, 10287, 10310

select orderNumber, customerNumber, "status", shippedDate from orders where orderNumber in (10165, 10287, 10310);



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor window displays the following query and its execution results:

```
27 -- 14 de la tabla customers obtener el nombre, pais de todos los clientes de pais "USA" o "France"
28 • select customerName, country from customers where country in ("USA","France");
29 -- 15 de la tabla customers obtener el nombre, pais y estado y limite de credito de los clientes pais "USA" o "France" y limite>100000
30 • select customerName, country, creditLimit from customers where country in ("USA","France") and creditLimit > 100000;
31 -- 16 de la tabla offices obtener codigo de oficina, ciudad, telefono y pais de aquellas personas que se encuentren en "USA" o "France"
32 • select officeCode, city, phone, country from offices where country = "USA" or country = "France";
33 -- 17 de la tabla offices obtener codigo de oficina, ciudad, telefono y pais de aquellas personas que no se encuentren en "USA" o "France"
34 • select officeCode, city, phone, country from offices where country != "USA" and country != "France";
35 -- 18 de la tabla orders, obtén el número de orden, número de cliente, estado, fecha de envío de las órdenes numero 10165, 10287, 10310
36 • select orderNumber, customerNumber, "status", shippedDate from orders where orderNumber in (10165, 10287, 10310);
37 -- 19 de la tabla customers obtener apellido, nombre y ordenar por apellido de forma Ascendente
38 • select contactLastName, contactFirstName from customers order by contactLastName;
```

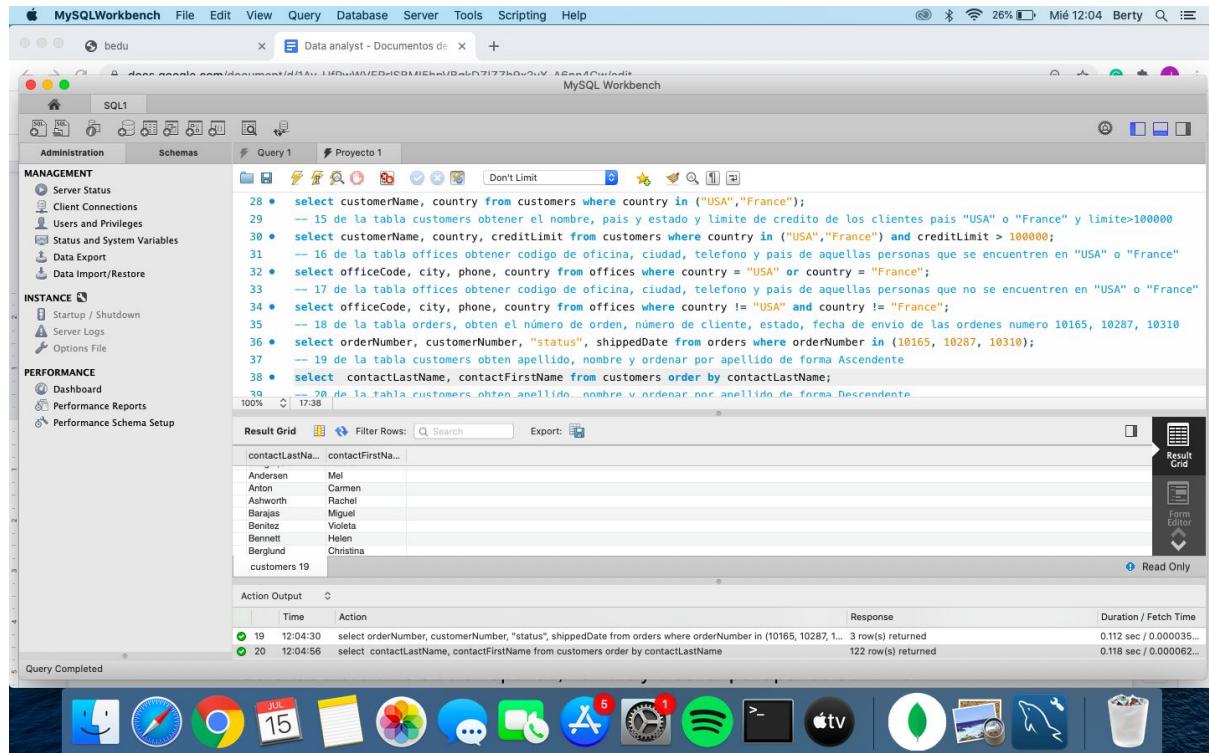
The results grid shows three rows of data:

orderNumber	customerNumber	"status"	shippedDate
10165	100017	In Progress	2003-10-03 00:00:00
10287	100018	In Progress	2003-10-03 00:00:00
10310	100019	In Progress	2003-10-03 00:00:00

The status bar at the bottom indicates "Query Completed".

-- 19 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Ascendente

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



```
28 • select customerName, country from customers where country in ("USA","France");
29   -- 15 de la tabla customers obtener el nombre, país y estado y límite de crédito de los clientes país "USA" o "France" y límite>100000;
30 • select customerName, country, creditLimit from customers where country in ("USA","France") and creditLimit > 100000;
31   -- 16 de la tabla offices obtener código de oficina, ciudad, teléfono y país de aquellas personas que se encuentren en "USA" o "France"
32 • select officeCode, city, phone, country from offices where country = "USA" or country = "France";
33   -- 17 de la tabla offices obtener código de oficina, ciudad, teléfono y país de aquellas personas que no se encuentren en "USA" o "France"
34 • select officeCode, city, phone, country from offices where country != "USA" and country != "France";
35   -- 18 de la tabla orders, obtén el número de orden, número de cliente, estado, fecha de envío de las órdenes numero 10165, 10287, 10310
36 • select orderNumber, customerNumber, "status", shippedDate from orders where orderNumber in (10165, 10287, 10310);
37   -- 19 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Ascendente
38 • select contactLastName, contactFirstName from customers order by contactLastName;
39   -- 20 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Descendente
40   100% 17:38
```

contactLastName	contactFirstName
Andersen	Mel
Anton	Carmen
Ashworth	Rachel
Barajas	Miguel
Bennet	Violeta
Bennett	Helen
Berglund	Christina

customers 19

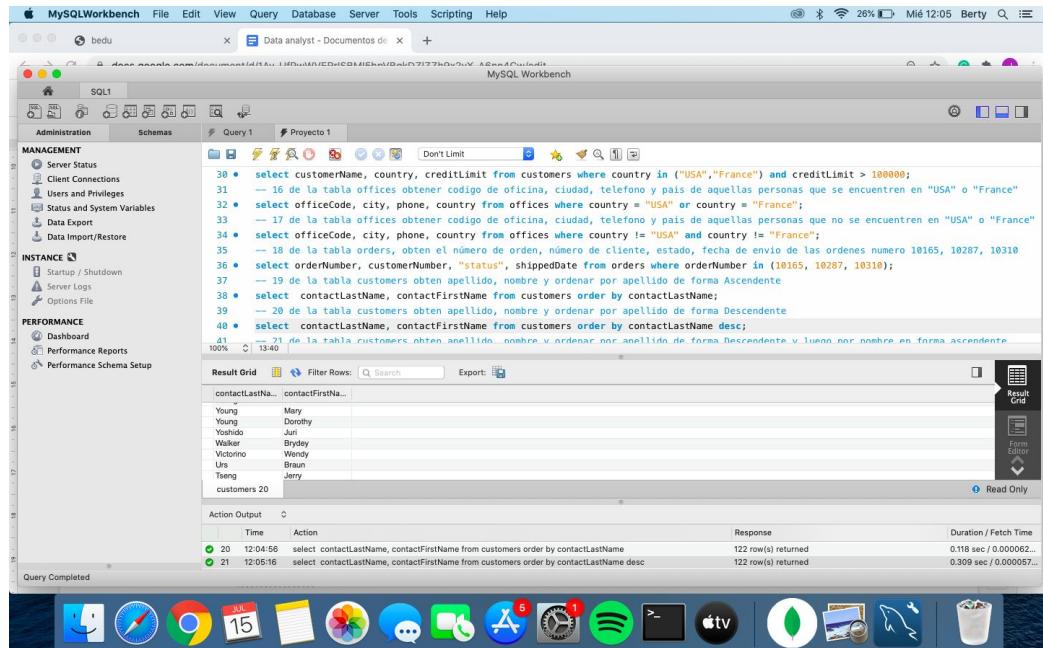
Action Output

Time	Action	Response	Duration / Fetch Time
19 12:04:30	select orderNumber, customerNumber, "status", shippedDate from orders where orderNumber in (10165, 10287, 1...	3 row(s) returned	0.112 sec / 0.000035...
20 12:04:56	select contactLastName, contactFirstName from customers order by contactLastName	122 row(s) returned	0.118 sec / 0.000062...

Query Completed

-- 20 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Descendente

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



```
30 • select customerName, country, creditLimit from customers where country in ("USA","France") and creditLimit > 100000;
31   -- 16 de la tabla offices obtener código de oficina, ciudad, teléfono y país de aquellas personas que se encuentren en "USA" o "France"
32 • select officeCode, city, phone, country from offices where country = "USA" or country = "France";
33   -- 17 de la tabla offices obtener código de oficina, ciudad, teléfono y país de aquellas personas que no se encuentren en "USA" o "France"
34 • select officeCode, city, phone, country from offices where country != "USA" and country != "France";
35   -- 18 de la tabla orders, obtén el número de orden, número de cliente, estado, fecha de envío de las órdenes numero 10165, 10287, 10310
36 • select orderNumber, customerNumber, "status", shippedDate from orders where orderNumber in (10165, 10287, 10310);
37   -- 19 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Ascendente
38 • select contactLastName, contactFirstName from customers order by contactLastName;
39   -- 20 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Descendente
40 • select contactLastName, contactFirstName from customers order by contactLastName desc;
41   100% 13:40
```

contactLastName	contactFirstName
Young	Mary
Young	Dorothy
Yoshida	Juri
Walker	Brydey
Wentzler	Wendy
Uro	Brian
Tseng	Jerry

customers 20

Action Output

Time	Action	Response	Duration / Fetch Time
20 12:04:56	select contactLastName, contactFirstName from customers order by contactLastName	122 row(s) returned	0.118 sec / 0.000062...
21 12:05:16	select contactLastName, contactFirstName from customers order by contactLastName desc	122 row(s) returned	0.309 sec / 0.000057...

Query Completed

-- 21 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Descendente y luego por nombre en forma ascendente

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

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

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The main window displays the results of a SQL query. The query was:

```
1 -- 21 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Descendente y luego por nombre en forma ascendente
2 • everything, if there is no selection FirstName from customers order by contactLastName desc;
3 • select contactLastName, contactFirstName from customers order by contactFirstName asc;
```

The results grid shows the following data:

contactLastName	contactFirstName
Young	Mary
Young	Dorothy
Yoshido	Juri
Walker	Bryony
Victorino	Wendy
Urs	Braun

Below the results grid, the Action Output pane shows two log entries:

Action	Time	Response	Duration / Fetch Time
select contactLastName, contactFirstName from customers order by contactLastName desc	26 12:07:05	122 row(s) returned	0.111 sec / 0.000058...
select contactLastName, contactFirstName from customers order by contactFirstName asc	27 12:07:06	122 row(s) returned	0.113 sec / 0.000032...

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The main window displays the results of a SQL query. The query was:

```
1 -- 21 de la tabla customers obtén apellido, nombre y ordenar por apellido de forma Descendente y luego por nombre en forma ascendente
2 • select contactLastName, contactFirstName from customers order by contactLastName desc;
3 • select contactLastName, contactFirstName from customers order by contactFirstName asc;
```

The results grid shows the following data:

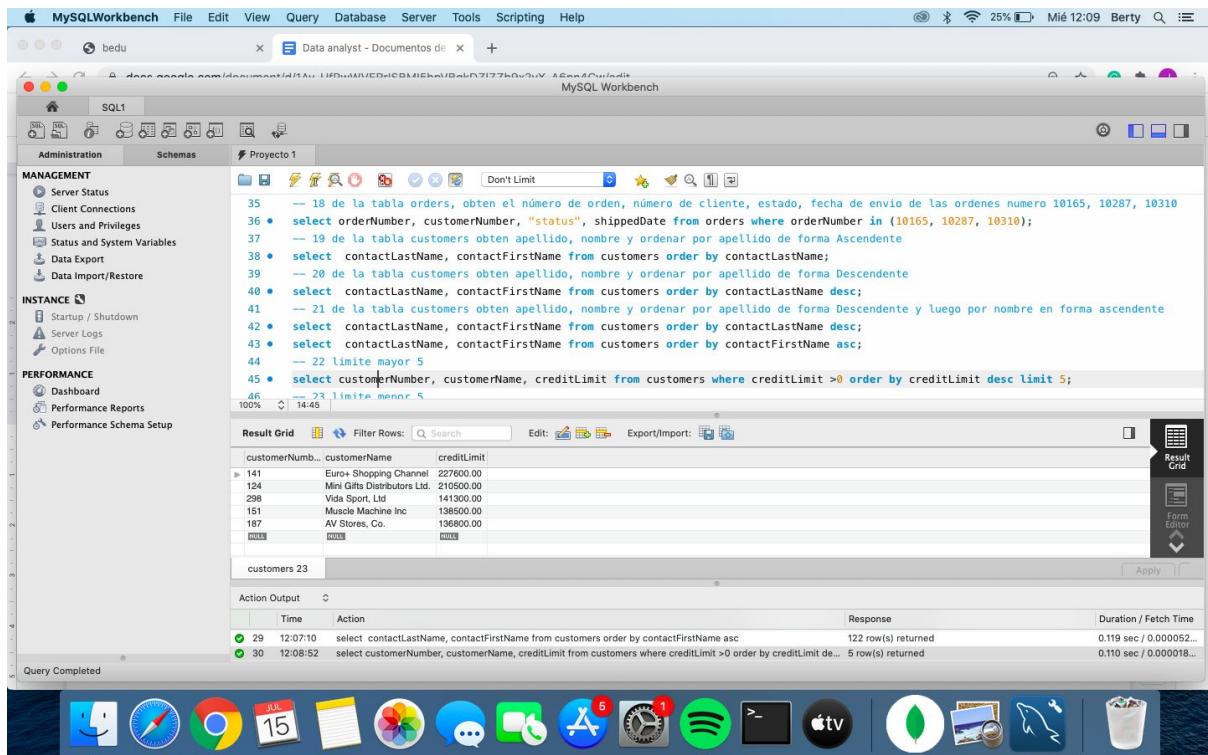
contactLastName	contactFirstName
Camino	Alejandra
Fauer	Alexander
Semenov	Alexander
Nelson	Allen
Brown	Ann
O'Hara	Anna

Below the results grid, the Action Output pane shows two log entries:

Action	Time	Response	Duration / Fetch Time
select contactLastName, contactFirstName from customers order by contactLastName desc	28 12:07:10	122 row(s) returned	0.111 sec / 0.000058...
select contactLastName, contactFirstName from customers order by contactFirstName asc	29 12:07:10	122 row(s) returned	0.113 sec / 0.000032...

-- 22 limite mayor 5

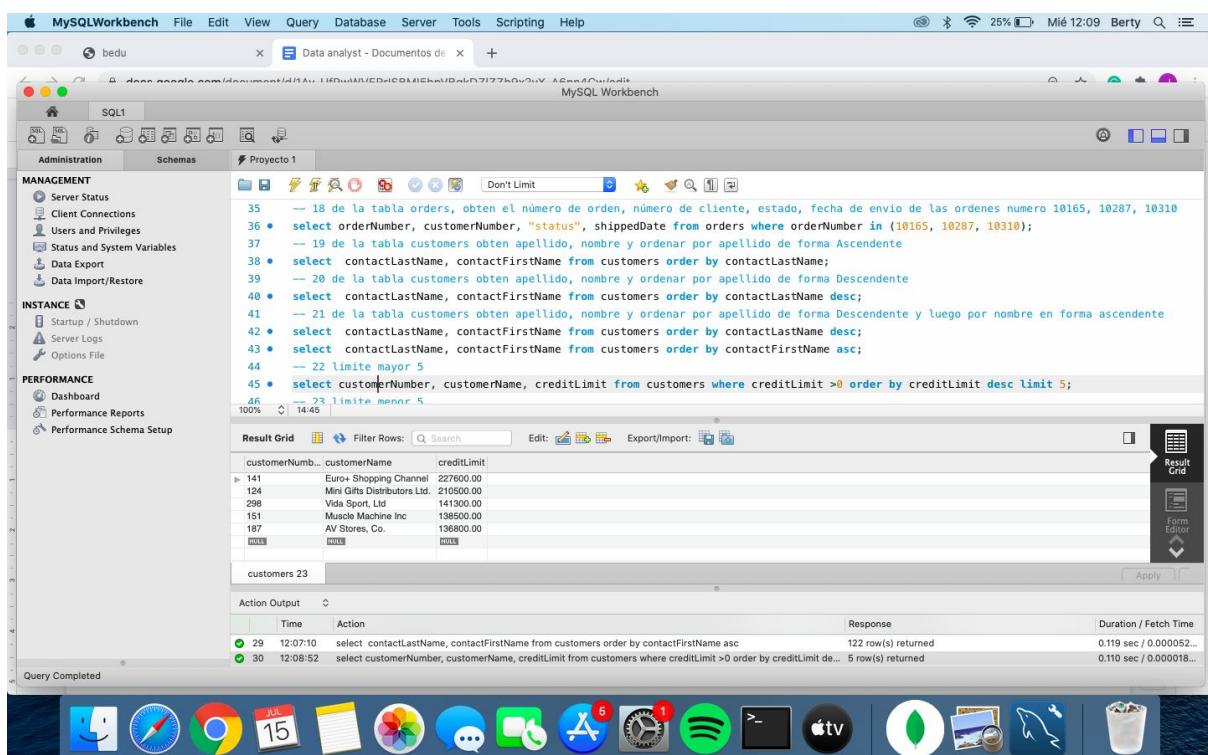
```
select customerNumber, customerName, creditLimit from customers where creditLimit >0  
order by creditLimit desc limit 5;
```



```
customerNum... 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
```

-- 23 limite menor 5

```
select customerNumber, customerName, creditLimit from customers where creditLimit >0  
order by creditLimit asc limit 5;
```



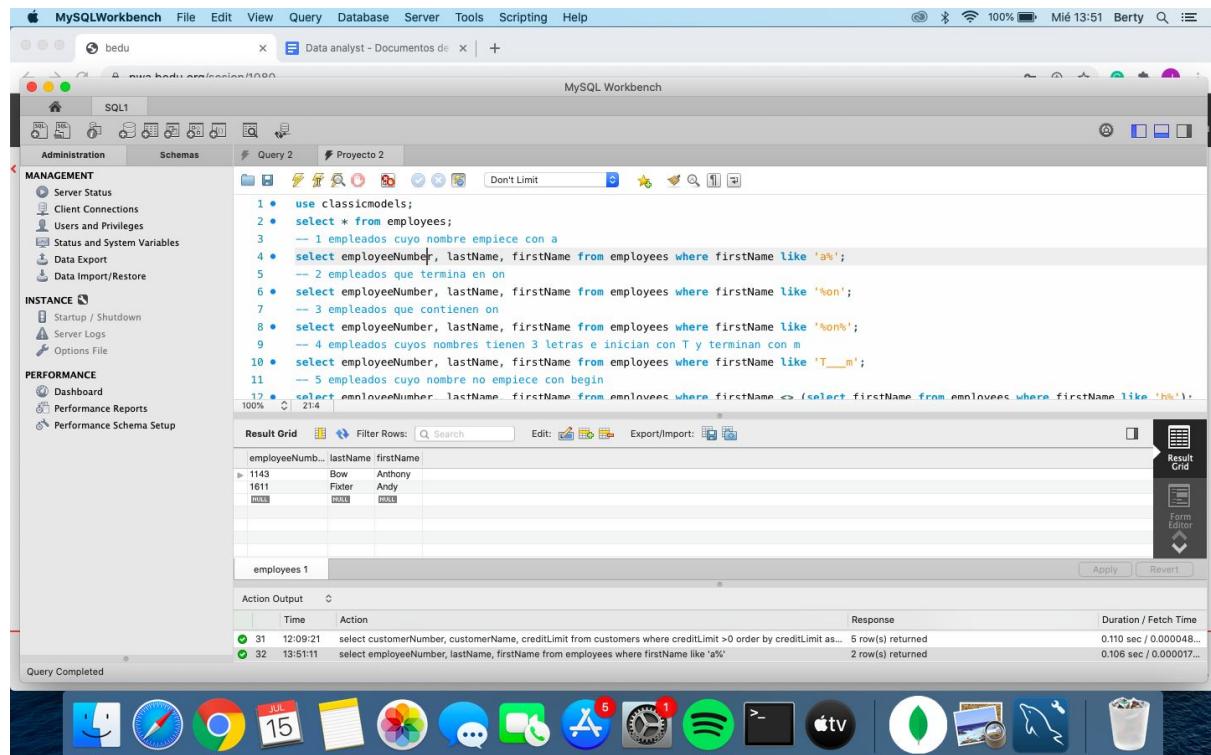
```
customerNum... customerName creditLimit
141 Euro+ Shopping Channel 227600.00
124 Mini Gifts Distributors Ltd 210500.00
298 Vida Sport, Ltd 141300.00
151 Muscle Machine Inc 138500.00
187 AV Stores, Co. 136800.00
```

Sesión 02 Agrupaciones y subconsultas

Proyecto 2

-- 1 empleados cuyo nombre empieza con a

```
select employeeNumber, lastName, firstName from employees where firstName like 'a%';
```



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The left sidebar contains navigation links for Administration, Schemas, Query 2, and Proyecto 2. The main area displays a SQL editor with the following code:

```
1 • use classicmodels;
2 • select * from employees;
3 — 1 empleados cuyo nombre empieza con a
4 • select employeeNumber, lastName, firstName from employees where firstName like 'a%';
5 — 2 empleados que terminan en on
6 • select employeeNumber, lastName, firstName from employees where firstName like '%on';
7 — 3 empleados que contienen on
8 • select employeeNumber, lastName, firstName from employees where firstName like '%on%';
9 — 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T___m';
11 — 5 empleados cuyo nombre no empieza con begin
12 • select employeeNumber, lastName, firstName from employees where firstName >= (select firstName from employees where firstName like 'b%');
100% 21:4
```

The Result Grid shows the following data:

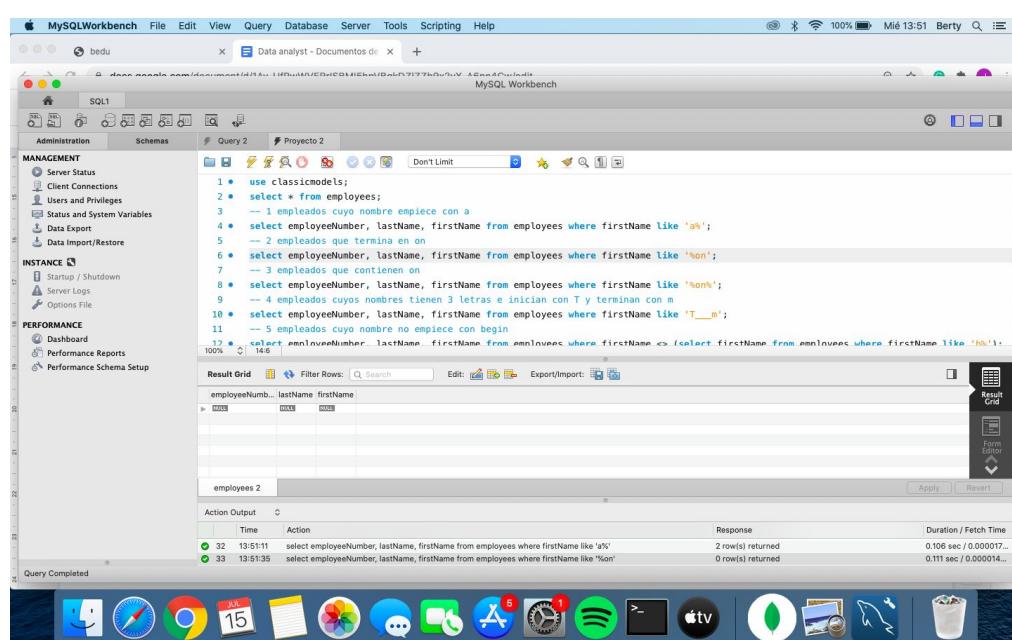
employeeNumber	lastName	firstName
1143	Bow	Anthony
1611	Fixter	Andy

The Action Output table shows two recent queries:

Action	Time	Response	Duration / Fetch Time
select customerNumber, customerName, creditLimit from customers where creditLimit > 0 order by creditLimit asc	12/09/21 12:09:21	5 row(s) returned	0.110 sec / 0.000048...
select employeeNumber, lastName, firstName from employees where firstName like 'a%'	13/51:11 13:51:11	2 row(s) returned	0.106 sec / 0.000017...

-- 2 empleados que termina en on

```
select employeeNumber, lastName, firstName from employees where firstName like '%on';
```



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The left sidebar contains navigation links for Administration, Schemas, Query 2, and Proyecto 2. The main area displays a SQL editor with the following code:

```
1 • use classicmodels;
2 • select * from employees;
3 — 1 empleados cuyo nombre empieza con a
4 • select employeeNumber, lastName, firstName from employees where firstName like 'a%';
5 — 2 empleados que termina en on
6 • select employeeNumber, lastName, firstName from employees where firstName like '%on';
7 — 3 empleados que contienen on
8 • select employeeNumber, lastName, firstName from employees where firstName like '%on%';
9 — 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T___m';
11 — 5 empleados cuyo nombre no empieza con begin
12 • select employeeNumber, lastName, firstName from employees where firstName >= (select firstName from employees where firstName like 'b%');
100% 10:56
```

The Result Grid shows the following data:

employeeNumber	lastName	firstName
1143	Bow	Anthony
1611	Fixter	Andy

The Action Output table shows two recent queries:

Action	Time	Response	Duration / Fetch Time
select employeeNumber, lastName, firstName from employees where firstName like 'a%'	13/51:11 13:51:11	2 row(s) returned	0.106 sec / 0.000017...
select employeeNumber, lastName, firstName from employees where firstName like '%on%'	13/51:35 13:51:35	0 row(s) returned	0.111 sec / 0.000014...

-- 3 empleados que contienen on

```
select employeeNumber, lastName, firstName from employees where firstName like '%on%';
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor window displays the following query and its results:

```
use classicmodels;
select * from employees;
-- 1 empleados cuyo nombre empieza con a
4 • select employeeNumber, lastName, firstName from employees where firstName like 'a%';
-- 2 empleados que terminan en on
6 • select employeeNumber, lastName, firstName from employees where firstName like '%on';
-- 3 empleados que contienen on
8 • select employeeNumber, lastName, firstName from employees where firstName like '%on%';
9 -- 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T___m';
11 -- 5 empleados cuyo nombre no empieza con begin
12 • select employeeNumber, lastName, firstName from employees where firstName >= (select firstName from employees where firstName like 'begin');
100% 14:6
```

The Result Grid shows the following data:

employeeNumber	lastName	firstName
NULL	NULL	NULL

Below the grid, the Action Output pane shows two log entries:

Action	Time	Response	Duration / Fetch Time
select employeeNumber, lastName, firstName from employees where firstName like 'a%'	32 13:51:11	2 row(s) returned	0.106 sec / 0.000017...
select employeeNumber, lastName, firstName from employees where firstName like '%on%'	33 13:51:35	0 row(s) returned	0.111 sec / 0.000014...

-- 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m

```
select employeeNumber, lastName, firstName from employees where firstName like 'T_m';
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor window displays the following query and its results:

```
use classicmodels;
select * from employees;
-- 1 empleados cuyo nombre empieza con a
4 • select employeeNumber, lastName, firstName from employees where firstName like 'a%';
-- 2 empleados que terminan en on
6 • select employeeNumber, lastName, firstName from employees where firstName like '%on';
-- 3 empleados que contienen on
8 • select employeeNumber, lastName, firstName from employees where firstName like '%on%';
9 -- 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T___m';
11 -- 5 empleados cuyo nombre no empieza con begin
12 • select employeeNumber, lastName, firstName from employees where firstName >= (select firstName from employees where firstName like 'begin');
100% 83:10
```

The Result Grid shows the following data:

employeeNumber	lastName	firstName
1619	King	Tom

Below the grid, the Action Output pane shows two log entries:

Action	Time	Response	Duration / Fetch Time
select employeeNumber, lastName, firstName from employees where firstName like 'T_m'	35 13:52:59	0 row(s) returned	0.110 sec / 0.000016...
select employeeNumber, lastName, firstName from employees where firstName like 'T_m'	36 13:54:19	1 row(s) returned	0.110 sec / 0.000016...

-- 5 empleados cuyo nombre no empiece con b

```
select employeeNumber, lastName, firstName from employees where firstName <> (select
firstName from employees where firstName like 'b%');
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor contains the following query:

```
4 • select employeeNumber, lastName, firstName from employees where firstName like 'a%';
5   -- 2 empleados que termina en on
6 • select employeeNumber, lastName, firstName from employees where firstName like '%on';
7   -- 3 empleados que contienen on
8 • select employeeNumber, lastName, firstName from employees where firstName like '%on%';
9   -- 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T_m';
11   -- 5 empleados cuyo nombre no empieza con b
12 • select employeeNumber, lastName, firstName from employees where firstName <> (select firstName from employees where firstName like 'b%');
13   -- 6 productos cuyo codigo de producto y nombre de los productos cuyo codigo incluye la cadena _20
14 • select productCode, productName from products where productCode like '%_20%';
15   -- 7 ordenes totales de cada orden
```

The Result Grid shows the following data:

employeeNumber	lastName	firstName
1002	Murphy	Diane
1056	Patterson	Mary
1076	Firelli	Jeff
1089	Patterson	William
1102	Bondur	Gerard
1143	Bow	Anthony
1165	Jennings	Leslie

Action Output shows two log entries:

Time	Action	Response	Duration / Fetch Time
13:54:19	select employeeNumber, lastName, firstName from employees where firstName like 'T_m'	1 row(s) returned	0.110 sec / 0.000016...
13:55:08	select employeeNumber, lastName, firstName from employees where firstName <> (select firstName from employees where firstName like 'b%')	22 row(s) returned	0.120 sec / 0.000022...

-- 6 productos cuyo codigo de producto y nombre de los productos cuyo codigo incluye la cadena _20

```
select productCode, productName from products where productCode like '%_20%';
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The SQL editor contains the following query:

```
4 • select employeeNumber, lastName, firstName from employees where firstName like 'a%';
5   -- 2 empleados que termina en on
6 • select employeeNumber, lastName, firstName from employees where firstName like '%on';
7   -- 3 empleados que contienen on
8 • select employeeNumber, lastName, firstName from employees where firstName like '%on%';
9   -- 4 empleados cuyos nombres tienen 3 letras e inician con T y terminan con m
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T_m';
11   -- 5 empleados cuyo nombre no empieza con b
12 • select employeeNumber, lastName, firstName from employees where firstName <> (select firstName from employees where firstName like 'b%');
13   -- 6 productos cuyo codigo de producto y nombre de los productos cuyo codigo incluye la cadena _20
14 • select productCode, productName from products where productCode like '%_20%';
15   -- 7 ordenes totales de cada orden
```

The Result Grid shows the following data:

productCode	productName
S24_2000	1960 BSA Gold Star DBD34
S24_2011	18th century schooner
S24_2022	1938 Cadillac V-16 Presidential Limousine
S24_3420	1937 Horch 930V Limousine
S24_4620	1961 Chevrolet Impala
S32_2206	1982 Ducati 996 R
S32_3207	1950's Chicago Surface Lines Streetcar

Action Output shows two log entries:

Time	Action	Response	Duration / Fetch Time
13:55:08	select employeeNumber, lastName, firstName from employees where firstName <> (select firstName from employees where firstName like 'b%')	22 row(s) returned	0.120 sec / 0.000022...
13:55:48	select productCode, productName from products where productCode like '%_20%'	10 row(s) returned	0.112 sec / 0.000030...

-- 7 orderdetails total de cada orden

```
select orderNumber, count(*) total from orderdetails group by orderNumber order by total desc;
```

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The query window contains the following code:

```
10 • select employeeNumber, lastName, firstName from employees where firstName like 'T_m';
11   -- 5 empleados cuya nombre no empieza con b
12 • select employeeNumber, lastName, firstName from employees where firstName <> (select firstName from employees where firstName like 'bs');
13   -- 6 productos cuya código de producto y nombre de los productos cuyo código incluye la cadena _20
14 • select productCode, productName from products where productCode like '%_20%';
15   -- 7 orderdetails total de cada orden
16 • select orderNumber, count(*) total from orderdetails group by orderNumber order by total desc;
17   -- 8 select orders total de números de año
18 • select (select count(orderNumber) from orders as a where orderDate like '2003%'),
19   (select count(orderNumber) from orders as b where orderDate like '2004%'),
20   (select count(orderNumber) from orders as c where orderDate like '2005%')
21   from orders ac
100% 42:16
```

The result grid shows the following data:

orderNumber	total
10165	18
10168	18
10222	18
10275	18
10316	18
10332	18
10360	18

The action output shows two rows of execution details:

Action	Time	Response	Duration / Fetch Time
38 13:56:48 select productCode, productName from products where productCode like '%_20%'		10 row(s) returned	0.112 sec / 0.000030...
39 13:56:15 select orderNumber, count(*) total from orderdetails group by orderNumber order by total desc		326 row(s) returned	0.148 sec / 0.00012...

-- 8 select orders total de numeros de año

```
select (select count(orderNumber) from orders as a where orderDate like '2003%'),
```

```
(select count(orderNumber) from orders as b where orderDate like '2004%'),
```

```
(select count(orderNumber) from orders as c where orderDate like '2005%')
```

```
from orders as e
```

```
group by "status";
```

The screenshot shows the MySQL Workbench interface with the SQL editor tab active. The query window contains the following code:

```
14 • select productCode, productName from products where productCode like '%_20%';
15   -- 7 orderdetails total de cada orden
16 • select orderNumber, count(*) total from orderdetails group by orderNumber order by total desc;
17   -- 8 select orders total de números de año
18 • select (select count(orderNumber) from orders as a where orderDate like '2003%'),
19   (select count(orderNumber) from orders as b where orderDate like '2004%'),
20   (select count(orderNumber) from orders as c where orderDate like '2005%')
21   from orders as e
22   group by "status";
23   -- 9 select empleados con oficina en USA oficinas 1, 2 y 3
24 • select lastName, firstName, officeCode from employees as a where officeCode in (select officeCode from offices where country
25   -- 10 Obten el número del cliente, número de cheque y cantidad del cliente que ha realizado el pago más alto.
26 • select amount, customerNumber, checkNumber from payments order by amount desc limit 1;
27   -- 11 Obten el número del cliente, número de cheque y cantidad de aquellos clientes cuyo pago es más alto que el promedio.
28 • select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount de
130% 24:18
```

The result grid shows the following data:

customerNumber	checkNumber	amount
111	111	111
161	161	161
64	64	64

The action output shows two rows of execution details:

Action	Time	Response	Duration / Fetch Time
39 13:57:15 select orderNumber, count(*) total from orderdetails group by orderNumber order by total desc		326 row(s) returned	0.146 sec / 0.00012...
40 13:57:53 select (select count(orderNumber) from orders as a where orderDate like '2003%'), (select count(orderNumber) from orders as b whe...		1 row(s) returned	0.116 sec / 0.000023...

-- 9 select empleados con oficina en USA oficinas 1, 2 y 3

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

The screenshot shows the SQL Developer interface with the following details:

- Schemas:** classicmodels
- Query:** Query 2 (Projeto 2*)
- Code:** The code for question 9 is visible in the editor.
- Result Grid:** Shows the results for employees in the USA, with columns: lastName, firstName, officeCode. The data includes rows for Murphy, Diana, Patterson, Mary, et al.
- Action Output:** Displays two log entries: a SELECT statement for orders and the current query for employees.

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

```
select amount, customerNumber, checkNumber from payments order by amount desc limit 1;
```

The screenshot shows the SQL Developer interface with the following details:

- Schemas:** classicmodels
- Query:** Query 2 (Projeto 2*)
- Code:** The code for question 10 is visible in the editor.
- Result Grid:** Shows the highest payment record with columns: amount, customerNumber, checkNumber. The data is a single row: 120166.58, 141, JE105477.
- Action Output:** Displays two log entries: a SELECT statement for payments and the current query for the highest payment.

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

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

Administration Schemas Query 2 Proyecto 2*

SCHEMAS

Q Filter objects

AEJ ALM AR AVH BMR CF classicmodels Tables Views Stored Proced... Functions DAJ EV JO MMS RC RL Ro SA SC SCZ SH sys tienda WH YC YCM

17 -- 8 select orders total de numeros de año
18 • select (select count(orderNumber) from orders as a where orderDate like '2003%'),
19 (select count(orderNumber) from orders as b where orderDate like '2004%'),
20 (select count(orderNumber) from orders as c where orderDate like '2005%')
21 from orders as e
22 group by "status";
23 -- 9 select empleados con oficina en USA oficinas 1, 2 y 3
24 • select lastName, firstName, officeCode from employees as a where officeCode in (select officeCode from offices where country
25 -- 10 Obten el número de cliente, número de cheque y cantidad del cliente que ha realizado el pago más alto.
26 • select amount, customerNumber, checkNumber from payments order by amount desc limit 1;
27 -- 11 Obten el número de cliente, número de cheque y cantidad de aquellos clientes cuyo pago es más alto que el promedio.
28 • select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount desc
29 -- 12 Obten el nombre de aquellos clientes que no han hecho ninguna orden.
30 • select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNumber
31 from customers as a) as Total where Total = '0'.
130% C 14:28

Result Grid Filter Rows: Search Edit Export/Import:

customerNumber	checkNumber	amount
141	JE105477	120166.58
141	ID10982	116209.49
124	KI131716	111654.40
148	KM172879	105743.00
124	AE215434	101244.59
321	DJ15149	85559.12
124	BG255406	85410.87
167	GN228846	85024.46
124	ET64396	83598.04
114	MV10015	82261.22
239	NQ865547	80375.24
323	AL493079	75020.13
***	PA14428	68571.06

payments 12

Action Output C

Time	Action	Response	Duration / Fetch Time
42 14:01:10	select amount, customerNumber, checkNumber from payments order by amount desc limit 1	1 row(s) returned	0.118 sec / 0.000018...
43 14:02:22	select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount...	134 row(s) returned	0.129 sec / 0.000060...

Query Completed

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

```
select customerName, Total from (select a.customerName, (select count(*) from orders as b  
where a.customerNumber=b.customerNumber) Total  
from customers a) as z where Total = '0';
```

SQL1

Administration Schemas

Query 2 Proyecto 2*

SCHEMAS

Q Filter objects

AEJ ALM AR AVH BMR CF

classicmodels

Tables Views Stored Proced... Functions

DAJ EV JO MMS RC RL Ro SA SC SCZ SH sys tienda WH YC YCM

17 -- 8 select orders total de numeros de año

18 • select (select count(orderNumber) from orders as a where orderDate like '2003%'),
(select count(orderNumber) from orders as b where orderDate like '2004%'),
(select count(orderNumber) from orders as c where orderDate like '2005%')

19 from orders as e
group by "status";

20 -- 9 select empleados con oficina en USA oficinas 1, 2 y 3

21 select lastName, firstName, officeCode from employees as a where officeCode in (select officeCode from offices where country = 'USA')

22 -- 10 Obten el número de cliente, número de cheque y cantidad del cliente que ha realizado el pago más alto.

23 select amount, customerNumber, checkNumber from payments order by amount desc limit 1;

24 -- 11 Obten el número de cliente, número de cheque y cantidad de aquellos clientes cuyo pago es más alto que el promedio.

25 select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount desc;

26 -- 12 Obten el nombre de aquellos clientes que no han hecho ninguna orden.

27 select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNumber) as Total) as z where Total = '0'.

31 from customers as a left join (select customerName, count(*) as Total from orders group by customerName) as b on a.customerNumber = b.customerNumber;

130% 14:30

Result Grid Filter Rows: Q Search Export: F

customerName	Total
Hanover & Zephryus Co.	0
American Souvenirs Inc.	0
Porto Imports Co.	0
Asian Shopping Network Co.	0
Natürlich Autó	0
AMG Recetas	0
Misumi Shopping Network	0
Franken Gifts Co.	0
B&E Collectables	0
Schnelle Reise	0
Der Handel Imports	0
Cramer Spezialitäten, Ltd.	0
Alpin Trading Inc.	0

Result 13

Action Output

Time	Action	Response	Duration / Fetch Time
43 14:02:22	select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount desc;	134 row(s) returned	0.129 sec / 0.00006...
44 14:02:56	select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNumber) as Total) as z where Total = '0'.	24 row(s) returned	0.114 sec / 0.00002...

Query Completed

Result Grid Form Editor Field Types Read Only

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

```
select orderNumber, max(quantityOrdered), min(quantityOrdered), avg(quantityOrdered)
from orderdetails group by orderNumber;
```

The screenshot shows the MySQL Workbench interface with the SQL tab selected. The query window contains the following SQL code:

```
23 — 9 select empleados con oficina en USA oficinas 1, 2 y 3
24 • select lastName, firstName, officeCode from employees as a where officeCode in (select officeCode from offices where country
25 — 10 Obten el número de cliente, número de cheque y cantidad del cliente que ha realizado el pago más alto.
26 • select amount, customerNumber, checkNumber from payments order by amount desc limit 1;
27 — 11 Obten el número de cliente, número de cheque y cantidad de aquellos clientes cuyo pago es más alto que el promedio.
28 • select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount desc
29 — 12 Obten el nombre de aquellos clientes que no han hecho ninguna orden.
30 • select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNu
31 from customers a) as z where Total = '0';
32 — 13 Obten el máximo, mínimo y promedio del número de productos en las órdenes de venta
33 • select orderNumber, max(quantityOrdered), min(quantityOrdered), avg(quantityOrdered) from orderdetails group by orderNumber;
34 — 14 Dentro de la tabla orders, obtén el número de órdenes que hay por cada estado.
35 • select status, count(*) from orders group by status;
36 — 14a órdenes por cada estado(ubicación)
37 • select (select state from customers a where a.customerNumber=b.customerNumber).state count(orderNumber).total
```

The result grid shows the following data:

orderNumber	max(quantityOrder...)	min(quantityOrder...)	avg(quantityOrder...)
10102	41	39	40.0000
10103	46	22	33.8125
10104	49	23	34.0769
10105	50	22	36.3333
10106	50	26	37.5000
10107	39	20	28.6250
10108	45	26	35.0625
10109	47	26	35.3333
10110	48	20	35.6250
10111	48	26	36.1667
10112	29	23	26.0000
10113	50	21	35.7500

The action output shows two queries:

Time	Action	Response	Duration / Fetch Time
14:02:56	select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNu... 24 row(s) returned	0.114 sec / 0.000023...	
14:03:33	select orderNumber, max(quantityOrdered), min(quantityOrdered), avg(quantityOrdered) from orderdetails group by orderNumber 326 row(s) returned	0.128 sec / 0.00014 s...	

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

```
select status, count(*) from orders group by status;
```

The screenshot shows the MySQL Workbench interface with the SQL tab selected. The query window contains the following SQL code:

```
27 — 11 Obten el número de cliente, número de cheque y cantidad de aquellos clientes cuyo pago es más alto que el promedio.
28 • select customerNumber, checkNumber, amount from payments where amount > (select avg(amount) from payments) order by amount
29 — 12 Obten el nombre de aquellos clientes que no han hecho ninguna orden.
30 • select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNu
31 from customers a) as z where Total = '0';
32 — 13 Obten el máximo, mínimo y promedio del número de productos en las órdenes de venta
33 • select orderNumber, max(quantityOrdered), min(quantityOrdered), avg(quantityOrdered) from orderdetails group by orderNumber;
34 — 14 Dentro de la tabla orders, obtén el número de órdenes que hay por cada estado.
35 • select status, count(*) from orders group by status;
36 — 14a órdenes por cada estado(ubicación)
37 • select (select state from customers a where a.customerNumber=b.customerNumber).state count(orderNumber).total
```

The result grid shows the following data:

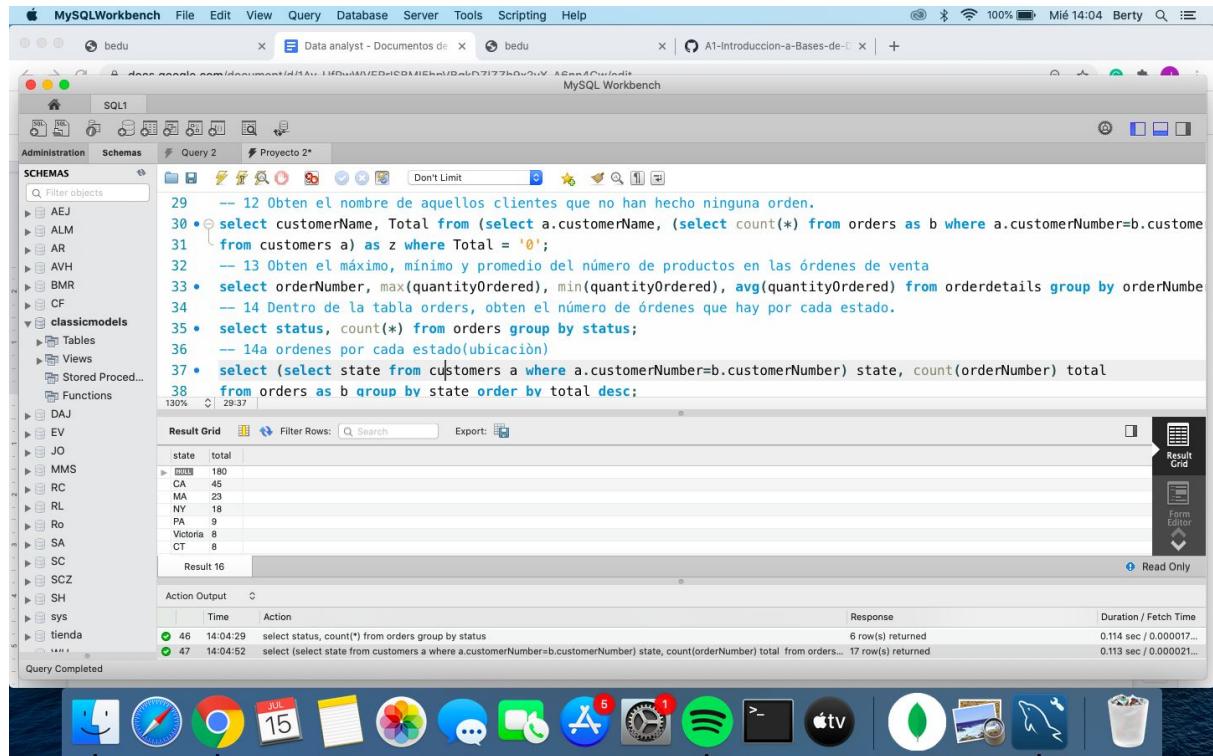
status	count(*)
Shipped	303
Resolved	4
Canceled	6
On Hold	4
Disputed	3
In Process	6

The action output shows two queries:

Time	Action	Response	Duration / Fetch Time
14:03:33	select orderNumber, max(quantityOrdered), min(quantityOrdered), avg(quantityOrdered) from orderdetails group by orderNumber 326 row(s) returned	0.128 sec / 0.00014 s...	
14:04:29	select status, count(*) from orders group by status 6 row(s) returned	0.114 sec / 0.000017...	

-- 14a ordenes por cada estado(ubicación)

```
select (select state from customers a where a.customerNumber=b.customerNumber) state,
count(orderNumber) total
from orders as b group by state order by total desc;
```



The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar reads "MySQL Workbench". The left sidebar shows the "Schemas" tree with several databases like AEJ, ALM, AR, AVH, BMR, CF, classicmodels, DAJ, EV, JO, MMS, RC, RL, Ro, SA, SC, SCZ, SH, sys, and tienda. The main pane contains a SQL editor with the following code:

```
29 -- 12 Obten el nombre de aquellos clientes que no han hecho ninguna orden.
30 • select customerName, Total from (select a.customerName, (select count(*) from orders as b where a.customerNumber=b.customerNumber) as Total from customers a) as z where Total = '0';
31 -- 13 Obten el máximo, mínimo y promedio del número de productos en las órdenes de venta
32 • select orderNumber, max(quantityOrdered), min(quantityOrdered), avg(quantityOrdered) from orderdetails group by orderNumber
33 -- 14 Dentro de la tabla orders, obten el número de órdenes que hay por cada estado.
34 • select status, count(*) from orders group by status;
35 • -- 14a ordenes por cada estado(ubicación)
36 • select (select state from customers a where a.customerNumber=b.customerNumber) state, count(orderNumber) total
37 • from orders as b group by state order by total desc;
```

The "Result Grid" tab is selected, showing the results of the last query:

state	total
NULL	180
CA	45
MA	23
NY	18
PA	9
Victoria	8
CT	8

Below the result grid, the "Action Output" table shows the execution details:

Action	Time	Response	Duration / Fetch Time
select status, count(*) from orders group by status	14:04:29	6 row(s) returned	0.114 sec / 0.000017...
select (select state from customers a where a.customerNumber=b.customerNumber) state, count(orderNumber) total from orders as b group by state order by total desc;	14:04:52	17 row(s) returned	0.113 sec / 0.000021...

Sesión 03 Agrupaciones y consultas

Proyecto 3

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

```
select a.productCode, a.productName, a.productDescription, b.productline, b.textDescription
from productlines b
right join products a
on a.productLine=b.productLine
group by a.productCode;
```

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

```
use classicmodels;
-- 1.- Obten el código de producto, nombre de producto y descripción de todos los productos.
select a.productCode, a.productName, a.productDescription, b.productline, b.textDescription
from productlines b
right join products a
on a.productLine=b.productLine
group by a.productCode;
-- 2.- Obten el número de orden, estado y costo total de cada orden.
select b.orderNumber, status, round(sum(priceEach), 1) from orders b
right join orderdetails a
on b.orderNumber = a.orderNumber
group by b.orderNumber;
```

The results grid displays the following data:

productCode	productName	productDescription	productline	textDescription
S10_2016	1996 Moto Guzzi 1100i	Official Moto Guzzi logos and insignias, saddle...	Motorcycles	Our motorcycles are state of the art replicas of classic as well as co...
S10_4698	2003 Harley-Davidson Eagle Drag Bike	Model features, official Harley Davidson logos a...	Motorcycles	Our motorcycles are state of the art replicas of classic as well as co...
S10_4757	1972 Alfa Romeo GTA	Features include: Turnable front wheels; steerin...	Classic Cars	Attention car enthusiasts: Make your wildest car ownership dreams...
S10_4902	1962 Lancia Delta 16V	Features include: Turnable front wheels; steerin...	Classic Cars	Attention car enthusiasts: Make your wildest car ownership dreams...
S12_1099	1968 Ford Mustang	Hood, doors and trunk all open to reveal highly...	Classic Cars	Attention car enthusiasts: Make your wildest car ownership dreams...
S12_1108	2001 Ferrari Enzo	Turnable front wheels; steering function; details...	Classic Cars	Attention car enthusiasts: Make your wildest car ownership dreams...
S12_1666	1958 Setra Bus	Model features 30 windows, skylights & glare re...	Trucks and Buses	The Truck and Bus models are realistic replicas of buses and spec...

The Action Output section shows two log entries:

Action	Time	Response	Duration / Fetch Time
use classicmodels	48 14:07:14	0 row(s) affected	0.109 sec
select a.productCode, a.productName, a.productDescription, b.productline, b.textDescription from productlines b right join produ...	49 14:07:20	110 row(s) returned	0.246 sec / 0.129 sec

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

```
select b.orderNumber, status, round(sum(priceEach), 1) from orders b  
right join orderdetails a  
on b.orderNumber = a.orderNumber  
group by orderNumber;  
describe products;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running at 98% battery life, with the date Mié 14:42 and user Berty. The main window has three tabs: SQL1, Query 2, and Proyecto 3. The SQL1 tab contains the following SQL code:

```
1 • use classicmodels;  
2 -- 1.- Obtener el código de producto, nombre de producto y descripción de todos los productos.  
3 • select a.productCode, a.productName, a.productDescription, b.productline, b.textDescription  
4 from productlines b  
5 right join products a  
6 on a.productline=b.productLine  
7 group by a.productCode;  
8 -- 2.- Obtener el número de orden, estado y costo total de cada orden.  
9 • select b.orderNumber, status, round(sum(priceEach), 1) from orders b  
10 right join orderdetails a  
11 on b.orderNumber = a.orderNumber  
12 group by orderNumber;
```

The Result Grid shows the output of the second query:

orderNumber	status	round(sum(priceEach), 1)
10100	Shipped	301.8
10101	Shipped	352.0
10102	Shipped	138.7
10103	Shipped	1520.4
10104	Shipped	1251.9
10105	Shipped	1479.7
10106	Shipped	1427.3

The Action Output pane shows two recent actions:

Time	Action	Response	Duration / Fetch Time
49 14:07:20	select a.productCode, a.productName, a.productDescription, b.productline, b.textDescription from productlines b right join products a	110 row(s) returned	0.246 sec / 0.129 sec
50 14:09:03	select b.orderNumber, status, round(sum(priceEach), 1) from orders b right join orderdetails a on b.orderNumber = a.orderNumber	326 row(s) returned	0.118 sec / 0.00014 s...

The bottom of the screen shows the Mac OS X dock with various application icons.

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

```
select a.orderNumber, a.orderDate, b.orderLineNumber, c.productName, b.quantityOrdered,
b.priceEach from orders as a
right join orderdetails as b
on a.orderNumber=b.orderNumber
right join products as c
on c.productCode=b.productCode
order by a.orderNumber;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running at 98% battery life, with the date Mié 14:43 and user Berty. The main window displays a SQL editor with the following code:

```
10      right join orderdetails as b
11      on b.orderNumber = a.orderNumber
12      group by orderNumber;
13 •    describe products;
14      -- 3.- Obten el número de orden, fecha de orden, línea de orden, nombre del producto, cantidad ordenada y precio de cada pieza que muestre los detalles
15 •    select a.orderNumber, a.orderDate, b.orderLineNumber, c.productName, b.quantityOrdered, b.priceEach from orders as a
16      right join orderdetails as b
17      on a.orderNumber=b.orderNumber
18      right join products as c
19      on c.productCode=b.productCode
20      order by a.orderNumber;
21      -- 4.- Obtén el número de orden, nombre del producto, el precio sugerido de fábrica (msrp) y precio de cada pieza.
```

The result grid shows the following data:

orderNumber	orderDate	orderLineNumber	productName	quantityOrdered	priceEach
10100	2003-01-06	2	1911 Ford Town Car	50	55.09
10100	2003-01-06	4	1932 Alfa Romeo 8C2300 Spider Sport	22	75.46
10100	2003-01-06	1	1936 Mercedes Benz 500K Roadster	49	35.29
10101	2003-01-09	4	1932 Model A Ford J-Coupe	25	108.06
10101	2003-01-09	1	1928 Mercedes-Benz SSK	26	167.06
10101	2003-01-09	3	1939 Chevrolet Deluxe Coupe	45	32.53
10101	2003-01-09	2	1938 Cadillac V-16 Presidential Limousine	46	44.35

The status bar at the bottom indicates "Query Completed". The system tray shows various application icons, including Finder, Mail, Safari, and others.

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

```
select a.orderNumber, b.productName, b.MSRP, b.buyPrice, a.priceEach precioVenta  
from products b  
right join orderdetails a  
on a.productCode=b.productCode  
order by a.orderNumber;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running at 98% battery life, with the date Mié 14:43 and user Berty. The main window displays a query editor with the following code:

```
18 right join products as c  
19 on c.productCode=b.productCode  
20 order by a.orderNumber;  
21 -- 4.- Obtén el número de orden, nombre del producto, el precio sugerido de fábrica (msrp) y precio de cada pieza.  
22 • select a.orderNumber, b.productName, b.MSRP, b.buyPrice, a.priceEach precioVenta  
23 from products b  
24 right join orderdetails a  
25 on a.productCode=b.productCode  
26 order by a.orderNumber;  
27 -- 5.- Obtén el número de cliente, nombre de cliente, número de orden y estado de cada cliente.  
28 • select a.customerNumber, a.customerName, b.orderNumber, b.status  
29 from customers a
```

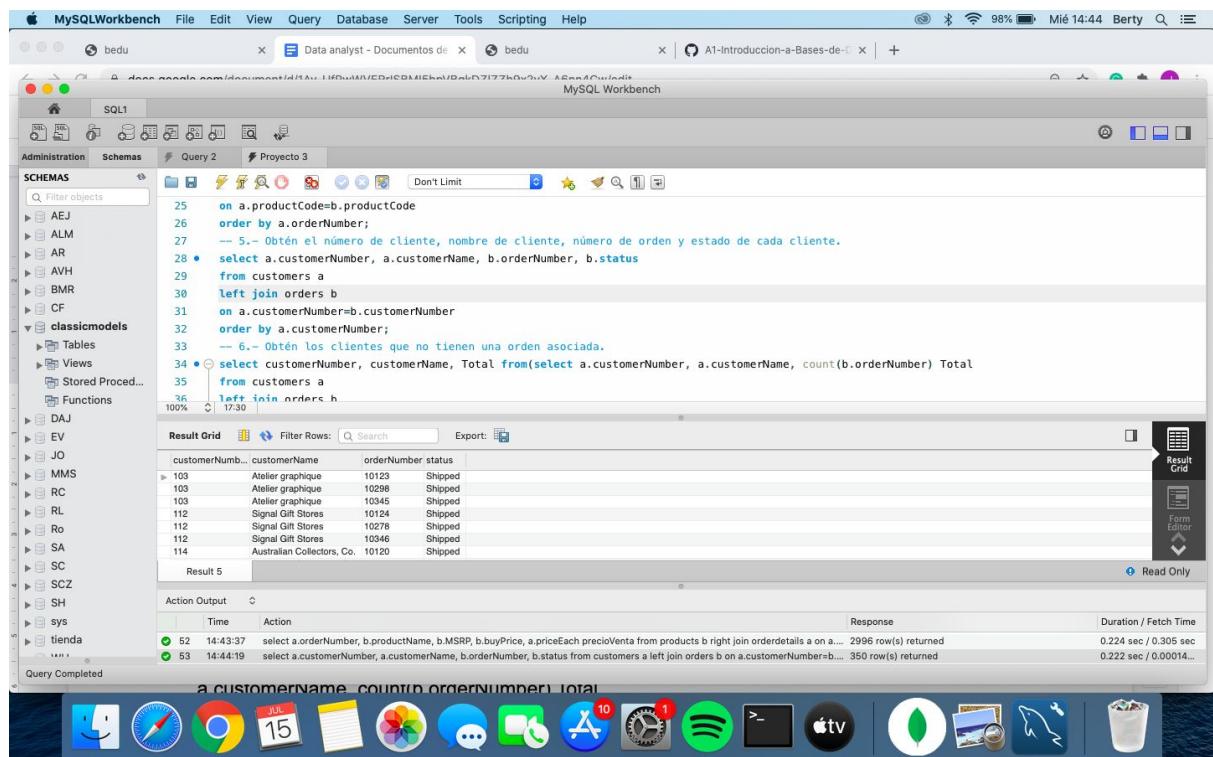
The result grid shows the output of the query:

orderNumber	productName	MSRP	buyPrice	precioVenta
10100	1932 Alfa Romeo 8C2300 Spider Sport	92.03	43.26	75.46
10100	1936 Mercedes Benz 500K Roadster	41.03	21.75	35.29
10101	1932 Model A Ford J-Coupe	127.13	58.48	108.06
10101	1928 Mercedes-Benz SSK	168.75	72.56	167.06
10101	1939 Chevrolet Deluxe Coupe	33.19	22.57	32.53
10101	1938 Cadillac V-16 Presidential Limousine	44.80	20.61	44.35
10102	1937 Lincoln Berline	102.74	60.62	95.55

The status bar at the bottom indicates "Query Completed". The Mac OS X dock is visible at the bottom, showing various application icons like Finder, Safari, Mail, Calendar, and Spotify.

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

```
select a.customerNumber, a.customerName, b.orderNumber, b.status
from customers a
left join orders b
on a.customerNumber=b.customerNumber
order by a.customerNumber;
```



```
MySQLWorkbench  File  Edit  View  Query  Database  Server  Tools  Scripting  Help
bedu  Data analyst - Documentos de  bedu  A1-Introduccion-a-Bases-de-Datos-Armando
MySQL Workbench

SQL1
Administration Schemas Query 2 Proyecto 3
Schemas
Filter objects
AEG ALM AR AVH BMR CF classicmodels
Tables Views Stored Proced...
Functions
DAJ EV JO MMS RC RL Ro SA SC SCZ SH sys tienda
Result Grid Filter Rows: Search Export:
customerNumber customerName orderNumber status
103 Atelier graphique 10123 Shipped
103 Atelier graphique 10298 Shipped
103 Atelier graphique 10345 Shipped
112 Signal Gift Stores 10124 Shipped
112 Signal Gift Stores 10278 Shipped
112 Signal Gift Stores 10346 Shipped
114 Australian Collectors, Co. 10120 Shipped
Action Output
Time Action Response Duration / Fetch Time
52 14:43:37 select a.orderNumber, b.productName, b.MSRP, b.buyPrice, a.priceEach precioVenta from products b right join orderdetails a on a... 2996 row(s) returned 0.224 sec / 0.305 sec
53 14:44:19 select a.customerNumber, a.customerName, b.orderNumber, b.status from customers a left join orders b on a.customerNumber=b.... 350 row(s) returned 0.222 sec / 0.00014...
```

Result 5

Action Output

Response

Duration / Fetch Time

52 14:43:37 select a.orderNumber, b.productName, b.MSRP, b.buyPrice, a.priceEach precioVenta from products b right join orderdetails a on a... 2996 row(s) returned 0.224 sec / 0.305 sec

53 14:44:19 select a.customerNumber, a.customerName, b.orderNumber, b.status from customers a left join orders b on a.customerNumber=b.... 350 row(s) returned 0.222 sec / 0.00014...

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

```
select customerNumber, customerName, Total from(select a.customerNumber,
a.customerName, count(b.orderNumber) Total
from customers a
left join orders b
on a.customerNumber=b.customerNumber
group by a.customerNumber) as e
where Total = '0' order by customerNumber desc;
select * from Sin_ordenes where Total = 0 order by customerNumber;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running at 98% battery life, with the date Mié 14:44 and user Berty. The main window displays two tabs: 'Query 2' and 'Projecto 3'. The 'Query 2' tab contains the SQL code provided above, with line numbers 25 through 36 visible. The 'Result Grid' tab shows the output of the query, listing customer details and order counts. The 'Action Output' tab shows the execution logs for the two queries. The system tray at the bottom shows various application icons, including Finder, Safari, and Mail, with some having notification counts.

customerNumber	customerName	orderNumber	status
103	Archer graphicx	10297	Shipped
103	Atelier graphique	10298	Shipped
103	Atelier graphique	10345	Shipped
112	Signal Gift Stores	10124	Shipped
112	Signal Gift Stores	10278	Shipped
112	Signal Gift Stores	10346	Shipped
114	Australian Collectors, Co.	10120	Shipped

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

```
select b.customerName, concat(a.lastName, " ",a.firstName) Empleado, c.checkNumber,
c.amount
from employees a
left join customers b
on a.employeeNumber=b.salesRepEmployeeNumber
left join payments c
on b.customerNumber=c.customerNumber
where customerName != 'Null';
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, which includes the 'classicmodels' schema. The main area contains the SQL editor with the following code:

```
38   group by a.customerNumber) as e
39   where Total = '0' order by customerNumber desc;
40 •  select * from Sin_ordenes where Total = 0 order by customerNumber;
41 -- 7.- Obtén el apellido de empleado, nombre de empleado, nombre de cliente, número de cheque y total de cada cheque, es decir, los clientes asociados
42 •  select b.customerName, concat(a.lastName, " ",a.firstName) Empleado, c.checkNumber, c.amount
43   from employees a
44   left join customers b
45   on a.employeeNumber=b.salesRepEmployeeNumber
46   left join payments c
47   on b.customerNumber=c.customerNumber
48   where customerName != 'Null';
49 -- 8. Fíjate en los resultados de la ejecución de la consulta.
```

The 'Result Grid' tab is selected, showing the following data:

customerName	Empleado	checkNumbe...	amount
Atelier graphique	Hernandez Gerard	HQ356398	6056.78
Atelier graphique	Hernandez Gerard	JM552025	14571.44
Atelier graphique	Hernandez Gerard	OM314933	1676.14
Signal Gift Stores	Thompson Leslie	BO864823	14191.12
Signal Gift Stores	Thompson Leslie	HO55022	32641.98
Signal Gift Stores	Thompson Leslie	ND748579	33347.88
Australian Collectors, Co.	Fixter Andy	GG31455	45864.03

The 'Action Output' tab shows the execution log:

Action	Time	Response	Duration / Fetch Time
select customerNumber, customerName, Total from(select a.customerNumber, a.customerName, count(b.orderNumber) Total fro...	14:44:59	24 row(s) returned	0.119 sec / 0.000024...
select b.customerName, concat(a.lastName, " ",a.firstName) Empleado, c.checkNumber, c.amount from employees a left join custo...	14:45:24	275 row(s) returned	0.232 sec / 0.00015...

-- 8 Ejercicios del 5 al 7 con right join

-- 5a

```
select a.customerNumber, a.customerName, b.orderNumber, b.status
from orders b
right join customers a
on a.customerNumber=b.customerNumber
order by a.customerNumber;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running at 97% battery life, with the date Mié 14:46 and user Berty.

The main window displays a SQL editor with the following code:

```
-- 8 Ejercicios del 5 al 7 con right join
-- 5a
select a.customerNumber, a.customerName, b.orderNumber, b.status
from orders b
right join customers a
on a.customerNumber=b.customerNumber
order by a.customerNumber;
```

The code is numbered from 49 to 69. Lines 57 and 58 are highlighted with a yellow circle icon. The result grid shows the following data:

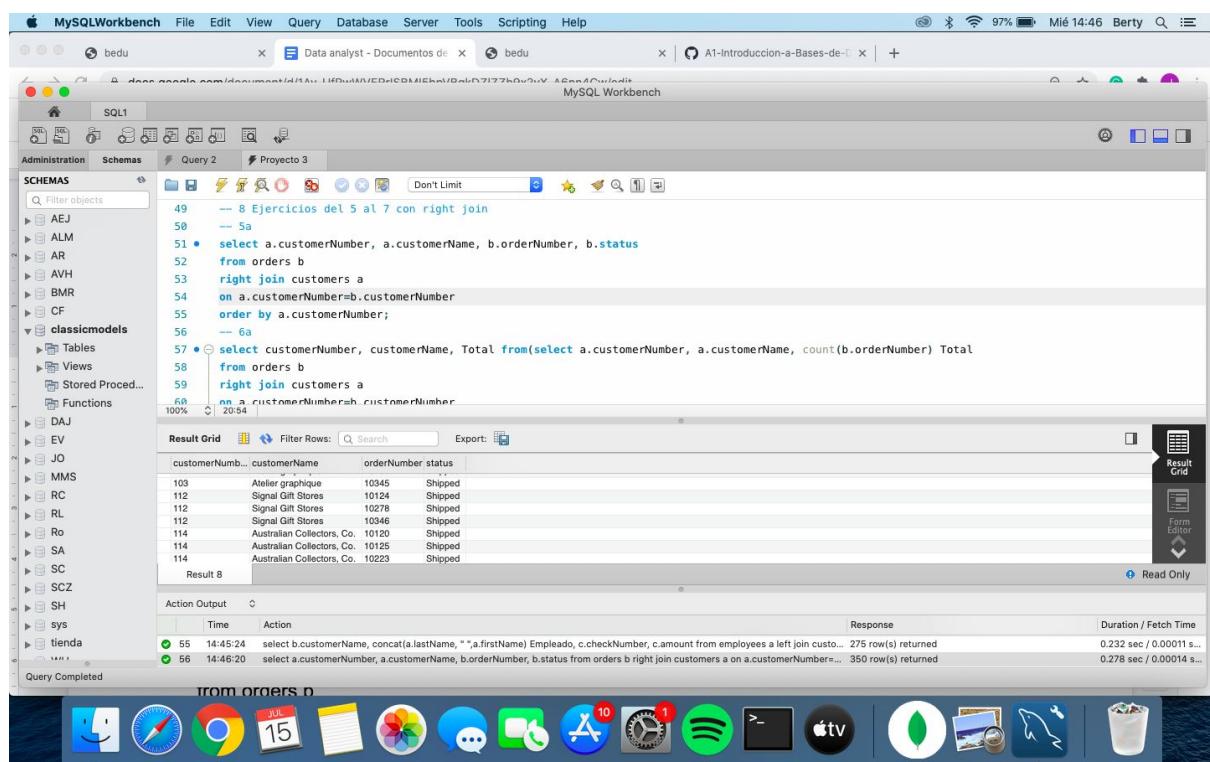
customerNumber	customerName	orderNumber	status
103	Atelier graphique	10345	Shipped
112	Signal Gift Stores	10124	Shipped
112	Signal Gift Stores	10278	Shipped
112	Signal Gift Stores	10346	Shipped
114	Australian Collectors, Co.	10120	Shipped
114	Australian Collectors, Co.	10125	Shipped
114	Australian Collectors, Co.	10223	Shipped

The status column contains the values 'Shipped' for all rows. The bottom of the window shows the action log:

Action Output	Time	Action	Response	Duration / Fetch Time
	55 14:45:24	select b.customerName, concat(a.lastName, " ",a.firstName) Empleado, c.checkNumber, c.amount from employees a left join custo...	275 row(s) returned	0.232 sec / 0.0001 s...
	56 14:46:20	select a.customerNumber, a.customerName, b.orderNumber, b.status from orders b right join customers a on a.customerNumber=...	350 row(s) returned	0.278 sec / 0.00014 s...

-- 6a

```
select customerNumber, customerName, Total from(select a.customerNumber,
a.customerName, count(b.orderNumber) Total
from orders b
right join customers a
on a.customerNumber=b.customerNumber
group by a.customerNumber) as e
where Total = '0' order by customerNumber desc;
```



The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Contains the SQL code for query 6a.
- Result Grid:** Displays the results of the query, showing columns: customerNumber, customerName, and orderNumber status. The data includes rows for Atelier graphique, Signal Gift Stores, and Australian Collectors, Co.
- Action Output:** Shows the log of actions taken, including the execution of the query and its duration.

customerNumber	customerName	orderNumber	status
103	Atelier graphique	10345	Shipped
112	Signal Gift Stores	10124	Shipped
112	Signal Gift Stores	10278	Shipped
112	Signal Gift Stores	10126	Shipped
114	Australian Collectors, Co.	10120	Shipped
114	Australian Collectors, Co.	10125	Shipped
114	Australian Collectors, Co.	10223	Shipped

-- 7a

```
select b.customerName, concat(a.lastName, " ",a.firstName) Empleado, c.checkNumber,
c.amount
from employees a
right join customers b
on a.employeeNumber=b.salesRepEmployeeNumber
right join payments c
on b.customerNumber=c.customerNumber;
```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** A sidebar listing various databases and their objects.
- Query Editor:** Contains the SQL code for query 7a.
- Result Grid:** Displays the output of the query, showing columns for customerName, Empleado, and checkNum... amount.
- Action Output:** Shows the execution log with two entries: a select statement at 14:47:10 and the current query at 14:47:40.
- Bottom Dock:** Features a Mac OS X-style dock with icons for Finder, Safari, Mail, and other applications.

customerName	Empleado	checkNum... amount
Atelier graphique	Hernandez Gerard	0014833 1676.14
Atelier graphique	Montgomery Leslie	0294409 1411.12
Signal Gift Stores	Thompson Leslie	HD16022 32641.98
Signal Gift Stores	Thompson Leslie	ND748579 33247.88
Australian Collectors, Co.	Fixter Andy	GG31455 45864.03
Australian Collectors, Co.	Fixter Andy	MA765515 82261.22
Australian Collectors, Co.	Fixter Andy	NP603840 7565.08

-- 9 crear vistas para 3 consultas más complejas

1:

```
create view 0_ordenes as (select customerNumber, customerName, Total from(select  
a.customerNumber, a.customerName, count(b.orderNumber) Total  
from customers a  
left join orders b  
on a.customerNumber=b.customerNumber  
group by a.customerNumber) as e  
where Total = '0' order by customerNumber desc);
```

```
select * from 0_ordenes;
```

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, Help, and a system status bar showing battery level (96%), signal strength, and the date (Mié 14:49). The main window has tabs for 'bedu' and 'Data analyst - Documentos de...'. The left sidebar shows 'Administration' and 'Schemas' sections, with 'classicmodels' selected. The central area contains a query editor with the following SQL code:

```
on b.customerNumber=rc.customerNumber;
-- 9 crear vistas para 3 consultas más complejas
create view _ordenes as (select customerNumber, customerName, Total from(select a.customerNumber, a.customerName, count(b.orderNumber) Total
from customers a
left join orders b
on a.customerNumber=b.customerNumber
group by a.customerNumber) as e
where Total = '0' order by customerNumber desc);
select * from _ordenes;
```

The 'Result Grid' tab is active, displaying the results of the last query:

customerNum...	customerName	Total
477	Mit Vergnügen & Co.	0
465	Anton Designs, Ltd.	0
459	Werbung Exchange	0
443	Feuer Online Stores, Inc	0
409	Stuttgart Collectable Exchange	0
376	Precious Collectables	0
369	Lisboa Souvenirs, Inc	0

Below the result grid, the 'Action Output' section shows two log entries:

Action	Time	Response	Duration / Fetch Time
select * from _ordenes	14:47:40	273 row(s) returned	0.234 sec / 0.00012...
select * from _ordenes	14:48:49	24 row(s) returned	0.111 sec / 0.000072...

The bottom status bar indicates 'Query Completed'.

2.-

```
create view Empleado_Cliente as (select b.customerName, concat(a.lastName, "
",a.firstName) Empleado, c.checkNumber, c.amount
from employees a
left join customers b
on a.employeeNumber=b.salesRepEmployeeNumber
left join payments c
on b.customerNumber=c.customerNumber
where customerName != 'Null');
```

```
select * from Empleado_Cliente;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running at 96% battery life, with the date Mié 14:49 and user Berty.

The main window displays two queries in the SQL editor:

```
78 • select * from 0_orden...
```

```
79
80 • create view Empleado_Cliente as (select b.customerName, concat(a.lastName, "
",a.firstName) Empleado, c.checkNumber, c.amount
81   from employees a
82   left join customers b
83     on a.employeeNumber=b.salesRepEmployeeNumber
84   left join payments c
85     on b.customerNumber=c.customerNumber
86   where customerName != 'Null');
87
88 • select * from Empleado_Cliente;
89
```

The Result Grid shows the output of the last query:

customerName	Empleado	checkNum...	amount
Atelier graphique	Hernandez Gerard	0314939	1676.14
Signal Gift Stores	Thompson Leslie	0205114	1676.14
Signal Gift Stores	Thompson Leslie	HQ50022	33641.98
Signal Gift Stores	Thompson Leslie	ND748579	33347.88
Australian Collectors, Co.	Fixter Andy	GG31455	45664.03
Australian Collectors, Co.	Fixter Andy	MA765515	82261.22
Australian Collectors, Co.	Fixter Andy	NP603840	7565.08

The status bar at the bottom shows the query completed and provides performance metrics:

```
Query Completed
create view Empleado_Cliente as (select b.customerName, concat(a.lastName, "
",a.firstName) Empleado, c.checkNumber, c.amount
from employees a
left join customers b
on a.employeeNumber=b.salesRepEmployeeNumber
left join payments c
on b.customerNumber=c.customerNumber
where customerName != 'Null');

Time Action Response Duration / Fetch Time
59 14:48:49 select * from 0_orden... 24 row(s) returned 0.111 sec / 0.000072...
60 14:49:52 select * from Empleado_Cliente 275 row(s) returned 0.242 sec / 0.00011 s...
```

3.-

```
create view Detalle_ordenes as (select a.orderNumber, a.orderDate, b.orderLineNumber,
c.productName, b.quantityOrdered, b.priceEach from orders as a
right join orderdetails as b
on a.orderNumber=b.orderNumber
right join products as c
on c.productCode=b.productCode
order by a.orderNumber);
```

```
select * from Detalle_ordenes;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The title bar indicates the application is running and the current date and time. The main window has two tabs: 'SQL1' and 'Query 2'. The 'SQL1' tab contains the SQL code for creating the 'Detalle_ordenes' view and executing a select statement against it. The 'Result Grid' pane displays the results of the query, showing 13 rows of data from the 'Detalle_ordenes' view. The bottom status bar shows the completion of the query.

orderNumber	orderDate	orderLineNumber	productName	quantityOrdered	priceEach
10100	2003-01-06	3	1985 Toyota Supra	30	136.00
10100	2003-01-06	2	1911 Grand Touring Sedan	50	55.09
10100	2003-01-06	4	1932 Alfa Romeo 8C2300 Spider Sport	22	75.46
10100	2003-01-06	1	1936 Mercedes Benz 500K Roadster	49	35.29
10101	2003-01-09	4	1932 Model A Ford J-Coupe	25	108.06
10101	2003-01-09	1	1928 Mercedes-Benz SSK	26	167.06
Detalle_ordenes 13					
Action Output	Time	Action	Response	Duration / Fetch Time	
	60	14:49:52 select * from Empleado_Cliente	275 row(s) returned	0.242 sec / 0.00011 sec	
	61	14:51:00 select * from Detalle_ordenes	2997 row(s) returned	0.233 sec / 0.354 sec	

Sesión 04: Fundamentos de MongoDB

Proyecto 4

1.- Obtén los datos de contacto de cada compañía.

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

The screenshot shows the MongoDB Compass interface. On the left, the sidebar displays the database structure: Local, HOST ec2-35-166-232-75.us-west-2.compute.amazonaws.com:27017, and the sample_training database. The sample_training.companies collection is selected. The main area shows a grid of document results with columns for _id, email_address, and phone_number. Below the grid, the query parameters are visible: PROJECT {email_address: 1, phone_number: 1}, SORT {}, and COLLATION {}. To the right, there are three panels for past queries: 'Proyecto 4-4' (filter { founded_year: 2008 }), 'Proyecto 4-3' (filter { founded_month: 10 }), and 'Proyecto 4-1' (PROJECT { email_address: 1, phone_number: 1 }).

2.- Obtén la fuente de cada tweet.

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

The screenshot shows the MongoDB Compass interface. On the left, the sidebar displays the database structure with 'sample_training.tweets' selected. The main pane shows the results of a query with the following projection:

```
{  
  source: 1  
}
```

The results list several documents, each with an '_id' field and a 'source' field. The 'source' field contains URLs from various sources like TweetDeck, BlackBerry Twitter, Echofon, and Power Twitter. The interface includes a 'FILTER' bar at the top and a 'PROJECT' panel on the right where the projection code is displayed.

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

```
{  
  filter: {  
    founded_month: 10  
  }  
}
```

The screenshot shows the MongoDB Compass interface. The main window displays the 'sample_training.companies' collection with a total of 9.5k documents. A search bar at the top contains the query '{founded_month: 10}'. The results table shows 20 documents from a total of 301. One document is expanded to show fields like _id, name, and various URLs. On the right, three previous queries are listed: 'Proyecto 4-4' (founded_year: 2008), 'Proyecto 4-3' (founded_month: 10), and 'Proyecto 4-1' (PROJECT). The Compass interface includes a sidebar with databases and collections, and a bottom dock with various Mac OS X application icons.

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

```
{  
  filter: {  
    founded_year: 2008  
  }  
}
```

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases and collections, with 'sample_training' expanded and 'companies' selected. The main pane displays the 'sample_training.companies' collection, showing a single document with various fields like '_id', 'name', 'permalink', and 'tag_list'. A sidebar on the right contains three project sections: 'Proyecto 4-4' (filtering by founded_year: 2008), 'Proyecto 4-3' (filtering by founded_month: 10), and 'Proyecto 4-1' (projecting email_address and phone_number). The bottom of the screen shows the Mac OS X dock with various application icons.

5.- Obtén todos los *post* del autor machine.

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

The screenshot shows the MongoDB Compass interface on a Mac OS X desktop. The left sidebar lists databases and collections, with 'sample_training.posts' selected. The main pane displays the results of a query for documents where the 'author' field is 'machine'. There are three results listed:

```
_id: ObjectId("50ab0f8bbcf1bfe2536dc3fb")
body: "An amendment I  
body: <p>Congress shall make no law respecting an establishment ...</p>
permalink: "NRNlZkJKTyspAloRGe"
author: "machine"
title: "Bill of Rights"
tags: Array
comments: Array
date: 2012-11-20T05:05:15.231+00:00

_id: ObjectId("50ab0f8bbcf1bfe2536dc3fa")
body: "We the People of the United States, in Order to form a more perfect Union...
permalink: "NSg9boVWyKEoNydis"
author: "machine"
title: "US Constitution"
tags: Array
comments: Array
date: 2012-11-20T05:05:15.231+00:00

_id: ObjectId("50ab0f8bbcf1bfe2536dc3fb")
body: "Four score and seven years ago our fathers brought forth on this conti...
permalink: "ekOGGzbYQzUPThSxRFU"
author: "machine"
title: "Gettysburg Address"
tags: Array
comments: Array
date: 2012-11-20T05:05:15.234+00:00
```

6.- Obtén todos los tweets provenientes de la web.

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

The screenshot shows the MongoDB Compass interface on a Mac OS X desktop. The main window displays the 'sample_training.tweets' collection with 24.8k documents. A filter is applied: {source: 'web'}. The results show two tweets. The first tweet's text is: "eu preciso de terminar de fazer a minha tabela, está muito foda **". The second tweet's text is: "First week of school is over :P". Both tweets have a source of 'web'. The interface includes a sidebar with other collections like sample_analytics, sample_geospatial, sample_mflix, sample_supplies, sample_training, companies, grades, inspections, posts, routes, stories, trips, and zips. A 'PROJECT' panel on the right contains the query code: {source: 'web'}.

7.- Obtén todas las compañías fundadas en octubre del 2008

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

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases and collections, with 'sample_training' selected. The main pane displays the 'sample_training.companies' collection, which contains 9.5k documents. A query is being run with the following filter:

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

The results pane shows one document from the collection:

```
_id: ObjectId("52cdef7c4bab8bd6752985ca")  
name:"tunesBag"  
permalink:"tunesBag"  
crunchbase_url:"http://www.crunchbase.com/company/tunesbag"  
homepage_url:"http://www.tunesBag.com"  
blog_url:"http://tunesBag.blogspot.com/"  
blog_feed_url:"http://blog.tunesbag.com/feeds/posts/default?alt=rss"  
twitter:"  
category_code:"games_video"  
number_of_employees:  
founded_year:  
founded_month:  
founded_day:  
deadpooled_year:  
deadpooled_month:  
deadpooled_day:  
deadpooled_art:  
tag_list:"music, cloud, locker, mp3, music-streaming, streaming"  
alias_list:  
email_address:"office@tunesBag.com"  
phone_number:"+43 688 215 27 96"  
description:"Social Music Player"  
created_at:"Thu Mar 20 15:45:40 UTC 2008"  
updated_at:"Thu Jan 19 00:37:48 UTC 2012"  
overview:"<p>Austria based tunesBag is a music player with social features on th...</p>"
```

Below the results, there is a link to 'SHOW 17 MORE FIELDS'. The bottom of the screen shows the macOS dock with various application icons.

8.- Obtén todas las compañías con más de 50 empleados.

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

The screenshot shows the MongoDB Compass interface on a Mac OS X desktop. The left sidebar lists databases and collections, with 'sample_training' selected. The main pane displays the 'companies' collection, which contains 9.5k documents. A search bar at the top has the filter '{number_of_employees: {\$gt: 50}}'. Below it, a table shows two document snippets. The first snippet is for Facebook, and the second is for Twitter. Both documents have '_id' fields and 'name' fields set to 'Facebook' and 'Twitter' respectively. The table includes columns for '_id', 'name', 'permalink', 'category_code', 'number_of_employees', 'founded_year', 'founded_month', 'founded_day', 'deadpooled_year', 'deadpooled_month', 'deadpooled_day', 'deadpooled_url', 'tag_list', 'alias_list', 'email_address', 'phone_number', 'description', 'created_at', 'updated_at', and 'overview'. On the right side, there are three collapsed project panels labeled 'Proyecto 4-8', 'Proyecto 4-7', and 'Proyecto 4-4', each with its own filter criteria. The bottom of the screen shows the Mac OS X dock with various application icons.

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

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

The screenshot shows the MongoDB Compass interface with the 'sample_training_stories' collection selected. The results pane displays two documents matching the query. The first document has 153 comments and the second has 93 comments. The query in the filter bar is:

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

The results pane shows the following documents:

```
_id: ObjectId("4ba267dc238d3ba3ca000001")  
href: "http://digg.com/people/Jedi_believer_who_refused_to_remove_hood_gets_an_apology!"  
title: "Jedi believer who refused to remove hood gets an apology!"  
comments: 153  
> container: Object  
  submit_date: 1268771801  
> topic: Object  
  promote_date: 1268915964  
  id: "19970866"  
  media: "news"  
  diggs: 484  
  description: "Chris Jarvis is a member of the International Church of Jediism - base..."  
  link: "http://www.dailymail.co.uk/news/article-1250365/Jedi-believer-wins-apo..."  
> user: Object  
  status: "popular"  
> shorturl: Array
```

```
_id: ObjectId("4ba267dc238d3ba3ca000003")  
href: "http://digg.com/odd_stuff/Man_Assaulted_Female_Police_Officer_With_His_Penis..."  
title: "Man Assaulted Female Police Officer With His Penis..."  
comments: 93  
> thumbnail: Object  
  > container: Object  
    submit_date: 1268755030  
> topic: Object  
    promote_date: 1268903731  
    id: "19963397"  
    media: "news"  
    diggs: 262  
    description: "A 28-year-old man who assaulted a female police officer with his penis..."
```

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  
}
```

The screenshot shows the MongoDB Compass interface. On the left, the sidebar displays the database structure with the 'sample_training' database selected. Under 'sample_training', there are several collections: 'companies', 'grades', 'inspections', 'posts', 'routes', 'stories', and 'trips'. The 'companies' collection is currently selected. The main pane shows the results of a query on the 'sample_training.companies' collection. The query is defined in the 'FILTER' section of the query builder:

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

The 'SORT' section specifies sorting by 'number_of_employees' in ascending order (1). The 'LIMIT' section is set to 1. The results pane displays one document from the collection:

```
_id: ObjectId("52cdef7c4bab8bd675297e60")  
name: "Fevote"  
permalink: "Fevote"  
crunchbase_url: "http://www.crunchbase.com/company/fevote"  
homepage_url: "http://www.fevote.com"  
blog_url: "http://blog.fevote.com/"  
blog_feed_url: "http://blog.fevote.com/feed/"  
twitter_username: null  
category_code: "web"  
number_of_employees: 1  
founded_year: 2007  
founded_month: 9  
founded_day: 1  
deadpooled_year: null  
deadpooled_month: null  
deadpooled_day: null  
deadpooled_url: null  
tag_list: null  
alias_list: null  
email_address: ""  
phone_number: ""  
description: null  
created_at: "Fri Sep 21 20:21:41 UTC 2007"  
updated_at: "Sun Aug 10 13:12:45 UTC 2008"  
overview: "<p>Fevote provides suggestion boards for companies and various subjects...</p>"
```

At the bottom of the screen, the Mac OS X dock is visible with various application icons.

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

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

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

- Host:** ec2-35-166-232-75.us-west-2.compute.amazonaws.com
- Collection:** sample_training.companies
- Documents:** 9.5k
- Aggregations:** 1
- Schema:** Not visible in the screenshot.
- Explain Plan:** Not visible in the screenshot.
- Indexes:** 1
- Filter:** `{ $and: [{ number_of_employees: { $ne: null } }, { number_of_employees: { $ne: 0 } }] }`
- Sort:** `{ number_of_employees: -1 }`
- Limit:** 1

The results pane displays one document from the collection:

```
_id: ObjectId("52cdef7c4bab8bd67529856a")  
name: "IBM"  
permalink: "ibm"  
crunchbase_url: "http://www.crunchbase.com/company/ibm"  
homepage_url: "http://www.ibm.com"  
blog_url: ""  
blog_feed_url: ""  
twitter_username: "IBM"  
category_code: "software"  
number_of_employees: 380000  
founded_year: 1896  
founded_month: null  
founded_day: null  
deadpooled_year: null  
deadpooled_month: null  
deadpooled_day: null  
deadpooled_url: ""  
tag_list: ""  
alias_list: ""  
email_address: "ews@us.ibm.com"  
phone_number: "914-499-1900"  
description: ""  
created_at: "Fri Mar 14 22:55:52 UTC 2008"  
updated_at: "Sat Jan 04 02:56:24 UTC 2014"  
overview: "<p>IBM, acronym for International Business Machines, is a multinationa..."
```

12.- Obtén la historia más comentada.

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

The screenshot shows the MongoDB Compass interface on a Mac OS X desktop. The main window displays a collection named 'sample_training.stories' with 9.8k documents. A query is being run to find the document with the highest number of comments, sorted by 'comments' in descending order (-1) and limited to 1 result. The results pane shows one document:

```
_id:ObjectId("4ba27ea0238d3ba3ca002251")  
title:"Republican Brown wins Massachusetts Senate seat!"  
comments: 1864  
thumbnail:Object  
container:Object  
submit_date:1263954202  
topic:Object  
promote_date:1263956432  
id:"10636777"  
media:"news"  
digs:2098  
description:"Striving for an epic upset in liberal Massachusetts, Republican Scott ..."  
link:"http://news.yahoo.com/s/ap/us_massachusetts_senate;_ylt=AuEva3AHQ0x0R..."  
user:Object  
status:"popular"  
shorturl:Array
```

The 'Past Queries' panel on the right contains a saved query named 'Proyecto 4-12' with the same sort and limit conditions. The system tray at the bottom shows various application icons, including Mail, Safari, Calendar, Notes, and Spotify.

13.- Obtén la historia menos comentada.

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

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases and collections, with 'sample_training' selected. In the main pane, the 'sample_training.stories' collection is displayed. The 'Documents' tab is active, showing a single document. The document details are as follows:

```
_id: ObjectId("4ba277c0238d3ba3ca001819")
url: "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"
digg: Object
description: "The world's smallest species of eel-loach fish has been discovered by ..."
link: "http://www.physorg.com/news184855478.html"
user: Object
status: "popular"
shorturl: Array
```

The query results are shown in the 'Past Queries' panel on the right, with three projects defined:

- Proyecto 4-13**:
FILTER: { comments: { \$gt: 0 } }
SORT: { comments: 1 }
LIMIT: 1
- Proyecto 4-12**:
SORT: { comments: -1 }
LIMIT: 1
- Proyecto 4-9**:
FILTER: { \$and: [{ }] }

Sesión 05: Consultas en MongoDB

Proyecto 5

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.

```
[$match: {  
    number_of_reviews: {$gt: 50}  
}, {$match: {  
    "address.country": "Brazil"  
}, {$match: {  
    'review_scores.review_scores_rating': {$gte: 80}  
}, {$match: {  
    amenities: {$in: [/Ethernet/i]}  
}, {$group: {  
    _id: null,  
    total: {  
        $sum: 1  
    }  
}}]
```

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

- Left Sidebar:** Shows the Local connection, 10 DBs, and 25 Collections. The `sample_airbnb` database is selected, and the `listingsAndReviews` collection is currently being edited.
- Top Bar:** The title bar says `sample_airbnb.listingsA...` and the tab is set to `Aggregations`.
- Aggregation Pipeline:** The pipeline consists of three stages:
 - `$match`: Filters documents where `'review_scores.review_scores_rating'` is greater than or equal to 80.
 - `$match`: Filters documents where the `amenities` field contains the regular expression `/Ethernet/i`.
 - `$group`: Groups the documents by `_id: null` and calculates the total count of documents, stored in the `total` field.
- Output:** The output shows a sample of 6 documents from the `listingsAndReviews` collection. Each document includes fields like `listing_url`, `name`, `summary`, `space`, and `description`. The `total` field is shown as 6.
- Bottom Buttons:** Buttons for `ADD STAGE` and `SAVE` are visible at the bottom of the pipeline editor.

Sesión 06: Agregaciones

Proyecto 6

El proyecto consiste en obtener, por país, el número de películas que hay de cada género:

```
[{$unwind: {  
    path: "$genres",  
    includeArrayIndex: 'string'  
}}, {$unwind: {  
    path: "$countries",  
    includeArrayIndex: 'string'  
}}, {$group: {  
    _id: {"genres": "$genres", "countries": "$countries"},  
    Total: {$sum: 1}  
}}, {$sort: {  
    Total: -1  
}}, {$addFields: {  
    Pais: "$_id.countries",  
    Genero: "$_id.genres",  
}}, {$project: {  
    _id: 0,  
    Pais: 1,  
    Genero: 1,  
    Total: 1  
}}]
```

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

- Project:** sample_mflix.Proyecto 6 (view on: sample_mflix.movies)
- Aggregation Pipeline:**
 - Documents
 - Aggregations
 - Schema
 - Explain Plan
 - Indexes
 - Validation
- Options:** Read Only
- Find:** FIND, RESET, ...
- MAXITEMS:** 5000
- SKIP:** 0
- LIMIT:** 0
- Results:** Displays documents 1 - 20 of 1712.
- Document Examples:**
 - Total: 6066
Pais: "USA"
Genero: "Drama"
 - Total: 3843
Pais: "USA"
Genero: "Comedy"
 - Total: 2262
Pais: "France"
Genero: "Drama"
 - Total: 1920
Pais: "USA"
Genero: "Romance"
 - Total: 1777
Pais: "UK"
Genero: "Drama"

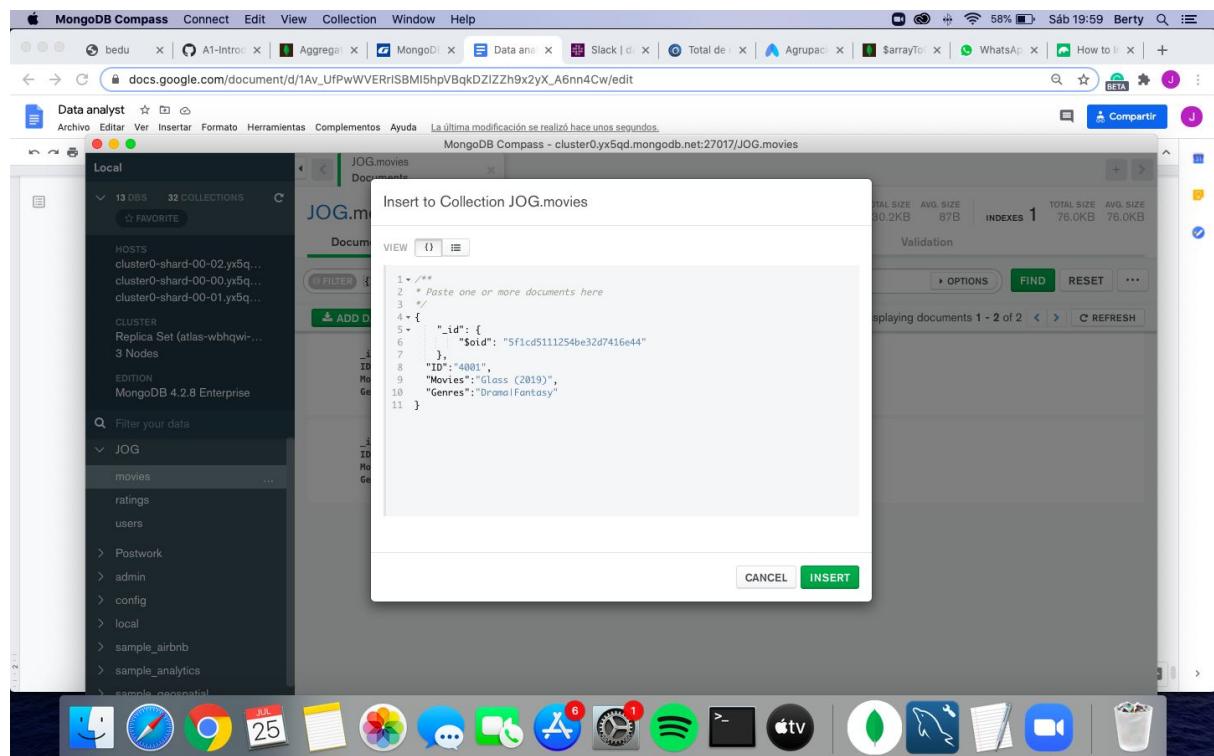
Sesión 07 Configuración de Bases de Datos Locales

Proyecto 7

1.- Agregar los siguientes registros en formato CSV a la Colección movies

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

```
{
  "_id": {
    "$oid": "5f1cd63e1254be32d7416e45"
  },
  "ID":"4000",
  "Movie":"Avengers: Endgame (2019)",
  "Genres":"Fantasy|Sci-Fi"
}
{
  "_id": {
    "$oid": "5f1cd63e1254be32d7416e46"
  },
  "ID":"4001",
  "Movie":"Glass (2019)",
  "Genres":"Drama|Fantasy"
}
```



Filtro para ambos documentos:

```
{ID: {$in:["4000","4001"]}}
```

The screenshot shows the MongoDB Compass interface. The left sidebar lists databases and collections, with 'JOG.movies' selected. The main pane displays the results of a query. The first document is for 'Avengers: Endgame (2019)' with id 4000, and the second is for 'Glass (2019)' with id 4001.

_id	Movie	Genre
ObjectId("5f1cd35b1254be32d7416e42")	Avengers: Endgame (2019)	Fantasy Sci-Fi
ObjectId("5f1cd3ae1254be32d7416e43")	Glass (2019)	Drama Fantasy

2.- Modificar el documento con `id=4001` en la Colección `movies` para que contenga la siguiente información:

```
{
  id:"4001",
  titulo:"Glass (2019)",
  genres:"Drama|Fantasy",
  valoraciones: [
    {
      userid: "1563",
      movieid: "4001",
      rating: "4"
    },
    {
      userid: "434",
      movieid: "4001",
      rating: "5"
    }
  ]
}
```

MongoDB Compass Connect Edit View Collection Window Help

Sáb 20:32 Berty

Local

HOSTS cluster0-shard-00-02.yx5q... cluster0-shard-00-00.yx5q... cluster0-shard-00-01.yx5q...

CLUSTER Replica Set (atlas-wbhqw1-...) 3 Nodes

EDITION MongoDB 4.2.8 Enterprise

Filter your data

JOG

movies

ratings

users

> Postwork

> admin

> config

> local

> sample_airbnb

> sample_analytics

> sample_geospatial

> sample_mflix

> sample_restaurants

JOG.movies

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {ID: {\$in: ["4000", "4001"]}}

ADD DATA VIEW

Displaying documents 1 - 2 of 2

`_id: ObjectId("5f1cd3ae1254be32d7416e43")
ID: "4001"
Movie: "Avengers: Endgame (2019)"
Genre: "Fantasy|Sci-Fi"

Valaciones: Array
 0: Object
 userID: "1563"
 movieId: "4001"
 rating: "4"
 1: Object
 userId: "434"
 movieId: "4001"
 rating: "5"`

Icons: Finder, Safari, Google Chrome, Calendar, Notes, Photos, Messages, FaceTime, App Store, Spotify, Terminal, Apple TV, Mail, Reminders, iBooks, Settings.

Sesión 08 Query competition

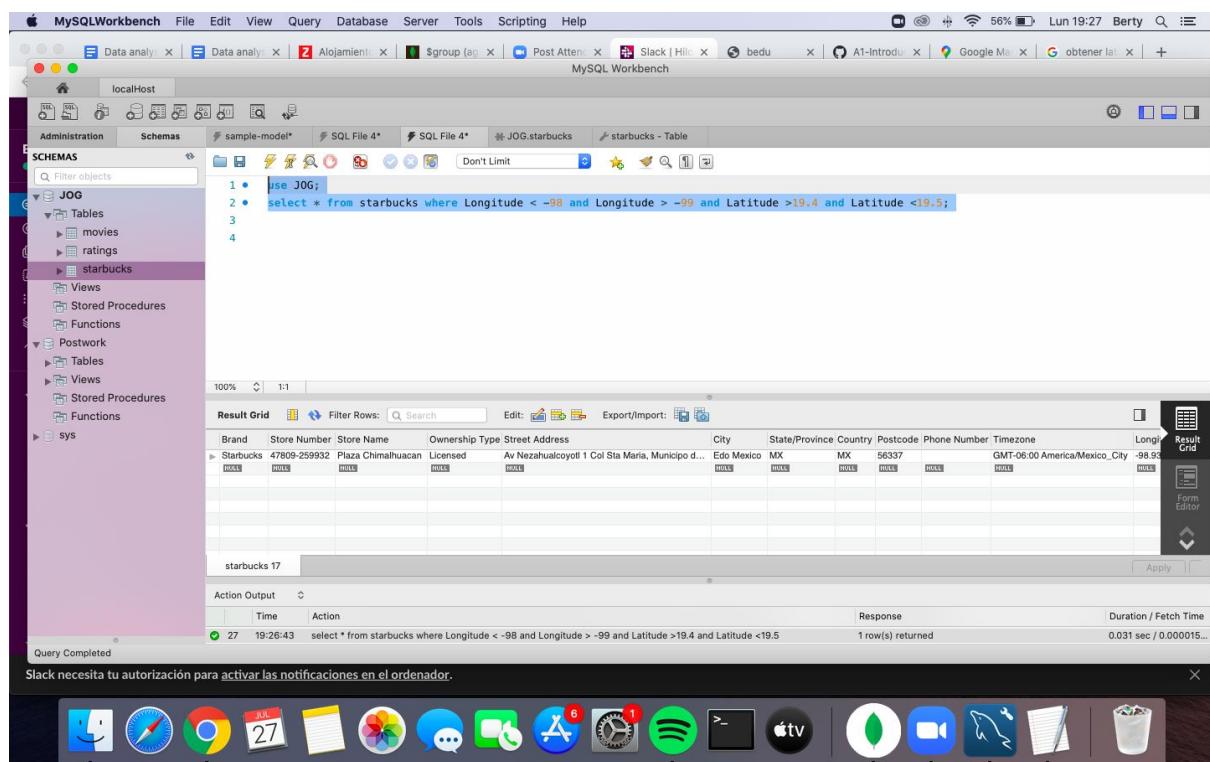
Proyecto 8

Reto 1:

Descarga la fuente de datos de los locales de Starbucks “directory.csv”. Analiza y limpia los datos. Elige MySQL o MongoDB y crea una base de datos para el conjunto de datos del reto. Carga los datos en la base de datos que elegiste y revisa que éstos se muestren correctamente.

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.

```
use JOG;  
select * from starbucks where Longitude < -98 and Longitude > -99 and Latitude >19.4 and  
Latitude <19.5;
```



The screenshot shows the MySQL Workbench interface. In the top navigation bar, the schema is set to 'localHost' and the current database is 'JOG'. The left sidebar shows the 'SCHEMAS' tree with 'JOG' selected, containing tables like 'movies', 'ratings', and 'starbucks'. The main area contains two SQL statements:

```
1 • use JOG;  
2 • select * from starbucks where Longitude < -98 and Longitude > -99 and Latitude >19.4 and  
Latitude <19.5;
```

Below the SQL editor is a 'Result Grid' showing the query results. The columns are: Brand, Store Number, Store Name, Ownership Type, Street Address, City, State/Province, Country, Postcode, Phone Number, Timezone, and Longitude. One row is visible:

Brand	Store Number	Store Name	Ownership Type	Street Address	City	State/Province	Country	Postcode	Phone Number	Timezone	Longitude
Starbucks	47809-259932	Plaza Chimalhuacan	Licensed	Av Nezahualcoyotl 1 Col Sta Maria, Municipio d...	Edo Mexico	MX	MX	56337	GMT-06:00 America/Mexico_City	98.93	

At the bottom of the interface, there is a status bar indicating 'Query Completed' and a notification from Slack asking for permission to activate notifications.

Reto 2

Descarga la fuente de datos de los locales del reto “Pandemic (H1N1) 2009.csv”. Analiza y limpia los datos. Elige MySQL o MongoDB y crea una base de datos para el conjunto de datos del reto. Carga los datos en la base de datos que elegiste y revisa que éstos se muestren correctamente. Responde a las siguientes preguntas usando consultas:

¿Cuál fue el país con mayor número de muertes?

```
[$sort: {  
  Cases: -1  
}, {$match: {  
  Country: {  
    $ne: 'Grand Total'  
},  
  Deaths: {  
    $ne: NaN  
}  
}, {$group: {  
  _id: '$Country',  
  Total: {  
    $sum: '$Deaths'  
}  
}, {$sort: {  
  Total: -1  
}, {$limit: 1}]
```

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

- Left Panel:** Shows the "Local" database with 13 DBs and 45 collections. A sidebar lists collections: JOG, COVID-19, H1N1, Starbucks, movies, ratings, users, and several sample collections.
- Top Bar:** Title "JOG.H1N1 Aggregations". Status bar shows DOCUMENTS 1.8k, TOTAL SIZE 158.7KB, AVG. SIZE 89B, INDEXES 1, TOTAL SIZE 32.0KB, AVG. SIZE 32.0KB.
- Aggregations Tab:** Active tab. Pipeline stages:
 - 1. `{ $sort: { Cases: -1 } }`
 - 2. `{ $match: { Country: { $ne: 'Grand Total' } }, { Deaths: { $ne: NaN } } }`
 - 3. `{ $group: { _id: '$Country', Total: { $sum: '$Deaths' } } }`
 - 4. `{ $sort: { Total: -1 } }`
 - 5. `{ $limit: 1 }`
- Output:** Two sections show the results of the aggregation.
 - Output after \$sort stage:** Sample of 20 documents. Shows two documents:
 - `_id: "Mexico"` `Total: 2271`
 - `_id: "United States of America"` `Total: 1150`
 - Output after \$limit stage:** Sample of 1 document. Shows the same Mexico document:
 - `_id: "Mexico"` `Total: 2271`

¿Cuál fue el país con menor número de muertes?

```
[$sort: {  
  Cases: -1  
}, {$match: {  
  Country: {  
    $ne: 'Grand Total'  
},  
  Deaths: {  
    $ne: NaN  
}  
}, {$group: {  
  _id: '$Country',  
  Total: {  
    $sum: '$Deaths'  
}  
}, {$sort: {  
  Total: 1  
}, {$match: {  
  Total: {$gt: 0}  
}, {$limit: 1}]
```

The screenshot shows the MongoDB Aggregations interface. On the left, the database and collection list is visible, with 'JOG.H1N1' selected. The main area displays the aggregation pipeline stages:

- Stage 1:** An empty stage.
- Stage 2:** A `$match` stage with the query: `/* query: The query in MQL. */ { Total: { $gt: 0 } }`. It shows results for 'Antigua and Barbuda' (Total: 0) and 'Germany' (Total: 0).
- Stage 3:** An empty stage.
- Stage 4:** A `$limit` stage with the value 1. It shows the result for 'Paraguay' (Total: 1).

The top right shows the overall statistics: DOCUMENTS 1.8k, TOTAL SIZE 158.7KB, AVG. SIZE 89B, INDEXES 1, TOTAL SIZE 32.0KB, AVG. SIZE 32.0KB. The bottom right shows 'SAMPLE MODE' and 'AUTO PREVIEW' toggles.

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

```
[{$sort: {
  Cases: -1
}}, {$match: {
  Country: {
    $ne: 'Grand Total'
  },
  Deaths: {
    $ne: NaN
  }
}}, {$group: {
  _id: '$Country',
  Total: {
    $sum: '$Cases'
  }
}}, {$sort: {
  Total: -1
}}, {$match: {
  Total: {$gt: 0}
}}, {$limit: 1}]
```

The screenshot shows the MongoDB Aggregations interface. On the left, the sidebar displays the database structure with 'JOG' selected. The main area shows a pipeline with two stages: a \$match stage and a \$limit stage. The output of the \$match stage is a sample of 20 documents, and the output of the \$limit stage is a single document.

Aggregation Pipeline:

```
[{$match: {
  "query": {
    "$gt": 0
  }
}}, {"$limit": 1}]
```

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

_id	Total
"United States of America"	340804
"Mexico"	142567
...	...

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

_id	Total
"United States of America"	340804

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

```
[{$sort: {
  Cases: -1
}, {$match: {
  Country: {
    $ne: 'Grand Total'
  },
  Deaths: {
    $ne: NaN
  }
}}, {$group: {
  _id: '$Country',
  Total: {
    $sum: '$Cases'
  }
}}, {$sort: {
  Total: 1
}}, {$match: {
  Total: {$gt: 0}
}}, {$limit: 1}]
```

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

- Left Sidebar:** Shows the database structure with "JOG" selected. Under "JOG", there are collections: Starbucks, movies, ratings, users, JOG_Postwork, admin, config, local, sample_airbnb, sample_analytics, sample_geospatial, sample_mflix, sample_restaurants, sample_supplies, and sample_training.
- Top Bar:** Shows the database name "JOG.H1N1" and the collection "Aggregations". It also displays statistics: DOCUMENTS 1.8k, TOTAL SIZE 158.7KB, AVG. SIZE 89B, INDEXES 1, TOTAL SIZE 32.0KB, and AVG. SIZE 32.0KB.
- Aggregation Pipeline:** The pipeline consists of several stages:
 - Initial stage: `1 {`, `2 { Total: 1`, `3 }`
 - \$match:** `1 {/** query: The query in MQL.`, `2 /*/`, `3 {`, `4 { Total: {$gt: 0}`, `5 }`, `6 }`
 - \$limit:** `1 1`
- Output Stages:**
 - Output after \$match stage:** Shows documents for France, Martinique, FOC and Saint Martin, FOC, both with Total: 1.
 - Output after \$limit stage:** Shows documents for Netherlands, Curacao, OT and United Kingdom, Isle of Man, Crown Dependency, both with Total: 1.
 - Final Output:** Shows the document for France, French Polynesia, FOC with Total: 1.

¿Cuál fue el número de muertes promedio?

```
[{$sort: {
  Cases: -1
}}, {$match: {
  Country: {
    $ne: 'Grand Total'
  },
  Deaths: {
    $ne: NaN
  }
}}, {$group: {
  _id: '$Country',
  Total: {
    $sum: '$Deaths'
  }
}}, {$match: {
  Total: {$gt: 0}
}}, {$group: {
  _id: 'Promedio',
  Promedio: {$avg: '$Total'}
}}]
```

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

- Local** database selected.
- 13 DBs, 45 COLLECTIONS** listed in the sidebar.
- JOG.H1N1** collection selected.
- Aggregations** tab selected.
- Documents** tab selected.
- Aggregation Pipeline:**
 - Stage 1: `$_id: '$Country'`, `Total: { $sum: '$Deaths' }`
 - Stage 2: `$_id: "France"`, `Total: 0`
 - Stage 3: `$_id: "Mar"`, `Total: 0`
- Output after \$match stage:** (Sample of 20 documents)
 - `$_id: "Brazil"`, `Total: 3`
 - `$_id: "Colombia"`, `Total: 19`
- Output after \$group stage:** (Sample of 1 document)
 - `$_id: "Promedio"`, `Promedio: 194.52380952380952`

¿Cuál fue el número de casos promedio?

```
[{$sort: {
  Cases: -1
}}, {$match: {
  Country: {
    $ne: 'Grand Total'
  },
  Deaths: {
    $ne: NaN
  }
}}, {$group: {
  _id: '$Country',
  Total: {
    $sum: '$Cases'
  }
}}, {$match: {
  Total: {$gt: 0}
}}, {$group: {
  _id: 'Promedio',
  Promedio: {$avg: '$Total'}
}}]
```

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

- Local** database selected.
- HOSTS**: cluster0-shard-00-00-yx5q..., cluster0-shard-00-01-yx5q..., cluster0-shard-00-02-yx5q... listed.
- CLUSTER**: Replica Set (atlas-wbhqwi-... 3 Nodes).
- EDITION**: MongoDB 4.2.8 Enterprise.
- JOG** collection selected.
- Aggregations** tab selected.
- Documents** tab: Pipeline stage 1: `/* query: The query in MQL. */ { "query": { "Total": { "$gt": 0 } } }`. Output after `$match` stage (Sample of 20 documents):
 - `_id: "French Polynesia, FOC"` `Total: 15`
 - `_id: "Guadalupe, FOC"` `Total: 2`
- Aggregations** tab: Pipeline stage 2: `/* id: The id of the group. * fieldN: The first field name. */ { "_id": "Promedio", "Promedio: { $avg: '$Total' } }` . Output after `$group` stage (Sample of 1 document):
 - `_id: "Promedio"` `Promedio: 5783.358108108108`

Top 5 de países con más muertes

```
[{$sort: {  
  Cases: -1  
}, {$match: {  
  Country: {  
    $ne: 'Grand Total'  
},  
  Deaths: {  
    $ne: NaN  
}  
}, {$group: {  
  _id: '$Country',  
  Total: {  
    $sum: '$Deaths'  
}  
}, {$sort: {  
  Total: -1  
}, {$match: {  
  Total: {$gt: 0}  
}, {$limit: 5}]
```

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

- Left Sidebar:** Shows the "Local" database with 13 DBs and 45 collections. The "JOG" collection is selected.
- Top Bar:** Shows the database name "JOG.H1N1" and the stage name "Aggregations". It also displays document counts (1.8k), total size (158.7KB), average size (89B), and index counts (1).
- Stages:**
 - \$match:** A query stage with the following MQL:

```
1 /**
2  * query: The query in MQL.
3 */
4 + {
5   Total: {$gt: 0}
6 }
```
 - \$limit:** A limit stage with value 5.
- Results:**
 - Output after \$match stage:** A sample of 20 documents. Two examples are shown:
 - `_id: "Mexico"` Total: 2271
 - `_id: "United States of America"` Total: 1150
 - Output after \$limit stage:** A sample of 5 documents. Three examples are shown:
 - `ates of America"`
 - `_id: "Canada"` Total: 194
 - `_id: "Argentina"` Total: 175

Top 5 de países con menos muertes

```
[$sort: {
  Cases: -1
}, {$match: {
  Country: {
    $ne: 'Grand Total'
  },
  Deaths: {
    $ne: NaN
  }
}}, {$group: {
  _id: '$Country',
  Total: {
    $sum: '$Deaths'
  }
}}, {$sort: {
  Total: 1
}}, {$match: {
  Total: {$gt: 0}
}}, {$limit: 5}]
```

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

- Left Sidebar (Local):** Shows 13 DBs and 45 collections. The JOG.H1N1 collection is selected.
- Top Bar:** JOG.H1N1 Aggregations. Status: DOCUMENTS 1.8k TOTAL SIZE 168.7KB AVG. SIZE 89B | INDEXES 1 TOTAL SIZE 32.0KB AVG. SIZE 32.0KB.
- Aggregation Pipeline:**
 - Stage 1:** \$match (Sample of 20 documents). Query:

```
1 /**
2  * query: The query in MQL.
3 */
4 + {
5   Total: {$gt: 0}
6 }
```

 Results:
 - `_id: "Paraguay"` `Total: 1`
 - `_id: "Costa Rica"` `Total: 2`
 - Stage 2:** \$limit (Sample of 5 documents). Results:
 - `1 5`
 - `_id: "Paraguay"` `total: 1`
 - `_id: "Costa Rica"` `Total: 2`
- Bottom Buttons:** ADD STAGE, SAVE, and other pipeline controls.

Reto 3

Descarga la fuente de datos de los locales del reto “2019-nCoV-cases-JHU.csv”. Analiza y limpia los datos. Elige MySQL o MongoDB y crea una base de datos para el conjunto de datos del reto. Carga los datos en la base de datos que elegiste y revisa que éstos se muestren correctamente. Responde a las siguientes preguntas usando consultas:

¿Cuál es país con mayor número de casos?

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

The screenshot shows the MongoDB Aggregations interface. On the left, the sidebar lists databases and collections, with 'JOG' selected. In the main area, the 'Aggregations' tab is active. The pipeline consists of three stages:

```
1. {  
  _id: '$Region',  
  maxCasesPerCountry: {  
    $sum: '$Confirmed'  
  }  
}  
2. {$sort: {  
  maxCasesPerCountry: -1  
}}  
3. {$limit: 1}
```

The output after the sort stage shows two documents: one for Mainland China with 2369152 cases and another for South Korea with 25723 cases. The output after the limit stage shows only the document for Mainland China.

¿Cuál es el país con mayor número de muertes?

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

The screenshot shows the MongoDB Aggregations interface. On the left, the sidebar displays the database structure with 'JOG' selected. The main area shows the aggregation pipeline stages:

- Stage 1:** An initial stage with the following pipeline document:

```
1 + {  
2   _id: '$Region',  
3   maxDeathsPerCountry: {  
4     $sum: '$Deaths'  
5   }  
6 }
```
- Stage 2:** A \$sort stage with the following pipeline document:

```
1 + {  
2   maxDeathsPerCountry: -1  
3 }
```

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

<code>_id: "Nepal"</code>	<code>maxDeathsPerCountry: 0</code>
<code>_id: "Algeria"</code>	<code>maxDeathsPerCountry: 0</code>
<code>_id: "Mainland China"</code>	<code>maxDeathsPerCountry: 65325</code>
<code>_id: "Iran"</code>	<code>maxDeathsPerCountry: 368</code>
- Stage 3:** A \$limit stage with the value '1'. Output after \$limit stage (Sample of 1 document):

<code>_id: "Mainland China"</code>	<code>maxDeathsPerCountry: 65325</code>
------------------------------------	---

Usando las coordenadas, encuentra el epicentro del virus.

```
[{$match: {  
    Lat: {  
        $ne: "  
    },  
    Long: {  
        $ne: "  
    }  
}, {$group: {  
    _id: null,  
    size: {  
        $sum: 1  
    },  
    sumLat: {  
        $sum: '$Lat'  
    },  
    sumLong: {  
        $sum: '$Long'  
    },  
    avgLat: {  
        $avg: '$Lat'  
    },  
    avgLong: {  
        $avg: '$Long'  
    }  
}}, {$project: {  
    checkAvgLat: {  
        $divide: [  
            '$sumLat',  
            '$size'  
        ]  
    },  
    checkAvgLong: {  
        $divide: [  
            '$sumLong',  
            '$size'  
        ]  
    }  
}}]
```

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

- Local** database selected.
- COLLECTIONS**: JOG.COVID-19
- EDITION**: MongoDB 4.2.8 Enterprise
- HOSTS**: cluster0-shard-00-00.yb5q..., cluster0-shard-00-01.yb5q..., cluster0-shard-00-02.yb5q...
- CLUSTER**: Replica Set (atlasis-wbhqw1...), 3 Nodes
- JOGL** collection selected.
- Aggregations** tab selected.
- Stages:**
 - \$group**:

```
1 -{  
2   _id: null,  
3   size: {  
4     $sum: 1  
5   },  
6   sumLat: {  
7     $sum: '$Lat'  
8   },  
9   sumLong: {  
10    $sum: '$Long'  
11  },  
12  avgLat: {  
13    $avg: '$Lat'  
14  },  
15  avgLong: {  
16    $avg: '$Long'  
17  }  
18 }
```
 - \$project**:

```
1 -{  
2   checkAvgLat: {  
3     $divide: [  
4       '$sumLat',  
5       '$size'  
6     ]  
7   },  
8   checkAvgLong: {  
9     $divide: [  
10      '$sumLong',  
11      '$size'  
12    ]  
13  }  
14 }
```
- Output after \$group stage**: (Sample of 1 document)

```
_id: null  
size: 109  
sumLat: 12233.0693  
sumLong: 12393.6979  
avgLat: 2.088841898492249  
avgLong: 2.631917158632486
```
- Output after \$project stage**: (Sample of 1 document)

```
_id: null  
checkAvgLat: 2.088841898492249  
checkAvgLong: 2.631917158632486
```

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

```
[$group: {  
  _id: "$Region",  
  Lat: {$avg: "$Lat"},  
  Long: {$avg: "$Long"}  
}, {$match: {  
  $and: [  
    {Lat: {$lt: 2}},  
    {Long: {$lt: 2}},  
    {Lat: {$gt: -1}},  
    {Long: {$gt: -1}},  
  ]  
}}]
```

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

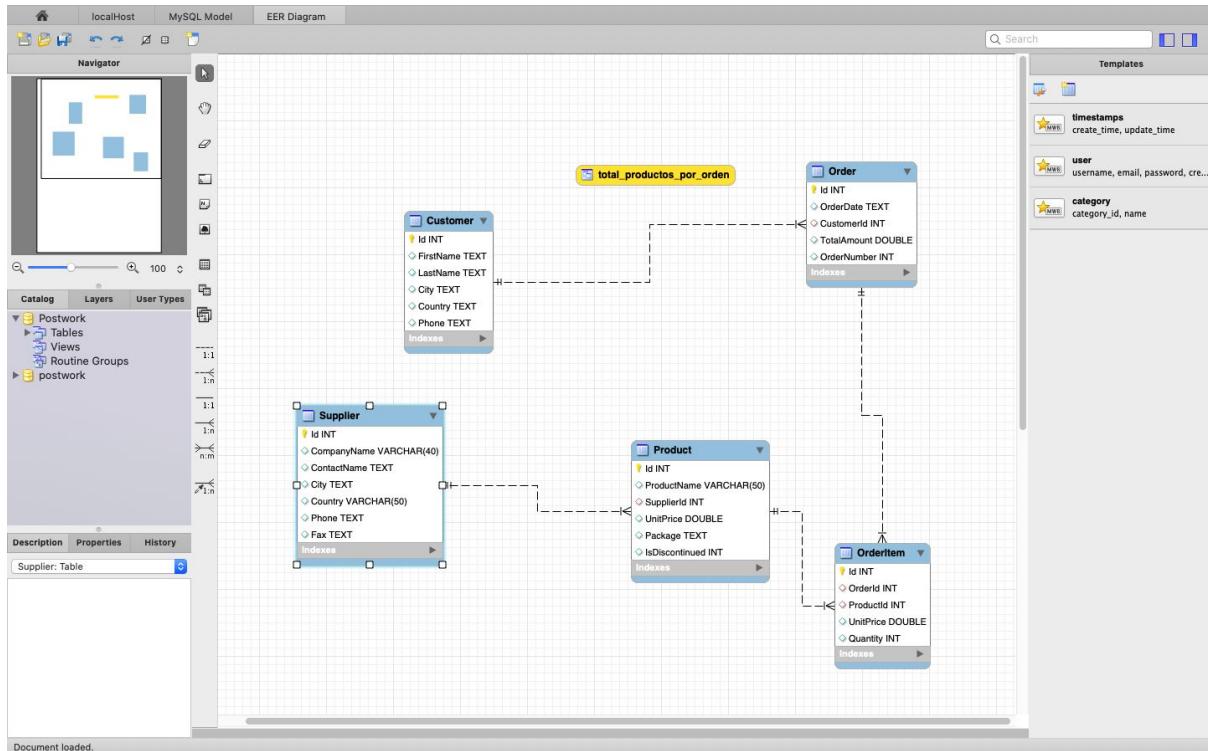
- Local** database selected.
- HOSTS**: cluster0-shard-00-00.yx5q..., cluster0-shard-00-01.yx5q..., cluster0-shard-00-02.yx5q...
- CLUSTER**: Replica Set (atlas-wbhqwi-... 3 Nodes
- EDITION**: MongoDB 4.2.8 Enterprise
- Filter your data**: JOG selected.
- JOG.COVID-19** collection selected.
- Aggregations** tab selected.
- Pipeline Stages:**
 - \$group**:

```
1 *  
2 * _id: The id of the group.  
3 * fieldN: The first field name.  
4 */  
5 * {  
6   _id: "$Region",  
7   Lat: {$avg: "$Lat"},  
8   Long: {$avg: "$Long"}  
9 }
```
 - \$match**:

```
1 *  
2 * query: The query in MQL.  
3 */  
4 * {  
5   $and: [  
6     {Lat: {$lt: 2}},  
7     {Long: {$lt: 2}},  
8     {Lat: {$gt: -1}},  
9     {Long: {$gt: -1}},  
10    ]  
11  }  
12 }
```
- Output after \$group stage**: (Sample of 20 documents)
 - Region: "Mainland China"
Lat: 36.0
Long: 104.0
Confirmed: 843200
Deaths: 3123
Recovered: 362088
 - Region: "South Korea"
Lat: 37.5
Long: 127.0
Confirmed: 5186
Deaths: 28
Recovered: 30
- Output after \$match stage**: (Sample of 5 documents)
 - _id: "United Arab Emirates"**
Lat: 25.0
Long: 55.0
 - _id: "Bahrain"**
Lat: 26.0
Long: 50.0
 - _id: "Azerbaijan"**
Lat: 40.0
Long: 47.0
 - _id: "North Ireland"**
Lat: 54.0
Long: 0.0

Post Work

0.- Añadir diagrama EER para entender la base de datos cargada:



1.- Seleccionar a todos los clientes que radiquen en México.

`select * from Customer where Country = "Mexico";`

The screenshot shows MySQL Workbench running on a Mac OS X desktop. The query editor displays the previously written SQL query. The results grid shows the following data for customers in Mexico:

ID	First Name	Last Name	City	Country	Phone
2	Ana	Trujillo	México D.F.	Mexico	(5) 555-4729
3	Antonio	Moreno	México D.F.	Mexico	(5) 555-3932
13	Francisco	Chang	México D.F.	Mexico	(5) 555-3982
58	Guillermo	Fernández	México D.F.	Mexico	(5) 552-3745
80	Miguel	Angel Paolino	México D.F.	Mexico	(5) 555-2933

The status bar at the bottom indicates the query completed successfully with 5 rows returned in 0.018 seconds.

-- 2.- Selecciona a todos los clientes que no radiquen en USA Y Argentina

```
select * from Customer where Country <> "USA" and Country <> "Argentina";
```

ID	First Name	Last Name	City	Country	Phone
3	Antonio	Moreno	México D.F.	Mexico	(5) 555-3932
4	Thomas	Hardy	London	UK	(171) 555-7788
5	Carmen	Berglund	Luleå	Sweden	(0) 555-4465
6	Hanna	Martinez	Mannheim	Germany	0621/504600
7	Frédéric	Olivet	Strasbourg	France	88.60.15.31
8	Martin	Sommer	Madrid	Spain	(91) 555-22.82
9	Laurence	Lebihan	Marseille	France	91.24.45.40
10	Elizabeth	Lincoln	Tsawassen	Canada	(604) 555-4729

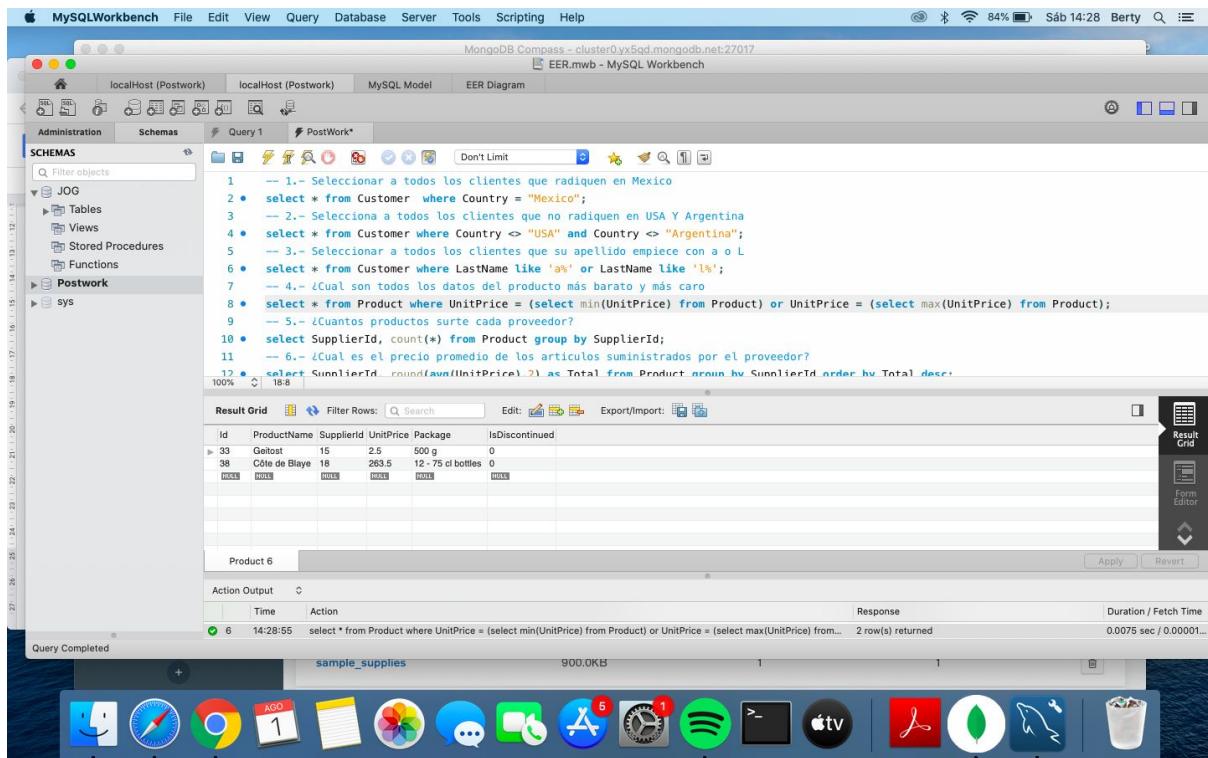
3.- Seleccionar a todos los clientes que su apellido empiece con a o L

```
select * from Customer where LastName like 'a%' or LastName like 'l%';
```

ID	First Name	Last Name	City	Country	Phone
1	Maria	Anders	Berlin	Germany	030-0074321
9	Laurence	Lebihan	Marseille	France	91.24.45.40
10	Elizabeth	Berglund	Luleå	Sweden	(0) 555-4465
11	Victoria	Ashworth	London	UK	(171) 555-1212
15	Pedro	Afonso	Sao Paulo	Brazil	(11) 555-7647
18	Janine	Lar不懈	Nantes	France	40.67.88.88
24	Maria	Larsson	Brücke	Sweden	0695-34.67.21
27	Paolo	Accorti	Torino	Italy	011-4988260
28	Ursula	Cronen	Paris	USA	(500) 555-6074

4.- ¿Cual son todos los datos del producto más barato y más caro

select * from Product where UnitPrice = (select min(UnitPrice) from Product) or UnitPrice = (select max(UnitPrice) from Product);



The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

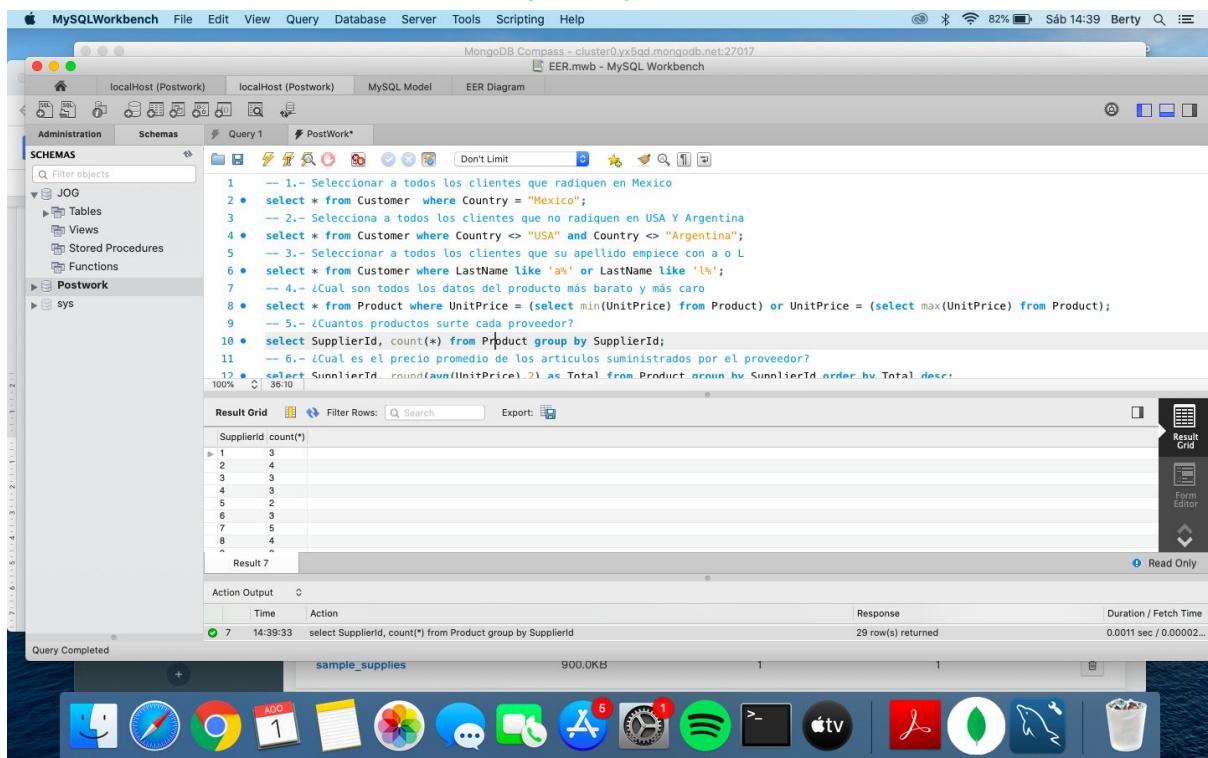
```
1 -- 1.- Seleccionar a todos los clientes que radiquen en Mexico
2 • select * from Customer where Country = "Mexico";
3 -- 2.- Selecciona a todos los clientes que no radiquen en USA Y Argentina
4 • select * from Customer where Country <> "USA" and Country <> "Argentina";
5 -- 3.- Seleccionar a todos los clientes que su apellido empiece con a o L
6 • select * from Customer where LastName like 'a%' or LastName like 'l%';
7 -- 4.- ¿Cuál son todos los datos del producto más barato y más caro
8 • select * from Product where UnitPrice = (select min(UnitPrice) from Product) or UnitPrice = (select max(UnitPrice) from Product);
9 -- 5.- ¿Cuantos productos surte cada proveedor?
10 • select SupplierId, count(*) from Product group by SupplierId;
11 -- 6.- ¿Cuál es el precio promedio de los artículos suministrados por el proveedor?
12 • select SupplierId, round(avg(UnitPrice), 2) as Total from Product group by SupplierId order by Total desc;
```

The result grid shows two rows of data:

SupplierId	Count(*)
33	15
38	18

5.- ¿Cuantos productos surte cada proveedor?

select SupplierId, count(*) from Product group by SupplierId;



The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
1 -- 1.- Seleccionar a todos los clientes que radiquen en Mexico
2 • select * from Customer where Country = "Mexico";
3 -- 2.- Selecciona a todos los clientes que no radiquen en USA Y Argentina
4 • select * from Customer where Country <> "USA" and Country <> "Argentina";
5 -- 3.- Seleccionar a todos los clientes que su apellido empiece con a o L
6 • select * from Customer where LastName like 'a%' or LastName like 'l%';
7 -- 4.- ¿Cuál son todos los datos del producto más barato y más caro
8 • select * from Product where UnitPrice = (select min(UnitPrice) from Product) or UnitPrice = (select max(UnitPrice) from Product);
9 -- 5.- ¿Cuantos productos surte cada proveedor?
10 • select SupplierId, count(*) from Product group by SupplierId;
11 -- 6.- ¿Cuál es el precio promedio de los artículos suministrados por el proveedor?
12 • select SupplierId, round(avg(UnitPrice), 2) as Total from Product group by SupplierId order by Total desc;
```

The result grid shows eight rows of data:

SupplierId	Count(*)
1	3
2	4
3	3
4	3
5	2
6	3
7	5
8	4

-- 6.- ¿Cuál es el precio promedio de los artículos suministrados por el proveedor?

select SupplierId, round(avg(UnitPrice),2) as Total from Product group by SupplierId order by Total desc;

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Contains the SQL query:

```
5 -- 3.- Seleccionar a todos los clientes que su apellido empiece con a o L
6 • select * from Customer where LastName like 'a%' or LastName like 'l%';
7 -- 4.- ¿Cuál son todos los datos del producto más barato y más caro
8 • select * from Product where UnitPrice = (select min(UnitPrice) from Product) or UnitPrice = (select max(UnitPrice) from Product);
9 -- 5.- ¿Cuántos productos surte cada proveedor?
10 • select SupplierId, count(*) from Product group by SupplierId;
11 -- 6.- ¿Cuál es el precio promedio de los artículos suministrados por el proveedor?
12 • select SupplierId, round(avg(UnitPrice),2) as Total from Product group by SupplierId order by Total desc;
13 -- 7.- ¿Cuáles son los 5 nombres de las compañías de los proveedores que más productos surten?
14 • select a.Id, CompanyName, count(*) total from Supplier a
15 left join Product b
16 on a.TdbSupplierID
```
- Result Grid:** Displays the results of the query, showing the SupplierId and Total for each supplier. The data is as follows:

SupplierId	Total
12	44.88
28	44.50
29	38.90
7	35.57
3	31.67
24	30.93
11	29.71
5	29.50

- Action Output:** Shows the query executed and its results.

-- 7.- ¿Cuáles son los 5 nombres de las compañías de los proveedores que más productos surten?

select a.Id, CompanyName, count(*) total from Supplier a

The screenshot shows the MySQL Workbench interface with the following details:

- Query Editor:** Contains the SQL query:

```
5 -- 3.- Seleccionar a todos los clientes que su apellido empiece con a o L
6 • select * from Customer where LastName like 'a%' or LastName like 'l%';
7 -- 4.- ¿Cuál son todos los datos del producto más barato y más caro
8 • select * from Product where UnitPrice = (select min(UnitPrice) from Product) or UnitPrice = (select max(UnitPrice) from Product);
9 -- 5.- ¿Cuántos productos surte cada proveedor?
10 • select SupplierId, count(*) from Product group by SupplierId;
11 -- 6.- ¿Cuál es el precio promedio de los artículos suministrados por el proveedor?
12 • select SupplierId, round(avg(UnitPrice),2) as Total from Product group by SupplierId order by Total desc;
13 -- 7.- ¿Cuáles son los 5 nombres de las compañías de los proveedores que más productos surten?
14 • select a.Id, CompanyName, count(*) total from Supplier a
15 left join Product b
16 on a.TdbSupplierID
```
- Result Grid:** Displays the results of the query, showing the Id, CompanyName, and total for each supplier. The data is as follows:

Id	CompanyName	total
12	Plutzer Lebensmittelgroßmärkte AG	5
7	Pavlova, Ltd.	5
2	New Orleans Cajun Delights	4
8	Specialty Biscuits, Ltd.	4
1	Exotic Liquids	3

- Action Output:** Shows the query executed and its results.

-- 8.- ¿Cuál es el total de ordenes por país?

```
select Country, round(count(b.TotalAmount),2) Total from Supplier a  
left join `Order` b on a.Id=b.CustomerId group by Country order by Total desc;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The query editor window displays the following SQL code:

```
20 -- 8.- ¿el total de ordenes por pais?  
21 • select Country, round(count(b.TotalAmount),2) Total from Supplier a  
22 left join `Order` b  
23 on a.Id=b.CustomerId  
24 group by Country  
25 order by Total desc;  
26 -- 9.- ¿Cuál es el país con más total de ingresos?  
27 • select Country, round(sum(b.TotalAmount),2) Total from Supplier a  
28 left join `Order` b  
29 on a.Id=b.CustomerId  
30 group by Country  
31 order by Total desc;
```

The result grid shows the following data:

Country	Total
Singapore	30
Sweden	23
USA	22
Canada	20
Japan	20
Sydney	19
France	18
Spain	18
...	...
Result	13

The status bar at the bottom indicates "Query Completed".

-- 9.- ¿Cuál es el país con más total de ingresos?

```
select Country, round(sum(b.TotalAmount),2) Total from Supplier a  
left join `Order` b on a.Id=b.CustomerId group by Country order by Total desc limit 1;
```

The screenshot shows the MySQL Workbench interface on a Mac OS X desktop. The query editor window displays the same SQL code as the previous screenshot, but the result grid shows only one row:

Country	Total
Singapore	113236.68

The status bar at the bottom indicates "Query Completed".

-- 10.- ¿Cuáles son las 3 ciudades con más órdenes?

```
select City, count(b.TotalAmount) Total from Supplier a left join `Order` b on a.Id=b.CustomerId group by City order by Total desc limit 3;
```

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

```
30 group by Country
31 order by Total desc
32 limit 1;
33 -- 10.- ¿Cuáles son las 3 ciudades con más órdenes?
34 select City, count(b.TotalAmount) Total from Supplier a
35 left join `Order` b
36 on a.Id=b.CustomerId
37 group by City
38 order by Total desc
39 limit 3;
40 -- 11.- ¿Cuál es el nombre de los 5 productos más vendidos?
41 create view Total_productos_por_orden as (select a.Id, a.ProductName, sum(b.Quantity) Total from Product a
100% | 1:34
```

The result grid displays the top 3 cities with the most orders:

City	Total
Singapore	30
Wendy Mackenzie	19
Oviedo	18

-- 11.- Crea una vista que muestra el total de unidades vendidas por producto

```
create view Total_productos_por_orden as (select a.Id, a.ProductName, sum(b.Quantity)
Total from Product a left join OrderItem b on a.Id=b.ProductId group by a.Id order by Total
desc);
```

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code for creating a view:

```
37 group by City
38 order by Total desc
39 limit 3;
40 -- 11.- Crea una vista que muestra el total de unidades vendidas por producto
41 create view Total_productos_por_orden as (select a.Id, a.ProductName, sum(b.Quantity) Total from Product a
42 left join OrderItem b
43 on a.Id=b.ProductId
44 group by a.Id
45 order by Total desc);
46 -- 11a.- ¿Cuál es el nombre de los 5 productos más vendidos?
47 select * from Total_productos_por_orden limit 5;
48 -- 12.- ¿Cuál es el producto que no se ha vendido?
49 select * from Total_productos_por_orden where Total is NULL;
50 -- 13.- ¿Cuál fue el total de la venta del producto mostrando el proveedor?
51 select a.Id, a.ProductName, round(sum(b.UnitPrice),2) Total, c.CompanyName from Product a
52 left join OrderItem b
53 on a.Id=b.ProductId
100% | 49:47
```

The result grid displays the top 5 products with the highest sales volume:

Id	ProductName	Total
60	Commençant Pielrot	1577
59	Raschita Courdouault	1496
31	Gorgonzola Telino	1387
86	Gnocchi di nonna Alice	1263
16	Pavlova	1158

-- 11a.- ¿Cual es el nombre de los 5 productos más vendidos?

```
select * from Total_productos_por_orden limit 5;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the schema structure under 'Postwork'. The main area shows a query editor with the following code:

```
37 group by City
38 order by Total desc
39 limit 3;
-- 11.- Crea una vista que muestra el total de unidades vendidas por producto
41 • create view Total_productos_por_orden as (select a.Id, a.ProductName, sum(b.Quantity) Total from Product a
42 left join OrderItem b
43 on a.Id=b.ProductId
44 group by a.Id
45 order by Total desc);
-- 11a.- ¿Cual es el nombre de los 5 productos más vendidos?
47 • select * from Total_productos_por_orden limit 5;
-- 12.- ¿Cual es el producto que no se ha vendido?
49 • select * from Total_productos_por_orden where Total is NULL;
-- 13.- ¿Cuál fue el total de la venta del producto mostrando el proveedor?
51 • select a.Id, a.ProductName, round(sum(b.UnitPrice),2) Total, c.CompanyName from Product a
52 left join OrderItem b
53 on a.Id=b.ProductId
```

The result grid shows the following data:

ID	ProductName	Total
60	Camembert Pierrot	1577
59	Raclette Courteau	1496
31	Gorgonzola Telino	1387
58	Gnocchi di nonna Alice	1263
16	Pavlova	1158

At the bottom, the status bar indicates "Query Completed".

-- 12.- ¿Cual es el producto que no se ha vendido?

```
select * from Total_productos_por_orden where Total is NULL;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the schema structure under 'Postwork'. The main area shows a query editor with the following code:

```
37 group by City
38 order by Total desc
39 limit 3;
-- 11.- Crea una vista que muestra el total de unidades vendidas por producto
41 • create view Total_productos_por_orden as (select a.Id, a.ProductName, sum(b.Quantity) Total from Product a
42 left join OrderItem b
43 on a.Id=b.ProductId
44 group by a.Id
45 order by Total desc);
-- 11a.- ¿Cual es el nombre de los 5 productos más vendidos?
47 • select * from Total_productos_por_orden limit 5;
-- 12.- ¿Cual es el producto que no se ha vendido?
49 • select * from Total_productos_por_orden where Total is NULL;
-- 13.- ¿Cuál fue el total de la venta del producto mostrando el proveedor?
51 • select a.Id, a.ProductName, round(sum(b.UnitPrice),2) Total, c.CompanyName from Product a
52 left join OrderItem b
53 on a.Id=b.ProductId
```

The result grid shows the following data:

ID	ProductName	Total
78	Stroopwafels	NULL

At the bottom, the status bar indicates "Query Completed".

-- 13.- ¿Cual fue el total de la venta del producto mostrando el proveedor?

```
select a.Id, a.ProductName, round(sum(b.UnitPrice),2) Total, c.CompanyName from Product
a left join OrderItem b on a.Id=b.ProductId left join Supplier c on c.Id=a.SupplierId group by
a.Id order by Total desc;
```

```

11a.- Crea una vista que muestra el total de unidades vendidas por producto
11a. select * from Total_productos_por_orden limit 5;
11b.- ¿Cuál es el producto que no se ha vendido?
11c.- ¿Cuál fue el total de la venta del producto mostrando el proveedor?
11d.- ¿Cuál son los 5 clientes que más dinero han gastado y cuantos productos compraron?
11e.- select concat(a.FirstName, " ", a.LastName) Nombre, round(sum(b.TotalAmount),2) Total from Customer a
      left join `Order` b on a.Id=b.CustomerId left join OrderItem c on b.Id=c.OrderId group by Nombre order by
      Total desc limit 5;
    
```

Id	ProductName	Total	CompanyName
36	Côte de Blaye	5902.40	Aux joyeux ecclésiastiques
29	Thüringer Rostbratwurst	3713.38	Plutzer Lebensmittelgroßmärkte AG
59	Raclette Courdavault	2761.00	Gai pâturage
62	Tarte au sucre	2227.80	Foires d'étables'
51	Manjimup Dried Apples	1971.60	Gray
56	Gnocchi di nonna Alice	1770.00	Pasta Buttini s.r.l.
60	Camembert Pierrot	1638.80	Gai pâturage
19	Le Petit Mâconnais	1638.00	Les Saveurs du soleil
28	Rössle Sauerkraut	1385.20	Plutzer Lebensmittelgroßmärkte AG
17	Alice Mutton	1349.40	Pavlova, Ltd.
72	Mozzarella di Giovanni	1217.40	Formaggi Fortini s.r.l.
20	Sir Rodney's Marmalade®	1215.00	Specialty Biscuits, Ltd.
43	Ipoh Coffee	1205.20	Leka Trading
69	Gubbrandsdalost	1036.80	Norske Meierier

-- 14.- ¿Cual son los 5 clientes que más productos compraron y cuanto dinero han gastado?

```
select concat(a.FirstName, " ", a.LastName) Nombre, round(sum(b.TotalAmount),2)
```

```
Total_Quantity, sum(c.Quantity) Total_Amount from Customer a left join `Order` b on
a.Id=b.CustomerId left join OrderItem c on b.Id=c.OrderId group by Nombre order by
Total_Quantity desc limit 5;
```

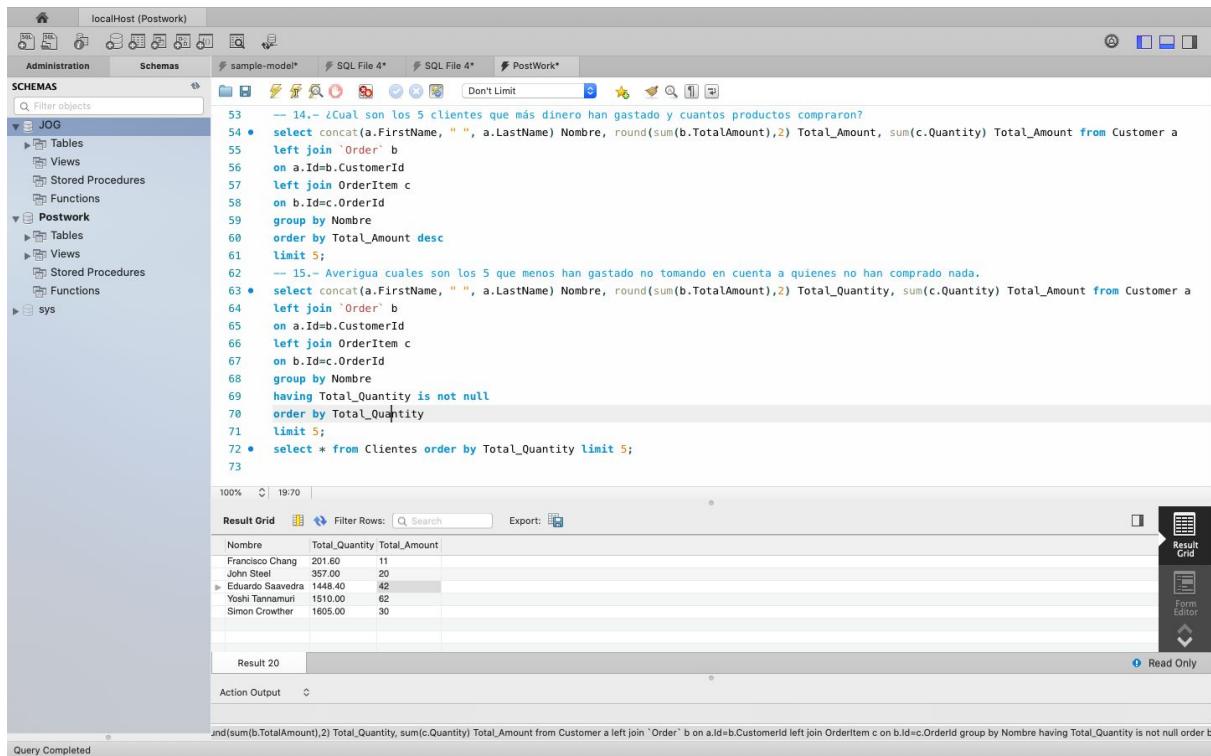
```

11a.- ¿Cuál es el nombre de los 5 productos más vendidos?
11b.- ¿Cuál es el producto que no se ha vendido?
11c.- ¿Cuál fue el total de la venta del producto mostrando el proveedor?
11d.- ¿Cuál son los 5 clientes que más dinero han gastado y cuantos productos compraron?
11e.- select concat(a.FirstName, " ", a.LastName) Nombre, round(sum(b.TotalAmount),2) Total_Amount, sum(c.Quantity) Total_Quantity from Customer a
      left join `Order` b on a.Id=b.CustomerId left join OrderItem c on b.Id=c.OrderId
      group by Nombre
      order by Total_Amount desc
    
```

Nombre	Total_Amount	Total_Quantity
Jose Manzanares	48107.00	4958
Herman Cervantes	426349.25	3456
Horst Kloss	417617.35	3961
Paula Wilson	179134.60	1383
Patricia McKenna	156949.88	1684

-- 15.- Averigua cuales son los 5 que menos han gastado no tomando en cuenta a quienes no han comprado nada.

```
select concat(a.FirstName, " ", a.LastName) Nombre, round(sum(b.TotalAmount),2)
Total_Quantity, sum(c.Quantity) Total_Amount from Customer a left join `Order` b on
a.Id=b.CustomerId left join OrderItem c on b.Id=c.OrderId group by Nombre having
Total_Quantity is not null order by Total_Quantity limit 5;
```



The screenshot shows the SQL Server Management Studio interface with the following details:

- Object Explorer:** Shows the schema structure under JOG, including Tables, Views, Stored Procedures, Functions, Postwork, and sys.
- Query Editor:** Displays the SQL query for question 15. The code includes comments explaining the joins and grouping. The final part of the query uses a subquery to filter rows where Total_Quantity is not null.
- Results Grid:** Shows the output of the query. The columns are Nombre, Total_Quantity, and Total_Amount. The data is as follows:

Nombre	Total_Quantity	Total_Amount
Francisco Chang	20.00	11
John Smith	287.00	20
Edwards Saez	1448.40	62
Yoshi Tannamuri	1510.00	62
Simon Crowther	1605.00	30

16.- Obtén los datos de contacto de cada compañía de proveedores

```
{  
  project: {  
    CompanyName: 1,  
    Phone: 1,  
    Fax: 1  
  }  
}
```

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases (Local, JOG, JOG_Postwork) and collections (Supplier). The main area shows the 'JOG_Postwork.Supplier' collection with 29 documents. The 'FILTER' panel has a query: {CompanyName: 1, Phone: 1, Fax: 1}. The results table shows four document snippets:

```
_id: ObjectId("5f1f5de719f51f02bc58754d")
CompanyName: "Exotic Liquids"
Phone: "(171) 555-2222"
Fax: "NULL"

_id: ObjectId("5f1f5de719f51f02bc58754e")
CompanyName: "New Orleans Cajun Delights"
Phone: "(100) 555-4822"
Fax: "NULL"

_id: ObjectId("5f1f5de719f51f02bc58754f")
CompanyName: "Grandma Kelly's Homestead"
Phone: "(313) 555-5735"
Fax: "(313) 555-3349"

_id: ObjectId("5f1f5de719f51f02bc587550")
CompanyName: "Tokyo Traders"
Phone: "(03) 3555-5011"
Fax: "NULL"
```

The right sidebar shows a history of past queries with their execution times and projection definitions:

- Mon Jul 27 2020 18:07:11 GMT-0500 (CET)
PROJECT
{
 CompanyName: 1,
 Phone: 1,
 Fax: 1
}
- Mon Jul 27 2020 18:06:42 GMT-0500 (CET)
PROJECT
{
 CompanyName: 1,
 City: 1
}
- Mon Jul 27 2020 18:06:35 GMT-0500 (CET)
PROJECT
{
 CompanyName: 1
}
- Mon Jul 27 2020 18:04:44 GMT-0500 (CET)
PROJECT
{
 Country: 1
}

17.- Filtra a los proveedores del reino unido

```
{  
  filter: {  
    Country: 'UK'  
  }  
}
```

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases (Local, JOG, JOG_Postwork) and collections (Supplier, admin, config, local, sample_airbnb, sample_analytics, sample_geospatial, sample_mflix, sample_restaurants, sample_supplies, sample_training, sample_weatherdata). The main area displays the 'JOG_Postwork.Supplier' collection with 29 documents. A query builder is open, showing a filter for 'Country: 'UK''. The results pane shows two documents:

```
_id:ObjectId("5f1f5de719f51f02bc58754d")
Id:1
CompanyName:"Exotic Liquids"
ContactName:"Charlotte Cooper"
City:"London"
Country:"UK"
Phone:"(171) 555-2222"
Fax:"NULL"

_id:ObjectId("5f1f5de719f51f02bc587554")
Id:8
CompanyName:"Specialty Biscuits Ltd."
ContactName:"Peter Wilson"
City:"Manchester"
Country:"UK"
Phone:"(161) 555-4448"
Fax:"NULL"
```

18.- Muestra a los clientes solo de México o Alemania

```
{
  filter: {
    $or: [
      {
        Country: 'Germany'
      },
      {
        Country: 'Mexico'
      }
    ]
  }
}
```

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

- Left Sidebar:** Shows the Local connection, 13 databases, and 38 collections. The JOG_Postwork database is selected.
- Top Bar:** Shows the database name "JOG_Postwork.Customer" and the collection name "Documents".
- Document List:** Displays 91 documents. The first few results are shown below:
 - Document 1:** _id: ObjectId("5f1f5ef519f51f02bc58756a")
Id: 1
FirstName: "Maria"
LastName: "Ander"City: "Berlin"
Country: "Germany"
Phone: "(030) 0874321"
 - Document 2:** _id: ObjectId("5f1f5ef519f51f02bc58756b")
Id: 2
FirstName: "Ana"
LastName: "Trujillo"
City: "Mexico D.F."
Country: "Mexico"
Phone: "(5) 559-4729"
 - Document 3:** _id: ObjectId("5f1f5ef519f51f02bc58756c")
Id: 3
FirstName: "Antonio"
LastName: "Moreno"
City: "Mexico D.F."
Country: "Mexico"
Phone: "(5) 555-3932"
 - Document 4:** _id: ObjectId("5f1f5ef519f51f02bc58756f")
Id: 6
FirstName: "Manuela"
LastName: "Moss"
City: "Mannheim"
Country: "Germany"
Phone: "0621-08460"
- Filter Bar:** Shows the query: {\$or: [{Country: 'Germany'}, {Country: 'Mexico'}]}.
- Logs:** On the right, there are four log entries showing the query and its execution time.

19.- Ubicar todas las órdenes con precio de unidad mayor a 30

```
{  
  filter: {  
    UnitPrice: {  
      $gt: 30  
    }  
  }  
}
```

The screenshot shows the MongoDB Compass interface. On the left, a sidebar displays the database structure: Local, HOSTS, CLUSTER, and EDITION. Under the 'Order' section, there are sub-items: OrderItem, Supplier, admin, config, local, sample_airbnb, sample_analytics, sample_geospatial, sample_mflix, sample_restaurants, sample_supplies, sample_training, and sample_weatherdata. The main area shows the results of a query on the 'JOG_Postwork.Order' collection. The query is defined in the top bar: FILTER {UnitPrice: {\$gt: 30}}. The results grid displays several documents, each with fields: _id, Id, OrderId, ProductId, UnitPrice, and Quantity. The first few documents are:

_id	Id	OrderId	ProductId	UnitPrice	Quantity
ObjectId("5f1f61a219f51f02bc587905")	3	1	72	34.8	5
ObjectId("5f1f61a219f51f02bc587907")	5	2	51	42.4	40
ObjectId("5f1f61a219f51f02bc587909")	7	3	51	42.4	35
ObjectId("5f1f61a219f51f02bc58790e")	12	5	20	64.8	40
ObjectId("5f1f61a219f51f02bc58791a")	24	8			

20.- ¿Cual es el producto más caro de la lista?

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

The screenshot shows the MongoDB Compass interface. On the left, the sidebar displays the database structure under the 'JOG' project, including collections like Customer, Order, OrderItem, Product, and Supplier. The main window is titled 'JOG_Postwork.Product' and shows a table of documents. A search bar at the top contains the query: {_id: ObjectId("5f1f624b19f51f02bc588193")}. The table has columns for _id, ProductName, SupplierId, UnitPrice, Package, and IsDiscontinued. One row is highlighted, showing the details: _id: ObjectId("5f1f624b19f51f02bc588193"), ProductName: "Côte de Blaye", SupplierId: 18, UnitPrice: 263.5, Package: "12 - 75 cl bottles", IsDiscontinued: false. The bottom status bar indicates the document is 1 of 1.

21.- ¿Muestra por agregación todos los productos que no estan descontinuados?

```
[$match: {  
  IsDiscontinued: false  
}]
```

The screenshot shows the MongoDB Compass interface on a Mac OS X desktop. The main window title is "JOG_Postwork.Product". The left sidebar shows the database structure with "Local" selected, displaying "13 DBS" and "38 COLLECTIONS". The "Aggregations" tab is active. The top right shows "DOCUMENTS 78" and "INDEXES 1". Below the tabs, there's a preview of 78 documents. A query builder panel shows the aggregation pipeline:

```
1 //**  
2 * query: The query in MQL.  
3 */  
4 + {  
5   IsDiscontinued: false  
6 }
```

The output stage shows two sample documents from the \$match stage:

```
_id: ObjectId("5f1f624b19f51f02bc58816e")  
Id: 1  
ProductName: "Chai"  
SupplierId: 1  
UnitPrice: 18  
Package: "10 boxes x 20 bags"  
IsDiscontinued: false
```

```
_id: ObjectId("5f1f624b19f51f02bc58816f")  
Id: 2  
ProductName: "Chang"  
SupplierId: 1  
UnitPrice: 19  
Package: "24 - 12 oz bottle"  
IsDiscontinued: false
```

22.- ¿Cuantos productos surte cada proveedor?

```
[{$match: {  
    IsDiscontinued: false  
}}, {$group: {  
    _id: "$SupplierId",  
    Total: {  
        $sum:1  
    }  
}}]
```

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

- Hosts:** cluster0-shard-00-02.y5q... (selected), cluster0-shard-00-00.y5q..., cluster0-shard-00-01.y5q...
- Cluster:** Replica Set (atlas-wbhqw-...)
- Edition:** MongoDB 4.2.8 Enterprise
- Collection:** JOG_Postwork.Product
- Aggregations Tab:** The pipeline consists of two stages:
 - \$match:** Filters documents where IsDiscontinued is false.
 - \$group:** Groups by SupplierId and calculates the total count of products.
- Output after \$match stage:** Shows sample documents for SupplierId 1 and 2.
- Output after \$group stage:** Shows two groups: one for SupplierId 17 with a total of 3, and another for SupplierId 3 with a total of 3.
- Document Count:** 78 DOCUMENTS
- Indexes:** 1 INDEXES

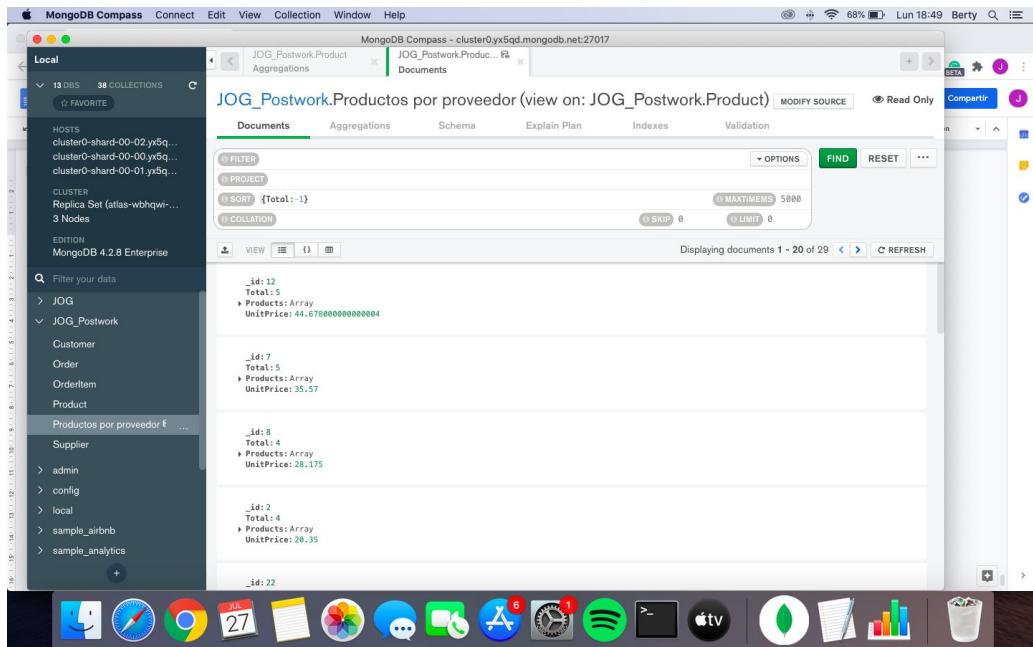
23.- Agrega como un array todos los productos suministrados no descontinuados por cada proveedor y el promedio de los precios de los productos de mayor a menor.

```
[{$match: {
  IsDiscontinued: false
}}, {$group: {
  _id: "$SupplierId",
  Total: {
    $sum:1
  },
  Productos: {$push:"$ProductName"},
  Precioneto: {$avg: "$UnitPrice"}
}}, {$sort: {
  Total: -1
}}]
```

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

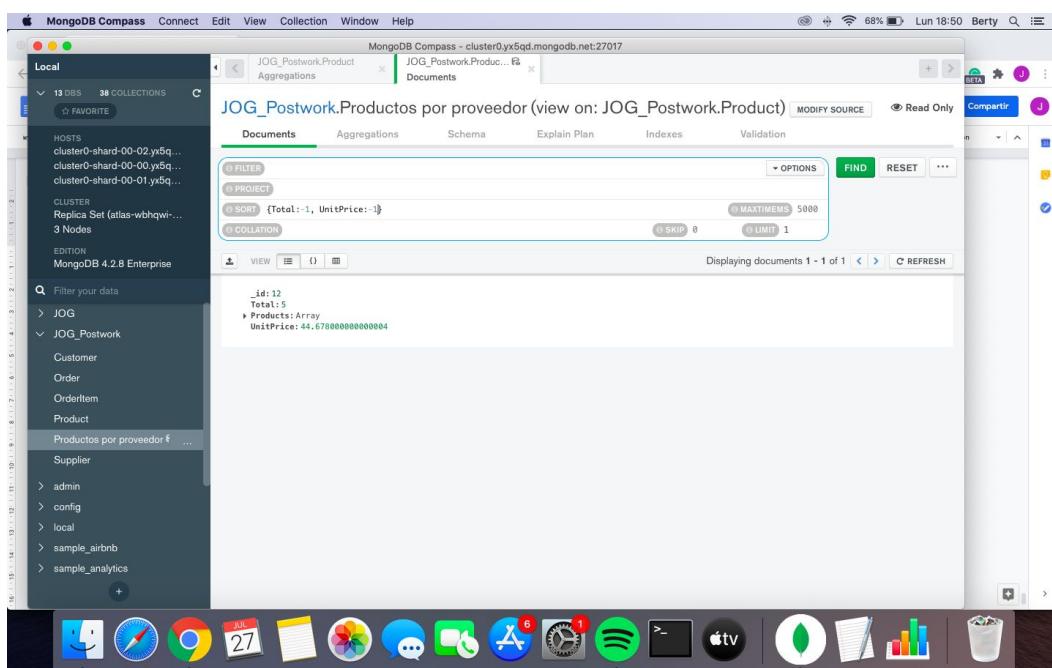
- Hosts:** cluster0-shard-00-02.yx5q... (selected), cluster0-shard-00-00.yx5q..., cluster0-shard-00-01.yx5q...
- Cluster:** Replica Set (atlas-wbhqw-... 3 Nodes)
- Edition:** MongoDB 4.2.8 Enterprise
- Collection:** JOG_Postwork.Product
- Aggregations:** A pipeline is being built with two stages:
 - \$group:** Groups documents by SupplierId. The output shows two groups: one for SupplierId 18 (Total: 2, Productos: ["ProductA", "ProductB"], Precioneto: 140.75) and one for SupplierId 20 (Total: 2, Productos: ["ProductC", "ProductD"], Precioneto: 32.725).
 - \$sort:** Sorts the grouped documents by Total in descending order. The output shows two sorted documents: one for SupplierId 7 (Total: 4, Productos: ["ProductE", "ProductF", "ProductG", "ProductH"], Precioneto: 34.7125) and one for SupplierId 8 (Total: 4, Productos: ["ProductI", "ProductJ", "ProductK", "ProductL"], Precioneto: 28.175).
- Output:** The pipeline has processed 78 documents and created 1 index.

24.- Crea una vista de esta tabla y arroja al proveedor con más productos y el precio promedio más alto



The screenshot shows the MongoDB Compass application running on a Mac OS X desktop. The main window title is "JOG_Postwork.Product". The left sidebar lists databases "Local" and "JOG_Postwork" containing collections like "Customer", "Order", "OrderItem", "Product", and "Productos por proveedor". The right panel displays a table titled "Productos por proveedor (view on: JOG_Postwork.Product)". It has tabs for "Documents", "Aggregations", "Schema", "Explain Plan", "Indexes", and "Validation". Under "Documents", there are five entries, each showing an _id, Total, and Products array. The first entry has _id: 12, Total: 5, and Products: [{UnitPrice: 44.678}, ...]. The second entry has _id: 7, Total: 5, and Products: [{UnitPrice: 35.5}, ...]. The third entry has _id: 8, Total: 4, and Products: [{UnitPrice: 28.175}, ...]. The fourth entry has _id: 3, Total: 4, and Products: [{UnitPrice: 20.35}, ...]. The fifth entry has _id: 22, Total: 1, and Products: [{UnitPrice: 44.678}, ...]. Below the table are buttons for "FIND", "RESET", and "...". The status bar at the bottom shows the date and time as "Lun 18:49" and the battery level as "68%".

```
{
  sort: {
    Total: 1,
    UnitPrice: -1
  },
  limit: 1
}
```



This screenshot shows the same MongoDB Compass interface after applying the aggregation query. The "Sort" field in the query builder is now set to {Total:-1, UnitPrice:-1}. The result table shows a single document with _id: 12, Total: 5, and Products: [{UnitPrice: 44.678}, ...]. The status bar at the bottom shows the date and time as "Lun 18:50" and the battery level as "68%".

25.- De la vista generada crea un documento por cada producto existente que incluya los datos de cada proveedor por producto. Usa esta vista para crear una vista

```
[$lookup: {
  from: 'Supplier',
  localField: 'SupplierId',
  foreignField: 'Id',
  as: 'SupplierId'
}], {$unwind: {
  path: '$SupplierId'
}]
```

The screenshot shows the MongoDB Compass interface with the aggregation pipeline for the "JOG_Postwork.Product" collection. The pipeline consists of two stages: \$lookup and \$unwind.

\$lookup Stage:

```
1. { $lookup: {
  2.   from: 'Supplier',
  3.   localField: 'SupplierId',
  4.   foreignField: 'Id',
  5.   as: 'SupplierId'
  6. }}
```

\$unwind Stage:

```
1. { $unwind: {
  2.   path: '$SupplierId'
  3. }}
```

Both stages are currently active, indicated by the green toggle switch. The output after each stage is shown as a sample of 20 documents. The first document in the \$lookup stage output is:

```
_id: ObjectId("5f1f624b19f51f02bc58816e")
Id: 1
ProductName: "Chai"
SupplierId: Array
  ▶ SupplierId: Object
    Id: 1
    ProductName: "Chai"
    UnitPrice: 18
    Package: "10 boxes x 20 bags"
    IsDiscontinued: false
```

The second document in the \$lookup stage output is:

```
_id: ObjectId("5f1f624b19f51f02bc58816f")
Id: 2
ProductName: "Chang"
SupplierId: Array
  ▶ SupplierId: Object
    Id: 2
    ProductName: "Chang"
    UnitPrice: 19
    Package: "24 - 12 oz bottle"
    IsDiscontinued: false
```

The first document in the \$unwind stage output is:

```
_id: ObjectId("5f1f624b19f51f02bc58816e")
Id: 1
ProductName: "Chai"
  ▶ SupplierId: Object
    Id: 1
    ProductName: "Chai"
    UnitPrice: 18
    Package: "10 boxes x 20 bags"
    IsDiscontinued: false
```

The second document in the \$unwind stage output is:

```
_id: ObjectId("5f1f624b19f51f02bc58816f")
Id: 2
ProductName: "Chang"
  ▶ SupplierId: Object
    Id: 2
    ProductName: "Chang"
    UnitPrice: 19
    Package: "24 - 12 oz bottle"
    IsDiscontinued: false
```

The screenshot shows the MongoDB Compass interface displaying the results of the completed aggregation pipeline. The results are shown in a table format with columns for _id, Id, ProductName, and SupplierId.

_id	Id	ProductName	SupplierId
5f1f624b19f51f02bc58816e	1	Chai	[{"Id": 1, "ProductName": "Chai", "UnitPrice": 18, "Package": "10 boxes x 20 bags", "IsDiscontinued": false}, {"Id": 2, "ProductName": "Chang", "UnitPrice": 19, "Package": "24 - 12 oz bottle", "IsDiscontinued": false}]
5f1f624b19f51f02bc58816f	2	Chang	[{"Id": 1, "ProductName": "Chai", "UnitPrice": 18, "Package": "10 boxes x 20 bags", "IsDiscontinued": false}, {"Id": 2, "ProductName": "Chang", "UnitPrice": 19, "Package": "24 - 12 oz bottle", "IsDiscontinued": false}]
5f1f624b19f51f02bc588178	3	Aniseed Syrup	[{"Id": 1, "ProductName": "Chai", "UnitPrice": 18, "Package": "10 boxes x 20 bags", "IsDiscontinued": false}, {"Id": 2, "ProductName": "Chang", "UnitPrice": 19, "Package": "24 - 12 oz bottle", "IsDiscontinued": false}]
5f1f624b19f51f02bc588171	4	Chai Anise Syrup	[{"Id": 1, "ProductName": "Chai", "UnitPrice": 18, "Package": "10 boxes x 20 bags", "IsDiscontinued": false}, {"Id": 2, "ProductName": "Chang", "UnitPrice": 19, "Package": "24 - 12 oz bottle", "IsDiscontinued": false}]

26.- Dentro de la tabla “Order” usa la vista creada para mostrar qué productos componen cada orden. Crea una vista llamada “OrderId con producto y proveedor”.

```
[{$lookup: {
  from: 'Productos con datos del proveedor',
  localField: 'ProductId',
  foreignField: 'Id',
  as: 'ProductId'
}}, {$addFields: {
  Total_Amount: {
    $multiply: [
      '$UnitPrice',
      '$Quantity'
    ]
  }
}}, {$unwind: {
  path: '$ProductId'
}}, {$group: {
  _id: '$OrderId',
  ProductId: {
    $push: '$ProductId'
  },
  Total: {
    $sum: '$Total_Amount'
  }
}}]
```

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

- Left Sidebar:** Shows the database structure with "Local" selected, displaying 13 DBs and 45 collections. A "FAVORITE" section lists "JOG_Postwork" and "JOG_Postwork.Order".
- Top Bar:** Shows the connection information: "MongoDB Compass - cluster0.yx5qd.mongodb.net:27017/JOG_Postwork.Order".
- Central Area:**
 - Aggregations Tab:** Selected tab. It displays the aggregation pipeline being modified:


```
1: { $unwind: { path: '$ProductId' } }
2: { $group: {
3:   _id: '$OrderId',
4:   ProductId: { $push: '$ProductId' },
5:   Total: { $sum: '$Total_Amount' }
6: } }
```
 - Output Area:** Shows the results of the aggregation pipeline. Two sample documents are shown:


```
_id: ObjectId("5f1f61a219f51f02bc587903")
Id: 1
OrderId: 1
> ProductId: Object
  UnitPrice: 14
  Quantity: 12
  Total_Amount: 168
```

```
_id: ObjectId("5f1f61a219f51f02bc587903")
Id: 2
OrderId: 1
> ProductId: Object
  UnitPrice: 9.8
  Quantity: 10
  Total_Amount: 98
```
- Bottom:** macOS Dock with various application icons.

27.- Desde la tabla OrderItem, agrega los documentos de la vista anterior para mostrar por cada cliente las ordenes que ha pedido y que productos incluye. Cada producto debe tener la información del proveedor.

```
[$lookup: {
  from: 'OrderItem',
  localField: '_id',
  foreignField: 'Id',
  as: 'Id'
}], {$unwind: {
  path: '$Id'
}}, {$addFields: {
  CustomerId: '$Id.CustomerId'
}}, {$project: {
  'Id.CustomerId': 0
}}, {$group: {
  _id: '$CustomerId',
  Ordenes: {
    $push: {
      Order: '$Id',
      Producto: '$ProductId',
      Total: '$Total'
    }
  }
}}
}}
```

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

- Title Bar:** JOG_Postwork.Cliente ... Documents
- Toolbar:** MODIFY SOURCE, Read Only, Options, FIND, REFRESH
- Header:** Documents, Aggregations, Schema, Explain Plan, Indexes
- Content:**
 - JOG_Postwork.Clients con cada orden (view on: JOG_Postwork.Customer)**
 - Document Structure:**
 - Customer:** Contains fields like Id, FirstName, LastName, City, Country, Phone, and Addresses.
 - Order:** Contains fields like Id, OrderDate, TotalAmount, OrderNumber, and Products.
 - Products:** Contains fields like ProductName, SupplierId, ContactName, City, Country, Phone, Fax, UnitPrice, and Package.
 - Expansion:** The 'Ordenes' and 'Producto' fields are expanded to show their respective details.
- Bottom:** View, Options, Find, Refresh buttons.

28.- Usa la agregación anterior para mostrar por ciudad el detalle de cada cliente:

```
[{$lookup: {  
    from: 'Base de datos unificada',  
    localField: 'Id',  
    foreignField: '_id',  
    as: 'Lista'  
}}, {$addFields: {  
    Ordenes: {  
        $arrayElemAt: [  
            '$Lista',  
            0  
        ]  
    }  
}}, {$addFields: {  
    Ordenes: '$Ordenes.Ordenes'  
}}, {$project: {  
    Lista: 0  
}}, {$group: {  
    _id: '$Country',  
    Clientes: {  
        $push: {  
            Id: '$Id',  
            City: '$City',  
            Ordenes: '$Ordenes',  
            Phone: '$Phone'  
        }  
    }  
}}]
```

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

- Left Sidebar:** Shows local databases (13 DBs, 47 collections), hosts (cluster0-shard-00-02.y5qd.mongodb.net, cluster0-shard-00-00.y5qd.mongodb.net, cluster0-shard-00-01.y5qd.mongodb.net), and a connection to "JOG_Postwork".
- Top Bar:** MongoDB Compass - cluster0.y5qd.mongodb.net:27017/JOG_Postwork.Clientes con cada orden
- Central Area:** Pipeline Editor for "JOG_Postwork.Customer".
 - Stages:**
 - \$project:** `1 - {
2 Lista: $
3 }`
 - \$group:** `1 - {
2 _id: '$Country'
3 Clientes: {
4 $push: {
5 _id: '$_id'
6 City: '$City'
7 Ordenes: '$Ordenes'
8 Phone: '$Phone'
9 }
10 }
11 }`
 - Output after \$project stage:** Sample of 20 documents.
 - `_id: ObjectId("5f1fe5f519f51f02bc58756a")
1
FirstName: "Maria"
LastName: "Andres"
City: "Berlin"
Country: "Germany"
Phone: "+38-0874321"
+ Ordenes: Array`
 - `_id: ObjectId("5f1fe5f519f51f02bc58756b")
1
FirstName: "Ana"
LastName: "Trujillo"
City: "Mexico D.F."
Country: "Mexico"
Phone: "(5) 555-4729"
+ Ordenes: Array`
 - `_id: ObjectId("5f1fe5f519f51f02bc58756c")
1
FirstName: "Antonio"
LastName: "Moreno"
City: "Mexico D.F."
Country: "Mexico"
Phone: "(5) 555-3932"
+ Ordenes: Array`
 - Output after \$group stage:** Sample of 20 documents.
 - `_id: "Norway"
+ Clientes: Array`
 - `_id: "Denmark"
+ Clientes: Array`
 - `_id: "Spain"
+ Clientes: Array`
- Bottom:** macOS Dock with various application icons.

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

- Header:** MongoDB Compass, Connect, Edit, View, Collection, Window, Help.
- Toolbar:** Local, JOG_Postwork.Cliente... (selected), Documents.
- Left Sidebar:** Local (13 DBs, 47 Collections), HOSTS (cluster0-shard-00-02.yx5qd...), CLUSTER (Replica Set (alias-wvhbwq...), 3 Nodes), EDITION (MongoDB 4.2.8 Enterprise).
- Search Bar:** Filter your data.
- Collection List:** JOG, JOG_Postwork, Clientes con cada orden, Customer, Order, OrderId con producto y..., OrderItem, Product, Productos con datos de..., Productos por proveedor & Supplier.
- Current Collection:** Clientes con cada orden (view on: JOG_Postwork.Customer).
- Document View:** The document structure is displayed with nested arrays. One object has an array of 13 elements, each containing a city name and a phone number. Another object has an array of 4 elements, each containing an order ID, date, total amount, and a list of order items. Each order item includes a product ID, quantity, and unit price.
- Bottom:** A row of Mac OS X-style icons including Finder, Safari, Google Chrome, Calendar, Notes, Photos, Messages, App Store, System Preferences, Spotify, and others.

29.- Añade a esta misma tabla desde el código de agregación el nombre concatenado del cliente.

```
[$lookup: {  
    from: 'Base de datos unificada',  
    localField: 'Id',  
    foreignField: '_id',  
    as: 'Lista'  
}], {$addFields: {  
    Ordenes: {  
        $arrayElemAt: [  
            '$Lista',  
            0  
        ]  
    }  
}, {$addFields: {  
    Ordenes: '$Ordenes.Ordenes'  
}}, {$project: {  
    Lista: 0  
}}, {$group: {  
    _id: '$Country',  
    Clientes: {  
        $push: {  
            Id: '$Id',  
            City: '$City',  
            Phone: '$Phone',  
            Nombre: {$concat: ["$FirstName", " ", "$LastName"]},  
            Ordenes: '$Ordenes'  
        }  
    }  
}}]
```

MongoDB Compass - cluster0yx5qd.mongodb.net:27017/JOG_Postwork.Clientes con cada orden

JOG_Postwork.Customer

Documents Aggregations Schema Explain Plan Indexes Validation

COLLATION UPDATE VIEW

Modifying pipeline backing "JOG_Postwork.Clientes con cada orden"

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

```
1- {
  2-   Lista: [
  3-     {
    _id: ObjectId("5f1f5ef519f51f02bc58756a")
    Id: 1
    FirstName: "Andres"
    LastName: "Berlin"
    City: "Berlin"
    Country: "Germany"
    Phone: "+49-8974321"
    > Ordenes: Array
  }
  ...
]
```

_id: ObjectId("5f1f5ef519f51f02bc58756b")
 Id: 2
 FirstName: "Maria"
 LastName: "Andres"
 City: "Berlin"
 Country: "Germany"
 Phone: "+30-8874321"
 > Ordenes: Array

_id: ObjectId("5f1f5ef519f51f02bc58756c")
 Id: 3
 FirstName: "Antonio"
 LastName: "Moreno"
 City: "Mexico D.F."
 Country: "Mexico"
 Phone: "(5) 555-4729"
 > Ordenes: Array

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

```
1- {
  2-   $id: '$Country',
  3-   Clientes: [
  4-     {
      _id: '$Id',
      FirstName: '$FirstName',
      LastName: '$LastName',
      City: '$City',
      Phone: '$Phone',
      Nombre: '$concat: ['$FirstName', ' ', '$LastName']',
      Ordenes: '$Ordenes'
    }
  ]
}
```

_id: "Mexico"
 Clientes:
 > 0:Object
 Id: 2
 FirstName: "Ana"
 LastName: "Trujillo"
 City: "Mexico D.F."
 Phone: "(5) 555-4729"
 Nombre: "Ana Trujillo"
 Ordenes: Array

_id: "Spain"
 Clientes:
 > 0:Object

_id: "Switzerland"
 Clientes:

ADD STAGE

MongoDB Compass - cluster0yx5qd.mongodb.net:27017/JOG_Postwork.Clientes con cada orden

JOG_Postwork.Clientes con cada orden (view on: JOG_Postwork.Customer)

MODIFY SOURCE

Read Only

Documents Aggregations Schema Explain Plan Indexes Validation

OPTIONS FIND RESET ...

MAXTIMEMS: 99999
 SKIP: 0
 LIMIT: 0

Displaying documents 1 - 20 of 21

VIEW

Document View

```
_id: "UK"
  Clientes: [
    {
      Id: 1,
      City: "London",
      Phone: "(171) 555-7788",
      Nombre: "Thomas Hardy",
      Ordenes: [
        {
          Id: 1,
          Cliente: {
            _id: ObjectId("5f1f615c19f51f02bc587790"),
            Id: 468,
            OrderDate: "1979-01-01T00:00:00.000Z",
            TotalAmount: 1784,
            OrderNumber: 542837
          },
          Products: [
            {
              Id: 1,
              ProductName: "Ravioli Angelo",
              SupplierId: ObjectId("5f1f5de719f51f02bc587566"),
              Id: 20,
              CompanyName: "Pasta Buttini s.r.l.",
              ContactName: "Giovanni Giudici",
              City: "Siena",
              Country: "Italy",
              Phone: "(089) 6547665",
              Fax: "(089) 6547667",
              UnitPrice: 19.5,
              MinOrder: "24 - 250 g pkgs."
            }
          ],
          IsDiscontinued: false
        },
        {
          Total: 1784
        }
      ]
    }
  ]
}
```

30.- De la agregación actual, indica el número de clientes por país.

```
{  
  $project: {  
    _id: 1,  
    Clientes: {  
      $size: '$Clientes'  
    }  
  }  
}
```

MongoDB Compass - cluster0.yx5qd.mongodb.net:27017/JOG_Postwork.Clientes con cada orden

Modifying pipeline backing "JOG_Postwork.Clientes con cada orden"

Documents Aggregations Schema Explain Plan Indexes Validation DOCUMENTS 91 TOTAL SIZE 12.2KB AVERAGE SIZE 137B INDEXES 2 TOTAL SIZE 40.0KB AVERAGE SIZE 20.0KB SAMPLE MODE AUTO PREVIEW

\$group

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

1+ {
 2+ _id: '\$Country',
 3+ Clientes: {
 4+ \$size: '\$Clientes'
 5+ }
 6+ Id: '\$_id',
 7+ City: '\$City',
 8+ Phone: '\$Phone',
 9+ Nombre: {
 10+ \$each: [
 11+ '\$FirstName',
 12+ '\$LastName'
 13+]
 14+ },
 15+ Ordenes: '\$Ordenes'
 16+ }
 17+ }
 18+ }

1+ {
 2+ _id: "Germany"
 3+ Clientes: Array
 4+ 0: Object
 5+ 1: Object
 6+ 2: Object
 7+ 3: Object
 8+ 4: Object
 9+ 5: Object
 10+ 6: Object
 11+ 7: Object
 12+ 8: Object
 13+ 9: Object
 14+ 10: Object
 15+ 11: Object
 16+ }
 17+ }
 18+ }

1+ {
 2+ _id: "Sweden"
 3+ Clientes: Array
 4+ 0: Object
 5+ 1: Object
 6+ }
 7+ }

1+ {
 2+ _id: "Italy"
 3+ Clientes: Array
 4+ 0: Object
 5+ 1: Object
 6+ }
 7+ }

\$project

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

1+ ***
2+ * specifications: The fields to
3+ * include or exclude.
4+ */
5+ {
 6+ _id: 1,
 7+ Clientes: {\$size: '\$Clientes'}
 8+ }

ADD STAGE