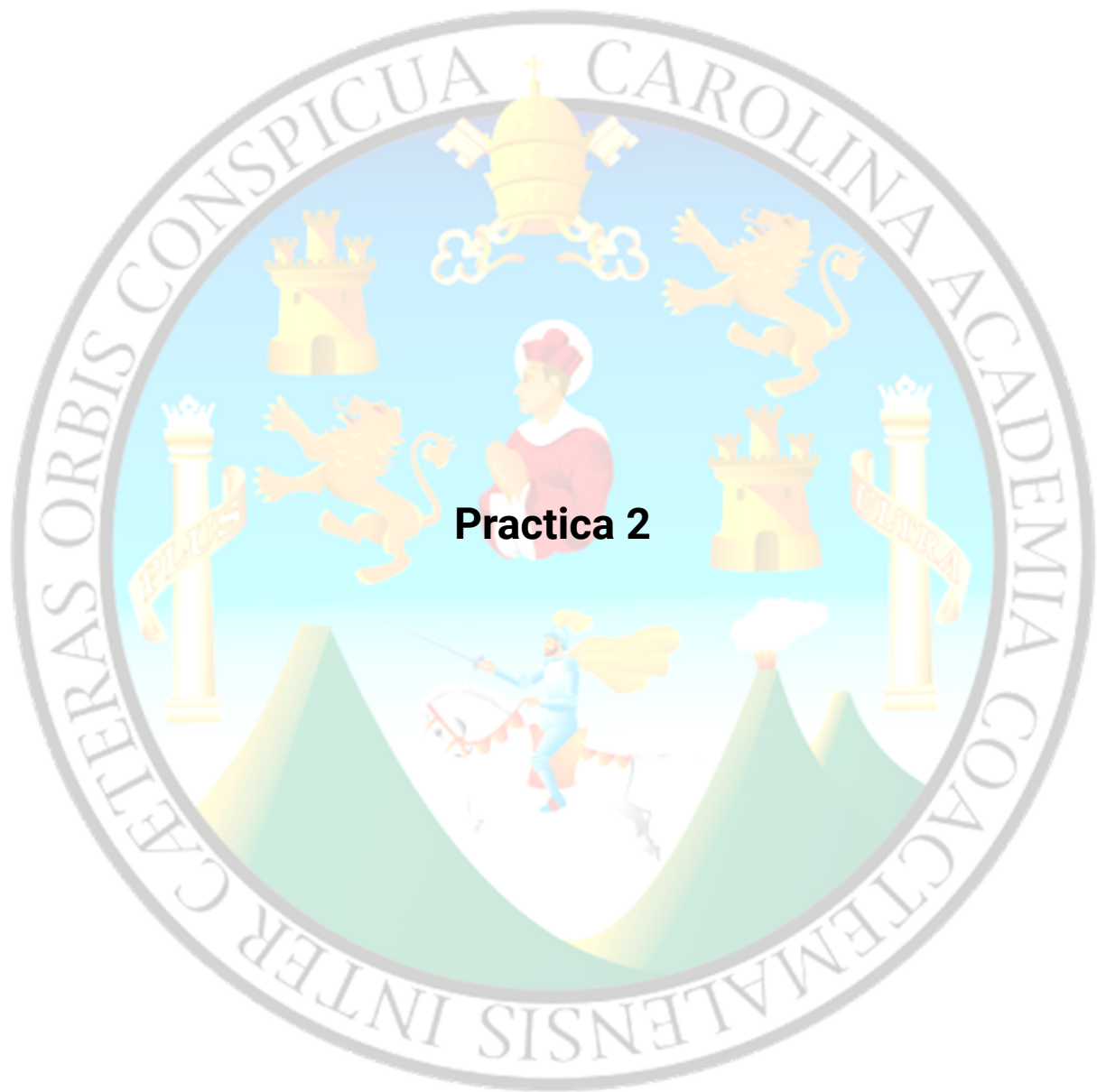


**UNIVERSIDAD DE SAN CARLOS DE GUATEMALA**

**LABORATORIO BASES DE DATOS 2**



## **Practica 2**

### **NOMBRES**

Luis Andres de la Peña Pineda

Angel Oswaldo Arteaga Garcia

Karen Lisbeth Morales Marroquin

### **CARNET:**

201900450

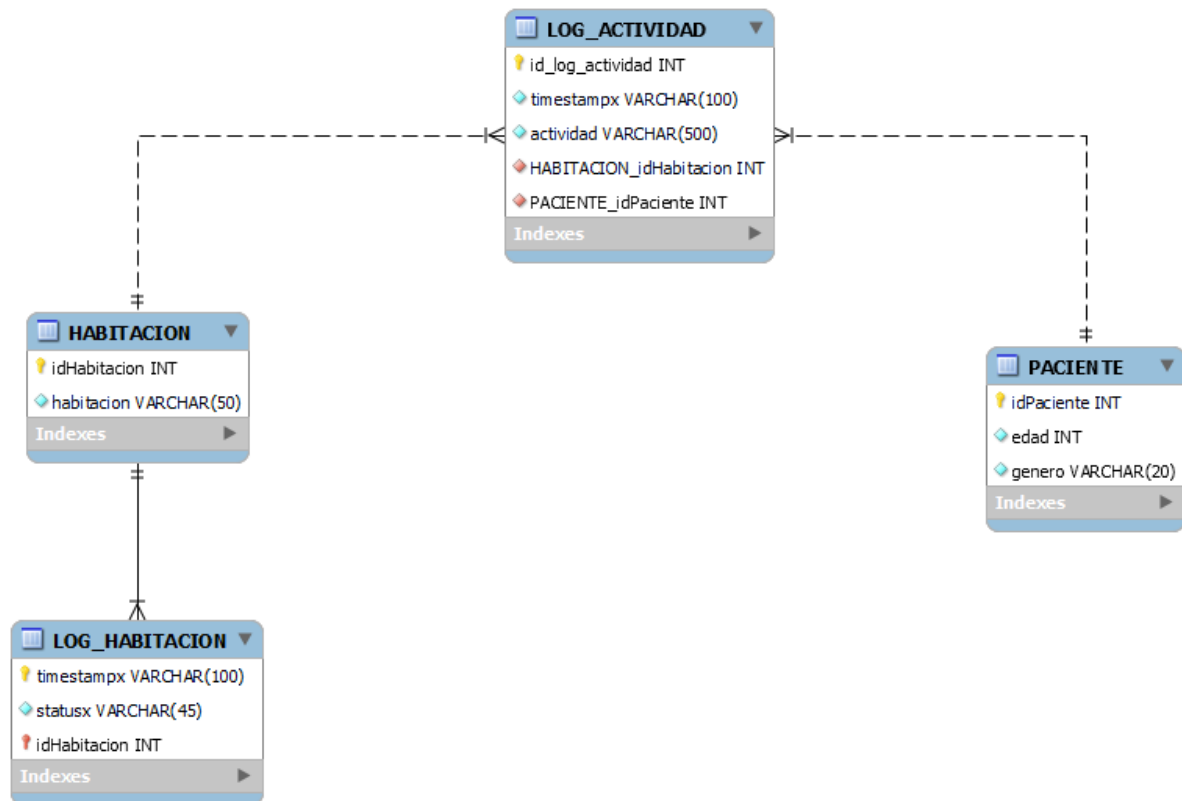
201901816

201908316

## Modelos

### MySQL

Este modelo de base de datos se utiliza para almacenar información sobre habitaciones, pacientes y registros de actividades de un hospital. Las tablas están diseñadas para registrar datos relacionados con la gestión de habitaciones y la interacción de los pacientes con la clínica.



### MongoDB

Se definió un modelo de datos en MongoDB con cuatro colecciones que representan pacientes, habitaciones y registros de actividades en la clínica. Cada colección tiene un esquema de validación que define la estructura y los campos requeridos para los documentos en esa colección. Esto garantiza la consistencia e integridad de los datos almacenados en MongoDB.

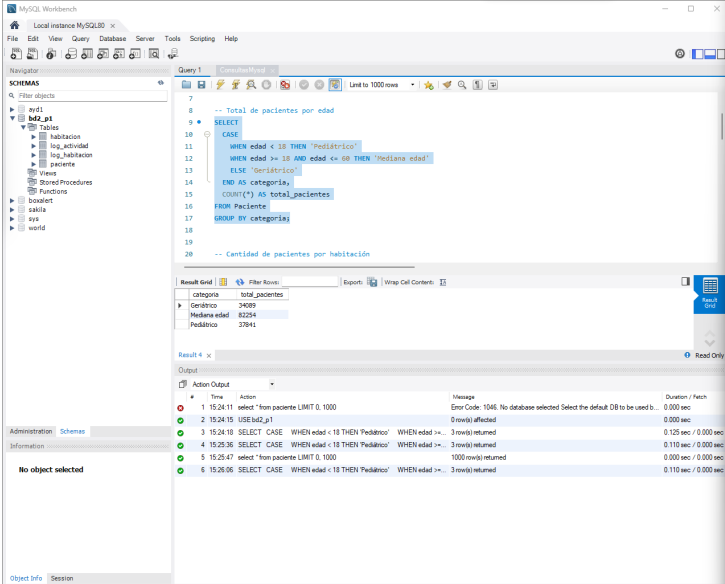


## Capturas

### Consulta 1:

#### 1. MySQL:

La consulta 1 está agrupando pacientes en tres categorías según su edad y luego cuenta cuántos pacientes hay en cada categoría



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

```
-- Total de pacientes por edad
7
8 SELECT
9 CASE
10 WHEN edad < 18 THEN "Pediátrico"
11 WHEN edad >= 18 AND edad <= 65 THEN "Mediana edad"
12 ELSE "Geriatrico"
13 END AS categoria,
14 COUNT(*) AS total_pacientes
15 FROM Paciente
16 GROUP BY categoria;
```

The Results grid shows the following data:

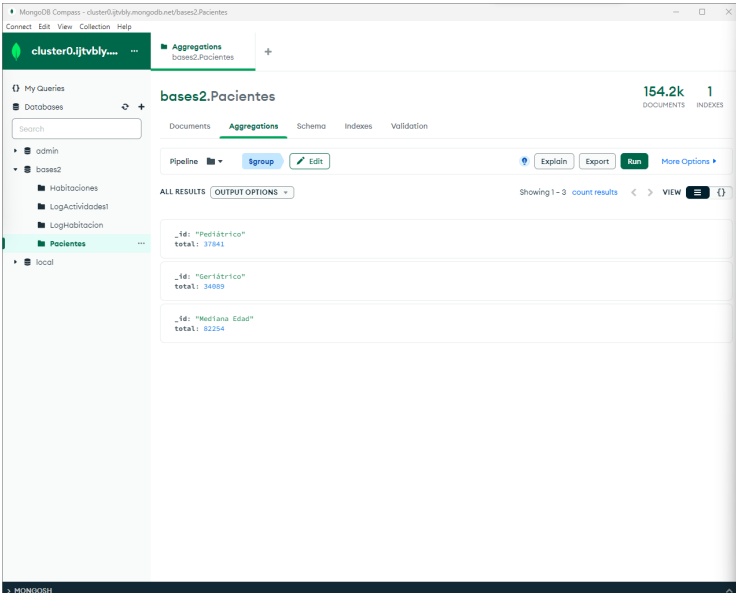
categoria	total_pacientes
Geriatrico	34089
Mediana edad	82244
Pediátrico	37941

The Output tab shows the execution log with the following messages:

```
1 15:24:11 select 'from paciente LIMIT 0, 1000' Error Code: 1046, No database selected Select the default DB to be used b... 0.000 sec
2 15:24:15 USE `bases2`.`p1` 0 rows affected 0.000 sec
3 15:24:18 SELECT CASE WHEN edad < 18 THEN 'Pediátrico' WHEN edad >= 18 THEN 'Mediana edad' WHEN edad >= 66 THEN 'Geriatrico' 3 rows returned 0.125 sec / 0.000 sec
4 15:25:36 SELECT CASE WHEN edad < 18 THEN 'Pediátrico' WHEN edad >= 18 THEN 'Mediana edad' WHEN edad >= 66 THEN 'Geriatrico' 3 rows returned 0.110 sec / 0.000 sec
5 15:25:47 select 'from paciente LIMIT 0, 1000' 1000 rows returned 0.000 sec / 0.000 sec
6 15:26:06 SELECT CASE WHEN edad < 18 THEN 'Pediátrico' WHEN edad >= 18 THEN 'Mediana edad' WHEN edad >= 66 THEN 'Geriatrico' 3 rows returned 0.110 sec / 0.000 sec
```

#### 2. MongoDB:

La consulta 1 agrupa a los pacientes en tres categorías según su edad utilizando una expresión condicional. Luego, cuenta cuántos pacientes hay en cada categoría y muestra el total



The screenshot shows the MongoDB Compass interface. The Aggregations tab is active, displaying the following pipeline:

```
[{"_id": "Pediátrico", "total": 37941}, {"_id": "Geriatrico", "total": 34089}, {"_id": "Mediana Edad", "total": 82244}]
```

The Results tab shows the following data:

_id	total
"Pediátrico"	37941
"Geriatrico"	34089
"Mediana Edad"	82244

La consulta 2 utiliza la operación \$lookup para combinar datos de dos colecciones: 'Habitaciones' y 'LogActividades1', relacionándolos por el campo 'idHabitacion'. Luego, utiliza \$project para proyectar el nombre de la habitación y calcular el tamaño del arreglo 'actividades' en cada documento, lo que representa la cantidad de

pacientes en cada habitación.

MongoDB Compass - cluster0.jhvjby.mongodb.net/bases2-Habitaciones

Connect Edit View Collection Help

cluster0.jhvjby....

Aggregations

bases2.Habitaciones

My Queries

Databases

Search

admin

bases2

Habitaciones

LogActividades1

LogHabitacion

Pacientes

local

bases2.Habitaciones

15 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Pipeline

Stoolup Sproject Edit

Explain Export Run More Options

ALL RESULTS OUTPUT OPTIONS

Showing 1 - 15 count results

VIEW

\_id: ObjectId('65135ed181d11a38663b128d')

habitacion: "Sala de exámenes 1"

pacientes\_en\_habitacion: 3778

\_id: ObjectId('65135ed181d11a38663b128e')

habitacion: "Sala de exámenes 2"

pacientes\_en\_habitacion: 2968

\_id: ObjectId('65135ed181d11a38663b128f')

habitacion: "Sala de exámenes 3"

pacientes\_en\_habitacion: 2139

\_id: ObjectId('65135ed181d11a38663b1228')

habitacion: "Sala de exámenes 4"

pacientes\_en\_habitacion: 1387

\_id: ObjectId('65135ed181d11a38663b1221')

habitacion: "Sala de imágenes 1"

pacientes\_en\_habitacion: 1786

\_id: ObjectId('65135ed181d11a38663b1222')

habitacion: "Sala de procedimientos 1"

pacientes\_en\_habitacion: 2154

\_id: ObjectId('65135ed181d11a38663b1223')

habitacion: "Sala de procedimientos 2"

pacientes\_en\_habitacion: 1816

MONODISH

Reloj

00:00:00.34

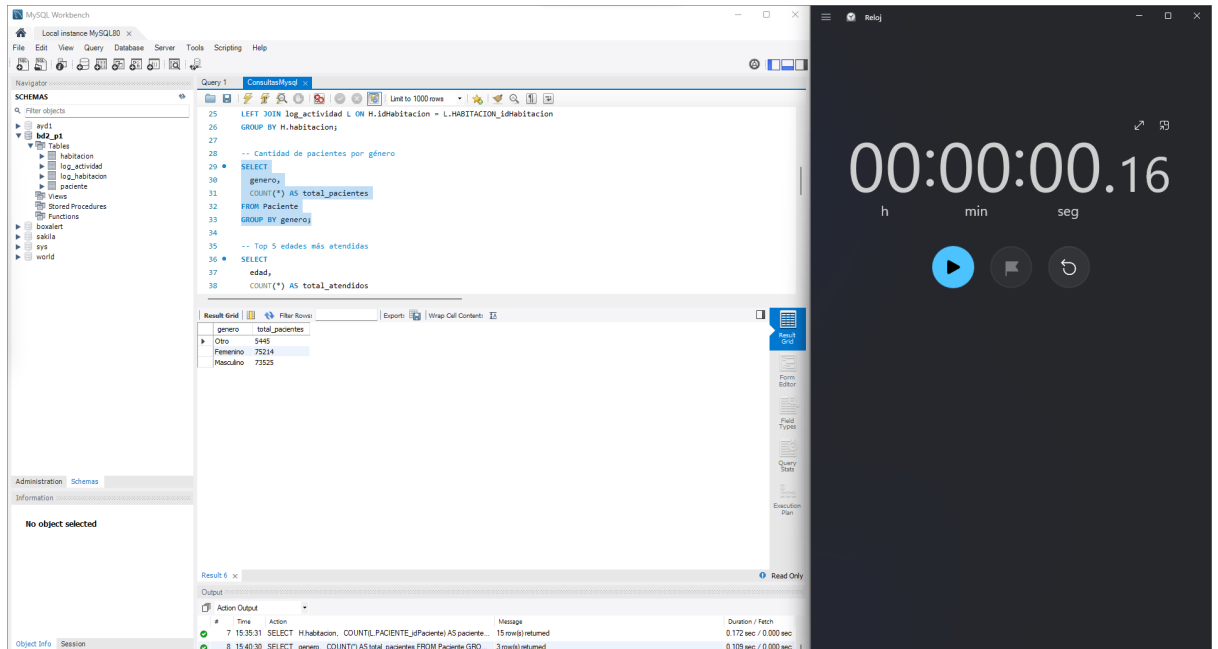
h min seg

Play Stop Refresh

### Consulta 3:

La consulta 3 cuenta el número total de pacientes agrupados por género en la tabla "Paciente" y muestra el resultado para cada género.

#### 1. MySQL:



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

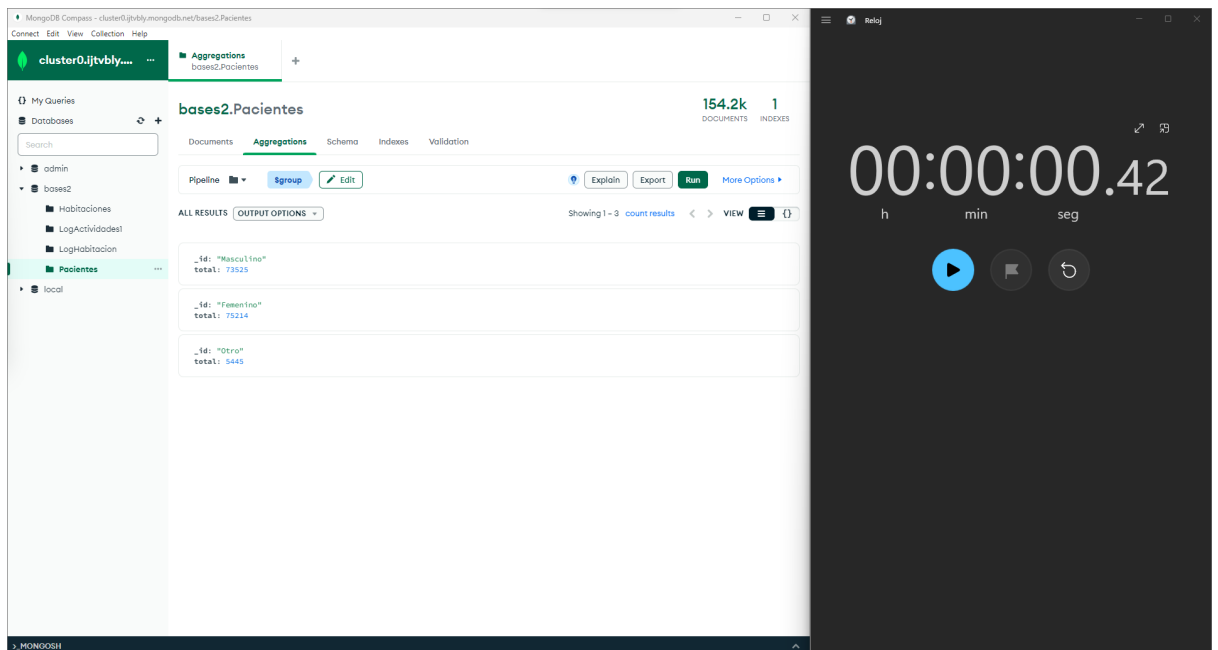
```
25 LEFT JOIN log_actividad L ON H.IdHabitacion = L.HABITACION_IdHabitacion
26 GROUP BY H.habitacion;
27
28 -- Cantidad de pacientes por género
29 SELECT
30     genero,
31     COUNT(*) AS total_pacientes
32 FROM Paciente
33 GROUP BY genero;
34
35 -- Top 5 edades más atendidas
36 SELECT
37     edad,
38     COUNT(*) AS total_atendidos
```

The results window shows the following data:

genero	total_pacientes
Otro	5445
Femenino	75214
Masculino	73525

On the right, a digital clock shows 00:00:00.16.

#### 2. MongoDB:



The screenshot shows the MongoDB Compass interface. The aggregation pipeline is defined as follows:

```
{
  "$group": {
    "_id": "$genero",
    "total": { "$sum": "$count" }
  }
}
```

The results window shows the following data:

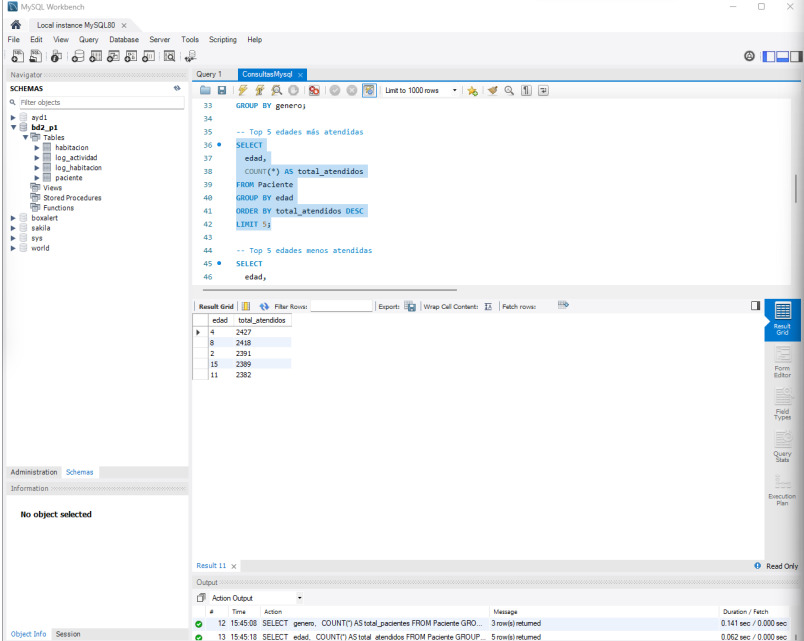
_id	total
"Masculino"	73525
"Femenino"	75214
"Otro"	5445

On the right, a digital clock shows 00:00:00.42.

## Consulta 4:

La consulta 4 cuenta cuántos pacientes han sido atendidos para cada edad en la tabla "Paciente" y muestra los cinco grupos de edad más atendidos en orden descendente según la cantidad total de pacientes atendidos.

### 1. MySQL:



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

```
GROUP BY genero;

-- Top 5 edades más atendidas
SELECT
  edad,
  COUNT(*) AS total_atendidos
FROM Paciente
GROUP BY edad
ORDER BY total_atendidos DESC
LIMIT 5;

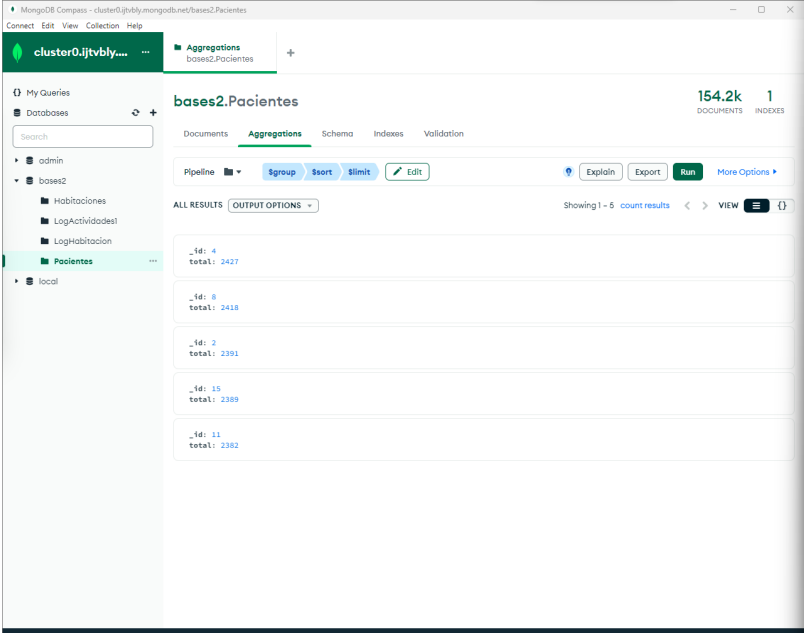
-- Top 5 edades menos atendidas
SELECT
  edad,
```

The results grid displays the following data:

edad	total_atendidos
4	2427
8	2418
2	2391
15	2389
11	2382

The output pane shows the execution of the query, indicating that 3 rows were returned for the first query and 5 rows for the second query.

### 2. MongoDB:



The screenshot shows the MongoDB Compass interface. The aggregation pipeline is defined as follows:

```
{
  "$group": {
    "_id": "$edad",
    "total": { "$sum": "$total_atendidos" }
  }
}
```

The results pane displays the following data:

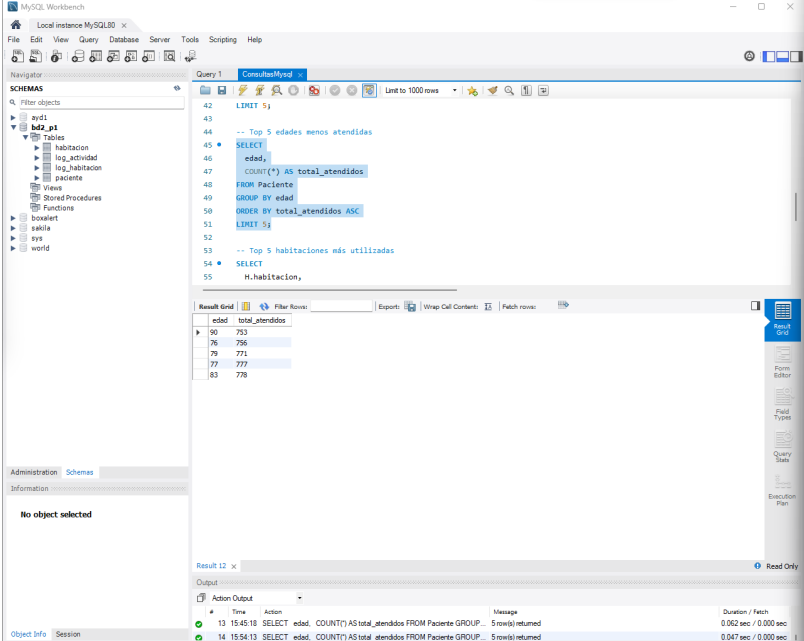
_id	total
4	2427
8	2418
2	2391
15	2389
11	2382

The interface also shows the document count as 154.2k and 1 index.

## Consulta 5:

La consulta 5 cuenta cuántos pacientes han sido atendidos para cada edad en la tabla "Paciente" y muestra los cinco grupos de edad más atendidos en orden ascendente según la cantidad total de pacientes atendidos.

### 1. MySQL:



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

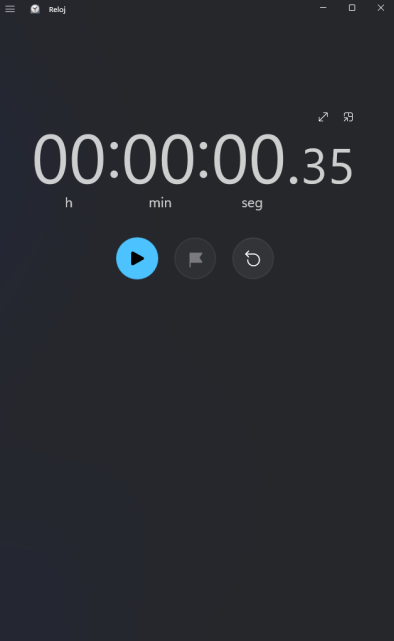
```
42 LIMIT 5;
43
44 -- Top 5 edades menos atendidas
45 -- SELECT
46 --     edad,
47 --     COUNT(*) AS total_atendidos
48 -- FROM Paciente
49 -- GROUP BY edad
50 -- ORDER BY total_atendidos ASC
51 LIMIT 5;
52
53 -- Top 5 habitaciones más utilizadas
54 -- SELECT
55 --     H.habitacion,
```

The Results grid shows the following data:

edad	total_atendidos
90	753
76	756
79	771
77	777
83	778

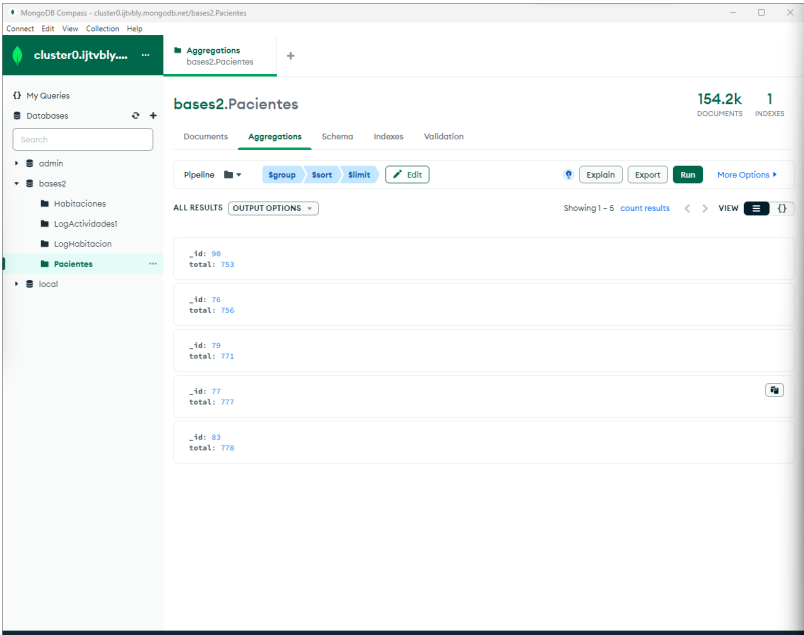
The Output tab shows the execution time and message:

#	Time	Action	Message	Duration / Fetch
13	15.45.18	SELECT	edad, COUNT(*) AS total_atendidos FROM Paciente GROUP BY edad ORDER BY total_atendidos ASC LIMIT 5; 5 row(s) returned	0.062 sec / 0.000 sec
14	15.54.13	SELECT	edad, COUNT(*) AS total_atendidos FROM Paciente GROUP BY edad ORDER BY total_atendidos ASC LIMIT 5; 5 row(s) returned	0.047 sec / 0.000 sec



The digital timer shows a time of 00:00:00.35, with units h, min, and seg. It includes a play button and a reset button.

### 2. MongoDB:

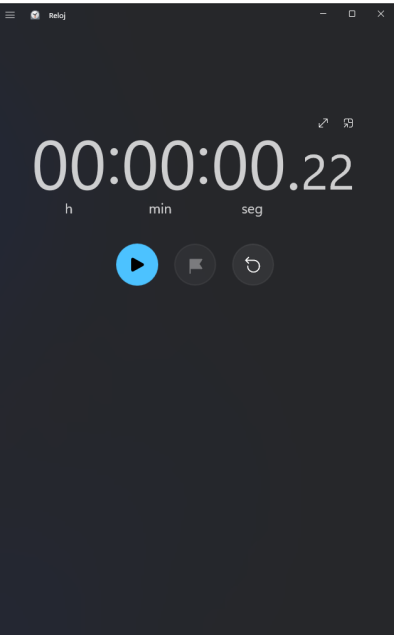


The screenshot shows the MongoDB Compass interface. The aggregation pipeline is defined as follows:

```
{
  "$group": {
    "_id": "$edad",
    "total": { "$sum": "$total_atendidos" }
  }
}
```

The Results tab shows the following data:

_id	total
90	753
76	756
79	771
77	777
83	778



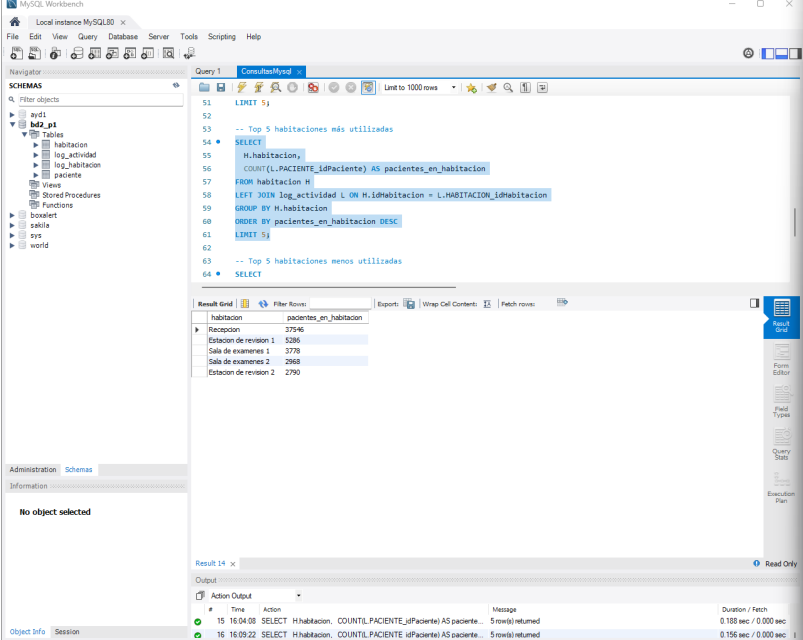
The digital timer shows a time of 00:00:00.22, with units h, min, and seg. It includes a play button and a reset button.



## Consulta 6:

### 1. MySQL:

La consulta 6 realiza una operación de JOIN izquierdo entre las tablas 'Habitacion' y 'LogActividad' utilizando el campo 'idHabitacion' como clave de relación. Luego, agrupa los resultados por 'habitacion' y cuenta cuántos pacientes hay en cada habitación. Finalmente, ordena los resultados en orden descendente según la cantidad de pacientes y limita la salida a los 5 primeros resultados.



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

```
51 LIMIT 5)
52
53 -- Top 5 habitaciones más utilizadas
54 SELECT
55     H.habitacion,
56     COUNT(L.PACIENTE_idPaciente) AS pacientes_en_habitacion
57 FROM habitacion H
58 LEFT JOIN Log_actividad L ON H.idHabitacion = L.HABITACION_idHabitacion
59 GROUP BY H.habitacion
60 ORDER BY pacientes_en_habitacion DESC
61 LIMIT 5)
62
63 -- Top 5 habitaciones menos utilizadas
64 SELECT
```

The Results Grid shows the following data:

habitacion	pacientes_en_habitacion
Recepcion	27546
Estacion de revision 1	5286
Sala de exámenes 1	3778
Sala de exámenes 2	2968
Estacion de revision 2	2790

The Output tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
15	16:04:08	SELECT	H.habitacion, COUNT(L.PACIENTE_idPaciente) AS paciente...	0.188 sec / 0.000 sec
16	16:09:22	SELECT	H.habitacion, COUNT(L.PACIENTE_idPaciente) AS paciente...	0.156 sec / 0.000 sec

### 2. MongoDB:

La consulta 6 agrupa los documentos por el campo 'idHabitacion' y cuenta cuántos documentos hay en cada grupo utilizando \$sum: 1. Luego, ordena los resultados en orden descendente según la cantidad total y limita la salida a los 5 primeros resultados.

MongoDB Compass - cluster0.jhby.mongodb.net/bases2.Habitaciones

Connect Edit View Collection Help

cluster0.jhby... Aggregations bases2.LogActivi... Aggregations bases2.Habitaciones

My Queries Databases Search

admin bases2 Habitaciones LogActividades LogHabitacion Pacientes local

bases2.Habitaciones 15 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Pipeline \$lookup \$project \$sort \$limit Explain Export Run More Options

ALL RESULTS OUTPUT OPTIONS Showing 1 - 5 count results VIEW

```
{
  "_id": ObjectId("65135ed181d11a38663b1216"),
  "habitacion": "Recepcion",
  "pacientes_en_habitacion": 37546
},
{
  "_id": ObjectId("65135ed181d11a38663b1218"),
  "habitacion": "Estación de revisión 2ª",
  "pacientes_en_habitacion": 5286
},
{
  "_id": ObjectId("65135ed181d11a38663b128d"),
  "habitacion": "Sala de exámenes 1ª",
  "pacientes_en_habitacion": 3778
},
{
  "_id": ObjectId("65135ed181d11a38663b128e"),
  "habitacion": "Sala de exámenes 2ª",
  "pacientes_en_habitacion": 2968
},
{
  "_id": ObjectId("65135ed181d11a38663b1219"),
  "habitacion": "Estación de revisión 2ª",
  "pacientes_en_habitacion": 2790
}
```

Reloj

00:00:00.43

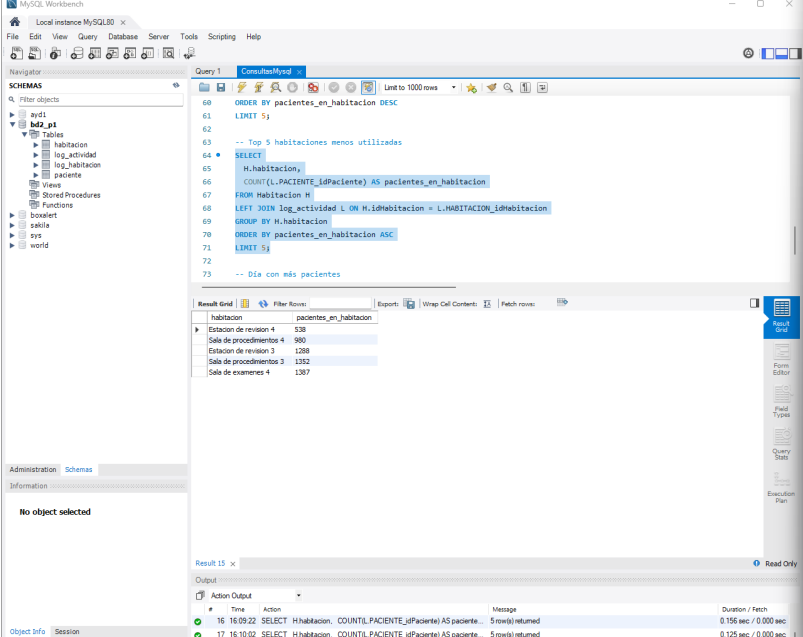
h min seg

Play Stop Refresh

## Consulta 7:

### 1. MySQL:

La consulta 6 realiza una operación de JOIN izquierdo entre las tablas 'Habitacion' y 'LogActividad' utilizando el campo 'idHabitacion' como clave de relación. Luego, agrupa los resultados por 'habitacion' y cuenta cuántos pacientes hay en cada habitación. Finalmente, ordena los resultados en orden ascendente según la cantidad de pacientes y limita la salida a los 5 primeros resultados.



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

```
60 ORDER BY pacientes_en_habitacion DESC
61 LIMIT 5;
62
63 -- Top 5 habitaciones menos utilizadas
64 SELECT
65     H.habitacion,
66     COUNT(L.PACIENTE_idPaciente) AS pacientes_en_habitacion
67 FROM Habitacion H
68 LEFT JOIN Log_actividad L ON H.idHabitacion = L.HABITACION_idHabitacion
69 GROUP BY H.habitacion
70 ORDER BY pacientes_en_habitacion ASC
71 LIMIT 5;
72
73 -- Día con más pacientes
```

The Results Grid shows the following data:

habitacion	pacientes_en_habitacion
Estacion de revision 4	138
Sala de procedimientos 4	900
Estacion de revision 3	1288
Sala de procedimientos 3	1352
Sala de exámenes 4	1387

The Output tab shows the execution log:

```
16:16:09.22 SELECT H.habitacion, COUNT(L.PACIENTE_idPaciente) AS paciente... 5 row(s) returned 0.156 sec / 0.000 sec
17:16:10.02 SELECT H.habitacion, COUNT(L.PACIENTE_idPaciente) AS paciente... 5 row(s) returned 0.125 sec / 0.000 sec
```

### 2. MongoDB:

La consulta 7 agrupa los documentos por el campo 'idHabitacion' y cuenta cuántos documentos hay en cada grupo utilizando \$sum: 1. Luego, ordena los resultados en orden ascendente según la cantidad total y limita la salida a los 5 primeros resultados.

MongoDB Compass - cluster0.jhvjby.mongodb.net/bases2.Habitaciones

Connect Edit View Collection Help

cluster0.jhvjby...

Aggregations bases2.LogActivi... Aggregations bases2.Habitacio... Aggregations bases2.Habitacio...

My Queries

Databases

Search

admin

bases2

Habitaciones

LogActividades

LogHabitacion

Pacientes

local

bases2.Habitaciones

15 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Indexes Validation

Pipeline \$lookup \$project \$sort \$limit Edit Explain Export Run More Options

ALL RESULTS OUTPUT OPTIONS Showing 1 - 5 count results VIEW

\_id: ObjectId('65135ed181d11a38663b121b')

habitacion: "Estación de revisión 4"

pacientes\_en\_habitacion: 538

\_id: ObjectId('65135ed181d11a38663b1215')

habitacion: "Sala de procedimientos 4"

pacientes\_en\_habitacion: 989

\_id: ObjectId('65135ed181d11a38663b121a')

habitacion: "Estación de revisión 3"

pacientes\_en\_habitacion: 1288

\_id: ObjectId('65135ed181d11a38663b1214')

habitacion: "Sala de procedimientos 3"

pacientes\_en\_habitacion: 1352

\_id: ObjectId('65135ed181d11a38663b1218')

habitacion: "Sala de exámenes 4"

pacientes\_en\_habitacion: 1387

Reloj

00:00:00.60

h min seg

## Consulta 8:

### 1. MySQL:

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

```
6 ORDER BY cantidad_registros DESC
7 LIMIT 1;
8
9 SELECT DATE_FORMAT(STR_TO_DATE(timestamp, '%c/%e/%Y %l:%i:%s'), '%c/%e/%Y %l:%i:%s') AS fecha_convertida, COUNT(*) AS cantidad_registros
10 FROM BD2_P1.log_actividad
11 GROUP BY fecha_convertida
12 ORDER BY cantidad_registros DESC
13 LIMIT 1;
14
```

The Results Grid shows the following data:

totalPacientes	fecha
1489	2021-06-04

The Action Output shows the execution of the query, including the time taken and the number of rows returned.

Query Completed

### 2. MongoDB:

The screenshot shows the MongoDB Compass interface. The aggregation pipeline is defined as follows:

```
{
  "$group": {
    "_id": "totalPacientes",
    "count": { "$sum": 1 }
  }
}
```

The Results section shows the output of the aggregation:

totalPacientes
1489

The interface also displays the number of documents (67.7k) and indexes (1) for the collection.

## **Conclusión y justificación**

La comparación de tiempos entre MySQL y MongoDB en las consultas realizadas es un aspecto crucial para determinar cuál de las dos bases de datos es más adecuada para las necesidades de la clínica médica. Estos resultados proporcionan información valiosa para elegir una base de datos sobre la otra. Analizando los datos de tiempo al ejecutar podemos observar que la mayoría son más rápidas en obtener los resultados en MySQL. podrías concluir que MySQL es más adecuado para las necesidades actuales, pero que MongoDB puede ser una opción valiosa a largo plazo a medida que la clínica crezca y sus necesidades evolucionen.

La elección entre MongoDB y MySQL también depende de la simplicidad o complejidad de los datos y consultas de la aplicación. Si bien MySQL es ideal para relaciones de datos complejas, MongoDB puede ser preferible cuando se trata de datos más simples y consultas rápidas.

Considerando que MySQL mostró un mejor rendimiento en la mayoría de las consultas actuales, parece ser la elección más adecuada para las necesidades inmediatas de la clínica médica. Proporciona respuestas más rápidas en consultas tanto simples como complejas.

MongoDB, aunque puede no ser la elección óptima en este momento, sigue siendo una opción valiosa para el futuro a medida que la clínica crece y sus necesidades evolucionan. Su flexibilidad y escalabilidad pueden ser beneficiosas a medida que se acumulan más datos y se plantean nuevos requisitos.

## **Repositorio**

[https://github.com/anddelap/BD2S22023\\_Grupo\\_1](https://github.com/anddelap/BD2S22023_Grupo_1)