

Introducción a bases de datos

Sesión 4

Adrian González Pardo

20 de mayo de 2021

1. MySQL

1.1. Ejemplos

```
1 # Ejemplo de creacion de BD en caso de existir no lo hace
2 CREATE DATABASE IF NOT EXISTS Nombre;
3
4 # Ejemplo de eliminacion de BD en caso de existir
5 DROP DATABASE IF EXISTS Nombre;
6
7 CREATE DATABASE IF NOT EXISTS MiNombre;
8
9 USE MiNombre;
```

```
1 USE MiNombre;
2 # Creacion de tabla 'users' en caso de no existir
3 CREATE TABLE IF NOT EXISTS users (
4     user_id INT PRIMARY KEY,
5     genero VARCHAR(1),
6     edad INT,
7     ocup INT,
8     cp VARCHAR(20)
9 );
10
11 # En caso de existir o no
12 # DROP TABLE IF EXISTS nombre;
```

```
1
2 # Para cargar los datos desde el terminal de nuestro servidor
3
4
5 SHOW VARIABLES LIKE "local_infile";
6 # En caso de off modificamos el acceso
7 SET GLOBAL local_infile = 'ON';
8 SHOW VARIABLES LIKE "local_infile";
9
10 # En caso de que el INFILE mande un error como
11 # The MySQL server is running with the
12 # --secure-file-priv option so it cannot execute this statement
13 # Ejecutar lo siguiente:
14 LOAD DATA LOCAL INFILE "ml-1m/users.dat"
15 INTO TABLE users
16 FIELDS TERMINATED BY ',';
17 LINES TERMINATED BY '\n';
```

1.2. Retos

```
1 USE MiNombre;
2 # Creacion de tabla "movies" si no existe
3 CREATE TABLE IF NOT EXISTS movies (
4     movie_id INT PRIMARY KEY,
5     titulo VARCHAR(200),
6     genero VARCHAR(200)
7 );
8 # Creacion de tabla "ratings" si no existe
9 CREATE TABLE IF NOT EXISTS ratings (
10     user_id INT,
11     movie_id INT,
12     rating INT,
13     time_stamp BIGINT,
14     FOREIGN KEY(user_id) REFERENCES users(user_id),
15     FOREIGN KEY(movie_id) REFERENCES movies(movie_id)
16 );
17
18 SELECT * FROM users LIMIT 10;
19
20 SELECT * FROM movies LIMIT 10;
21
22 SELECT * FROM ratings LIMIT 10;
```



```
1 USE MiNombre;
2 # Carga de datos en "movies" del archivo "movies.dat"
3 LOAD DATA LOCAL INFILE "ml-1m/movies.dat"
4 INTO TABLE movies
5 FIELDS TERMINATED BY ','
6 LINES TERMINATED BY '\n';
7
8 # Carga de datos en "ratings" del archivo "ratings.dat"
9 LOAD DATA LOCAL INFILE "ml-1m/ratings.dat"
10 INTO TABLE ratings
11 FIELDS TERMINATED BY ','
12 LINES TERMINATED BY '\n';
```

1.3. Capturas de ejecución en servidor

```
mysql> CREATE TABLE IF NOT EXISTS users (
->   user_id INT PRIMARY KEY,
->   genero VARCHAR(1),
->   edad INT,
->   ocup INT,
->   cp VARCHAR(20)
-> );
Query OK, 0 rows affected (0.07 sec)

mysql> CREATE TABLE IF NOT EXISTS movies (
->   movie_id INT PRIMARY KEY,
->   titulo VARCHAR(200),
->   genero VARCHAR(200)
-> );
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> CREATE TABLE IF NOT EXISTS ratings (
->   user_id INT,
->   movie_id INT,
->   rating INT,
->   time_stamp BIGINT,
->   FOREIGN KEY(user_id) REFERENCES users(user_id),
->   FOREIGN KEY(movie_id) REFERENCES movies(movie_id)
-> );
Query OK, 0 rows affected (0.06 sec)

mysql> LOAD DATA LOCAL INFILE './ml-lm/users.dat'
-> INTO TABLE users
-> FIELDS TERMINATED BY ','
-> LINES TERMINATED BY '\n';
ERROR 1290 (HY000): The MySQL server is running with the --secure-file-priv option so it cannot execute this statement
```

```
Database changed
mysql> LOAD DATA LOCAL INFILE 'ml-lm/users.dat'
-> INTO TABLE users
-> FIELDS TERMINATED BY ','
-> LINES TERMINATED BY '\n';
Query OK, 6040 rows affected (0.51 sec)
Records: 6040 Deleted: 0 Skipped: 0 Warnings: 0

mysql> show tables;
+-----+
| Tables in MNombre |
+-----+
| movies             |
| ratings            |
| users              |
+-----+
1 rows in set (0.00 sec)

mysql> LOAD DATA LOCAL INFILE 'ml-lm/movies.dat'
-> INTO TABLE movies
-> FIELDS TERMINATED BY ','
-> LINES TERMINATED BY '\n';
Query OK, 3883 rows affected, 1 warning (0.40 sec)
Records: 3883 Deleted: 0 Skipped: 0 Warnings: 1

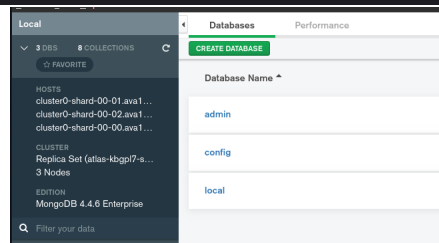
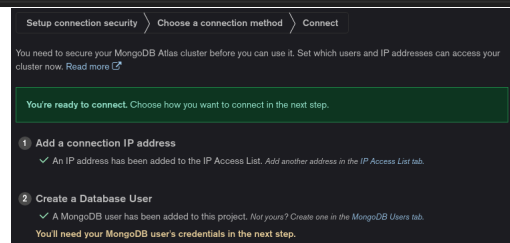
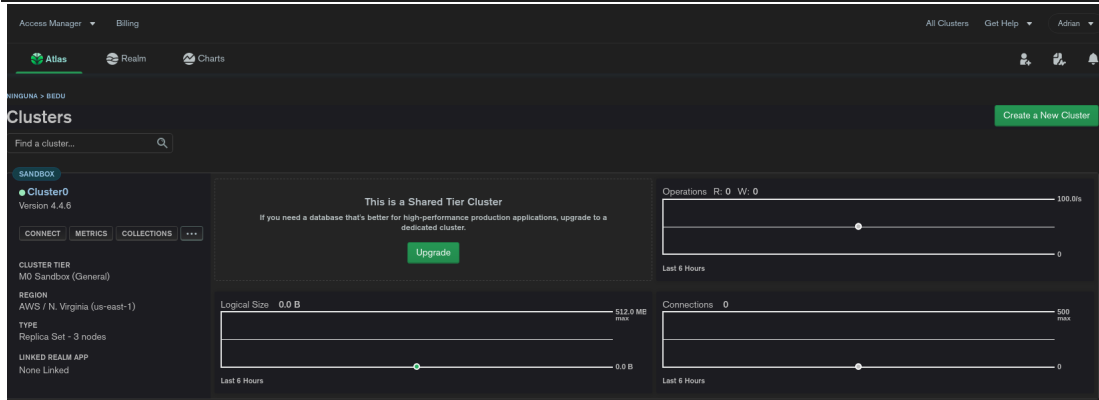
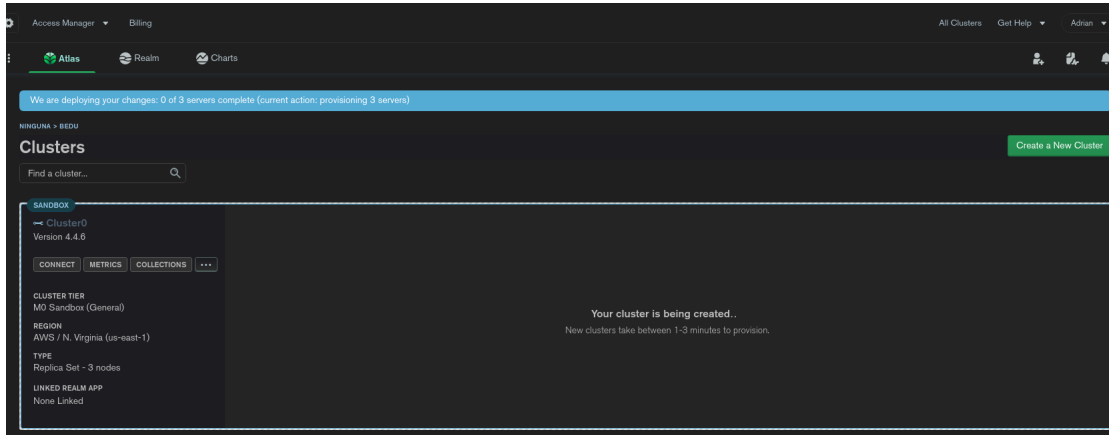
mysql> LOAD DATA LOCAL INFILE 'ml-lm/ratings.dat'
-> INTO TABLE ratings
-> FIELDS TERMINATED BY ','
-> LINES TERMINATED BY '\n';
Query OK, 1000209 rows affected (41.32 sec)
Records: 1000209 Deleted: 0 Skipped: 0 Warnings: 0
```

```
mysql> SELECT * FROM users LIMIT 10;
+-----+
| user_id | genero | edad | ocup | cp |
+-----+
| 1 | F | 1 | 10 | 48007 |
| 2 | M | 56 | 16 | 70072 |
| 3 | M | 25 | 15 | 55117 |
| 4 | M | 45 | 7 | 02460 |
| 5 | M | 25 | 20 | 55455 |
| 6 | F | 50 | 9 | 55117 |
| 7 | M | 35 | 1 | 06810 |
| 8 | M | 25 | 12 | 11412 |
| 9 | M | 25 | 17 | 61614 |
| 10 | F | 35 | 1 | 95370 |
+-----+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM movies LIMIT 10;
+-----+
| movie_id | titulo | genero |
+-----+
| 1 | Toy Story (1995) | Animation|Children's|Comedy |
| 2 | Jumanji (1995) | Adventure|Children's|Fantasy |
| 3 | Grumpier Old Men (1995) | Comedy|Romance |
| 4 | Waiting to Exhale (1995) | Comedy|Drama |
| 5 | Father of the Bride Part II (1995) | Comedy |
| 6 | Heat (1995) | Action|Crime|Thriller |
| 7 | Sabrina (1995) | Comedy|Romance |
| 8 | Tom and Huck (1995) | Adventure|Children's |
| 9 | Sudden Death (1995) | Action |
| 10 | GoldenEye (1995) | Action|Adventure|Thriller |
+-----+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM ratings LIMIT 10;
+-----+
| user_id | movie_id | rating | time_stamp |
+-----+
| 1 | 1193 | 5 | 978300760 |
| 1 | 661 | 3 | 978302109 |
| 1 | 914 | 3 | 978301968 |
| 1 | 3408 | 4 | 978300275 |
| 1 | 2355 | 5 | 978824291 |
| 1 | 1197 | 3 | 978302268 |
| 1 | 1207 | 5 | 978302039 |
| 1 | 2804 | 5 | 978300719 |
| 1 | 594 | 4 | 978302268 |
| 1 | 910 | 4 | 978301368 |
+-----+
10 rows in set (0.01 sec)
```

2. MongoDB



Usuario y contraseña: `introabd:introabd1234`
Más la creación del servidor de MongoDB

Import To Collection MiNombre.users

Select File
/run/media/g3vc4ck/externData/materias_ultimo/curso/bedu/session4/csv/ [BROWSE](#)

Select Input File Type
JSON CSV

Options
Select delimiter COMMA
☒ Ignore empty strings
☐ Stop on errors

Specify Fields and Types

	id	gen	edad	ocup	cp
	string	string	string	string	string
1	1	F	1	18	48867
2	2	M	56	16	78872
3	3	M	25	15	55117
4	4	M	45	7	92468
5	5	M	25	29	55455
6	6	F	58	9	55117
7	7	M	35	1	96818
8	8	M	25	12	11413
9	9	M	25	17	61614
10	10	F	35	1	95378

MiNombre.users

Documents Aggregations Schema

Filter { field: 'value' }

ADD DATA VIEW

```

{
  "_id": "objectId('65a0fdae6e0c0368498432')",
  "id": 1,
  "gen": "F",
  "edad": 1,
  "ocup": 18,
  "cp": 48867
}

```

```

{
  "_id": "objectId('65a0fdae6e0c0368498432')",
  "id": 2,
  "gen": "M",
  "edad": 56,
  "ocup": 16,
  "cp": 78872
}

```

Creación de colección users

Collection Name ^	Documents	Avg. Document Size
movies	3,883	87.0 B
ratings	0	-
users	6,040	69.0 B

Import To Collection MiNombre.ratings

Select File
/run/media/g3vc4ck/externData/materias_ultimo/curso/bedu/session4/csv/ [BROWSE](#)

Select Input File Type
JSON CSV

Options
Select delimiter COMMA
☒ Ignore empty strings
☐ Stop on errors

Specify Fields and Types

	u_id	r_id	edad	ocupacion
	Number	Number	Number	Number
1	1	1193	5	978398768
2	1	661	3	9783982109
3	1	914	3	9783981968
4	1	3408	4	9783980275
5	1	2355	5	978824291
6	1	1197	3	9783982268
7	1	1287	5	9783982939
8	1	2884	5	9783980719
9	1	594	4	9783982268
10	1	919	4	9783981368

Importing documents... Stop 48,000 / ~195,269

Collection Name ^	Documents	Avg. Document Size
movies	3,883	87.0 B
ratings	1,000,209	69.0 B
users	6,040	69.0 B

Creación de colección movies y ratings con sus archivos csv respectivo



Uso de la herramienta de MongoDB Compass para añadir, modificar, buscar información dentro de nuestra coleccion users

Import To Collection MNombre.movies

Select File

/run/media/d3vc4ck/externData/materias_utlino/curso/bedu/sesion4/cvsv

BROWSE

Select Input File Type

JSON

CSV

Options

Select delimiter

COMMA

☒ Ignore empty strings

☐ Stop on errors

Specify Fields and Types

	id	titulo	gen
	Number	String	String
1	4000	Avengers: Endgame (2019)	Fantasy Sci-Fi
2	4001	Glass (2019)	Drama Fantasy

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

ADD DATA

VIEW

```

{
  "_id": "ObjectId('60a70eeef6e6cd3080913c9')",
  "id": 4000,
  "titulo": "Avengers: Endgame (2019)",
  "gen": "Fantasy|Sci-Fi"
}

{
  "_id": "ObjectId('60a70eeef6e6cd3080913c1')",
  "id": 4001,
  "titulo": "Glass (2019)",
  "gen": "Drama|Fantasy"
}

```

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

ADD DATA

VIEW

```

{
  "_id": "ObjectId('60a70eeef6e6cd3080913c0')",
  "id": 4000,
  "titulo": "Avengers: Endgame (2019)",
  "gen": "Fantasy|Sci-Fi"
}

{
  "_id": "ObjectId('60a70eeef6e6cd3080913c1')",
  "id": 4001,
  "titulo": "Glass (2019)",
  "gen": "Drama|Fantasy",
  "valoraciones": Array
  ~ 0: Object
    ~ user: 1563
    ~ movie: 4001
    ~ rating: 4
  ~ 1: Object
    ~ user: 424
    ~ movie: 4001
    ~ rating: 5
}

```

Ejercicio de MongoDB en la colección movies

7