PRACTICA 3 INI

3.2 CLONADO DE UN REPOSITORIO REMOTO

1.-mkdir practicas

2.-cd practicas

3.-git clone https://github.com/tomasmachin/iii-tmachin.git

4.-ls

5.-cd iii-tmachin

3.3 CREACIÓN Y MODIFICACIÓN DE FICHEROS

1.-vim HolaMundo.java

i

public class HolaMundo{ aaapublic static void main(String[] args){ System.out.println("Hola mundo"); } }

esc

:wq

2.-git status (el estado del fichero no esta añadido)

3.-git add *

4.-git status

5.-git commit -am “Primera version de HolaMundo”

6.-vim instrucciones.txt

i

Instrucciones para la ejecución de HolaMundo.

esc

:wq

7.-git add *

8.-git commit -am “Instrucciones para ejecutarHolaMundo”

9.-rm instrucciones.txt

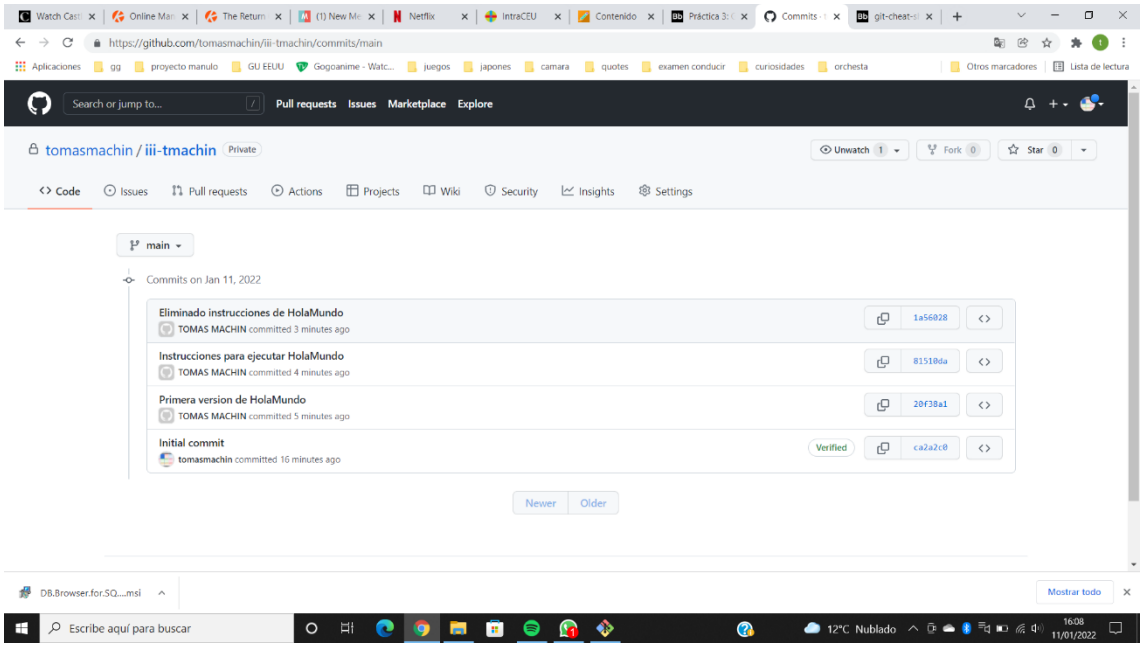
10.-git status

11.-git commit -am “Eliminado instrucciones de HolaMundo”

3.4 ACTUALIZACION DEL REPOSITORIO REMOTO

1.- git push

2.-



3.5 RESOLVIENDO CONFLICTOS EN FICHEROS

1.- cd ..

2.- mkdir otroalumno

cd otroalumno

3.- git clone https://github.com/tomasmachin/iii-tmachin.git

4.- cd iii-tmachin

ls

5.-vim HolaMundo.java

i

public class HolaMundo{

public static void main(String[] args){ System.out.print("Nuevo Hola mundo"); } }

esc

:wq

6.-git add *

7.- git commit -am “Nueva forma de imprimir HolaMundo”

8.-git push

10.- cd ../..

11.-cd iii-tmachin

12.- vim HolaMundo.java

i

```
public class HolaMundo{

public static void main(String[] args){

  System.out.println("Otro Hola mundo");

  }

}
```

esc

:wq

13.- git add *

14.- git commit -a “Otro HolaMundo”

15.- git push (no deja actualizarse ya que hay información que choca y el repositorio no sabe que hacer)

16.- git pull (indica que no ha podido hacerde debido a que no puede mezclar, informacion de HolaMundo.java)

```
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 389 bytes | 12.00 KiB/s, done.
From https://github.com/tomasmachin/iii-tmachin
   1a56028..9282806  main       -> origin/main
Auto-merging HolaMundo.java
CONFLICT (content): Merge conflict in HolaMundo.java
Automatic merge failed; fix conflicts and then commit the result.
```

17.-git status (me aparece que no tengo mezclado cierto archivo y me dice alguna forma para corregirlo)

18.-vim HolaMundo.java

i

```
public class HolaMundo{

  public static void main(String[] args){

    System.out.println("Nuevo Hola mundo");

  }

}
```

esc

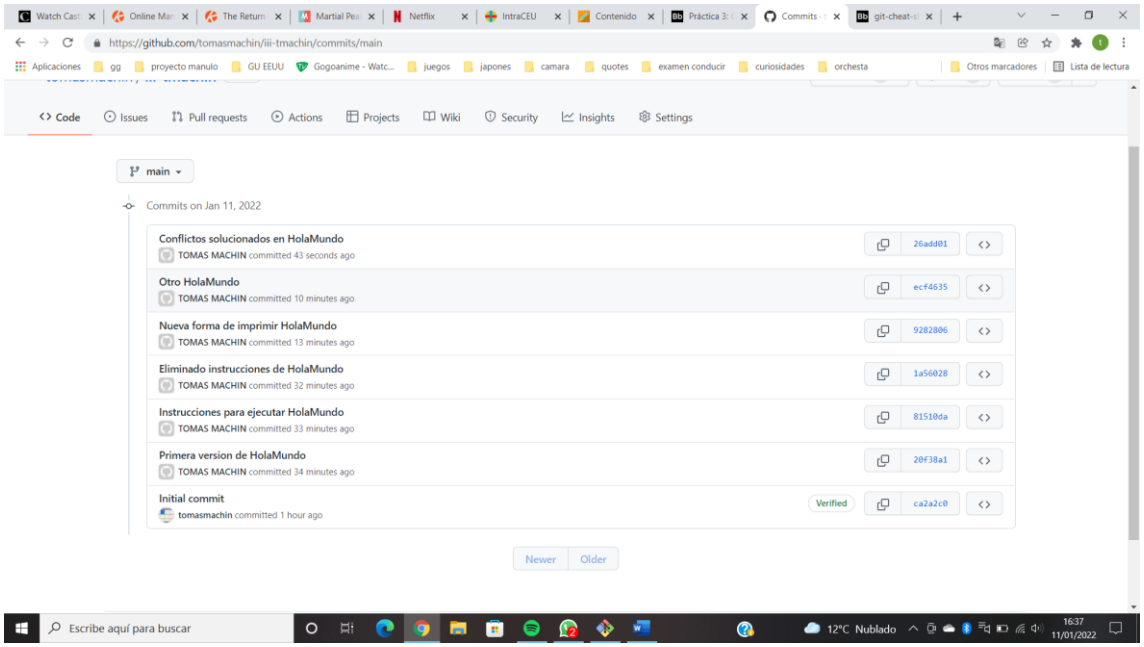
:wq

19.- git add *

20.- git commit -am “Conflictos solucionados en HolaMundo”

21.-git push

22.-



3.6 TRABAJR CON RAMAS Y SQUASH DE COMMITS

3.6.1 CREAR RAMA

1.-git brach rama_nueva

2.-git checkout rama_nueva

3.6.2 MERGE Y SQUASH COMMITS DE LA MISMA RAMA

3.-vim HolaMundo.java

i

```
public class HolaMundo{
  public static void main(String[] args){
    System.out.println("Otro Hola mundo"); }
  public static void printCourse(){
    System.out.println("Introducción Ingenieria Informática");}
}
```

esc

:wq

4.-git status (el estado del fichero es modificado y que es necasio un commit para guardarlo)

5.-git add *

6.-git status

7.- git commit -am “Nuevo método printCourse en HolaMundo”

8.- vim HolaMundo.java

i

```
public class HolaMundo{
```

```
public class HolaMundo{
  public static void main(String[] args){
    System.out.println("Otro Hola mundo");
    printCourse();
  }
  public static void printCourse(){
    System.out.println("Introducción Ingenieria Informática");
  }
}
```

esc

:wq

9.-git add *

10.-git commit -am “Llamada a imprimir nombre de la asignatura en HolaMundo”

11.-git rebase

12.-git merge rama_nueva

13.-git push

3.7 GITHUB Y LA TRAZABILIDAD

1.-ha tenido 16 contribuidores

2.-a) No hay ningun commit hecho en esa fecha

b)

c) Martin Kleusberg Mkleusberg

4 PARTE 2. INTERACCIÓN CON UNA BASE DE DATOS RELACIONAL

4.1 INSPECCIONAR LAS TABLAS DE LA BASE DE DATOS Y SUS ESQUEMAS

TABLAS:

Actor: 2 columnas con el id del actor y el id de la película

Director: 2 columnas con el id del director y el id de la película

Movie: 7 columnas con: el respectivo id, el nombre, el año, la calificación, la duración, el género y las ganancias de cada película.

Oscar: 4 columnas con: el id de la película, el id de la persona que ganó, el tipo y el año.

Person: 4 columnas con: su respectivo id, el nombre, la fecha de nacimiento, y donde nació.

4.2 VISUALIZAR/MODIFICAR LOS DATOS DE LAS TABLAS

1.-Tablas: 3281 registros

1	0000138	0120338
2	0000701	0120338
3	0000709	0120338

Director: 671 registros

1	0000116	0120338
2	0000184	0076759
3	0011470	0298148

Movie: 652 registros

1	0499549	Avatar	2009	PG-13	162	AVYS	1
2	0120338	Titanic	1997	PG-13	194	DR	2
3	0369610	Jurassic World	2015	PG-13	124	A	3

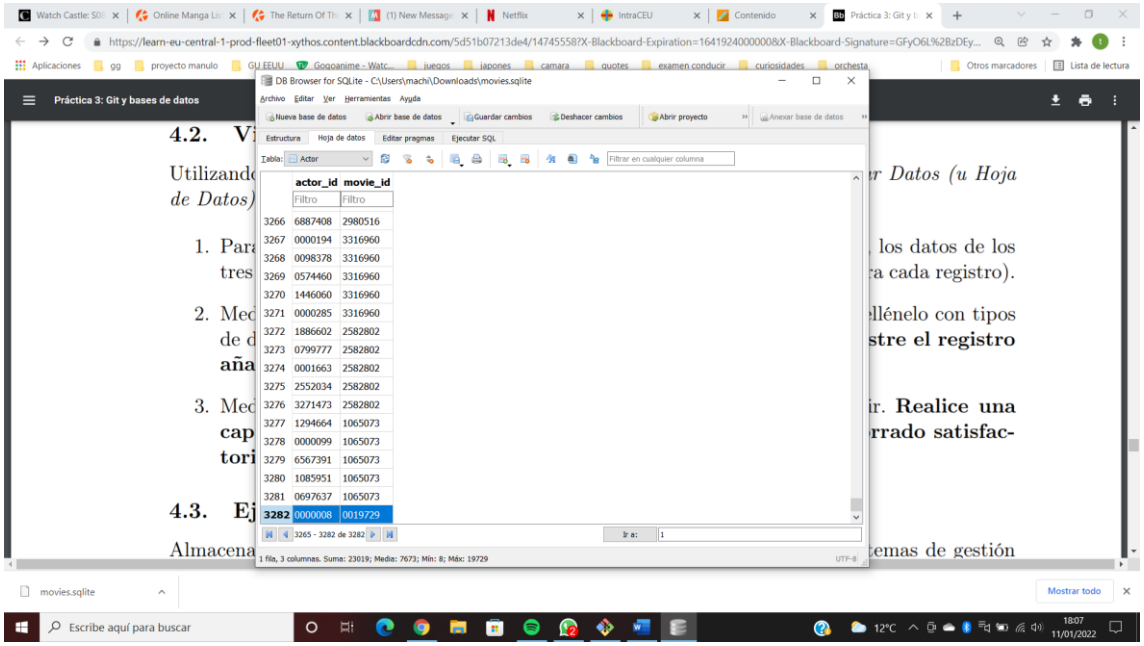
Oscar: 504 registros

1	2562232	NULL	BEST-PICTURE	2015
2	2980516	1519666	BEST-ACTOR	2015
3	3316960	0000194	BEST-ACTRESS	2015

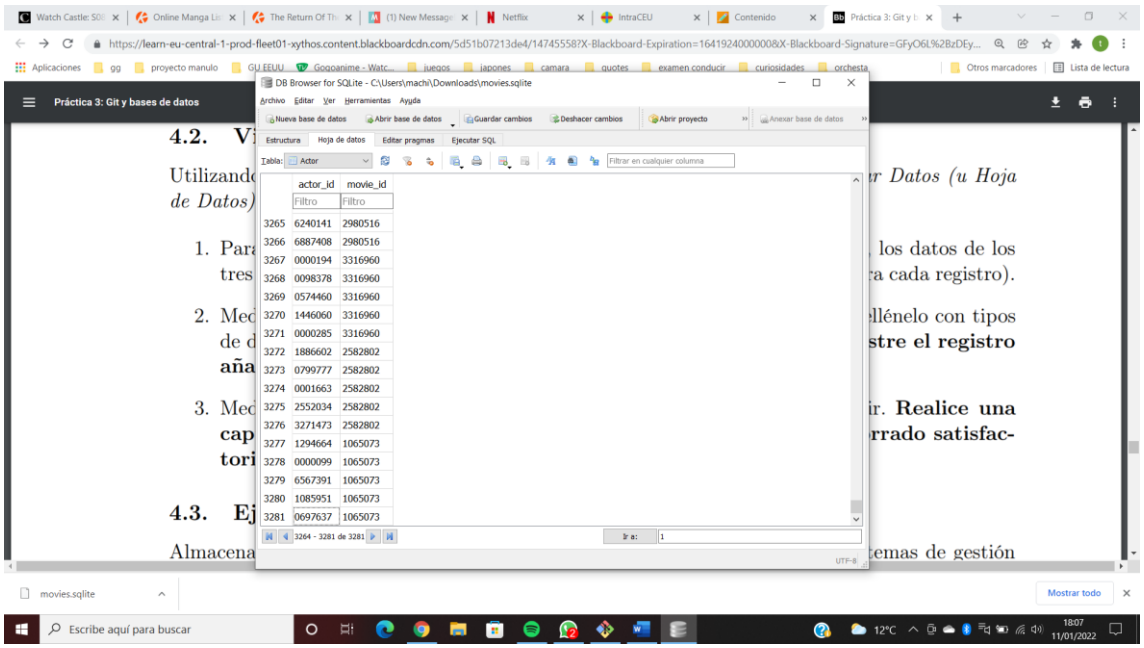
Person: 2416 registros

1	0000002	Lauren Bacal	1924-09-16	New York, New York, USA
2	0000004	John Belushi	1949-01-24	Chicago, Illinois; USA
3	0000006	Ingrid Bergam	1915-08-29	Stockholm, Sweden

2.-

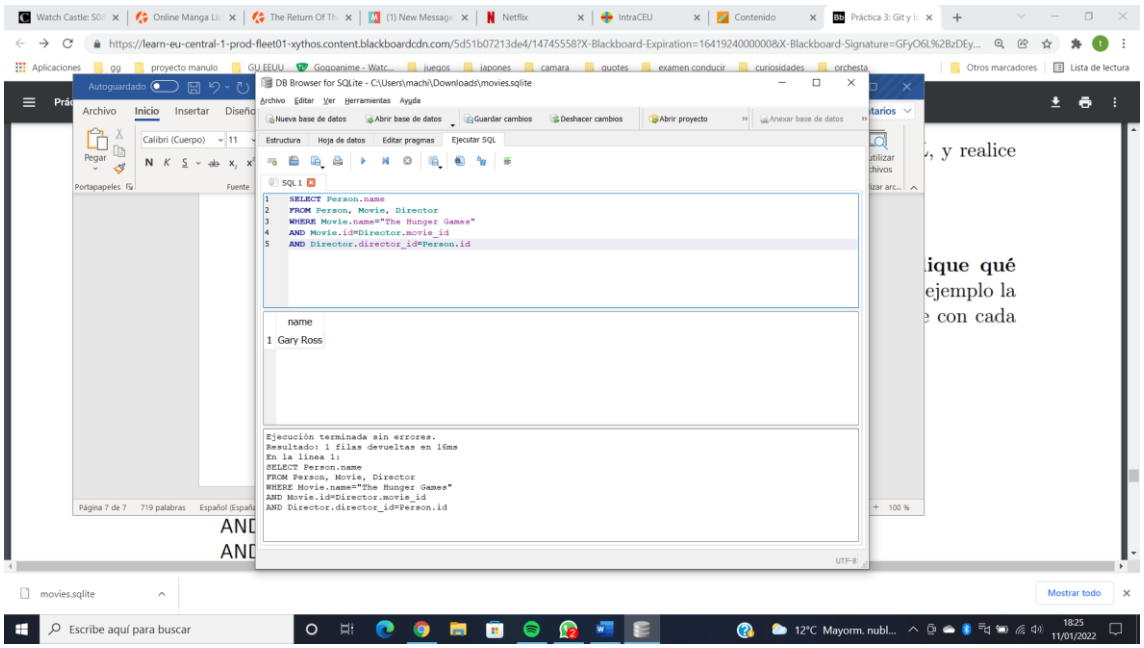

DB Browser for SQLite - C:\Users\machi\Downloads\movies.sqlite
Práctica 3: Git y bases de datos
actor_id movie_id
3265 6240141 2980516
3266 6887408 2980516
3267 0000194 3316960
3268 0098378 3316960
3269 0574460 3316960
3270 1446060 3316960
3271 0000285 3316960
3272 1886602 2582802
3273 0799777 2582802
3274 0001663 2582802
3275 2552034 2582802
3276 3271473 2582802
3277 1294664 1065073
3278 0000099 1065073
3279 6567391 1065073
3280 1085951 1065073
3281 0697637 1065073
3282 0000008 0019729
movies.sqlite
Mostrar todo
Escribe aquí para buscar

3.-


DB Browser for SQLite - C:\Users\machi\Downloads\movies.sqlite
Práctica 3: Git y bases de datos
actor_id movie_id
3265 6240141 2980516
3266 6887408 2980516
3267 0000194 3316960
3268 0098378 3316960
3269 0574460 3316960
3270 1446060 3316960
3271 0000285 3316960
3272 1886602 2582802
3273 0799777 2582802
3274 0001663 2582802
3275 2552034 2582802
3276 3271473 2582802
3277 1294664 1065073
3278 0000099 1065073
3279 6567391 1065073
3280 1085951 1065073
3281 0697637 1065073
movies.sqlite
Mostrar todo
Escribe aquí para buscar

4.3 EJECUCIÓN DE CONSULTAS

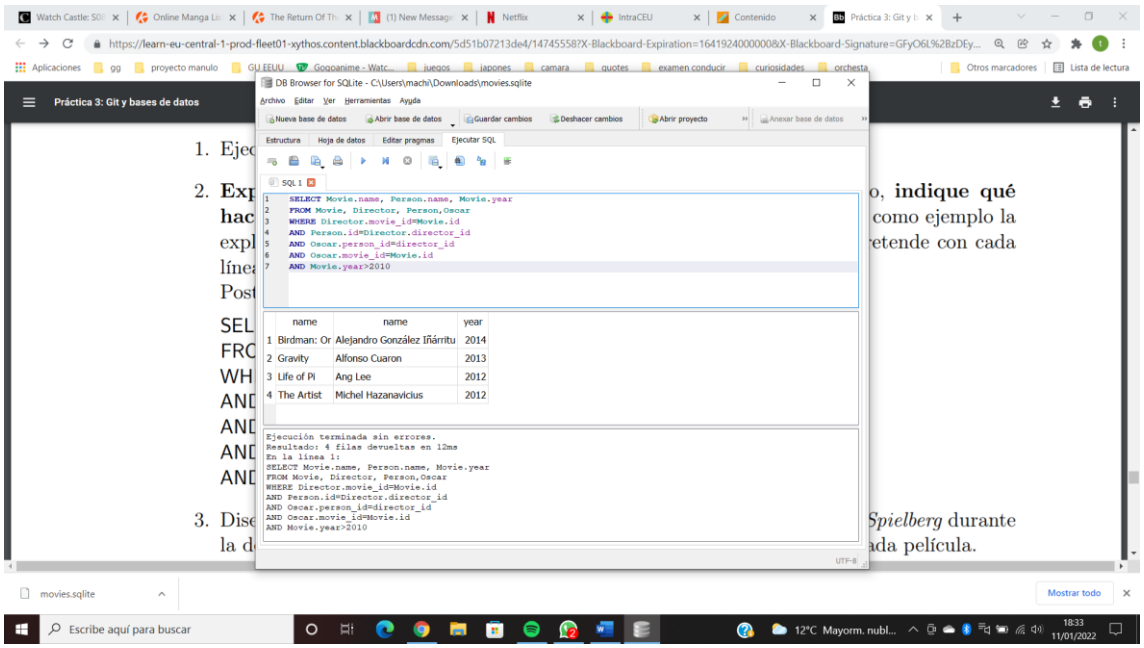
1.-



2.- SELECT: se selccionan los datos que queremos obtener, en este caso, el nombre de la película, le nombre de la persona y el año en el que salió la película.

FROM: es de dónde queremos sacar esa información, en este caso de las tablas Movie, Director, Person y Oscar.

WHERE: establece las condiciones que tiene que tener para que se pueda imprimir espécificamente la información que queremos. Primero se requiere de una relación entre tablas para poder sacer toda la información necesitada (Director.movie_id=Movie.id, Person.id=Director.director_id, Oscar.person_id=director_id, Oscar.movie_id=Movie.id). Además queremos que se cumplan una serie de condiciones para poder mostrar solo la información que queremos por patalla. Hay una sola condición: que el año en el que salió la película sea posterior a 2010 (Movie.year>2010).



3.-

SELECT Movie.name, Movie.year

FROM Movie, Director, Person,Oscar

WHERE Director.movie_id=Movie.id

AND Person.id=Director.director_id

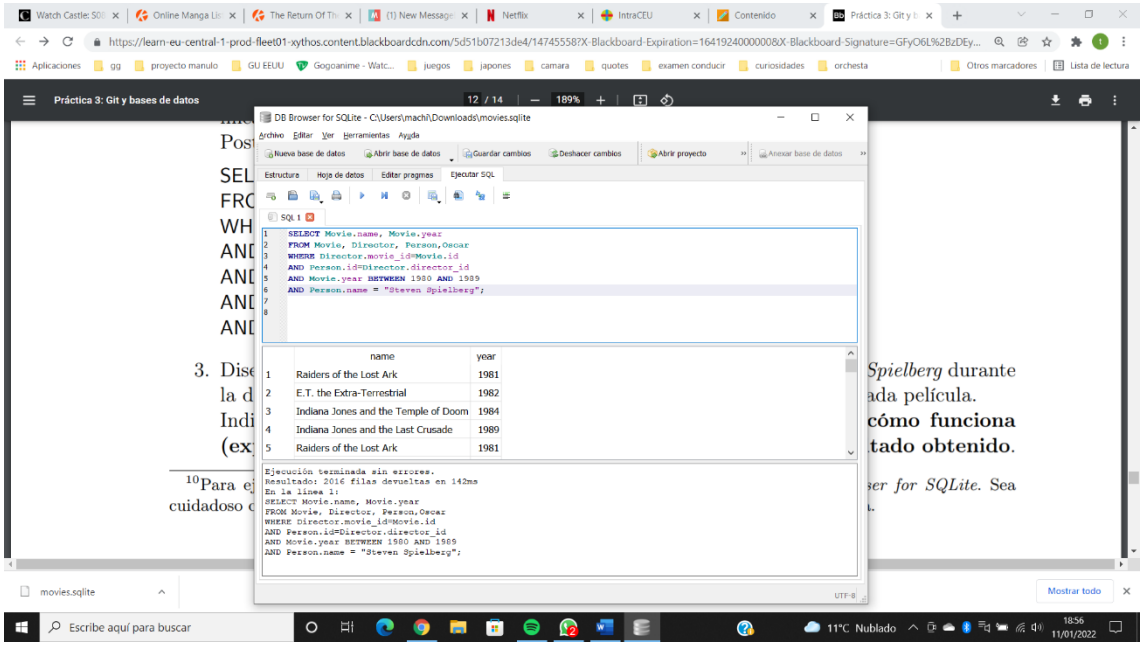
AND Movie.year BETWEEN 1980 AND 1989

AND Person.name = "Steven Spielberg";

SELECT sirve para seleccionar solo esa información para mostarla (año y nombre de la película)

FROM sive para saber de dónde se saca la información (tablas Director, Movie, Oscar, Person)

WHERE establece relaciones entre las tablas y las condiciones necesarias para imprimir la información que se quiere (en este caso que sean dirigidas por Steven Spielberg y que la película de los años 80)



4.- SELECT Movie.name, Movie.year, Person.name, Movie.earnings_rank

FROM Director, Person, Movie, Oscar

WHERE Director.movie_id=Movie.id

AND Person.id=Director.director_id

AND Oscar.person_id=director_id

AND Oscar.movie_id=Movie.id

AND Oscar.type = "BEST-ACTOR"

AND Movie.earnings_rank < 200

SELECT sirve para seleccionar la información que queremos (nombre, año y rango de ganancias de la película y el nombre de la persona).

FROM sirve para decir de dónde sacamos esa información.

WHERE establece una relación entre las tablas y establece una serie de condiciones para que la información que se muetra haya seguido esas pautas (top 200 de ganancias, y ganadores del BEST-ACTOR).

5.- SELECT Person.name, Movie.name, Movie.year

FROM Director, Person, Movie, Oscar

WHERE Director.movie_id=Movie.id

AND Person.id=Director.director_id

AND Oscar.person_id=director_id

AND Oscar.movie_id=Movie.id

AND Oscar.type = "BEST-SUPORTING-ACTRESS"

SELECT sirve para seleccionar la información que queremos (nombre de la actriz, año y nombre de la película).

FROM sirve para decir de dónde sacamos esa información.

WHERE establece una relación entre las tablas y establece una serie de condiciones para que la información que se muetra haya seguido esas pautas (ganadoras del oscar BEST-SUPPORTING-ACTRESS).