1) Ejecuta una sentencia **ALTER** en el que agregas la columna **date\_viewed** a la tabla **viewed**.

La columna **date\_viewed** debe ser:

- Del tipo **DATETIME** para que nos permita persistir tanto la fecha como la hora exacta en la que consumimos el contenido multimedia.

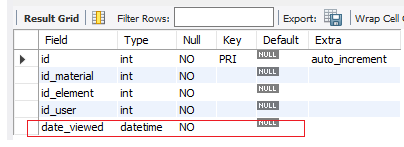
- **NOT NULL**. Recuerda que cada una de las filas de esta tabla representa un contenido multimedia YA visto.

**ALTER TABLE `viewed` ADD `date\_viewed` DATETIME NOT NULL;**

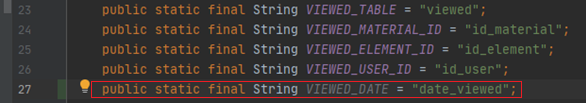
2) Comprueba que se haya agregado de forma exitosa el campo **date\_viewed** ejecutando una sentencia **DESCRIBE** para la tabla **viewed**.

**DESCRIBE viewed;**

La sentencia debería mostrar un resultado similar al de la siguiente imagen:



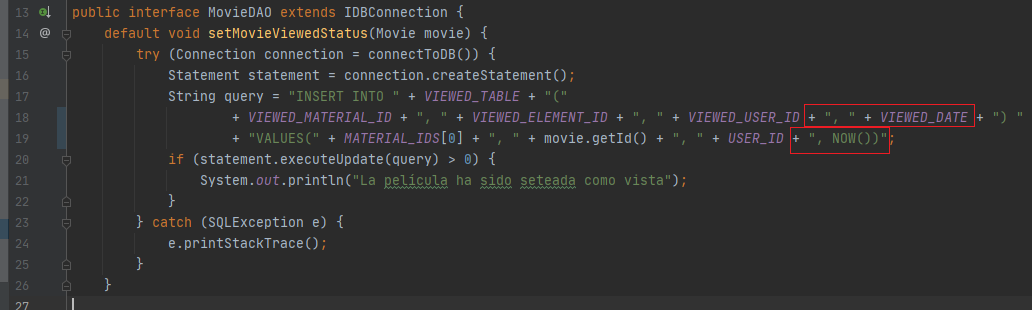
3) Crea en la clase Database de tu proyecto amazonviewer una constante que contenga el nombre de la nueva columna. Esto te permitirá que puedas agregar más tarde el nombre de la nueva columna al statement INSERT que setea una Movie como vista, dentro de la clase MovieDAO.



4) En la interfaz MovieDAO, modifica el statement INSERT del métodoque setea como vista una Movie:

- Agrega la constante que creaste en el paso anterior para indicar una nueva columna para la cual se persistirá información.

- Escribe la función de MySQL **NOW()** para el valor que corresponda a la posición de esa columna. Esta función cual creará un string que contiene una instancia de la fecha y la hora actual.

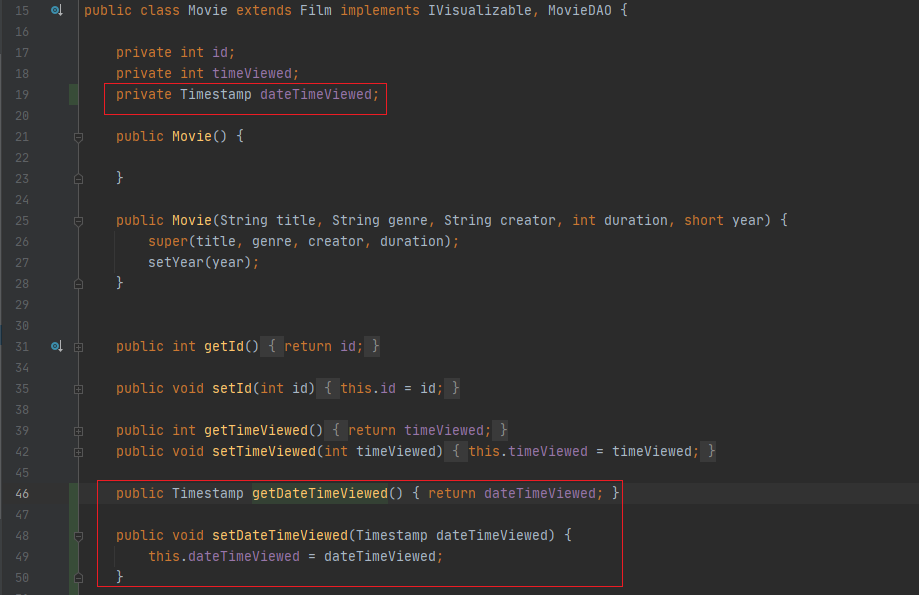


5) En la interfaz MovieDAO, crea un método privado que se encargue de devolver la fecha y la hora en la que fue vista una Movie. Yo lo decidí llamarlo **getDateTimeViewed**, e hice que devuelva un objeto **Timestamp**, ya que es un tipo de dato que no genera problemas de compatibilidad con los campos DATETIME de MySQL.

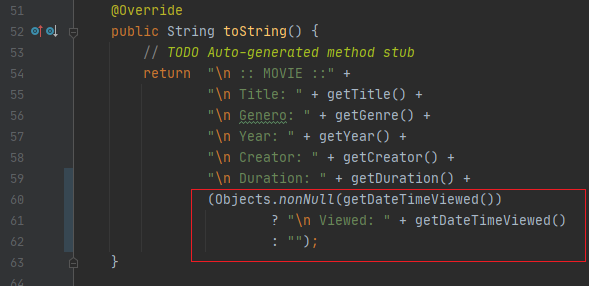


6) Haz los siguientes cambios en la clase Movie:

- Crea la propiedad dateTimeViewed y sus getter y setter. DISCLAIMER: No incluyas la nueva propiedad en el constructor, ya que a los objetos Movie cuando son creados no le asignamos nunca un valor a la propiedad dateTimeViewed. Recuerda que las películas no estarían vistas todavía si fueron agregadas recién.



- Agrega al método toString() la opción de que se muestre la fecha y la hora en la que fue vista la Movie en caso de que haya sido vista.



7) En el método default read() de la interfaz MovieDAO, refactoriza el código que setea el campo viewed de un objeto Movie a partir del método getMovieViewedStatus (ver imagen del paso 4):

- Haz que primero se obtenga el resultado de getMovieViewedStatus, y que luego basado en si la Movie fue vista o no, que se setee un Boolean.TRUE o un Boolean.FALSE en el campo viewed del objeto Movie, respectivamente. DATO CURIOSO: Es una buena prácitca utilizar Boolean.TRUE o Boolean.FALSE en vez de las literales “true” o “false”.

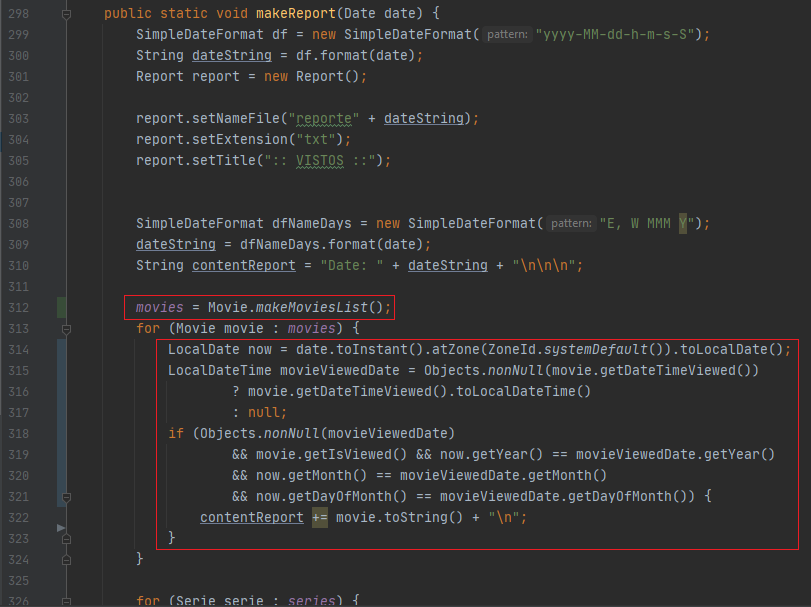
- Si la Movie fue vista, llama al setter para el campo dateTimeViewed que creamos en el paso 6 en la clase Movie, y por parámetros pasale una llamada a la función getDateTimeViewed, como podemos ver en la imagen de abajo. Con esto, obtienes la información de cuándo fue vista la Movie que será de utilidad al llamar al elemento del Menú que muestra la información sobre películas o a los que generan los reportes.



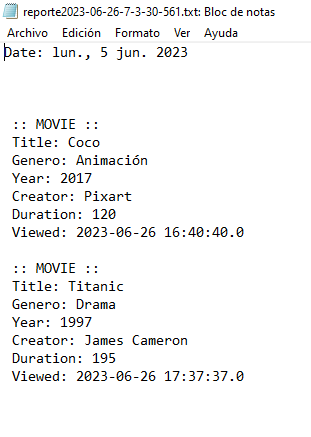
8) En el método makeReport(Date date) de la clase Main, actualiza la lista de movies llamando al método Movie.makeMoviesList(). Este método se trae los últimos registros de la base de datos. Luego, dentro del forEach que itera por cada Movie recuperada de la base de datos, antes de añadir una Movie como contenido del reporte:

- Verifica que la propiedad movieViewedDate del objeto Movie no esté vacía.

- Verifica que la fecha en la que fue vista una Movie coincida con la fecha date que recibimos por parámetro.



9) ¡Listo! Ahora cuando elijas la opción de crear un reporte por fecha, solamente se reportaran las Movies los que hayan sido vistas hoy tal que así:



\* \* \*

\*\*Introducción\*\*

En este tutorial, aprenderás a resolver el [reto](https://platzi.com/clases/1236-java-avanzado/10988-reto-reporte-por-fecha/) de la Clase 31 del [Curso Avanzado de Java SE](https://platzi.com/cursos/java-avanzado/) dictado por la profe [@anncode](https://platzi.com/profes/anncode/). Pese a que este tutorial incluya capturas de pantalla con la solución en código que pueden ser de gran ayuda, el objetivo principal sería que te enfoques en el \*\*por qué\*\* detrás de cada cambio que realizarás 💚.

\* \* \*

\_\*\*Paso 1\*\*\_: Ejecuta una sentencia ALTER en el que agregas la columna date\_viewed a la tabla viewed.

La columna date\_viewed debe ser de los tipos:

- DATETIME para que nos permita persistir tanto la fecha como la hora exacta en la que consumimos el contenido multimedia.

- NOT NULL. ❕ Recuerda que cada una de las filas de esta tabla representa un contenido multimedia \*\*YA\*\* visto.

`ALTER TABLE 'viewed' ADD 'date\_viewed' DATETIME NOT NULL;`

\_\*\*Paso 2\*\*\_: Comprueba que se haya agregado de forma exitosa el campo date\_viewed ejecutando una sentencia DESCRIBE para la tabla viewed.

`DESCRIBE viewed;`

Luego de que ejecutes la sentencia, debería mostrarse en pantalla un resultado similar al de la siguiente imagen:

![Resultado de la ejecución de la sentencia describe](https://i.imgur.com/hmsAam0.png)

\_\*\*Paso 3\*\*\_: Crea en la clase Database de tu proyecto amazonviewer una constante de tipo String que contenga el nombre de la nueva columna. Esto te permitirá que puedas agregar más adelante el nombre de la nueva columna al statement INSERT que se encarga de setear una Movie como 'viewed', ubicado en la clase MovieDAO.

![Se agrega la constante VIEWED\_DATE](https://i.imgur.com/KYurff8.png)

\_\*\*Paso 4\*\*\_: En la interfaz MovieDAO, modifica el statement INSERT del método que setea como 'viewed' una Movie:

1) Agrega la constante que creaste en el paso anterior para indicar que se persistirá información en la columna 'date\_viewed'.

2) Escribe la función de MySQL NOW() donde iría ubicado el valor que corresponde a la posición de la columna 'date\_viewed'. 🛑 La función \*\*NOW()\*\* retorna una instancia de la fecha y la hora actuales.

![Se añade información adicional a la query](https://i.imgur.com/EsOsYe5.png)

\_\*\*Paso 5\*\*\_: En la interfaz MovieDAO, crea un método \*\*privado\*\* que se encargue de devolver <ins>la fecha y la hora</ins> en la que fue vista una Movie. 👀 Yo decidí llamarlo getDateTimeViewed, e hice que devuelva un objeto \*\*Timestamp\*\*, ya que es un tipo de dato que no genera problemas de compatibilidad con los campos DATETIME de MySQL.

![Se crea el método getDateTimeViewed](https://i.imgur.com/d2ESOWd.png)

\_\*\*Paso 6\*\*\_: Haz los siguientes cambios en la clase Movie:

1. Crea la propiedad dateTimeViewed junto a sus métodos getter y setter. DISCLAIMER ⚠️: \*\*No\*\* incluyas la nueva propiedad en el constructor, ya que a la propiedad dateTimeViewed de los objetos Movie \*\*no\*\* le asignamos un valor cuando son creados. 💡 Recuerda que las películas teóricamente no estarían vistas todavía si fueron agregadas hace un instante.

![Se agrega la propiedad dateTimeViewed, su getter y su setter a la clase Movie](https://i.imgur.com/F3oT108.png)

2. Añádele al método toString() la opción de que se muestre la fecha y la hora en la que fue vista la Movie en caso de que haya sido vista.

![Se agrega la propiedad dateTimeViewed al método toString](https://i.imgur.com/xVPLwrw.png)

\_\*\*Paso 7\*\*\_: Dentro del método default read() de la interfaz MovieDAO, refactoriza el código que setea el campo viewed de un objeto Movie a partir del método getMovieViewedStatus (ver imagen del \_\*\*Paso 4\*\*\_ ⬆️) de la siguiente manera:

1. Haz que primero se obtenga el resultado de getMovieViewedStatus, y que luego basado en si la Movie fue vista o no, que se setee un Boolean.TRUE o un Boolean.FALSE, respectivamente, como valor para la propiedad 'viewed' del objeto tipo Movie. 📝DATO CURIOSO: Es una buena prácitca utilizar Boolean.TRUE o Boolean.FALSE en vez de las literales “true” o “false”. 👈🤓

2. Si la Movie fue vista, llama al método setter de la propiedad dateTimeViewed (creado en el \*\*\_Paso 6\_\*\* dentro de la clase Movie ⬆️), y pásale por parámetros una llamada a la función getDateTimeViewed, como puedes observar en la imagen de abajo. Al llamar a este último método, obtienes la fecha y la hora en la que la Movie fue seteada como 'viewed'. Esta información te será de utilidad para cuando elijas funcionalidades desde Menú tales como la que se encarga de mostrarte la información sobre las películas cargadas, o la que se encarga de generar los reportes.

![Se obtiene la fecha y hora en la que fue seteada la Movie como 'viewed'](https://i.imgur.com/5sKAtfE.png)

\_\*\*Paso 8\*\*\_: Dentro del método makeReport(Date date) de la clase Main, llama al método Movie.makeMoviesList(). Este método se trae los últimos registros de Movies desde la base de datos. Luego, dentro del forEach que itera por cada Movie recuperada, antes de añadir una Movie como contenido del reporte:

1. ❕ Verifica que la propiedad movieViewedDate del objeto Movie no esté vacía.

2. ❕ Verifica que la fecha en la que fue vista una Movie coincida con la fecha 'date' que recibimos por parametro al llamar a makeReport(Date date).

![Se agregan las validaciones](https://i.imgur.com/lCdcHaU.png)

\* \* \*

\*\*\_¡Listo!\_\*\* Ahora cuando elijas la opción de crear un reporte por fecha, sólamente se reportaran las Movies que hayan sido vistas el mismo día en el que estés creando el reporte, tal que \*\*así\*\*:

![Report por día](https://i.imgur.com/SshSeku.png)