

Entrega II: Diseño de una aplicación

1. Descripción general

Finalmente ha llegado el momento de trabajar en tu aplicación! En esta entrega debes diseñar una buena base de datos y una interfaz web para hacer consultas. El objetivo de la entrega es aplicar los conceptos aprendidos acerca del buen diseño de base de datos al problema del proyecto, y ejecutar consultas en un entorno realista. También aprenderás a trabajar con una conexión a una base de datos local.

2. Enunciado

2.1. Crear modelo

En esta parte deberás crear un modelo relacional usando los conceptos aprendidos en clases. Debes hacer lo siguiente:

- Construir un modelo de Entidad/Relación que represente de manera fiel la situación planteada en la entrega 1, pero ahora respaldados por un conjunto de datos y que les permitirán aclarar varias dudas que les pudieron quedar a partir del acercamiento inicial (recuerda que la aplicación es distinta para los grupos par o impar). Se deben ilustrar las entidades, relaciones y multiplicidades usando la notación vista en clases.
- Traspasar el modelo Entidad/Relación al modelo relacional, especificando el tipo/dominio de cada atributo e incluyendo llaves primarias y llaves foráneas como restricciones de integridad (no es necesario agregar otro tipo de restricciones adicionales). Las llaves foráneas pueden ser explicadas textualmente o implícitamente por el nombre de los campos (mientras se entienda).

2.2. Justificar modelo

En esta parte debes justificar que tu modelo relacional esta teóricamente correcto, recuerden que debe ser aplicado a su modelo en particular y no a algo general. Específicamente:

- Debes mostrar que tu modelo está en BCNF. Si no se puede, entonces deben mostrar que está en 3NF. Para ello debes listar las dependencias funcionales **no triviales de cada tabla** y mostrar que está todo normalizado.

Puede que necesites hacer algunos ajustes a tu modelo relacional para normalizarlo. El modelo final es el que debe ser entregado en el reporte.

2.3. Importar datos

Una vez que tengas el modelo preciso, es tiempo de importar los datos. Con esta entrega se subirán archivos .CSV con todos los datos que debes guardar en tu base de datos.

Ojo: Los datos no están normalizados, por lo que deberás trabajar un poco para importarlos¹. Puedes usar una interfaz externa ² o desde la consola. ³

Específicamente, hay que:

- Pre-procesar los datos sin pérdida de información para que se ajusten a tu esquema final.
- Ingresar a la base de datos del servidor llamada **grupoXe2** (por ejemplo grupo166e2, si tu grupo es 166) y crear todas las tablas de tu modelo relacional⁴. Posteriormente, importar los datos de manera que la importación sea sin pérdida de información.

2.4. Consultas en SQL

En esta sección se detallan las consultas para cada grupo. Estas consultas pueden tener un *input* dado que se efectuarán en el contexto de una página web. Los detalles de la página web se explican en la siguiente sección. Ojo que **no son las mismas que la entrega 1**.

Consultas grupo impar:

¹En general, puedes procesar estos *csv* con Python o tu lenguaje de programación favorito para facilitar la importación.

²Como DataGrip.

³Como se explica en la Wiki.

⁴En cada entrega de proyecto tendrán que usar una BD distinta. Esto con el objetivo de congelar las anteriores para poder corregir.

1. Muestre todas las películas junto con sus proveedores, siempre y cuando el proveedor las ofrezca de manera gratuita.
2. Dado un número n ingresado por el usuario, muestre todas las series que tengan al menos n temporadas.
3. Dado un título ingresado por el usuario, muestre todas las películas/series con ese título y los proveedores que las ofrecen.
4. Dado un género seleccionado por el usuario, muestre todas las películas que pertenezcan a ese género, o que pertenezcan a alguno de sus subgéneros **inmediatos**⁵.
5. Dado un username ingresado por el usuario, muestre todas las películas a las que tiene acceso dicho usuario.
6. Dado un username ingresado por el usuario, muestre todas las series para las cuales el usuario ingresado ha visto más de un capítulo en el último año.
7. Muestre la suma de dinero gastada por cada usuario en películas no incluidas en planes de suscripción.

Consultas grupo par:

1. Muestre todos los juegos y los proveedores que los ofrecen.
2. Dado un número n ingresado por el usuario, muestre todos los juegos con al menos n reseñas positivas.
3. Dado un título de juego ingresado por el usuario, muestre todas los juegos con ese título y los proveedores que los ofrecen.
4. Dado un género seleccionado por el usuario, muestre todos los juegos que pertenezcan a ese género, o que pertenezcan a alguno de sus subgéneros **inmediatos**⁶.
5. Dado un username ingresado por el usuario, muestre todos los juegos, junto con el nombre del proveedor respectivo de ese usuario.
6. Dado un username ingresado por el usuario, muestre todos los proveedores para los cuales el usuario ingresado ha preordenado más de un juego.
7. Muestre el gasto total de cada usuario en juegos por subscripción.

⁵En particular para esta consulta considerar lo siguiente: si un género C es subgénero de B, y B subgénero de A, eso **NO** implica que C es subgénero de A. En otras palabras, los subgéneros solo son relevantes en un nivel, **no hay transitividad**. Esto puede afectar la modelación de sus esquemas, pero debería simplificarla.

⁶Ídem 5

Ojo: Las búsquedas que solicitan ingresar un dato, deben ser *case-insensitive* y con *matching* parcial. Esto quiere decir, por ejemplo, que las búsquedas por “Punta Arenas”, “punta arenas”, “PUNTA arenas” o “Punta Arena” deben entregar los mismos resultados. Las que solicitan seleccionar deben permitir elegir entre las opciones posibles.

2.5. Página Web

Tu grupo deberá implementar una interfaz web en PHP para visualizar sus consultas. La interfaz puede ser simple; para las consultas que no tienen input, solo se requiere que cada consulta despliegue el resultado en la misma página, o se pueden mostrar utilizando un **link de ruta relativa**⁷ a una nueva página con el resultado de la consulta. Para las consultas con input, tu interfaz debe ser capaz de recibir sólo el parámetro necesario (no la consulta SQL entera). Por ejemplo, si la consulta requiere una comuna, el input debe funcionar solamente escribiendo o seleccionando el nombre o identificador de esa comuna.

Adicionalmente, se debe hacer la distinción entre los dos tipos de inputs con los que deberían trabajar, aquellos en los que se solicita seleccionar información, los cuales requieren del uso de un *dropdown*, y aquellos campos que piden ingresar un dato, en los que el usuario **debe** escribir el parámetro a utilizar.

3. Detalles adicionales

Ubicación de su entrega

1. Los archivos de la app web deben estar ubicados en el directorio `/grupoX/Sites`
2. El *homepage* debe ser `/grupoX/Sites/index.php`, es decir, ubicarse en el directorio del item anterior y llamarse `index.php`.
3. El reporte se debe entregar en la carpeta `/grupoX/Entrega2`. El nombre del archivo no es relevante, pero el de la carpeta si

Es importante que siga estas tres instrucciones, de lo contrario su trabajo **no será corregido**. También asegúrense de que el archivo o carpeta se subió correctamente, navegando dentro de ella en el servidor y accediendo a su página web desde un navegador⁸. Recuerden que al final de la rúbrica contarán con una imagen de como debería verse la distribución de sus carpetas al ingresar al servidor.

⁷Tiene que ser relativo. Para entender la diferencia, considerar que un link a `/resultado1.php` es relativo, mientras que un link a `https://bachman.ing.puc.cl/~grupoX/resultado1.php` no lo es.

⁸Como se indica en la Wiki del curso

Reporte

Además de la aplicación web, deberá entregar un archivo PDF que contenga:

- Un **diagrama E/R** de su dominio incluyendo las multiplicidades de las relaciones entre entidades. Si alguna entidad tiene un nombre no sugerente deberá explicar su significado.
- El esquema relacional que resulta del diagrama E/R anterior, incluyendo el tipo de dato de cada atributo y todas las restricciones de llaves primarias y foráneas de las relaciones.
- Las dependencias funcionales no triviales y la justificación de por qué ese esquema se halla en BCNF o 3NF.
- Todas las consultas en SQL que implementaron en su aplicación.
- Incluir cualquier supuesto que hayan realizado sobre la entrega, mientras sea razonable.
- Deseablemente, incluir cualquier detalle que facilite la corrección estilo `readme`.

Corrección de las consultas

Debes asegurarte que el resultado de las consultas sea consistente con los datos. Para corregir las consultas, se probarán 4 casos de prueba predeterminados en aquellas que reciban parámetros. Las otras se prueban *one-time*. La nota dependerá de la correctitud del resultado en cada caso.

Bonificación +0.5

Para los alumnos interesados, se ofrecerá hasta 0.5 puntos extra en esta entrega a aquellos grupos cuya página sea sobresaliente en su diseño (a juicio del corrector). Se sugiere buscar herramientas o librerías hechas para esto en la web⁹.

4. Detalles Académicos

Deberán trabajar según los grupos asignados. Expliquen adecuadamente su trabajo. El equipo corrector se reserva el derecho de bajar la nota de aquellos trabajos que no estén bien

⁹Una herramienta muy utilizada es Bootstrap.

explicados, que cuenten con demasiadas faltas de ortografía o trabajos en que se dificulte la corrección (por ejemplo, al no adjuntar el informe). Además, el corrector puede optar por **no asignar puntaje a una consulta** si esta no se logra correr en la aplicación web.

Finalmente, deben exponer su interfaz utilizado el servidor otorgado en la entrega anterior. La interfaz debe estar construida en PHP, y debe permitir realizar todas las consultas que se señalan. Para más detalles sobre esta entrega, haremos una ayudantía el **viernes 29 de septiembre**.

El plazo para esta entrega vence el día viernes 20 de octubre a las 20:00 hrs.