

PROYECTO FINAL

1.- ANÁLISIS DEL TEXTO

Queremos un registro de las **películas** que tenemos en stock. Estas tendrán un número de identificación, su título y género, el proveedor al que se la hemos comprado, el precio (copia) que hemos pagado por la copia, el precio de alquiler que le hemos asignado y la cantidad de copias que tenemos en estos momentos.

También queremos almacenar la información de los **socios**. En este caso les adjudicaremos un número de socio, y almacenaremos su nombre y apellidos junto a su teléfono para poder contactar con ellos llegado la necesidad.

Guardaremos también en la BBDD los datos de nuestros **proveedores**. Su nombre y su teléfono de contacto serán suficientes para poder realizar reclamaciones o ponernos en contacto con ellos llegado el momento.

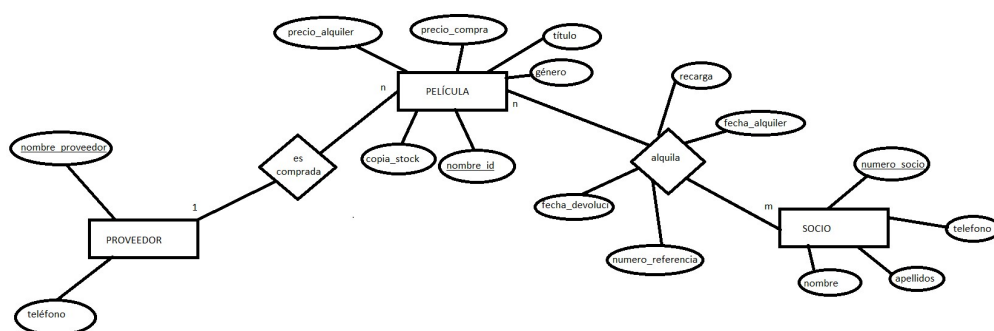
Llevaremos un registro de los **alquileres** que se realicen, marcándolos con un número de referencia. Necesitaremos almacenar la fecha de alquiler y la fecha de devolución.

Consideramos que si la fecha de devolución no está informada la película sigue alquilada.

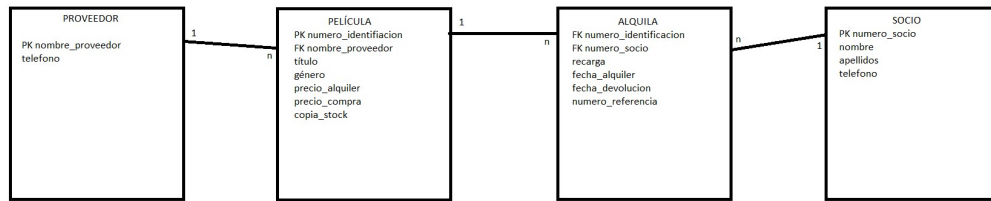
También queremos registrar las recargas que se apliquen a los socios en caso de retraso en su devolución.

Debe tenerse en consideración la eliminación de redundancias en la medida de lo posible, y quedará en vuestras manos la elección de restricciones aplicables tanto a nivel de columna como a nivel de tabla.

2.- DIAGRAMA DE FLUJO



3.- PASO A TABLAS: ENTIDAD RELACIÓN



4.- CREACIÓN DE TABLAS EN SAP WEB for SAP HANA

Tabla proveedor

```
11 /
12 -- Crear tabla Proveedor
13 CREATE COLUMN TABLE Proveedor (
14     nombre_proveedor VARCHAR(50) PRIMARY KEY,
15     telefono VARCHAR(20)
16 );
17
```

Tabla película

```
35 -- Crear tabla Pelicula
36 CREATE COLUMN TABLE Pelicula (
37     nombre_id INTEGER PRIMARY KEY,
38     nombre_proveedor VARCHAR(50),
39     copia_stock INTEGER,
40     precio_alquiler DECIMAL(10, 2),
41     precio_compra DECIMAL(10, 2),
42     titulo VARCHAR(100),
43     genero VARCHAR(50)
44 );
45
```


Tabla socio

```
96 -- Crear tabla Socio
97 CREATE COLUMN TABLE Socio (
98     numero_socio INTEGER PRIMARY KEY,
99     telefono VARCHAR(20),
00     nombre VARCHAR(50),
01     apellidos VARCHAR(50)
02 );
03
```

Tabla alquiler

```
168 -- Crear tabla Alquiler
169 CREATE COLUMN TABLE Alquiler (
170     nombre_id INTEGER,
171     numero_socio INTEGER,
172     recarga DECIMAL(10, 2),
173     fecha_alquiler DATE,
174     fecha_devolucion DATE,
175     PRIMARY KEY (nombre_id, numero_socio),
176     FOREIGN KEY (nombre_id) REFERENCES Pelicula(nombre_id),
177     FOREIGN KEY (numero_socio) REFERENCES Socio(numero_socio)
178 );
---
```

Tablas creadas

 ALQUILA
 PELICULA
 PROVEEDOR
 SOCIO

5.- Inserción de datos

-- Insertar datos tabla Proveedor

```
INSERT INTO Proveedor (nombre_proveedor, telefono)
VALUES ('Peliculas Paco SL', '653121313');
```

```
INSERT INTO Proveedor (nombre_proveedor, telefono)
VALUES ('CineMax', '654987654');
```

```
INSERT INTO Proveedor (nombre_proveedor, telefono)
VALUES ('CineWorld', '657890123');
```

```
INSERT INTO Proveedor (nombre_proveedor, telefono)
VALUES ('MoviePlanet', '656787878');
```

```
INSERT INTO Proveedor (nombre_proveedor, telefono)
VALUES ('FilmCity', '651234567');
```

-- Insertar datos tabla Pelicula

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (1, 'Cines Alfredo', 5, 2.99, 9.99, 'Los hombres de Paco', 'accion');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (2, 'CineMax', 8, 3.50, 12.99, 'El Gran Escape', 'drama');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (3, 'CineWorld', 3, 2.99, 9.99, 'La La Land', 'musical');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (4, 'MoviePlanet', 6, 2.50, 8.99, 'El Padrino', 'drama');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (5, 'FilmCity', 2, 1.99, 6.99, 'Regreso al Futuro', 'ciencia ficcion');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (6, 'CineStar', 4, 2.99, 9.99, 'Pulp Fiction', 'drama');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (7, 'CineMundo', 7, 3.99, 11.99, 'Titanic', 'romance');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (8, 'CinePlus', 1, 2.50, 8.99, 'El Rey León', 'animacion');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (9, 'CineMagic', 3, 2.99, 9.99, 'Matrix', 'ciencia ficcion');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (10, 'CineCity', 5, 2.50, 8.99, 'Memento', 'suspense');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (11, 'Películas Paco SL', 6, 3.50, 12.99, 'Harry Potter y la Piedra Filosofal', 'fantasia');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (12, 'CineMax', 4, 2.99, 9.99, 'El Resplandor', 'terror');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (13, 'CineWorld', 2, 1.99, 6.99, 'Jurassic Park', 'aventura');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (14, 'MoviePlanet', 3, 2.99, 9.99, 'El Señor de los Anillos: La Comunidad del Anillo', 'fantasia');
```

```
INSERT INTO Pelicula (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
VALUES (15, 'FilmCity', 7, 3.99, 11.99, 'Interestelar', 'ciencia ficcion');
```

-- Insertar datos tabla Socio

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (123, '765345213', 'Gustavo', 'Gonzalez Giraldez');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (124, '654789321', 'María', 'López García');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (125, '789654123', 'Juan', 'Martínez Sánchez');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (126, '987321456', 'Laura', 'Rodríguez Fernández');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (127, '123456789', 'Carlos', 'Hernández Morales');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (128, '456789123', 'Ana', 'Gómez Torres');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (129, '321654987', 'Pedro', 'Pérez Ríos');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (130, '789123456', 'Sara', 'Jiménez Navarro');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (131, '654987321', 'David', 'Ruiz Medina');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (132, '321789654', 'Marta', 'García Romero');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (133, '789456321', 'Eduardo', 'López Mendoza');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (134, '987654321', 'Laura', 'González Soto');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (135, '321456789', 'Carlos', 'Torres Hernández');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
VALUES (136, '456321789', 'Lucía', 'Martínez López');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
```

```
VALUES (137, '987123456', 'Pedro', 'Gómez Torres');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
```

```
VALUES (138, '654321789', 'Ana', 'Pérez Ríos');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
```

```
VALUES (139, '321789123', 'Hugo', 'García Romero');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
```

```
VALUES (140, '789654321', 'Laura', 'Ruiz Medina');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
```

```
VALUES (141, '456789321', 'Diego', 'López Mendoza');
```

```
INSERT INTO Socio (numero_socio, telefono, nombre, apellidos)
```

```
VALUES (142, '987654789', 'María', 'González Soto');
```

```
-- Insertar datos tabla Alquila
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (1, 123, 0.50, '2023-03-07', '2023-03-10');
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (2, 124, 0.50, '2023-03-08', '2023-03-11');
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (3, 135, 0.50, '2023-03-09', '2023-03-12');
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (4, 128, 0.50, '2023-03-10', '2023-03-13');
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (5, 127, 0.50, '2023-03-11', '2023-03-14');
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (6, 128, 0.50, '2023-03-12', '2023-03-15');
```

```
INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
```

```
VALUES (7, 127, 0.50, '2023-03-13', '2023-03-16');
```

INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)

VALUES (8, 135, 0.50, '2023-03-14', '2023-03-17');

INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)

VALUES (9, 131, 0.50, '2023-03-15', '2023-03-18');

INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)

VALUES (10, 139, 0.50, '2023-03-16', '2023-03-19');

INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)

VALUES (11, 142, 0.50, '2023-03-17', '2023-03-20');

INSERT INTO Alquila (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)

VALUES (12, 135, 0.50, '2023-03-18', '2023-03-21');

6.- Las tablas creadas

Tabla proveedor:

SQL Console 1.sql x

Analyze

Connected to: PABLO

```
--DROP TABLE Socio;
--DROP TABLE Alquila;
--comproban

Select * from Proveedor;
Select * from Pelicula;
Select * from Socio;
Select * from Alquila;

-- Crear tabla Proveedor
```

Result x Messages x

Rows (5)	NOMBRE_PROVEEDOR	TELEFONO
1	Películas Paco SL	653121313
2	CineMax	654987654
3	CineWorld	657890123
4	MoviePlanet	656787878
5	FilmCity	651234567

Tabla película:

SQL Console 1.sql x

Analyze

Connected to: PABLO

```
--DROP TABLE Socio;
--DROP TABLE Alquila;
--comproban

Select * from Proveedor;
Select * from Pelicula;
Select * from Socio;
Select * from Alquila;

-- Crear tabla Proveedor
```

Result x Messages x

Rows (15)	NOMBRE_ID	NOMBRE_PROVEEDOR	COPIA_STOCK	PRECIO_ALQUILER	PRECIO_COMPRA	TITULO	GENERO
1	1	Cines Alfredo	5	2.99	9.99	Los hombres de Paco	accion
2	2	CineMax	8	3.50	12.99	El Gran Escape	drama
3	3	CineWorld	3	2.99	9.99	La La Land	musical
4	4	MoviePlanet	6	2.50	8.99	El Padrino	drama
5	5	FilmCity	2	1.99	6.99	Regreso al Futuro	ciencia ficcion
6	6	CineStar	4	2.99	9.99	Pulp Fiction	drama
7	7	CineMundo	7	3.99	11.99	Titanic	romance
8	8	CinePlus	1	2.50	8.99	El Rey León	animacion
9	9	CineMagic	3	2.99	9.99	Matrix	ciencia ficcion
10	10	CineCity	5	2.50	8.99	Memento	suspense

Tabla socio:

SQL Console 1.sql x

Analyze

Connected to: PABLO

```
--DROP TABLE Socio;
--DROP TABLE Alquila;
--comprobar

Select * from Proveedor;
Select * from Pelicula;
Select * from Socio;
Select * from Alquila;
-- Crear tabla Proveedor
```

Result x Messages x

Rows (20)

	NUMERO_SOCIO	TELEFONO	NOMBRE	APELLIDOS
1	123	765345213	Gustavo	Gonzalez Giraldez
2	124	654789321	María	López García
3	125	789654123	Juan	Martínez Sánchez
4	126	987321456	Laura	Rodríguez Fernández
5	127	123456789	Carlos	Hernández Morales
6	128	456789123	Ana	Gómez Torres
7	129	321654987	Pedro	Pérez Ríos
8	130	789123456	Sara	Jiménez Navarro
9	131	654987321	David	Ruiz Medina
10	132	321789654	Marta	García Romero

Tabla alquila:

SQL Console 1.sql x

Analyze

Connected to: PABLO

```
--DROP TABLE Socio;
--DROP TABLE Alquila;
--comprobar

Select * from Proveedor;
Select * from Pelicula;
Select * from Socio;
Select * from Alquila;
-- Crear tabla Proveedor
```

Result x Messages x

Rows (12)

	NOMBRE_ID	NUMERO_SOCIO	RECARGA	FECHA_ALQUILER	FECHA_DEVOLUCION
1	1	123	0.50	2023-03-07	2023-03-10
2	2	124	0.50	2023-03-08	2023-03-11
3	3	135	0.50	2023-03-09	2023-03-12
4	4	128	0.50	2023-03-10	2023-03-13
5	5	127	0.50	2023-03-11	2023-03-14
6	6	128	0.50	2023-03-12	2023-03-15
7	7	127	0.50	2023-03-13	2023-03-16
8	8	135	0.50	2023-03-14	2023-03-17
9	9	131	0.50	2023-03-15	2023-03-18
10	10	139	0.50	2023-03-16	2023-03-19

7.- Control de inserción de datos

Procedimiento para la tabla proveedor

```
220 -- Crear procedimiento para insertar datos en la tabla Proveedor
221 CREATE OR REPLACE PROCEDURE InsertarProveedor(
222     IN telefono VARCHAR(20)
223 )
224 AS BEGIN
225     DECLARE id INTEGER;
226     SELECT COALESCE(MAX(nombre_proveedor), 0) + 1 INTO id FROM "CURSO_17_HDI_BBDD_3"."PROVEEDOR";
227     INSERT INTO "CURSO_17_HDI_BBDD_3"."PROVEEDOR" (nombre_proveedor, telefono) VALUES (:id, TRIM(:telefono));
228 END;
```

Procedimiento creado para insertar de forma controlada datos en la tabla proveedor.

Se toma un valor de “teléfono” como entrada, se obtiene el próximo id disponible y se inserta un nuevo registro.

En la consulta *Select* obtenemos el valor máximo de *nombre_proveedor* de la tabla “Proveedor” y lo añade a la variable *id*. En caso de que la tabla esté vacía se le asignará el valor 0 a *id* y se le suma 1. Después, se realiza una inserción con los parámetros proporcionados y utilizando la función *TRIM* para eliminar los espacios en blanco. Se utiliza *COALESCE* para determinar el primer valor nulo.

El punto 8, se hacen una inserción ejemplo para este procedimiento.

Procedimiento para la tabla película:

```
230 -- Crear procedimiento para insertar datos en la tabla Pelicula
231 CREATE OR REPLACE PROCEDURE InsertarPelicula(
232     IN nombre_proveedor INTEGER,
233     IN copia_stock INTEGER,
234     IN precio_alquiler DECIMAL(10, 2),
235     IN precio_compra DECIMAL(10, 2),
236     IN titulo VARCHAR(100),
237     IN genero VARCHAR(50)
238 )
239 AS BEGIN
240     DECLARE id INTEGER;
241     SELECT COALESCE(MAX(nombre_id), 0) + 1 INTO id FROM "CURSO_17_HDI_BBDD_3"."PELICULA";
242     INSERT INTO "CURSO_17_HDI_BBDD_3"."PELICULA" (nombre_id, nombre_proveedor, copia_stock, precio_alquiler, precio_compra, titulo, genero)
243     VALUES (:id, :nombre_proveedor, :copia_stock, :precio_alquiler, :precio_compra, TRIM(:titulo), TRIM(:genero));
244 END;
```

Procedimiento para crear inserciones de datos de forma controlada en la tabla Película.

Tomándose varios valores de entrada, se obtiene el próximo id disponible de la tabla Película y se inserta un nuevo registro.

En la consulta *Select* obtenemos el valor máximo de *nombre_id* de la tabla “Película” y lo añade a la variable *id*. En caso de que la tabla esté vacía se le asignará el valor 0 a *id* y se le suma 1. Después, se realiza una inserción con los parámetros proporcionados y utilizando la función TRIM para eliminar los espacios en blanco. Se utiliza *COALESCE* para determinar el primer valor nulo.

Procedimiento para la tabla socio:

```
246 -- Crear procedimiento para insertar datos en la tabla Socio
247 CREATE OR REPLACE PROCEDURE InsertarSocio(
248     IN telefono VARCHAR(20),
249     IN nombre VARCHAR(50),
250     IN apellidos VARCHAR(50)
251 )
252 AS BEGIN
253     DECLARE id INTEGER;
254     SELECT COALESCE(MAX(numero_socio), 0) + 1 INTO id FROM "CURSO_17_HDI_BBDD_3"."SOCIO";
255     INSERT INTO "CURSO_17_HDI_BBDD_3"."SOCIO" (numero_socio, telefono, nombre, apellidos) VALUES (:id, TRIM(:telefono), TRIM(:nombre), TRIM(:apellidos));
256 END;
257
```

Procedimiento para crear inserciones de datos de forma controlada en la tabla Socio.

Tomando los tres valores de entrada que se muestran, se obtiene el próximo número de socio disponible para la tabla socio y después se inserta un nuevo registro con los valores que se proporcionan.

En la consulta *Select* obtenemos el valor máximo de la columna *numero_socio* de la tabla “Socio” y lo añade a la variable *id*. En caso de que la tabla esté vacía se le asignará el valor 0 a *id* y se le suma 1. Después, se realiza una inserción con los parámetros proporcionados y utilizando la función TRIM para eliminar los espacios en blanco. Se utiliza *COALESCE* para determinar el primer valor nulo.

Procedimiento para la tabla alquiler:

```
258 -- Crear procedimiento para insertar datos en la tabla Alquiler
259 CREATE OR REPLACE PROCEDURE InsertarAlquiler(
260     IN nombre_id INTEGER,
261     IN numero_socio INTEGER,
262     IN recarga DECIMAL(10, 2),
263     IN fecha_alquiler DATE,
264     IN fecha_devolucion DATE
265 )
266 AS
267 BEGIN
268     DECLARE stock INTEGER;
269     SELECT p.copia_stock - COUNT(*) INTO stock
270     FROM "CURSO_17_HDI_BBDD_3"."PELICULA" as p
271     LEFT JOIN "CURSO_17_HDI_BBDD_3"."ALQUILA" as a ON p.nombre_id = a.nombre_id
272     WHERE p.nombre_id = :nombre_id AND a.fecha_devolucion IS NULL
273     GROUP BY p.copia_stock;
274
275     IF stock > 0 THEN
276         INSERT INTO "CURSO_17_HDI_BBDD_3"."ALQUILA" (nombre_id, numero_socio, recarga, fecha_alquiler, fecha_devolucion)
277         VALUES (:nombre_id, :numero_socio, :recarga, :fecha_alquiler, :fecha_devolucion);
278         SELECT 'La película se alquiló correctamente.' AS resultado FROM DUMMY;
279     ELSE
280         SELECT 'No hay copias disponibles de la película seleccionada.' AS resultado FROM DUMMY;
281     END IF;
282 END;
```

Procedimiento para crear inserciones de datos. Tanto si hay como copias disponibles como no de una película, devuelve un mensaje de si se alquila correctamente una película o de si no hay stock de esa misma película.

Lo primero es que se realiza una consulta de las tablas película y alquiler para saber el número de stock. Esta consulta hace un conteo de la cantidad de registros de la tabla alquiler para una película en concreto donde la fecha de devolución sea nula.

A continuación, se resta el valor de copia_stock y el resultado se introduce en la variable stock. Si el valor de stock es mayor que 0 es que existen copias disponibles.

Se mostrará un mensaje tanto de si se realizó la inserción correcta ("La película se alquiló correctamente") como si no hay copias disponibles ("No hay copias disponibles de la película seleccionada").

8.- Pruebas (ejemplo) para las llamadas de los procedimientos

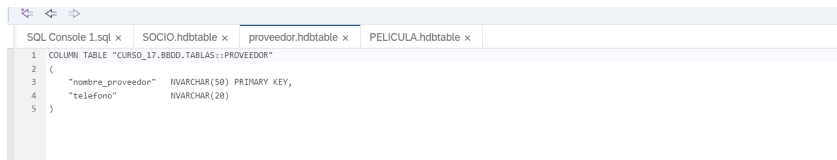
```
285 --Llamar procedimientos:
286
287 --Tabla proveedor
288 CALL InsertarProveedor('123456789');
289
290 --Tabla pelicula
291 CALL InsertarPelicula(123, 5, 10.99, 29.99, 'Torrente', 'Accion');
292
293 --Tabla socio
294 CALL InsertarSocio('123456789', 'Juan', 'Perez Alvarez');
295
296 --Tabla Alquiler
297 CALL InsertarAlquiler(123, 456, 10.5, '2023-06-07', '2023-06-10');
298
```

Se realizan pruebas de inserción con los parámetros solicitados por el ejercicio en cada una de las tablas.

9.- Creación de ficheros .hdbtable

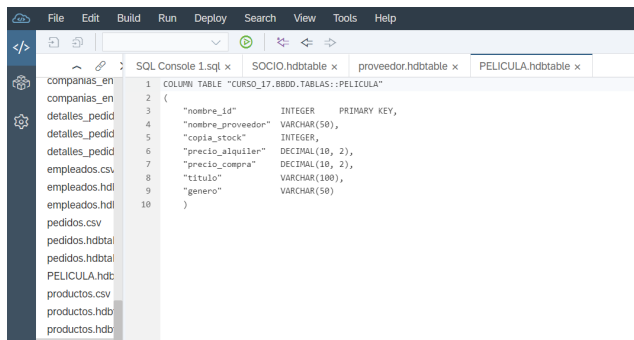
No es suficiente crear una tabla en SQL, sino que tienen que estar creada en un fichero .hdbtable.

-Proveedor



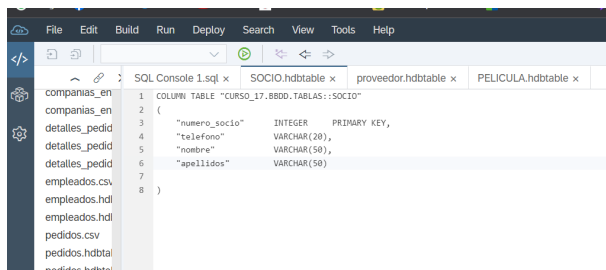
```
1 COLUMN TABLE "CURSO_17_8800.TABLAS::PROVEEDOR"  
2 (  
3     "nombre_proveedor"    NVARCHAR(50) PRIMARY KEY,  
4     "telefono"            NVARCHAR(20)  
5 )
```

-Película



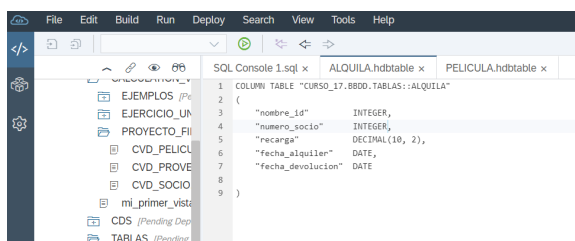
```
1 COLUMN TABLE "CURSO_17_8800.TABLAS::PELICULA"  
2 (  
3     "nombre_id"           INTEGER PRIMARY KEY,  
4     "nombre_proveedor"    VARCHAR(50),  
5     "copia_stock"         INTEGER,  
6     "precio_alquiler"     DECIMAL(10, 2),  
7     "precio_compra"       DECIMAL(10, 2),  
8     "titulo"              VARCHAR(100),  
9     "genero"              VARCHAR(50)  
10 )
```

-Socio



```
1 COLUMN TABLE "CURSO_17_8800.TABLAS::SOCIO"  
2 (  
3     "numero_socio"        INTEGER PRIMARY KEY,  
4     "telefono"            VARCHAR(20),  
5     "nombre"              VARCHAR(50),  
6     "apellidos"           VARCHAR(50)  
7 )
```

-Alquila



```
1 COLUMN TABLE "CURSO_17_8800.TABLAS::ALQUILA"  
2 (  
3     "nombre_id"           INTEGER,  
4     "numero_socio"        INTEGER,  
5     "recarga"              DECIMAL(10, 2),  
6     "fecha_alquiler"       DATE,  
7     "fecha_devolucion"     DATE  
8 )
```

10.- Vistas calculadas

- Vistas tipo dimensión:

El procedimiento es el mismo para todos, creamos una vista calculada con el nombre (CVD_X). Para ello las agrupé todas en la misma carpeta (PROYECTO_FINAL en la carpeta de CALCULATION_VIEW).

En el nodo *projection* en el “+” seleccionamos la carpeta de la que vamos a sacar la información (clientes, empleados...). En cada de una de ellas seleccionamos los campos que se nos piden para cada vista dimensión y con qué atributo se relacionan con el siguiente campo.

Para sacar los campos que se nos piden basta con mapear los campos que queremos. Guardamos y construimos.

Para cambiarle el nombre, una vez las mapeamos, seleccionamos la columna de salida y en propiedades le ponemos el nombre que queremos.

Guardamos y construimos.

En *Data preview* podremos ver los datos que hemos mapeado y ya con el nombre cambiado.

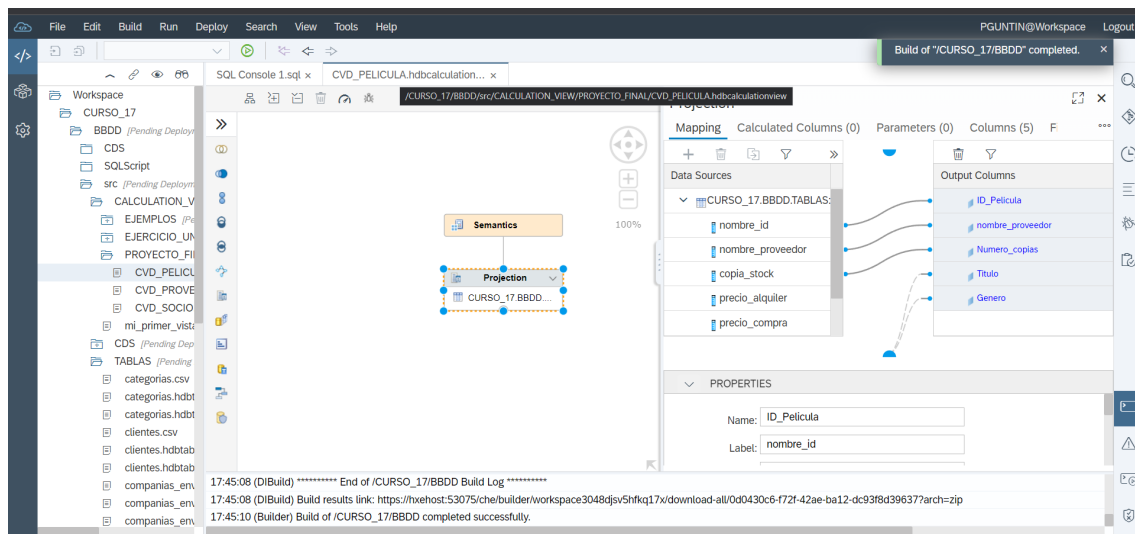
CVD_SOCIO:

The screenshot shows the SAP Web IDE interface with the 'CVD_SOCIO' projection node selected. The 'Mapping' tab is active, showing a data source 'CURSO_17.BBDD.TABLAS::SOCIO' with columns 'numero_socio', 'telefono', 'nombre', and 'apellidos'. These are mapped to output columns 'Numero_socio', 'Telefono', 'Nombre', and 'Apellidos'. The 'Properties' tab shows the 'Name' set to 'Telefono' and the 'Label' set to 'telefono'. The bottom log shows a successful build completion for 'CURSO_17/BBDD'.

CVD_PROVEEDOR

The screenshot shows the SAP Web IDE interface with the 'CVD_PROVEEDOR' projection node selected. The 'Mapping' tab is active, showing a data source 'CURSO_17.BBDD.TABLAS::PROVEED' with columns 'nombre_proveedor' and 'telefono'. These are mapped to output columns 'Nombre_proveedor' and 'Telefono'. The 'Properties' tab shows the 'Name' set to 'Telefono' and the 'Label' set to 'telefono'. The bottom log shows a successful build completion for 'CURSO_17/BBDD'.

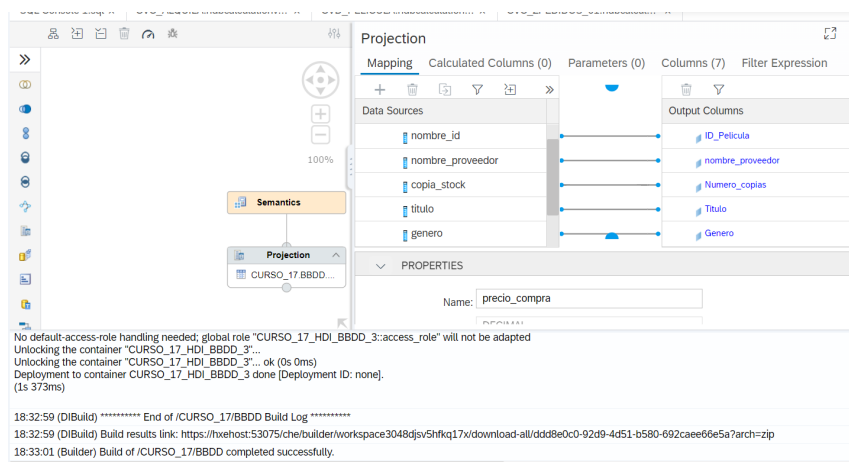
CVD_PELICULA



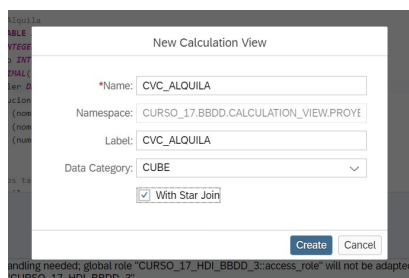
- Vistas tipo cubo

CVC_ALQUILER

Para empezar, en el ejercicio nos pide un sumatorio del precio del alquiler y del precio de compra, por lo tanto, debemos mapearlo en la tabla Película.



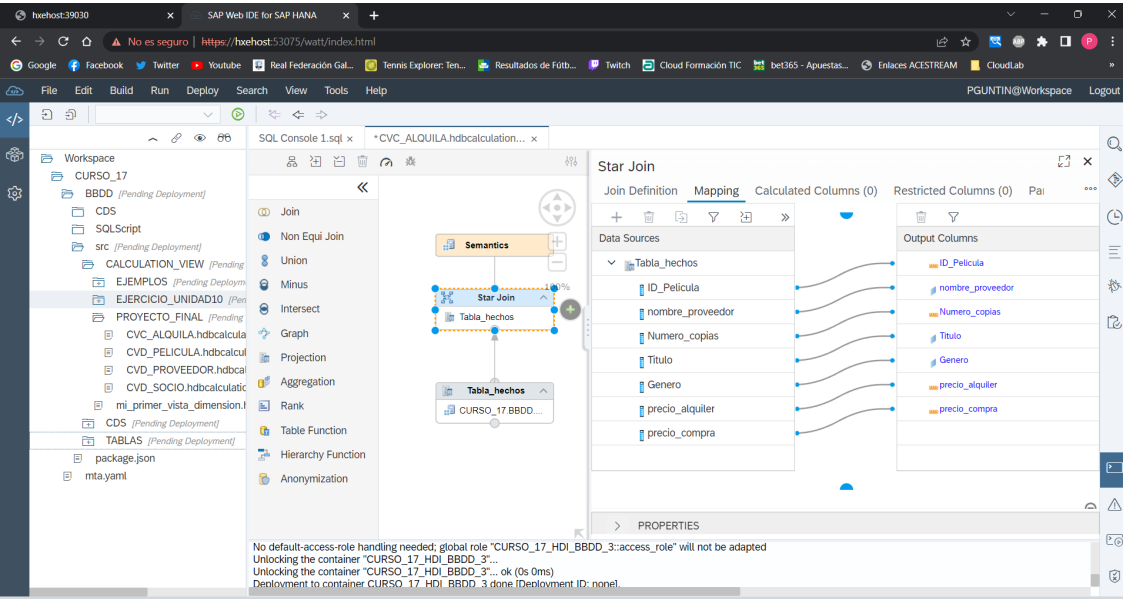
Creamos una vista calculada star join de tipo cubo:



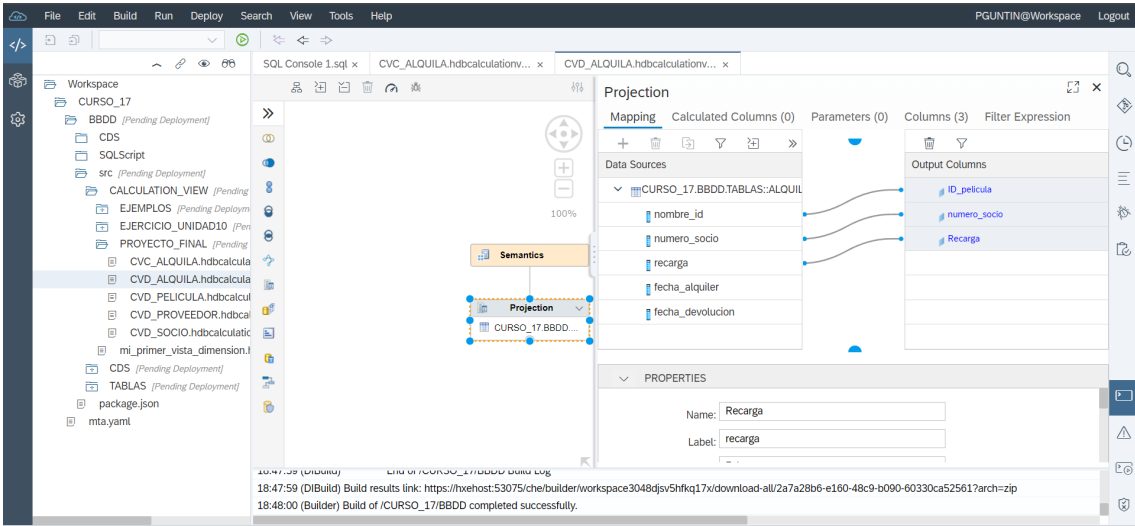
Ahora debemos crear un nodo proyección con la vista calculada de películas para mapear los campos que se nos piden.

Mapeamos todos los campos que teníamos mapeados anteriormente y los nuevos.

Se une la tabla de hechos con nodo star join y mapeamos todos los campos anteriormente mapeados (en el star join).



Creamos también una vista de tipo dimensión de alquiler para utilizar en el star join:



Agregamos la vista calculada de Alquiler al star join:

Find Data Sources

Search for an object name

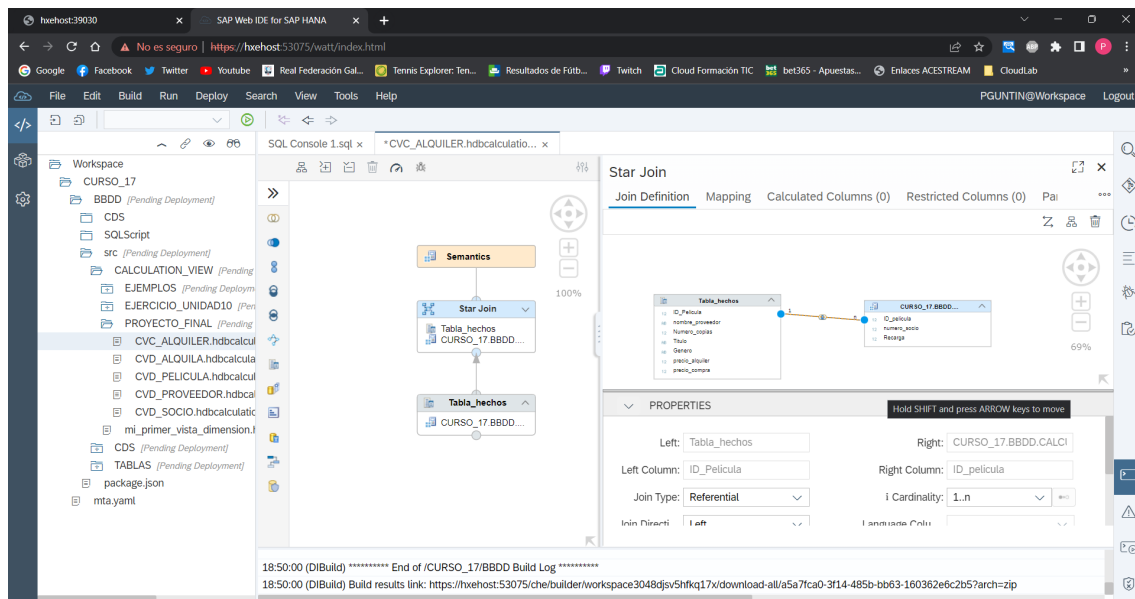
All Types Selected

Results (1) Selected Objects (1)

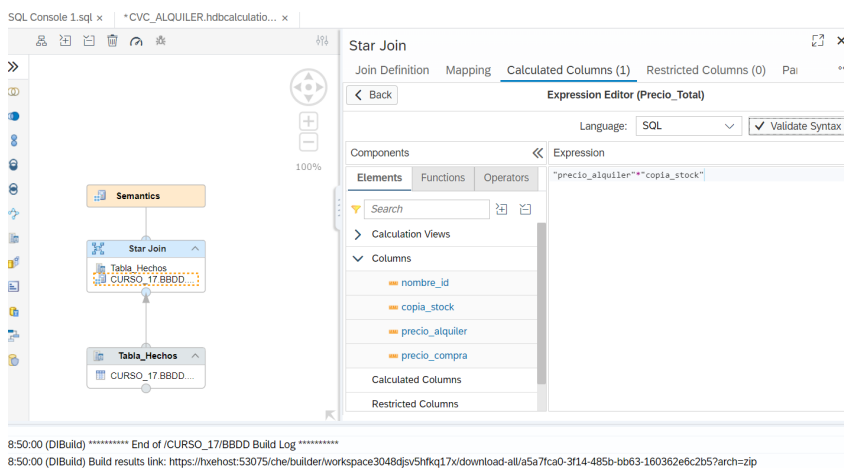
<input checked="" type="checkbox"/>	Type	Name	Schema	Synonym
<input checked="" type="checkbox"/>		CURSO_17_BBDD.CALCUL...	CURSO_17_HDI...	
			CURSO_17_BBDD.CALCULATION_VIEW.PROYECTO_FINAL:CVD_ALQUILA	

Finish Cancel

Hacemos la relación en *Join definition*



En el star join, debemos calcular el precio total (n° copias*precio de compra). Para ello, debemos ir a parameters, en expresion y expresion editor introducimos el código y validamos el texto.



Guardamos y construimos.

Una vez hecho esto, podemos ir a Data preview para visualizar los datos. Para ello, tendríamos que crear tablas .csv (p.e.: proveedor.csv).