MEMORIA

TIENDA DE APLICACIONES DISPOSITIVO MÓVIL

GRUPO L1-3D 5:

Iván Gómez Ruiz
Luis Alberto Martínez Bravo
Edwin Makoveev Routskaia
Pablo Saiz Fernández

ÍNDICE

1.- Descripción del sistema de información, requisitos de información y de procesos del caso proporcionado por los profesores ampliado.

Se quiere poner en marcha una tienda de aplicaciones para un nuevo dispositivo móvil que saldrá al mercado. La información que se necesita es la siguiente.

Los usuarios que no tengan una cuenta necesitarán registrarse para poder acceder. La información requerida es el identificador de usuario, el nombre, el correo electrónico, que es único, la fecha de registro, y contraseña. Un usuario puede iniciar sesión, introduciendo su email y contraseña, **en caso de que el usuario se haya olvidado de su contraseña se le enviará un correo electrónico asociado a su cuenta donde restablecerá su contraseña y se actualizará.**

El usuario puede modificar los datos de su cuenta cuando haya iniciado sesión. Por ejemplo, podrá cambiar el correo electrónico, su nombre, su contraseña, y datos bancarios (tarjetas de crédito). El sistema comprobará que estos datos no se encuentran vacíos y que no coinciden con otros existentes (excepto los datos bancarios, ya que estos sí pueden ser nulos).

Un usuario podrá ser propietario de una aplicación, y una vez que la termine podrá enviarla a la tienda para su revisión. Después de un proceso de revisión, si se considera adecuada, se aprobará para que pueda ser subida y que el usuario/cliente pueda descargarlo en la tienda. Las aplicaciones tienen un nombre (que es único), una descripción, una o varias categorías, fecha de alta y tipo de licencia. Una vez creada una aplicación se le asigna un código interno, que es un identificador único. Las categorías tienen un código y una descripción, y cada aplicación ha de clasificarse al menos en una de ellas.

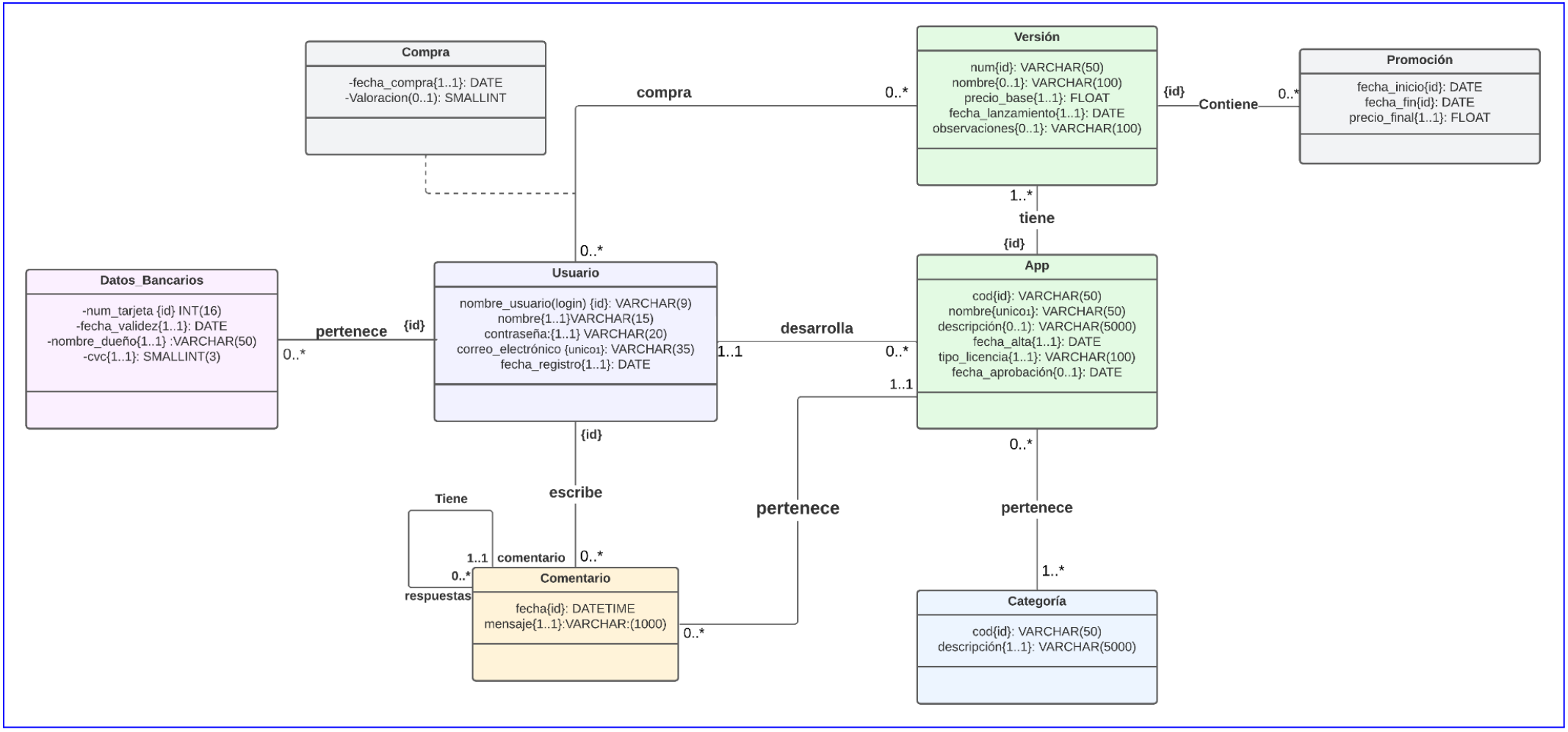
Una vez la aplicación es aprobada, se guarda la fecha, y a partir de entonces su propietario puede ir creando versiones, que son las que los usuarios se descargarán. Cada versión se identifica por un número único dentro de cada aplicación, puede tener un nombre, un precio propuesto por el propietario, una fecha de lanzamiento y unas observaciones, donde se pueden indicar otras particularidades o restricciones de la versión.

Cada vez que un usuario descarga una versión de una aplicación se guarda la fecha. La puede descargar muchas veces, aunque sólo se le cobrará la primera vez (suponiendo que no sea gratuita). De las versiones descargadas el usuario podrá añadir una (y sólo una) valoración entre 1 y 5. Por otra parte, el usuario también puede añadir tantos comentarios como desee sobre las aplicaciones, guardándose para cada uno el instante (fecha y hora), el mensaje, y en caso de que el comentario sea respuesta a otro (de la misma aplicación), también se guardará a qué comentario que se responde. Un comentario se identifica con el identificador del usuario que lo crea y el instante de creación. **Además, una respuesta también puede contener varias respuestas.**

Cuando se compre una aplicación el usuario introducirá su tarjeta de crédito o débito para pagar, en el caso de no tener ninguna tarjeta asociada en su cuenta. En el momento que introduzca estos datos, habrá una opción donde el usuario podrá si lo desea guardar los datos de esa tarjeta de crédito o débito para futuras compras seleccionando la tarjeta existente en el sistema.

Normalmente el importe que el usuario paga por las versiones de las aplicaciones es el propuesto por el propietario, aunque periódicamente la tienda lanza promociones para reducir el precio real de algunas descargas, por lo que interesa tener un histórico de precios de cada versión y así poder obtener el precio en función del momento de la descarga. De esta promoción se necesita saber la fecha de inicio, la fecha final y el precio final a pagar. Una versión sólo puede tener un precio en cierto instante.

1.2- Diagrama de clases diseñado con el LucidChart



1.3- Restricciones de Integridad

1) Una respuesta necesita estar relacionada sí o sí con un comentario existente (id_usuario_r y hora_r deben referenciar a un comentario existente en el sistema).
2) En “Historial” fecha_inicio debe de ser anterior a fecha_fin.
3) Una misma versión solo puede tener una única compra por usuario.
4) Si al comprar una versión NO hay una promoción activa (fecha_compra no se encuentra dentro del rango fecha_inicio-fecha_fin de una promoción para cierta versión), el precio de compra sería el precio base(precio_compra será igual a precio_base).
5) No puede haber más de una promoción activa para una misma versión en un mismo instante.
6) La valoración debe ser un valor comprendido entre 1 y 5

1.4- Consultas

1) Obtener el número de usuarios que se han comprado cierta versión de una aplicación llamada PoliformaT.
2) Obtener la nota media de las valoraciones aportadas por los usuarios a cierta versión de una aplicación llamada PoliformaT.
3) Obtener la valoración media de la primera versión de la aplicación PoliformaT.
4) Obtener la aplicación que más categorías tiene.
5) Obtener el número de usuarios que no han añadido ningún dato bancario.

2- Diseño Lógico

Usuario (nombre_usuario: varchar(9), nombre: varchar(15), contraseña: varchar(20), correo_electronico: varchar(35), fecha_registro: DATE)

- CP:{nombre_usuario}
- VNN:{correo_electronico, nombre, contraseña, fecha_registro}
- Uni:{correo_electronico}

Datos_Bancarios (num_tarjeta: int, fecha_validez: DATE, nombre_dueño: varchar(50), cvc: smallint, nombre_usuario: varchar(9))

- CP:{num_tarjeta, nombre_usuario}
- VNN:{fecha_validez, nombre_dueño, cvc}
- CAj:{nombre_usuario}->Usuario(nombre_usuario)

App (cod: varchar(50), nombre: varchar(50), descripción: varchar (5000), fecha_alta: DATE, tipo_licencia: varchar(100), fecha_aprobacion: DATE, nombre_usuario: varchar(9))

- CP:{cod}
- UNI:{nombre}
- VNN:{fecha_alta, tipo_licencia, nombre_usuario}
- CAj:{nombre_usuario}->Usuario{nombre_Usuario}

Categoría (cod: varchar(50), descripción: varchar(5000))

- CP:{cod}
- VNN:{descripción}

Pertenece (cod1, cod2)

- CP:{cod1, cod2}
- CAj:{cod1}->App{cod}
- CAj:{cod2}->Categoria{cod}

Versión (num: varchar(50), nombre: varchar(100), precio_base: Float, Fecha_lanzamiento: Date observaciones: varchar(100), cod: varchar(50))

- CP:{num, cod}
- VNN:{precio_base, fecha_lanzamiento}
- CAj:{cod}->App{cod}

Promoción (fecha_inicio: DATE, fecha_fin: DATE, precio_final: float, num: varchar(50), cod: VARCHAR(50))

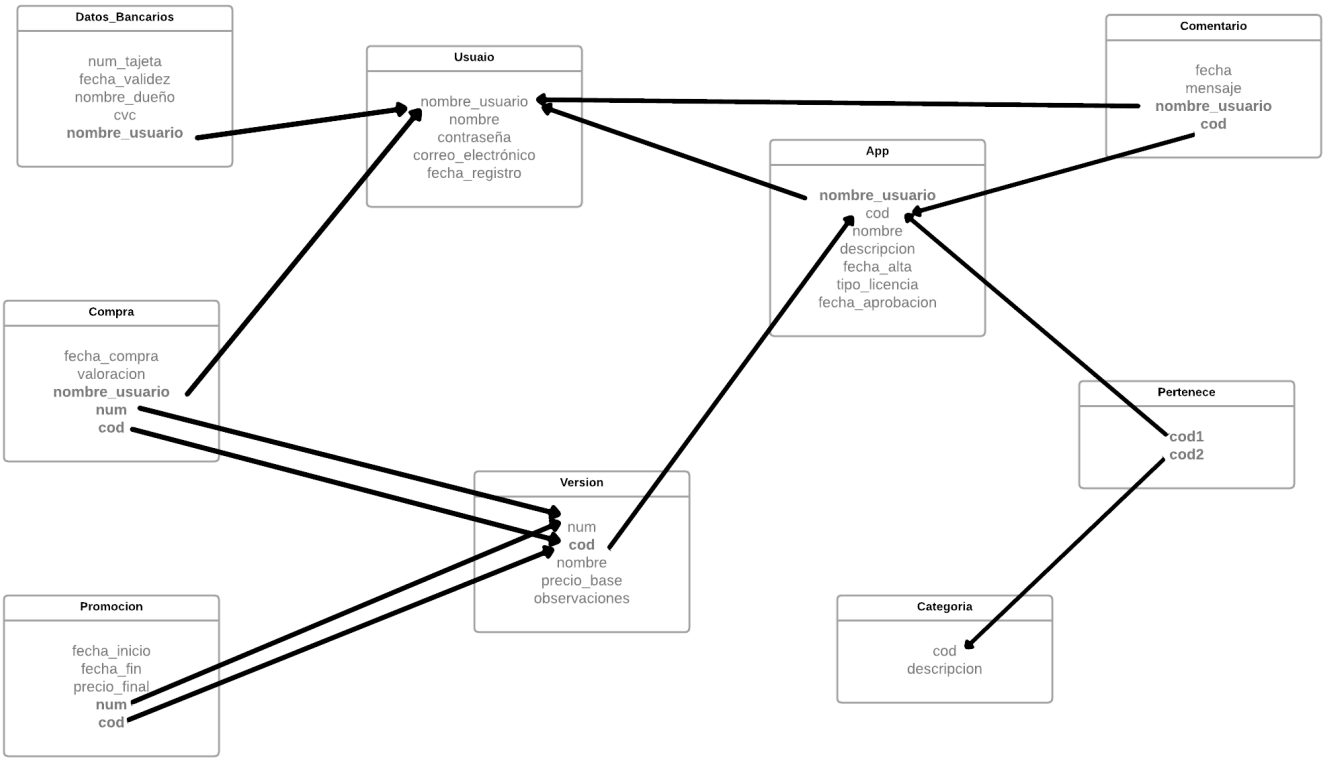
- CP:{fecha_inicio, fecha_fin, num, cod}
- VNN:{precio_final}
- CAj:{num, cod}->Version{num, cod}

Compra (fecha_compra: DATE, valoracion: smallint, num: varchar(50), nombre_usuario: varchar(9), cod: varchar(50))

- CP:{num, nombre_usuario, cod }
- VNN:{fecha_compra}
- CAj:{num, cod}->Version{num, cod}
- CAj:{nombre_usuario}->Usuario{nombre_usuario}

Comentario (fecha: DATETIME, mensaje: varchar(1000), nombre_usuario: varchar(9), cod: varchar(50))

- CP:{fecha, nombre_usuario}
- VNN:{mensaje, cod}
- CAj:{nombre_usuario}->Usuario{nombre_usuario}
- CAj:{cod}->App{cod}
- CAj:(fecha, nombre_usuario)->Comentario{fecha, nombre_usuario}



2.1- Restricciones de Integridad adicionales

- En la tabla “Pertenece” ha de aparecer mínimo una categoría por cada aplicación que haya.
- Una “App” ha de tener siempre una versión asociada.

2.3- 3ª Forma Normal

Este diagrama se encuentra en 3a Forma Normal (3NF) porque cumple con las reglas establecidas en el proceso de normalización de bases de datos. En primer lugar, todas las tablas tienen una clave principal única y todas las columnas están relacionadas directamente con la clave principal, lo que garantiza que no existen dependencias funcionales transitivas. Además, no existen columnas redundantes en ninguna de las tablas, lo que significa que toda la información es necesaria y no se puede calcular a partir de otras columnas en la misma tabla. Esto asegura que el almacenamiento de datos sea eficiente y que no haya problemas de actualización y redundancia de datos. En resumen, este diagrama se encuentra en 3NF porque cumple con las reglas establecidas en el proceso de normalización de bases de datos, garantizando la integridad de los datos, la eficiencia en el almacenamiento y la facilidad de actualización.

2.4- Instrucciones necesarias para dar de alta la base de datos diseñada.

Tabla Usuario

```
CREATE TABLE Usuario (
        nombre_usuario VARCHAR(9),
        nombre VARCHAR(15) NOT NULL,
        contraseña VARCHAR(20) NOT NULL,
        correo_electronico VARCHAR(35) NOT NULL UNIQUE,
        fecha_registro DATE NOT NULL,
        PRIMARY KEY (nombre_usuario)
);
```

Tabla Datos Bancarios

```
CREATE TABLE Datos_Bancarios (
        num_tarjeta INT,
        fecha_validez DATE NOT NULL,
        nombre_dueño VARCHAR(50) NOT NULL,
        cvc SMALLINT NOT NULL,
        nombre_usuario VARCHAR(9) NOT NULL,
        PRIMARY KEY (num_tarjeta, nombre_usuario),
        FOREIGN KEY (nombre_usuario) REFERENCES Usuario(nombre_usuario)
);
```

Tabla App

```
CREATE TABLE App (
        cod VARCHAR(50),
        nombre VARCHAR(50) NOT NULL,
        descripción VARCHAR(5000),
        fecha_alta DATE NOT NULL,
        tipo_licencia VARCHAR(100) NOT NULL,
        fecha_aprobacion DATE,
        nombre_usuario VARCHAR(9) NOT NULL
        PRIMARY KEY (cod),
        CONSTRAINT fk_nombre_usuario FOREIGN KEY (nombre_usuario) REFERENCES
        Usuario(nombre_usuario)

);
```

Tabla Comentario

```
CREATE TABLE Comentario (
        fecha DATE,
        mensaje VARCHAR(1000) NOT NULL,
        nombre_usuario VARCHAR(9),
        cod VARCHAR(50) NOT NULL,
        CONSTRAINT pk_comentario PRIMARY KEY (fecha, nombre_usuario),
        CONSTRAINT fk_usuario FOREIGN KEY (nombre_usuario) REFERENCES
                Usuario(nombre_usuario),
        CONSTRAINT fk_app FOREIGN KEY (cod) REFERENCES App(cod),
        CONSTRAINT fk_comentario FOREIGN KEY (fecha, nombre_usuario) REFERENCES
```

```
                Comentario(fecha, nombre_usuario)
);
```

Tabla Categoría

```
CREATE TABLE Categoria (
        cod VARCHAR(50),
        descripcion VARCHAR(5000) NOT NULL
        PRIMARY KEY (cod)
);
```

Tabla Pertenece

```
CREATE TABLE Pertenece (
        cod1 VARCHAR(50),
        cod2 VARCHAR(50),
        PRIMARY KEY (cod1, cod2),
        FOREIGN KEY (cod1) REFERENCES App(cod),
        FOREIGN KEY (cod2) REFERENCES Categoria(cod)
);
```

Tabla Versión

```
CREATE TABLE Version (
        num VARCHAR(50),
        nombre VARCHAR(100),
        precio_base FLOAT NOT NULL,
        fecha_lanzamiento DATE NOT NULL,
        observaciones VARCHAR(100),
        cod VARCHAR(50),
        PRIMARY KEY (num, cod),
        FOREIGN KEY (cod) REFERENCES App(cod)
);
```

Table Promoción

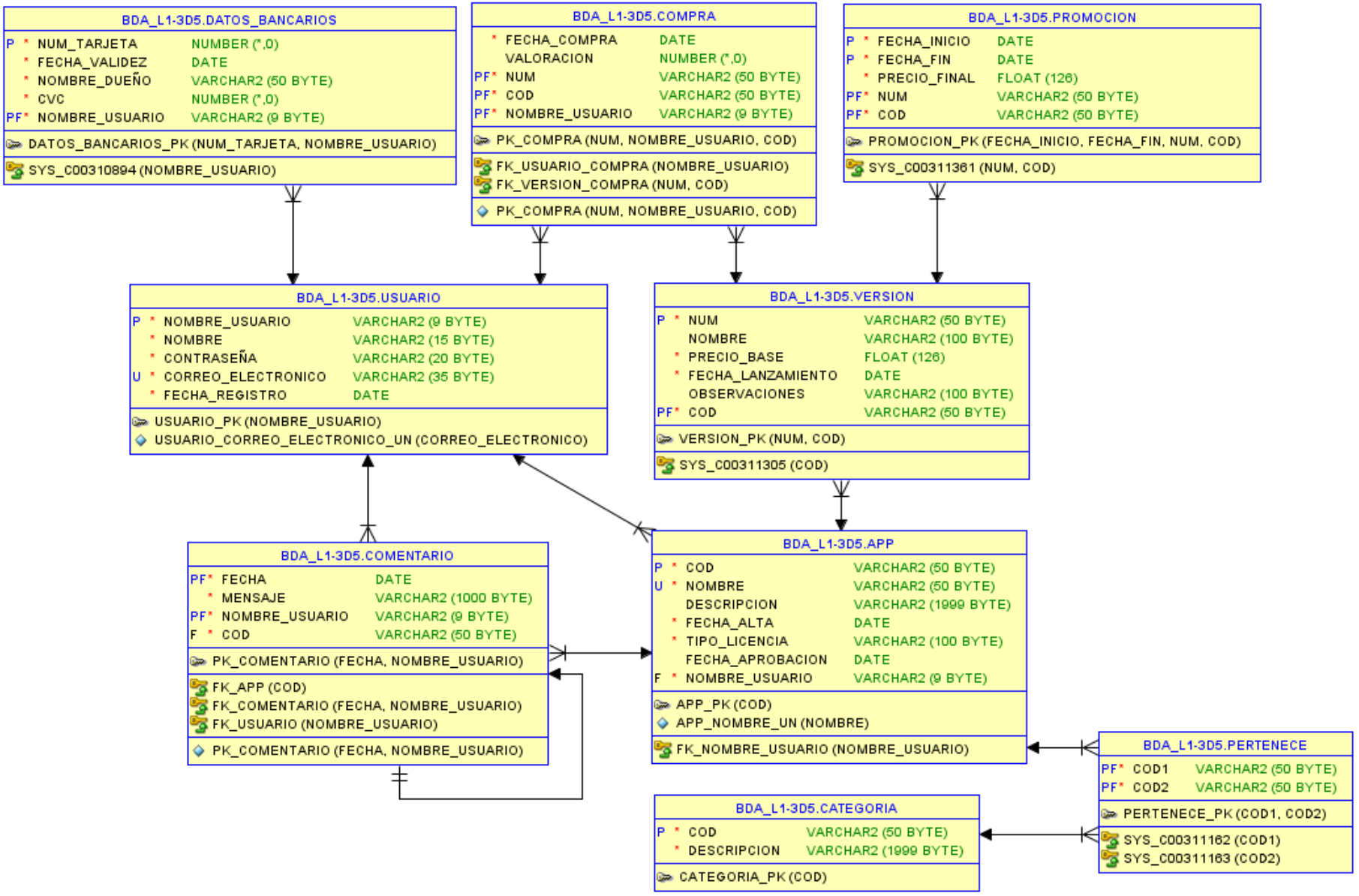
```
CREATE TABLE Promocion (
        fecha_inicio DATE,
        fecha_fin DATE,
        precio_final FLOAT NOT NULL,
        num VARCHAR(50),
        cod VARCHAR(50),
        PRIMARY KEY (fecha_inicio, fecha_fin, num,cod),
        FOREIGN KEY (num, cod) REFERENCES Version(num, cod)
);
```

Tabla Compra

```
CREATE TABLE Compra (
        fecha_compra DATE NOT NULL,
        valoracion SMALLINT,
        num VARCHAR(50),
        cod VARCHAR(50),
        nombre_usuario VARCHAR(9),
        CONSTRAINT pk_compra PRIMARY KEY (num, nombre_usuario, cod),
        CONSTRAINT fk_version_compra FOREIGN KEY (num,cod) REFERENCES Version(num,cod),
        CONSTRAINT fk_usuario_compra FOREIGN KEY (nombre_usuario) REFERENCES
        Usuario(nombre_usuario)

);
```

2.5- Diagrama Generado por Oracle



2.5.1- ¿Cómo se han creado los datos?

Para la creación de los datos se ha utilizado la herramienta que dispone la página web Mockaroo que utiliza “Ruby” como lenguaje de programación y en el que se ha introducido todas las restricciones para que a la hora de generar todos esa información sea coherente con el diseño que se ha realizado en el trabajo. De forma que a la hora de proceder con la inserción masiva sea mucho más sencillo, pues permite exportarlo a diversos formatos compatibles con Oracle.

2.5.2- ¿Cómo se ha realizado la carga de datos?

Asimismo, como se ha comentado anteriormente, a partir de un fichero csv hemos cogido los datos y los hemos separado, para introducirlo a la base de datos mediante el gestor de base de datos, de forma que la tabla usuario tiene nombre_usuario, nombre, contraseña, correo_electronico, fecha_registro, entonces se seleccionan esas columnas del fichero csv y se ponen en nuestra base de datos, creando asi la tabla usuario, asi sucesivamente con todas las tablas.

Cabe recalcar, a la hora de realizar la inserción hubo ciertos problemas, como que un nombre_usuario tiene como maximo 9 caracteres, pero en la base de datos generada existian algunos datos que tenia más de 9 caracteres, por lo que tuvimos que modificar esos datos para que cumplieran con las restricciones.