Práctica 05

Lenguaje de Definición de Datos (DDL)

1. Objetivo General

Construir el esquema de una base de datos utilizando el Lenguaje Estructurado de Consultas (SQL por sus siglas en inglés).

2. Objetivos Secundarios

- Conocer la sintaxis y los parámetros del Lenguaje de Definición de Datos (DDL por sus siglas en inglés).
- Transformar un diagrama de Clases en el esquema de una base de datos.

3. Introducción

Las bases de datos relacionales están conformadas por una colección de tablas o *relaciones*. Para poder construir estas tablas o relaciones se necesita primero crear a la base que las contendrá. La base de datos junto con las tablas que la componen serán creadas y administradas con ayuda de un SMBD, en este caso PosgtreSQL tal como se trabajó en *Sistemas Manejadores de Bases de Datos* (Práctica04), y un lenguaje de programación de propósito específico, SQL.

En esta práctica se trabajará por una parte con SQL, uno de los primeros lenguajes comerciales y fue propuesto por Donald Chamberlin y Raymond Boyce. Por otra parte el modelo relacional creado por Edgar F. Codd y que el mismo describió como "Un modelo relacional de datos para grandes bancos de datos compartidos". A pesar de no apegarse íntegramente al modelo relacional tal como lo describe Codd , SQL se convirtió en el lenguaje de bases de datos más utilizado.

SQL se convirtió en un estándar de la American National Standards Institute (ANSI) en 1986 y de la Organización Internacional de Normalización (ISO) en 1987. Desde entonces, el estándar se ha mejorado varias veces con características añadidas pero aún no es completamente transferible entre sistemas de bases de datos diferentes, es decir, cada SMBD tiene una combinación específica del código, ya que los diferentes fabricantes no siguen estrictamente el estándar al añadir extensiones.

SQL fue originalmente basado en el Álgebra Relacional y el Cálculo Relacional de tuplas. Está constituido por el siguiente conjunto de sublenguajes:

- Lenguaje de definición de datos (DDL)
- Lenguaje de manipulación de datos (DML)
- Lenguaje de control (DCL)
- Lenguaje de transacciones (TCL)

El alcance de SQL incluye aspectos como la inserción de datos, consulta, actualización y supresión, creación y modificación de esquemas, y el control de acceso a datos. Para ésta práctica revisaremos la parte correspondiente a la definición de datos, es decir el DDL.

3.1. Creación de la base de datos

Para crear una base de datos sobre la cual construiremos las tablas, se necesitan algunas palabras reservadas de SQL. En la Figura 5.1 se presenta la sintaxis utilizada para la creación de una base de datos. Para fines didácticos, y a partir de esta práctica, se utilizarán mayúsculas al escribir aquellas palabras reservadas pertenecientes a SQL.

```
CREATE DATABASE mi_primera_base;
```

Figura 5.1 - Ejemplo básico de la sintaxis para creación de una base de datos.

Esta sintaxis está compuesta de diferentes partes:

1. CREATE DATABASE
   Palabra reservada que se utiliza cada vez que se necesite crear una base de datos.
2. mi_primera_base
   El nombre de la base de datos que se desea crear, es elegido por el diseñador de la base de datos. Toda nueva base que se cree dentro de un SMBD tendrá entonces un nombre diferente a los ya existentes.
3. ;
   Indica la finalización de la instrucción.

Lo anterior corresponde a la sintaxis básica y mínima necesaria para crear una base de datos. Sin embargo el DDL permite definir características avanzadas sobre una base de datos, tal como se muestra en la Figura 5.2.

```
CREATE DATABASE mi_primera_BD

WITH
        OWNER = postgres
        ENCODING = 'UTF8'
        TABLESPACE = pg_default
        CONNECTION LIMIT = -1
        ;
```

Figura 5.2 - Ejemplo avanzado de la sintaxis para creación de una base de datos.

La sintaxis anterior se explica a continuación:

1. OWNER
   Indica el nombre del usuario que será el propietario de la base de datos creada. En caso de no definirla o que se escriba la opción DEFAULT, el SMBD asignará como OWNER al usuario que ejecuta el comando.
2. ENCODING
   Conjunto de caracteres para utilizar en la base de datos creada. Se puede especificar una cadena constante, por ejemplo 'SQL_ASCII', un número entero de codificación o la opción DEFAULT para utilizar la codificación por defecto
   Otro ejemplo de codificación es: ISO 8859-15. Puede usarse para representar el alfabeto y otros caracteres importantes para almacenar textos en inglés, francés, alemán, español y portugués, entre otros idiomas de Europa occidental.
3. TABLESPACE
   Indica la ruta del equipo donde se guardarán los archivos de información de las bases de datos.
4. CONNECTION LIMIT
   Permite establecer cuántas conexiones simultáneas se pueden hacer a la base de datos creada. El número -1, escrito por defecto, significa que no hay límite.

Hay que tomar en cuenta que si no se personalizan los parámetros de las opciones de la base de datos, el SMBD aplicará la configuración por defecto.

3.2. Creación de tablas

Una vez que la base de datos ha sido creada, se procederá a construir las tablas necesarias, y una vez más, a través de SQL.

La construcción de las tablas requiere de comandos o palabras reservadas, además de variables y constantes en un determinado orden para que el SMBD lo ejecute correctamente. En la Figura 5.3 se muestra la sintaxis básica para crear una tabla.

```
CREATE TABLE mi_primera_tabla
(
        primera_columna   numeric,
        segunda_columna   text
);
```

Figura 5.3 - Ejemplo básico de la sintaxis para creación de tablas.

Las partes que componen la sintaxis anterior se describen a continuación:

1. CREATE TABLE
   Palabra reservada que se utilizará cada vez que se necesite crear una nueva tabla.
2. mi_primera_tabla

El nombre de la tabla que se quiere crear, es elegido por el diseñador de la base de datos. Toda nueva tabla que se cree dentro de un SMBD tendrá entonces un nombre diferente a los ya existentes.

3. (

   El símbolo de apertura de paréntesis, después del nombre de la tabla, sirve para indicar que comenzará con la descripción de las columnas de esa tabla.

4. primera_columna

   El nombre de la columna deberá describir el dato que se almacenará en ella. Este nombre lo elige el diseñador de la base de datos y cada columna deberá tener un nombre único dentro de la tabla, no así en toda la base de datos. Se aconseja evitar repetir nombres de columnas dentro de la misma base de datos. De ser necesario se recomienda utilizar la siguiente notación: *nombreColumna_nombreTabla*.

5. numeric

   Indica la naturaleza del dato que se va a almacenar, es decir, el tipo del dato. PostgreSQL soporta varios tipos de datos, como se presentó durante el *Modelado con Diagramas de Clases* (Práctica 03).

6. ,

   La coma, actúa como separador de columnas, indica que la descripción del nombre, tipo y otras características de la columna ha finalizado. Después de la coma se puede introducir la descripción de otra columna siguiendo el mismo procedimiento de los pasos 4 y 5.

7.)

   El símbolo de cierre de paréntesis, después de la descripción de la última columna que integrará la tabla, indica al SMBD que ha terminado la definición de tabla.

8. ;

   Indica la finalización de la instrucción.

SQL tiene la capacidad de personalizar ciertas características de las tablas para cumplir con las necesidades del diseñador. En la Figura 5.4 se muestra un ejemplo donde se declara un número mayor de columnas con diferentes tipos de datos.

```
CREATE TABLE cliente
(
        id_unico SERIAL,
        nombre_cliente VARCHAR(40),
        ap_paterno_cliente VARCHAR(40),
        ap_materno_cliente VARCHAR(40),
        clasificacion_de_cliente BOOLEAN,
        status_activacion BOOLEAN,
        fecha_ingreso DATE
);
```

Figura 5.4 - Ejemplo avanzado de la sintaxis para creación de tablas.

La característica de *character varying (40)* expresa el tipo de la columna y la longitud máxima del dato. Es decir, que en esa columna tendremos la posibilidad de guardar cadenas de caracteres con una longitud variable pero máxima de 40 posiciones. Así como ésta restricción, existen algunos tipos de datos que requieren ciertos parámetros para determinar con mayor exactitud el tipo de dato que recibirán. La lista de los tipos de datos y sus características se definió durante el *Modelado con Diagramas de Clases* (Práctica 03).

4. Ejemplo de la pizzería

Recordando lo trabajado en *Modelado con Diagramas de Clases* (Práctica 03), se obtuvo el diagrama de Clases para la Pizzería. Ver Figura 5.5.

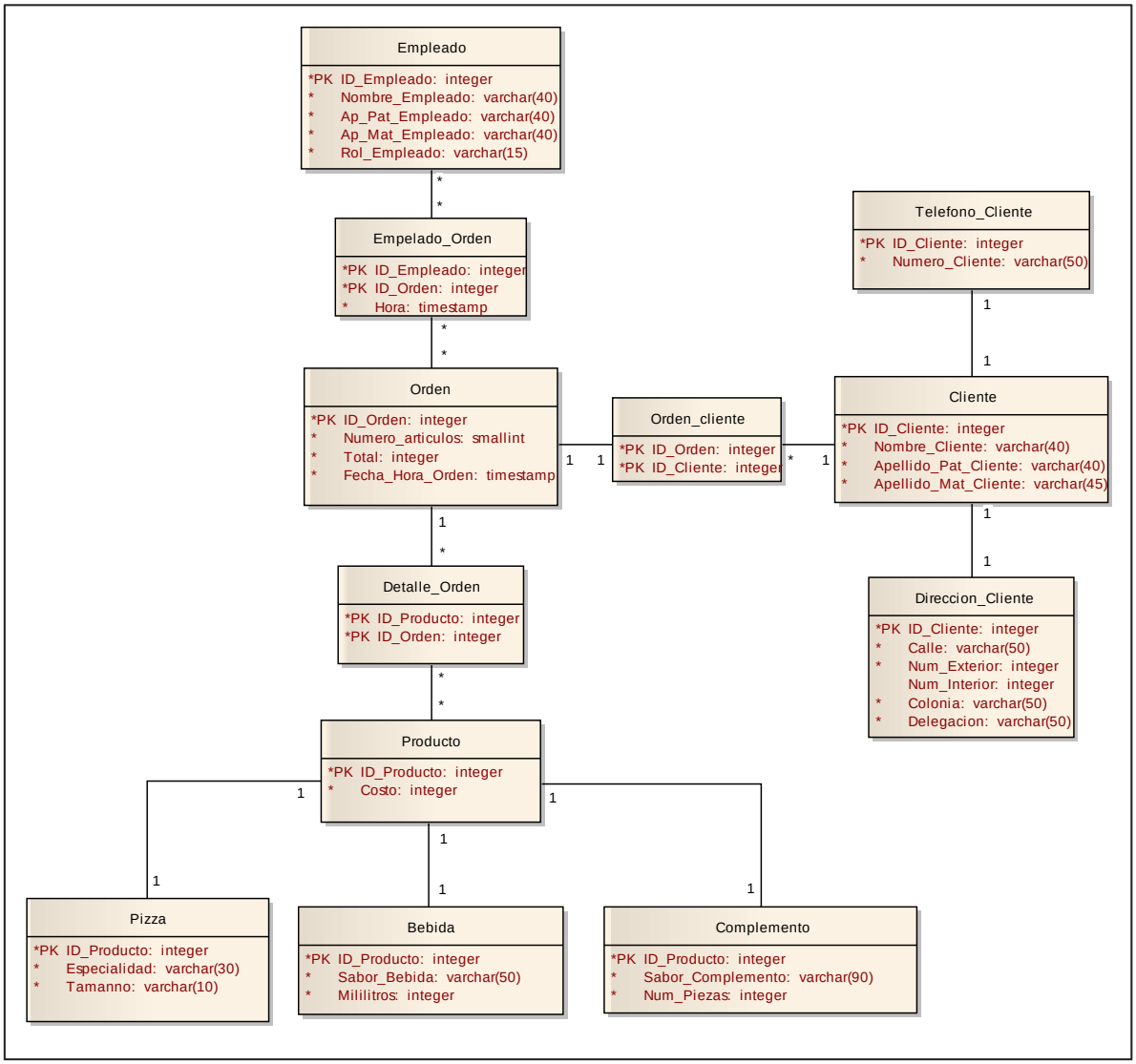

Figura 5.5 - Diagrama de Clases de la Pizzería.

Como se revisó en el punto 3 de ésta práctica, para comenzar la construcción del esquema asociado a este diagrama, el primer paso será crear la base de datos en la que se cargarán todas las tablas. El código para lograr esto es el siguiente:

```
CREATE DATABASE pizzeria;
```

Una vez creada la base de datos, ingresaremos a ella ya sea mediante la consola de PostgreSQL o seleccionándola dentro de la pantalla principal de pgAdminIII. Al estar dentro de ésta, procederemos a construir cada una de sus tablas.

A continuación se muestra la sintaxis asociada a la clase Pizza que se convertirá en la tabla Pizza. Ver Figura 5.6.

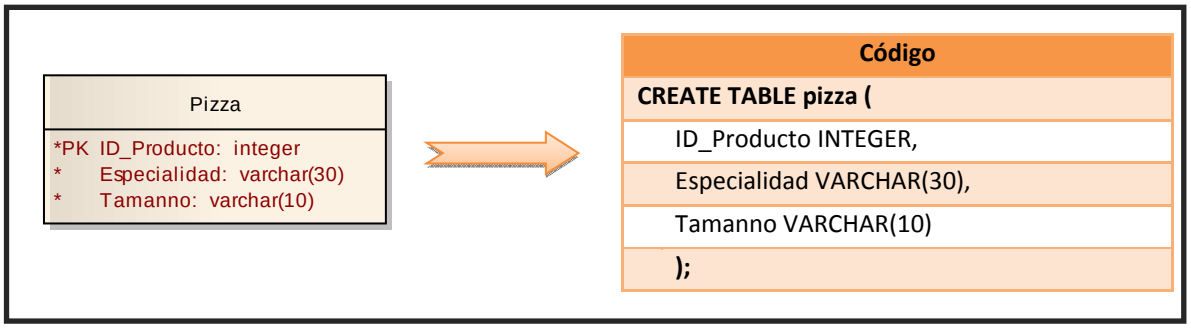

Figura 5.6 - Conversión de una clase a una tabla.

En la Figura 5.7 se muestra el código de cada una de las tablas asociadas a nuestra base de datos. Dicho código fue construido de la misma manera como se hizo con la tabla Pizza.

```
CREATE TABLE Producto (
                ID_Producto INTEGER,
                Costo INTEGER);
CREATE TABLE Bebida (
                ID_Producto INTEGER,
                Sabor_Bebida VARCHAR(50),
                Mililitros INTEGER);
CREATE TABLE Complemento (
                ID_Producto INTEGER,
                Sabor_Complemento VARCHAR(90),
                Num_Piezas INTEGER);
CREATE TABLE Pizza (
                ID_Producto INTEGER,
                Especialidad VARCHAR(35),
                Tammano VARCHAR(10));
CREATE TABLE Orden (
                ID_Orden INTEGER,
                Numero_Articulos SMALLINT,
                Total INTEGER);
CREATE TABLE Detalle_orden (
                ID_Producto INTEGER,
                ID_Orden INTEGER);
CREATE TABLE Empleado (
                ID_Empleado INTEGER,
                Nombre_Empleado VARCHAR(40),
                Ap_Pat_Empleado VARCHAR (40),
                Ap_Mat_Empleado VARCHAR (45),
                Area_Trabajo VARCHAR(15));
CREATE TABLE Empleado_Orden (
                ID_Empleado INTEGER,
                ID_Orden INTEGER,
                Hora TIMESTAMP);
CREATE TABLE Cliente (
                ID_Cliente INTEGER,
                Nombre_Cliente VARCHAR(40),
                Apellido_Pat_Cliente VARCHAR(40),
                Apellido_Mat_Cliente VARCHAR(45));
CREATE TABLE Orden_Cliente (
                ID_Orden INTEGER,
                ID_Cliente INTEGER);
CREATE TABLE Telefono_Cliente (
                ID_Orden INTEGER,
                ID_Cliente VARCHAR(20));
CREATE TABLE Direccion_cliente (
                ID_Cliente INTEGER,
                Calle VARCHAR(60),
                Num_Exterior INTEGER ,
                Num_Interior INTEGER,
                Colonia VARCHAR(70),
                Delegacion VARCHAR (65));
```

Figura 5.7 - Código DDL completo para la Pizzería.

Como podemos apreciar no se utilizan acentos ni caracteres especiales como la letra ñ, ya que el SMBD no los interpreta con precisión, lo cual puede ocasionar errores que al final se reflejarán en la eficiencia de la base de datos.

5. Ejercicios

(NOTA: Resuelve los siguientes ejercicios en relación al proyecto que realizarás durante el curso, en dado caso que no tengas un proyecto, utiliza la información en el apéndice NFL-ONEFA parte 05 al final de esta práctica para realizarlos)

1. Crea la base de datos de tu proyecto.
2. Utilizando código DDL, construye las tablas que integrarán el **Esquema la base de datos** de acuerdo al diagrama de Clases de tu proyecto.
3. Genera un listado en donde describas el nombre de cada tabla y lo que representa cada una.
4. Dentro del listado del punto 3, agrega para cada tabla a todos los atributos que la conforman, lo que representa cada uno de ellos y el tipo de dato asociado a éstos.

Entregables requeridos para prácticas subsecuentes:

- Esquema de la base de datos

6. Apéndice NFL-ONEFA parte 05

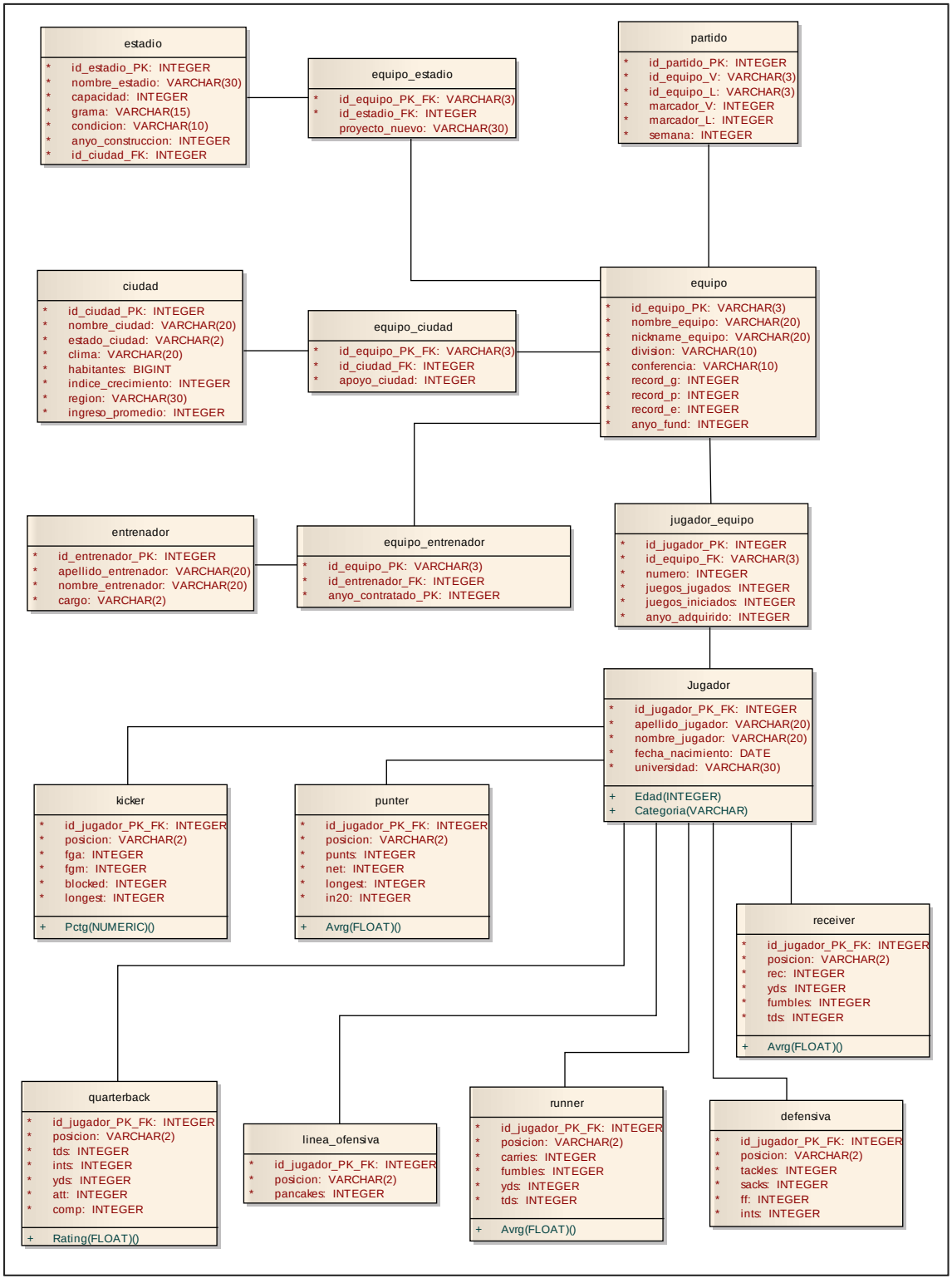

Figura 5.7 - Diagrama de Clases UML del proyecto NFL-ONEFA