UNIVERSIDAD DEL VALLE

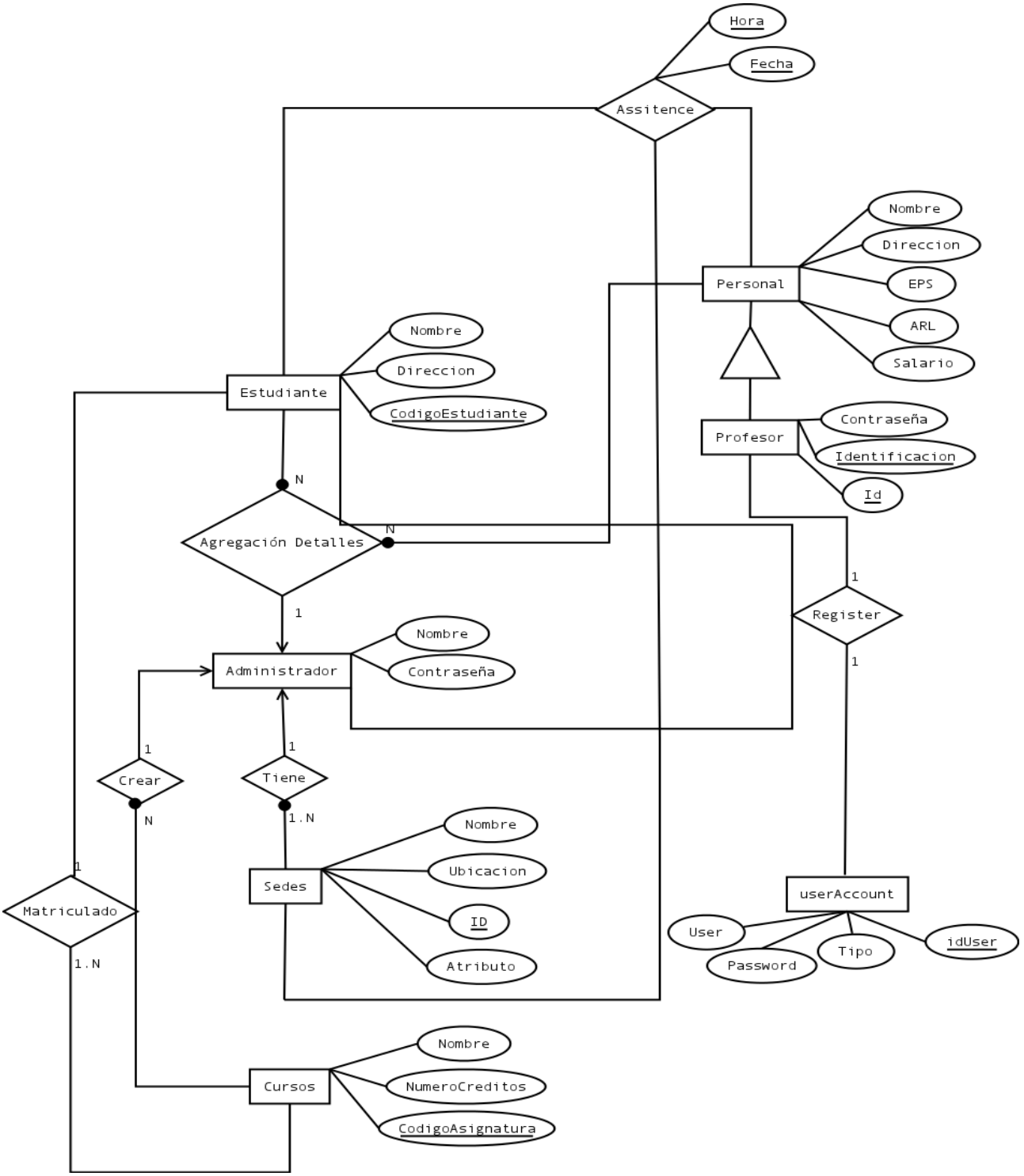
FACULTAD DE INGENIERÍA

Tec Desarrollo De Software

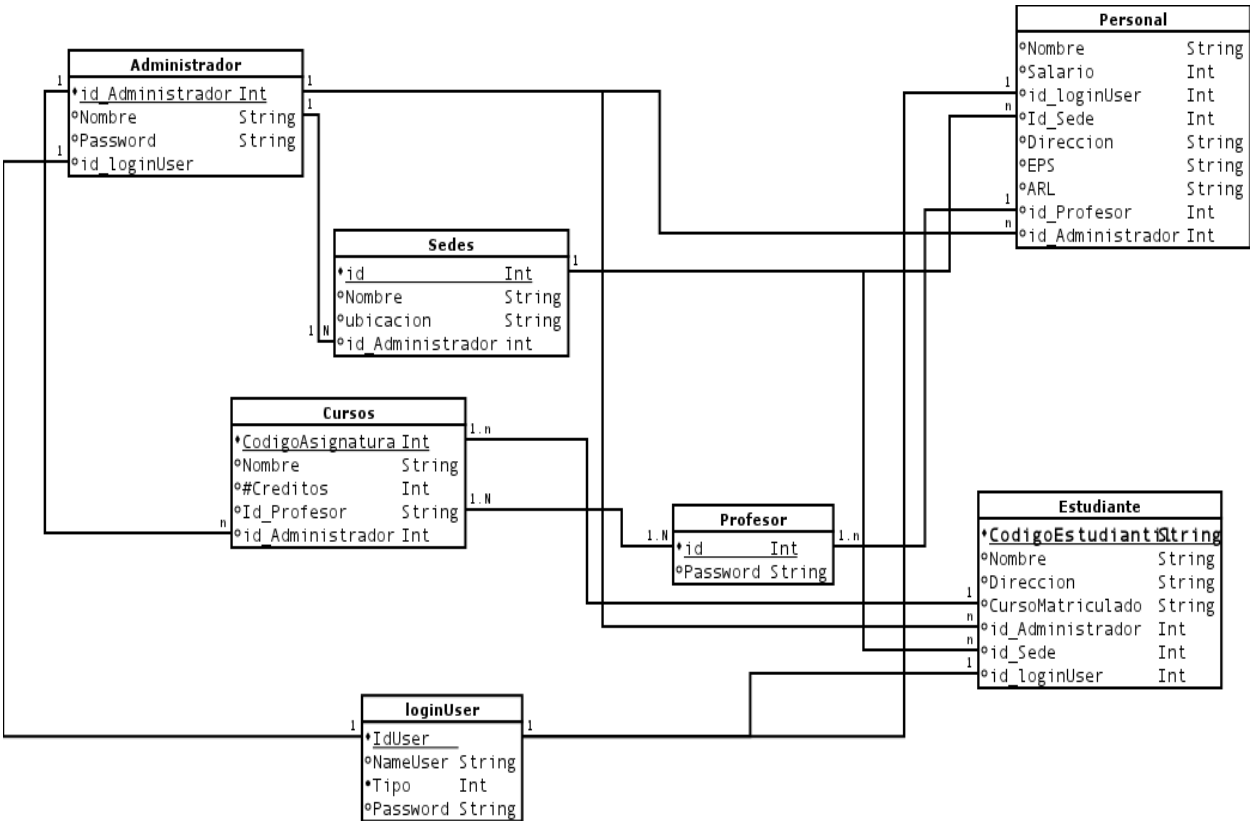
Docente: Jefferson A. Peña Torres

Estudiante: Juan Felipe Osorio Zapata

1.) Crear un diseño utilizando el modelo entidad Relación (ER) con notación Chen o Chen Extendida.



2.) Crear un diseño utilizando el modelo Relacional utilizando como base el diseño ER.



3.) Crear los archivos con las instrucciones SQL para llevar sus diseños a PostgreSQL; (en la carpeta que adjunto del parcial envió todos los archivos SQL, aquí solo son screenshots).

```
COPY public.personal (nombre, salario, direccion, eps, arl, id_loginuser, id_sede, id_profesor, id_admin) FROM stdin;
Carlos Mario C  3500000 Av 4ta carrera 12 calle 23  SURA    Colmena Seguros 1   2   1   2
Enrique Castrillon  3500000 Km 3 vda las palmas casa 129    Comfenalco  Colmena Seguros 2   3   2   3
Maria Camila    3500000 Av 4ta carrera 6 calle 43   SURA    Colmena Seguros 3   4   3   4
Jefferson Amado Pena    3500000 Cra 12A #23-32 Calle 45 Porvenir    Comfenalco  Colmena Seguros 4   5   4   5
\.


--
-- Data for Name: profesor; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.profesor (idprofesor, password) FROM stdin;
1   profe
2   Contraprofe123
3   colombiaTierraquerida20
4   Ajedrez_manzana23
\.


--
-- Data for Name: sedes; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.sedes (idsedes, nombre, ubicacion, id_admin) FROM stdin;
2   Melendez    Sur - Cali  2
3   San fernando    Sur - Cali  3
4   Melendez    Sur - Cali  4
5   Melendez    Sur - Cali  5
\.
```

```
ALTER TABLE ONLY public.administrador
    ADD CONSTRAINT administrador_id_loginuser_fkey FOREIGN KEY (id_loginuser) REFERENCES public.loginuser(iduser);


--
-- Name: cursos cursos_id_admin_fkey; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.cursos
    ADD CONSTRAINT cursos_id_admin_fkey FOREIGN KEY (id_admin) REFERENCES public.administrador(id_administrador);


--
-- Name: cursos cursos_id_profesor_fkey; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.cursos
    ADD CONSTRAINT cursos_id_profesor_fkey FOREIGN KEY (id_profesor) REFERENCES public.profesor(idprofesor);


--
-- Name: estudiante estudiante_id_admin_fkey; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.estudiante
    ADD CONSTRAINT estudiante_id_admin_fkey FOREIGN KEY (id_admin) REFERENCES public.administrador(id_administrador);

```

4.) Encuentre las dependencias funcionales de cada una de sus tablas.

- Tabla Profesor: En esta tabla encuentro únicos valores SERIAL del id de profesor el cual identifica a cada profesor con su respectiva contraseña que es el otro atributo de la tabla 'Profesor'.

```
attendance=# select * from profesor;
 idprofesor |        password
------------+------------------------
          1 | profe
          2 | Contraprofe123
          3 | colombiaTierraquerida20
          4 | Ajedrez_manzana23
```

- Tabla Login User: En esta tabla se registran los logins de los usuarios que han entrado a la herramienta Attendance mediante un inicio de sesión web en el cual se informa dinámicamente que hay tres tipos de usuarios numerados del 1-3.

  A. Tipo 1: Es el usuario administrador.
  B. Tipo 2: Es el usuario Profesor.
  C. Tipo 3: Es el usuario Estudiante.

adicionalmente después de este proceso, se digitan unos

nombres de usuarios y una contraseña única en cada registro de
cada usuario para sus futuros logins dentro de la herramienta,

cada uno siendo atributos funcionalmente dependiente uno del otro, un usuario no puede tener varias contraseñas y
La contraseña no puede ser de varios usuarios.

```
attendance=# select * from loginUser;
 iduser |   nombre   | tipo |    password
--------+------------+------+----------------
      3 | Cami23     |    3 | realMadrid2005
      4 | Japeto10   |    2 | JapetoA
      1 | CarlosM97  |    1 | CarlosM21
      2 | EnriqueC10 |    2 | ContraProfe123
```

- Tabla Administrador: En esta tabla se puede evidenciar el registro de datos de usuarios que han ingresado a la herramienta 'Attendance' como tipo de usuario 1, porque son administradores, con un nombre de administrador y una contraseña que se establece después de poner los datos y el username, todos relacionados entre sí dependientes funcionalmente.

```
attendance=# select * from administrador;
 id_administrador |      nombre       |    password     | tipo_user
------------------+-------------------+-----------------+-----------
                2 | Carlos Mario      | CarlosM21       |         1
                3 | Fernando Mauricio | FernanCross1998 |         1
                4 | Maria Camila      | Cam12           |         1
                5 | Paola Diaz        | Pao502          |         1
```

- Tabla Sedes: En esta tabla se registran los datos de las diversas sedes que existen para el registro de asistencia de tanto docentes, estudiantes, como personal administrativo, misional y no misional, abarcando datos de ubicación geográfica dentro de la ciudad, nombre de la sede y cuál administrador está a cargo, teniendo en cuenta que un administrador puede tener varias sedes, en este caso la dependencia funcional está presente con todos los datos bidireccionalmente, el administrador 2 está en dos sedes tanto (Melendez como Pance).

```
attendance=# select * from sedes;
 idsedes |    nombre    | ubicacion  | id_admin
---------+--------------+------------+----------
       2 | Melendez     | Sur - Cali |        2
       3 | San fernando | Sur - Cali |        3
       5 | Farallones   | Sur - Cali |        5
       4 | Centro       | Sur - Cali |        4
       6 | Pance        | Sur - Cali |        2
```

- Tabla Estudiante: En esta tabla se evidencia los datos guardados de los estudiantes que se han registrado y han accedido a la herramienta Attendance, con el nombre del estudiante, dirección de residencia, código del programa estudiantil, y como super claves id de administrador que puede agregar detalles del estudiante, una sede a la cual asiste y ve los programas académicos, y por último un id de usuario que hace referencia a la tabla loginUser con el tipo de usuario 3, que es para estudiante.

```
attendance=# select * from estudiante;
 idestudiante |        nombre         |            direccion             | cursomatriculado |
id_admin | id_sede | id_loginuser
--------------+-----------------------+----------------------------------+------------------+-
---------+---------+--------------
            4 | Maria Camila          | AV 6ta - calle 24N barrio campina | 710193           |
       2 |       2 |            3
            5 | Carlos Andres Bello   | Calle 23 Nte - casa #124          | 750030           |
       3 |       3 |            3
            6 | René martinez Dubeq   | KM 3 corregimiento la castilla    | 750091           |
       4 |       4 |            3
            7 | Juan Fernando Sanchez | Calle 5ta norte - Mira Flores     | 730069           |
       5 |       5 |            3
```

- Tabla Cursos: En esta tabla se evidencia los cursos que está presentes y quienes dictan los cursos, con un nombre de curso, y un id que hace referencia a la tabla de profesores con cada profesor que enseña cada curso, un curso no lo enseñan varios profesores; por último un id de administrador para cada curso.

```
attendance=# select * from cursos;
 id_curso |        nombre         | num_creditos | id_profesor | id_admin
----------+-----------------------+--------------+-------------+----------
        1 | Arquitectura C II     |            3 |           1 |        2
        2 | Bases Datos I         |            4 |           2 |        3
        3 | Desarrollo S I        |            3 |           3 |        4
        4 | Constitucion Politica |            2 |           4 |        5
```

- Tabla Personal: En esta tabla se muestran los registros de datos de información tanto de profesores, como estudiantes y administradores, llevando a cabo los registros dependientes funcionalmente uno de otro, como nombre Personal, dirección, salario, eps, arl, y los id's foráneos de las tablas: loginUser que hace referencia al id al cual hace parte del login que hicieron en la herramienta Attendance, Sedes que hace referencia a las sedes de las cuales hacen parte, el id de profesor el cual hace referencia al profesor que está dentro del personal o el administrador y de cuál profe está bajo su mando, por último el id de administrador.

```
attendance=# select * from personal;
       nombre        | salario |           direccion            |    eps    |       arl       | id_loginuser | id_sede | id_profesor | id_admin
---------------------+---------+--------------------------------+-----------+-----------------+--------------+---------+-------------+----------
 Enrique Castrillon  | 3500000 | Km 3 vda las palmas casa 129   | Comfenalco | Colmena Seguros |            2 |       3 |           2 |        3
 Jefferson Amado Pena | 3500000 | Cra 12A #23-32 Calle 45 Porvenir | Comfenalco | Colmena Seguros |            4 |       5 |           4 |        5
 Carlos Mario C      | 4230000 | Av 4ta carrera 12 calle 23     | SURA       | Colmena Seguros |            1 |       2 |           1 |        2
```

5.) Describa porque su diseño se encuentra normalizado:

El diseño que cree, se encuentra normalizado, porque los datos que se integran, se han instanciado de forma precisa y clara en otras tablas sin tener un factor redundante apuntando a la optimización de los datos que se guardan para a posteriori poder acceder a ellos de la mejor forma posible y sin fallas o inconsistencias, además de eso, se tomó el proceso ilustrado por el curso en lo que se lleva del mismo, diseñando y creando primero de antemano los modelos teóricos lógicos de nuestra base de datos, empezando por un modelo Entidad-Relación, para después pasar al modelo relacional y con base a eso poder empezar a diseñar la base de datos con PostgreSQL desde el container en Docker.

6.) Despliegue utilizando la versión contenerizada de PostgreSQL docker.

- Cabe aclarar que esto solo son muestras de screenshots, en el enlace que dejo a continuación está todo detallado.
  Enlace: https://www.youtube.com/watch?v=X9WYaqWcnXc

- Primero se accede como usuario postgres, con el comando 'psql -U postgres.

```
root@adf422b1e45e:/# bash
root@adf422b1e45e:/# psql -U postgres
```

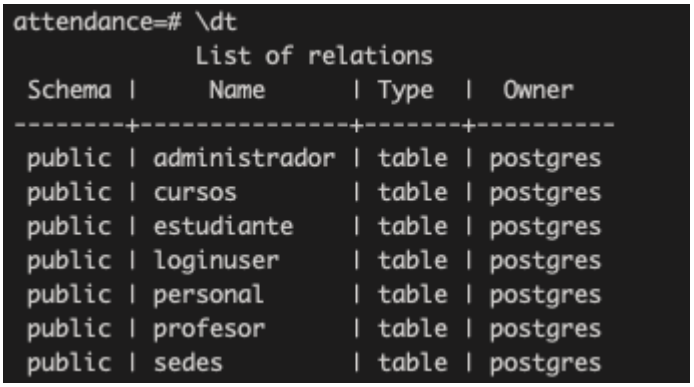
- Segundo se despliegan las bases de datos existentes con el comando '\l' , para poder ver nuestra base de datos contenerizada que es la que está subrayada en azul.

```
postgres=# \l
                                  List of databases
   Name    |  Owner   | Encoding |  Collate   |   Ctype    |   Access privileges
-----------+----------+----------+------------+------------+-----------------------
 attendance | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
 postgres   | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
 template0  | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
            |          |          |            |            | postgres=CTc/postgres
 template1  | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres          +
            |          |          |            |            | postgres=CTc/postgres
```

- Tercero se accede a la base de datos, con el comando '\c <Nombre Base Datos>'.

```
postgres=# \c attendance
You are now connected to database "attendance" as user "postgres".
```

- Cuarto y último, se enlistan las tablas que tengamos creadas en nuestra base de datos, con el comando '\dt', y eso es todo.



```
attendance=# \dt
             List of relations
 Schema |     Name      | Type  |  Owner
--------+---------------+-------+----------
 public | administrador | table | postgres
 public | cursos        | table | postgres
 public | estudiante    | table | postgres
 public | loginuser     | table | postgres
 public | personal      | table | postgres
 public | profesor      | table | postgres
 public | sedes         | table | postgres
```

7.) Todos los archivos relacionados al parcial No.1 están alojados en un repositorio que creé de github, el cual es el siguiente enlace: https://github.com/JFOZ1010/ATTENDANCE

Eso es todo, muchas gracias.