Entrega 2 proyecto

Hernán Arenas - Rafael Fernández

I.- Esquema

Usuarios(uid INT PRIMARY KEY, nombre VARCHAR(50), nacimiento DATE, correo VARCHAR(40), nacionalidad VARCHAR(35))

Regiones(rid SMALLINT PRIMARY KEY, nombre VARCHAR(50), descripcion VARCHAR(200))

Hoteles(hotid INT PRIMARY KEY, rid SMALLINT, nombre VARCHAR(50), direccion VARCHAR(80), telefono VARCHAR(12), descripcion VARCHAR(300), estrellas SMALLINT, FOREIGN KEY(rid) REFERENCES Regiones(rid))

Habitaciones(habid INT PRIMARY KEY, hotid INT, nombre VARCHAR(50), precio FLOAT, FOREIGN KEY(hotid) REFERENCES Hoteles(hotid))

Reservas(resvid INT PRIMARY KEY, uid INT, habid INT, fecha_inicio DATE, fecha_fin DATE, FOREIGN KEY(uid) REFERENCES Usuarios(uid) , FOREIGN KEY(habid) REFERENCES Habitaciones(habid))

Restaurantes(restid INT PRIMARY KEY, rid SMALLINT, nombre VARCHAR(50), direccion VARCHAR(100), telefono VARCHAR(12), descripcion VARCHAR(300), FOREIGN KEY(rid) REFERENCES Regiones(rid))

Platos(pid INT PRIMARY KEY, restid INT, nombre VARCHAR(100), descripcion VARCHAR(250), precio FLOAT, FOREIGN KEY(restid) REFERENCES Restaurantes(restid))

Agencias(aid INT PRIMARY KEY, nombre VARCHAR(100), direccion VARCHAR(100), telefono VARCHAR(12))

Agencias_Regiones(aid INT, rid SMALLINT, FOREIGN KEY (aid) REFERENCES Agencias(aid) , FOREIGN KEY(rid) REFERENCES Regiones(rid) , PRIMARY KEY (aid, rid))

Tours(tid INT PRIMARY KEY, aid INT,descripcion VARCHAR(300), precio FLOAT, FOREIGN KEY(aid) REFERENCES Agencias(aid))

➢ **Forma del Esquema:**

Se puede observar que el esquema se encuentra en BCNF. Esto se debe a que las relaciones están en tercera forma normal (sin dependencias parciales ni transitivas) y además, porque para cada dependencia funcional de la forma X → Y, X es llave primaria.

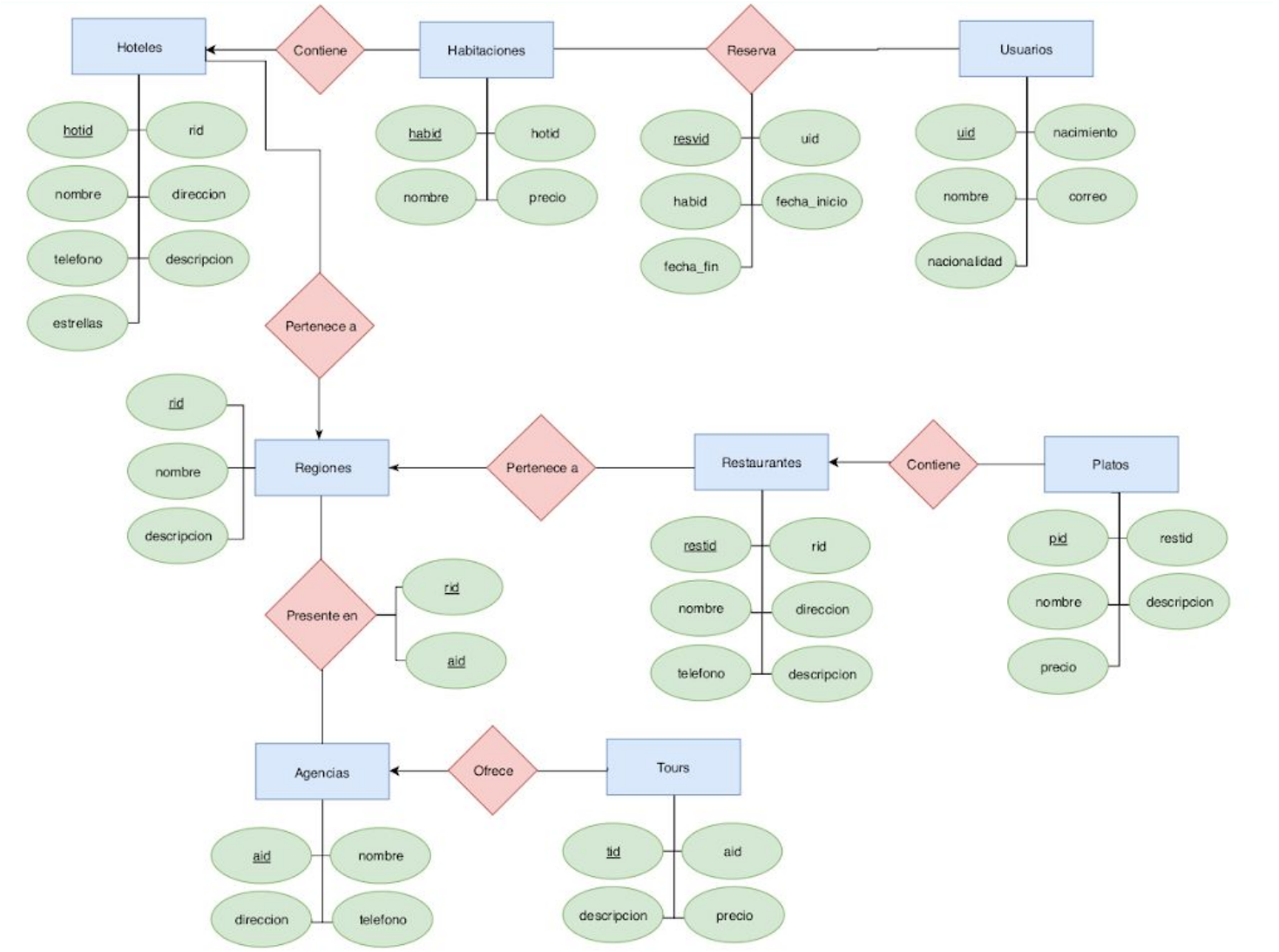
Para comprobar esto, se mostrarán a continuación las dependencias funcionales de cada relación, de manera que se puedan comparar fácilmente con el esquema propuesto.

- Usuarios: uid → nombre, nacimiento, correo, nacionalidad
- Regiones: rid → nombre, descripcion
- Hoteles: hotid → rid, nombre, direccion, telefono, descripcion, estrellas
- Habitaciones: habid → hotid, nombre, precio
- Reservas: resvid → uid, habid, fecha_inicio, fecha_fin
- Restaurantes: restid → rid, nombre, direccion, telefono, descripcion
- Platos: pid → restid, nombre, descripcion, precio
- Agencias: aid → nombre, direccion, telefono
- Agencias_Regiones: aid, rid → aid, rid (dependencia funcional trivial)
- Tours: tid → aid, descripcion, precio

➢ **LLaves del esquema:**

Dado que el esquema se presentó en un formato similar a SQL, se puede observar en el apartado anterior cuáles son los atributos en cada relación que son considerados como llaves primarias. En ciertas relaciones se puede observar también los atributos que deben ser tratados como llaves foráneas.

➤ **Diagrama Entidad-Relación:**



II.- SQL

➢ **Comandos utilizados para crear tablas:**

Para crear las tablas en PostgreSQL, se utilizó el siguiente comando: CREATE TABLE <R>, donde <R> corresponde a cada una de las relaciones presentadas en el esquema de la primera página.

Ejemplo: *"CREATE TABLE Agencias(aid INT PRIMARY KEY, nombre VARCHAR(100), direccion VARCHAR(100), telefono VARCHAR(12))"*

Se procedió de manera análoga para crear las demás tablas.

Luego, a partir de los archivos entregados, se generaron nuevos archivos .csv utilizando la librería Pandas en Python. Estos archivos nuevos contienen la información exacta que irá en cada tabla, de modo que no será necesario procesar más los datos a la hora de importar los archivos en PostgreSQL.

Los comandos utilizados para poblar las tablas fueron los siguientes:

- \cd csv_files
- \COPY Usuarios FROM 'Usuario.csv' DELIMITER ',' CSV HEADER;
- \COPY Regiones FROM 'Region.csv' DELIMITER ',' CSV HEADER;
- \COPY Hoteles FROM 'Hoteles.csv' DELIMITER ',' CSV HEADER;
- \COPY Habitaciones FROM 'Habitaciones.csv' DELIMITER ',' CSV HEADER;
- \COPY Reservas FROM 'Reserva.csv' DELIMITER ',' CSV HEADER;
- \COPY Restaurantes FROM 'Restaurantes.csv' DELIMITER ',' CSV HEADER;
- \COPY Platos FROM 'Platos.csv' DELIMITER ',' CSV HEADER;
- \COPY Agencias FROM 'Agencias.csv' DELIMITER ',' CSV HEADER;
- \COPY Agencias_Regiones FROM 'Agencias_Regiones.csv' DELIMITER ',' CSV HEADER;
- \COPY Tours FROM 'Tour.csv' DELIMITER ',' CSV HEADER;

4. Consultas SQL

Para las siguientes consultas se asumió que no se ingresaran valores inválidos (ids negativos, etc)

1) *Dada una región, muestre todos los platos de los restaurantes ubicados en dicha región.*

```
SELECT DISTINCT P.nombre, P.descripcion, P.precio
FROM platos as P, restaurantes as Res, regiones as Reg
WHERE Res.restid=P.restid AND Reg.rid=$region_id AND Reg.rid=Res.rid;
```

2) *Dado un número de estrellas, muestre todas las habitaciones de hoteles con más de esa cantidad de estrellas, junto al nombre del hotel en el que está.*

```
SELECT DISTINCT H.habid, H.nombre, Ho.nombre
FROM Habitaciones as H, Hoteles as Ho
WHERE Ho.estrellas=$estrellas AND H.hotid=Ho.hotid;
```

3) *Muestre todas las reservas a habitaciones realizadas por el usuario con id i entre las fechas a y b.*

```
SELECT Res.resvid, Res.habid, Res.fecha_inicio, Res.fecha_fin
FROM Reservas as Res
WHERE Res.uid=$uid AND Res.fecha_inicio > '$fecha_inicio'
    AND Res.fecha_fin < '$fecha_termino'";
```

4) *Entregue todos los tour de las agencias que están presente sólo en una región.*

```
SELECT T.tid, T.descripcion, T.precio, A.aid, A.nombre
FROM tours as T, agencias as A
WHERE T.aid = A.aid AND
                    (SELECT COUNT(DISTINCT AR.rid)
                     FROM agencias_regiones as AR
                     WHERE AR.aid = A.aid) = 1;
```

5) *Para cada región, entregue la habitación que ha sido reservada más veces.*

```
SELECT R.rid, R.nombre, HAB.habid, HAB.nombre, HAB.precio
FROM regiones R, habitaciones HAB
WHERE (SELECT HAB2.habid
    FROM habitaciones HAB2, hoteles HOT, reservas_habitacion RH
    WHERE HAB2.hotid = HOT.hotid AND RH.habid = HAB2.habid AND
        HOT.rid = R.rid
    ORDER BY RH.reservas DESC
    LIMIT 1) = HAB.habid
ORDER BY R.rid;
```

6) *Entregue todos los usuarios que han reservado la habitación más barata en la región II.*

```
SELECT U.uid, U.nombre, U.nacimiento, U.correo, U.nacionalidad
FROM usuarios U, reservas RES
WHERE RES.uid = U.uid AND
        (SELECT HAB2.habid
        FROM habitaciones HAB2, hoteles HOT
        WHERE HAB2.hotid = HOT.hotid AND HOT.rid = 1
        ORDER BY HAB2.precio
        LIMIT 1) = RES.habid;
```

7) *Dado un id de reserva, muestre el nombre del usuario que hizo la reserva junto al monto total que paga por esa reserva.*

```
SELECT U.nombre, H.precio * (R.fecha_fin - R.fecha_inicio)
FROM usuarios U, reservas R, habitaciones H
WHERE U.uid = R.uid AND R.habid = H.habid AND R.resvid=$resvid
```

8) *Dado un número i, entrega la i-ésima habitación más cara. En caso de empate muestre las dos.*

```
SELECT HAB1.habid, HAB1.hotid, HAB1.nombre, HAB1.precio
FROM habitaciones HAB1
WHERE (SELECT DISTINCT HAB2.precio
        FROM habitaciones HAB2
        ORDER BY HAB2.precio DESC
        LIMIT 1 OFFSET $i) = HAB1.precio;
```