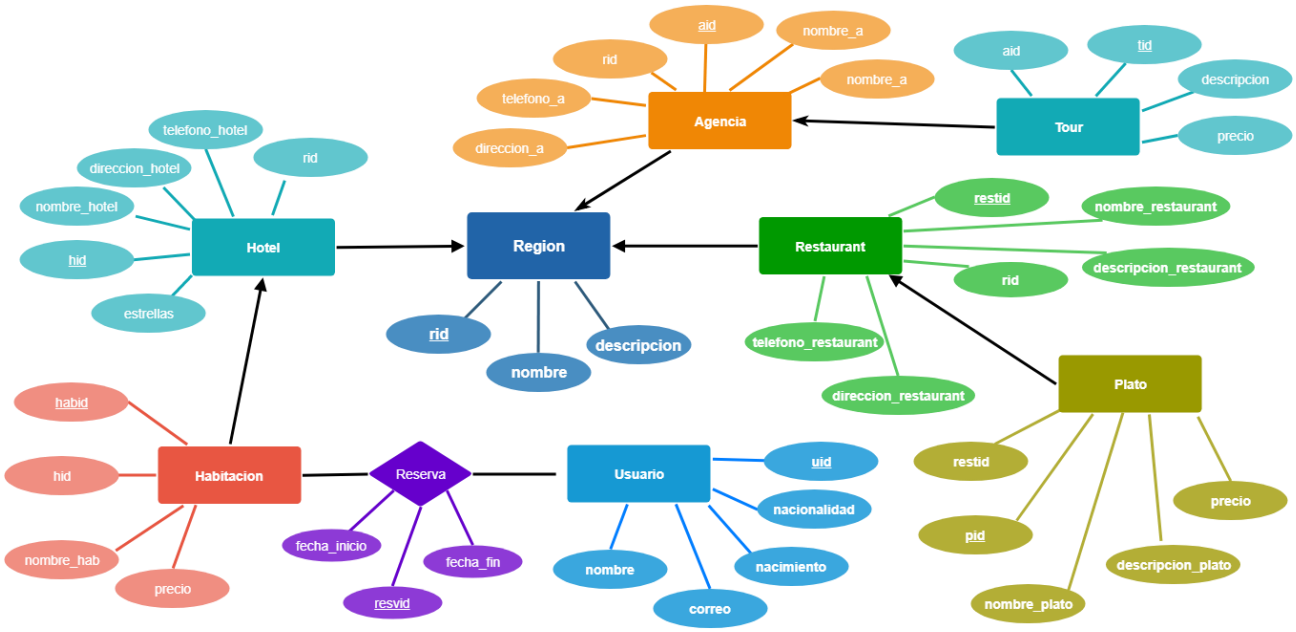
IIC2413 2019-1 E2 |Grupo 47

Benjamín Domínguez
Pablo Vejar

Esquema de la base de datos

- Usuario (uid int, nombre varchar(100), nacimiento date, correo varchar(100), nacionalidad varchar(100))
- Region (rid int, nombre varchar(100), descripción varchar(1000))
- Tour (tid int, aid int, descripción varchar(100), precio float)
- Restaurant (restid int, nombre_restaurant varchar(100), direccion_restaurant varchar(100), telefono_restaurant int, descripcion_restaurant varchar(100), rid int)
- Reserva (resvid int, uid int, habid int, fecha_inicio date, fecha_fin date)
- Plato (pid int, restid int, nombre_plato varchar(100), descripcion_plato varchar(100), precio_plato float)
- Hotel (hid int, nombre_hotel varchar(100), direccion_hotel varchar(100), telefono_hotel int, estrellas int, rid int)
- Habitacion (habid int, hid int, nombre_hab varchar(100), precio_hab float)
- Agencia (aid int, nombre_a varchar(100), direccion_a varchar(100), telefono_a int, rid int)

Diagrama E/R


aid
nombre_a
rid
nombre_a
telefono_a
Agencia
direccion_a
aid
tid
descripcion
Tour
precio
telefono_hotel
rid
direccion_hotel
nombre_hotel
Hotel
hid
estrellas
Region
rid
nombre
descripcion
restid
nombre_restaurant
Restaurant
rid
descripcion_restaurant
telefono_restaurant
direccion_restaurant
Plato
habid
hid
Habitacion
nombre_hab
precio
Reserva
fecha_inicio
fecha_fin
resvid
Usuario
uid
nacionalidad
nombre
nacimiento
correo
restid
precio
pid
descripcion_plato
nombre_plato

Restricciones de llaves primarias y foráneas

Nombre Tabla	Llaves Primarias	Llaves Foráneas
Usuario	uid	
Region	rid	
Tour	tid	aid (tabla agencia)
Restaurant	restid	rid (tabla región)
Reserva	resvid	uid (tabla usuario), habid (tabla habitacion)
Platos	pid	restid (tabla restaurant)
Hotel	hid	rid (tabla región)
Habitacion	habid	hid (tabla hotel)
Agencia	aid	rid (tabla región)

Justificación BCNF

Una relación R está en BCNF si para toda dependencia funcional no trivial X → Y, X es llave

Un esquema está en BCNF si todas sus relaciones están en BCNF. Por lo tanto, para asegurarnos que nuestro modelo esté en BCNF, revisaremos cada tabla, agregaremos las dependencias funcionales y veremos que están en BCNF. Esto es, que será una llave donde X será la clave primaria de la tabla.

Lo que hicimos fue descomponer las relaciones que tenían dependencias con problemas para BCNF. De este modo, descompucimos agencia_agenciaregion_tour en agencia, región y tour, hoteles_habitaciones en hoteles, habitaciones y restaurante_platos en restaurante, plato. Ahora veremos que con el resultado para todas las tablas cada dependencia funcional es una llave.

- Usuario

  uid -> nombre, nacimiento date, correo, nacionalidad

- Agencia
  aid -> nombre, direccion_a, telefono_a, rid
- Region
  rid -> nombre, descripción
- Tour
  tid -> descripción, precio float, aid
- Restaurant
  restid -> nombre_restaurant, direccion_restaurant, telefono_restaurant, descripcion_restaurant, rid

- Platos
  pid -> restid, nombre_plato, descripcion_plato, precio_plato
- Reserva
  resvid -> fecha_inicio, fecha_fin, uid, habid
- Hotel
  hid -> nombre_hotel, direccion_hotel, telefono_hotel, estrellas, rid
- Habitacion
  Habid -> hid, nombre_hab, precio_hab

Comandos SQL

Primero se realiza la conexión al servidor para grupo 47 con comando:

$ssh grupo47@bases.ing.puc.cl

Luego, tras ingresar contraseña de grupo, se asocia la base de datos a PostfresSQL mediante el comando:

$psql

Luego se crean las tablas, en principio vacias, definiendo únicamente su esquema (nombre a las columnas, con tipo de argumento y llaves -primarias y foráneas):

- CREATE TABLE usuario(uid int PRIMARY KEY, nombre varchar(100), nacimiento date, correo varchar(100), nacionalidad varchar(100))
- CREATE TABLE region(rid int PRIMARY KEY, nombre varchar(100), descripción varchar(1000))
- CREATE TABLE tour(tid int PRIMARY KEY, aid int, descripción varchar(100), precio float, FOREIGN KEY(aid) REFERENCES(agencia(aid)))
- CREATE TABLE restaurant(restid int PRIMARY KEY, nombre_restaurant varchar(100), direccion_restaurant varchar(100), telefono_restaurant int, descripcion_restaurant varchar(100), rid int, FOREIGN KEY(rid) REFERENCES(region(rid)))
- CREATE TABLE reserva(resvid int PRIMARY KEY, uid int, habid int, fecha_inicio date, fecha_fin date, FOREIGN KEY(uid) REFERENCES(usuario(uid)), FOREIGN KEY(habid) REFERENCES(habitacion(habid)))
- CREATE TABLE plato(pid int PRIMARY KEY, restid int, nombre_plato varchar(100), descripcion_plato varchar(100), precio_plato float, FOREIGN KEY(restid) REFERENCES(restaurant(restid)))
- CREATE TABLE hotel(hid int PRIMARY KEY, nombre_hotel varchar(100), direccion_hotel varchar(100), telefono_hotel int, estrellas int, rid int, FOREIGN KEY(rid) REFERENCES(region(rid)))
- CREATE TABLE habitacion(habid int PRIMARY KEY, hid int, nombre_hab varchar(100), precio_hab float, FOREIGN KEY(hid) REFERENCES(hotel (hid)))
- CREATE TABLE agencia(aid int PRIMARY KEY, nombre_a varchar(100), direccion_a varchar(100), telefono_a int, rid int, FOREIGN KEY(rid) REFERENCES(region(rid)))

Por último, llenamos las tablas al instanciarlas con los archivos csv, con los siguientes comandos:

● \copy usuario FROM '~/Sites/Entrega2/app/Database/usuario.csv' delimiter ',' csv HEADER

● \copy region FROM '~/Sites/Entrega2/ app/Database/region.csv' delimiter ';' csv HEADER

● \copy tour FROM '~/Sites/Entrega2/ app/Database/tours.csv' delimiter ',' csv HEADER

● \copy restaurant FROM '~/Sites/Entrega2/ app/Database/restaurants.csv' delimiter ',' csv HEADER

● \copy reserva FROM '~/Sites/Entrega2/ app/Database/reserva.csv' delimiter ',' csv HEADER

● \copy plato FROM '~/Sites/Entrega2/ app/Database/platos.csv' delimiter ',' csv HEADER

● \copy hotel FROM '~/Sites/Entrega2/ app/Database/hoteles.csv' delimiter ',' csv HEADER

● \copy habitacion FROM '~/Sites/Entrega2/ app/Database/habitaciones.csv' delimiter ';' csv HEADER

● \copy agencia FROM '~/Sites/Entrega2/ app/Database/agencias.csv' delimiter ',' csv HEADER

Consultas en SQL

1. SELECT plato.nombre_plato, plato.descripcion_plato, plato.precio_plato

   FROM plato,

   restaurant,

   region

   WHERE region.rid = '$region'

   AND restaurant.re_rid = region.rid

   AND plato.restid_p = restaurant.restid

2. SELECT habitacion.nombre_hab, habitacion.habid, habitacion.precio_hab, hotel.nombre_hotel,

   hotel.estrellas

   FROM habitacion, hotel

   WHERE hotel.estrellas > '$stars' AND habitacion.h_hid = hotel.hid;

3. SELECT reserva.resvid, reserva.fecha_inicio, reserva.fecha_fin, habitacion.habid,
   habitacion.nombre_hab
   FROM reserva, habitacion
   WHERE reserva.r_uid = '$usuario' AND reserva.r_habid = habitacion.habid
   AND reserva.fecha_inicio > '$inicio' AND reserva.fecha_fin < '$fin'

4. SELECT tour.tid, tour.descripcion_tour, tour.precio_tour
   FROM tour, (SELECT agencia1.aid, agencia.rid
   FROM (SELECT agencia.aid, COUNT(*) AS cantidad
   FROM agencia
   GROUP BY agencia.aid
   HAVING COUNT(aid) = 1) AS agencia1, agencia
   WHERE agencia1.aid = agencia.aid) AS agencia_n
   WHERE tour.aid = agencia_n.aid AND agencia_n.rid = $region
   GROUP BY tour.tid;

5. (*) Se assume por issue #66 del repositorio del curso que en caso de empate se muestra una sola habitación arbitraria.
   SELECT hab.habid, hab.h_hid, hab.nombre_hab, hab.precio_hab, COUNT(hab.habid) AS countres
   FROM reserva AS res, hotel AS hot, habitacion AS hab
   WHERE res.r_habid = hab.habid AND hab.h_hid = hot.hid AND hot.h_rid = '$region'
   GROUP BY hab.habid
   HAVING COUNT(hab.habid) = (SELECT MAX(countres2)
   FROM (SELECT COUNT(hab2.habid) AS countres2
   FROM reserva AS res2, hotel AS hot2, habitacion AS hab2
   WHERE res2.r_habid = hab2.habid AND hab2.h_hid = hot2.hid AND hot2.h_rid = '$region'
   GROUP BY hab2.habid) AS foo)

LIMIT 1

6. SELECT usuario.uid, usuario.nombre_usuario, consulta1.habid, consulta1.nombre_hab, consulta1.precio_hab

FROM reserva, usuario, (SELECT H1.habid, H1.nombre_hab, H1.precio_hab, H2.hid, H2.nombre_hotel, H2.h_rid

FROM hotel H2, habitacion H1

WHERE H2.h_rid = '$region'

AND H1.h_hid = H2.hid

AND H1.precio_hab = (SELECT MIN(H1.precio_hab)

FROM habitacion H1, hotel H2 WHERE H2.h_rid = '$region' AND H1.h_hid = H2.hid)) AS consulta1

WHERE reserva.r_uid = usuario.uid AND reserva.r_habid = consulta1.habid

7. SELECT reserva.resvid, usuario.uid, usuario.nombre_usuario, usuario.correo_usuario,

habitacion.precio_hab

FROM usuario, reserva, habitacion

WHERE usuario.uid = reserva.r_uid AND reserva.resvid = '$reserva' AND

habitacion.habid = reserva.r_habid;

8. SELECT hab.habid, hab.h_hid, hab.nombre_hab, hot.nombre_hotel, hab.precio_hab
FROM habitacion AS hab, hotel AS hot
WHERE hab.h_hid = hot.hid AND ($numhab - 1) = (SELECT COUNT(habita2.habid)
FROM habitacion AS habita2
WHERE habita2.precio_hab > hab.precio_hab);