IIC2413 2019-1 E2 |Grupo 47

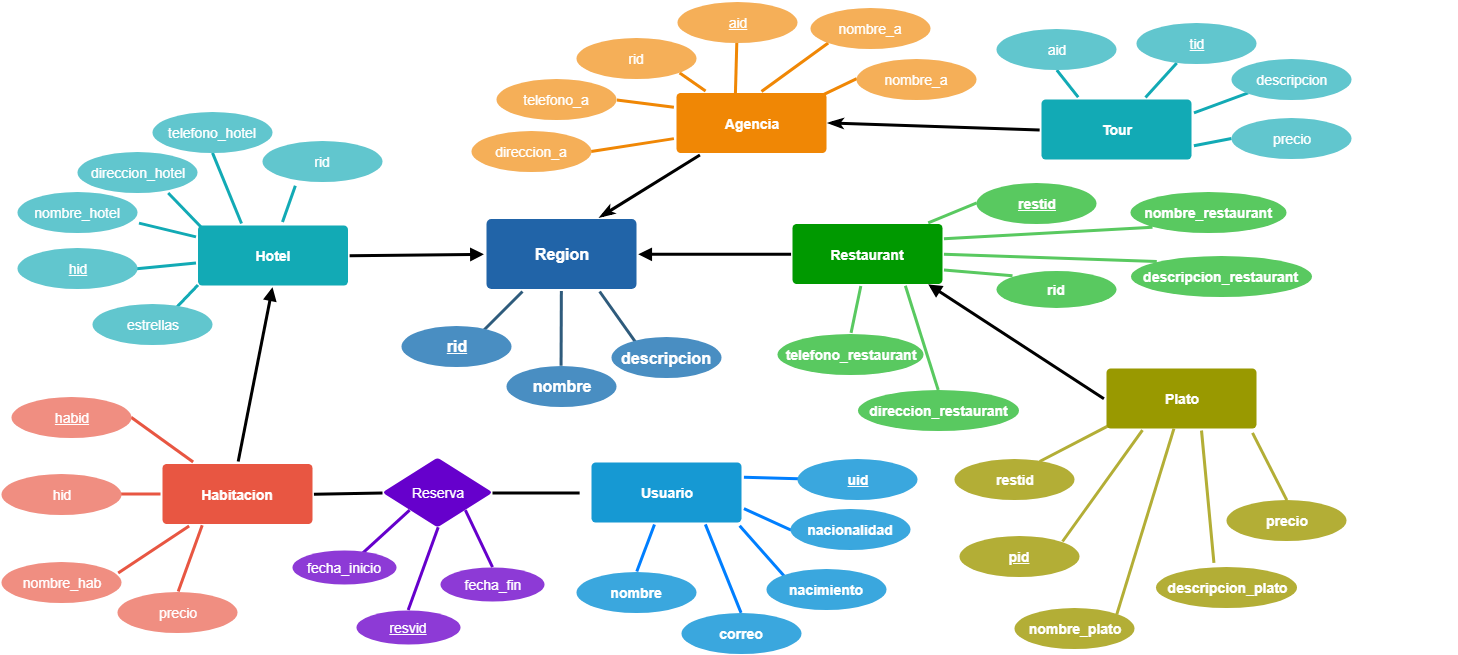
Benjamín Domínguez

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



## 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

1. 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;

1. 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'

1. 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;

1. (\*) 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

1. 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

1. 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;

1. 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);