

PONTIFICIA UNIVERSIDAD CATÓLICA

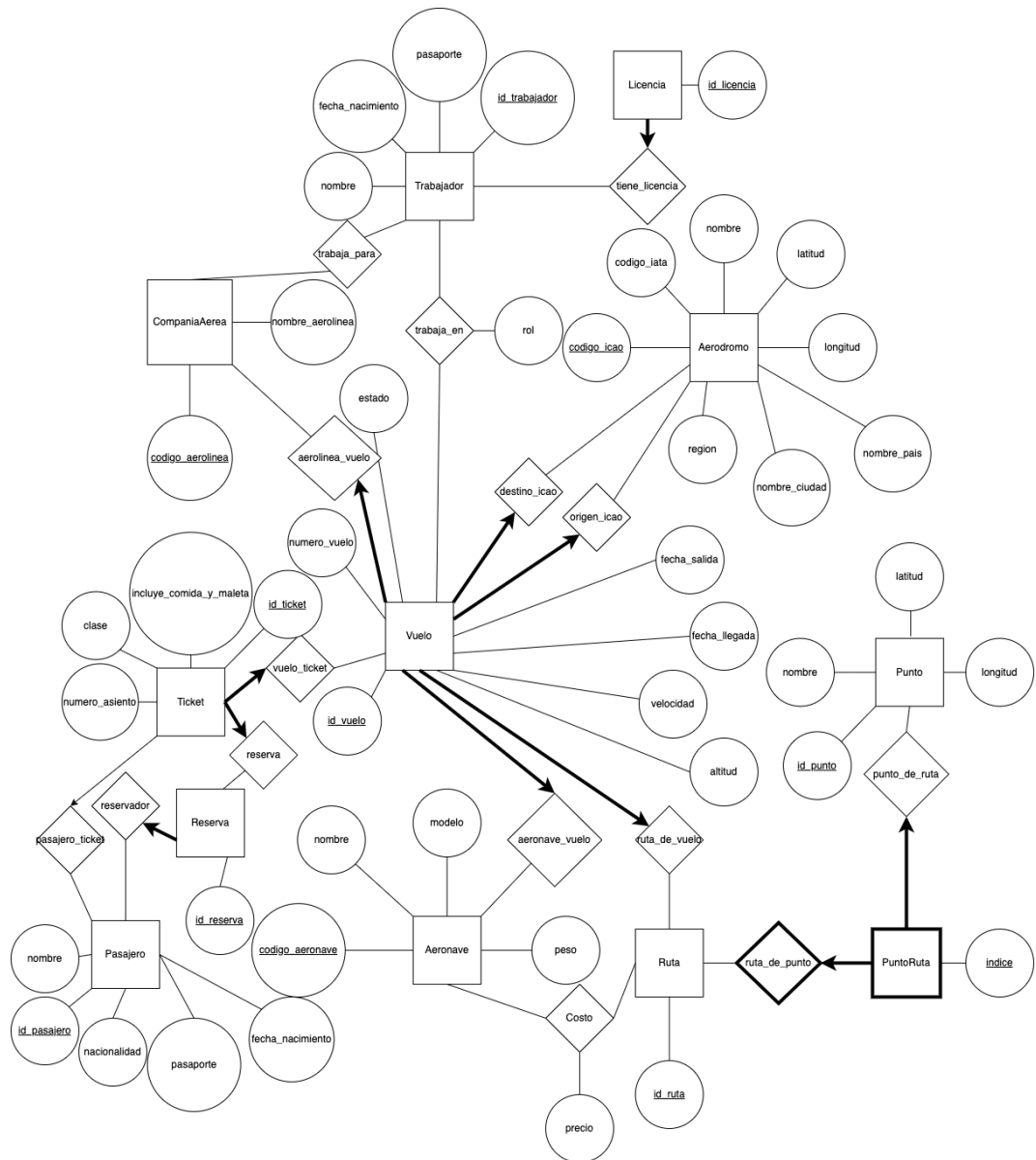
IIC2413

Entrega 2

Nombres: Martín Andrighetti, Lucas Fernández

Fecha entrega: 2022-05-19

1. Diagrama E/R



2. Modelo relacional

■ Aerodromo

(codigo_icao char(4), codigo_iata char(3), nombre varchar(255), latitud float,
longitud float, nombre_ciudad varchar(255), nombre_pais varchar(255))

- **Aeronave**

(codigo_aeronave **char(7)**, **nombre varchar(255)**, **modelo varchar(255)**, **peso float**)

- **CompaniaAerea**

(codigo_aerolinea **char(3)**, **nombre_aerolinea varchar(255)**)

- **Costo**

(id_ruta **int**, codigo_aeronave **char(7)**, **precio float**)

id_ruta es llave foránea de Ruta.id_ruta

codigo_aeronave es llave foránea de Aeronave.codigo_aeronave

- **Licencia**

(id_licencia **int**, id_trabajador **int**)

id_trabajador es llave foránea de Trabajador.id_trabajador

- **Pasajero**

(id_pasajero **int**, **pasaporte varchar(15)**, **nombre varchar(255)**,

fecha_nacimiento date, **nacionalidad varchar(255)**)

- **Punto**

(id_punto **int**, **nombre varchar(255)**, **latitud float**, **longitud float**)

- **PuntoRuta**

(id_ruta **int**, indice **int**, id_punto **int**)

Restricción: Para cada ruta existe una colección secuencial de índices, comenzando desde cero.

id_ruta es llave foránea de Ruta.id_ruta

id_punto es llave foránea de Punto.id_punto

- **Reserva**

(id_reserva **int**, id_reservador **int**)

id_reservador es llave foránea de Pasajero.id_pasajero

- **Ruta**

(id_ruta **int**)

- **Trabajador**

(id_trabajador **int**, **pasaporte varchar(15)**, **nombre varchar(255)**,

fecha_de_nacimiento date)

- TrabajaEn

(id_vuelo int, id_trabajador int, rol varchar(255))

id_vuelo es llave foránea de Vuelo.id_vuelo

id_trabajador es llave foránea de Trabajador.id_trabajador

- TrabajaPara

(id_trabajador int, codigo_aerolinea char(3))

id_trabajador es llave foránea de Trabajador.id_trabajador

codigo_aerolinea es llave foránea de CompaniaAerea.codigo_aerolinea

- Ticket

(id_ticket int, id_reserva int, id_pasajero int, id_vuelo int, numero_asiento int, clase varchar(31), incluye_comida_y_maleta boolean)

id_pasajero puede tomar valores nulos.

id_reserva es llave foránea de Reserva.id_reserva

id_pasajero es llave foránea de Pasajero.id_pasajero

id_vuelo es llave foránea de Vuelo.id_vuelo

- Vuelo

(id_vuelo int, numero_vuelo varchar(31), origen_icao char(4), destino_icao char(4), codigo_aerolinea char(3), fecha_salida timestamp, fecha_llegada timestamp, velocidad float, altitud float, id_ruta int, codigo_aeronave char(7), estado varchar(31))

origen_icao es llave foránea de Aerodromo.codigo_icao

destino_icao es llave foránea de Aerodromo.codigo_icao

codigo_aerolinea es llave foránea de CompaniaAerea.codigo_aerolinea

id_ruta es llave foránea de Ruta.id_ruta

codigo_aeronave es llave foránea de Aeronave.codigo_aeronave

3. Justificación del modelo

El modelo relacional anterior se encuentra en forma BCNF. Para demostrar esto, se listan

3.1. Supuestos

1. Un vuelo disponible para comprar es sinónimo de un vuelo con estado "publicado". Esto es, los asientos nunca se acaban.
2. El número de pasaporte consiste en hasta 9 letras y números, según lo acordado internacionalmente.
3. Las coordenadas geográficas se representan como un par de latitud y longitud, cada uno representado como un número real. Para la latitud, un número positivo indica coordenada este. Para la longitud, un número positivo indica coordenada norte.
4. El peso de una aeronave se redondea al kilogramo más cercano y se almacena como entero. De esta manera se pueden agrupar pesos de nave con un mismo costo.
5. El precio se especifica como un número flotante de dólares.
6. No se incluye la dependencia del país a partir de la latitud y longitud. Esto se realiza por simplicidad y porque hay puntos del planeta en que la soberanía es ambigua o incluso polémica.

3.2. Justificación de cumplimiento con BCNF

■ Aerodromo

codigo_icao → < todos >

codigo_iata → < todos >

codigo_icao y *codigo_iata* son llaves candidatas. Dado que el lado izquierdo de todas las dependencias no triviales son llaves candidatas se cumple BCNF.

■ Aeronave

codigo_aeronave → < todos >

Dado que el lado izquierdo de la única dependencia no trivial es llave, se cumple BCNF.

■ CompaniaAerea

codigo_aerolinea → *nombre_aerolinea*

Dado que la parte izquierda de la dependencia no trivial es una llave, se cumple BCNF.

- Costo

$id_ruta, codigo_aeronave \rightarrow precio$

Dado que el lado izquierdo de la única dependencia es llave, se cumple BCNF.

- Licencia

$id_licencia \rightarrow id_trabajador$

Dado que el lado izquierdo de la única dependencia funcional es llave, se cumple BCNF.

- Pasajero

$id_pasajero \rightarrow < todos >$

$pasaporte \rightarrow < todos >$

Dado que los lados izquierdos de ambas dependencias no triviales son llaves candidatas, se cumple BCNF.

- Punto

$id_punto \rightarrow < todos >$

Asumiendo que no hay puntos duplicados: $latitud, longitud \rightarrow < todos >$

Dado que el lado izquierdo de ambas dependencias no triviales son llaves candidatas, se cumple BCNF.

- PuntoRuta

$id_ruta, indice \rightarrow id_punto$

Asumiendo que las rutas no pasan dos veces por el mismo punto: $id_ruta, id_punto \rightarrow indice$

$id_ruta, indice$ es llave, y id_ruta, id_punto es llave candidata. Como estas son las partes izquierdas de las dependencias no triviales, se cumple BCNF.

- Reserva

$id_reserva \rightarrow id_reservador$

Dado que el lado izquierdo de la única dependencia no trivial, $id_reserva$, es llave candidata, se cumple BCNF.

- Ruta

No hay atributos suficientes para formar una dependencia, por lo que se cumple BCNF.

- Trabajador

$id_trabajador \rightarrow < todos >$

$pasaporte \rightarrow < todos >$

Las llaves candidatas son $id_trabajador$ y $pasaporte$. Las dependencias funcionales no triviales tienen en el lado izquierdo llaves candidatas, por lo que se cumple BCNF.

- TrabajaEn

$id_vuelo, id_trabajador \rightarrow rol$

Dado que el lado izquierdo de la única dependencia funcional no trivial es llave, se cumple BCNF.

- TrabajaPara

No hay dependencias funcionales no triviales, por lo que se cumple BCNF.

- Ticket

$id_ticket \rightarrow < todos >$

Asumiendo que un pasajero no puede comprar 2 asientos para sí: $id_pasajero, id_vuelo \rightarrow < todos >$

Asumiendo que no se puede comprar dos veces el mismo asiento: $id_vuelo, numero_asiento \rightarrow < todos >$

Todas las dependencias no triviales definen llaves candidatas, por lo que se cumple BCNF.

- Vuelo

$id_vuelo \rightarrow < todos >$

Dado que el lado izquierdo de la única dependencia funcional no trivial es llave, se cumple BCNF.

4. Consultas SQL

1. Muestre todos los vuelos pendientes de ser aprobados por DGAC

```
SELECT Vuelo.numero_vuelo, Origen.nombre as origen, Destino.nombre as destino,  
       Vuelo.fecha_salida, Vuelo.fecha_llegada, Vuelo.estado  
FROM Vuelo, Aerodromo as Origen, Aerodromo as Destino  
WHERE estado = 'pendiente'  
       AND Vuelo.origen_icao = Origen.codigo_icao  
       AND Vuelo.destino_icao = Destino.codigo_icao;
```

2. Dado un código ICAO de un aeródromo ingresado por el usuario y una aerolínea seleccionada por el usuario, liste todos los vuelos aceptados de dicha aerolínea que tienen como destino el aeródromo

Tomando *\$codigo* e *\$aerolinea_escogida* como lo entregado por el usuario:

```
SELECT Vuelo.numero_vuelo, Origen.nombre as origen, Destino.nombre as destino,
       Vuelo.fecha_salida, Vuelo.fecha_llegada, Vuelo.estado
FROM Vuelo, CompaniaAerea, Aerodromo as Origen, Aerodromo as Destino
WHERE UPPER(CompaniaAerea.nombre_aerolinea) LIKE '%$aerolinea_escogida%'
      AND CompaniaAerea.codigo_aerolinea = Vuelo.codigo_aerolinea
      AND UPPER(Vuelo.destino_icao) LIKE '%$codigo%'
      AND Vuelo.estado = 'aceptado'
      AND Vuelo.origen_icao = Origen.codigo_icao
      AND Vuelo.destino_icao = Destino.codigo_icao;
```

3. Dado un código de reserva ingresado por el usuario, liste los tickets asociados a esta junto a sus pasajeros y costos

Tomando *< \$codigo_reserva >* como lo entregado por el usuario y asumiendo que este codigo tiene matching completo debido a la naturaleza de la variable:

```
SELECT Ticket.id_ticket, Vuelo.numero_vuelo, Origen.nombre as origen,
       Destino.nombre as destino, Vuelo.fecha_salida, Vuelo.fecha_llegada,
       Vuelo.estado, Ticket.numero_asiento, Ticket.clase,
       Ticket.incluye_comida_y_maleta, Pasajero.pasaporte,
       Pasajero.nombre, Costo.precio
FROM Ticket, Pasajero, Vuelo, Costo, Aerodromo as Origen, Aerodromo as Destino
WHERE Ticket.id_reserva = $codigo_reserva
      AND ticket.id_pasajero = Pasajero.id_pasajero
      AND Vuelo.id_vuelo = Ticket.id_vuelo
      AND Vuelo.id_ruta = Costo.id_ruta
      AND Vuelo.codigo_aeronave = Costo.codigo_aeronave
      AND Vuelo.origen_icao = Origen.codigo_icao
      AND Vuelo.destino_icao = Destino.codigo_icao;
```


4. Por cada aerolínea, muestre al cliente que ha comprado la mayor cantidad de tickets

```
SELECT CompaniaAerea.nombre_aerolinea, Pasajero.pasaporte,
       Pasajero.nombre, Cantidades.cantidad
FROM CompaniaAerea, Pasajero, (
    SELECT Cantidades.codigo_aerolinea,
           MAX(Cantidades.cantidad) as max_cantidad
    FROM (
        SELECT Vuelo.codigo_aerolinea,
               Reserva.id_reservador as id_cliente, COUNT(*) as cantidad
        FROM Ticket, Vuelo, Reserva
        WHERE Ticket.id_reserva = Reserva.id_reserva
              AND Ticket.id_vuelo = Vuelo.id_vuelo
        GROUP BY Vuelo.codigo_aerolinea, Reserva.id_reservador
    ) as Cantidades
    GROUP BY Cantidades.codigo_aerolinea
) as MaxCantidades, (
    SELECT Vuelo.codigo_aerolinea,
           Reserva.id_reservador as id_cliente, COUNT(*) as cantidad
    FROM Ticket, Vuelo, Reserva
    WHERE Ticket.id_reserva = Reserva.id_reserva
          AND Ticket.id_vuelo = Vuelo.id_vuelo
    GROUP BY Vuelo.codigo_aerolinea, Reserva.id_reservador
) as Cantidades
WHERE Cantidades.codigo_aerolinea = MaxCantidades.codigo_aerolinea
     AND Cantidades.cantidad = MaxCantidades.max_cantidad
     AND Cantidades.codigo_aerolinea = CompaniaAerea.codigo_aerolinea
     AND Cantidades.id_cliente = Pasajero.id_pasajero
ORDER BY CompaniaAerea.nombre_aerolinea, Pasajero.nombre;
```

5. Al ingresar el nombre de una aerolínea, liste la cantidad de vuelos que tienen en cada uno de los estados

Tomando $\langle \$nombre_escogido \rangle$ como lo entregado por el usuario:

```
SELECT CompaniaAerea.nombre_aerolinea, Vuelo.estado, COUNT(Vuelo.id_vuelo) as cantidad
FROM Vuelo, CompaniaAerea
WHERE UPPER(CompaniaAerea.nombre_aerolinea) LIKE '%$nombre_escogido%'
AND CompaniaAerea.codigo_aerolinea = Vuelo.codigo_aerolinea
GROUP BY CompaniaAerea.nombre_aerolinea, Vuelo.estado;
```

6. Muestre la aerolínea que tiene el mayor porcentaje de vuelos aceptados.

```
SELECT PorcentajesAprobados.nombre_aerolinea, PorcentajesAprobados.porcentaje
FROM (
    SELECT porcentaje
    FROM (
        SELECT CompaniaAerea.nombre_aerolinea,
            100 * VuelosAprobados.cantidad_aprobada/VuelosTotales.cantidad
            as porcentaje
        FROM CompaniaAerea, (
            SELECT codigo_aerolinea, count(id_vuelo) as cantidad_aprobada
            FROM Vuelo
            WHERE estado = 'aceptado'
            GROUP BY codigo_aerolinea
        ) as VuelosAprobados, (
            SELECT codigo_aerolinea, count(id_vuelo) as cantidad
            FROM Vuelo
            GROUP BY codigo_aerolinea
        ) as VuelosTotales
        WHERE CompaniaAerea.codigo_aerolinea = VuelosTotales.codigo_aerolinea
            AND CompaniaAerea.codigo_aerolinea
            = VuelosAprobados.codigo_aerolinea
            AND VuelosTotales.cantidad > 0
    ) as PorcentajesAprobados
    ORDER BY porcentaje DESC
    LIMIT 1
) as PorcentajeMayor, (
    SELECT CompaniaAerea.nombre_aerolinea,
        100 * VuelosAprobados.cantidad_aprobada/VuelosTotales.cantidad
```

```

        as porcentaje
FROM CompaniaAerea, (
    SELECT codigo_aerolinea, count(id_vuelo) as cantidad_aprobada
    FROM Vuelo
    WHERE estado = 'aceptado'
    GROUP BY codigo_aerolinea
) as VuelosAprobados, (
    SELECT codigo_aerolinea, count(id_vuelo) as cantidad
    FROM Vuelo
    GROUP BY codigo_aerolinea
) as VuelosTotales
WHERE CompaniaAerea.codigo_aerolinea = VuelosTotales.codigo_aerolinea
    AND CompaniaAerea.codigo_aerolinea
    = VuelosAprobados.codigo_aerolinea
    AND VuelosTotales.cantidad > 0
) as PorcentajesAprobados
WHERE PorcentajesAprobados.porcentaje = PorcentajeMayor.porcentaje;

```

5. Pagina Web

Con el siguiente link se puede ingresar a la pagina web: <https://codd.ing.puc.cl/~grupo19/index.php?>