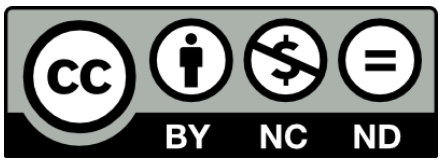
FBD
Examen V

Los Del DGIIM, losdeldgiim.github.io

Doble Grado en Ingeniería Informática y Matemáticas
Universidad de Granada



FBD
Examen V

Los Del DGIIM, `losdeldgiim.github.io`

Arturo Olivares Martos

Granada, 2024-2025

Asignatura Fundamentos de Bases de Datos.

Curso Académico 2020-21.

Descripción Convocatoria Extraordinaria. Práctico Parcial 2 (Seminarios 3-4).

1. Modelo 1

Considerar el esquema relacional de la Figura 1, donde cada paciente tiene asignada una aseguradora que cubre sus gastos médicos. La tabla `Consulta` almacena citas en las que un médico ha atendido a un paciente en una fecha dada.

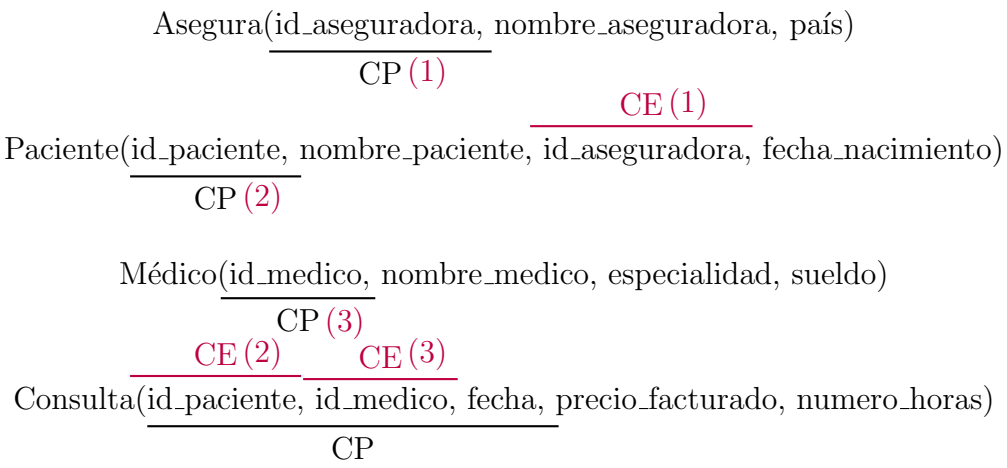


Figura 1: Esquema relacional del Modelo 1.

Ejercicio 1.1 (SQL)**.** Encontrar el id del paciente y el id de la aseguradora de los pacientes atendidos en consulta para los que el precio más alto de sus consultas es más bajo que el precio facturado medio del paciente de código `PAC01`.

```sql
SELECT id_paciente, id_aseguradora, MAX(precio_facturado)
    FROM paciente NATURAL JOIN consulta
    GROUP BY id_paciente, id_aseguradora
    HAVING MAX(precio_facturado) < (
        SELECT AVG(precio_facturado)
            FROM consulta
            GROUP BY id_paciente
            HAVING id_paciente = 'PAC01'
    );
```

Ejercicio 1.2 (SQL)**.** Encontrar el id y el nombre de los médicos que, no habiendo atendido en consulta nunca a un paciente asegurado por la aseguradora de id `MAP01`, han atendido al menos una consulta en los tres primeros meses de 2017.

```sql
SELECT id_medico, nombre_medico
    FROM medico
    WHERE id_medico IN (
        SELECT id_medico
            FROM consulta
            WHERE TO_CHAR(fecha, 'MM/YYYY')
                BETWEEN TO_DATE('01/2017', 'MM/YYYY') AND TO_DATE('03/2017',
                'MM/YYYY')
        MINUS
        SELECT id_medico
```

```
10          FROM consulta NATURAL JOIN paciente
            WHERE id_aseguradora = 'MAPO1'
    );
```

Ejercicio 1.3 (AR)**.** Encontrar, para cada médico que ha atendido en consulta, el precio más alto que ha facturado. Es suficiente con mostrar el id del médico, junto al precio facturado.

Establecemos los siguientes alias:

$$\rho(\text{Consulta}) = C_1, C_2$$

Busquemos en primer lugar, para cada médico, los precios que no son candidatos a ser el precio más alto facturado por un médico.

$$\pi_{\substack{C_1.\text{id\_médico} \\ C_1.\text{precio\_facturado}}} \left(\sigma_{\substack{C_1.\text{id\_médico}=C_2.\text{id\_médico} \\ \wedge \\ C_1.\text{precio\_facturado}<C_2.\text{precio\_facturado}}} (C_1 \times C_2) \right)$$

Por tanto, el precio más alto facturado por cada médico es:

$$\pi_{\substack{\text{id\_médico} \\ \text{precio\_facturado}}} (\text{Consulta}) - \pi_{\substack{C_1.\text{id\_médico} \\ C_1.\text{precio\_facturado}}} \left(\sigma_{\substack{C_1.\text{id\_médico}=C_2.\text{id\_médico} \\ \wedge \\ C_1.\text{precio\_facturado}<C_2.\text{precio\_facturado}}} (C_1 \times C_2) \right)$$

2. Modelo 2

Considerar el esquema relacional de la Figura 2, donde cada proyecto tiene asignada la empresa que lo elabora. La tabla `Revisión` registra revisiones indicando qué revisor revisa qué proyecto en qué fecha.

Empresa(id_empresa, nombre_empresa, país)
CP (1)

Proyecto(id_proyecto, titulo, id_empresa, fecha_elaboracion)
CP (3) CE (1)

Revisor(id_revisor, nombre_revisor, categoria, tarifa)
CP (2)

Revisión(id_revisor, id_proyecto, fecha, puntuacion)
CP CE (2) CE (3)

Figura 2: Esquema relacional del Modelo 2.

Ejercicio 2.1 (SQL)**.** Encontrar el id y el nombre de los revisores que, no habiendo revisado nunca un proyecto de la empresa con id `EMP04`, han hecho al menos una revisión en los primeros seis meses de 2018.

```sql
SELECT nombre_revisor, id_revisor
    FROM revisor
    WHERE id_revisor IN (
        SELECT id_revisor
            FROM revisión
            WHERE TO_CHAR(fecha, 'MM/YYYY')
                BETWEEN TO_DATE('01/2018', 'MM/YYYY') AND TO_DATE('06/2018',
                'MM/YYYY')
        MINUS
        SELECT id_revisor
            FROM revisión NATURAL JOIN proyecto
            WHERE id_empresa = 'EMP04'
    );
```

Ejercicio 2.2 (AR)**.** Encontrar, para cada proyecto revisado, la puntuación más baja que ha obtenido en sus revisiones. Es suficiente con mostrar el id del proyecto, junto a la puntuación solicitada.

Establecemos los siguientes alias:

$$\rho(\text{Revisión}) = R_1, R_2$$

Busquemos en primer lugar, para cada proyecto, las puntuaciones que no son candidatas a ser la puntuación más baja obtenida por un proyecto.

$$\pi_{\substack{R_1.\text{id\_proyecto} \\ R_1.\text{puntuación}}} \left(\sigma_{\substack{R_1.\text{id\_proyecto}=R_2.\text{id\_proyecto} \\ \wedge \\ R_1.\text{puntuación}>R_2.\text{puntuación}}} (R_1 \times R_2) \right)$$

Por tanto, la puntuación más baja obtenida por cada proyecto es:

$$\pi_{\substack{\text{id\_proyecto}\\ \text{puntuación}}}(\text{Revisión}) - \pi_{\substack{R_1.\text{id\_proyecto}\\ R_1.\text{puntuación}}}\left(\sigma_{\substack{R_1.\text{id\_proyecto}=R_2.\text{id\_proyecto}\\ \wedge\\ R_1.\text{puntuación}>R_2.\text{puntuación}}}(R_1 \times R_2)\right)$$

3. Modelo 3

Considerar el esquema relacional de la Figura 3, donde cada vehículo tiene asignado un modelo de una marca determinada. La tabla `Repara` registra reparaciones indicando qué mecánico repara qué vehículo en qué fecha.

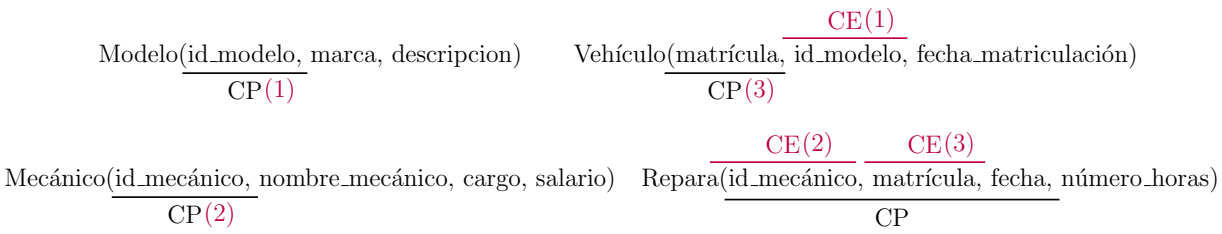


Figura 3: Esquema relacional del Modelo 3.

Ejercicio 3.1 (SQL)**.** Encontrar la matrícula y el id del modelo de los vehículos reparados para los que la duración más corta de sus reparaciones es más alta que la duración media de las reparaciones del vehículo con matrícula `1234`.

```sql
SELECT matrícula, id_modelo
    FROM vehículo NATURAL JOIN repara
    GROUP BY matrícula, id_modelo
    HAVING MIN(número_horas) > (
        SELECT AVG(número_horas)
            FROM repara
            WHERE matrícula = '1234'
            GROUP BY matrícula
    );
```

Ejercicio 3.2 (SQL)**.** Encontrar la matrícula y la fecha de matriculación de los vehículos reparados para los que la fecha más reciente de reparación es anterior a la reparación más antigua del vehículo con matrícula `5678` (recuerde que, en SQL, las fechas más recientes son mayores que las más antiguas).

```sql
SELECT matrícula, fecha_matriculación
    FROM vehículo NATURAL JOIN repara
    GROUP BY matrícula, fecha_matriculación
    HAVING MAX(fecha) < (
        SELECT MIN(fecha)
            FROM repara
            WHERE matrícula = '5678'
            GROUP BY matrícula
    );
```

Ejercicio 3.3 (SQL)**.** Encontrar el id y el nombre de los mecánicos que, no habiendo reparado nunca un coche con id de modelo `AURIS`, han hecho al menos una reparación en los últimos seis meses de 2020.

```sql
SELECT id_mecánico, nombre_mecánico
    FROM mecánico
    WHERE id_mecánico IN (
        SELECT id_mecánico
            FROM repara
            WHERE TO_CHAR(fecha, 'MM/YYYY')
                BETWEEN TO_DATE('07/2020', 'MM/YYYY') AND TO_DATE('12/2020',
                'MM/YYYY')
        MINUS
        SELECT id_mecánico
            FROM repara NATURAL JOIN vehículo
            WHERE id_modelo = 'AURIS'
    );
```

Ejercicio 3.4 (AR)**.** Encontrar, para cada modelo, la fecha de matriculación más reciente de vehículos de ese modelo (considere un orden en las fechas, en las que las más recientes son mayores que las más antiguas). Es suficiente con mostrar el id del modelo, junto a la fecha solicitada.

Establecemos los siguientes alias:

$$\rho\left(\pi_{\substack{\text{id\_modelo}\\ \text{fecha\_matriculación}}}(\text{Vehículo})\right) = V_1, V_2$$

Busquemos en primer lugar, para cada modelo, las fechas que no son candidatas a ser la fecha de matriculación más reciente de un vehículo de ese modelo.

$$\pi_{\substack{V_1.\text{id\_modelo}\\ V_1.\text{fecha\_matriculación}}}\left(\sigma_{\substack{V_1.\text{id\_modelo}=V_2.\text{id\_modelo}\\ \wedge\\ V_1.\text{fecha\_matriculación}<V_2.\text{fecha\_matriculación}}}(V_1 \times V_2)\right)$$

Por tanto, la fecha de matriculación más reciente de cada modelo es:

$$\pi_{\substack{\text{id\_modelo}\\ \text{fecha\_matriculación}}}(\text{Vehículo}) - \pi_{\substack{V_1.\text{id\_modelo}\\ V_1.\text{fecha\_matriculación}}}\left(\sigma_{\substack{V_1.\text{id\_modelo}=V_2.\text{id\_modelo}\\ \wedge\\ V_1.\text{fecha\_matriculación}<V_2.\text{fecha\_matriculación}}}(V_1 \times V_2)\right)$$

4. Modelo 4

Considerar el esquema relacional de la Figura 4, donde cada grupo musical tiene asignado un género musical. La tabla `Actuación` registra actuaciones indicando qué grupo actúa en qué población y cuándo.

Género(id_género, nombre_género, país_origen)
CP (1)

Grupo(id_grupo, nombre_grupo, id_género, fecha_creación)
CP (3) CE (1)

Población(id_población, nombre_población, número, habitantes)
CP (2)

Actuación(id_poblacion, id_grupo, fecha, duración)
CP CE (2) CE (3)

Figura 4: Esquema relacional del Modelo 4.

Ejercicio 4.1 (SQL)**.** Encuentre el id y la fecha de creación de los grupos con actuaciones para los que la fecha más reciente de actuación es anterior a la actuación más antigua del grupo de código `GRUP77` (recuerde que, en SQL, las fechas más recientes son mayores que las más antiguas).

```sql
SELECT id_grupo, fecha_creación
    FROM grupo
    WHERE id_grupo IN (
        SELECT id_grupo
            FROM actuación
            GROUP BY id_grupo
            HAVING MAX(fecha) < (
                SELECT MIN(fecha)
                    FROM actuación
                    WHERE id_grupo = 'GRUP77'
                    GROUP BY id_grupo
            )
    );
```

Ejercicio 4.2 (AR)**.** Elaborar un listado de los países que han dado origen a un solo género musical.

Creamos en primer lugar el siguiente alias.

$$\rho(\text{Género}) = G_1, G_2$$

Los países buscados son los que no han dado origen a dos géneros musicales distintos:

$$\pi_{\text{país}}(\text{Género}) - \pi_{G_1.\text{país}}\left(\sigma_{\substack{G_1.\text{país}=G_2.\text{país} \\ \wedge \\ G_1.\text{id\_género}\neq G_2.\text{id\_género}}}(G_1 \times G_2)\right)$$