ejercicios de consultas y normalización

Sea la tabla `prendas` *de una BD para la gestión de una tienda de ropa, definida como:*

```
CREATE TABLE prendas (
  nombre     VARCHAR2(24),
  color      VARCHAR2(24),
  talla      NUMBER(2)
);
```

A partir de los datos de la tabla se pretenden extraer las propiedades de los atributos. Ejemplo:

NOMBRE	COLOR	TALLA
FALDA	ROJO	4
FALDA	AZUL	5
FALDA	AZUL	4
FALDA	ROJO	5
PANTALON	VERDE	6
PANTALON	ROJO	6
CALCETIN		8

ejercicios de consultas y normalización

1 *Obtener las claves candidatas de la tabla* `prendas`

Para ello, comprobaremos si (`nombre`, `talla`, `color`*) es una clave de la tabla* `prendas`

```
SELECT prendas.*, count(*) veces
 FROM prendas
 GROUP BY nombre, color, talla
 HAVING count(*) > 1;
```

NOMBRE	COLOR	TALLA	VECES
FALDA	AZUL	5	2
FALDA	ROJO	4	2

problema: hay que eliminar las tuplas repetidas:

ejercicios de consultas y normalización

2 *eliminar las tuplas repetidas de la tabla* `prendas`

```
-- eliminar tuplas repetidas (no se ha especificado clave primaria de prendas)
DELETE FROM prendas
 WHERE rowid NOT IN (
  SELECT min(rowid) FROM prendas GROUP BY nombre, color, talla
 );
```

también se podría haber creado una vista para hacer la consulta paso a paso

ejercicios de consultas y normalización

2 *También se puede crear un "script" para automatizar la comprobación de claves:*

tst_clave.sql

```
SELECT &2, count(*) veces FROM &1 GROUP BY &2 HAVING count(*) > 1;

SELECT '   NO    ' "es clave" FROM dual
 WHERE EXISTS (
  SELECT * FROM &1 GROUP BY &2 HAVING count(*) > 1
 )
UNION
SELECT '   SI    ' "es clave" FROM dual
 WHERE NOT EXISTS (
  SELECT * FROM &1 GROUP BY &2 HAVING count(*) > 1
 );
```

```
@tst_clave prendas "nombre, talla, color"
```

ejercicios de consultas y normalización

```
SQL> @tst_clave prendas "nombre, talla, color"

listado de tuplas con valores repetidos de (nombre, talla, color)

antiguo   1: select &2, count(*) veces from &1 group by &2 having count(*) > 1
nuevo   1: select nombre, talla, color, count(*) veces from prendas group by nombre,
talla, color having count(*) > 1

ninguna fila seleccionada

antiguo   3:   select * from &1 group by &2 having count(*) > 1
nuevo   3:   select * from prendas group by nombre, talla, color having count(*) > 1
antiguo   8:   select * from &1 group by &2 having count(*) > 1
nuevo   8:   select * from prendas group by nombre, talla, color having count(*) > 1

es clave
---------
   SI
```

---

sólo queda comprobar que (nombre, color), (nombre, talla) y (color, talla) no son claves

¿qué pasa con los atributos nulos?

ejercicios de consultas y normalización

1 *Dados dos conjuntos de atributos X, Y de una tabla, comprobar si* $X \rightarrow Y$

por ejemplo, se comprobará si, en la tabla `prendas`, `nombre` → `color`

```
SELECT nombre, color, count(*) veces
 FROM prendas
 GROUP BY nombre, color
 HAVING count(*) > 1;
```

NOMBRE	COLOR	VECES
FALDA	AZUL	2
FALDA	ROJO	2

problema: hay tuplas repetidas:

ejercicios de consultas y normalización

1 *comprobar si, en la tabla* `prendas, nombre → color`

```
-- valores de nombre para los que no se cumple la DF
SELECT * FROM prendas P1
 WHERE EXISTS (
   SELECT * FROM prendas P2
     WHERE P2.nombre = P1.nombre
             and (P2.color <> P1.color
              or P2.color IS NULL and P1.color IS NOT NULL
              or P2.color IS NOT NULL and P1.color IS NULL)
 );
```

```
-- número de valores de nombre para los que no se cumple la DF
SELECT count(*) FROM prendas P1
 WHERE EXISTS (
   SELECT * FROM prendas P2
     WHERE P2.nombre = P1.nombre
             and (P2.color <> P1.color
              or P2.color IS NULL and P1.color IS NOT NULL
              or P2.color IS NOT NULL and P1.color IS NULL)
 );
```

ejercicios de consultas y normalización

1 *comprobar si, en la tabla* `prendas, nombre → color`

```
SELECT nombre, count(*), count(color), count(DISTINCT color)
FROM prendas
GROUP BY nombre;
```

```
NOMBRE                    COUNT(*) COUNT(COLOR) COUNT(DISTINCTCOLOR)
------------------------- -------- ----------- --------------------
CALCETIN                         1           0                    0
FALDA                            4           4                    2
PANTALON                         2           2                    2
```

```
CREATE VIEW nombre_color (nombre, color, veces) AS
 SELECT nombre, color, count(*)
 FROM prendas
 GROUP BY nombre, color;
```

ejercicios de consultas y normalización

1 *comprobar si, en la tabla* `prendas, nombre → color`

```
-- valores de nombre para los que no se cumple la DF
SELECT nombre, count(*)-1 C_sobran, sum(veces)-max(veces) min_Tpl_sobran
FROM nombre_color
GROUP BY nombre
HAVING count(*) > 1;
```

```
CREATE VIEW fallos(numFallos) AS
SELECT sum(tuplas)
FROM (SELECT sum(veces)-max(veces) tuplas
      FROM nombre_color
      GROUP BY nombre);
```

```
CREATE VIEW totPrendas(numPrendas) AS SELECT count(*) FROM prendas;
```

```
SELECT numFallos/numPrendas*100 "% fallos" FROM fallos, totPrendas;
```

```
 % fallos
----------
42,8571429
```

ejercicios de consultas y normalización

1 *sin utilizar vistas auxiliares*

```
SELECT numFallos/totPrendas*100 "% fallos"
 FROM
  (SELECT sum(tuplas) numFallos
    FROM (
     SELECT sum(veces)-max(veces) tuplas
      FROM (SELECT nombre, color, count(*) veces FROM prendas GROUP BY nombre, color)
      GROUP BY nombre
    )
  ), (SELECT count(*) totPrendas FROM prendas);
```

```
 % fallos
----------
42,8571429
```

ejercicios de consultas y normalización

1 *También se puede crear un "script" para automatizar la comprobación de DF*

tst_DF_nw.sql

```
SELECT numFallos/tot_tuplas*100 "% fallos"
 FROM
  (SELECT sum(tuplas) numFallos
   FROM (SELECT sum(veces)-max(veces) tuplas
          FROM (SELECT &2, &3, count(*) veces FROM &1 GROUP BY &2, &3)
          GROUP BY &2)
  ), (SELECT count(*) tot_tuplas FROM &1);
```

```
@tst_DF_nw prendas "nombre,color" talla
@tst_DF_nw prendas nombre "color,talla"
@tst_DF_nw prendas color talla
```

ejercicios de consultas y normalización

1 *comprobar si, en la tabla* prendas, nombre $\rightarrow\rightarrow$ color *es una DMV*

```
CREATE VIEW nombresFallo AS (
  SELECT DISTINCT nombre FROM prendas PF
   WHERE EXISTS (
     SELECT talla FROM prendas TP
      WHERE nombre = PF.nombre
        AND EXISTS(
              SELECT color FROM prendas WHERE nombre = PF.nombre
              MINUS
              SELECT color FROM prendas WHERE nombre = PF.nombre AND talla = TP.talla
            )
   )
);
```

ejercicios de consultas y normalización

1 *se podría definir un disparador que garantice la integridad*

```
CREATE OR REPLACE TRIGGER tst4FN
AFTER INSERT OR DELETE OR UPDATE ON prendas
DECLARE numFallos NUMBER(9);
BEGIN
 SELECT count(*) INTO numFallos FROM prendas PF
   WHERE EXISTS (
     SELECT talla FROM prendas TP
      WHERE nombre = PF.nombre
        AND EXISTS(
          SELECT color FROM prendas WHERE nombre = PF.nombre
          MINUS
          SELECT color FROM prendas WHERE nombre = PF.nombre AND talla = TP.talla
        )
   );
 IF numFallos > 0 THEN
  raise_application_error( -20501, 'volación de la 4FN');
 END IF;
END;
/
SHOW ERRORS TRIGGER tst4FN
```

ejercicios de consultas y normalización

1 *Obtención de las tuplas que debería haber para que se verificase la DMV*

```
CREATE OR REPLACE VIEW coloresPrenda AS (
  SELECT DISTINCT nombre, color FROM prendas WHERE color IS NOT NULL
);
```

```
CREATE OR REPLACE  VIEW tallasPrenda AS (
  SELECT DISTINCT nombre, talla FROM prendas WHERE talla IS NOT NULL
);
```

```
-- tuplas que debería haber
SELECT CP.nombre, color, talla
 FROM coloresPrenda CP, tallasPrenda TP WHERE CP.nombre = TP.nombre
UNION
SELECT CP.nombre, color, NULL
 FROM coloresPrenda CP WHERE nombre NOT IN (SELECT nombre FROM tallasPrenda)
UNION
SELECT TP.nombre, NULL, talla
 FROM tallasPrenda TP WHERE nombre NOT IN (SELECT nombre FROM coloresPrenda);
```

ejercicios de consultas y normalización

1 *Obtención de las tuplas que faltan para que se verifique la DMV*

```
-- tuplas que faltan
CREATE VIEW tuplasFaltan AS (
 SELECT CP.nombre, color, talla
  FROM coloresPrenda CP, tallasPrenda TP WHERE CP.nombre = TP.nombre
 UNION
 SELECT CP.nombre, color, NULL
  FROM coloresPrenda CP WHERE nombre NOT IN (SELECT nombre FROM tallasPrenda)
 UNION
 SELECT TP.nombre, NULL, talla
  FROM tallasPrenda TP WHERE nombre NOT IN (SELECT nombre FROM coloresPrenda)
 MINUS
 SELECT nombre, color, talla FROM prendas
);
```

ejercicios de consultas y normalización

1 *Obtención de las tuplas que sobran para que se verifique la DMV*

```
-- tuplas que sobran
CREATE VIEW tuplasSobran AS (
 SELECT nombre, color, talla FROM prendas
 MINUS (
 SELECT CP.nombre, color, talla
  FROM coloresPrenda CP, tallasPrenda TP WHERE CP.nombre = TP.nombre
 UNION
 SELECT CP.nombre, color, NULL
  FROM coloresPrenda CP WHERE nombre NOT IN (SELECT nombre FROM tallasPrenda)
 UNION
 SELECT TP.nombre, NULL, talla
  FROM tallasPrenda TP WHERE nombre NOT IN (SELECT nombre FROM coloresPrenda)
 )
);
```

ejercicios de consultas y normalización

1 *Obtención de las tuplas que no verifican la DMV*

```
CREATE TABLE coloresP AS (
  SELECT DISTINCT nombre, color FROM prendas WHERE color IS NOT NULL
);
CREATE TABLE tallasP AS (
  SELECT DISTINCT nombre, talla FROM prendas WHERE talla IS NOT NULL
);
SELECT coloresP.nombre nombre, color, talla
 FROM coloresP, tallasP WHERE coloresP.nombre = tallasP.nombre(+)
UNION
SELECT tallasP.nombre nombre, color, talla
 FROM coloresP, tallasP WHERE tallasP.nombre = coloresP.nombre(+);
```

ejercicios de consultas y normalización

1 *Obtención de las tuplas que no verifican la DMV*

```
-- utilizando sintaxis SQL2
SELECT coloresP.nombre, color, talla
FROM tallasP FULL OUTER JOIN coloresP ON tallasP.nombre = coloresP.nombre;
-- para solventar que en el full join no sale el nombre si color es null
SELECT tallasP.nombre, color, talla
FROM tallasP LEFT OUTER JOIN coloresP ON tallasP.nombre = coloresP.nombre
UNION
SELECT coloresP.nombre, color, talla
FROM tallasP RIGHT OUTER JOIN coloresP ON tallasP.nombre =
coloresP.nombre;
```