

# consultas en álgebra relacional a la B.D. de la vuelta ciclista

1 Dorsal, nombre y equipo del corredor que ha ganado más etapas

$R1(dorsal, num\_etapas) = \text{AGRUPAR}_{count(*)}(\sigma_{ordenMeta = 1}(Llegada); dorsal)$   
 $\equiv$  corredores y nº de etapas que han ganado

$R2(num\_etapas) = \text{AGRUPAR}_{max(num\_etapas)}(R1) \equiv$  nº máximo de etapas que ha ganado un corredor

$R = \Pi_{dorsal, nombre, idEquipo}((R1 \bowtie R2) \bowtie corredor)$

# consultas en SQL a la B.D. de la vuelta ciclista

1 Dorsal, nombre y equipo del corredor que ha ganado más etapas

```
SELECT dorsal, nombre, idEquipo
FROM Corredor
WHERE dorsal IN (
    SELECT dorsal FROM Llegada WHERE ordenMeta = 1
    GROUP BY dorsal
    HAVING count(*) = (SELECT max(count(*)) FROM Llegada
        WHERE ordenMeta = 1
        GROUP BY dorsal)
);
```

# consultas en SQL a la B.D. de la vuelta ciclista

1 Dorsal, nombre y equipo del corredor que ha ganado más etapas, y etapas ganadas

```
SELECT dorsal, nombre, idEquipo,  
       (SELECT count(*) FROM Llegada WHERE ordenMeta = 1 and dorsal = C.dorsal) etapasG  
FROM Corredor  
WHERE dorsal IN (  
    SELECT dorsal FROM Llegada WHERE ordenMeta = 1  
    GROUP BY dorsal  
    HAVING count(*) = (SELECT max(count(*)) FROM Llegada  
                      WHERE ordenMeta = 1  
                      GROUP BY dorsal)  
);
```

# consultas en SQL a la B.D. de la vuelta ciclista

1 Dorsal, nombre y equipo del corredor que ha ganado más etapas, y etapas ganadas

*/\* Si se añade un atributo con el nº de etapas ganadas, se simplifica la consulta:  
\*/*

```
SELECT dorsal, nombre, idEquipo, etapasG  
FROM Corredor  
WHERE etapasG = (SELECT max(etapasG) FROM Corredor);
```

*pero habrá que especificar un disparador que actualice el valor de etapasG :*

# consultas en SQL a la B.D. de la vuelta ciclista

```
CREATE OR REPLACE TRIGGER ganarEtapa
AFTER INSERT OR DELETE OR UPDATE OF ordenMeta ON Llegada
FOR EACH ROW
BEGIN
  IF UPDATING OR DELETING THEN
    IF (:old.ordenMeta = 1) THEN
      UPDATE corredor SET etapasG=etapasG-1 WHERE dorsal=:old.dorsal;
    END IF;
  END IF;
  IF UPDATING OR INSERTING THEN
    IF (:new.ordenMeta = 1) THEN
      UPDATE corredor SET etapasG=etapasG+1 WHERE dorsal=:new.dorsal;
    END IF;
  END IF;
END ganarEtapa;
/
```

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2 *corredores que han puntuado en todos los puertos de primera categoría.*

$$R1 = \prod_{idPuerto} (\sigma_{categoria=1} (Puerto)) \quad \equiv \text{puertos de primera categoría}$$

$$R2 = \prod_{dorsal, idPuerto} (Coronar \bowtie (Hito \bowtie R1)) \quad \equiv \text{corredores que puntúan en algún puerto de 1ª categoría}$$

$$R = R2 \div R1 \quad \equiv \text{corredores que han puntuado en todos los puertos de 1ª categoría}$$

*No es correcto si hay algún puerto de 1ª categoría que no está en ninguna etapa o no ha coronado nadie todavía*

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2 corredores que han puntuado en todos los puertos de primera categoría.

$$R1 = \prod_{idPuerto} (\sigma_{categoria=1} (Puerto)) \quad \equiv \text{puertos de primera categoría}$$

$$R2 = \prod_{dorsal, idPuerto} (Coronar \bowtie (Hito \bowtie R1)) \quad \equiv \text{corredores que puntúan en algún puerto de 1ª categoría}$$

$$R3 = \prod_{idPuerto} (R2) \quad \equiv \text{puertos de 1ª categoría en los que ha puntuado algún corredor}$$

$$R = R2 \div R3 \quad \equiv \text{corredores que han puntuado en todos los puertos de 1ª categoría}$$

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2 corredores que han puntuado en todos los puertos de primera categoría.

$$R1 = \prod_{idEtapa, idHito} (Hito \bowtie \sigma_{categoria=1} (Puerto)) \quad \equiv \text{Hitos correspondientes a puertos de primera categoría}$$

$$R2 = \prod_{dorsal} (R1 \times \prod_{dorsal} (Corredor) - \prod_{idEtapa, idHito, dorsal} (Coronar))$$

$\equiv$  corredores que **no** han puntuado en algún puerto de 1ª categoría

$$R = \prod_{dorsal} (Corredor) - R2 \quad \equiv \text{corredores que han puntuado en todos los puertos de 1ª categoría}$$

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2 corredores que han puntuado en todos los puertos de primera categoría.

```
SELECT dorsal, nombre FROM Corredor D
WHERE NOT EXISTS (
  SELECT * FROM Puerto P
  WHERE categoria = 1
  AND NOT EXISTS (
    SELECT * FROM Coronar C, Hito H
    WHERE C.idEtapa = H.idEtapa AND C.idHito = H.idHito
    AND idPuerto = P.idPuerto AND dorsal = D.dorsal
  )
);
```

*¿y si algún puerto de 1ª categoría no está en ningún hito?*

*¿y si por algún puerto de 1ª categoría no ha pasado nadie todavía?*

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2 corredores que han puntuado en todos los puertos de primera categoría.

```
SELECT dorsal, nombre FROM Corredor D
WHERE NOT EXISTS (
  SELECT * FROM Coronar C, Hito H, Puerto P
  WHERE C.idEtapa = H.idEtapa AND C.idHito = H.idHito
  AND H.idPuerto = P.idPuerto AND categoria = 1
  AND NOT EXISTS (
    SELECT * FROM Coronar C, Hito H
    WHERE C.idEtapa = H.idEtapa AND C.idHito = H.idHito
    AND idPuerto = P.idPuerto AND dorsal = D.dorsal
  )
);
```

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2b) *corredores que han puntuado en todos los puertos de primera categoría.*

```
SELECT dorsal, nombre FROM Corredor D
WHERE NOT EXISTS (
  SELECT idEtapa, idHito FROM Hito H, Puerto P
  WHERE H.idPuerto = P.idPuerto AND categoria = 1
MINUS
  SELECT idEtapa, idHito FROM Coronar WHERE dorsal = D.dorsal
);
```

*problemas similares*

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2b) *corredores que han puntuado en todos los puertos de primera categoría.*

```
SELECT dorsal, nombre FROM Corredor D
WHERE NOT EXISTS (
  SELECT H.idEtapa, H.idHito FROM Coronar C, Hito H, Puerto P
  WHERE C.idEtapa = H.idEtapa AND C.idHito = H.idHito
  AND H.idPuerto = P.idPuerto AND categoria = 1
MINUS
  SELECT idEtapa, idHito FROM Coronar WHERE dorsal = D.dorsal
);
```

• • •

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2c *corredores que han puntuado en todos los puertos de primera categoría.*

*/\* vista con corredores que han puntuado en algún puerto de primera categoría \*/*

```
CREATE VIEW puntuan_puerto_cat1 AS (  
  SELECT DISTINCT dorsal, H.idPuerto  
  FROM Coronar C, Hito H, Puerto P  
  WHERE C.idEtapa = H.idEtapa AND C.idHito = H.idHito  
  AND H.idPuerto = P.idPuerto AND categoria = 1  
);  
  
SELECT dorsal, nombre FROM Corredor D  
WHERE NOT EXISTS (  
  SELECT idPuerto FROM puntuan_puerto_cat1  
  MINUS  
  SELECT idPuerto FROM puntuan_puerto_cat1 WHERE dorsal = D.dorsal  
);
```

# consultas con álgebra relacional a la B.D. de la vuelta ciclista

2d *corredores que han puntuado en todos los puertos de primera categoría.*

```
SELECT dorsal, nombre FROM Corredor D  
WHERE (SELECT count(*) FROM puntuan_puerto_cat1  
  WHERE dorsal = D.dorsal)  
  =(SELECT count(distinct idPuerto) FROM puntuan_puerto_cat1);  
  
/* eliminar los elementos añadidos para la prueba */  
DROP VIEW puntuan_puerto_cat1;
```

• • •

*¿qué + se podría mejorar?*