Pràctica PHP

Darius Natan, Santa
u1994947
P. Inf-1 GEINF

Consultes utilitzades

a) Donar d'alta un vehicle

Dropdown vehicle
SELECT codi, descripcio
FROM grupsvehicles
ORDER BY descripcio;

Dropdown combustible
SELECT descripcio
FROM combustibles
ORDER BY descripcio;

Dropdown propietari
SELECT alias, nom || ' ' || cognoms as nom
FROM usuaris
ORDER BY nom;

Aquestes tres senzilles consultes, les he utilitzat en el moment de entrar les dades, per a mantenir una cohesió de la base de dades, ja que aquestes 3 característiques del vehicle que es vol donar d'alta són claus foranes.
Tot i que podria utiltzar els alias com a valor a mostrar, he considerat més raonable utiltizar una forma més "humana", ja que ningú reconeix una dada pel seu codi, sinó pel nom d'aquesta.

Després, utilitzant les dades del formulari, primer he generat el codi del vehicle utiltzant funcions com *substr()* i *str_replace()* per a manipular strings i generant un nombre aleatori de 3 xifres utiltzant *rand(0,999)* per a posar al final. Seguidament utilitzo la comanda:

SELECT codi, descripcio
FROM vehicles
WHERE codi=:codiVehicle;

Substituint *:codiVehicle* per el codi generat, per comprovar que no existeixi un vehicle amb exactament el mateix codi. Si es genera un codi que ja existeix, es torna a intentar fins que es troba un que no.

Finalment, utilitzant la comanda:

INSERT INTO Vehicles (codi, descripcio, color, consum, datacompra, preu, grupvehicle, combustible, propietari)
VALUES (:codi, :descripcio, :color, :consum, :datacompra, :preu, :grupvehicle, :combustible, :propietari);

I substituint les respectives variables per les dades del formulari (i *:codi* per el codi generat), s'insereix el nou vehicle a la base de dades.

b) Mostrar Vehicles

Mostrar vehicles ha sigut mitjanament senzill, seguint l'exemple proporcionat i modificant la comanda per la necessària, es pot mostrar tots els vehicles. Com a decisió personal, per a fer més còmoda la lectura, he ordenat els vehicles per codi. La comanda utilitzada és la següent:

```
SELECT v.codi, v.descripcio, v.color, v.combustible, v.consum, v.consum*c.preuunitat as costcombustible100km, u.nom, u.cognoms
FROM vehicles v
JOIN combustibles c ON v.combustible=c.descripcio
LEFT OUTER JOIN usuaris u ON v.propietari=u.alias
ORDER BY v.codi;
```

He fet servir un left outer join, perquè encara que en teoria no hauria d'haver cap vehicle sense propietari, no va malament assegurar.

c) Inscripcions en una cursa

Primer de tot, he fet una petita pàgina per seleccionar la cursa. Com que estem fent inscripcions, he de triar només les curses les quals *inicireal* és null, ja que és la columna que separa una cursa ja realitzada (o en progés) i una planificada. La comanda per a les opcions de la tria és senzilla:

SELECT codi, nom
FROM curses
WHERE inicireal IS NULL;

En la mateixa pàgina s'insereix la data i la hora d'inici real de la cursa.
En cambiar de pàgina, afegim les dades de la cursa i l'inici en variables $_SESSION, per a poder afegir múltiples parelles Vehicle-Personatge. També actualitzem la hora d'inici real de la carrera amb la sentència:

UPDATE curses
SET inicireal=TO_TIMESTAMP(:inici, 'YYYY-MM-DD HH24:MI')
WHERE codi=:cursa;

On *:cursa* és substituïda per la opció seleccionada en la pàgina anterior.
Al seleccionar la cursa, podem inscriure la primera parella Vehicle-Personatge. El formulari té 2 entrades, una pel vehicle i una pel Personatge. En el moment de seleccionar el personatge i el vehicle, m'ha semblat raonable, per a facilitar la inscripció, afegir al final del nom l'usuari al què pertany el respectiu Personatge o Vehicle, ja que per a poder inscriure una parella, amdòs elements han de pertànyer al mateix Usuari. Per tant, les sentències de les opcions són les següents:

Dropdown Personatge
SELECT alias,usuari
FROM personatges;

Dropdown Vehicle
SELECT codi, descripcio, propietari
FROM vehicles;

A la pàgina d'inserció a la base de dades. Comprovo que el vehicle i el personatge seleccionat són del mateix usuari utilitzant les sentències:

SELECT usuari FROM personatges WHERE alias=:personatge;
SELECT propietari FROM vehicles WHERE codi=:vehicle;

Finalment, utilitzant les variables de $_SESSION de la cursa seleccionada i les variables del formulari seleccionant el personatge i el vehicle, s'inscriu la parella Vehicle-Personatge utilitzant la comanda:

```
INSERT INTO participantscruses (cursa, vehicle, personatge)
VALUES (:cursa, :vehicle, :personatge);
```

No abans de comprovar si la parella ja ha estat inscrita mitjançant:

```
SELECT *
FROM participantscurses
WHERE cursa=:cursa AND vehicle=:vehicle AND personatge=:personatge;
```

d) Entrar temps de tots els participants

De la mateixa manera que en les inscripcions, primer seleccionem una cursa. En aquest cas, necssitem les curses que han acabat, però que no tenen els temps entrats (ja que les volem entrar ara):

```
SELECT codi, nom
FROM curses
WHERE inicireal IS NOT NULL AND millortemps IS NULL;
```

En el formulari per entrar els temps, utilitzem la següent comanda per a seleccionar els participants de la cursa en qüestió, juntament amb el personatge, vehicle i temps entrat anteriorment. Aquest últim era necessari en la 1a entrega per a modificar rankings de carreres anteriors, però no en la 2a, tot i així, m'ha semblat una bona pràctica. El formulari mostra un camp per entrar el temps per cada usuari de la carrera, tot mostrant el Personatge i el Vehicle utilitzat pel respectiu usuari. La sentència utilitzada és la següent:

```
SELECT p.codi, u.alias, p.personatge, v.descripcio, p.temps, p.vehicle
FROM participantscurses p
JOIN vehicles v ON p.vehicle=v.codi
JOIN usuaris u ON v.propietari=u.alias
WHERE p.cursa =:cursa;
```

Al entrar la informació al formulari i passar a la pàgina següent, primer actualitzem el temps de la parella Vehicle-Personatge amb la comanda:

```
UPDATE participantscurses
SET temps=:temps
WHERE codi=:participant AND cursa=:cursa;
```

Per a la 2a entrega, es demanava la creació d'una taula revisions, i en el moment d'entrar els temps dels participants en aquesta opció, calia inserir una nova revisió en certes condicions. Per a fer això primer hem de comprovar si el Vehicle compleix alguna de les condicions. Una de les condicions era si el Vehicle no n'ha passat mai cap revisió. Per tant, és raonable fer servir una sentència per a poder comprovar aquest fet:

```
SELECT curses, personatge FROM revisions WHERE codiVehicle=:vehicle;
```

Utilitzant aquesta sentència, podem saber si hi ha alguna revisió d'aquest vehicle. Si la sortida és buida, sabem que hem d'inserir una revisió. Una altra condició és que hagin passat més de 3

curses des de la última revisió, per tant, amb la mateixa sentència podem saber el nombre de carreres. Utilitzant aquesta informació, podem entrar a la mateixa branca del *if* que abans. Per útlim, la última condició és que el vehicle hagi abandonat la carrera. Aquesta es pot saber simplement mirant el valor del formulari que l'usuari acaba de emplenar en la pàgina anterior, i si és buit, entrem també en la mateixa branca. Aquesta branca executa la comanda que insereix una nova revisió:

```
INSERT INTO revisions (codiVehicle, data, curses, personatge)
VALUES (:vehicle, :data, :curses, :personatge)
```

No cal utilitzar un to_date en :data, ja que la data la hem extret de la mateixa base de dades, per tant el format ja és correcte.
En la branca contrària (la última revisió és més recent de 3 carreres i el vehicle no ha abandonat la carrera), s'executa la següent revisió, que incrementa el comptador de carreres i actualitza el personatge al actual:

```
UPDATE revisions
SET curses=curses+1, personatge=:personatge
WHERE codivehicle=:vehicle AND data=(
        SELECT MAX(data)
        FROM revisions
        WHERE codiVehicle=:vehicle);
```

Finalment, la pàgina actualitza la taula **Curses** amb el millor temps de la carrera utilitzant la comanda:

```
UPDATE curses
SET millortemps=(
        SELECT MIN(temps)
        FROM participantscurses
        WHERE cursa=:cursa)
WHERE codi=:cursa;
```

e) Consultar els participants d'una cursa

Al fer el selector de les curses, hem de tenir en compte que les úniques curses que tenen participants, són les curses on *inicireal* no és null, ja que significa que aquestes ja han tancat les seves inscripcions. Aconseguim això amb la comanda (utilitzant *order by* per facilitar la lectura):

```
SELECT codi, nom
FROM curses
WHERE inicireal IS NOT NULL
ORDER BY nom;
```

Al seleccionar la cursa, comprovem si ja s'ha acabat, és a dir, si ja s'han entrat els temps dels participants. Podem saber aixo mirant la columna *millortemps* de la taula *curses*. Ja que aquesta només pot tenir valor, si els participants han acabat la carrera. Fem servir la sentència:

```
SELECT codi, millortemps
FROM curses
WHERE codi=:carrera AND millortemps IS NOT NULL;
```

I amb un simple if, comprovem si hi ha alguna fila resultat. Si no hi ha cap, significa que la carrera encara no té ranking, per tant mostrem a la pàgina la capçalera **Llistat de participants** i fem servir la comanda:

```
SELECT v.codi, v.descripcio, p.personatge
FROM participantscurses p
JOIN vehicles v ON p.vehicle=v.codi
WHERE p.cursa =:cursa
```

En el cas que la carrera ja té els rankings, fem servir una altra comanda, i mostrem la **Clasificació de la cursa**. La diferència és que en el darrer cas, mostrem per ordre de temps. És important mencionar el *coalesce* utiltizat, per a tractar valors null i mostrar-los com a **Abandonat** i per a poder substituir un valor numèric amb un de text, cal fer un *cast*. La comanda utilitzada és la següent:

```
SELECT v.codi, v.descripcio, p.personatge, COALESCE(CAST(p.temps AS VARCHAR(9)), 'Abandonat') AS temps
FROM participantscurses p
JOIN vehicles v ON p.vehicle=v.codi
WHERE p.cursa=:cursa
ORDER BY temps;
```

f) Consultar Revisions:

Aquesta opció et porta a un menú secundari per a seleccionar el mètode de consulta de les revisions que té com a operacions disponibles segons 2 dates (inici i final) o segons l'usuari

i) Data

Primer es demana en un formulari la data d'inici i la data final. Després utiltizant com a plantilla el codi de mostrar Vehicles utilitzem una senzilla select per a mostrar totes les revisions entre les dades mencionades ordenades per la data:

```
SELECT *
FROM revisions
WHERE data >= TO_DATE(:datainici, 'YYYY-MM-DD') AND data <=TO_DATE(:datafinal, 'YYYY-MM-DD')
ORDER BY data;
```

És important, una altra vegada, la funció *to_date* ja que és la que permet SQL entendre el format de la data que rebem del formulari d'html.

ii) Usuari

Aquesta opció, molt similar a la seva bessona (consultar les revisions entre dates), demanem el usuari desitjat en un formulari, en aquest cas, un selector. Aquest té les opcions següents:

```
SELECT cognoms || ' ' || nom AS nomUsuari,  alias
FROM usuaris
ORDER BY nomUsuari;
```

Les opcions tenen entre parèntesis el àlies de l'usuari, ja que pot ser útil (sobretot per debugging). Després, en el moment de mostrar (utilitzant, altra vegada, com a plantilla el mostrarVehicles) utilitzem una select sobre un conjunt d'un subcomanda, per a mostrar totes les revisions de tots els vehicles de l'usuari seleccionat:

```
SELECT * FROM revisions
WHERE codiVehicle IN (
        SELECT codi
        FROM vehicles
        WHERE propietari=:usuari);
```