Procediments emmagatzemats i disparadors, per a què són necessaris?

NOM I COGNOMS: Cristian Galán Augé

El Comitè Olímpic de la UOC, encarregat d'organizar els Jocs Atlètics Olímpics UOC, ara que disposa de la base de dades que hem construït a la UOC com a part de l'assignatura Bases de dades per a Data Warehousing, ha obtingut el pressupost necessari per implementar una sèrie de millores. De nou, s'ha posat en contacte amb nosaltres perquè implementeu els requisits que ens han proposat.

Per a la proposta de solució d'aquesta PAC, heu de crear una base de dades nova anomenada dbdw_pec3. Primer executeu el script adjunt DB_olympic_structure.sql. A continuació excuteu el script DB_olympic_data.sql. Ambdós scripts SQL crearan l'estructura i el conjunt de dades necessaris pel desenvolupament d'aquesta PAC.

EXERCICI 1 (30%)

a)

```
SET search_path TO dbdw_pec3;

ALTER TABLE olympic.tb_register
    ALTER COLUMN register_date TYPE TIMESTAMP,
    ALTER COLUMN register_date SET DEFAULT CURRENT_TIMESTAMP;
--  ADD CONSTRAINT ck_date CHECK(register_date = CURRENT_TIMESTAMP);

ALTER TABLE olympic.tb_register
    RENAME register_date TO register_ts;
```

Es canvia la columna registre_date per a que agafi la hora i de natural agafi la data i hora actual.

b)

```
ALTER TABLE olympic.tb_register
    ADD COLUMN register_updated TIMESTAMP;


--FUNCIO PER ESTABLIR VALOR INICIAL
--DROP FUNCTION default_upd()
CREATE FUNCTION fn_register_inserted()
RETURNS trigger AS $$
BEGIN
    --IF (OLD.register_updated IS NULL) THEN
    NEW.register_updated := NEW.register_ts;
    --END IF;
    RETURN NEW;
    END
    $$ LANGUAGE plpgsql;

CREATE TRIGGER tg_register_inserted
BEFORE INSERT ON olympic.tb_register
FOR EACH ROW
EXECUTE PROCEDURE fn_register_inserted();
```

Es van afegint els valors de register_ts a register_updated quan és el primer cop que s'entra el valor.

c)

```
CREATE FUNCTION fn_register_updated()
RETURNS trigger AS $$
BEGIN
    --IF (OLD.register_updated IS NULL) THEN
    NEW.register_updated := CURRENT_TIMESTAMP;
    --END IF;
    RETURN NEW;
    END
    $$ LANGUAGE plpgsql;

CREATE TRIGGER tg_register_updated
BEFORE UPDATE ON olympic.tb_register
FOR EACH ROW
EXECUTE PROCEDURE fn_register_updated();
```

Quan un registre s'actualitza, es posa la data i hora actual, substituïnt la data d'entrada.

EXERCICI 2 (50%)

a) D'aquest primer apartat val la pena comentar l'extensió CITEXT que permet no diferenciar entre majúscules i minúscules. La solució que proposo es que el domini identifiqui que conté el valor "@". Tot i això podria no ser suficient. Malgrat no aconsegueixo implementar la solució fins on arriben els meus coneixements de SQL, aquesta solució valdria parcialment, ens hauríem d'assegurar que el domini es col·loca correctament.

Per internet, he trobat una solució que és la utilitzada a HTML5 (línia posada en comentaris). Aquesta delimita els caràcters que hi poden anar abans i després de l'arroba. Funciona bé però al no haver sapigut arribar jo sol a aquesta solució ni saber com acabar de defensar-la, només l'anomeno per poder-la tenir en compte en un futur.

```
SET search_path TO olympic, "$user", olimpic;
CREATE EXTENSION citext; -- citext permet que les majúscules i les minúscules signifiquin el mateix, tal i com passa als correus electrònics.
CREATE DOMAIN olympic.email_type AS olympic.citext
  --CHECK ( value ~ '^[a-zA-Z0-9.!#$%&''*+/=?^_`{|}~-]+@[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?(?:\.[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?)*$' );
  CHECK (value LIKE '%@%');

ALTER TABLE olympic.tb_collaborator
    --DROP COLUMN email,
    ADD COLUMN email olympic.email_type;

ALTER TABLE olympic.tb_sponsor
    --DROP COLUMN email,
    ADD COLUMN email olympic.email_type;
```

b)

```
CREATE TABLE olympic.tb_athletes_info_log(
    athlete_id CHAR(7),
    discipline_id INT,
    round_number INT,
    athlete_name VARCHAR(50),
    discipline_name VARCHAR(50),
    mark VARCHAR(12),
    rating INT,
    info_log_dt DATE,
    PRIMARY KEY (athlete_id,discipline_id,round_number),
    FOREIGN KEY (athlete_id,discipline_id,round_number) REFERENCES olympic.tb_register(athlete_id,discipline_id,round_number));
```

c) La primera de les accions que programo és la d'INSERT. A destacar com a tònica de l'exercici que a l'utilitzar "TO_CHAR" per convertir les mesures, decideixo que els números podran tenir un màxim de 4 valors sencers i dos de decimals.

```
CREATE OR REPLACE FUNCTION fn_athletes_info()
RETURNS trigger AS $$ BEGIN
    IF (TG_OP = 'INSERT') THEN
        IF (NEW.register_measure IS NOT NULL) THEN
            INSERT INTO olympic.tb_athletes_info_log VALUES (
               NEW.athlete_id, NEW.discipline_id,NEW.round_number,
               (SELECT a.name FROM olympic.tb_athlete a WHERE a.athlete_id = NEW.athlete_id),
               (SELECT d.name FROM olympic.tb_discipline d WHERE d.discipline_id = NEW.discipline_id),
                TO_CHAR(NEW.register_measure,'9999D99'),--les mesures que s'entrin no podran tenir més de 4 valors sencers i 2 decimals.
                NEW.register_position,
                CURRENT_DATE); --de l'enunciat interpreto que no es vol la hora que s'ha entrat sinó una propia pel log.
        END IF;
        IF (NEW.register_time IS NOT NULL) THEN
            INSERT INTO olympic.tb_athletes_info_log VALUES (
               NEW.athlete_id, NEW.discipline_id,NEW.round_number,
               (SELECT a.name FROM olympic.tb_athlete a WHERE a.athlete_id = NEW.athlete_id),
               (SELECT d.name FROM olympic.tb_discipline d WHERE d.discipline_id = NEW.discipline_id),
                TO_CHAR(NEW.register_time,'HH24:MI:SS'),
                NEW.register_position,
                CURRENT_DATE);
        END IF;
        RETURN NEW;
```

A la segona captura es mostra la opció de UPDATE. Aquesta és la part que considero un major repte. Hem de considerar la opció que al log no s'hi hagi guardat anteriorment la fila i per tant, si això no ha sigut així, directament s'han de fer les mateixes accions que a INSERT. Si sí que hi és, llavors no cal entrar la clau primaria i només s'han d'actualitzar els camps nous.

```
    ELSIF (TG_OP = 'UPDATE') THEN
        IF ((SELECT COUNT(*) FROM olympic.tb_athletes_info_log WHERE athlete_id = NEW.athlete_id AND discipline_id = NEW.discipline_id AND round_number=NEW.round_number)=1)
            IF (NEW.register_measure IS NOT NULL) THEN
                UPDATE olympic.tb_athletes_info_log SET
                mark = TO_CHAR(NEW.register_measure,'9999D99'),--les mesures que s'entrin no podran tenir més de 4 valors sencers i 2 decimals.
                rating = NEW.register_position,
                info_log_dt = CURRENT_DATE
                WHERE athlete_id = NEW.athlete_id AND discipline_id = NEW.discipline_id AND round_number=NEW.round_number;
            END IF;
            IF (NEW.register_time IS NOT NULL) THEN
                UPDATE olympic.tb_athletes_info_log SET
                --NEW.athlete_id, NEW.discipline_id,NEW.round_number,
                --(SELECT a.name FROM olympic.tb_athlete a WHERE a.athlete_id = NEW.athlete_id),
                --(SELECT d.name FROM olympic.tb_discipline d WHERE d.discipline_id = NEW.discipline_id),
                mark = TO_CHAR(NEW.register_time,'HH24:MI:SS'),
                rating = NEW.register_position,
                info_log_dt = CURRENT_DATE
                WHERE athlete_id = NEW.athlete_id AND discipline_id = NEW.discipline_id AND round_number=NEW.round_number;
            END IF;
        ELSE
            IF (NEW.register_measure IS NOT NULL) THEN
                INSERT INTO olympic.tb_athletes_info_log VALUES (
                   NEW.athlete_id, NEW.discipline_id,NEW.round_number,
                   (SELECT a.name FROM olympic.tb_athlete a WHERE a.athlete_id = NEW.athlete_id),
                   (SELECT d.name FROM olympic.tb_discipline d WHERE d.discipline_id = NEW.discipline_id),
                    TO_CHAR(NEW.register_measure,'9999D99'),--les mesures que s'entrin no podran tenir més de 4 valors sencers i 2 decimals.
                    NEW.register_position,
                    CURRENT_DATE); --de l'enunciat interpreto que no es vol la hora que s'ha entrat sinó una propia pel log.
            END IF;
            IF (NEW.register_time IS NOT NULL) THEN
                INSERT INTO olympic.tb_athletes_info_log VALUES (
                   NEW.athlete_id, NEW.discipline_id,NEW.round_number,
                   (SELECT a.name FROM olympic.tb_athlete a WHERE a.athlete_id = NEW.athlete_id),
                   (SELECT d.name FROM olympic.tb_discipline d WHERE d.discipline_id = NEW.discipline_id),
                    TO_CHAR(NEW.register_time,'HH24:MI:SS'),
                    NEW.register_position,
                    CURRENT_DATE);
            END IF;
```

Finalment, el DELETE elimina aquella fila del log_record que s'ha eliminat a tb_register (en el cas que existeixi.

```
    ELSIF (TG_OP = 'DELETE') THEN
        IF ((SELECT COUNT(*) FROM olympic.tb_athletes_info_log WHERE athlete_id = OLD.athlete_id AND discipline_id = OLD.discipline_id AND round_number=OLD.round_number)=1)
            DELETE FROM olympic.tb_athletes_info_log VALUES WHERE athlete_id = OLD.athlete_id AND discipline_id = OLD.discipline_id AND round_number=OLD.round_number;
        END IF;
    END IF;
    RETURN NEW;
    END;
$$ LANGUAGE plpgsql;
```

Finalment, decideixo formar dos triggers, un per UPDATE i INSERT i un segon per DELETE. Això és perquè no es pot fer un DELETE "AFTER" sense tocar la primary key de la taula.

```
|
--creo dos triggers per poder borrar del log i evitar errors, degut a la clau primaria del 2b.
CREATE OR REPLACE TRIGGER tg_athletes_info
AFTER INSERT OR UPDATE ON olympic.tb_register
FOR EACH ROW
EXECUTE PROCEDURE fn_athletes_info();

CREATE OR REPLACE TRIGGER tg_athletes_info2
BEFORE DELETE ON olympic.tb_register
FOR EACH ROW
EXECUTE PROCEDURE fn_athletes_info();
--before per poder ser borrat correctament. la opció de cascade a la taula no la contemplo degut als requeriments del 2b.
--l'escenari ideal seria dir en el log que una fila s'ha borrat per tal que quedi constància.
```

d) Pel tercer exercici, he agafat algunes variables de la taula log generada als anterior dos exercicis. Això ho he fet per poder agafar "mark" en el format especificat. La resta de variables bé es podrien treure de tb_register o tb_discipline. He considerat que idealment el log hauria d'estar configurat des del principi de la BBDD i que per tant, hi haurien de ser-hi tots els registres ja que és la funció d'un log.

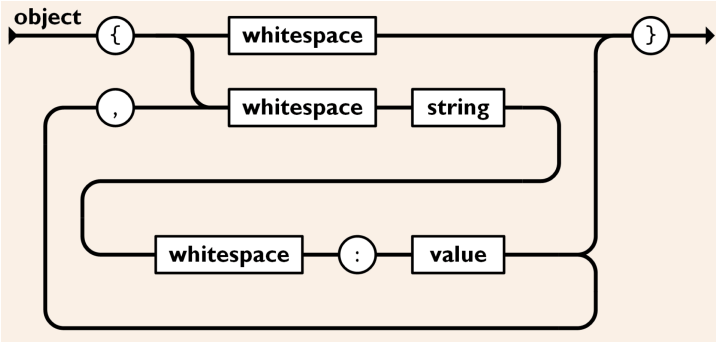
```
CREATE OR REPLACE FUNCTION fn_get_info_by_sponsor(select_date DATE, sponsor olympic.tb_sponsor.name%type)
RETURNS SETOF tipus_dades_sponsor AS $$
DECLARE
    dades_sponsor tipus_dades_sponsor;
BEGIN
    FOR dades_sponsor IN SELECT s.email, s.name, a.name, l.discipline_name, l.round_number, l.mark, l.rating, l.info_log_dt --Al ser un log, hauria de contenir tota la info
        FROM (((olympic.tb_finance f NATURAL JOIN olympic.tb_athlete a) JOIN olympic.tb_sponsor s ON s.name = f.sponsor_name) NATURAL JOIN olympic.tb_athletes_info_log l)
        WHERE sponsor = s.name AND select_date = l.info_log_dt LOOP
    RETURN NEXT dades_sponsor;
    END LOOP;
    RETURN;
END;
$$ LANGUAGE plpgsql;
```

EXERCICI 3 (20%)

Els JSON són un format de representació d'estructures de dades simples i llistes associatives. Aquest estàndard, prové del llenguatge JavaScript i és molt útil per la transmissió de dades estructurades. És un format estandarditzat i el més estés en la transmissió d'informació desde servidors web degut a la senzillesa d'anàlisis a partir de JavaScript, el qual es troba present en la gran majoria de navegadors web.

Per tant, els arxius en format JSON són una bona opció per poder compartir dades ja sigui mitjançant una API o amb un tercer que hagi de fer una manipulació o guardar les dades en algun altre lloc que a la pròpia base de dades.

Els JSON es troba format per un objecte que és constituït pel parell format a partir dels atributs i el seu valor. És un conjunt desordenat on l'objecte es troba delimitat per les claus.



L'array és uns llista ordenada de valors els quals poden ser de qualsevol tipus. Les seves variables es troben separades per comes i delimitades entre claudàtors. És un tipus de dades que es pot trobar present dins dels arxius JSON. La principal diferència que hi ha entre els arrays i els JSON és l'estructuració de dades i la seva ordenació. Els JSON permet la jerarquia i les subcategories entre els diferents atributs mentres que un array simplement permet llistar aquelles característiques de forma seqüencial i amb relacions menys específiques.

B) Entrego l'exercici incomplert per incomprensió del que es demana. Entenc que totes les dades han de ser entrades com a JSON però no aconsegueixo que aquestes s'integrin de forma correcte en una funció ni com s'han de cridar.

Criteris de valoració

En l'enunciat s'indica el pes/valoració de cada exercici.

Per aconseguir la puntuació màxima en els exercicis, cal explicar amb claredat la solució que es proposa.

Format i data de lliurament

El format de l'arxiu que conté la vostra solució pot **ser .pdf, .doc i .docx**. **Per a altres opcions, si us plau, contacteu prèviament amb el vostre consultor.** El nom de l'arxiu ha de contenir el codi de l'assignatura, el vostre cognom i el vostre nom, així com el nombre d'activitat (PAC3).

El fitxer .zip que contingui tots els fitxers de la PAC (tant els fitxers .sql com el document que mostra els resultats de les vostres solucions) l'heu d'enviar a la bústia de Lliurament i registre d'AC disponible a l'aula (apartat Avaluació).

La data límit per lliurar la PAC3 és el **09/12/2021**.

Nota: **Propietat intel·lectual**

Al presentar una pràctica o PAC que faci ús de recursos aliens, s'ha de presentar juntament amb ella un document en el qual es detallin tots ells, especificant el nom de cada recurs, el seu autor, el lloc on es va obtenir i el seu estatus legal: si l'obra està protegida pel copyright o s'acull a cap altra llicència d'ús (*Creative Commons*, llicència GNU, GPL etc.).

L'estudiant s'haurà d'assegurar que la llicència que sigui no impedeixi específicament el seu ús en el marc de la pràctica o PAC. En el cas de no trobar la informació corresponent haurà d'assumir que l'obra està protegida pel copyright. A més, serà necessari adjuntar els fitxers originals quan les obres utilitzades siguin digitals, i el seu codi font si així correspon.