# Base de Donnée TP4

1. Ecrire un code PL/SQL qui permet d’afficher pour chaque sport le nombre total des gymnases qui organisent des séances pour ce sport.

SET SERVEROUTPUT ON;

declare

cursor cor is select s.idsport, libelle, count(distinct ss.idgymnase) as nbgym  from sports         s, seances ss

             where s.idsport=ss.idsport

             group by s.idsport, libelle;

begin

for item in cor

loop

dbms\_output.put\_line('sport '||item.libelle||' is orgnized by '||item.nbgym||'th gym');

end loop;

exception

when NO\_DATA\_FOUND  then dbms\_output.put\_line('DB does not have data');

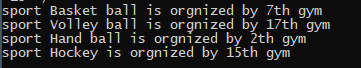
when others then dbms\_output.put\_line('Error : '||sqlcode||sqlerrm);

close cor;

end;

/

Voici le résultat de l'exécution :



1. Ajouter la contrainte suivante : l’âge d’un sportif doit être entre 20 ans et 45 ans. On veut augmenter l’âge des sportifs par 5 ans. Ecrire une procédure qui augmente l’âge de chaque sportif. Désactiver la contrainte d’intégrité pour effectuer les mises à jour. Afficher pour chaque sportif son nouvel âge.
   1. Ajout de la contrainte :



* 1. Désactivation de la contrainte :



* 1. La procédure :

CREATE OR REPLACE PROCEDURE INCR\_AGE is

cursor CR is select IdSportif,Nom,Prenom,Age

from Sportifs;

New\_age  Sportifs.Age %TYPE;

BEGIN

FOR item in cr

LOOP

    New\_age:= item.Age + 5;

    dbms\_output.put\_line('The Player :' ||item.Nom||  's age chnaged from ' ||item.Age|| ' years old to ' ||New\_age|| ' years old ');

    Update SPORTIFS set Age = New\_age where IdSportif = item.IdSportif;

END LOOP;

Exception

When NO\_DATA\_FOUND then dbms\_output.put\_line('Il n y pas de sportifs');

end INCR\_AGE;

/

Show errors PROCEDURE INCR\_AGE;

SET SERVEROUTPUT ON;

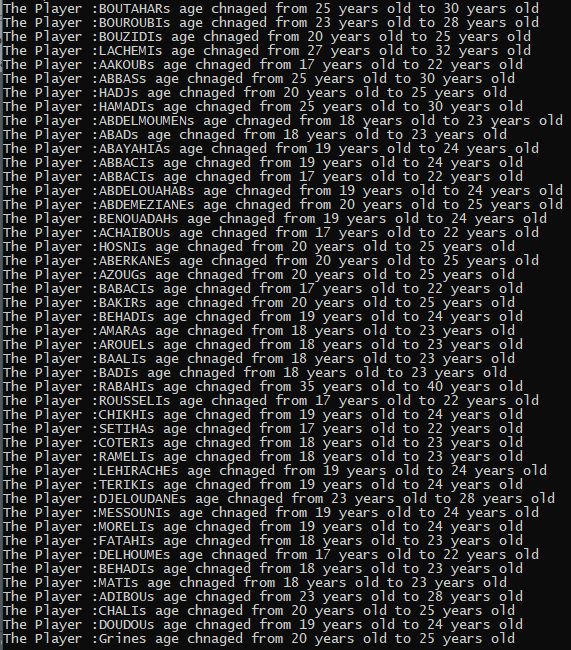
Begin

INCR\_AGE;

End;

/

Résultat de l'exécution :



* 1. Réactivation de la contrainte:



1. Ecrire une procédure qui effectue cet affichage : Pour chaque gymnase, donner par jour d’ouverture, les horaires des premières et dernières séances. Tester la procédure pour les gymnases de superficie >400 m².

D'abord nous commencerons par créer une procédure permettant d'afficher La première et la dernière séance pour un gymnase donné :

CREATE OR REPLACE Procedure Timing (IdGym INTEGER, jour IN VARCHAR2) AS

    cursor ASCE is

    select \* from (

        Select x.IdGymnase, y.Jour, y.Horaire from Gymnases x, Seances y

        where x.IdGymnase = y.IdGymnase and x.IdGymnase = IdGym and y.Jour = jour

        GROUP BY x.IdGymnase,y.Jour,y.Horaire

        ORDER BY x.IdGymnase,y.Horaire asc)

    where ROWNUM = 1;

    cursor DESCE is

    select \* from (

        Select x.IdGymnase, y.Jour, y.Horaire from Gymnases x, Seances y

        where x.IdGymnase = y.IdGymnase and x.IdGymnase = IdGym and y.Jour = jour

        GROUP BY x.IdGymnase,y.Jour,y.Horaire

        ORDER BY x.IdGymnase,y.Horaire DESC)

    where ROWNUM = 1;

    Test Seances %ROWTYPE;

    BEGIN

    for item in ASCE

    loop

    dbms\_output.put\_line('The 1st session of GYM ' ||item.idgymnase ||' in '||jour|| ' at '||item.horaire);

    end loop;

    for item in DESCE

    loop

    dbms\_output.put\_line('The last session of GYM '||item.idgymnase ||' in '|| jour || ' at '||item.horaire);

    end loop;

    End;

    /

Puis nous ferons appel à une autre procédure qui fera le parcours de tous les gymnases (dont la superficie>400 m²)

Et pour chaque instance de Gymnase nous ferons appel à notre première procédure :

create or Replace procedure JUMP(IdGym INTEGER) as

    cursor days is

    Select distinct jour from seances

    where IdGymnase = IdGym;

    BEGIN

    for item in days

    loop

        dbms\_output.put\_line(item.jour);

        Timing(IdGym,item.jour);

    end loop;

    end;

    /

CREATE OR REPLACE Procedure WALK AS

    Cursor cr is

    Select x.IdGymnase from Gymnases x, Seances y

    where x.IdGymnase = y.IdGymnase and x.surface > 400;

    BEGIN

    for i in cr

    loop

    JUMP(i.IdGymnase);

    end loop;

    end;

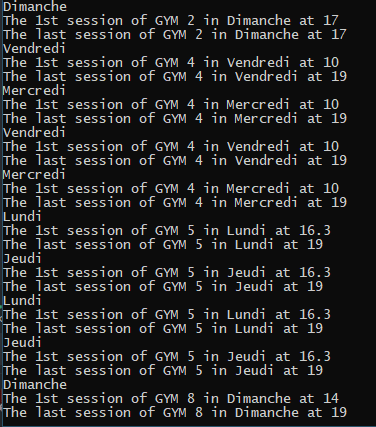
    /

Show errors PROCEDURE WALK;

set serveroutput on;

execute WALK;

Les résultats d'exécution :



1. Ecrire une fonction qui retourne, pour chaque sportif donné, le nombre des sports entrainés. Exécuter la fonction pour plusieurs sportifs.

Code de la fonction :

Create or REPLACE FUNCTION SPORT\_NMB (Idc INTEGER) RETURN INTEGER AS

    Cursor COR is

        Select IdSportifEntraineur,Count(IdSportifEntraineur) as temp ,Nom,Prenom from (

        Select x.IdSportifEntraineur,y.Nom,y.Prenom from Sportifs y, Entrainer x,Sports z

        where x.IdSportifEntraineur = y.IdSportif and x.IdSport = z.IdSport)

        Group by IdSportifEntraineur,Nom,Prenom;

    Begin

    for i in COR

    loop

    IF i.IdSportifEntraineur = Idc

    THEN RETURN i.temp;

    END IF;

    END LOOP;

    END;

    /

SET SERVEROUTPUT ON

Declare

    Cursor oui IS Select distinct(x.IdSportifEntraineur) as coach,y.Nom,y.Prenom from Sportifs y, Entrainer x,Sports z

        where x.IdSportifEntraineur = y.IdSportif and x.IdSport = z.IdSport;

    IdCoach  INTEGER;

BEGIN

    for j in oui

    loop

        IdCoach := SPORT\_NMB(j.coach);

        dbms\_output.put\_line('Coach '||j.Nom||' '|| j.Prenom||' trained '||IdCoach|| ' .');

    end loop;

    END;

    /

Résultat d'exécution :



1. Créer une procédure qui permet d’ajouter une séance à partir de tous les attributs nécessaires. N’oublier pas de vérifier l’unicité de la clé et l’existence de clé étrangère vers les tables référencées. Affichez les messages d’erreurs en cas de problèmes.

CREATE OR REPLACE PROCEDURE Insert\_Seance(IdGymnase1 in Seances.IdGymnase%TYPE,Idsport1 in Seances.IdSport%TYPE, IdSportifEntraineur1 in Seances.IdSportifEntraineur%TYPE,Jour1 in Seances.Jour%TYPE,Horaire1 in  Seances.Horaire%TYPE,Duree1  in Seances.Duree%TYPE) AS

Cursor cr is Select count(idgymnase) as nb from seances

    Where IdGymnase=IdGymnase1

    And IdSport =Idsport1

    And IdSportifEntraineur=IdSportifEntraineur1

    And horaire=Horaire1

    And jour=Jour1 ;

Cursor cr1 is Select count(IdGymnase) as nbgym from Gymnases

    Where IdGymnase = IdGymnase1;

Cursor cr2 is Select count(IdSport) as nbsport  from Sports

    Where  IdSport = Idsport1;

Cursor cr3 is Select count(IdSportif) as nbtrain from Sportifs

    Where IdSportif = IdSportifEntraineur1;

excpt1 EXCEPTION;

excpt2 EXCEPTION;

x INT;

Begin

for i in cr

loop

IF i.nb!=0 THEN

    RAISE excpt1;

END IF;

END LOOP;

for j in cr1

loop

IF j.nbgym =0  THEN

    RAISE excpt2;

END IF;

END LOOP;

for j in cr2

loop

IF j.nbsport =0  THEN

    RAISE excpt2;

END IF;

END LOOP;

for j in cr3

loop

IF j.nbtrain =0  THEN

    RAISE excpt2;

END IF;

END LOOP;

INSERT INTO Seances

VALUES(IdGymnase1,Idsport1,IdSportifEntraineur1,Jour1,Horaire1,Duree1);

COMMIT;

EXCEPTION

WHEN excpt1  THEN dbms\_output.put\_line('Erreur : Contrainte d unicité non respecté ');

WHEN excpt2  THEN dbms\_output.put\_line('Erreur : Contrainte de clé étrangère non respecté ');

END;

/

Les résultats :

