**TP : LES PROCEDURES STOCKEES**

**Objectif** A4.1.1Proposition d'une solution applicative

**Enoncés** Vous utiliserez la base de données **CALAO**

**Exercice 1 : Procédure permettant de supprimer une espèce.**

**Objectif On souhaite supprimer une espèce**

**Remarque** il ne faut pas oublier que des soignants peuvent s'occuper de l'espèce que l'on souhaite supprimer

==> La procédure contient deux ordres de suppression les soins apportés à l'espèce table **SOIGNER**

et ensuite l’espèce table **ESPECE**

**Ce qu'il faut faire précédemment :**

* ajouter 1 enregistrement "Gazelle" dans la table ESPECE , on repère le *codeEspece* associé
* ajouter deux tuples dans la table SOIGNER pour l'espèce GAZELLE

**Rappel** DELETE FROM *nomTable* FROM *condition*

**Temps 1 : on crée la procédure qui supprime les enregistrements des tables pour le code *espece = @codeEspece***

**CREATE PROCEDURE** **SuppressionEspece** (@codeEspece int) **AS**

delete from SOIGNER

where codeEspece = @codeEspece

delete from ESPECE

where codeEspece = @codeEspece

**Temps 2 : on l'exécute en lui passant la valeur du paramètre**

**EXECUTE** **SuppressionEspece** **18**

Comme leur nom l'indique, les procédures sont stockées. Où les trouve-t-on ?

**Exercice 2 : Ajouter un animal d'une nouvelle espèce**

**Objectif On veut créer une nouvelle espèce '*Impala*' et un nouvel animal '*Venusia*' de cette espèce**

Les attributs de la nouvelle espèce seront passés en paramètres. Cette procédure affecte un nouvel animal à cette espèce, les informations de l'animal sont également passés en paramètres.

**Remarque :** On doit d’abord récupérer le code existant dans les espèces, lui ajouter 1 et ensuite on peut faire l'ajout d'une nouvelle espèce. Puisqu’on crée une nouvelle espèce, on est sûr que le nom de cet animal n’existe pas déjà pour cette espèce donc pas de soucis.

**CREATE PROCEDURE** **CreationEspeceEtAnimal** (@nomEspece char(32), @nomBapteme char(32), @sexe char(32), @dateNaissance int, @datedeces int) **AS**

DECLARE @codeMaxEspece int

SET @codeMaxEspece = (SELECT MAX(codeEspece) from ESPECE)

SET @codeMaxEspece = (@codeMaxEspece + 1)

insert into ESPECE values (@codeMaxEspece, @nomEspece)

insert into ANIMAL values (@codeMaxEspece, @nomBapteme, @sexe, @dateNaissance, @datedeces);

**EXECUTE** **CreationEspeceEtAnimal** 'Impala','Venusia','F',2012,''

**Exercice 3 : Retourner l'âge moyen des animaux d'une espèce avec RETURN**

**Objectif Age moyen des animaux d'une espèce**

Procédure permettant de retourner l’âge moyen des animaux d’une espèce en particulier, dont le code est passé en paramètre. Cette procédure utilisera l’instruction RETURN.

**CREATE PROCEDURE** AgeMoyenEspece (@codeEspece int) **AS**

DECLARE @AgeMoyen int

SET @AgeMoyen = (SELECT AVG(2012 - dateNaissance) FROM ANIMAL

WHERE CODEESPECE = @codeEspece)

**RETURN** @AgeMoyen

DECLARE @Age int

**EXECUTE** @Age = AgeMoyenEspece 1

**PRINT** @Age

**Exercice 4 : Retourner l'âge moyen des animaux d'une espèce avec OUTPUT**

Idem que la 3. avec l’instruction OUTPUT.

Attention il ne faut pas oublier la virgule entre les arguments.

**CREATE PROCEDURE** **AgeMoyenEspeceAvecOutput** (@codeEspece int, @AgeMoyen int OUTPUT) **AS**

SET @AgeMoyen = (SELECT AVG(2012 - dateNaissance) FROM ANIMAL

WHERE CODEESPECE = @codeEspece)

DECLARE @Age int

**EXECUTE** **AgeMoyenEspeceAvecOutput** 1, @Age **OUTPUT**

**PRINT** @Age

**Exercice 5 : Les aliments qui composent les menus d'une espèce**

Procédure permettant d'afficher les aliments qui composent les menus types d'une espèce particulière. On affiche l'aliment qui compose le menu. Le code espèce est présent en tant que clé étrangère dans la table **MENUTYPE** ==> seule cette tables est utile au traitement.

On utilise un **CURSEUR**

**CREATE PROCEDURE** ListeMenuEspece (@codeEspece int, @ListeMenu **CURSOR VARYING OUTPUT**) **AS**

SET @ListeMenu = **CURSOR FOR**

SELECT LibelleAlimentCompose FROM MENUTYPE WHERE codeEspece = @codeEspece

**OPEN** @ListeMenu

DECLARE @LibelleMenu CURSOR

DECLARE @LibelleAliment char(32)

**EXECUTE** ListeMenuEspece 1,@LibelleMenu OUTPUT

**FETCH** @LibelleMenu INTO @libelleAliment

PRINT 'Les menus pour cette espèce sont : '

**WHILE** **(@@FETCH\_STATUS = 0) ---- *test de fin de curseur ----***

**BEGIN**

**PRINT** ' --> ' + @libelleAliment

**FETCH** @LibelleMenu **INTO** @libelleAliment

**END**

**CLOSE** @LibelleMenu

**DEALLOCATE** @LibelleMenu

**Exercice 6 : Espèce à la charge d’un soignant dont on fait passer le nom et le prénom en paramètre.**

**CREATE PROCEDURE** ListeEspeceSoignerParUnSoignant (@nomSoignant char(32), @prenomSoignant char(32), @ListeEspece **CURSOR VARYING OUTPUT**) **AS**

**SET** @ListeEspece = **CURSOR FOR**

SELECT ESPECE.LIBELLE

FROM ESPECE INNER JOIN SOIGNER ON ESPECE.CODEESPECE = SOIGNER.CODEESPECE INNER JOIN SOIGNANT ON SOIGNER.CODESOIGNANT = SOIGNANT.CODESOIGNANT

WHERE NOMSOIGNANT = @nomSoignant

AND PRENOMSOIGNANT = @prenomSoignant

**OPEN** @ListeEspece

DECLARE @listeDesEspeces CURSOR

DECLARE @libelleEspece char(32)

**EXECUTE** ListeEspeceSOignerParUnSoignant 'Roulous', 'Maxime', @ListeDesEspeces OUTPUT

**FETCH** @listeDesEspeces INTO @libelleEspece

**PRINT** 'Ce soignant prend soins des espèces suivantes : '

**WHILE** (@@FETCH\_STATUS = 0)

**BEGIN**

**PRINT** ' --> ' + @libelleEspece

**FETCH** @listeDesEspeces INTO @libelleEspece

**END**

**Exercice 7 : Les espèces qui ne cohabitent pas**

Liste des espèces qui ne cohabitent pas avec une autre espèce dont on fait passer le nom en paramètre.

**CREATE PROCEDURE** **ListeEspeceNeCohabitePas** (@codeEspece int) **AS**

DECLARE ListeEspece CURSOR FOR

SELECT libelle

FROM ESPECE

WHERE codeespece not in (select CODEESPECE1

from COHABITER

where CODEESPECE = @codeEspece)

DECLARE @nomEspece char(32)

OPEN ListeEspece

PRINT 'Liste des especes qui ne cohabitent pas avec cette espèce : '

FETCH ListeEspece INTO @nomEspece

WHILE (@@FETCH\_STATUS = 0)

BEGIN

PRINT ' --> ' + @nomEspece

FETCH ListeEspece INTO @nomEspece

END

CLOSE ListeEspece

DEALLOCATE ListeEspece

**EXECUTE**  ListeEspeceNeCohabitePas 2

**Exercice 8 : Les espèces qui ne cohabitent pas**

Procédons par étapes.

On veut connaître le nombre d'objets prêtés et placés dans l'enclos N° 1

Quelle est la requête ?

SELECT nomObjet

FROM TYPEOBJET, OBJETPRETER

where TYPEOBJET.codeObjet = OBJETPRETER.codeObjet

and OBJETPRETER.codeEnclos = 1;

On veut connaître le nombre d'objets achetés et placés dans l'enclos N° 1

Quelle est la requête ?

SELECT nomObjet

FROM TYPEOBJET, OBJETACHETER

where TYPEOBJET.codeObjet = OBJETACHETER.codeObjet

and OBJETACHETER.codeEnclos = 1;

**Mais comment peut-on connaître la liste des objets achetés et prêts pour l'enclos N° 1 ????**

***Ecrire la procédure stockée qui permet d'afficher la liste des objets dans un enclos***

**CREATE PROCEDURE** ListeObjetsEnclos ( @unEnclos int,

@ListePreter CURSOR VARYING OUTPUT,

@ListeAcheter CURSOR VARYING OUTPUT) **AS**

SET @ListePreter = CURSOR FOR

SELECT nomOBjet FROM TYPEOBJET, OBJETPRETER

WHERE TYPEOBJET.codeObjet = OBJETPRETER.CodeObjet

AND OBJETPRETER.codeEnclos = @unEnclos

OPEN @ListePreter

SET @ListeAcheter = CURSOR FOR

SELECT nomObjet FROM TYPEOBJET, OBJETACHETER

WHERE TYPEOBJET.codeObjet = OBJETACHETER.codeObjet

AND OBJETACHETER.codeEnclos = @unEnclos

OPEN @ListeAcheter

**DECLARE** @nomObjetAchete char(32)

DECLARE @nomObjetPrete char(32)

DECLARE @ListeRetourObjetAcheter CURSOR

DECLARE @ListeRetourObjetPreter CURSOR

EXECUTE ListeObjetsEnclos 1, @ListeRetourObjetPreter OUTPUT, @ListeRetourObjetAcheter OUTPUT

FETCH @ListeRetourObjetPreter INTO @nomObjetPrete

PRINT 'Liste des objets pretés pour cet enclos :'

WHILE (@@FETCH\_STATUS = 0)

BEGIN

PRINT ' --> ' + @nomObjetPrete

FETCH @ListeRetourObjetPreter INTO @nomObjetPrete

END

FETCH @ListeRetourObjetAcheter INTO @nomObjetAchete

PRINT 'Liste des objets achetés pour cet enclos :'

WHILE (@@FETCH\_STATUS = 0)

BEGIN

PRINT ' --> ' + @nomObjetAchete

FETCH @ListeRetourObjetAcheter INTO @nomObjetAchete

END

close @ListeRetourObjetPreter

deallocate @ListeRetourObjetPreter

close @ListeRetourObjetAcheter

deallocate @ListeRetourObjetAcheter