Dictionnaire 1 :

select initcap(username) as "Nom de l'utilisateur", to\_char(created, 'DD/MM/YYYY HH24:MI') as "Date de création ", default\_tablespace as "Tbl Sp Dflt",

coalesce(expiry\_date, created + to\_yminterval('02-00') ) as "Date d'expiration"

from user\_users;

Dictionnaire 2 :

Select object\_name

From (select object\_name

from all\_objects

where object\_type='VIEW' and object\_name like 'USER@\_%’ escape ‘@’ order by object\_name ASC)

where rownum<6;

Dictionnaire 3 :

select table\_name as "Nom de la table", index\_name as "Nom de l'index",

CASE uniqueness when 'UNIQUE' then 'OUI' else 'NON' end Unicité

from user\_indexes

order by table\_name ASC;

Dictionnaire 4 :

select object\_type as "type de l'object", object\_name as "nom de l'objet", to\_char(created, 'DD/MM/YYYY HH24:MI') as "date de création" ,

case status when 'VALID' then 'YES' else 'NO' end Statut

from User\_objects

order by object\_type;

Dictionnaire 5 :

SELECT USER\_CONS\_COLUMNS.COLUMN\_NAME AS "ColumnName", USER\_CONSTRAINTS.CONSTRAINT\_NAME AS "ConstraintName",

CASE CONSTRAINT\_TYPE

WHEN 'R' THEN 'REFERENTIEL'

WHEN 'C' THEN 'CHECK'

WHEN 'P' THEN 'PRIMARY'

WHEN 'U' THEN 'UNIQUE'

END AS "Type",

CASE USER\_CONSTRAINTS.DEFERRED

WHEN 'IMMEDIATE' THEN 'NO'

ELSE 'YES'

END AS "Differee",

STATUS AS "Statut", SEARCH\_CONDITION AS "Condition"

FROM USER\_CONSTRAINTS

INNER JOIN USER\_CONS\_COLUMNS ON (USER\_CONSTRAINTS.CONSTRAINT\_NAME = USER\_CONS\_COLUMNS.CONSTRAINT\_NAME)

ORDER BY USER\_CONS\_COLUMNS.COLUMN\_NAME;

Dictionnaire 6:

SET ECHO OFF

SET TERMOUT OFF

SET FEEDBACK OFF

SET VERIFY OFF

SET HEADING OFF

SET NEWPAGE NONE

SPOOL C:\Users\Roberto\OneDrive\Scuola\LEONARD\Labo\Labo1\Dico

SELECT ' DROP TABLE ' || table\_name || 'CASCADE CONSTRAINTS';

FROM USER\_TABLES;

SPOOL OFF

SET TERMOUT ON

SET FEEDBACK ON

SET VERIFY ON

SET HEADING ON

SET ECHO ON

SQLPLUS2 :

SELECT NOM, NOMDEP

FROM DEPARTEMENTS

INNER JOIN EMPLOYES ON ( DEPARTEMENTS.NUMDEP = EMPlOYES.NUMDEP)

WHERE nom ='&nom%';

SQLPLUS3 :

SELECT &colonne

FROM EMPLOYES

WHERE NumSecu NOT IN (SELECT NumSecu FROM EMPPRO);

SQLPLUS4 :

SELECT NOMDEP FROM DEPARTEMENTS

INNER JOIN EMPLOYES ON (DEPARTEMENTS.NUMDEP = EMPLOYES.NUMDEP)

GROUP BY NOMDEP

HAVING COUNT(\*) > &nbdepa;

SQLPLUS5 :

COLUMN nom HEADING "CHEF" FORMAT A20

COLUMN chefde HEADING "CHEF DE" FORMAT A20

COLUMN employes HEADING "EMPLOYE" FORMAT A20

BREAK ON nom skip 1

COMPUTE COUNT OF nom ON nom

select c1.nom ,('CHEF DE') AS chefde ,c2.nom as employes

from EMPLOYES c1 , employes c2

where c1.numsecu = c2.numchef

order by nom;

SQLPLUS6 :

SET PAGESIZE 50

SET LINESIZE 100

TTITLE "User: InfoSoft"

COLUMN NOMDEP HEADING "Nom du departement" FORMAT A30

COLUMN NOMPRO HEADING "Nom du projet" FORMAT A20

COLUMN NOM HEADING "Nom de l'employe" FORMAT A20

BREAK ON NOMDEP SKIP 1

COMPUTE COUNT OF NOMDEP ON NOMDEP

SET COLSEP " "

SELECT NOMDEP, NOMPRO, NOM

FROM DEPARTEMENTS

INNER JOIN PROJETS ON (PROJETS.NUMDEP = DEPARTEMENTS.NUMDEP)

INNER JOIN EMPPRO ON (EMPPRO.NUMPRO = PROJETS.NUMPRO)

INNER JOIN EMPLOYES ON (EMPLOYES.NUMSECU = EMPPRO.NUMSECU)

ORDER BY NOMDEP;

Concept De Base 1:

START ListeEmployesProjets.sql

/ #en cas de problème pour lancer la procédure

1) select to\_char(created, 'DD/MM/YYYY HH24:MI') as "date de création" , status

FROM USER\_OBJECTS

WHERE OBJECT\_TYPE = ‘PROCEDURE’

AND OBJECT\_NAME = ‘ListeEmployesProjets’ ;

2) ALTER TABLE Employes ADD Prenom2 VARCHAR2 (30)

On modifie le niveau logIque

3) Idem 1

Le statut est ‘invalid’ parce qu’on a modifié le contenu de la table.

4) EXECUT ListeEmployesProjets parce qu’on a une indépendance entre le niveau logique et applicatif = INDEPENDANCE LOGIQUE. C’est une indépendance entre les données et le programme.

Concept De Base 2:

1) CREATE UNIQUE INDEX IndexProc ON Employes

(Nom, Prenom) ;

Le niveau physique

2) Augmenter la rapidité d’accès aux tuples (nom et prénom dans ce cas).

3) Car un index ne modifie pas le niveau logique = les données sur lequel se trouve la procédure

Concept De Base 3:

1)ALTER TABLE Employes

ADD CONSTRAINT CONTRAINTENOM CHECK (Nom IS NOT NULL);

2)On modifie le niveau logique, mais one ne modifie rien dont la procédure à besoin. On ne modifie pas le contenu de la table. La contrainte doit déjà être respecter sinon il faut modifier et cela créera un problème.

Concept De Base 4:

1) ALTER TABLE Employes DROP COLUMN Nom ;

2) select status

FROM USER\_OBJECTS

WHERE OBJECT\_TYPE = ‘PROCEDURE’

AND OBJECT\_NAME = ‘ListeEmployesProjets’ ;

3) Parce qu’on a modifié les données, le niveau logique.

**EXERCICE D’ALGEBRE RELATIONNELLE**

AlgebreRelationnelle1 :

NUMSECU

935132

123456

864231

121212

999999

192357

654321

192357

123457

NULL

Opérateur ensembliste :

Union

Opérateur relationnel :

PROJECTION

Opérateurs additionnels :

Représentez les ensembles :

NumSecu des chefs de départements

NumChef de la tabe Employés

Quelle est la question a poser pour obtenir ce résultat :

Donnez le numéro de sécurité sociale de tous les employés qui sont chef ou responsables d’un département

Requete sql :

SELECT NumSecu

FROM Departements

UNION

SELECT CASE WHEN NumChef IS NULL THEN ‘NULL’ ELSE NumChef END

FROM Employes;

AlgebreRelationnel2 :

NumSecu

999999

Opérateur ensembliste :

-(difference)

Opérateur relationnel :

PROJECTION

Opérateurs additionnels :

Representez les ensembles :

NumSecu des chefs de départements

-

NumChef de la tabe Employés

Quelle est la question a poser pour obtenir ce résultat :

Afficher les numéros de sécurité de tous les employés responsables d’un départements mais qui ne sont pas un chef de départements

Requete sql :

SELECT NumSecu

FROM Departements

MINUS

SELECT NumChef

FROM Employes;

AlgebreRelationnel3 :

NumDep Nom

D0001 REDFORD

D0001 MOORE

D0001 MONROE

D0002 REDFORD

D0002 MOORE

D0002 MONROE

D0003 REDFORD

D0003 MOORE

D0003 MONROE

D0004 REDFORD

D0004 MOORE

D0004 MONROE

D0005 REDFORD

D0005 MOORE

D0005 MONROE

Opérateur ensembliste :

Produit cartésien (X)

Opérateur relationnel :

PROJECTION

SELECTION

Opérateurs additionnels :

Quelle est la question a poser pour obtenir ce résultat :

Combinez tous les numéros de départements avec le nom de chacun des employés ayant un numéro de sécurité sociale supérieur à 900000

Requête Sql :

Select departements.numdep, nom

From departements, employes

Where employes.numsecu > 900000 ;

Algebrerelationnelle4 :

935132

123456

864231

121212

Opérateur ensembliste :

Opérateur relationnel :

PROJECTION

Opérateurs additionnels :

Intersection (∩)

Quelle est la question a poser pour obtenir ce résultat :

Affichez les numeros de securite des employés qui sont chef d’un départements et responsable

Requête sql :

SELECT NumChef

FROM Employes

INTERSECT

SELECT NumSecu

FROM Departements;

SELECT DISTINCT Departements.NumSecu

FROM Departements, Employes

WHERE Departements.NumSecu = Employes.NumChef;

Algebrerelationnelle5 :

NomDep

Applications Main Frame

Opérateur ensembliste :

Opérateur relationnel :

PROJECTION

SELECTION

Opérateurs additionnels

JOINTURE

Quelle est la question a poser pour obtenir ce résultat :

Affichez les noms des departements dont l’employé de Nom ‘DE NIRO’ est associé

Requête sql :

SELECT NomDep

FROM Departements, Employes

WHERE Departements.NumDep = Employes.NumDep

AND UPPER(Nom) = ‘DE NIRO’;

Algebrerelationnelle6 :

999999

935133

192357

Opérateur ensembliste :

Opérateur relationnel :

PROJECTION

SELECTION

Opérateurs additionnels :

JOINT\_EXT LEFT (jointure externe gauche)

Quelle est la question a poser pour obtenir ce résultat :

Donnez le numéro de sécurité scale des employés qui n’ontpas de projet en cours

Requête sql:

SELECT Employes.NumSecu

FROM Employes LEFT JOIN EmpPro

ON Employes.NumSecu = EmpPro.NumSecu

WHERE EmpPro.NumSecu IS NULL;

Algebrerelationnelle7 :

STALLONE

CURTIS

HOFFMAN

BOHRINGER

BERTHIER

DE NIRO

CELARIE

BEART

CLAVIER

LAFONT  
REDFORD

Opérateur ensembliste :

Opérateur relationnel :

PROJECTION

Opérateurs additionnels :

JOINT\_EXT RIGHT (jointure externe gauche)

Quelle est la question a poser pour obtenir ce résultat :

Donnez le nom de tous les employés qui travaillent sur un projet

Requête sql :

SELECT DISTINCT Nom

FROM Employes RIGHT JOIN EmpPro

ON Employes.NumSecu = EmpPro.NumSecu ;

Integrite1 :

EXECUTE GestionProjets.AjouterProjet (‘p10345’, ‘Test intégrité 1’, TO\_DATE (‘19/10/2016’, ‘DD/MM/YYYY’), ‘LIEGE’, ‘d00001’)

Integrite2

EXECUTE GestionProjets.AjouterProjet (‘p11111’, ‘Test intégrité 2’, TO\_DATE(‘19/10/2016’, ‘DD/MM/YYYY’) , ‘LIEGE’, ‘d11111’)

Integrite3

EXECUTE GestionProjets.SupprimerProjet (‘p10348’) ;

Integrite4

EXECUTE GestionProjets.AjouterProjet (‘p11111’, ‘test integrité 1’, TO\_DATE (‘19/10/2016’, ‘DD/MM/YYYY’), ‘LIEGE’, NULL)

Integrite5

EXECUTE GestionProjets.AjouterProjet (‘p11122’, ‘test integrité 1’, ‘Salut’, ‘LIEGE’, ‘d00004’)

Contraintes Applicatives :

EmployesSexe

Integrite de Reference :

REFEmpNumDepINDepartements

REFNulChefINEmployes

REFDeptNumSecuINEmployes

REFProjetsNumDepINDepartements

REFEmpProNumDepINDepartements

REFEmpProNumProINProjets

Integrite de Relations :

CPEmployesNumSecu

CPDepartements

CPProjets

CPEmpPro

LDD1 :

--LDD Hopital

drop table Patients cascade constraints;

drop table Specialites cascade constraints;

drop table Services cascade constraints;

drop table Medecins cascade constraints;

drop table Chambres cascade constraints;

drop table PatientsChambres cascade constraints;

create table Patients(

nrsis char(13)

constraint PKNrsis primary key,

nom varchar2(20)

constraint NNPatientsNom not NULL,

prenom varchar2(30)

constraint NNPatientsPrenom not NULL,

sexe varchar2(1)

constraint CKsexe check(sexe is NULL or sexe in('M', 'F')),

datenais date,

hospitalise varchar2(1) default 'N'

constraint CKHospitalise check(Hospitalise in ('N', 'O')),

constraint CKnrsisFormat check(nrsis like (to\_char(coalesce(datenais, to\_date('01/01/1000', 'DD/MM/YYYY')), 'YYMMDD')||'%'))

);

create table Specialites(

specialite number(4)

constraint PKSpecialite primary key,

libelle varchar2(30)

);

create table Services(

numservice char(4)

constraint PKServices primary key,

nomservice varchar2(20),

numchefservice char(11)

constraint NNServicesNumchefservice not null

);

create table Medecins(

numordre char(11)

constraint PKMedecins primary key,

nom varchar2(20)

constraint NNMedecinsNom not null,

prenom varchar2(20)

constraint NNMedecinPrenom not null,

datediplome date,

specialite number(4)

constraint FKMedecinSpecialiteSpecialites references Specialites (specialite),

numservice char(4)

constraint FKMedecinNumserviceServices references Services (numservice)

constraint NNMedecinsNumservice not null,

constraint CKNumordre check(numordre like to\_char(coalesce(datediplome, to\_date('01/01/1000', 'DD/MM/YYYY')), 'YYYYMM')||'%')

);

alter table Services

add constraint FKServicesNumchefserviceMedecins foreign key(numchefservice) references Medecins(numordre);

create table Chambres(

numservice char(4)

constraint FKChambresNumserviceServices references Services (numservice),

numchambre number(4),

constraint PKChambres primary key(numservice, numchambre)

);

create table PatientsChambres(

nrsis char(11)

constraint FKPatientschambresNrsisPatients references Patients (nrsis),

dateentree date,

numservice char(4),

numchambre number(4),

datesortie date,

constraint CKDatesortie check(datesortie is null or datesortie > dateentree),

constraint PKPatientschambres primary key (nrsis, dateentree),

constraint FKPatientschambresNumserviceNumchambre foreign key(numservice, numchambre) references Chambres (numservice, numchambre)

);

LDD2 :

INSERT INTO CHEVAUX

VALUES

('2020E', 'PETIT TONNERE', 'E', 2020, 150);

INSERT INTO CHEVAUX

VALUES

('2014H', 'PETIT MONSTRE', 'H', 2014, 180);

INSERT INTO CHEVAUX

VALUES

('2000J', 'GRAND TONNERE', 'J', 2000, 400);

INSERT INTO CODEPOSTAUX

VALUES

(4000, 'LIEGE');

INSERT INTO CODEPOSTAUX

VALUES

(4100, 'SERAING');

INSERT INTO CODEPOSTAUX

VALUES

(1000, 'BRUXELLES');

SET CONSTRAINT FK\_CLUBS\_PRESIDENT DEFERRED;

INSERT INTO CLUBS

VALUES

(22, 'club1', 'rue du club 1', 4000, 1, '04626532121', NULL, 10);

INSERT INTO CLUBS

VALUES

(1, 'club11', 'rue du club 45', 1000, 5, '04621864686', NULL, 6);

INSERT INTO CLUBS

VALUES

(60, 'club60', 'rue du club 47', 4000, 3, '04798632315', 'club60@gmail.com', 3);

INSERT INTO MEMBRES

VALUES

(1, 'Porte', 'Salva', TO\_DATE('1990-14-12', 'yyyy-dd-mm'), 'M', 22, 'rue de salauds 44', 4100, '0456123886', 'Porte.salva@salaud.com');

INSERT INTO MEMBRES

VALUES

(5, 'dd', 'Salva', TO\_DATE('1999-1-10', 'yyyy-dd-mm'), 'M', 1, 'rue de valves 1', 1000, '046168632', 'dd.salva@hotmail.com');

INSERT INTO MEMBRES

VALUES

(3, 'coucou', 'Salva', TO\_DATE('1998-20-05', 'yyyy-dd-mm'), 'M', 1, 'rue de k 4', 4100, '045321568', 'coucou.salva@gmail.com');

SET CONSTRAINT FK\_CLUBS\_PRESIDENT IMMEDIATE;

INSERT INTO COMPETITIONS

VALUES

(123, '4545s', 'Porte');

INSERT INTO COMPETITIONS

VALUES

(456, 'ijsdjf ', 'Robeeerrt');

INSERT INTO COMPETITIONS

VALUES

(789, 'bulle bulle', 'Mohahahaha');

INSERT INTO COMPETITIONANNEE

VALUES

(123, 1999, TO\_DATE('03-12-2016', 'dd-mm-yyyy'), 5, 45.21, TO\_DATE('03-09-2016', 'dd-mm-yyyy'));

INSERT INTO COMPETITIONANNEE

VALUES

(456, 2018, '03-8-2020', 8, 45.21, '03-12-2016');

INSERT INTO COMPETITIONANNEE

VALUES

(123, 2005, '05-2-2010', 7, 45.21, '03-09-2009');

INSERT INTO JOURNEES

VALUES

(123, 1999, 4, TO\_DATE('5:02:03', 'HH:MI:SS'), 500);

INSERT INTO JOURNEES

VALUES

(123, 2005, 57, TO\_DATE('06:00:00', 'HH:MI:SS'), 20);

INSERT INTO JOURNEES

VALUES

(456, 2018, 90, '15:08:12', 1);

INSERT INTO EPREUVES

VALUES

(123, 1999, 4, 8, 'merde', 'confiance', 'plume', '09-06-2000');

INSERT INTO EPREUVES

VALUES

(456, 2018, 90, 18, 'ah', 'trippooooo', 'yes', NULL);

INSERT INTO EPREUVES

VALUES

(123, 2005, 57, 2, 'chut', 'quoi', 'non', NULL);

INSERT INTO RESULTATS

VALUES

(1, '2020E', 123, 1999, 4, 8, '09-09-2020', 88, 5);

INSERT INTO RESULTATS

VALUES

(5, '2020E', 123, 1999, 4, 8, '01-09-2010', 05, 50);

INSERT INTO RESULTATS

VALUES

(3, '2014H', 456, 2018, 90, 18, '20-02-2015', 12, 41);

LDD3 :

SET CONSTRAINT FK\_CLUBS\_PRESIDENT DEFERRED;

SET CONSTRAINT FK\_RESULTATS\_IDMEMBRE DEFERRED;

UPDATE MEMBRES SET IDMEMBRE = 7 WHERE IDMEMBRE = 3;

UPDATE CLUBS SET PRESIDENT = 7 WHERE PRESIDENT = 3;

SET CONSTRAINT FK\_CLUBS\_PRESIDENT IMMEDIATE;

SET CONSTRAINT FK\_RESULTATS\_IDMEMBRE IMMEDIATE;

LDD4 :

SET CONSTRAINT FK\_CLUBS\_PRESIDENT DEFERRED;

SET CONSTRAINT FK\_RESULTATS\_IDMEMBRE DEFERRED;

DELETE FROM MEMBRES WHERE IDMEMBRE IN(SELECT IDMEMBRE FROM RESULTATS);

DELETE FROM CLUBS WHERE PRESIDENT NOT IN(SELECT IDMEMBRE FROM MEMBRES);

DELETE FROM RESULTATS;

SET CONSTRAINT FK\_CLUBS\_PRESIDENT IMMEDIATE;

SET CONSTRAINT FK\_RESULTATS\_IDMEMBRE IMMEDIATE;

LDD5 :

SET CONSTRAINT FK\_CLUBS\_PRESIDENT DEFERRED;

SET CONSTRAINT FK\_RESULTATS\_IDMEMBRE DEFERRED;

DELETE FROM MEMBRES WHERE IDMEMBRE IN(SELECT PRESIDENT FROM CLUBS FIRST FETCH ROWS ONLY);

DELETE FROM CLUBS WHERE PRESIDENT NOT IN(SELECT IDMEMBRE FROM MEMBRES);

DELETE FROM RESULTATS WHERE IDMEMBRE NOT IN(SELECT IDMEMBRE FROM MEMBRES);

SET CONSTRAINT FK\_CLUBS\_PRESIDENT IMMEDIATE;

SET CONSTRAINT FK\_RESULTATS\_IDMEMBRE IMMEDIATE;

LDD6 :

DROP TABLE RESULTATS;

DROP TABLE CHEVAUX;

DROP TABLE EPREUVES;

DROP TABLE JOURNEES;

DROP TABLE COMPETITIONANNEE;

DROP TABLE COMPETITIONS;

ALTER TABLE CLUBS

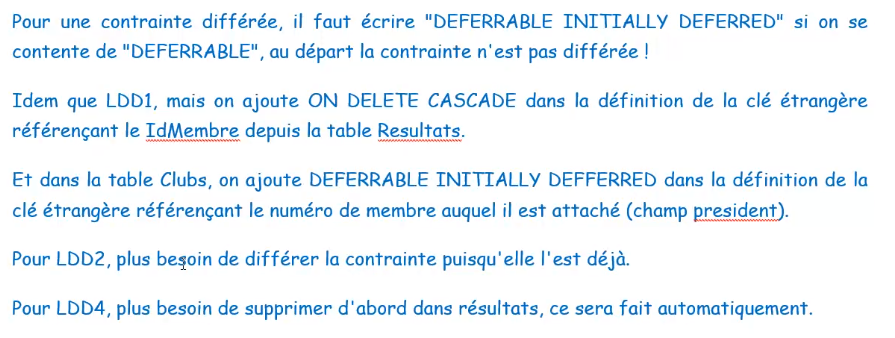
DROP CONSTRAINT FK\_CLUBS\_PRESIDENT;

DROP TABLE MEMBRES;

DROP TABLE CLUBS;

DROP TABLE CODEPOSTAUX;

LDD7 :



LDD8 :

DROP TABLE RESULTATS CASCADE CONSTRAINTS;

DROP TABLE MEMBRES CASCADE CONSTRAINTS;

DROP TABLE CLUBS CASCADE CONSTRAINTS;

DROP TABLE EPREUVES CASCADE CONSTRAINTS;

DROP TABLE JOURNEES CASCADE CONSTRAINTS;

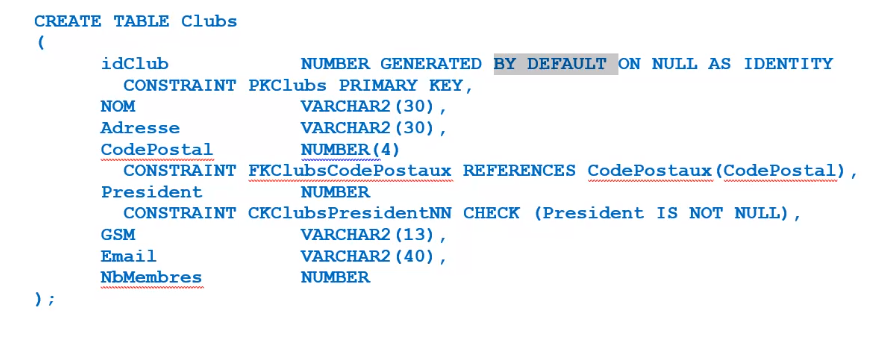
DROP TABLE COMPETITIONS CASCADE CONSTRAINTS;

DROP TABLE CODEPOSTAUX CASCADE CONSTRAINTS;

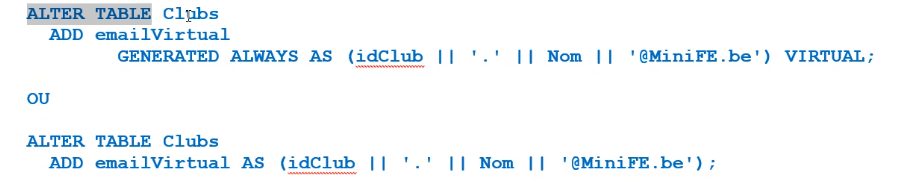
DROP TABLE CHEVAUX CASCADE CONSTRAINTS;

DROP TABLE COMPETITIONANNEE CASCADE CONSTRAINTS;

LDD9 :



LDD10 :



LDD11 :

CREATE INDEX IdxResultatsPlace ON Resultats(Place);

LDD12 :



LMD 1:

select competitions.libelle, count(distinct categorie) "Nb catégories"

from competitions inner join epreuves on(competitions.nrcompetition = epreuves.nrcompetition)

group by competitions.nrcompetition, competitions.libelle;

LMD 2:

select competitions.libelle, cast(avg(nbjours) as decimal(10, 2)) "Nb moyen de jours"

from competitions inner join competitionannee on(competitions.nrcompetition = competitionannee.nrcompetition)

group by competitions.nrcompetition, competitions.libelle

order by "Nb moyen de jours" desc;

LMD 3:

--a)

select nom, prenom

from membres

where sexe = 'M'

and idmembre not in (

select distinct president from clubs

)

order by nom desc, prenom desc;

--b)

select nom, prenom

from membres

where sexe = 'M'

and not exists (

select distinct president from clubs

where idmembre = president

)

order by nom desc, prenom desc;

--c)

select membres.nom, prenom

from membres left join clubs on (membres.idmembre = clubs.president)

where sexe = 'M'

and president is null

order by membres.nom desc, prenom desc;

--d)

select nom, prenom

from membres

where sexe = 'M'

minus

select nom, prenom

from membres

where idmembre in (

select distinct president from clubs

)

order by nom desc, prenom desc;

LMD 4:

select nom, prenom

from clubs inner join membres on(clubs.idclub = membres.idclub)

left join proprietede on(membres.idmembre = proprietede.idmembre)

where idcheval is null

and add\_months(SYSDATE, -12\*30) < 0

LMD 5:

SELECT idcheval, nom, (EXTRACT (YEAR FROM sysdate) - anneenais) Age, coalesce(nomrobe,'robe inconue')

FROM chevaux left outer join robes on (chevaux.idrobe=robes.idrobe)

where idcheval in(SELECT idcheval

from proprietede

where idmembre in (select idmembre from membres where nom = 'Leblond'));

LMD 6:

select clubs.nom, count(idClub) from clubs

inner join membres using(idClub)

group by clubs.nom

order by count(idclub) desc;

LMD 7:

select competitions.libelle, count(nrcompetition) from competitions

inner join competitionannee using(nrcompetition)

group by competitions.libelle

having count(nrcompetition) <= all(select count(c.nrcompetition) from competitions c

inner join competitionannee cc on(cc.nrcompetition = c.nrcompetition)

group by c.libelle);

select competitions.libelle, count(nrcompetition) from competitions

inner join competitionannee using(nrcompetition)

group by competitions.libelle

having count(nrcompetition) = (select min(count(c.nrcompetition)) from competitions c

inner join competitionannee cc on(cc.nrcompetition = c.nrcompetition)

group by c.libelle);

-- lmd 8 --

SELECT clubs.nom, COUNT (\*) AS "Nb Membres"

FROM Membres INNER JOIN Clubs Using (IDClub)

GROUP BY idclub, clubs.nom

ORDER BY "Nb Membres" DESC;

-- lmd 9 --

SELECT nrcompetition,libelle,datedebut, count(\*) as "nb jours"

from competitionannee inner join competitions using (nrcompetition)

inner join journees using (nrcompetition,annee)

where annee = 2019

group by nrcompetition,libelle,datedebut

order by datedebut asc;

-- lmd 10 --

SELECT libelle, count(\*) "nombre d'éditions"

from competitions inner join competitionannee using (nrcompetition)

group by nrcompetition,libelle

having count(\*) <=all (select count(\*)

from competitionannee

group by nrcompetition);

-- deuxieme manière

SELECT libelle, count(\*) "nombre d'éditions"

from competitions inner join competitionannee using (nrcompetition)

group by nrcompetition,libelle

having count(\*)=(select min(count(\*))

from competitionannee

group by nrcompetition);

--lmd 11 --

-- lmd 12--

(sans all)

select discipline, disciplines.libelle, count (\*) nboccurrences

from epreuves inner join disciplines on (disciplines.code = epreuves.discipline)

group by discipline, disciplines.libelle

having count(\*) = (select max(count(\*))

from epreuves

group by discipline)

or count(\*) = (select min(count(\*))

from epreuves

group by discipline);

(avec all)

select discipline, disciplines.libelle, count (\*) nboccurrences

from epreuves inner join disciplines on (disciplines.code = epreuves.discipline)

group by discipline, disciplines.libelle

having count(\*) >= all(select (count(\*))

from epreuves

group by discipline)

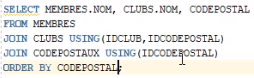
or count(\*) <= all (select (count(\*))

from epreuves

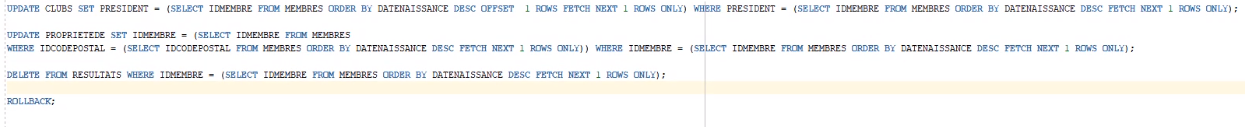
group by discipline);

--lmd 13—

LMD14 :



LMD15 :



Transaction :

Préambule :

drop table Chiens;

create table Chiens(

Id Number

constraint CpChiens primary key,

Nom Varchar2(30),

IdProprio Varchar2(15),

DateNais Date,

Poids Number(6,2),

Taille Number(6,2)

);

insert into chiens values

(11, 'Idefix', '19630401-123.45', to\_date('04/01/1983', 'DD/MM/YYYY'), 18.6, 35.2);

insert into chiens values

(22, 'Milou', '19301201-175.12', to\_date('01/01/1955', 'DD/MM/YYYY'), 20.2, 33.4);

insert into chiens values

(33, 'Rantanplan', '19620515-283.51', to\_date('31/12/1992', 'DD/MM/YYYY'), 23.5, 42.5);

insert into chiens values

(44, 'Pongo', '19610125-321.45', to\_date('02/02/1986', 'DD/MM/YYYY'), 24.3, 51.6);

insert into chiens values

(55, 'Perdita', '19630630-221.65', to\_date('03/03/1990', 'DD/MM/YYYY'), 22.3, 48.3);

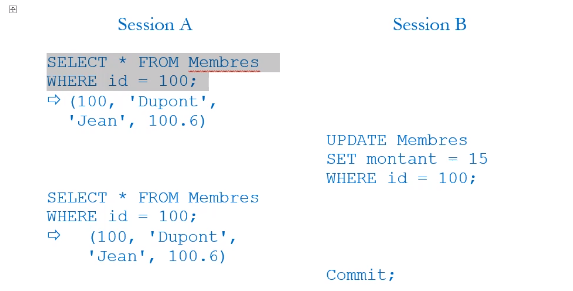
select \* from Chiens;

Transaction 1 :

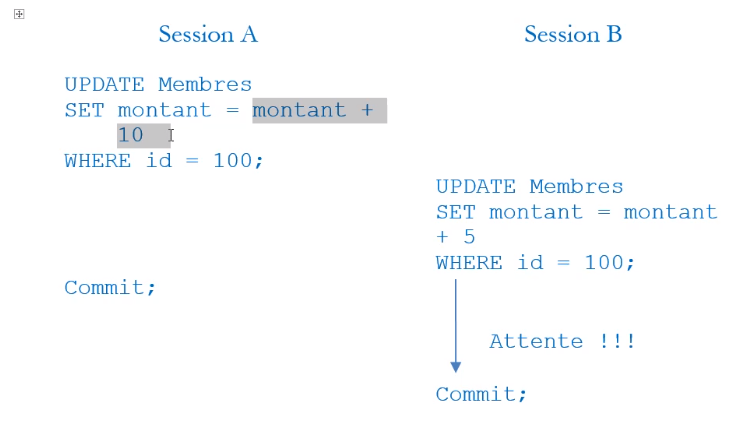
UPDATE MEMBRES SET MONTANT = 1 WHERE ID = 100; a executer en premier

ROLLBACK; a executer en troisieme

SELECT MONTANT FROM MEMBRES WHERE ID = 100; a executer en deuxième



Transaction 2 :



Transaction 3 :

select \* from membres where id=200;

update membres set montant = montant + 50 where id=200;

commit ;

select \* from membres where id=200;

commit;

Transaction 4 :

UPDATE MEMBRES

SET MONTANT = MONTANT+10

WHERE Id : 200 ;

UPDATE MEMBRES

SET MONTANT = MONTANT+10

WHERE Id = 100 ;

UPDATE MEMBRES

SET MONTANT = MONTANT+10

WHERE Id = 100 ;

UPDATE MEMBRES

SET MONTANT = MONTANT+10

WHERE Id : 200 ;

Transaction 5 :

Set transaction read only;

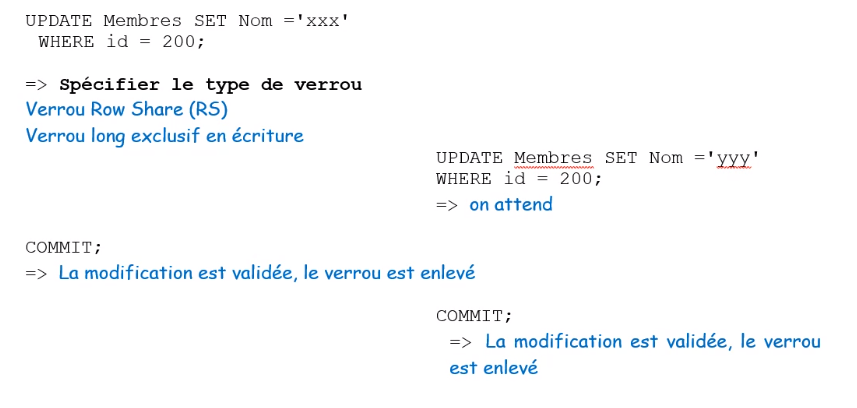
select montant from membres where id=100;

update membres set montant = montant - 100 where id=100;

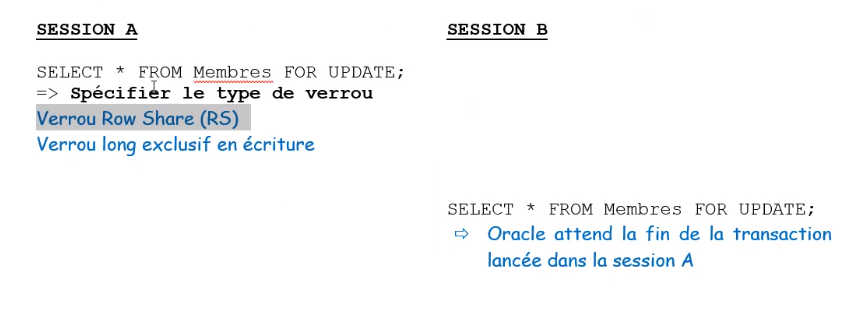
commit;

select montant from membres where id=100;

Transaction 6 :



Transaction 7 :



Transaction 8 :



Vues et Privilèges 1

CREATE USER c##Roberto;

CREATE USER c##Marullo;

GRANT CREATE SESSION To c##Roberto;

GRANT CREATE SESSION TO c##Marullo;

CREATE ROLE Accès\_Minife;

GRANT SELECT ON c##minife.Clubs to Minife;

GRANT SELECT ON c##minife.Chevaux To Accès\_minife;

GRANT SELECT ON c##minife.membres to Accès\_minife;

grant Accès\_MiniFE to c##Marullo;

Vues et Privilèges 2

Create View info\_club (nom, nom\_president, gsm, email, localite, code\_postal)

as select clubs.nom, membres.nom, clubs.gsm, clubs.email, codepostaux.localite, codepostaux.codepostal,

(select count(\*) from membres where membres.idclub = clubs.idclub) as "NbMembres"

from clubs

inner join membres on (membres.idmembre = clubs.President)

inner join codepostaux on (codepostaux.idcodepostal = clubs.idcodepostal)

where codepostal >= 4000 and codepostal =< 4999;

Vues et Privilèges 3

Create View info\_chevaux\_cavaliers (nom, annéenaiss, sexe, taille, nomcavalier)

Select chevaux.nom, chevaux.anneenais, chevaux.sexe, chevaux.taille, membres.nom

from chevaux

inner join membres on (chevaux.cavalier = membres.idmembre)

where taille > 150

Union

Select chevaux.nom, chevaux.anneenais, chevaux.sexe, chevaux.taille, "Inconnu"

from chevaux

inner join membres on (chevaux.cavalier = membres.idmembre)

where taille < 150

grant select on info\_chevaux\_cavaliers to public

Vues et Privilèges 4

select \* from all\_tab\_privs

where grantee = 'C##Marullo'

or grantee = 'C##Roberto'

or grantee = 'C##MiniFE'

or grantor = 'C##Marullo'

or grantor = 'C##Roberto'

or grantor = 'C##MiniFE'