Lamia AID

11608696

Licence 3

Groupe 2

Rapport BDD
(TP 1-7)

2019/2020

TP 1

Création d’une base de données sous ORACLE

A-Lancer SqlPlus

1.ouvrir linux
2.gedit .bashrc
→ retirer le # devant oracle
3.mkdir BD
cd BD
4.sqlplus
5.login : L3INFO_122(chaqun a son log/mdp attribué, mais comme beugs sur ça, possibilité que
d'autres personnes aient le même)
mdp : L3INFO_122

B-Première création de table

1- Créez la table ETUDIANTS

```
SQL>
CREATE TABLE ETUDIANTS
(
  NUMERO        NUMBER(4)       PRIMARY KEY,
  NOM           VARCHAR2(25)    NOT NULL,
  PRENOM        VARCHAR2(25)    NOT NULL,
  SEXE          CHAR(1)         CHECK(SEXE IN('F', 'M')),
  DATENAISSANCE DATE            NOT NULL,
  POIDS         NUMBER,
  ANNEE         NUMBER
);
Table cr??e.
```

2- Vérifiez que la table est correctement créée avec « desc nom_table ;» et « select * from nom_table ; ». Quelle est la différence entre cesdeux commandes

```
SQL> desc ETUDIANTS;

Nom                                       NULL ?   Type
 ----------------------------------------- -------- ----------------------------
 NUMERO                                    NOT NULL NUMBER(4)
 NOM                                       NOT NULL VARCHAR2(25)
 PRENOM                                    NOT NULL VARCHAR2(25)
 SEXE                                               CHAR(1)
 DATENAISSANCE                             NOT NULL DATE
 POIDS                                              NUMBER
 ANNEE                                              NUMBER
```

```
SQL> select * from ETUDIANTS;
```

aucune ligne s�lectionn�e

#Explication : desc donne une description de la table, sa structure, tandis que select renvoi les données
contenues dans la table

3- Quel attribut est défini comme clé primaire ici ?

#Explication : Ici c'est le numéro qui est défini comme clé primaire

4- Quelles contraintes ont été définies ?

#Explication : le sexe doit être 'M' ou 'F', le numéro est la primary key, le nom, le prenom et la date de
naissance doivent être non nuls

5- On peut ajouter des lignes avec la commande « INSERT INTO ETUDIANTS VALUES (valeur1, valeur2, ?) ; »

-Insérer les lignes (individus) suivantes :

```
SQL>
insert into etudiants values(71,'Traifor', 'Benoit', 'M', '10/12/1978', 77, 1);
insert into etudiants values(72, 'Génial', 'Clément', 'M', '10/04/1978', 72, 1);
insert into etudiants values(73, 'Paris', 'Adam', 'M', '28/06/1974', 72, 1);
insert into etudiants values(74, 'Parees', 'Clémence', 'F', '20/09/1977', 72,null);
insert into etudiants values(69, 'Saitout', 'Inès', 'F', '22/11/1969', 69, 2);
insert into etudiants values(55, 'Serafoub', 'Izouaf', 'M', '19/09/2013', 1, 0);
```

```
SQL> SELECT * FROM ETUDIANTS;
```

NUMERO	NOM	PRENOM	S	DATENAIS	POIDS	ANNEE
71	Traifor	Benoit	M	10/12/78	77	1
72	Génial	Clément	M	10/04/78	72	1
73	Paris	Adam	M	28/06/74	72	1
74	Parees	Clémence	F	20/09/77	72	
69	Saitout	Inès	F	22/11/69	69	2
55	Serafoub	Izouaf	M	19/09/13	1	0

6- Essayez d’insérer de nouvelles lignes qui violent les contraintes définies pour cette table. Que se passe-t-il ? Toutes les contraintes sont bien vérifiées ?

```
SQL> insert into etudiants values(55, 'Sumner', '', 'M', '09/07/2001', 1, 0);
```

#
ERREUR la ligne 1 :

ORA-01400: impossible d'ins�rer NULL dans
("L3INFO_122"."ETUDIANTS"."PRENOM")

C-Base de données avec plusieurs tables

1. Créez les tables dans un fichier texte ecole.sql. N’oubliez pas les contraintes.

2. Si les tables existent déjà, il est nécessaire de supprimer les tables et leurs contraintes avant de les créer à nouveau. Pour cela utilisez « DROP TABLE table_name CASCADE CONSTRAINTS; » dans le script.

3. Pour définir des clés primaires avec plusieurs attributs, utilisez « CONSTRAINT constraint_name PRIMARY KEY (attribut_1, attribut_2, ... attribut_n) » lors de la création de la table.

4. N’oubliez pas les clés étrangères : « CONSTRAINT constraint_name FOREIGN KEY (attributs) REFERENCES table_name (attributs)».

5. Lancez le script avec la commande « @ecole », puis vérifiez que les tables ont bien été créées.
(ANNEXE TP1)

SQL> @ecole.sql

Validation effectu�e.

TP2

Modification d’une base de données sous ORACLE

A-Modification de contraintes

1- La table suivante ne respecte pas la norme sur la définition de contraintes :

```
SQL> DROP TABLE ETUDIANTS;

SQL>
CREATE TABLE ETUDIANTS
(
   NUMERO              NUMBER(4)       NOT NULL,
   Nom                 VARCHAR2(25)    NOT NULL,
   PRENOM              VARCHAR2(25)    NOT NULL,
   SEXE                CHAR(1)         CHECK(SEXE IN ('F', 'M')),
   DATENAISSANCE       DATE            NOT NULL,
   POIDS               NUMBER,
   ANNEE               NUMBER,
   CONSTRAINT PK_ETUDIANTS PRIMARY KEY (NUMERO)
);

SQL>  ttitle center 'La liste des contraintes créées, sur la table
ETUDIANTS, est :'
select constraint_name from user_constraints where
table_name='ETUDIANTS';

CONSTRAINT_NAME
------------------------------
SYS_C001020735
SYS_C001020736
SYS_C001020737
SYS_C001020738
SYS_C001020739
PK_ETUDIANTS
```

#Explication : Sans attribuer un nom explicite à une contrainte, on ne sait pas a quoi elle correspond quand on
les affiiches toutes.

2- Redéfinir la table en nommant les contraintes.

```
SQL> DROP TABLE ETUDIANTS CASCADE constraints;
```

```
SQL>
CREATE TABLE ETUDIANTS
(
   NUMERO    NUMBER(4)    CONSTRAINT  TMP_NN_ETUDIANTS_NUMERO
NOT NULL,
   NOM       VARCHAR2(25)  CONSTRAINT TMP_NN_ETUDIANTS_NOM NOT
NULL,
   PRENOM    VARCHAR2(25)  CONSTRAINT TMP_NN_ETUDIANTS_PRENOM
NOT NULL,
   SEXE      CHAR(1)       CONSTRAINT TMP_CK_ETUDIANTS_SEXE
CHECK(SEXE IN('F', 'M')),
   DATENAISSANCE DATE      CONSTRAINT
TMP_NN_ETUDIANTS_DATENAISSANCE NOT NULL,
   POIDS     NUMBER,
   ANNEE     NUMBER,
   CONSTRAINT TMP_PK_ETUDIANTS PRIMARY KEY (NUMERO)
);

SQL>
ttitle ?La liste des contraintes créées, sur la table ETUDIANTS,
est :'
select constraint_name from user_constraints where
table_name= 'ETUDIANTS';

?Lalistedescontraintescréées,surlatableETUDIANTS,
CONSTRAINT_NAME
------------------------------
TMP_NN_ETUDIANTS_NUMERO
TMP_NN_ETUDIANTS_NOM
TMP_NN_ETUDIANTS_PRENOM
TMP_NN_ETUDIANTS_DATENAISSANCE
TMP_CK_ETUDIANTS_SEXE
TMP_PK_ETUDIANTS
```

3- Ajoutez deux contraintes :

-L'année doit être égale à 1 ou 2
-Le poids doit être supérieur à 30kg et inferieur à 200kg

```
SQL>
ALTER TABLE ETUDIANTS ADD CONSTRAINT CK_ETUDIANTS_ANNEE
Check(ANNEE IN (1, 2));
ALTER TABLE ETUDIANTS ADD CONSTRAINT CK_ETUDIANTS_POIDS
Check(POIDS > 30 AND POIDS < 200);
```

SQL> select constraint_name from user_constraints where table_name='ETUDIANTS';

?Lalistedescontraintescréées,surlatableETUDIANTS,
CONSTRAINT_NAME

TMP_NN_ETUDIANTS_NUMERO
TMP_NN_ETUDIANTS_NOM
TMP_NN_ETUDIANTS_PRENOM
TMP_NN_ETUDIANTS_DATENAISSANCE
TMP_CK_ETUDIANTS_SEXE
TMP_PK_ETUDIANTS
CK_ETUDIANTS_ANNEE
CK_ETUDIANTS_POIDS

4- Renommez les contraintes selon la norme suivante :

SQL>
ALTER TABLE ETUDIANTS RENAME CONSTRAINT TMP_PK_ETUDIANTS to PK_ETUDIANTS;
ALTER TABLE ETUDIANTS RENAME CONSTRAINT TMP_NN_ETUDIANTS_NUMERO to NN_ETUDIANTS_NUMERO;
ALTER TABLE ETUDIANTS RENAME CONSTRAINT TMP_NN_ETUDIANTS_NOM to NN_ETUDIANTS_NOM;
ALTER TABLE ETUDIANTS RENAME CONSTRAINT TMP_NN_ETUDIANTS_PRENOM to NN_ETUDIANTS_PRENOM;
ALTER TABLE ETUDIANTS RENAME CONSTRAINT TMP_CK_ETUDIANTS_SEXE to CK_ETUDIANTS_SEXE;
ALTER TABLE ETUDIANTS RENAME CONSTRAINT TMP_NN_ETUDIANTS_DATENAISSANCE to NN_ETUDIANTS_DATENAISSANCE;

SQL> select constraint_name from user_constraints where table_name='ETUDIANTS';

?Lalistedescontraintescréées,surlatableETUDIANTS,
CONSTRAINT_NAME

NN_ETUDIANTS_NUMERO
NN_ETUDIANTS_NOM
NN_ETUDIANTS_PRENOM
NN_ETUDIANTS_DATENAISSANCE
CK_ETUDIANTS_SEXE
PK_ETUDIANTS
CK_ETUDIANTS_ANNEE
CK_ETUDIANTS_POIDS

B-Manipulations de la BD École

1- Téléchargez et lancez le script ecole.sql :

2- Ajoutez les clés étrangères.

```
SQL>
ALTER TABLE RESULTATS ADD CONSTRAINT
FK_RESULTATS_NUM_ELEVE_ELEVES FOREIGN KEY(NUM_ELEVE) REFERENCES
ELEVES;
ALTER TABLE RESULTATS ADD CONSTRAINT
FK_RESULTATS_NUM_COURS_COURS FOREIGN KEY(NUM_COURS) REFERENCES
COURS;
ALTER TABLE CHARGE ADD CONSTRAINT
FK_CHARGE_NUM_PROF_PROFESSEURS FOREIGN KEY(NUM_PROF)
REFERENCES PROFESSEURS;
ALTER TABLE CHARGE ADD CONSTRAINT FK_CHARGE_NUM_COURS FOREIGN
KEY(NUM_COURS) REFERENCES COURS;
ALTER TABLE ACTIVITES_PRATIQUEES ADD CONSTRAINT
FK_ACTPRATIQUES_NUM_ELEVES FOREIGN KEY(NUM_ELEVE) REFERENCES
ELEVES;
ALTER TABLE ACTIVITES_PRATIQUEES ADD CONSTRAINT
FK_ACTPRATIQUES_NIV_NOM_ACT FOREIGN KEY(NIVEAU, NOM) REFERENCES
ACTIVITES;
```

3- Affichez la structure de la table ELEVES et listez son contenu.

```
SQL> desc ELEVES;
```

Nom	NULL ?	Type
NUM_ELEVE	NOT NULL	NUMBER(4)
NOM		VARCHAR2(25)
PRENOM		VARCHAR2(25)
DATE_NAISSANCE		DATE
POIDS		NUMBER
ANNEE		NUMBER
SEXE		CHAR(1)

```
SQL>  select*from ELEVES;
```

NUM_ELEVE	NOM	PRENOM	DATE_NAISS	POIDS	ANNEE	S
1	Brisefer	Benoit	10-12-1978	35	1	M
2	Génial	Olivier	10-04-1978	42	1	M
3	Jourdan	Gil	28-06-1974	72	2	F
4	Spring	Jerry	16-02-1974	78	2	M

5	Tsuno	Yoko	29-10-1977	45	1	F
6	Lebut	Marc	29-04-1974	75	2	M
7	Lagaffe	Gaston	08-04-1975	61	1	M
8	Dubois	Robin	20-04-1976	60	2	M
9	Walthéry	Natacha	07-09-1977	59	1	F
10	Danny	Buck	15-02-1973	82	2	M

4- Modifiez la structure de la table ELEVES en a joutant les attributs suivants :

-CodePostal de type numérique avec 5 chiffres
-Ville de type caractère avec 20 caractères max

```
SQL>
ALTER TABLE ELEVES ADD CODEPOSTAL NUMBER(5);
ALTER TABLE ELEVES ADD VILLE VARCHAR2(20);
```

```
SQL> desc ELEVES;
```

Nom	NULL ?	Type
NUM_ELEVE	NOT NULL	NUMBER(4)
NOM		VARCHAR2(25)
PRENOM		VARCHAR2(25)
DATE_NAISSANCE		DATE
POIDS		NUMBER
ANNEE		NUMBER
SEXE		CHAR(1)
CODEPOSTAL		NUMBER(5)
VILLE		VARCHAR2(20)

5- Mettez à jour les adresses des ELEVES de N° 1, 2, 5 et 7 (respectivement) avec les données suivantes :

75013 ; paris
93800 ; EPINAY / seine
93430 ; EPINAY SUR SEINE
91000 ; EPINAY / ORGE

```
SQL>
 UPDATE ELEVES SET CodePostal = 75013, Ville = 'paris' WHERE NUM_ELEVE = 1;
 UPDATE ELEVES SET CodePostal = 93800, Ville = 'EPINAY / seine' WHERE NUM_ELEVE = 2;5
 UPDATE ELEVES SET CodePostal = 93430, Ville = 'EPINAY SUR SEINE' WHERE NUM_ELEVE = 5;
 UPDATE ELEVES SET CodePostal = 91000, Ville = 'EPINAY / ORGE' WHERE NUM_ELEVE = 7;
```

6- Créez une nouvelle table AGGLOMERATION dont le schéma est le suivant :

AGGLOMERATION(CP, VILLE)

SQL>
CREATE TABLE AGGLOMERATION(CP NUMBER(5), VILLE VARCHAR2(20),
CONSTRAINT PK_AGGLOMERATION PRIMARY KEY(CP),
CONSTRAINT UNIQUE_AGGLOMERATION_VILLE UNIQUE(VILLE));

Table créée.

7- Ajoutez des contraintes pour que cette table s'intègre dans le schéma relationne

Et que le nom de la ville soit toujours en majuscule (fonction UPPER)

SQL>
ALTER TABLE AGGLOMERATION ADD CONSTRAINT CK_AGGLO_VILLE
CHECK (VILLE = UPPER(VILLE));
ALTER TABLE ELEVES ADD CONSTRAINT FK_ELEVES_CPVILLE_AGGLO
FOREIGN KEY(CODEPOSTAL, VILLE) REFERENCES AGGLOMERATION(CP,
VILLE);

Table modifiée.

8- Parmi les données suivantes, lesquelles vont donner une erreur (le tester puis corriger pour remplir la table) :

CodePostal Ville
75001 PARIS
75013 PARIS
93800 EPINAY SUR SEINE
93430 Villetaneuse
91000 EPINAY SUR ORGE
93800 EPINAY / SEINE

#Explication : aucune erreur par rapport a ce que j'ai ajouté dans question 7.

9- Mettez à jour la table des élèves pour corriger le nom des villes selon le code postal. Utilisez une seule requête.

SQL>
update eleves set ville = (select ville from agglomeration where (codepostal = cp));

10 lignes mises à jour.

TP3

Fonctions ORACLE

A-Exploration de quelques fonctions ORACLE

1- Testez les commandes ci-dessous et expliquez le résultat obtenu :

1-
SQL> SELECT RPAD('Soleil',17,'bla') "RPAD exemple" FROM DUAL;

```
RPAD exemple
-----------------
Soleilblablablabl
```

#Explication : ça renvoit soleil et complète avec bla bla bla jusqu a avoir 17 char

2-
SQL> SELECT LPAD('Master 2 EID',15,'*.') "LPAD exemple" FROM DUAL;

```
LPAD exemple
---------------
*.*Master 2 EID
```

#Explication : ça renvoit master 2 eid a droite puis complete avec *.

3-
SQL> SELECT SUBSTR('DESS EID',6,3) "SUBSTR exemple" FROM DUAL;

```
SUB
---
EID
```

#Explication : ça renvoit 3 char a partir du 6e char

4-
SQL> SELECT SUBSTR('ABCDEFGHIJ',-5,4) "SUBSTR exemple" FROM DUAL;

```
SUBS
----
FGHI
```

#Explication : ça renvoit 4 char à partir du 5e en partant de la fin

5-
SQL> SELECT TO_CHAR (SYSDATE, 'MM-DD-YYYY HH24:MI:SS') "Now" FROM DUAL;

```
Now
-------------------
11-27-2019 22:48:05
```

#Explication : ça renvoit le mois, le jour, l annee l heure les minutes et les secondes actuelles

6-

```
SQL> SELECT LENGTH('WEB WAREHOUSE') "Longueur en caractères" FROM DUAL;

Longueur en caractères
----------------------
                    13
```

#Explication : ça renvoit la longueur de la chaine "WEB WAREHOUSE"

7-

```
SQL> SELECT ROUND(17.0958,1) "ROUND exemple" FROM DUAL;

ROUND exemple
-------------
         17,1
```

#Explication : ça renvoit le nombre arrondi à 1 chiffre après la virgule

8-

```
SQL> SELECT ROUND(17.58,2) "ROUND exemple" FROM DUAL;

ROUND exemple
-------------
        17,58
```

#Explication : ça renvoit le nombre arrondi à 2 chiffre après la virgule

9-

```
SQL> SELECT TRUNC(1958.0917,1) "TRUNC exemple" FROM DUAL;

TRUNC exemple
-------------
         1958
```

#Explication : ça renvoit le nombre sans la partie située après 1 chiffre après la virgule (non compris) et enlève les 0 inutils

10-

```
SQL> SELECT TRUNC(1958.0917,2) "TRUNC exemple" FROM DUAL;

TRUNC exemple
-------------
     1958,09
```

#Explication : ça renvoit le nombre sans la partie située après 2 chiffre après la virgule (non compris) et enlève les 0 inutils

11-
SQL> SELECT ROUND(TO_DATE('17/09/2009'), 'YEAR') "New Year" FROM DUAL;

```
New Year
--------
01/01/10
```

#Explication : ça renvoit l annne d après reinitialisé

12-
SQL> SELECT SYSDATE FROM DUAL;

```
SYSDATE
--------
11/10/19
```

#Explication : ça renvoit la date du jour (jour courant dans l horloge systeme)

13-
SQL> SELECT EXTRACT(YEAR FROM SYSDATE) FROM DUAL;

```
EXTRACT(YEARFROMSYSDATE)
------------------------
                    2019
```

#Explication : ça renvoit l année actuelle (courante dans l horloge systeme)

14-
SQL> SELECT ADD_MONTHS(SYSDATE,7) FROM DUAL;

```
ADD_MONT
--------
27/06/20
```

#Explication : ça renvoit la date précise qui se trouve 7 mois après la date actuelle (les années sont incrémentés naturellement, la date actuelle correspond à celle enregistrer dans la base)

15-
SQL>SELECT TRUNC(MONTHS_BETWEEN(SYSDATE,
TO_DATE('19/06/2001'))) AS
AGEBB FROM DUAL;

```
    AGEBB
----------
       221
```

#Explication : ça renvoit le nombre de mois compris entre le 19/06/2001 et maintenant (plus précisément maintenant correspondant à la date courante de la base de données)

16-

```
SQL> SELECT TO_NUMBER(TO_CHAR(SYSDATE, 'YYYY')) FROM DUAL;

TO_NUMBER(TO_CHAR(SYSDATE,'YYYY'))
----------------------------------
                              2019
```

#Explication : ça renvoit l annee actuelle

2- Changez le format du type date avec :

```
SQL> ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MM-YYYY';
```

B-Exemple sur une vraie table

1- Créez la table ETUDIANTS (ou vérifiez qu elle existe) :

```
SQL> DROP TABLE ETUDIANTS;

SQL> CREATE TABLE ETUDIANTS
(   NUMERO          NUMBER(4) NOT NULL,
    NOM             VARCHAR2(25) NOT NULL,
    PRENOM          VARCHAR2(25) NOT NULL,
    SEXE            CHAR(1) CHECK (SEXE IN ('F', 'M')),
    DATENAISSANCE DATE NOT NULL,
    POIDS           NUMBER,
    ANNEE           NUMBER,

CONSTRAINT PK_ETUDIANTS PRIMARY KEY (NUMERO) ) ;

SQL> desc ETUDIANTS

 Nom                                       NULL ?   Type
 ----------------------------------------- -------- ----------------------------
 NUMERO                                    NOT NULL NUMBER(4)
 NOM                                       NOT NULL VARCHAR2(25)
 PRENOM                                    NOT NULL VARCHAR2(25)
 SEXE                                               CHAR(1)
 DATENAISSANCE                 NOT NULL DATE
 POIDS                                              NUMBER
 ANNEE                                              NUMBER
```

2- Si besoin, insérez les lignes (tuples) ci-dessous :

SQL> Insert into ETUDIANTS (NUMERO, NOM, PRENOM, SEXE, DATENAISSANCE,POIDS,
ANNEE) VALUES (71, 'Traifor', 'Benoît', 'M', '10/12/1978', 77,1) ;
INSERT into ETUDIANTS (NUMERO, NOM, PRENOM, SEXE, DATENAISSANCE, POIDS,
ANNEE) VALUES(72, 'Génial', 'Clément', 'M', '10/04/1978', 72,1) ;
INSERT into ETUDIANTS (NUMERO, NOM, PRENOM, SEXE, DATENAISSANCE, POIDS,
ANNEE) VALUES(73, 'Paris', 'Adam', 'M', '28/06/1974', 72,2) ;
INSERT INTO ETUDIANTS (NUMERO, NOM, PRENOM, SEXE, DATENAISSANCE, POIDS,
ANNEE) VALUES(74, 'Parees', 'Clémence', 'F', '20/09/1977', 72, NULL);
INSERT INTO ETUDIANTS (NUMERO, NOM, PRENOM, SEXE, DATENAISSANCE, POIDS,
ANNEE) VALUES(69, 'Saitout', 'Inès', 'F', '22/11/1969', 69, 2);
insert INTO ETUDIANTS (NUMERO, NOM, PRENOM, SEXE, DATENAISSANCE, POIDS,
ANNEE) VALUES(55, 'Serafoub', 'Izouaf', 'M', '19/09/2013', 81, 1);

SQL> SELECT * FROM ETUDIANTS;

NUMERO	NOM	PRENOM	S	DATENAIS	POIDS	ANNEE
71	Traifor	Benoît	M	12/10/78	77	1
72	Génial	Clément	M	04/10/78	72	1
73	Paris	Adam	M	28/06/74	72	2
74	Parees	Clémence	F	20/09/77	72	
69	Saitout	Inès	F	22/11/69	69	2
55	Serafoub	Izouaf	M	19/09/13	81	1

3- Testez les requêtes ci-dessous :

1-
SQL> SELECT DECODE(ANNEE, 1, 'Première', 2, 'Seconde', 'Valeur différente de 1 et de 2 !!') AS ANETUDE FROM ETUDIANTS;

```
ANETUDE
---------------------------------
Première
Première
Seconde
Valeur différente de
1 et de 2 !!

Seconde
Première
```

2-
SQL> SELECT UPPER(NOM) FROM ETUDIANTS;

UPPER(NOM)

TRAIFOR
GéNIAL
PARIS
PAREES
SAITOUT
SERAFOUB

3-
SQL> SELECT LOWER(NOM) FROM ETUDIANTS;

LOWER(NOM)
----
traifor
g��nial
paris
parees
saitout
serafoub

4-
SQL> SELECT NVL(ANNEE, 'Valeur NON renseignée') FROM ETUDIANTS;
SELECT NVL(ANNEE, 'Valeur NON renseignée') FROM ETUDIANTS

ERREUR ligne 1 :
ORA-01722: Nombre non valide

5-
SQL> SELECT NVL(ANNEE, 'Valeur NON renseignée') AS AN_ETUDE FROM ETUDIANTS;

ERREUR ligne 1 :
ORA-01722: Nombre non valide

#Explication : La fonction NVL retourne arg2 si arg1 est nulle, sinon la fonction retourne arg1. Les paramètres
utilisés par NVL peuvent être de tout type de données.

#Correction :
-

SQL> SELECT NVL(TO_CHAR(ANNEE), 'Valeur NON renseignée') FROM ETUDIANTS;

NVL(TO_CHAR(ANNEE),VALEURNONRENSEIGNéE

1
1
2
Valeur NON renseignée
2
1

-
SQL> SELECT NVL(TO_CHAR(ANNEE), 'Valeur NON renseignée') AS AN_ETUDE FROM ETUDIANTS;

AN_ETUDE

1
1
2
Valeur NON renseignée
2
1

4- Gestion de l'affichage :

- Affichez les lignes de la table ETUDIANTS. Utilisez les commandes telles que :
>COL attribut FORMAT format
TTITLE 'Un titre'
SET PAGES n
SET LINES m

#Remarque : ça ne marche pas sur ORACLE

5- Interrogation de la BD :

5.1. Affichez le nom et le prénom de tous les étudiants en une seule colonne.

SQL> SELECT NOM || ' ' || PRENOM FROM ETUDIANTS;
'Untitre'
NOM||''||PRENOM

Traifor Benoît
Génial Clément
Paris Adam
Parees Clémence
Saitout Inès
Serafoub Izouaf

5.2. Affichez la première lettre du prénom en majuscule suivie d'un point et d'un

espace (.) suivis du nom en majuscule en une seule colonne, pour les étudiants de sexe masculin.

```
SQL> SELECT UPPER(SUBSTR(PRENOM, 0, 1)) || '. ' || UPPER(NOM) FROM ETUDIANTS WHERE SEXE = 'M';
'Untitre'
UPPER(SUBSTR(PRENOM,0,1))||
----------------------------
B. TRAIFOR
C. GéNIAL
A. PARIS
I. SERAFOUB
```

5.3. Affichez le nom et l'année de naissance des étudiants dont le nom se prononce comme 'Paris'.

```
SQL> SELECT NOM, TO_CHAR(DATENAISSANCE, 'YYYY') FROM ETUDIANTS WHERE SOUNDEX(NOM)=SOUNDEX('Paris');
'Untitre'
NOM                       TO_C
------------------------- ----
Paris                     1974
Parees          1977
```

5.4. Affichez le nom et l'année de naissance des étudiants dont le prénom commence par la lettre I.

```
SQL> SELECT NOM, TO_CHAR(DATENAISSANCE, 'YYYY') FROM ETUDIANTS WHERE
SUBSTR(PRENOM, 1 ,1) = 'I';
'Untitre'
NOM                       TO_C
------------------------- ----
Saitout                   1969
Serafoub                  2013
```

TP4

SQL Simple, Tri et regroupements

A-Table employés

1- Créez et remplissez la table ci-dessous, n'oubliez pas les contraintes

```
SQL> CREATE TABLE EMPLOYES
(
     NumEmp       NUMBER(4),
     NomEmp       VARCHAR2(25),
     DateEmb      DATE,
     DateSortie   DATE,
  CONSTRAINT NN_EMPLOYES_NumEmp NOT NULL,
  CONSTRAINT NN_EMPLOYES_NomEmp NOT NULL,
  CONSTRAINT NN_EMPLOYES_DateEmb NOT NULL,
  CONSTRAINT PK_EMPLOYES PRIMARY KEY(NumEmp)
);
```

```
SQL> desc EMPLOYES;
```

Nom	NULL ?	Type
NUMEMP	NOT NULL	NUMBER(10)
NOMEMP		VARCHAR2(10)
DATEEMB		DATE
DATESORTIE		DATE

```
SQL>
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9007', 'CHEVALIER', '01/01/96');
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9701', 'LEROY', '17/09/97');
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9703', 'LAMI', '17/09/97');
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9801', 'SULTAN', '20/03/98');
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9802', 'CLEMENCE', '16/10/98');
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9803', 'CAVALIER', '22/11/98');
INSERT INTO EMPLOYES(NumEmp, NomEmp, DateEmb)
VALUES('9901', 'ALEXANDRE', '21/02/99');
```

2- Formulez et testez les requêtes suivantes en SQL :

1-

```
SQL> SELECT * FROM EMPLOYES;
```

```
NUMEMP     NOMEMP      DATEEMB  DATESORT
---------- ---------- -------- --------
      9007 CHEVALIER  01/01/96
      9701 LEROY      17/09/97
      9703 LAMI       17/09/97
      9801 SULTAN     20/03/98
      9802 CLEMENCE   16/10/98
      9803 CAVALIER   22/11/98
      9901 ALEXANDRE  21/02/99
```

2-Liste des noms de tous les employés

```
SQL> SELECT NomEmp FROM EMPLOYES;

NOMEMP
----------
CHEVALIER
LEROY
LAMI
SULTAN
CLEMENCE
CAVALIER
ALEXANDRE
```

3-Nom des employés embauchés à partir du 1er janvier 1999

```
SQL> SELECT NomEmp FROM EMPLOYES WHERE DateEmb BETWEEN
'01/01/1999' AND SYSDATE;

NOMEMP
----------
ALEXANDRE
```

4-Liste des employés (Num et Nom) dont le nom commence par la lettre C

```
SQL> SELECT NumEmp, NomEmp FROM EMPLOYES WHERE NomEmp LIKE
'C%';

NUMEMP NOMEMP
---------- ----------
  9007 CHEVALIER
  9802 CLEMENCE
  9803 CAVALIER
```

5-Liste des employés triés par ordre décroissant sur les noms

```
SQL> SELECT * FROM EMPLOYES ORDER BY NomEmp ASC;
```

```
NUMEMP NOMEMP       DATEEMB  DATESORT
---------- ---------- -------- --------
  9901      ALEXANDRE  21/02/99
  9803      CAVALIER    22/11/98
  9007      CHEVALIER   01/01/96
  9802      CLEMENCE    16/10/98
  9703      LAMI         17/09/97
  9701      LEROY        17/09/97
  9801      SULTAN      20/03/98
```

6-Nombre d'employés embauchés chaque année

```
SQL> SELECT SUBSTR(DateEmb, 7,8), COUNT(*) FROM EMPLOYES GROUP
BY SUBSTR(DateEmb, 7,8);

SU   COUNT(*)
-- ----------
99          1
97          2
98          3
96          1
```

7-Nombre d'employés embauchés chaque année ayant un nom de plus de 5 lettres

```
SQL> SELECT SUBSTR(DateEmb, 7,8), COUNT(*) FROM EMPLOYES WHERE
LENGTH(NomEmp)>5 GROUP BY SUBSTR(DateEmb, 7, 8);

SU   COUNT(*)
-- ----------
99          1
98          3
96          1
```

8-Nombre d'employés embauchés chaque année ayant un nom commençant par L ou C, en ne gardant que les années avec au moins deux employés

```
SQL> SELECT SUBSTR(DateEmb, 7, 8), COUNT(*) FROM EMPLOYES WHERE
NomEmp LIKE 'L%' OR NomEmp LIKE 'C%' HAVING COUNT(*) >= 2 GROUP BY
SUBSTR(DateEmb, 7,8);

SU   COUNT(*)
-- ----------
97          2
98          2
```

B-Table postes

1- Créez et remplissez la table ci-dessous, n'oubliez pas les contraintes :

```
SQL> CREATE TABLE postes
(
      NumEmp    NUMBER(4),
      Poste         VARCHAR2(25),
      Salaire       NUMBER(4),
      NumServ   VARCHAR2(2),
      DateDeb    DATE,
      DateFin     Date,

  CONSTRAINT NN_POSTES_NumEmp   NOT NULL,
  CONSTRAINT NN_POSTES_Poste    NOT NULL,
  CONSTRAINT NN_POSTES_Salaire  NOT NULL,
  CONSTRAINT NN_POSTES_NumServ  NOT NULL,
  CONSTRAINT NN_POSTES_DateDeb  NOT NULL,
  CONSTRAINT NN_POSTES_DateFin  NOT NULL,
  CONSTRAINT PK_POSTES PRIMARY KEY(NumEmp, DateDeb),
  CONSTRAINT FK_POSTES FOREIGN KEY (NumEmp) REFERENCES
EMPLOYES(NumEmp)

);

SQL> desc POSTES;

Nom                                       NULL ?           Type
----------------------------------------------------------------------------
NUMEMP                                NOT NULL          NUMBER(4)
POSTE                                                        VARCHAR2(10)
SALAIRE                                                     NUMBER(10)
NUMSERV                                                   VARCHAR2(2)
DATEDEB                               NOT NULL          DATE
DATEFIN                                                      DATE

SQL>
INSERT INTO POSTES(NumEmp, Poste, Salaire, NumServ, DateDeb,
 DateFin) VALUES(9701, 'PRESIDENT', 5800, 'S2', '09/17/97', NULL);
INSERT INTO POSTES(NumEmp, Poste, Salaire, NumServ, DateDeb,
DateFin) VALUES(9703, 'SECRETAIRE', 950, 'S1', '09/17/97',
'12/31/98');
INSERT INTO POSTES(NumEmp, Poste, Salaire, NumServ, DateDeb,
DateFin) VALUES(9703, 'SECRETAIRE', 1200, 'S1', '01/01/99', NULL);
INSERT INTO POSTES(NumEmp, Poste, Salaire, NumServ, DateDeb,
DateFin) VALUES(9801, 'DIRECTEUR', 5300, 'S1', '07/07/97',
'12/31/98');
```

2- Formulez et testez les requêtes suivantes en SQL :

1-Liste de tous les noms de postes

SQL> SELECT * FROM POSTES;

NUMEMP	POSTE	SALAIRE	NU	DATEDEB	DATEFIN
9701	PRESIDENT	5800	S2	17/09/97	
9703	SECRETAIRE	950	S1	17/09/97	13/12/98
9703	SECRETAIRE	1200	S1	01/01/99	
9703	DIRECTEUR	5300	S1	07/01/97	31/12/98
9801	DIRECTEUR	3200	S5	20/03/98	
9802	DIRECTEUR	3500	S2	16/10/98	
9803	INGENIEUR	2600	S4	22/11/98	
9901	DIRECTEUR	3000	S3	21/02/99	

2-Postes occupés dont le salaire de l'employé est supérieur ou égal à 3000

SQL> SELECT * FROM POSTES WHERE SALAIRE >= 3000;

NUMEMP	POSTE	SALAIRE	NU	DATEDEB	DATEFIN
9701	PRESIDENT	5800	S2	17/09/97	
9703	DIRECTEUR	5300	S1	07/01/97	31/12/98
9801	DIRECTEUR	3200	S5	20/03/98	
9802	DIRECTEUR	3500	S2	16/10/98	
9901	DIRECTEUR	3000	S3	21/02/99	

3-Postes occupés, triés par ordre décroissant et salaires par ordre croissant

SQL> SELECT POSTE, SALAIRE FROM POSTES ORDER BY POSTE DESC,SALAIRE ASC;

POSTE	SALAIRE
SECRETAIRE	950
SECRETAIRE	1200
PRESIDENT	5800
INGENIEUR	2600
DIRECTEUR	3000
DIRECTEUR	3200
DIRECTEUR	3500
DIRECTEUR	5300

4-Salaire le plus bas

SQL> SELECT MIN(Salaire) AS Salaire_plusBas FROM POSTES;

SALAIRE_PLUSBAS
950

5-Moyenne des salaires

SQL> SELECT AVG(SALAIRE) AS Moyenne_Salaire FROM POSTES;

```
MOYENNE_SALAIRE
---------------
        3193,75
```

6-Moyenne des salaires pour les postes actuellement occupés

SQL> SELECT AVG(SALAIRE) AS MOYENNE_ACTUELLE FROM POSTES WHERE DateFin is NULL;

```
MOYENNE_ACTUELLE
----------------
    3216,66667
```

7-Nombre de salariés avec un salaire > 3000

SQL> SELECT COUNT(NumEmp) AS NbSalaries FROM POSTES WHERE SALAIRE > 3000;

```
NBSALARIES
----------
         4
```

8-Moyenne des salaires actuels pour chaque service

SQL> SELECT NumServ, AVG(SALAIRE) FROM POSTES GROUP BY NumServ;

```
NU AVG(SALAIRE)
-- ------------
S4         2600
S3         3000
S5         3200
S2         4650
S1    2483,33333
```

9-Moyenne des salaires pour chaque poste avec au moins 2 employés

SQL> SELECT POSTE, AVG(SALAIRE) FROM POSTES GROUP BY POSTE HAVING COUNT(*) >=2;

```
POSTE               AVG(SALAIRE)
---------- ------------
DIRECTEUR       3750
SECRETAIRE      1075
```

C-Table Etudiants

1- Reprenez la table étudiant du TP précédent et formulez les requêtes suivantes :

1- Moyenne des poids par sexe

SQL> SELECT SEXE, AVG(POIDS) FROM ETUDIANTS GROUP BY SEXE;

SEXE	AVG(POIDS)
M	75.5
F	70.5

2-Moyenne des poids par sexe et par tranche d'âge

SQL>
SELECT SEXE, ANNEE, TRUNC(MONTHS_BETWEEN(SYSDATE, TO_DATE(DATENAISSANCE))/12), AVG(POIDS) FROM ETUDIANTS GROUP BY SEXE, ANNEE, TRUNC(MONTHS_BETWEEN(SYSDATE, TO_DATE(DATENAISSANCE))/12);

SEXE	ANNEE	TRUNC(MONTHS_BETWEEN(SYSDATE,TO_DATE(DATENAISSANCE))/12)	AVG(POIDS)
M	2	44	72
F	-	41	72
M	1	40	72
M	1	5	81
M	1	39	77
F	2	48	69

TP5

SQL : Jointures

A-Gestion d'un café

1- Téléchargez (lipn.univ-paris13.fr/~cabanes/INFO1/) et lancez les deux scripts SQL suivant pour créer et remplir les tables :

SQL> creatcafe.sql
SQL> insertcafe.sql

3- Formulez et testez les requêtes suivantes en SQL :

1-Liste du contenu de chaque table de la base.
SQL> SELECT * FROM COMPREND;

NUMFACTURE	NUMCONS	QTE
1200	101	3
1200	106	1
1200	120	1
1201	105	2
1201	106	2
1202	100	2
1202	122	1
1203	102	1
1203	108	1
1203	121	1
1203	130	1
1204	122	4
1204	124	2
1205	100	2
1206	108	3
1207	108	1
1207	110	2
1208	108	2

SQL> SELECT * FROM FACTURE;

NUMFACTURE	NUMTABLE	NUMSERVEUR	DATEFACT
1200	1	53	01/02/10
1201	5	53	01/02/10
1202	3	52	01/02/10
1203	5	50	01/02/10
1204	4	52	02/02/10
1205	1	53	02/02/10
1206	3	52	02/02/10
1207	5	53	02/02/10
1208	7	54	02/02/10

SQL> SELECT * FROM SERVEUR;

NUMSERVEUR	NOMSERVEUR	RUESERVEUR	CPSERVEUR	VILLESERVEUR	DATENSER
50	Pizzi	3 rue des lilas	90000	BELFORT	12/01/76
51	Cathy	25 av Roosevelt	90100	DELLE	05/04/78
52	Totof	46 grande rue	90500	BAVILLIERS	30/09/84
53	Pilou	5 impasse Martin	90000	BELFORT	17/08/86
54	Alice	15 rue de la barre	95880	ENGHIEN	13/03/86

SQL> SELECT * FROM LESTABLES;

NUMTABLE	NOMTABLE	NBPLACE
1	entree-gche	6
2	entree-dte	10
3	fenetre1	3
4	fenetre2	8
5	fenetre3	4
6	fond-gche	4
7	fond-dte	2

SQL> SELECT * FROM CONSOMMATION;

NUMCONS	LIBCONS	PRIXCONS
100	Cafe	,9
101	Cafe double	1,3
102	Cafe creme	1
105	Chocolat	1,5
106	Biere pression	1,8
107	Biere 25cl	2
108	Biere 33cl	2,2
110	Biere 50cl	2,5
120	Jus de fruits	1,7
121	Jus de fruits presse	2,6
122	Perrier	1,6
124	Orangina	1,4
130	Coca Cola	1,7

2-Nombre de places de la table N°4 (Nbplace)

SQL> SELECT NBPLACE FROM LESTABLES WHERE NUMTABLE = 4;

NBPLACE
8

3-Liste des consommations dont le prix unitaire est supérieur à 1 euro (Numcons, Libcons, Prixcons).

SQL> SELECT * FROM CONSOMMATION WHERE PRIXCONS > 1;

NUMCONS	LIBCONS	PRIXCONS
101	Cafe double	1,3
105	Chocolat	1,5
106	Biere pression	1,8
107	Biere 25cl	2
108	Biere 33cl	2,2
110	Biere 50cl	2,5
120	Jus de fruits	1,7
121	Jus de fruits presse	2,6
122	Perrier	1,6
124	Orangina	1,4
130	Coca Cola	1,7

4-Liste des serveurs de Belfort et de Delle (Numserv, Nomserv, Villeserveur).

SQL> SELECT NUMSERVEUR, NOMSERVEUR, VILLESERVEUR FROM SERVEUR WHERE VILLESERVEUR IN ('BELFORT','DELLE');

NUMSERVEUR	NOMSERVEUR	VILLESERVEUR
50	Pizzi	BELFORT
51	Cathy	DELLE
53	Pilou	BELFORT

5-Liste des factures du 2 février servies par le serveur 52 (Numfact,Numtable).

SQL> SELECT NUMFACTURE, NUMTABLE FROM FACTURE WHERE TO_CHAR(DATEFACTURE, 'MM-DD') = '02-02' AND NUMSERVEUR = 52;

NUMFACTURE	NUMTABLE
1204	4
1206	3

6-Liste des consommations de la facture 1203 (Numcons, Qte).

SQL> SELECT NUMFACTURE, NUMCONS FROM COMPREND WHERE NUMFACTURE = 1203;

NUMFACTURE	NUMCONS
1203	102
1203	108
1203	121
1203	130

7-Liste des consommations des factures 1200 et 1201 (sans lignes en double)(Numcons).

SQL> SELECT DISTINCT NUMCONS FROM COMPREND WHERE NUMFACTURE IN ('1200', '1201');

```
NUMCONS
----------
   120
   101
   105
   106
```

8-Liste des serveurs qui sont nés en 1976 (Nomserv, Datenserveur)

```
SQL> SELECT NOMSERVEUR, DATENSERVEUR FROM SERVEUR WHERE
TO_CHAR(DATENSERVEUR, 'YY') = 76;

NOMSERVEUR          DATENSER
------------------- --------
Pizzi               12/01/76
```

9-Liste des consommations de type bière (Numcons, Libcons, Prixcons).

```
SQL> SELECT * FROM CONSOMMATION WHERE LOwER(LIBCONS) LIKE '%biere%';

  NUMCONS LIBCONS                        PRIXCONS
---------- ------------------------------ ----------
       106 Biere pression                        1,8
       107 Biere 25cl                            2
       108 Biere 33cl                          2,2
       110 Biere 50cl                          2,5
```

10-Liste des tables servies après le 1 février.

```
SQL> SELECT * FROM FACTURE WHERE TO_CHAR(DATEFACTURE, 'MM-DD') > '02-
01';

NUMFACTURE   NUMTABLE NUMSERVEUR DATEFACT
---------- ---------- ---------- --------
      1204          4         52 02/02/10
      1205          1         53 02/02/10
      1206          3         52 02/02/10
      1207          5         53 02/02/10
      1208          7         54 02/02/10
```

11-Liste des serveurs dont le nom contient i en deuxième position (Nomserv)

```
SQL> SELECT NOMSERVEUR FROM SERVEUR WHERE NOMSERVEUR LIKE '_i%';

NOMSERVEUR
--------------------
Pizzi
Pilou
```

12-Liste des serveurs dont le nom commence par un P (Nomserv)

```
SQL> SELECT NOMSERVEUR FROM SERVEUR WHERE NOMSERVEUR LIKE 'P%';
NOMSERVEUR
--------------------
Pizzi
```

Pilou

Q13-Liste des serveurs par ville (Nomserv, Villeserveur).

SQL> SELECT NOMSERVEUR, VILLESERVEUR FROM SERVEUR ORDER BY VILLESERVEUR;

NOMSERVEUR	VILLESERVEUR
Totof	BAVILLIERS
Pilou	BELFORT
Pizzi	BELFORT
Cathy	DELLE
Alice	ENGHIEN

14-Liste des consommations classées par ordre alphabétique sur le libellé(Libcons, Numcons, Prixcons).

SQL> SELECT LIBCONS, NUMCONS, PRIXCONS FROM CONSOMMATION ORDER BY LIBCONS ASC;

LIBCONS	NUMCONS	PRIXCONS
Biere pression	106	1,8
Biere 25cl	107	2
Biere 33cl	108	2,2
Biere 50cl	110	2,5
Cafe	100	,9
Cafe creme	102	1
Cafe double	101	1,3
Chocolat	105	1,5
Coca Cola	130	1,7
Jus de fruits	120	1,7
Jus de fruits presse	121	2,6
Orangina	124	1,4
Perrier	122	1,6

15-Liste des villes où habitent des serveurs (sans lignes en double)(Villeserveur).

SQL> SELECT DISTINCT VILLESERVEUR FROM SERVEUR;

VILLESERVEUR
DELLE
BAVILLIERS
ENGHIEN
BELFORT

16-Le nombre de tables du restaurant.

SQL> SELECT COUNT(NUMTABLE) FROM LESTABLES ;

COUNT(NUMTABLE)

7

17-Le nombre de places disponibles sur l'ensemble des tables

```
SQL> SELECT SUM(NBPLACE) FROM LESTABLES;

SUM(NBPLACE)
------------
          37
```

18-Nombre de factures établies par chaque serveur (Numserv, Nbfacture).

```
SQL> SELECT NUMSERVEUR, COUNT(*) "Nbfacture" FROM FACTURE GROUP BY
NUMSERVEUR;

NUMSERVEUR  Nbfacture
---------- ----------
        54          1
        53          4
        52          3
        50          1
```

19-Nombre de factures établies chaque jour (Datefacture, Nbfacture).

```
SQL> SELECT DATEFACTURE, COUNT(DATEFACTURE) "Nbfacture" FROM FACTURE
GROUP BY DATEFACTURE;

DATEFACT  Nbfacture
-------- ----------
01/02/10          4
02/02/10          5
```

20-Liste des serveurs qui ont établi plus de 3 factures (Numserv, Nbfacture).

```
SQL> SELECT NUMSERVEUR, COUNT(*) "Nbfacture" FROM FACTURE GROUP BY
NUMSERVEUR;

NUMSERVEUR  Nbfacture
---------- ----------
        54          1
        53          4
        52          3
        50          1
```

21-Prix moyen des consommations (Prixmoyen).

```
SQL> SELECT ROUND(AVG(PRIXCONS),2) "Prixmoyen" FROM CONSOMMATION;

 Prixmoyen
----------
      1,71
```

22-Prix moyen du café (Prixmoyen).

SQL> SELECT ROUND(AVG(PRIXCONS),2) "Prixmoyen" FROM CONSOMMATION WHERE LOWER(LIBCONS) LIKE '%cafe%';

```
 Prixmoyen
----------
      1,07
```

23-Quantité moyenne consommée pour chaque consommation (Numcons,Qtemoyenne).

SQL> SELECT NUMCONS, (SUM(QTE)/COUNT(NUMFACTURE)) "Qtemoyenne" FROM COMPREND GROUP BY NUMCONS;

```
   NUMCONS Qtemoyenne
---------- ----------
       100          2
       120          1
       121          1
       108       1,75
       102          1
       110          2
       101          3
       130          1
       105          2
       124          2
       106        1,5
       122        2,5
```

24-Nombre de serveurs par ville (Villeserveur, Nbserveur)

SQL> SELECT VILLESERVEUR, COUNT(*) "Nbserveur" FROM SERVEUR GROUP BY VILLESERVEUR;

```
VILLESERVEUR         Nbserveur
-------------------- ----------
DELLE                         1
BAVILLIERS                    1
ENGHIEN                       1
BELFORT                       2
```

25-Liste des villes dans lesquelles habitent plus d'un serveur (Villeserveur,Nbserveur).

SQL> SELECT VILLESERVEUR, COUNT(*) "Nbserveur" FROM SERVEUR GROUP BY VILLESERVEUR HAVING COUNT(*) > 1;

```
VILLESERVEUR         Nbserveur
-------------------- ----------
BELFORT                       2
```

26-Nombre de types de consommations par factures (Numfacture, Nbcons).

SQL> SELECT NUMFACTURE, COUNT(NUMCONS) "Nbcons" FROM COMPREND GROUP BY NUMFACTURE;

NUMFACTURE	Nbcons
1200	3
1201	2
1202	2
1203	4
1204	2
1205	1
1206	1
1207	2
1208	1

27-Nombre total de consommations (en comptant la quantité) par facture(Numfacture, Qtecons).

SQL> SELECT NUMFACTURE, SUM(QTE) "Qtecons" FROM COMPREND GROUP BY NUMFACTURE;

NUMFACTURE	Qtecons
1207	3
1208	2
1200	5
1205	2
1201	4
1204	6
1203	4
1202	3
1206	3

28-Nombre de factures par consommation (Numcons, Nbfactures)

SQL> SELECT NUMCONS, COUNT(NUMFACTURE) "Nbfacture" FROM COMPREND GROUP BY NUMCONS;

NUMCONS	Nbfacture
100	2
120	1
121	1
108	4
102	1
110	1
101	1
130	1
105	1
124	1
106	2
122	2

29-Consommations qui interviennent dans plus de 2 factures (Numcons,Nbfactures)

SQL> SELECT NUMCONS, COUNT(NUMFACTURE) "Nbfacture" FROM COMPREND GROUP BY NUMCONS HAVING COUNT(NUMFACTURE) >= 2;

```
  NUMCONS  Nbfacture
---------- ----------
       100          2
       108          4
       106          2
       122          2
```

30-Liste des serveurs, triés par nom de ville croissant, puis nom de serveur croissant.

SQL> SELECT NOMSERVEUR, VILLESERVEUR FROM SERVEUR ORDER BY VILLESERVEUR ASC, NOMSERVEUR ASC;

```
NOMSERVEUR           VILLESERVEUR
-------------------- --------------------
Totof                BAVILLIERS
Pilou                BELFORT
Pizzi                BELFORT
Cathy                DELLE
Alice                ENGHIEN
```

31-iste des serveurs, triés par nom de ville décroissant, puis nom de serveur croissant.

SQL> SELECT NOMSERVEUR, VILLESERVEUR FROM SERVEUR ORDER BY VILLESERVEUR DESC, NOMSERVEUR ASC;

```
NOMSERVEUR           VILLESERVEUR
-------------------- --------------------
Alice                ENGHIEN
Cathy                DELLE
Pilou                BELFORT
Pizzi                BELFORT
Totof                BAVILLIERS
```

32-Liste des factures avec leur numéro de table et le nom du serveur(Numfacture, Numtable, Nomserveur)

SQL> SELECT NUMFACTURE, NUMTABLE, NOMSERVEUR FROM FACTURE NATURAL JOIN SERVEUR;

```
NUMFACTURE   NUMTABLE NOMSERVEUR
---------- ---------- --------------------
      1200          1 Pilou
      1201          5 Pilou
      1202          3 Totof
      1203          5 Pizzi
      1204          4 Totof
      1205          1 Pilou
      1206          3 Totof
      1207          5 Pilou
      1208          7 Alice
```

33-Liste des factures de la table 5 avec le nom du serveur (Numfacture,Nomserveur).

SQL> SELECT NUMFACTURE, NOMSERVEUR FROM FACTURE NATURAL JOIN SERVEUR WHERE NUMTABLE = 5;

```
NUMFACTURE NOMSERVEUR
---------- --------------------
      1201 Pilou
      1203 Pizzi
      1207 Pilou
```

34-Liste des factures avec leur nom de table et le nom du serveur (Numfacture, Nomtable, Nomserveur).

SQL> SELECT NUMFACTURE, NOMTABLE, NOMSERVEUR
FROM FACTURE NATURAL JOIN LESTABLES NATURAL JOIN SERVEUR;

```
NUMFACTURE NOMTABLE      NOMSERVEUR
---------- --------------- --------------------
      1205 entree-gche     Pilou
      1200 entree-gche     Pilou
      1206 fenetre1        Totof
      1202 fenetre1        Totof
      1204 fenetre2        Totof
      1207 fenetre3        Pilou
      1203 fenetre3        Pizzi
      1201 fenetre3        Pilou
      1208 fond-dte        Alice
```

35-Liste des serveurs et des tables qu'ils ont servies ordonnés selon le nom du serveur (pas de ligne double) (Nomserveur, Nomtable).

SQL> SELECT DISTINCT NOMSERVEUR, NOMTABLE FROM FACTURE NATURAL JOIN SERVEUR NATURAL JOIN LESTABLES ORDER BY NOMSERVEUR;

```
NOMSERVEUR           NOMTABLE
-------------------- ---------------
Alice                fond-dte
Pilou                entree-gche
Pilou                fenetre3
Pizzi                fenetre3
Totof                fenetre1
Totof                fenetre2
```

36-Liste des consommations de la facture 1203 avec leur nom, leur prix et leur quantité (Numcons, Libcons, Prixcons, Qte).

SLQ> SELECT NUMCONS, LIBCONS, PRIXCONS, QTE
FROM COMPREND NATURAL JOIN CONSOMMATION
WHERE NUMFACTURE = 1203;

NUMCONS	LIBCONS	PRIXCONS	QTE
102	Cafe creme	1	1
108	Biere 33cl	2,2	1
121	Jus de fruits presse	2,6	1
130	Coca Cola	1,7	1

37-Liste des consommations du premier février de la table 5 avec leur nom, leur prix et leur quantité (Numcons, Libcons, Prixcons, Qte).

```
SQL> SELECT NUMCONS, LIBCONS, PRIXCONS, QTE
FROM COMPREND NATURAL JOIN FACTURE NATURAL JOIN CONSOMMATION
WHERE TO_CHAR(DATEFACTURE, 'MM-DD') = '02-01';
```

NUMCONS	LIBCONS	PRIXCONS	QTE
100	Cafe	,9	2
101	Cafe double	1,3	3
102	Cafe creme	1	1
105	Chocolat	1,5	2
106	Biere pression	1,8	2
106	Biere pression	1,8	1
108	Biere 33cl	2,2	1
120	Jus de fruits	1,7	1
121	Jus de fruits presse	2,6	1
122	Perrier	1,6	1
130	Coca Cola	1,7	1

38-Liste des tables et des numéros de factures qui leur sont associées. Attention, on veut voir toutes les tables même si elles n'ont pas de factures. La table de départ (celle du FROM) sera LESTABLES (Nomtable, Numfacture).

```
SQL> SELECT NOMTABLE, NUMFACTURE FROM LESTABLES NATURAL LEFT JOIN
FACTURE;
```

NOMTABLE	NUMFACTURE
entree-gche	1200
fenetre3	1201
fenetre1	1202
fenetre3	1203
fenetre2	1204
entree-gche	1205
fenetre1	1206
fenetre3	1207
fond-dte	1208
entree-dte	
fond-gche	

39-Même question que précédemment, mais avec FACTURE comme table de départ

```
SQL> SELECT NOMTABLE, NUMFACTURE FROM FACTURE NATURAL RIGHT JOIN
LESTABLES;
```

```
NOMTABLE     NUMFACTURE
---------- ----------
entree-gche         1200
fenetre3            1201
fenetre1            1202
fenetre3            1203
fenetre2            1204
entree-gche         1205
fenetre1            1206
fenetre3            1207
fond-dte            1208
entree-dte
fond-gche
```

40-Liste des tables qui n'ont eu aucune factures (Numtable,Nomtable).

```
SQL> SELECT DISTINCT NUMTABLE, NOMTABLE FROM LESTABLES WHERE NOT
EXISTS(SELECT NUMTABLE FROM FACTURE WHERE LESTABLES.NUMTABLE =
FACTURE.NUMTABLE);

  NUMTABLE NOMTABLE
---------- ----------
         2 entree-dte
         6 fond-gche
```

41-Liste des consommations qui ont déjà été servies par le serveur 52 (Numcons, Libcons).

```
SQL> SELECT NUMCONS, LIBCONS FROM CONSOMMATION
WHERE EXISTS (SELECT * FROM COMPREND NATURAL JOIN FACTURE
WHERE FACTURE.NUMSERVEUR = 52 AND CONSOMMATION.NUMCONS =
COMPREND.NUMCONS);

   NUMCONS LIBCONS
---------- ------------------------------
       100 Cafe
       108 Biere 33cl
       122 Perrier
       124 Orangina
```

42-Liste des consommations qui n'ont jamais été servies (Numcons, Libcons)

```
SQL> SELECT NUMCONS, LIBCONS FROM CONSOMMATION
WHERE NOT EXISTS (SELECT * FROM COMPREND NATURAL JOIN FACTURE
WHERE CONSOMMATION.NUMCONS = COMPREND.NUMCONS);

   NUMCONS LIBCONS
---------- ------------------------------
       107 Biere 25cl
```

43-La liste des factures avec leur date et leur nombre de consommations (prendre en compte la quantité) (Numfacture, Datefacture, Nbcons).

```
SQL> SELECT NUMFACTURE, DATEFACTURE, SUM(QTE)"Nb CONSOMMATION"
FROM FACTURE JOIN COMPREND USING (NUMFACTURE)
GROUP BY NUMFACTURE, DATEFACTURE;
```

NUMFACTURE	DATEFACT	Nb CONSOMMATION
1202	01/02/10	3
1204	02/02/10	6
1203	01/02/10	4
1205	02/02/10	2
1200	01/02/10	5
1207	02/02/10	3
1201	01/02/10	4
1206	02/02/10	3
1208	02/02/10	2

44-La liste des factures et le montant de leur addition (Numfacture, Prixfacture).

```
SQL> SELECT NUMFACTURE, SUM(QTE*PRIXCONS) "Prixfacture"
FROM COMPREND JOIN CONSOMMATION USING (NUMCONS) GROUP BY
NUMFACTURE;
```

NUMFACTURE	Prixfacture
1207	7,2
1208	4,4
1200	7,4
1205	1,8
1201	6,6
1204	9,2
1203	7,5
1202	3,4
1206	6,6

45-Nombre de consommations servies par jour (Datefacture, Nbcons).

```
SQL> SELECT DATEFACTURE, SUM(QTE) "Nb CONSOMMATION"
FROM FACTURE JOIN COMPREND USING (NUMFACTURE)
GROUP BY DATEFACTURE;
```

DATEFACT	Nb CONSOMMATION
01/02/10	16
02/02/10	16

46-Montant global du chiffre d'affaire par jour (Datefacture, ca).

```
SQL> SELECT DATEFACTURE, SUM(PRIXFACTURE) AS CA
FROM FACTURE JOIN ( SELECT NUMFACTURE, SUM(QTE * PRIXCONS) AS
PRIXFACTURE FROM COMPREND JOIN CONSOMMATION USING (NUMCONS)
GROUP BY NUMFACTURE ORDER BY NUMFACTURE ASC) USING (NUMFACTURE)
GROUP BY DATEFACTURE;
```

```
DATEFACT      CA
-------- ----------
01/02/10      24,9
02/02/10      29,2
```

47-La liste des serveurs par nom et leur nombre de factures. Attention, les serveurs n'ayant fait aucune facture doivent apparaître dans le résultat(Nomserveur, Nbfactures).

```
SQL> SELECT NOMSERVEUR, COUNT(NUMFACTURE) "Nbfactures"
FROM SERVEUR LEFT JOIN FACTURE USING (NUMSERVEUR) GROUP BY
NOMSERVEUR;

NOMSERVEUR           Nbfactures
-------------------- ----------
Totof                         3
Pizzi                         1
Alice                         1
Cathy                         0
Pilou                         4
```

48-La liste des serveurs par nom et le nombre de consommations qu'ils ont servies (NomServeur, Nbcons).

```
SQL> SELECT NOMSERVEUR, SUM(QTE)"Nbcons" FROM SERVEUR JOIN (SELECT *
FROM COMPREND NATURAL JOIN FACTURE) USING (NUMSERVEUR)
GROUP BY NOMSERVEUR

NOMSERVEUR           NBCONS
-------------------- ----------
Totof                        12
Pizzi                         4
Alice                         2
Pilou                        14
```

49-La liste des serveurs par nom et leur chiffre d'affaire (somme des additions encaissées) (Nomserveur, Ca).

```
SQL> SELECT NOMSERVEUR, SUM(PRIXFACTURE) AS CA
FROM (SELECT * FROM FACTURE LEFT JOIN SERVEUR USING (NUMSERVEUR))
JOIN (SELECT NUMFACTURE, SUM(QTE * PRIXCONS) AS PRIXFACTURE FROM
COMPREND JOIN CONSOMMATION USING(NUMCONS) GROUP BY NUMFACTURE
) USING (NUMFACTURE) GROUP BY NOMSERVEUR;

NOMSERVEUR           CA
-------------------- ----------
Totof                      19,2
Alice                      4,4
Pizzi                      7,5
Pilou                      23
```

50-Le nom des tables qui ont eu au moins deux factures (Nomtable,Nbfactures).

```
SQL> SELECT NOMTABLE, COUNT(NUMTABLE) AS Nbfactures
FROM FACTURE JOIN LESTABLES USING (NUMTABLE)
GROUP BY NOMTABLE HAVING  COUNT(NUMTABLE) >=2;
```

NOMTABLE	NBFACTURES
fenetre1	2
fenetre3	3
entree-gche	2

51-La liste complète des consommations et le nombre de factures dans lesquels elles apparaissent (Libcons, Nbfactures).

```
SQL> SELECT LIBCONS, COUNT(NUMFACTURE) AS NBFACTURES
FROM CONSOMMATION LEFT JOIN COMPREND USING (NUMCONS)
GROUP BY LIBCONS;
```

LIBCONS	NBFACTURES
Coca Cola	1
Biere 50cl	1
Biere pression	2
Biere 33cl	4
Jus de fruits presse	1
Cafe double	1
Jus de fruits	1
Cafe	2
Biere 25cl	0
Perrier	2
Chocolat	1
Cafe creme	1
Orangina	1

52-La liste complète des tables et leur chiffre d'affaire (Nomtable, Ca)

```
SQL> SELECT NOMTABLE, CA FROM LESTABLES LEFT JOIN (
SELECT NUMTABLE, SUM(QTE * PRIXCONS) AS CA
FROM (SELECT * FROM FACTURE LEFT JOIN COMPREND USING (NUMFACTURE))
LEFT JOIN(CONSOMMATION) USING (NUMCONS) GROUP BY NUMTABLE;
) USING (NUMTABLE) GROUP BY NOMTABLE;
```

NUMTABLE	CA
1	9.2
5	21.3
4	9.2
3	10
7	4.4

B-Généalogie royale

1- Téléchargez et lancez le script SQL suivant pour créer et remplir la table

SQL> genealogie.sql

3-Formulez et testez les requêtes suivantes en SQL :

1-Liste des enfants d'Elizabeth II (Nom, DateNaissance).

```
SQL> SELECT NOM, DATENAISSANCE FROM GENEALOGIE
WHERE MERE = (SELECT NUMPER FROM GENEALOGIE WHERE LOWER(NOM) =
'elizabeth ii');
```

```
NOM                                DATENAIS
---------------------------------- --------
Prince Charles                     14/11/48
Princesse Anne                     15/08/50
Prince Andrew                      19/02/60
Prince Edward                      10/03/64
```

2-La mère du prince William (Nom, DateNaissance).

```
SQL> SELECT NOM, DateNaissance FROM GENEALOGIE
WHERE numPer =(SELECT MERE FROM GENEALOGIE WHERE LOwER(NOM) =
'prince william');
```

```
NOM                                DATENAIS
---------------------------------- --------
Diana Spencer                      01/07/61
```

Q3-Les parents d'Elizabeth II (Nom, DateNaissance).

```
SQL> SELECT NOM, DateNaissance FROM GENEALOGIE
WHERE numPer =(SELECT MERE FROM GENEALOGIE WHERE LOwER(NOM) =
'elizabeth ii') OR numPer = (SELECT PERE FROM GENEALOGIE WHERE LOwER(NOM)
= 'elizabeth ii');
```

```
NOM                                DATENAIS
---------------------------------- --------
George VI                          14/12/95
Elizabeth Bowes-Lyon               04/08/00
```

4-Les frères et sœurs du Prince Charles (Nom, DateNaissance).

```
SQL> SELECT NOM, DATENAISSANCE FROM GENEALOGIE
WHERE MERE = (SELECT NUMPER FROM GENEALOGIE WHERE LOWER(NOM) =
'elizabeth ii' AND MERE != 3 AND PERE != 5);
```

NOM	DATENAIS
Prince Charles	14/11/48
Princesse Anne	15/08/50
Prince Andrew	19/02/60
Prince Edward	10/03/64

5-Le nom des individus, le nom de leur père (ou NULL) et le nom de leur mère (ou NULL) (Nom, Nompere, Nommere).

```
SQL> SELECT INDIVIDU.NOM, P.NOM AS PERE, M.NOM AS MERE
FROM GENEALOGIE INDIVIDU
LEFT OUTER JOIN GENEALOGIE P on P.NUMPER = INDIVIDU.PERE
LEFT OUTER JOIN GENEALOGIE M on M.NUMPER = INDIVIDU.MERE;
```

NOM	PERE	MERE
Elizabeth II	George VI	Elizabeth Bowes-Lyon
Margaret du Royaume-Uni	George VI	Elizabeth Bowes-Lyon
Prince Charles	Philip Mountbatten	Elizabeth II
Princesse Anne	Philip Mountbatten	Elizabeth II
Prince Andrew	Philip Mountbatten	Elizabeth II
Prince Edward	Philip Mountbatten	Elizabeth II
Prince William	Prince Charles	Diana Spencer
George VI		
Elizabeth Bowes-Lyon		
Philip Mountbatten		
Diana Spencer		

6-La liste des individus et le nombre de leurs enfants étant dans la base de données (Nom, NbEnfants).

```
SQL> SELECT NOM, COALESCE(NB_FILS,0)+COALESCE(NB_FILLE,0) AS
NB_ENFANTS
FROM GENEALOGIE A LEFT JOIN (SELECT COALESCE(PERE,0) AS PERE,
SUM(1) AS NB_FILS FROM GENEALOGIE GROUP BY PERE)B ON A.NUMPER =
B.PERE LEFT JOIN (SELECT COALESCE(MERE,0) AS MERE, SUM(1) AS
NB_FILLE FROM GENEALOGIE GROUP BY MERE)C ON A.NUMPER = C.MERE;
```

NOM	NB_ENFANTS
Elizabeth Bowes-Lyon	2
Elizabeth II	4
Diana Spencer	1
Philip Mountbatten	4
Prince Andrew	0
George VI	2
Prince William	0
Prince Charles	1
Princesse Anne	0
Margaret du Royaume-Uni	0
Prince Edward	0

TP6

SQL : Requêtes avancées

A-Téléchargez (lipn.univ-paris13.fr/~cabanes/INFO1/) et lancez le script ecole.sql pour créer et remplir les tables :

SQL> @ecole.sql;

B-Formulez et testez les requêtes suivantes en SQL en proposant au moins

1-Donner la liste des noms, des prénoms et des dates de naissance de tous les élèves.

SQL> SELECT NOM, PRENOM, DATE_NAISSANCE FROM ELEVES;

NOM	PRENOM	DATE_NAISS
Brisefer	Benoit	10-12-1978
Génial	Olivier	10-04-1978
Jourdan	Gil	28-06-1974
Spring	Jerry	16-02-1974
Tsuno	Yoko	29-10-1977
Lebut	Marc	29-04-1974
Lagaffe	Gaston	08-04-1975
Dubois	Robin	20-04-1976
Walthéry	Natacha	07-09-1977
Danny	Buck	15-02-1973

2-Donner tous les renseignements sur toutes les activités.

SQL> SELECT * FROM ACTIVITES;

NIVEAU	NOM	EQUIPE
1	Mini foot	Amc Indus
1	Surf	Les planchistes ...
2	Tennis	Ace Club
3	Tennis	Ace Club
1	Volley ball	Avs80
2	Mini foot	Les as du ballon
2	Volley ball	smash

3-Lister les spécialités des professeurs.

SQL> SELECT DISTINCT SPECIALITE FROM PROFESSEURS;
SQL> SELECT SPECIALITE FROM PROFESSEURS GROUP BY SPECIALITE;

SPECIALITE
poésie

poo

réseau
sql

4-Obtenir le nom et prénom des élèves pesant moins de 45 kilos et inscrits en 1ère année ou des élèves inscrits en 2ème année.

SQL> SELECT NOM, PRENOM FROM ELEVES WHERE (POIDS < 45 AND ANNEE = 1) OR ANNEE = 2;

NOM	PRENOM
Brisefer	Benoit
Génial	Olivier
Jourdan	Gil
Spring	Jerry
Lebut	Marc
Dubois	Robin
Danny	Buck

5-Obtenir le nom des élèves dont le poids est compris entre 60 et 80 kilos.

SQL> SELECT NOM FROM ELEVES WHERE POIDS BETWEEN '60' AND '80';
SQL> SELECT NOM FROM ELEVES WHERE POIDS <= 80 AND POIDS >= 60;

NOM
Jourdan
Spring
Lebut
Lagaffe
Dubois

6-Obtenir le nom des professeurs dont la spécialité est ‘poésie’ ou SQL.

SQL> SELECT NOM FROM PROFESSEURS WHERE SPECIALITE = 'poésie' OR SPECIALITE = 'sql';
SQL> SELECT NOM FROM PROFESSEURS WHERE SPECIALITE IN ('poésie', 'sql');

NOM
Bottle
Pastecnov
Selector
Pucette

7-Obtenir le nom des élèves dont le nom commence par ‘L’.

SQL> SELECT NOM FROM ELEVES WHERE NOM LIKE 'L%';

```
NOM
------------------------
Lebut
Lagaffe
```

8-Obtenir le nom des professeurs dont la spécialité est inconnue.

SQL> SELECT NOM FROM PROFESSEURS WHERE SPECIALITE IS NULL;

```
NOM
-------------------------
Francesca
```

9-Obtenir le nom et prénom des élèves pesant moins de 45 kilos et inscrits en 1ère année.

SQL> SELECT NOM, PRENOM FROM ELEVES WHERE POIDS < 45 AND ANNEE = 1;

```
NOM                  PRENOM
-------------------- ------------------------
Brisefer             Benoit
Génial               Olivier
```

10-Obtenir, pour chaque professeur, son nom et sa spécialité. Si cette dernière est inconnue, on souhaite afficher la chaîne de caractères : ‘****’.

SQL> SELECT NOM, COALESCE(SPECIALITE, '****') FROM PROFESSEURS;
SQL> SELECT NOM,
CASE
WHEN (SPECIALITE IS NOT NULL) THEN SPECIALITE
ELSE '****'
END
AS SPECIALITE FROM PROFESSEURS;

```
NOM                  COALESCE(SPECIALITE,
-------------------- --------------------
Bottle               poésie
Bolenov              réseau
Tonilaclasse         poo
Pastecnov            sql
Selector             sql
Vilplusplus          poo
Francesca            ****
Pucette              sql
```

11-Quels sont les noms et prénoms des élèves qui pratiquent du surf au niveau 1. Rédigez cette requête de cinq façons différentes.

SQL> SELECT NOM, PRENOM FROM ELEVES WHERE NUM_ELEVE IN (SELECT NUM_ELEVE FROM ACTIVITES_PRATIQUEES WHERE NOM = 'Surf' AND NIVEAU = 1);

SQL> SELECT NOM, PRENOM FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE FROM ACTIVITES_PRATIQUEES WHERE NOM = 'Surf' AND NIVEAU = 1);

SQL> SELECT NOM, PRENOM FROM ELEVES WHERE NUM_ELEVE = ANY (SELECT NUM_ELEVE FROM ACTIVITES_PRATIQUEES WHERE NOM = 'Surf' AND NIVEAU = 1);

SQL> SELECT NOM, PRENOM FROM ELEVES WHERE 0 < (SELECT COUNT(*) FROM ACTIVITES_PRATIQUEES WHERE NOM = 'Surf' AND NIVEAU = 1 AND ELEVES.NUM_ELEVE = ACTIVITES_PRATIQUEES.NUM_ELEVE);

SQL> SELECT NOM, PRENOM FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE, NIVEAU FROM ACTIVITES_PRATIQUEES WHERE ACTIVITES_PRATIQUEES.NOM ='Surf' AND NIVEAU = 1);

NOM	PRENOM
Brisefer	Benoit
Spring	Jerry
Tsuno	Yoko

12-Obtenir le nom des élèves de l’équipe AMC INDUS.

SQL> SELECT NOM FROM ELEVES WHERE NUM_ELEVE IN (SELECT NUM_ELEVE FROM ACTIVITES_PRATIQUEES NATURAL JOIN ACTIVITES WHERE EQUIPE IN ('Amc Indus'));
SQL> SELECT NOM FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE, NIVEAU FROM ACTIVITES_PRATIQUEES NATURAL JOIN (SELECT * FROM ACTIVITES WHERE EQUIPE = 'Amc Indus'));

NOM
Brisefer
Génial
Tsuno
Dubois
Walthéry
Danny

13-Obtenir les pairs de noms de professeurs qui ont la même spécialité.

```
SQL> SELECT J1.NOM, J2.NOM , J1.SPECIALITE FROM PROFESSEURS J1,
PROFESSEURS J2 WHERE J1.SPECIALITE = J2.SPECIALITE AND J1.NOM !=
J2.NOM;

NOM                  NOM                  SPECIALITE
-------------------- -------------------- --------------------
Vilplusplus          Tonilaclasse         poo
Pucette              Pastecnov            sql
Selector             Pastecnov            sql
Pucette              Selector             sql
Pastecnov            Selector             sql
Tonilaclasse         Vilplusplus          poo
Selector             Pucette              sql
Pastecnov            Pucette              sql
```

14-Pour chaque spécialité sql/SQL, on demande d'obtenir son nom son salaire mensuel actuel et son augmentation mensuelle depuis son salaire de base.

```
SQL> SELECT NOM , SALAIRE_ACTUEL, ROUND((SALAIRE_ACTUEL -
SALAIRE_BASE)/(MONTHS_BETWEEN(DER_PROM,DATE_ENTREE))) AS
Augmentation_mensuelle FROM PROFESSEURS WHERE SPECIALITE = 'sql';

NOM                  SALAIRE_ACTUEL AUGMENTATION_MENSUELLE
-------------------- -------------- ----------------------
Pastecnov                    2500000
Selector                     1900000                      0
Pucette                      2500000                   5764
```

15-Obtenir le nom des professeurs dont l'augmentation relative au salaire de base dépasse 25%.

```
SQL> SELECT NOM FROM PROFESSEURS WHERE
(SALAIRE_ACTUEL/SALAIRE_BASE)>= 1.25;

NOM
--------------------
Bottle
Bolenov
Francesca
Pucette
```

16-Afficher les points de Tsuno obtenus dans chaque cours sur 100 plutôt que sur 20.

```
SQL> SELECT NUM_COURS, (POINTS*5) AS POINTS_SUR_CENT FROM
RESULTATS NATURAL JOIN ELEVES WHERE NOM = 'Tsuno';
SQL> SELECT NUM_COURS, (POINTS*5) AS POINTS_SUR_CENT FROM
RESULTATS WHERE NUM_ELEVE = (SELECT NUM_ELEVE FROM ELEVES
WHERE NOM ='Tsuno');
```

```
NUM_COURS POINTS_SUR_CENT
---------- ---------------
         1              25
         2            32,5
         4              65
         5              65
```

17-Obtenir le poids moyen des élèves de 1ère année.

```
SQL> SELECT AVG(POIDS) AS MOYENNE_POIDS FROM ELEVES WHERE
ANNEE = 1;

MOYENNE_POIDS
-------------
         48,4
```

```
--Q18--
--Obtenir le total des points de l'élève numéro 3.
SELECT SUM(POINTS) AS TOTAL_POINTS FROM RESULTATS WHERE
NUM_ELEVE = 3;

TOTAL_POINTS
------------
          65
```

19-Obtenir la plus petite et la plus grande cote de l'élève Brisefer.

```
SQL> SELECT MIN(POINTS) AS MIN, MAX(POINTS)AS MAX FROM RESULTATS
WHERE NUM_ELEVE = (SELECT NUM_ELEVE FROM ELEVES WHERE NOM =
'Brisefer');
SQL> SELECT MIN(POINTS) AS MIN, MAX(POINTS) AS MAX FROM
RESULTATS  NATURAL JOIN ELEVES WHERE NOM = 'Brisefer';

         MIN       MAX
---------- ----------
         8           20
```

20-Obtenir le nombre d'élèves inscrits en deuxième année.

```
SQL> SELECT COUNT(*) FROM ELEVES WHERE ANNEE = 2;

  COUNT(*)
----------
         5
```

21-Quelle est l'augmentation mensuelle moyenne des salaires des professeurs de SQL ?

```
SQL> SELECT ROUND(AVG((SALAIRE_ACTUEL - SALAIRE_BASE)/(MONTHS_BETWEEN(DER_PROM,DATE_ENTREE)))) AS MOYENNE_Augmentation_mensuelle FROM PROFESSEURS WHERE SPECIALITE = 'sql';

MOYENNE_AUGMENTATION_MENSUELLE
------------------------------
                          2882
```

22-Obtenir l'année de la dernière promotion du professeur Pucette.

```
SQL> SELECT to_char(DER_PROM, 'YYYY') FROM PROFESSEURS WHERE NOM = 'Pucette';

TO_C
----
1996
```

23-Pour chaque professeur, afficher sa date d'ébauche, sa date de dernière promotion ainsi que le nombre d'années écoulées entre ces deux dates.

```
SQL> SELECT DATE_ENTREE, DER_PROM, ROUND(MONTHS_BETWEEN(DER_PROM,DATE_ENTREE)/12) AS DIFF FROM PROFESSEURS;

DATE_ENTRE DER_PROM     DIFF
---------- ---------- ----------
01-10-1970 01-10-1988         18
15-11-1968 01-10-1998         30
01-10-1979 01-01-1989          9
01-10-1975
15-10-1982 01-10-1988          6
25-04-1990 05-06-1994          4
01-10-1975 11-01-1998         22
06-12-1988 29-02-1996          7
```

24-Afficher l'âge moyen des élèves. Cet âge moyen sera exprimé en année

```
SQL> SELECT AVG(ROUND((CURRENT_DATE - DATE_NAISSANCE)/365)) AS AGE_MOYEN FROM ELEVES;

 AGE_MOYEN
----------
         44
```

25-Afficher le nom des professeurs pour lesquels il s'est écoulé plus de 50 mois entre l'embauche et la première promotion.

SQL> SELECT NOM FROM PROFESSEURS WHERE (MONTHS_BETWEEN(DER_PROM,DATE_ENTREE)) > 50 ;

```
NOM
-------------------------
Bottle
Bolenov
Tonilaclasse
Selector
Francesca
Pucette
```

26-Obtenir la liste des élèves qui auront au moins 24 ans dans moins de 4 mois.

SQL> SELECT NOM FROM ELEVES WHERE (MONTHS_BETWEEN(SYSDATE, DATE_NAISSANCE)+4)/12 > 24 ;

```
NOM
-------------------------
Brisefer
Génial
Jourdan
Spring
Tsuno
Lebut
Lagaffe
Dubois
Walthéry
Danny
```

27-Obtenir une liste des élèves classés par année et par ordre alphabétique.

SQL> SELECT * FROM ELEVES ORDER BY ANNEE, NOM ASC;

NUM_ELEVE	NOM	PRENOM	DATE_NAISS	POIDS	ANNEE	S
1	Brisefer	Benoit	10-12-1978	35	1	M
2	Génial	Olivier	10-04-1978	42	1	M
7	Lagaffe	Gaston	08-04-1975	61	1	M
5	Tsuno	Yoko	29-10-1977	45	1	F
9	Walthéry	Natacha	07-09-1977	59	1	F
10	Danny	Buck	15-02-1973	82	2	M
8	Dubois	Robin	20-04-1976	60	2	M
3	Jourdan	Gil	28-06-1974	72	2	F
6	Lebut	Marc	29-04-1974	75	2	M
4	Spring	Jerry	16-02-1974	78	2	M

28-Afficher en ordre décroissant les points de Tsuno obtenus dans chaque cours sur 100 plutôt que sur 20.

SQL> SELECT NUM_COURS, (POINTS*5) AS POINTS_SUR_CENT FROM RESULTATS NATURAL JOIN ELEVES WHERE NOM = 'Tsuno' ORDER BY (POINTS*5) DESC ;

```
 NUM_COURS POINTS_SUR_CENT
---------- ---------------
         4              65
         5              65
         2            32,5
         1              25
```

29-Obtenir pour chaque élève de 1ère année son nom et sa moyenne.

SQL> SELECT NOM, AVG(POINTS) AS MOYENNE_ELEVES FROM RESULTATS NATURAL JOIN ELEVES WHERE ANNEE = 1 GROUP BY NOM;
SQL> SELECT NOM, MOYENNE_ELEVES FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE, AVG(POINTS) AS MOYENNE_ELEVES FROM RESULTATS GROUP BY (NUM_ELEVE)) WHERE ANNEE = 1;

```
NOM                       MOYENNE_ELEVES
------------------------- --------------
Brisefer                          13,375
Génial                               9,5
Lagaffe                           10,375
Tsuno                              9,375
Walthéry                          15,875
```

30-Obtenir la moyenne des points de chaque élève de 1ère année dont le total des points est supérieur à 40.

SQL> SELECT NOM, MOYENNE_ELEVES FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE, AVG(POINTS) AS MOYENNE_ELEVES FROM RESULTATS GROUP BY (NUM_ELEVE) HAVING SUM(POINTS)> 40) WHERE ANNEE = 1 ;

```
 NOM                       MOYENNE_ELEVES
 ------------------------- --------------
 Brisefer                          13,375
 Lagaffe                           10,375
 Walthéry                          15,875
```

SQL> SELECT NUM_ELEVE, AVG(POINTS) AS MOYENNE_ELEVES FROM RESULTATS GROUP BY(NUM_ELEVE) HAVING SUM(POINTS) > 40 AND NUM_ELEVE IN(SELECT NUM_ELEVE FROM ELEVES WHERE ANNEE = 1);

```
 NUM_ELEVE       MOYENNE_ELEVES
---------- --------------
         1              13,375
         7              10,375
         9              15,875
```

31-Obtenir le maximum parmi les totaux de chaque élève.

SQL> SELECT MAX(SUM(POINTS)) AS TOTAL_MAX FROM RESULTATS GROUP BY NUM_ELEVE;
SQL> SELECT MAX(TOTAL) AS TOTAL_MAX FROM (SELECT SUM(POINTS) AS TOTAL FROM RESULTATS GROUP BY NUM_ELEVE);

```
 TOTAL_MAX
----------
        65
```

32-Obtenir le nom des élèves qui jouent dans l'équipe AMC INDUS.

SQL> SELECT NOM FROM ELEVES WHERE NUM_ELEVE IN (SELECT NUM_ELEVE FROM ACTIVITES_PRATIQUEES NATURAL JOIN ACTIVITES WHERE EQUIPE IN ('Amc Indus'));
SQL> SELECT NOM FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE, NIVEAU FROM ACTIVITES_PRATIQUEES NATURAL JOIN (SELECT * FROM ACTIVITES WHERE EQUIPE = 'Amc Indus'));

```
NOM
-------------------------
Brisefer
Génial
Tsuno
Dubois
Walthéry
Danny
```

33-Quels sont les élèves de 1ère année dont la moyenne est supérieure à la Moyenne de la 1ère année ?

SQL> SELECT NUM_ELEVE, AVG(POINTS) FROM RESULTATS GROUP BY NUM_ELEVE HAVING NUM_ELEVE
IN(SELECT NUM_ELEVE FROM ELEVES WHERE ANNEE = 1) AND AVG(POINTS) >
(SELECT AVG(POINTS) FROM RESULTATS WHERE NUM_ELEVE IN (SELECT NUM_ELEVE FROM ELEVES WHERE ANNEE = 1));

SQL> SELECT NUM_ELEVE, AVG(POINTS) FROM (SELECT NUM_ELEVE FROM ELEVES WHERE ANNEE =

1) NATURAL LEFT JOIN RESULTATS GROUP BY NUM_ELEVE HAVING AVG(POINTS) > (SELECT AVG(POINTS) FROM ELEVES NATURAL JOIN RESULTATS);

NUM_ELEVE	AVG(POINTS)
1	13,375
9	15,875

34-Obtenir le nom et le poids des élèves de 1ère année plus lourds que n'importe quel élève de 2ème année.

SQL> SELECT NOM, POIDS FROM ELEVES WHERE ANNEE = 1 AND POIDS > ALL(SELECT POIDS FROM ELEVES WHERE ANNEE = 2);
SQL> SELECT NOM, POIDS FROM ELEVES WHERE ANNEE = 1 AND POIDS > (SELECT MAX(POIDS) FROM ELEVES WHERE ANNEE = 2);

aucune ligne s  lectionne

35-Obtenir le nom et le poids des élèves de 1ère année plus lourds qu'un élève quelconque de 2ème année.

SQL> SELECT NOM, POIDS FROM ELEVES WHERE ANNEE = 1 AND POIDS > ANY(SELECT POIDS FROM ELEVES WHERE ANNEE = 2);
SQL> SELECT NOM, POIDS FROM ELEVES WHERE ANNEE = 1 AND POIDS > (SELECT MIN(POIDS) FROM ELEVES WHERE ANNEE = 2);

NOM	POIDS
Lagaffe	61

36-Obtenir le nom, le poids et l'année des élèves dont le poids est supérieur au poids moyen des élèves étant dans la même année d'études.

SQL> SELECT NOM, POIDS, ANNEE FROM ELEVES NATURAL JOIN (SELECT ANNEE, AVG(POIDS) AS MOYENNE_POIDS FROM ELEVES GROUP BY ANNEE) WHERE POIDS > MOYENNE_POIDS;

NOM	POIDS	ANNEE
Walthéry	59	1
Lagaffe	61	1
Danny	82	2
Lebut	75	2
Spring	78	2

37-Obtenir le nom des professeurs qui ne donnent pas le cours numéro 1.

```
SQL> SELECT DISTINCT NOM FROM PROFESSEURS WHERE NUM_PROF
IN(SELECT NUM_PROF FROM CHARGE WHERE NUM_COURS != 1 );
SQL> SELECT DISTINCT NOM FROM PROFESSEURS NATURAL JOIN
(SELECT NUM_PROF FROM CHARGE WHERE NUM_COURS != 1);
```

```
NOM
------------------------
Pastecnov
Bottle
Francesca
Tonilaclasse
Pucette
```

38-Obtenir le nom des élèves de 1ère année qui ont obtenu plus de 60 % et qui jouent au tennis.

```
SQL> SELECT NOM FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE,
AVG(POINTS)*5 AS POURCENTAGE FROM RESULTATS GROUP BY
NUM_ELEVE)
WHERE POURCENTAGE > 60 AND NUM_ELEVE IN (SELECT NUM_ELEVE
FROM ACTIVITES_PRATIQUEES WHERE NOM = 'Tennis');

SQL> SELECT NOM FROM ELEVES NATURAL JOIN (SELECT NUM_ELEVE,
AVG(POINTS)*5 AS POURCENTAGE FROM RESULTATS GROUP BY
NUM_ELEVE)
WHERE POURCENTAGE > 60 AND NUM_ELEVE = ANY (SELECT NUM_ELEVE
FROM
ACTIVITES_PRATIQUEES WHERE NOM = 'Tennis');
```

```
NOM
------------------------
Brisefer
Jourdan
```

39-Professeurs qui prennent en charge tous les cours de deuxième année ; on demande le Numéro et le nom.

```
SQL> SELECT P1.NUM_PROF, P1.NOM
FROM PROFESSEURS P1
WHERE NOT EXISTS
( SELECT *
FROM COURS P2
```

```
WHERE P2.ANNEE = 2
AND NOT EXISTS
(SELECT * FROM CHARGE P3
  WHERE P1.NUM_PROF = P3.NUM_PROF
  AND P2.NUM_COURS = P3.NUM_COURS ));

SQL> SELECT NUM_PROF, NOM
  FROM PROFESSEURS A
  WHERE
  (SELECT COUNT(*) FROM COURS NATURAL JOIN CHARGE P
  WHERE ANNEE = 2
  AND
  P.NUM_PROF = A.NUM_PROF )
  =
  (SELECT COUNT(*) FROM COURS WHERE ANNEE = 2);

   NUM_PROF NOM
---------- -------------------------
         3 Tonilaclasse
         8 Pucette
```

40-Élèves qui pratiquent toutes les activités ; on demande le Numéro et le nom.

```
SQL> SELECT P1.NUM_ELEVE, P1.NOM
FROM ELEVES P1
WHERE NOT EXISTS
(SELECT * FROM ACTIVITES P2
WHERE NOT EXISTS
(SELECT * FROM ACTIVITES_PRATIQUEES P3
WHERE P1.NUM_ELEVE = P3.NUM_ELEVE
AND P2.NOM = P3.NOM
AND P2.NIVEAU = P3.NIVEAU));

SQL> SELECT NUM_ELEVE, NOM
  FROM ELEVES A
  WHERE
    (SELECT COUNT(*) FROM ACTIVITES NATURAL JOIN
ACTIVITES_PRATIQUEES P
    WHERE P.NUM_ELEVE = A.NUM_ELEVE) =
    (SELECT COUNT(*) FROM ACTIVITES);

    aucune ligne s?lectionn?e
```

TP7

SQL : Vues et Arbres

1-Téléchargez (lipn.univ-paris13.fr/~cabanes/INFO1/) et lancez le script famille.sql pour créer et remplir la table.

SQL> @famille.sql;

Les personnes

NUMERO	NOM	PRENOM	DATENAISSA	S	PERE	MERE
98	CLEMENT	JEAN-BAPTISTE	01-01-1890	M		
87	GABRIEL	EVE	01-01-1892	F		
99	LEBON	NICOLAS	01-01-1895	M		
88	HONNEUR	CLEMENCE	01-01-1900	F		
22	CLEMENT	JEAN-BAPTISTE	13-11-1910	F	98	
1	LEBON	MICHEL	04-08-1920	M	99	88
77	PARIS	LOUIS	20-03-1924	M		
2	CLEMENT	EVE	13-11-1928	F	98	87
78	GATEAU	EVELYNE	20-03-1936	F		
15	LEBON	GABRIEL	04-08-1936	M	99	88
34	CLEMENT	RAOUL	01-01-1941	M	22	
33	LEBON	ROSE	16-06-1951	F	1	2
7	LEBON	FRANCOIS	22-02-1954	M	1	2
3	LEBON	NICOLAS	17-09-1958	M	1	2
76	PARIS	AMELIA	20-10-1958	F	77	78
9	LEBON	FRANCOISE	01-09-1963	F	15	
4	PARIS	INES	22-11-1969	F	77	78
55	CLEMENT	MARIE	13-08-1978	F	33	34
8	LEBON	MICHEL	05-09-1987	M	7	9
75	AMMAR	SERGES	20-10-1987	M		76
10	LEBON	AIME	24-05-1993	M	7	9
11	LEBON	ALEXANDRE	16-07-1994	M	7	9
5	LEBON	CLEMENCE	19-06-2001	F	3	4
6	LEBON	ADAM	19-06-2001	M	3	4
56	MEDECIN	LINA	22-02-2002	F	55	

2-Dessinez le Schémas E/A de cette table

3-Créez une vue permettant d'afficher les ancêtres d'une personne avec la commande CONNECT BY. Le numéro de la personne doit être demandé au moment où la requête se lance (utilisez &num).

```
SQL> CREATE VIEW ARBRE
  2  AS SELECT DISTINCT * FROM PERSONNES
  3  START WITH NUMERO = &NUM
  4  CONNECT BY NUMERO = PRIOR PERE OR NUMERO = PRIOR MERE;

Entrez une valeur pour num : 12
ancien   3 : START WITH NUMERO = &NUM
nouveau   3 : START WITH NUMERO = 12

Vue créée.
```