**fBilan Algèbre Relationnel**

union : T = T1 U T2

différence : T = T1 - T2

intersection : T = T1 inter T2

produit cartésien : T = T1 \* T2 (concaténé deux listes différentes)

division : R2 = R/R1 (R contient A,B; R1 contient A alors R2 contient B qui correspond toutes les valeurs de A)

ex : A : s1 p1; s1 p2; s2 p2; s3 p3 B : p1 p2

alors A/B = s1 car s1 correspond à p1 et p2

pi (= select) : restriction/selection

ex : pi nom Tab\_Pilote

sigma (=where) : projection

ex : sigma capacite = 10 Avion

nature join (where A.xxx=B.xxx) : jointure

ex : T3 = T1 nature join (T1.xx=T2.xx) T2

rho : renommer un champs

ex : R = rho champ->champ’ (R1)

**Bilan SQL**

1 - création des tableaux

create table nom\_table(

nom\_attribut type\_attribut(taille) not null/…

nom\_attribut type default 10

constraint cle\_nom primary key (nom\_attribut),

constraint fk\_nomfk foreign key (nom\_attribut) references nom\_table(nom\_attribut)

constraint check\_name check(div.name = dd.name),

);

type : int / smallint / float(p,s) / real(p,s) / double(p,s) / numeric(p,s) / decimal(p,s)/ char(n) / varchar(n) / date

date ; ‘2011-10-01’

check : condition

p - ca fait combien de chiffres

s - nb de chiffre après le virgule

p-s de chiffre avant le virgule

2 - insertion des lignes

insert into nom\_table/nom\_vue values(xxx,xxx,xxx);

insert into table (nom\_colonne1, nom\_colonne2, ……) values (xxx,xxx,xxx,......);

3 - modifier des valeurs d’une ligne

update nom\_tab

set nom\_col = valeur

where condition

4 - supprimer/éliminer les tuples d’un tableau ou d’une vue

delete from nom\_tab

where condition

5 - selection (select...from) + projection (where...) + Jointure (A.xx = B.xx)

select nom\_col

from nom\_tab1, nom\_tab2

where tab1.xx = tab2.xx;

order by…. asc/desc;

select distinct -- pour ne pas avoir doublons

6 - prédicats

=/</>/not/and/between x and y

chaine like ‘%xx%’ --pour chercher une chaîne de caractères selon une partie de la chaîne

%xx selon partie droite

xx% selon partie gauche

%xx% selon partie au milieu

7 - fonctions

lower(a) : minuscul

upper(a) : majuscul

soundex(a) : retourner 4 caractères en fonction de la sonorité d’une chaîne de caractère

ex: where soundex(prenom) = soundex(‘jone’)

résultat : John smith

abs(a) : renvoie la valeur absolue |-1| = 1

sqrt(a) : mettre au carré

length(a) : longeur

8 - agrégats

avg(a) : moyenne

count(a) : nb de ligne

min(a) : minimum

max(a) : maximum

sum(a) : somme

stddev(a) : appliquer la fonction à tous les valuers (all par défaut)

variance(a) : variance (方差)

9 - jointure interne

inner join...on : il faut précisier attribut de jointure

select xxx

from tab1

inner join nom\_tab2 on (tab1.col = tab2.col)

where condition

ou bien

inner join nom\_tab2 using (nom\_col)

where condition;

natual inner join : ne pas précise l’attribut de jointure

select tab1.col

from tab1 natual inner join tab2

where condition

10 - jointure externe

select xx

from tab1

full outer join tab2 on tab1.xx=tab2.xx : garder toutes les données d’une des deux tables même si elles ne vérifient pas la condition de jointure

ou bien

full outer join tab1 using(cle\_prim)

revoyer les lignes de tab1 existant dans tab2, s’il n’existe pas, remplir par null

select xxx

from tab1

left outer join tab2 on tab1.xx = tab2.xx;

ou

right outer join on

11 - opérateurs ensemblistes

union

except/minus : tab1 - tab2

intersect

assign to temp1

select xx

from xx

where xx;

assigne to temp2

select xx

from xxx

where xx;

temp1 union/minus/intersect temp2

12 - requêtes imbriquées

select xx

from xx

where in/exists/not exists/all/any

(select xx

from xx

where xx);

ex : where ville in (select entrepot from xx where cap > 250)

13 - group by

partitionne la table

14 - having

introduire une qualification sur les sous tables issues du group by

select numav

from xx

group by num av

having count(numav) > 3;

15 - some, any, all

some = any : exists

all : tous

select matricule

from pilote

where age<30

and salaire > any

(select salaire from pilote where age > 40);

16 - drop table

drop table tab\_nom

drop table tab\_nom cascade constraints;

17 - alter table

alter table nom\_tab

rename to new\_name

/add nom\_col

/modify nom\_col default valeur/d’autre constraintes

/drop column nom\_col

/rename column old\_name to new\_name

ex:

//ajouter des colonnes

alter table country

add (duty decimal(2,2) default 1 check duty<10, visa\_need varchar(20));

//modifier des colonnes

alter table country

modify (duty decimal(3,2));

//supprimer des colonnes

alter table country

drop column duty cascade constraints;

18 - langage controle de données

system\_user : nom de l’utilisateur connecté

session\_user : nom d’utilisateur qui a ouvert une session

current\_user : nom de l’utilisateur courant

(utiliser avec select, ex : select system\_user; )

create user nom with login password ‘abab’ createdb nocreaterole;

alter role nom rename to new\_name

alter role nom with password null/createdn/nocreaterole…

drop role if exists nom

19 - droit d’accès

grant select,update,insert,delete,... to bastide

grant select on nom\_table to bastide

grant select on mot\_reserve to public

grant select on database nom\_data to qqn

revoke select on nom\_tab to lamine

20 - create view

create view nom\_view

as select xx

from xx

where condition

with check option

21 - schéma

create schema authorization qqn08;

create table qqn08.nomtab as table nomtab;

select \* from qqn08.tab;

22 - cast

where cast('2000-09-15' - date\_naiss as int)/365 <=30;

where cast(current\_date - date\_naiss as int) <

cast(current\_date -

(select date\_naiss from sportif

where upper(nom) = upper('Ferrari')) as int

);

left join gagner\_individuel g1 on g1.num\_licence = sf.num\_licence

add column date\_de\_naissance date;

alter table pilote add constraint ck\_birth check (date\_de\_naissance >= ‘01/01/1950’ and date\_de\_naissance<01/01/1981);

alter table Pilote add constraint ck\_birth check(Extract (year from date\_de\_naissance) between 1950 and 1980;

alter table pilote add constraint ck\_birth2 check(date\_part(‘year’, date\_de\_naissance) between 1950 and 1980);

set datastyle to ‘sql, DMY’;

select round(avg(salaire),2) as ‘salaire moyen’ from pilote;

select p1.plnom, extract(year from p1.date\_de\_naissance) as “Annee de naissance”

from pilote p1

where Exsits(

select p2.plnom, p2 date\_de\_naissance from pilote p2

where date\_part(‘year’, p2.date\_de\_naissance) =

date\_part(‘year’, p1.date\_de\_naissance)

and p1.plnum<>p2.plnum);

where date\_part(‘year’, p2.date de naissance) = date\_part(‘year’, p1.date\_de\_naissance)

and p1.plnum!=p2.plnum;

select plnom, extract(‘year’ from age(date\_de\_naissance)) as Age from pilote;

select plnom, data\_trunc(‘year’,age(date\_de\_naissance)) as Age from pilote;