type

entiers: bit, short, int, long
réels: float real
text: varchar, text
boolean: bit
date: date

alter → modifier
add → ajouter
drop → supprimer

LDD

Creation des tables :

```
create table NOM_TABLE(CHAMP1 TYPE, CHAMP2 TYPE, ....)
```

Les contraints

1. Clé primaire :

a)***Méthode 1 :***

```
create table NOM_TABLE(CHAMP TYPE primary key);
```

b) Méthode 2 :

```
create table NOM_TABLE (CHAMP TYPE,constraint PK_NOM_TABLE primary key(CHAMP));
```

a)Méthode 3 :

```
create table NOM_TABLE (CHAMP TYPE );
alter table NOM_TABLE add constraint PK_NOM_TABLE primary key(CHAMP);
```

2. Clé etrengaire:

a)Méthode 1 :

```
create table TABLE1(CHAMP TYPE foreign key references TABLE2(CHAMP2)));
```

b) Méthode 2 :

```
create table TABLE1(CHAMP TYPE,constraint FK_TABLE1_TABLE2 foreign key(CHAMP1)
references TABLE2(CHAMP2));
```

a)Méthode 1 :

```
create table TABLE1(CHAMP TYPE );
alter table TABLE1 add constraint FK_TABLE1_TABLE2 foreign key(CHAMP1)
references TABLE2(CHAMP2);
```

3. Check (condition):

```
alter table NOM_TABLE add constraint NOM_CHACK check(condtion)
```

4. Default:

```
alter table NOM_TABLE add constraint NOM_DE_DEFAUNT default VALEUR for
NOM_CHAMP
```

5. Unique

- alter table NOM_TABLE add constraint NOM_DE_UNIQUE unique (LES CHAMPS)

Ajouter, supprimer ou Modifier les Champs dans une table :

ajouter: alter table NOM_TABLE add NOM_CHAMP TYPE;
supprimer: alter table NOM_TABLE drop column NOM_CHAMP;
Modefire: alter table NOM_TABLE alter column NOM_CHAMP TYPE;

index

create index NOM_INDEX on NOM_TABLE(NOM_CHAMP)
drop index NOM_INDEX on NOM_TABLE

LMD

insert → inserer
delete → supprimer
update → modifier
select → selectioner

Insertion des donnees

a)insert into NOM_TABLE (CHAMP1,CHAMP2,...CHAMP_n)
values(VALEUR1,VALEUR2,...VALEUR_n)
b)insert into NOM_TABLE values (VALEUR1,VALEUR2,.....)
//les valuer de tous les champs
c)**insérer plusieurs enregistrements en même temps**
insert into NOM_TABLE
values (VALEUR1_1,VALEUR2_1,.....),(VALEUR1_2,VALEUR2_2,.....).....

Sélection:

select CHAMP1,CHAMP2... from TABLE1,TABLE2... where (condtion)

Modification

update NOM_TABLE set CHAM =NEW_VALUE [where (condition)]

Suppression

delete from NOM_TABLE where (condetion)

Création d'une table avec insertion

select CHAMP1,CHAMP2... into NOM_TABLE_NEW from TABLE1,TABLE2...

Les View

create view NOM_VIEW as (code select)
-**supprimer,utiliser,modifier View**
drop view NOM_VIEW

```
 select * from NOM_VIEW
 alter view NOM_VIEW as (code select)
```

les Fonction d'agrégation : max,min sum,avg,count

```
select chm1,FUNCTION(cham2)
from table1
group by chm1
having condition(FUNCTION(cham2))
```

Les Fonctions Utilasateur :

Crier la fonction

```
create Function NOM_FUNCTION(@PAR1 type,@PAR2 type ....)
returns TYPE_FONCTION
as
begin
      <<code>>
      return @valeur
end
```

Utiliser la fonction

```
select dbo.NOM_FUNCTION(@PAR1 ,@PAR2  ....)
```

Les Fonctions Table :

Crier la fonction table

```
create function NOM_FUNCTION(@PAR1 type,@PAR2 type ....)
returns table
as
      return
      <<code select >>
```

Afficher fonction table

```
select * from dbo.NOM_FONCTION_TABLE
```

Les procédures stockées :

```
create proc NOM_PROCEDUR
as
begin
<<code>>
end
```

Exécute proc

```
exec NOM_PROCEDUR
```

```
create proc NOM_PROCEDUR(@PAR1 type,@PAR2 type ....)
as
begin
<<code>>
end
```

Exécute proc

```
exec NOM_PROCEDUR @PAR1,@PAR2.....
```

```
create proc NOM_PROCEDUR(@PAR1 type,@PAR2 type output ....)
as
begin
          <<code>>
end
```

Exécute proc

```
declare @var type,
exec NOM_PROCEDUR @PAR1,@var output
select @var
```

Les transactions :

```
begin try
     begin tran
          <<code>>
     commit tran
end try

begin catch
     rollback tran
end catch
```

Les cursor :

```
declare NOM_CURSOR cursor for select chmp1,chmp2 from table1...
declare @var1 type,
          @var2 type ..

open NOM_CURSOR
     fetch next from NOM_CURSOR into @var1,@var2...
     while(@@fetch_status=0)
     begin
          <<code>>
          fetch next from NOM_CURSOR into @var1,@var2...
     end
close NOM_CURSOR
deallocate NOM_CURSOR
```

Les Trigger

```
create Trigger NOM_TRIGGER on NOM_TABLE
for insert,[delete,select,update]
as
begin
     <<code>>
end
```

RAZ ID identity à ZERO

```
DBCC checkident (NOM_TABLE,reseed,0);
```

UNION, EXCEPT et INTERSECT

- UNION

Pour unir les lignes de résultat de deux requêtes select, on a une requête de la forme **:**

```
SELECT ... FROM table1 WHERE ...
UNION
SELECT ... FROM table2 WHERE ...
```

- **EXCEPT**

Pour faire la différence entre les lignes de résultat de deux requêtes select, on utilise INTERSECT comme suit **:**

```
SELECT ... FROM table1 WHERE ...
EXCEPT
SELECT ... FROM table2 WHERE ...
```

- **INTERSECT**

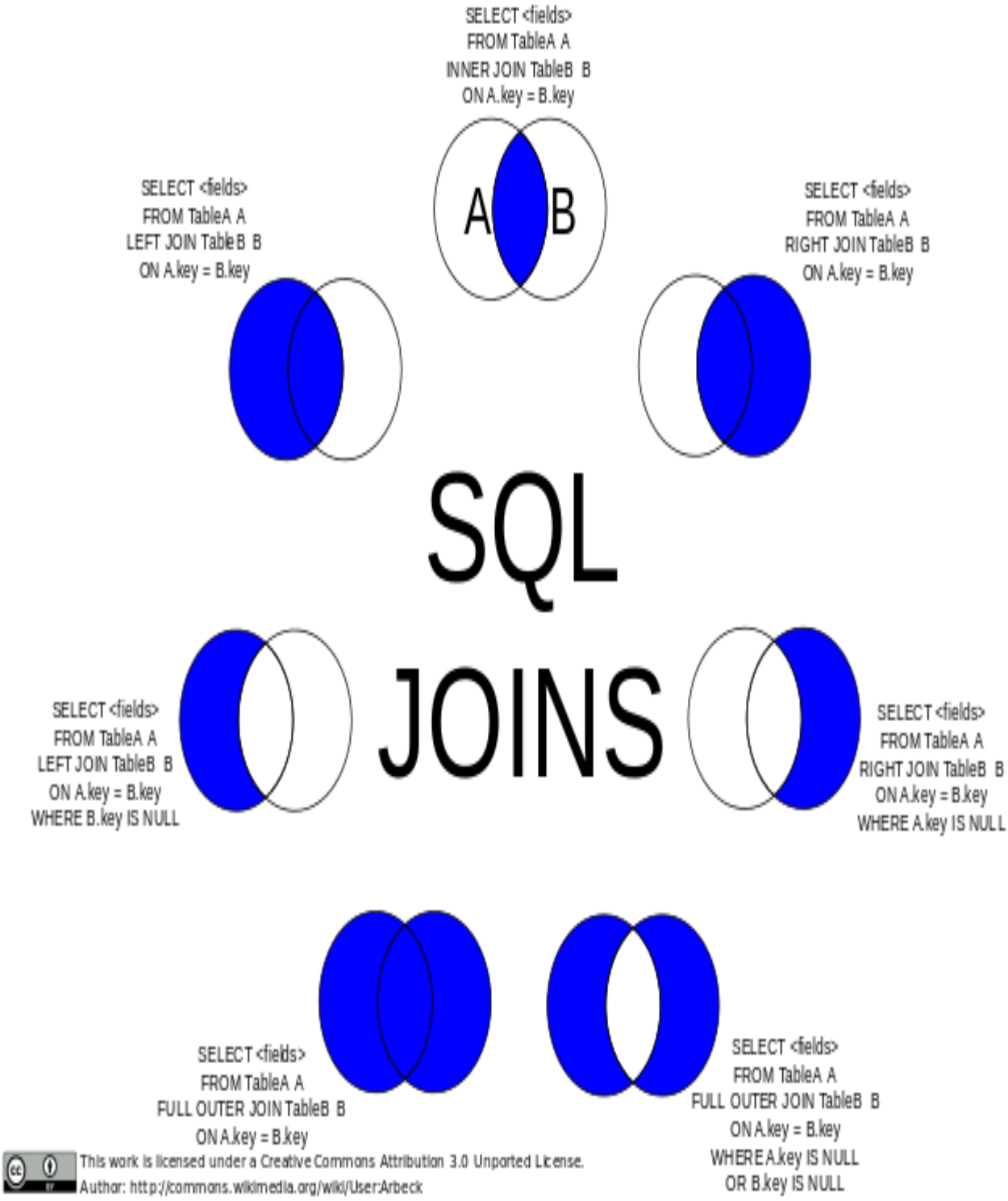
Pour avoir une intersection entre les lignes de résultat de deux requêtes select, voici la requête correspondante **:**

```
SELECT ... FROM table1 WHERE ...
INTERSECT
SELECT ... FROM table2 WHERE ...
```

Sytaxe iif,case,while :

```
iif(condition, si vrai, si faut)
-------------------------------------------------------
case when <<condition 1>> then valeur1
      when <<condition 2>> then valeur2
      ...
      else valeur_n end
-------------------------------------------------------
while(condition)
begin
<<code>>
end
```

Les jointeurs



LSD

Sauvegarder

```
backup database NOM_DATA_BASE to disk = 'd:\NOM_DATA_BASE.bak'
```

Déconnecter les utilisateurs

```
alter database NOM_DATA_BASE set single_user  with rollback after DUREE
```

Restaurer

```
restore database NOM_DATA_BASE from disk = 'd:\NOM_DATA_BASE.bak'
```

Redonner la possibilité aux utilisateurs de se connecter

```
alter database NOM_DATA_BASE set multi_user
```

Création de schéma

```
create schema NOM_SCHEMA
```

Création table associer un schéma

```
create table NOM_SCHEMA.NOM_TABLE(champ1 type ,....)
```

Création des logins

```
create login NOM_LOGIN with password = 'PASS WORD' must_change,
 CHECK_EXPIRATION=ON
```

Création des utilisateurs

```
create user NOM_USER for login NOM_LOGIN
```

Ajouter des droits

Pour table

```
grant select, [insert, delete, update] on SCHEMA.NOM_TABLE to NOM_USER
```

Pour schéma

```
grant select, [insert, delete, update] on schema :: NOM_SCHEMA to NOM_USER
```

Pour champ

```
grant select (NOM_CHAM) on SCHEMA.NOM_TABLE to NOM_USER
```

Enlever des droits

Pour table

```
revoke select, [insert, delete, update] on SCHEMA.NOM_TABLE to NOM_USER
```

Pour schéma

```
revoke select, [insert, delete, update] on schema :: NOM_SCHEMA to NOM_USER
```

Ajouter des droits

Pour table

```
deny select, [insert, delete, update] on SCHEMA.NOM_TABLE to NOM_USER
```

Pour schéma

```
deny select, [insert, delete, update] on schema :: NOM_SCHEMA to NOM_USER
```

Création des roles

```
create role NOM_ROLE
```

Ajout des utilisateurs au role

```
exec sp_addrolemember NOM_ROLE, NOM_USER
```

Les Collections

```
create database test99 COLLATE Arabic_ci_ai;
```

Les fonctions

```
--ABS
select abs(-25)



--ASCII
select ascii('B')
select char(97)

--CAST //permet de faire la conversion
select cast(15.5 as varchar) + 1
select cast(15.5 as varchar) + cast(1
as varchar)
use librairie
select cast(prixvente as
int),prixvente from tarifer
--CEILING
select ceiling(15.9)
select ceiling(prixvente),prixvente
from tarifer
--floor
select floor(15.9)
select floor(prixvente),prixvente from
tarifer
--CHAR
select char(65)

--NCHAR
select nchar(65);

--CHARINDEX
select charindex('b','yaxcd')

--COALESCE // permet de recupérer la
première valeure non null
dans une collection
SELECT COALESCE(Null, 1, NULL, null,
3, NULL, 4);

--CONCAT // a partir de 2012
SELECT CONCAT('ismo', '.com');

--Concat with +
select 'ismo' + '.com'

select nomecr, prenomecr,
isnull(nomecr,'') + ' ' +
isnull(prenomecr,'') from ecrivain
select nomecr, prenomecr,
concat(nomecr , ' ' , prenomecr) from
ecrivain
```

```
--CONVERT
select cast('15' as int) + 1
select convert(int, '15') + 1
select
convert(decimal(18,3),15.545454);
select
convert(decimal(18,3),prixvente/3) as
nvPrix from tarifer
select
convert(decimal(18,2),prixvente) from
tarifer
select prixvente from tarifer
--CURRENT_TIMESTAMP
SELECT CURRENT_TIMESTAMP;

select getdate();

select GETUTCDATE();



--CURRENT_USER
select CURRENT_USER

--SESSION_USER
select SESSION_USER;

--SYSTEM_USER
select SYSTEM_USER

--USER_NAME
select USER_NAME(5)

--DATALENGTH
SELECT datalength('ismo.com');
select len('aze');

--DATEADD
SELECT DATEADD(year, 1, '28/07/2014');
SELECT DATEADD(month, 1,
'28/12/2014');
SELECT DATEADD(day, 80, '28/07/2014');

--DATEDIFF
SELECT DATEDIFF(year, '2012/03/28',
'2012/12/28');
SELECT DATEDIFF(month, '2012/03/28',
'2014/04/28');
SELECT DATEDIFF(day, '2012/03/28',
'2014/04/28');
--DATENAME
SELECT DATENAME(mm, '2014/04/28');
SELECT DATENAME(yy, '2014/04/28');
```

```sql
--DATEPART
SELECT DATEPART(yy, '2014/04/28');
SELECT DATEPART(mm, '2014/04/28');
SELECT DATEPART(dd, '2014/04/28');

--DAY
SELECT DAY('2014/04/28');

--MONTH
select MONTH('2014/04/28');

--YEAR
select YEAR('2014/04/28');

--ISDATE
select isdate('2017/18/18')
select isdate('18/04/2017')

--ISNULL

si la premiere valeur est null donne
moi la valeur de remplacement
sinon donne moi la premiere valeur

select isnull(15,'a')
select isnull(null,'a')

select isnull(teled,'pas de
telephone') from editeur
select nomecr + ' ' +  prenomecr from
ecrivain

select nomecr, prenomecr,
isnull(nomecr,'') + ' ' +
isnull(prenomecr,'') as nomComplet
from ecrivain

update ecrivain set prenomecr = null
where prenomecr = 'F.'
--ISNUMERIC
select ISNUMERIC('15')
select ISNUMERIC(ascii('a'))

--LEFT
SELECT LEFT('ismo.com', 4);

--RIGHT
SELECT RIGHT('ismo.com', 4);
```

```sql
--LOWER
select lower('ABCD');
select lower(nomecr),nomecr from
ecrivain

--UPPER
select UPPER('abcd');

--LTRIM
select '---'+ltrim('  ABCD  ') +'---'

--RTRIM
select '---'+rtrim('  ABCD  ') +'---'

select '---'+RTRIM(ltrim('    abcd
'))+'---';

--NULLIF  //return null si 'a' = 'b'
sinon il retourn 'a'
select NULLIF('aaa','bbbb');
select NULLIF('a','a');

--RAND // permet d'afficher une valeur
aléatoire
select ceiling( RAND()*100000)
select rand()*100000

--REPLACE
select replace('abcdefab','ab','yy');

--STUFF
SELECT STUFF('abcdefab', 2,4, 'yy');

--ROUND
select round(15.546721,2);

--SIGN
select sign(-15)
select sign(15)

--SPACE
select 'aa' + space(30) + 'bb'

--STR
select STR (15.5434,9,2)

select str(prixvente/3,18,2) from
tarifer

--SUBSTRING
select substring('abcd',2,1)
```