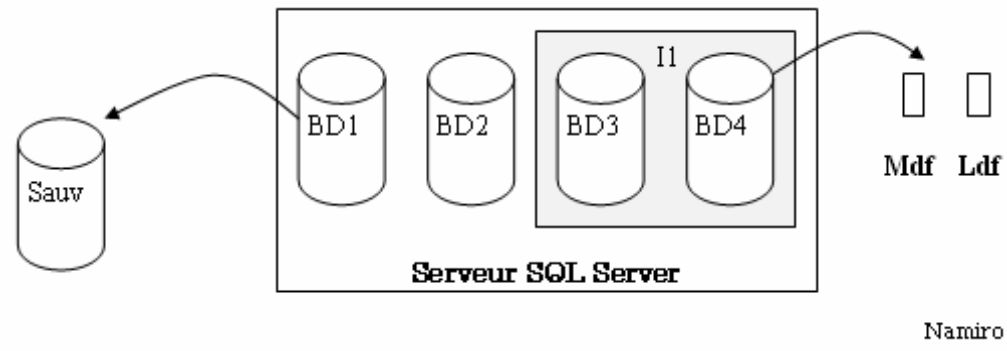
Commandes

SQL SERVER

Par la pratique



Réalisé par :

Structure des Bases de données :

CRÉER UNE BASE DE DONNÉE :

```
CREATE DATABASE BASE_1
```

➔ Pour supprimer une base de donnée (pas utilisée en cours) :

```
DROP DATABASE BASE_1
```

➔ Pour aller plus loin et paramétrer les différents fichiers :

```
CREATE DATABASE BASE_1
ON PRIMARY
( NAME = BASE_1_dat,
  FILENAME = 'C:\program files\Microsoft SQL Server\mssql\data\base_1dat.mdf',
  SIZE = 10,
  MAXSIZE = 50,
  FILEGROWTH = 5 %)
LOG ON
( NAME = BASE_1_log,
  FILENAME = 'C:\program files\Microsoft SQL Server\mssql\data\base_1log.ldf',
  SIZE = 5MB,
  MAXSIZE = 25MB,
  FILEGROWTH = 5MB )
```

➔ Pour utiliser une base de donnée :

```
USE BASE_1
```

MODIFIER LA STRUCTURE D'UNE BD :

➔ Pour ajouter un fichier supplémentaire :

```
ALTER DATABASE BASE_1
ADD FILE
(
 NAME = BASE_1dat2,
 FILENAME = 'C:\Program Files\Microsoft SQL Server\MSSQL\Data\t1dat2.ndf',
 SIZE = 5MB,
 MAXSIZE = 100MB,
```

FILEGROWTH = 5MB
)

➔ Supprimer un fichier de BD :

```
ALTER DATABASE BASE_1
REMOVE FILE BASE_1dat2
```

➔ Pour modifier l'un des fichiers d'une BD (Modification du Journal par exemple) :

```
ALTER DATABASE BASE_1
MODIFY file (Name=BASE_1_log, MAXSIZE = 80MB)
```

➔ Pour renommer une BD:

```
Sp_renamedb 'BASE_1', 'BASE_2'
```

ATTACHER / DÉTACHER UNE BD :

➔ Pour détacher une BD (pas en cours d'exécution) :

```
Sp_detach_db 'BASE_1'
```

➔ Pour attacher une BD (mdf + ldf) :

```
Sp_attach_db @dbname = 'BASE_1',
  @filename1 = 'C:\Program Files\Microsoft SQL Server\MSSQL\Data\base_1dat.mdf',
  @filename2 = 'C:\Program Files\Microsoft SQL Server\MSSQL\Data\base_1log.ldf'
```

➔ Pour attacher une base de données ne contenant qu'un seul fichier de données :

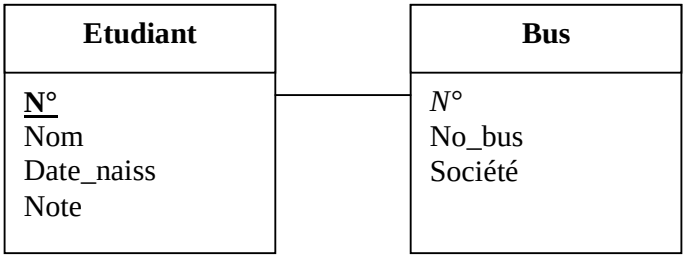
```
Sp_attach_single_file_db @dbname = BASE_1,
  @physname = 'C:\Program Files\Microsoft SQL Server\MSSQL\Data\base_1dat.mdf'
```

PROPRIÉTAIRE DE BD :

➔ Pour modifier le propriétaire de la base de données courante :

```
Sp_changedbowner 'Namiro'
```

Structure des Tables :



CRÉER UNE TABLE :

➔ Pour créer une table :

```
CREATE TABLE Etudiant (
        n° integer primary key,
        nom char(10),
        date_naiss date,
        note float) ;

CREATE TABLE Bus (
        n° integer,
        no_bus integer,
        société char(10)) ;
```

➔ Pour supprimer une table :

```
DROP TABLE Profession ;
```

TYPES DES CHAMPS (OBLIGATOIRE) :

int – char(x) – real – float – decimal(x,y) – logical – time – datetime – numeric – money – varchar – text(x) – longtext – longbinary – bit – image …

➔ Pour ajouter un type utilisateur :

```
Sp_addtype telephone, 'varchar(24)', 'NOT NULL'
```

PROPRIÉTÉS DES CHAMPS (FACULTATIFS) :

[not] null – unique – primary key – check(condition) – default – foreign key

CLÉ PRIMAIRE / SECONDAIRE :

➔ Pour Créer une table avec un clé primaire (un seul champ) :

```
CREATE TABLE Personnes (n° identity(1,1), nom char(12) default 'BTH',
        CONSTRAINT CP primary key (n°)) ;
```

➔ Pour Créer une table un clé primaire (2 champs ou plus) :

```
CREATE TABLE Profession (n° integer, code char(8), prof varchar,
        CONSTRAINT CP2 primary key (n°, code)) ;
```

➔ Pour créer une table avec un clé étrangère :

```
CREATE TABLE Bus (n° integer, code char(8), prof varchar,
        CONSTRAINT CE1 foreign key (n°) REFERENCES Etudiant (n°);
```

➔ Pour supprimer une contrainte (clé primaire, clé étrangère, condition, not null, unique) :

```
ALTER TABLE Bus
DROP CONSTRAINT CE1 ;
```

➔ Pour Ajouter un clé étrangère (Modification) :

```
ALTER TABLE Bus
ADD CONSTRAINT CE1 foreign key (n°) REFERENCES Etudiant (n°);
```

➔ Pour Créer une table avec une condition sur un champ :

```
CREATE TABLE Salles (n° integer, désignation char(15),
        CONSTRAINT C1 check (designation like 'Salle%');
```

MODIFIER LA STRUCTURE DE TABLE :

```
ALTER TABLE Etudiant
ADD adresse char(30), age integer ;                    « Ajout »

ALTER TABLE Etudiant
DROP nom, note ;                                       « Suppression »

ALTER TABLE Etudiant
MODIFY nom char(15) ;                                  « Modification propriétés »
```

ACTIVER / DÉSACTIVER UNE CONTRAINTE :

➔ Pour désactiver une contrainte :

ALTER TABLE BUS **NOCHECK** CONSTRAINT C1

➔ Pour réactiver une contrainte :

ALTER TABLE BUS **CHECK** CONSTRAINT C1

CRÉATION DES VUES :

➔ Pour créer une Vue à partir d'une table :

```
CREATE VIEW Etudiant_Admis ( n°, nom, date_naiss, note)
AS
SELECT(n°, nom, date_naiss, note)
FROM Etudiant
WHERE note>10 ;
```

➔ Pour supprimer une vue :

DROP VIEW Etudiant_Admis ;

CRÉATION DES INDEX :

➔ Pour créer un index sur 2 champs d'une table :

```
CREATE INDEX Index1
ON Etudiant (n°, nom);
```

➔ Pour créer un index en forçant l'unicité sur un champ :

```
CREATE UNIQUE CLUSTERED INDEX Index2
ON Bus (société);
```

➔ Pour supprimer un index :

DROP INDEX Index2

RENOMMER DES OBJETS :

➔ Pour renommer un objet (table, vue, procédure stocké, trigger, contrainte, règles …)

Sp_rename 'Etudiant', 'Stagiaires', '**Object**'

➔ Pour renommer une colonne d'une table :

Sp_rename 'Etudiant.nom', 'Nom_E', '**COLUMN**'

➔ Pour renommer un index :

Sp_rename 'Index1', 'Index01', '**INDEX**'

➔ Pour renommer un type utilisateur :

Sp_rename 'telephone', 'phone', '**USERDATATYPE**'

Définition des enregistrements dans les Tables :

AJOUTER DES ENREGISTREMENTS :

➔ 1ère méthode :

INSERT INTO Etudiant (n°, nom, note) **VALUES** (5, 'Mounir', 12) ;

➔ 2ème méthode :

INSERT INTO Etudiant **VALUES** (3, 'Amine', '15/08/1985', 13.25) ;

INSERT INTO Etudiant **VALUES** (7, 'Meriem', **null**, 14) ;

INSERT INTO Bus **VALUES** (5, 63, 'medina') ;

INSERT Bus **VALUES** (5, 20, 'raha') ;

➔ Pour insérer des enregistrements à partir d'une autre table :

```
INSERT INTO Etudiant (n°, nom)
SELECT n°, nom
FROM Personnes
WHERE nom like 'n%' ;
```

MISE À JOUR DES ENREGISTREMENTS :

```
UPDATE Etudiant
SET n°=6, nom='Namiro' WHERE n°=5 ;

UPDATE Etudiant
SET note=13.26 WHERE n°=6 ;

UPDATE Bus
SET société='medina' WHERE no_bus=70 ;

UPDATE  Articles
SET prix = prix * 0.25
```

SUPPRIMER DES ENREGISTREMENTS :

DELETE FROM Etudiant **WHERE** n°=6 « Suppression des enregistrements précis »

DELETE FROM Bus ; « Suppression de toutes les enregistrements »

Requêtes (Select) :

SÉLECTIONNER DES ENREGISTREMENTS (RÈGLE GÉNÉRALE):

SELECT n°, nom
FROM Etudiant
WHERE note>10 « Condition »
GROUP BY nom « Regroupement par »
HAVING nom like ‘ _J%’ « like remplace « = » dans ce cas »
ORDER BY note « Tri »
UNION / **IN** / **NOT IN**
SELECT ……

LES FONCTIONS D’AGRÉGATION (D’ENSEMBLE) :

SUM ( )	Somme
AVG ( )	Moyenne
MAX ( )	Maximum
MIN ( )	Minimum
COUNT ( )	Nombre d’enregistrements

SELECT nom, **AVG** (note) **FROM** Etudiant ;

SELECT nom, **AVG** (note **AS** Note_Moyenne) **FROM** Etudiant ;

SELECT nom **FROM** Etudiant **WHERE** note **IN** (SELECT **MAX** (note) FROM Etudiant) ;

LA CLAUSE LIKE :

Il remplace le signe « = » dans le cas dont on recherche des enregistrements a l’aide des signes _ et **%**

LA CLAUSE BETWEEN

SELECT * **FROM** Etudiant **WHERE** note **BETWEEN** 8 **AND**15 ;
SELECT * **FROM** Etudiant **WHERE** note **NOT BTWEEN** 5 **AND** 10 ;

LA CLAUSE IN / NOT IN :

--**SELECT** nom **FROM** Etudiant WHERE note **IN** (‘10’,’20’) ;

--**SELECT** nom **FROM** Etudiant WHERE note=10 OR note=20 ;

SELECT no_bus **FROM** Bus WHERE société **NOT IN** (‘medina’) ;

DISTINCT :

SELECT n°, count (**DISTINCT** note)
FROM Etudiant
GROUP BY n° DESC;

JOINTURE :

➔ Utilisation de la méthode ensembliste :

SELECT nom, no_bus
FROM Etudiant
WHERE n° **IN** (SELECT n° FROM **Bus** WHERE société=’casa’) ;

➔ Utilisation de la méthode prédicative :

Ancienne :

SELECT Etudiant.nom, Bus.no_bus
FROM Etudiant, **Bus**
WHERE Etudiant.n° = Bus.n°
AND Bus.société = ‘casa’ ;

Actuelle :

SELECT Etudiant.nom, Bus.no_bus
FROM Etudiant INNER JOIN Bus **ON Etudiant**.n° = Bus.n°
WHERE Bus.société = ‘casa’ ;

AUTO JOINTURE :

--Afficher les noms des personnes ayant la même note.

```
SELECT Etudiant.nom, Etudiant2.note
FROM Etudiant INNER JOIN Etudiant AS Etudiant2 ON Etudiant.note = Etudiant2.note
WHERE Etudiant.n°<> Etudiant2.n° ;
```

REQUÊTES PLUS PRÉCISES :

LIKE '%[M-R]' ➔ Finissent par M, N, O, P, Q ou R

LIKE '%[FMR]%' ➔ ne contenant ni F ni M ni R

TOP 10 ➔ Affiche les 10 premières lignes du résultat.

Fonctions SQL Server :

FONCTIONS DATE & HEURE :

➔ Pour ajouter un nombre à une partie de la date :

DATEADD (Partie_Date, nombre, colonne)
DATEADD (month, 5, datenaiss)

➔ Pour soustrait la date de fin de celle de départ :

DATEDIFF (Partie_Date, Date_départ, Date_fin)
DATEDIFF (year, datenaiss, Gatedate())

➔ Pour retourner une chaîne de caractères d'une partie de la date :

DATENAME (Partie_Date, colonne)
DATENAME (month, datenaiss)

➔ Pour retourner une partie d'une date :

DATEPART (Partie_Date, colonne)
DATEPART (day, colonne)

Ou utiliser :

Day (colonne)
Month (colonne)
Year (colonne)

Formats (Partie_Date) :

dddd, mmmm dd, yyyy hh:mm:ss.ffff tt
Friday, December 28, 2001 11:16:41.0021 AM

FONCTIONS CHAÎNES :

➔ Pour concaténer deux chaînes ou plus :

CONCAT (chaine1, chaine2)
CONCAT (Nom, Prénom)

➔ Pour extraire une partie du chaîne :

SUBSTR (colonne, Position_départ, longueur)
SUBSTR (Nom, 1, 1)

➔ Pour remplacer une valeur dans une chaîne par une autre valeur :

REPLACE (colonne, valeur1, valeur2)
REPLACE (Nom, 'Mb', 'Mn')

➔ Pour convertir une chaîne en majuscule / minuscule :

UPPER (chaîne) ou **LOWER** (chaîne)
UPPER (Nom) ou **LOWER** (Prénom)

➔ Pour retourner la longueur de la chaîne de caractères :

LENGTH (chaîne)
LENGTH (Nom)

➔ Pour retourner la valeur ASCII d'un jeu de caractères indiqué :

ASCII (jeu_caractères)
ASCII (Nom)

FONCTIONS MATHÉMATIQUES :

➔ Pour retourner la valeur absolue d'une expression :

ABS (Expression)
ABS (Remise)

➔ Pour retourner la puissance de l'expression ($Expression^{Exposant}$) :

POWER (Expression, Exposant)
POWER (Prix, 2)

➔ Pour retourner la racine carrée de l'expression :

SQRT (Expression)
SQRT (Remise)

Procédures stockés & Triggers :

PROCÉDURES STOCKÉES :

➔ Pour créer une procédure stockée :

```
CREATE PROCEDURE [nom]
AS
        Instructions
```

```
CREATE PROCEDURE Proc1
AS
SELECT * FROM Etudiant
```

➔ Procédure stockée avec des variables :

```
CREATE PROCEDURE proc2
@date_debut datetime, @date_fin datetime
AS
SELECT *
FROM Etudiant
WHERE datenaiss BETWEEN @date_debut AND @date_fin
```

➔ Pour executer la procedure stockée :

```
DECLARE @date_debut datetime          ← Declaration
DECLARE @date_fin datetime

SET @date_debut='1/1/1997'            ← Affectation
SET @date_fin='1/20/1997'

EXECUTE proc2 @date_debut, @date_fin  ← Execution
```

➔ Pour modifier une procédure stockée :

```
ALTER PROC proc1
AS
SELECT *
FROM BUS
```

➔ Pour supprimer une procédure stockée :

```
DROP PROC proc1
```

TRIGGERS (EXEMPLES) :

Commandes
N_Com N_Art Com_qt

Articles
N_Art **Nb_Com** Type

➔ Déclencheur qui affiche un message lors de l'insertion d'un enregistrement :

```
CREATE TRIGGER Tr1
ON Commandes AFTER INSERT
AS
PRINT ('Une Commande est insérée')
```

➔ Déclencheur qui mettre à jour un champ sur une table après Insertion :

```
CREATE TRIGGER Tr2
ON Commandes AFTER INSERT
AS
UPDATE Articles SET Nb_Com = Nb_Com + Com_qt
FROM Articles A, Inserted I
WHERE A.N_Art = I.N_Com
```

➔ Déclencheur qui mettre à jour un champ sur une table après Suppression:

```
CREATE TRIGGER Tr3
ON Commandes AFTER DELETE
AS
UPDATE Articles SET Nb_Com = Nb_Com - Com_qt
FROM Articles A, Deleted D
WHERE A.N_Art = D.N_Com
```

➔ Déclencheur qui mettre à jour un champ sur une table après une Mise à jour :

```
CREATE TRIGGER Tr4
ON Commandes AFTER UPDATE
AS
UPDATE Articles SET Nb_Com = Nb_Com + I.Com_qt – D.Com_qt
FROM Articles A, Deleted D, Inserted I
WHERE A.N_Art = I.N_Com
AND A.N_Art = D.N_Com
```

➔ Déclencheur qui empêche la modification du champ Type de la table Articles :

```
CREATE TRIGGER Tr5
ON Articles AFTER UPDATE
AS
IF UPDATE (Type)
BEGIN
RAISERROR ('Le Type ne peut pas être modifié', 0, 1)
ROLLBACK TRANSACTION
END
```

➔ Déclencheur INSTEAD OF (Au lieu de) :

```
CREATE TRIGGER Tr6
ON Commandes INSTEAD OF INSERT
AS
INSERT Historique_Com SELECT * FROM Inserted
```

➔ Pour modifier un trigger :

```
ALTER TRIGGER Tr1
ON Commandes AFTER INSERT
AS
PRINT ('Une Commande est insérée sur la table Commandes')
```

➔ Pour supprimer un trigger :

```
DROP TRIGGER Tr2
```

➔ Pour suspendre provisoirement un déclencheur (sans le supprimer) :

```
ALTER TABLE Articles
DISABLE TRIGGER Tr5
```

Informations supplémentaires :

COMMENTAIRES :

➔ Pour ajouter un commentaire sur une seule ligne :

-- c'est un commentaire

➔ Pour ajouter un commentaire de plusieurs lignes :

/* c'est
Un
Commentaire */

TRANSACTIONS, BLOCS ET LOTS:

➔ Pour délimiter une transaction, ajouter :

BEGIN TRANSACTION
…
COMMIT TRANSACTION

➔ Pour délimiter un bloc d'instructions :

BEGIN
…
END

➔ Pour exécuter un lot par lot :

…
GO
…

Comptes & Rôles :

Créer une connexion :

➔ Pour créer une connexion SQL de toute pièce :

Sp_addlogin 'Nom_Connexion', 'MDP', 'BD_par_défault'
Sp_addlogin 'Namiro', '123456', 'Bae_1'

➔ Pour créer une connexion hérité de Windows :

Sp_grantlogin 'Domaine\Compte_Util'
Sp_defaultdb 'Domaine\Compte_Util', 'BD_par_défault'
Sp_grantlogin 'ISTA.com\Namiro'
Sp_defaultdb 'ISTA.com\Namiro', 'Base_1'

➔ Pour supprimer une connexion SQL :

Sp_droplogin 'Nom_Connexion'
Sp_droplogin 'Namiro'

➔ Pour empêcher un Utilisateur Windows d'accéder au serveur :

Sp_denylogin 'Domaine\Compte_Util'
Sp_denylogin 'ISTA.com\Namiro'

Donner accès à une connexion :

➔ Pour donner une connexion accès à une base de donnée (en cours obligatoire) :

Sp_grantdbaccess 'Nom_Connexion'
Sp_grantdbaccess 'Namiro'

➔ Pour restreindre l'accès à une base de donnée :

Sp_revokedbaccess 'Nom_Connexion'
Sp_revokedbaccess 'Namiro'

➔ On peut également créer tout simplement un Utilisateur :

Sp_adduser 'Nom_Connexion', 'Nom_Util', 'Nom_Rôle'
Sp_adduser 'Namiro', 'Mounir', 'Role1'

➔ Pour autoriser les connexions sans compte d'utilisateur associé à accéder à BD :

Sp_grantdbaccess '**guest**'

CRÉER UN RÔLE STANDARD :

➔ Pour créer un rôle standard :

Sp_addrole 'Nom_Rôle'
Sp_addrole 'Réseau'

➔ Pour supprimer un rôle :

Sp_droprole 'Nom_Rôle'
Sp_droprole 'Réseau'

AFFECTER DES UTILISATEURS À UN RÔLE :

➔ Pour affecter un compte de connexion à un rôle de base de donnée (fixe ou personnalisé):

Sp_addrolemember 'Nom_Rôle', 'Nom_Connexion'
Sp_addrolemember 'Réseau', 'Namiro'

➔ Pour affecter un compte de connexion à un rôle fixe serveur :

Sp_addsrvrolemember 'Nom_Connexion', 'Nom_Rôle_Serveur'
Sp_addsrvrolemember 'Namiro', 'sysadmins'

➔ Pour retirer un compte de connexion à un rôle de base de donnée (fixe ou personnalisé):

Sp_droprolemember 'Nom_Rôle', 'Nom_Connexion'
Sp_droprolemember 'Réseau', 'Namiro'

➔ Pour affecter un compte de connexion à un rôle fixe serveur :

Sp_dropsrvrolemember 'Nom_Connexion', 'Nom_Rôle_Serveur'
Sp_dropsrvrolemember 'Namiro', 'sysadmins'

CRÉER UN RÔLE D'APPLICATION :

➔ Pour créer un rôle d'application (Sécurité) :

Sp_addapprole 'Nom_Rôle_App', 'MDP'
Sp_addapprole 'App1', '123456'

➔ Pour Activer les autorisations associées à un rôle d'application dans la BD courante :

Sp_setapprole 'Nom_Rôle_App', 'MDP'
Sp_addapprole 'App1', '123456'

➔ Pour Activer les autorisations associées en envoyant le MDP crypté :

Sp_setapprole 'Nom_Rôle_App', {Encrypt N 'MDP'}, 'type de cryptage'
Sp_addapprole 'App1', {Encrypt N '123456}, 'odbc'

➔ Pour supprimer un rôle d'application :

Sp_dropapprole 'Nom_Rôle_App'
Sp_dropapprole App1

Autorisations SQL :

AUTORISATIONS SUR DES OBJETS :

➔ Pour autoriser la sélection sur une table (Etudiant) :

Grant Select **ON** Etudiant **TO** Namiro

➔ Pour empêcher les autres instructions à 2 utilisateurs:

Deny Insert, Update, Delete **ON** Etudiant **TO** Namiro, [ISTA.com\Namiro]

➔ Pour autoriser la modification mais seulement sur un champ (nom de l'étudiant) :

Grant Update (nom) **ON** Etudiant **TO** Namiro

➔ Pour autoriser un rôle d'insérer sur une table (Etudiant) :

Grant Insert **ON** Etudiant **TO** Réseau

Les objets sont :

Select, *Insert*, *Update*, *Delete*, *References* sur une table ou vue.
Select, Update, References sur une colonne.
Exec sur une procédure stocké.

AUTORISATIONS SUR DES INSTRUCTIONS :

➔ Pour autoriser la création des tables et des vues :

Grant Create table, Create view **TO** Namiro

➔ Pour empêcher la création des bases de données :

Deny Create database **TO** Namiro

➔ Pour supprimer (Lever les autorisations et les empêchements) :

REVOKE Create database, Create view **TO** Namiro

➔ Pour autoriser toutes les instructions :

GRANT ALL TO Namiro

Les instructions sont :

CREATE *DATABASE*, CREATE *TABLE*, CREATE *VIEW*, CREATE *PROCEDURE*, CREATE *RULE*, CREATE *DEFAULT*, CREATE *FUNCTION*, BACKUP DATABASE, BACKUP LOG.

AUTORITÉ POUR DONNER DES AUTORISATIONS :

➔ Pour autoriser Namiro à faire la sélection sur la table Etudiant, ainsi de la possibilité de la donner à d'autre utilisateurs :

GRANT SELECT **ON** Etudiant **TO** Réseau
WITH GRANT OPTION

➔ Le rôle Réseau à l'autorité de faire Select sur la table Etudiant, Namiro est un membre du rôle Réseau, Pour que Namiro puisse donner des autorisations Select sur Etudiant :

GRANT SELECT **ON** Etudiant **TO** Util1 **AS** Réseau

Sauvegarde & Restauration :

CRÉATION D'UNITÉS DE SAUVEGARDE :

➔ Pour créer une unité de sauvegarde nommé US1 sur un disque :

Sp_addumpdevice '**DISK**', 'US1', '\\servername\sharename\path\filename.**bak**'

➔ Pour créer une unité sur un emplacement physique d'une bande magnétique:

Sp_addumpdevice '**TAPE**', 'US2', '\\.\tape0'

SAUVEGARDER UNE BD :

➔ Pour sauvegarder une BD sans unité de sauvegarde permanente :

BACKUP DATABASE Bse_1 **TO DISK** = 'C:\Temp\Mycustomers.**bak**'

➔ Pour sauvegarder une BD sur 2 unités de sauvegarde en même temps :

BACKUP DATABASE Bse_1 **TO** US1, US2 **WITH MEDIANAME** = US

➔ Pour effectuer une sauvegarde complète d'une BD sur une unité avec description :

BACKUP DATABASE Base_1 **TO** US1 **WITH DESCRIPTION** = 'Première sauvegarde'

➔ Pour effectuer une sauvegarde complète d'une BD sur une unité en écrasant le précédent :

BACKUP DATABASE Base_1 **TO** US1 **WITH INIT**

➔ Pour effectuer une sauvegarde différentielle d'une BD sur une unité :

BACKUP DATABASE Base_1 **TO** US1 **WITH DIFFERENTIAL**

➔ Pour sauvegarder le journal des transactions :

BACKUP LOG Base_1 **TO** US2

➔ Pour supprimer la partie inactive d'un journal sans faire de copie de sauvegarde :

BACKUP LOG Base_1 **WITH TRUNCATE_ONLY**

➔ Pour supprimer la partie inactive d'un journal plein sans en faire de copie de sauvegarde.

BACKUP LOG Base_1 **WITH NO_LOG**

➔ Pour sauvegarder un fichier ou un groupe de fichier :

BACKUP DATABASE Base_1
FILE = Base_1dat **TO** US1
BACKUP LOG Base_1 to US2

VÉRIFIER UNE SAUVEGARDE AVANT RESTAURATION :

➔ Pour obtenir les informations d'en-tête d'un fichier de sauvegarde particulier :

RESTORE HEADERONLY

➔ Pour obtenir des informations sur les fichiers de base de données ou de journal :

RESTORE FILELISTONLY

➔ Pour obtenir des informations sur le support de sauvegarde contenant un fichier :

RESTORE LABELONLY

➔ Pour assurer que les différents fichiers constituant le jeu de sauvegardes sont complets :

RESTORE VERIFYONLY

RESTAURER UNE BD :

➔ Pour restaurer une BD à partir d'une unité de sauvegarde :

USE master
RESTORE DATABASE Base_1
FROM US1

➔ Pour restaurer une BD à partir de plusieurs unités de sauvegarde (2 par exemple) :

USE master
RESTORE DATABASE Base_1
FROM US1
WITH NORECOVERY ← pas de validation

```
RESTORE DATABASE Base_1
FROM US2                          ← validation (WITH RECOVERY par default)
```

➔ Pour restaurer une sauvegarde spécifique (l'un des fichiers existant sur une unité) :

```
USE master
RESTORE DATABASE Base_1
FROM US1
WITH FILE = 2, RECOVERY        ← Restaurer la deuxième sauvegarde.
```

➔ Pour restaure tous les enregistrements du journal des transactions écrits dans la base de données avant un certain moment défini :

```
USE master
RESTORE DATABASE Base_1
FROM US1
WITH NORECOVERY

RESTORE LOG Base_1
FROM US2
WITH FILE = 1, NORECOVERY

RESTORE LOG Base_1
FROM US2
WITH FILE = 2, RECOVERY, STOPAT = '3 janvier 2000, 01:00'
```