

# 5. Exécution de tâches administratives

## 5.1 Tâches liées à la configuration

5.2 Tâches de routine liées à l'administration de SQL Server

5.3 Automatisation des tâches de maintenance de routine

5.4 Création d'alertes

5.5 Résolution des problèmes liés à l'automatisation de SQL Server

5.6 Automatisation de travaux sur plusieurs serveurs

# 5.1 Tâches liées à la configuration

- Configuration de l'Agent SQL Server
- Configuration de SQLAgentMail et de SQLMail
- Configuration de serveurs liés
- Configuration des noms de sources de données
- Configuration de la prise en charge du langage XML de SQL Server dans les services Internet
- Configuration de SQL Server pour partager les ressources de mémoire avec les autres applications serveur

# Configuration de l'Agent SQL Server

- SQL Server Agent : **chargé** d'automatiser les **tâches** administratives de SQL Server
- **Doit être exécuté en permanence** et disposer des autorisations adéquates
- Doit être **configuré** pour **démarrer automatiquement** à chaque démarrage de Windows 2000
- Peut être configuré pour **redémarrer automatiquement** en cas d'arrêt inattendu
- Pour redémarrer automatiquement, son compte doit être membre du groupe **Administrateurs** local

# Configuration de SQLAgentMail et SQL Mail

- SQL Server peut envoyer et recevoir du **courrier électronique** en établissant une connexion cliente avec un serveur de messagerie
- Il envoie et reçoit du courrier électronique à l'aide d'une session **SQLAgentMail** ou **SQL Mail**
- SQLAgentMail nécessite que le service SQL Server Agent utilise un compte d'utilisateur de domaine avec un **profile de messagerie** (à indiquer avec SQL Server Enterprise Manager)

# Configuration de SQLAgentMail et SQL Mail

- SQL Mail contient des **procédure stockées pour traiter les messages électroniques** entrants et renvoyer des résultats
- Permet d'envoyer des messages électroniques à partir de l'application de BD en exécutant par exemple la procédure stockée **xp\_sendmail** à partir d'un déclencheur

# Configuration : Etapes

- Utiliser un serveur de messagerie **compatible** MAPI-1 (*Messaging Application Programming Interface*)
- **Configurer** un client de messagerie sur l'ordinateur qui exécute SQL Server
- Utiliser un compte de **connexion d'utilisateur de domaine** pour les services SQL Server et SQL Server Agent
- Configurer un **profil de messagerie** pour chaque compte d'utilisateur de domaine en vue d'une connexion au serveur de messagerie
- Utiliser SQL Server Enterprise Manager pour **spécifier un profil de messagerie** pour chaque session SQL Mail et SQLAgentMail

# Configuration de serveurs liés

- Permet à SQL Server d'exécuter des commandes sur des **sources de données** OLE DB sur **différents serveurs**
- Inscrire **les informations sur la connexion et la source de données** OLE DB auprès de SQL Server
- La source de donnée peut toujours être **référéncée** par un seul **nom logique**
- Créer la définition d'un serveur lié par **sp\_addlinkedserver** ou sur SQL Server Enterprise Manager

# Etablissement de la sécurité des serveurs liés

- Lors d'une connexion à un serveur lié, le serveur émetteur fournit **un nom et un mot de passe** d'ouverture de session pour se connecter au serveur récepteur
- On crée des **mappages de connexion** entre les serveurs liés par **sp\_addlinkedsvlogin** ou sous SQL Server Enterprise Manager



# Configuration des noms de sources de données

Un nom de source de données définit:

- Le **pilote ODBC** à utiliser,
- les informations de connexion (nom et emplacement de la source de données, compte de connexion et mot de passe)
- Options propres au pilote pour la connexion

# Configuration de la prise en charge du langage XML de SQL Server dans les services Internet

- On peut **configurer** les services Internet (**IIS**) pour permettre **l'accès** à SQL Server
- On peut **accéder** à SQL Server par le biais du protocole HTTP à l'aide d'une **URL**

# Configuration de la prise en charge du langage XML de SQL Server dans les services Internet

L'URL effectue les tâches suivantes :

- accède directement aux objets de BD comme les tables
- exécute les fichiers modèles (Fichiers XML comprenant des instructions T-SQL)
- exécute des requêtes XPath (XML Path Language).

# Configuration d'un annuaire virtuel dans les services Internet

- Pour accéder à une BD SQL Server par le protocole HTTP, on doit configurer un annuaire virtuel approprié
- Utiliser l'utilitaire **IIS Virtual Directory Management** pour SQL Server pour définir et inscrire un nouveau répertoire virtuel appelé également **racine virtuelle** sur l'ordinateur exécutant les services Internet
- Cet utilitaire ordonne aux services Internet de créer une **association** entre le nouvel annuaire et une instance de SQL Server

# Accès du langage XML à SQL Server

Opérations se produisant dans un échange XML :

- Le **nom du serveur** exécutant les services Internet est **indiqué à l'URL**
- Le serveur exécutant les services Internet **examine la racine virtuelle** indiquée dans l'URL et détermine si **l'extension de nom de fichier DLL** (Sqlisapi.dll) ISAPI (Internet Server API) **a été inscrite** pour cette racine virtuelle

# Accès du langage XML à SQL Server

Suite des opérations se produisant dans un échange XML :

- Le serveur exécutant les services Internet **charge la bibliothèque DLL** et lui **transmet la demande d'URL**
- L'extension de nom de fichier **Sqlisapi.dll** **communique avec le fournisseur OLE DB** pour SQL Server et **établit une connexion avec l'instance de SQL Server** identifiée dans la racine virtuelle

# Configuration de SQL Server pour partager les ressources de mémoire avec les autres applications serveur

- Configuration des options de mémoire :
  - **min server memory** : définit un niveau sous lequel SQL Server ne libère pas de mémoire
  - **max server memory** : empêche SQL Server d'utiliser plus de mémoire que la quantité spécifiée

# 5. Exécution de tâches administratives

5.1 Tâches liées à la configuration

**5.2 Tâches de routine liées à l'administration de SQL Server**

5.3 Automatisation des tâches de maintenance de routine

5.4 Création d'alertes

5.5 Résolution des problèmes liés à l'automatisation de SQL Server

5.6 Automatisation de travaux sur plusieurs serveurs



## 5.2 Tâches de routine liées à l'administration de SQL Server

- Exécution de tâches planifiées régulièrement
  - Sauvegarde de BD
  - Importation et exportation de données
- Identification et résolution de problèmes potentiels
  - Surveillance de l'espace des BD et des journaux
  - Surveillance des performances

# 5. Exécution de tâches administratives

5.1 Tâches liées à la configuration

5.2 Tâches de routine liées à l'administration de SQL Server

**5.3 Automatisation des tâches de maintenance de routine**

5.4 Création d'alertes

5.5 Résolution des problèmes liés à l'automatisation de SQL Server

5.6 Automatisation de travaux sur plusieurs serveurs

## 5.3 Automatisation des tâches de maintenance de routine

- Automatisation de l'administration de SQL Server
- Création de travaux
- Vérification des autorisations
- Définition des étapes d'un travail
- Création d'un organigramme des actions par étape de travail
- Planification de travaux
- Création d'opérateurs à notifier
- Analyse et configuration de l'historique des travaux

# Automatisation de l'administration de SQL Server

- Services utilisés pour automatiser SQL Server :
  - Agent SQL Server, SQL Server et Observateur d'événement Windows 2000
- Composants de l'Agent SQL Server :
  - Les alertes, les travaux et les opérateurs
- Association de travaux et d'alertes

**Exemple** : si un travail échoue à cause d'une erreur système, une **alerte** définie pour répondre à ce numéro d'erreur peut **exécuter un autre travail** destiné à résoudre ce problème

# Création de travaux

- Sous SQL Server Entreprise Manager ou **sp\_add\_job**
- La définition du travail est stockée dans **msdb..sysjobs**
- Quand on définit un travail, on doit :
  - **Vérifier** qu'il est activé (par défaut oui)
  - Indiquer le **propriétaire** chargé de l'exécution du travail (par défaut compte d'ouverture de session)
  - Déterminer **l'emplacement de l'exécution** du travail (serveur local ou serveurs distants)
  - **Créer des catégories** de travail (pour aider à organiser, filtrer et gérer)

# Vérification des autorisations

- Si un compte user non **sysadmin** possède un travail ➔ vérifier que le propriétaire du travail dispose des autorisations appropriées pour l'exécuter
- **Exécution des travaux Transact-SQL :**
  - Tous les utilisateurs peuvent démarrer des travaux T-SQL qui opèrent dans le cadre de la sécurité du propriétaire du travail ou de l'utilisateur indiqué

# Vérification des autorisations (suite)

## ➤ Exécution de commandes du SE ou de travaux de script ActiveX :

- Les étapes qui appartiennent aux utilisateurs membres du rôle **sysadmin** sont exécutées dans le cadre de la sécurité du compte de connexion du service SQL Server.
- Si un compte user non **sysadmin** possède un travail, l'Agent SQL Server doit vérifier que son propriétaire dispose des autorisations appropriées pour l'exécuter
- **Par défaut**, les utilisateurs non **sysadmin** ne sont pas autorisés à exécuter les commandes du SE ni les travaux de script ActiveX

# Vérification des autorisations (suite)

- Un administrateur peut autoriser les utilisateurs qui ne sont pas membre du rôle **sysadmin** à exécuter les commandes du SE ou les travaux de script ActiveX
- Le cas échéant, les étapes de travail sont exécutées dans le cadre de la sécurité d'un compte d'utilisateur de domaine défini appelé **compte proxy**
- On peut définir ce compte proxy dans SQL Server Entreprise Manager ou par la procédure stockée étendue **xp\_sqlagent\_proxy\_account**



# Définition des étapes d'un travail

- Sous SQL Server Enterprise Manager ou par **sp\_add\_jobstep**
- Les définitions des étapes d'un travail sont stockées dans la table système **msdb..sysjobsteps**
- On peut définir les étapes d'un travail pour exécuter des instructions T-SQL, des commandes du SE, des scripts ActiveX ou des tâches de réplication SQL Server (**un seul type d'exécution par étape de travail**)

# Utilisation d'instructions T-SQL

- On doit identifier la BD à utiliser
- On doit inclure les variables et les paramètres requis dans l'étape de travail
- On doit envoyer l'ensemble des résultats d'une étape de travail vers un fichier de sortie (capturer les messages d'erreur)
- Le fichier de sortie d'une étape de travail ne peut servir d'entrée dans l'étape suivante

# Utilisation de commandes du SE (Fichiers .exe, .bat, .cmd ou .com)

- On doit identifier un code de sortie du processus pour indiquer que la commande a réussi
- On doit indiquer le chemin d'accès complet de l'application exécutable dans la zone de texte

## **Commande**

- Le chemin permet à l'Agent SQL Server de rechercher l'application source

# Utilisation de scripts ActiveX

- On doit identifier le langage de script dans lequel est écrite l'étape du travail
- Ecrire ou ouvrir le script actif
- On peut utiliser l'objet SQLActiveScriptHost pour imprimer la sortie vers l'historique des étapes de travail ou créer des objets
- On peut aussi compiler de façon externe des scripts ActiveX (à l'aide par exemple de VB) puis les exécuter en tant que commandes du SE

# Utilisation de la réplication

- les processus de réplication sont appelés

Agents

- Ils sont implémentés sous forme de travaux de l'Agent SQL Server

# Création d'un organigramme des actions par étape de travail

- Lors de la création de travaux, un administrateur de BD peut être amené à spécifier **l'action appropriée** que doit exécuter SQL Server en cas **d'échec** de l'exécution de travail
- **Par défaut**, SQL Server **passe à l'étape suivante** du travail en cas de réussite d'une étape et s'arrête en cas d'échec
- Les étapes de travail peuvent renvoyer à une étape quelconque définie dans le travail en cas de réussite ou d'échec
- On peut indiquer le **nombre de tentatives** en cas d'échec
- On peut également définir des **intervalles de reprise** en mn

# Planification des travaux

- Sous SQL Server Entreprise Manager ou par **sp\_add\_jobschedule**
- Les planifications des travaux sont stockées dans la table système **msdb..sysjobschedules**
- Les travaux **s'exécutent** conformément aux **planifications** définies ou en réponse à des alertes
- Si on travaille dans un environnement multiserveurs, on peut définir le travail à s'exécuter sur plusieurs serveurs cibles
- Un travail est exécuté conformément à la **planification** uniquement lorsque cette dernière est **activée**

# Planification des travaux (suite)

- On peut planifier le **démarrage automatique** des travaux dans les situations suivantes :
  - lorsque **l'Agent SQL Server est démarré**
  - **A une heure spécifique** (une seule fois)
  - **régulièrement** (tous les jours, toutes les semaines ou tous les mois)
  - lorsque le **processeur est inactif**
- Le compte d'utilisateur de domaine utilisé par le service SQL Server doit être **membre du groupe Administrateurs local** Windows pour exécuter un travail lorsque le processeur est inactif



# Planifications multiples

- **Plusieurs planifications** peuvent être définies **par travail**
- **Par exemple**, on peut planifier un travail qui sauvegarde le journal des transactions d'une BD du Lundi au Vendredi, toutes les 2 heures pendant les heures de pointes (de 08:00 à 17:00), puis établir une autre planification qui exécute le travail toutes les 4 heures hors des heures de pointe

# Création d'opérateurs à notifier

- Sous SQL Server Entreprise Manager ou par **sp\_add\_operator**
- La définition de l'opérateur est stockée dans la table système **msdb..sysoperators**
- Une fois le travail exécuté ou en cas d'échec d'une de ses étapes, on peut écrire l'événement dans le journal d'applications de Windows, supprimer le travail ou **notifier** un opérateur par radiomessagerie, courrier électronique ou une commande **net send**

# Tâches à effectuer lors de la création d'opérateurs

- Utiliser un **alias de messagerie** de groupe pour demander à plusieurs personnes de résoudre des problèmes potentiels
- **tester chaque méthode de notification** utilisée pour s'assurer que l'opérateur peut recevoir le message
- établir une **planification de travail** pour chaque opérateur à notifier par radiomessagerie
- Utiliser une commande **net send** pour envoyer des messages aux opérateurs et serveurs du réseau qui exécutent Windows 2000 ou Windows NT

# Résolution des problèmes liés aux notifications d'opérateur

Si un opérateur ne reçoit pas de notifications, on doit:

- **vérifier** que l'**opérateur** est **disponible** pour en recevoir
- **s'assurer** que le **service d’Affichage** des messages s'exécute sur l'ordinateur de l'opérateur à notifier par une commande **net send**
- **consulter les dernières tentatives de notification** pour connaître la date et l'heure auxquelles la dernière notification s'est produite
- **tester des méthodes de notification individuelles** en dehors de SQL Server en vérifiant qu'on peut envoyer des messages électroniques, avertir un opérateur par radiomessagerie ou exécuter une commande **net send**

# Analyse et configuration de l'historique des travaux

- l'Agent SQL Server capture l'état d'exécution des étapes du travail et stocke les informations dans la table système msdb..sysjobhistory
- On peut afficher l'historique des travaux dans SQL Server Enterprise Manager

# Analyse de l'historique des travaux

- Si un travail échoue, on peut afficher le **journal d'historique des travaux** pour obtenir des informations sur chaque étape du travail, la raison de son échec et les solutions permettant de résoudre le problème
- **L'historique** des travaux enregistre les éléments suivants:
  - La **date et l'heure** auxquelles l'étape du travail a eu lieu
  - **L'échec ou la réussite** de l'étape du travail
  - **L'opérateur notifié et la méthode de notification**
  - La **durée de l'étape** du travail
  - Les **erreurs ou messages**

# Configuration de la taille de l'historique des travaux

- Pour conserver des informations sur chaque travail, on doit augmenter la taille maximale de la table système **sysjobhistory**
- L'historique des travaux est automatiquement écrasé lorsque la taille maximale des lignes est atteinte
- Par défaut, l'extension automatique est définie pour les propriétés des fichiers de la BD **msdb**
- Par défaut, l'option de BD **Vider le journal au point de contrôle** est activée
- Par défaut, la taille maximale de l'historique des travaux est de 1000 lignes
- Par défaut, la taille maximale de l'historique des travaux par travail est de 100 lignes
- Lorsque les **limites sont atteintes**, les lignes sont **supprimées** de la table système **sysjobhistory** selon le mode FIFO

# 5. Exécution de tâches administratives

5.1 Tâches liées à la configuration

5.2 Tâches de routine liées à l'administration de SQL Server

5.3 Automatisation des tâches de maintenance de routine

**5.4 Création d'alertes**

5.5 Résolution des problèmes liés à l'automatisation de SQL Server

5.6 Automatisation de travaux sur plusieurs serveurs



## 5.4 Création d'alertes

- Utilisation d'alertes en réponse à des problèmes potentiels
- Consignation d'événements dans le journal d'applications
- Création d'alertes en réponse à des erreurs SQL Server
- Création d'alertes pour une erreur définie par l'utilisateur
- Réponse à des alertes de condition de performances
- Attribution d'un opérateur de prévention de défaillance

# Utilisation d'alertes en réponse à des problèmes potentiels

- Les alertes répondent aux erreurs SQL Server ou aux erreurs définies par l'utilisateur (événements) qui ont été consignées dans le journal d'application de Windows
- Les erreurs SQL Server apparaissent en réponse à des problèmes prédéfinis notamment lorsque l'utilisateur ne dispose pas des autorisations requises pour modifier une table ou lorsque le journal des transactions est saturé
- Pour faire apparaître des messages définis par l'utilisateur, l'application de BD doit appeler l'instruction **RAISERROR**

# Exemple de scénario

- Une gestionnaire de comptes souhaite être **notifiée par courrier électronique** chaque fois **qu'un client est supprimé de la BD**
- Elle veut également connaître le nom de l'employé qui **a effectué cette procédure** dans l'éventualité où cette suppression impliquerait une action

# Procédure d'alertes pour répondre à la demande de la gestionnaire des comptes

1. **Emp1**, employé du service clientèle, **supprime** le client **Cl1** de la table Client. La procédure stockée **RemoveCustomer** est exécutée et **l'erreur** numéro 50099 s'affiche
2. Cette **erreur** (Evt) est **consignée** dans le **journal d'applications** Windows
3. Le **journal d'applications** Windows **prévient** l'**Agent SQL Server** qu'un événement s'est produit
4. L'**Agent SQL Server** **compare** alors **l'erreur** aux **alertes définies** dans la table système **msdb..sysalerts** qui est conservée en mémoire cache
5. L'**Agent SQL Server** **traite** alors **l'alerte** :
  - a. en consultant la table système **msdb..notifications** afin **d'envoyer un message électronique**
  - b. en consultant la table système **msdb..operators** qui **identifie la personne** à laquelle envoyer la notification

# Consignation d'événement dans le journal applications

- Lorsque le service **SQL Server Agent** démarre, il **s'inscrit** auprès de l'Observateur d'événement et **se connecte** au service SQL Server
- Ainsi, **l'Agent SQL Server** peut être informé lorsque des événements SQL Server sont consignés dans le journal d'applications Windows
- L'Agent SQL Server compare alors ces événements aux travaux et alertes mis en cache pour déterminer si une action définie doit être exécutée

# Consignation d'événement dans le journal applications (suite)

SQL Server **consigne des événements** dans le journal d'applications Windows **dans les cas suivants :**

- Des **erreurs** SQL Server ayant un **niveau de gravité** compris entre 19 et 25 se produisent
- Des **messages d'erreur** sont définis pour être consignés dans le journal d'applications Windows à l'aide de la procédure stockée système **sp\_addmessage** ou **sp\_altermessage**
- L'instruction **RAISERROR WITH LOG** est exécutée
- La procédure stockée étendue **xp\_logevent** est exécutée

# Création d'alertes en réponse à des erreurs SQL Server

- Définition d'alertes pour des **numéros d'erreur SQL Server**
- Définition d'alertes pour des **niveaux de gravité d'erreur**

# Définition d'alertes pour des numéros d'erreurs SQL Server

- Sous SQL Server Entreprise Manager ou par la procédure stockée système **sp\_add\_alert**
- La définition de l'alerte est stockée dans la table système **msdb..sysalerts**
- Cette table est conservée en mémoire cache pour améliorer les performances



# Définition d'alertes pour des numéros d'erreurs SQL Server (suite)

Lorsqu'on définit des alertes pour un numéro d'erreur SQL Server, on doit tenir compte de :

- Les numéros d'erreurs doivent être **consignés** dans le **journal d'application Windows**
- On peut définir des alertes pour n'importe quel numéro d'erreur fourni par le système SQL Server ou défini par l'utilisateur et stocké dans la table système **master..sysmessages**
- On peut définir **plusieurs alertes pour un numéro d'erreur** SQL Server. Toutefois, chaque alerte doit être limitée à une BD particulière ou s'appliquer à toutes les BD
- Lorsqu'on crée une alerte qui s'applique à toutes les BD, s'assurer que le **message d'erreur est suffisamment clair**
- On peut définir une chaîne particulière dans le texte du message d'erreur outre le numéro d'erreur (**Exemple** : Tentative de connexion par l'utilisateur '%s')

# Définition d'alertes pour des niveaux de gravité d'erreurs

Lorsqu'on définit des alertes en utilisant des niveaux de gravité d'erreur comme condition, **on doit tenir compte de :**

- Les erreurs SQL Server pourvues des niveaux de gravité situés **entre 19 et 25** sont **automatiquement consignées** dans le journal d'applications Windows
- Les niveaux de gravité situés **entre 20 et 25** correspondent à des **erreurs fatales**. On doit toujours définir l'opérateur à notifier en cas d'apparition d'erreurs SQL Server à ces niveaux de gravité
- SQL Server fournit des **alertes prédéfinies** qu'on peut utiliser. On doit modifier les alertes prédéfinies sur des erreurs fatales en définissant l'opérateur à notifier et en supprimant le terme « demo » du nom de l'alerte

# Définition d'alertes pour des niveaux de gravité d'erreurs (suite)

- On peut **créer une alerte qui se déclenche** lorsqu'une erreur d'un niveau de gravité donné apparaît dans toutes les BD ou dans une BD spécifique
- On peut définir une chaîne particulière dans le texte du message d'erreur outre le niveau de gravité (**Exemple** : créer une alerte avertit, en affichant « espace disque », qu'une erreur de niveau 17 se produit dans une BD quelconque)

# Configuration du transfert d'événement

- On peut **configurer l'Agent SQL Server** pour désigner un **serveur qui recevra tous** les messages d'événement dont le niveau de gravité est  $\geq$  à une valeur donnée
- On **peut transférer des événements vers un serveur** dont le trafic est  $<$  à celui des autres serveurs du domaine

**Exemple** : On peut configurer le transfert des erreurs ayant un niveau de gravité 18 au minimum vers le serveur Comptabilité. Si une erreur de niveau de gravité 19 se produit sur le serveur, l'événement est transféré automatiquement sur le serveur Comptabilité qui devra résoudre le problème

# Création d'alertes pour une erreur définie par l'utilisateur

- Création du message d'erreur
- Déclenchement de l'erreur à partir de l'application de BD
- Définition d'une alerte pour le message d'erreur

# Création du message d'erreur

- Sous SQL Server Entreprise Manager ou par la procédure stockée système **sp\_addmessage**
- Les **numéros des erreurs** définies par l'utilisateur doivent être **> à 50000**. Les numéros d'erreur **< à 50000** sont réservés aux erreurs système SQL Server prédéfinies
- Toutes les erreurs définies par l'utilisateur sont **stockées** dans la table système **sysmessages** de la BD **master**
- Les messages d'erreur peuvent comporter des **paramètres pour capturer des détails spécifiques** afin que les informations appropriées soient fournies pour le statut de détail ou pour résoudre le problème

# Création du message d'erreur (suite)

- Les messages d'erreurs SQL Server s'affichent dans la **langue sélectionnée lors de l'installation**. Si vous administrez un environnement SQL Server multilingue, vous pouvez également créer des messages définis par l'utilisateur pour d'autres langues
- On **doit consigner le message** d'erreur dans le journal d'applications Windows si on envisage de déclencher une alerte pour ce message

# Exemple de création du message d'erreur

```
EXEC sp_addmessage 50099, 16,  
'Client %d supprimé par %s', 'French', 'true'
```

- Cet exemple crée un message d'erreur défini par l'utilisateur numéro 50099, de niveau de gravité 16 et qui est consigné dans le journal d'applications Windows (**true**) lorsque l'erreur se produit
- Le paramètre %d est remplacé par le numéro du client supprimé et le paramètre %s par le nom de l'utilisateur qui a exécuté l'instruction DELETE sur la table **Client**



# Déclenchement de l'erreur à partir de l'application de BD

- Utiliser l'instruction **RAISERROR** dans une procédure stockée ou un déclencheur
- RAISERROR renvoie un message d'erreur défini par l'utilisateur et définit un indicateur système (dans la fonction système @@error) pour enregistrer une erreur survenue

# Déclenchement de l'erreur à partir de l'application de BD (Exemple)

```
CREATE PROCEDURE removecustomer
@CustomerID varchar(5)=NULL
AS
.
DECLARE @username varchar(60)
SELECT @username=suser_sname()
BEGIN TRANSACTION
    DELETE Customers
    WHERE CustomerID=@CustomerID
    RAISERROR (50099, 16, 1, @CustomerID, @username)
COMMIT TRANSACTION
```

# Définition d'une alerte pour le message d'erreur

- Une alerte est créée pour l'erreur numéro 50099 afin qu'un message électronique contenant le texte du message d'erreur soit envoyé au gestionnaire des comptes
- Lorsqu'un utilisateur exécute la procédure stockée **RemoveCustomer**, l'erreur 50099 est déclenchée, elle est consignée dans le journal d'applications Windows
- L'alerte pour le numéro d'erreur est déclenchée et envoie au gestionnaire des comptes un message électronique contenant le texte du message d'erreur

**Résultats** : Error: 50099, Severity: 16, State 1

A. ETTALBI Client 732 supprimé par ACCOUNTING\evacorets

# Réponse à des alertes de condition de performance

- On peut utiliser des **alertes à la fois en réponse** à des erreurs **SQL Server** et à des **conditions de performances** SQL Server comme celles définies dans le Moniteur système Windows
- Par exemple**, on peut créer une alerte de condition de performances qui se déclenche lorsque le journal des transactions de la BD dépasse 75% de sa capacité
- La réponse à l'alerte peut exécuter un travail qui sauvegarde le journal des transactions et notifie l'administrateur de la BD
- On peut créer des alertes de condition de performances sur des ressources SQL Server comportant certains des objets suivants : ***Méthodes d'accès, Gestionnaires de tampons, Gestionnaire de cache, Bases de données, Verrous, Statistiques SQL***

# Attribution d'un opérateur de prévention de défaillance

-On peut attribuer un opérateur de prévention de défaillance pour répondre à une alerte déclenchée par l'échec des notifications envoyées par radiomessagerie à des opérateurs définis

-**Par exemple**, Si un utilisateur n'est pas de service le Jeudi et qu'une alerte s'est déclenchée ce jour-là à 01:30, l'opérateur de prévention de défaillance est contacté

Un **opérateur de prévention** de défaillance **est contacté** lorsque **toutes ces conditions sont réunies** :

-Des **notifications** par radiomessagerie ont été **créées** pour la réponse à l'alerte

-Aucun des opérateurs à prévenir par radiomessagerie n'est de service

-Un opérateur de prévention de défaillance est défini

-La session de messagerie de **l'Agent SQL Server est démarrée**

# Attribution d'un opérateur de prévention de défaillance (suite)

Lorsqu'on attribue un opérateur de prévention de défaillance, on doit tenir compte de :

- Les informations de l'opérateur de prévention de défaillance sont mises en mémoire cache, elles ne dépendent pas donc de la connexion à la BD **msdb**
- On ne peut avoir qu'un seul opérateur de prévention de défaillance
- **On ne peut pas supprimer un opérateur qui a été désigné comme opérateur de prévention de défaillance.** Mais, on peut supprimer l'attribution d'opérateur de prévention de défaillance puis supprimer l'opérateur

# 5. Exécution de tâches administratives

5.1 Tâches liées à la configuration

5.2 Tâches de routine liées à l'administration de SQL Server

5.3 Automatisation des tâches de maintenance de routine

5.4 Création d'alertes

**5.5 Résolution des problèmes liés à l'automatisation de SQL Server**

5.6 Automatisation de travaux sur plusieurs serveurs

# 5.5 Résolution des problèmes liés à l'automatisation de SQL Server

Si les travaux, alertes ou notifications automatisés ne fonctionnent pas correctement alors :

- Vérifier que l'Agent SQL Server est **démarré**, que le rôle **sysadmin** est attribué à son compte de connexion et que le mot de passe est valide
- Vérifier que le travail, la planification, l'alerte et l'opérateur sont **activés**
- Vérifier que les utilisateurs disposent des autorisations pour exécuter ces types de travaux et que le compte d'utilisateur de domaine utilisé en tant que compte proxy dispose des autorisations pour exécuter les travaux



## 5.5 Résolution des problèmes liés à l'automatisation de SQL Server (suite)

- **Consulter les messages d'erreurs** dans le journal d'applications Windows et les journaux d'erreurs de l'Agent SQL Server pour déterminer l'origine du problème et le résoudre
- **Vérifier que la taille du fichier** et la taille de croissance de la BD **msdb** correspond au nombre de lignes conservées dans la table système **sysjobhistory**
- **Vérifier le fonctionnement du client de messagerie**

# Résolution des problèmes liés aux alertes

## ➤ Facteurs pouvant entraîner un retard de traitement des alertes

- Le journal d'application Windows est plein
- L'utilisation du processeur est inhabituellement élevée
- Le nombre de réponses aux alertes est élevé

## ➤ Résolution du retard de traitement des alertes

- Désactivez temporairement l'alerte
- Augmenter le délai entre les réponses
- Corriger le problème des ressources globales
- Vider le journal d'applications Windows

# 5. Exécution de tâches administratives

5.1 Tâches liées à la configuration

5.2 Tâches de routine liées à l'administration de SQL Server

5.3 Automatisation des tâches de maintenance de routine

5.4 Création d'alertes

5.5 Résolution des problèmes liés à l'automatisation de SQL Server

**5.6 Automatisation de travaux sur plusieurs serveurs**

## 5.6 Automatisation de travaux sur plusieurs serveurs

- Définition d'un **serveur principal** qui définit, planifie et gère les travaux sur tous les serveurs cibles et de **serveurs cibles**
- Utiliser SQL Server Enterprise Manager ou la procédure stockée système **sp\_msx\_enlist** pour définir un serveur principal
- On doit lui inscrire au **moins un serveur cible**
- Une ligne sera stockée dans la table système **systargetservers**
- Un **compte et un mot de passe** d'ouverture de session SQL Server sont **automatiquement créés** pour **chaque serveur cible** avec le suffixe **msx\_probe**

## 5.6 Automatisation de travaux sur plusieurs serveurs (suite)

- L'Assistant crée un opérateur **MSXOperator** sur le serveur principal et sur chaque serveur cible
- Le serveur principal représente généralement le serveur d'un service principal ou d'une unité d'entreprise principale. Dans une structure plus petite, il dessert toute l'entreprise
- On peut aussi désigner le serveur principal comme serveur de transfert d'événements
- Les **serveurs cibles** sont attribués à un **seul serveur principal**
- Ils doivent être situés dans le **même domaine Windows** que le serveur principal

# Définition des travaux sur plusieurs serveurs

- Après avoir défini le serveur principal et les serveurs cibles, on peut **créer sur le serveur principal des travaux** qui seront **exécutés sur un ou plusieurs serveurs cibles**
- Pour traiter les travaux dans un environnement multiserveurs, SQL Server exécute la procédure suivantes :
  - Le serveur principal poste des travaux pour les serveurs cibles dans une liste de téléchargement de la table système **msdb..sysdownloadlist**
  - Les serveurs cibles se connectent périodiquement au serveur principal pour déterminer si des travaux nouveaux ou mis à jour ont été postés
  - A la fin du travail, le serveur cible envoie l'état du résultat du travail

# Modification des définition de travaux sur plusieurs serveurs

- Les définitions des travaux ne peuvent être modifiés sur le serveur cible ; les **modifications doivent être effectuées sur le serveur principal**
- SQL Server Entreprise Manager **poste automatiquement** les instructions nécessaires dans la liste de téléchargement

# 6. Sauvegarde de Base de Données

## 6.1 Protection contre les pertes de données

6.2 Définition et changement de mode de récupération de BD

6.3 Sauvegarde de SQL Server

6.4 Moment approprié pour sauvegarder des BD

6.5 Exécution de sauvegardes

6.6 Types de méthodes de sauvegarde

6.7 Planification d'une stratégie de sauvegarde



# 6.1 Protection contre les pertes de données

- Pertes dues à une mauvaise utilisation de DELETE, UPDATE, propagation de virus destructeurs, catastrophes naturelles (incendie,...), vol
- Définition d'une stratégie de sauvegarde
  - Pour minimiser les pertes de données
  - Pour récupérer les données perdues
  - Pour restaurer les données avec un impact minimal sur la production
- Sauvegardes régulières

# Sauvegardes régulières

- Faire des **sauvegardes fréquentes** si le système se trouve dans un **environnement OLTP**
- Faire des **sauvegardes moins fréquentes** si le système a **peu d'activités (OLAP)**
- **Planifier les sauvegardes** à un moment où **aucune mise à jour majeure** n'est effectuée dans SQL Server
- Après avoir déterminé la stratégie de sauvegarde, **on peut automatiser le processus** à l'aide de l'Assistant Plan de maintenance de BD

# 6. Sauvegarde de Base de Données

6.1 Protection contre les pertes de données

**6.2 Définition et changement de mode de récupération de BD**

6.3 Sauvegarde de SQL Server

6.4 Moment approprié pour sauvegarder des BD

6.5 Exécution de sauvegardes

6.6 Types de méthodes de sauvegarde

6.7 Planification d'une stratégie de sauvegarde

## 6.2 Définition et changement de mode de récupération de BD

### ➤ Définition d'un mode de récupération de BD

- Mode récupération complète
- Mode de récupération Bulk\_Logged
- Mode de récupération simple

### ➤ Changement de mode de récupération de BD

ALTER DATABASE NomBD

SET RECOVERY {**FULL** | **SIMPLE** | **BULK\_LOGGED**}

# Mode de récupération complète

- Utilisé lorsque la **récupération totale du support** endommagé est la **plus prioritaire**
- Utilise des **copies de la BD** et toutes les **informations** figurant dans les **journaux** pour restaurer la BD
- SQL Server consigne **toutes les modifications apportées à la BD**, y compris les opérations de chargement en bloc et les créations d'index.

# Mode de récupération complète (Suite)

- Si les journaux eux-mêmes ne sont pas endommagés, SQL Server peut récupérer toutes les données sauf les transactions en cours au moment de la défaillance
- On peut **insérer des marques** nommées dans le journal des transactions afin de permettre la récupération jusqu'à cette marque précise

# Mode de récupération Bulk\_Logged

- Utilise aussi les sauvegardes des BD et des journaux pour recréer la BD
- Occupe moins de place dans le journal pour les opérations suivantes : CREATE INDEX, les opérations de chargement de bloc, SELECT INTO, WRITETEXT et UPDATETEXT
- Le journal n'enregistre que l'occurrence de ces opérations
- permet de restaurer toutes les données **mais pas partiellement** (à une marque donnée par exemple)

# Mode de récupération simple

- Utilisé généralement pour les BD **peu volumineuses** ou dans lesquelles les données sont rarement modifiées
- Utilise des copies complètes ou différentielles de la BD
- La récupération **se limite** à la restauration de la BD jusqu'au moment de la dernière sauvegarde
- Toutes les modifications effectuées après la sauvegarde **sont perdues** et doivent être recréées
- **Avantage** : faible espace de stockage occupé par les journaux et facilité de mise en œuvre



# 6. Sauvegarde de Base de Données

6.1 Protection contre les pertes de données

6.2 Définition et changement de mode de  
récupération de BD

## 6.3 Sauvegarde de SQL Server

6.4 Moment approprié pour sauvegarder des BD

6.5 Exécution de sauvegardes

6.6 Types de méthodes de sauvegarde

6.7 Planification d'une stratégie de sauvegarde

## 6.3 Sauvegarde de SQL Server

- Permet de **sauvegarder une BD** alors qu'elle est **en cours d'utilisation**
- Sauvegarde les fichiers de BD originaux et enregistre leur emplacement
- La sauvegarde contient le **schéma et la structure de fichiers**, les **données** et des **parties des fichiers journaux** de transactions contenant les activités de BD survenues depuis le début du processus de sauvegarde
- SQL Server utilise ces sauvegardes pour **recréer** les fichiers à **leur emplacement d'origine** avec les objets et les données lorsqu'on restaure la BD

## 6.3 Sauvegarde de SQL Server (suite)

SQL Server **capture les activités de BD** qui se produisent au cours du processus de sauvegarde :

- SQL Server **exécute un point de contrôle** sur la BD et **enregistre le numéro de séquence de journal (LSN)** de l'enregistrement le plus ancien dans le journal des transactions actif
- SQL Server **écrit toutes les pages** sur le support de sauvegarde en lisant directement les disques
- SQL Server écrit tous les enregistrements du journal des transactions **ajoutés pendant le processus de sauvegarde** (entre le LSN et la fin)

# Exécution et stockage des sauvegardes

## ➤ *Personnes autorisées à effectuer des sauvegardes:*

- Membres du rôle fixe de serveur **sysadmin**
- Membres des rôles fixes de BD **db\_owner** et **db\_backupoperator**
- Membres d'un rôle défini par l'utilisateur

## ➤ *Emplacements de stockage des sauvegardes:*

- Fichier sur le disque dur local ou en réseau
- Bande (le lecteur de bande doit être connecté localement à SQL Server)
- Canal nommé

# 6. Sauvegarde de Base de Données

6.1 Protection contre les pertes de données

6.2 Définition et changement de mode de  
récupération de BD

6.3 Sauvegarde de SQL Server

**6.4 Moment approprié pour sauvegarder des BD**

6.5 Exécution de sauvegardes

6.6 Types de méthodes de sauvegarde

6.7 Planification d'une stratégie de sauvegarde

## 6.4 Moment approprié pour sauvegarder des BD

- Sauvegarde de BD système
- Sauvegarde de BD utilisateur
- Activités à éviter pendant la sauvegarde

# Sauvegarde de BD système

## ➤ Après modification de la BD **master**

- En utilisant l'instruction CREATE DATABASE, ALTER DATABASE, ou DROP DATABASE
- En utilisant certaines procédures stockées système (sp\_logdevice, sp\_addserver, sp\_dropserver, sp\_addlinkedserver et sp\_addmessage)

## ➤ Après modification de la BD **msdb**

## ➤ Après modification de la BD **model**

# Après modification de la BD master

- La BD **master** contient des informations sur toutes les BD d'un serveur SQL Server
- Donc, il faut la sauvegarder à chaque création, changement ou suppression de BD
- Si la BD master est endommagé, on peut alors récupérer et restaurer les BD utilisateur



# Après modification de la BD master (suite)

- **sp\_logdevice** : qui modifie le journal des transactions
- **sp\_addserver, sp\_dropserver, sp\_addlinkedserver** : qui ajoutent ou suppriment des serveurs
- **sp\_addmessage** : qui ajoute un message d'erreur

# Après modification de la BD msdb

- La BD **msdb** contient des informations sur les travaux, les alertes et les opérateurs utilisés par l'Agent SQL Server
- Donc, en cas de défaillance du système, on doit reconstruire toutes les BD système puis recréer chaque travail, alerte et opérateur

# Après modification de la BD model

- Contient la configuration par défaut des BD utilisateur
- A sauvegarder si on lui apporte des modifications

# Sauvegarde de BD utilisateur

- Après création de BD ou son chargement en données
- Après création d'index
- Après vidage du journal des transactions par `BACKUP LOG WITH TRUNCATE_ONLY` ou `BACKUP LOG WITH NO_LOG`
- Après exécution d'opérations non journalisées (non stockées dans le journal des transactions)

# Opérations non journalisées

- BACKUP LOG WITH TRUNCATE\_ONLY ou BACKUP LOG WITH NO\_LOG
- Instructions WRITETEXT ou UPDATETEXT (On peut spécifier l'option WITH LOG pour consigner ces activités dans le journal des transactions)
- Instruction SELECT...INTO lors de la création d'une table permanente ou l'utilisation du programme de copie en bloc (*Bulk Copy Program*)

# Activités à éviter pendant les sauvegardes

- Création ou modification de BD
- Exécution d'opérateurs de croissance automatique
- Création d'index
- Exécution d'opérations non journalisées
- Compactage d'une BD

# 6. Sauvegarde de Base de Données

6.1 Protection contre les pertes de données

6.2 Définition et changement de mode de  
récupération de BD

6.3 Sauvegarde de SQL Server

6.4 Moment approprié pour sauvegarder des BD

**6.5 Exécution de sauvegardes**

6.6 Types de méthodes de sauvegarde

6.7 Planification d'une stratégie de sauvegarde

## 6.5 Exécution de sauvegardes

- Création **d'unités** de sauvegarde
- Création de **fichiers de sauvegarde** sans unité permanente
- Utilisation de **plusieurs fichiers** de sauvegarde pour stocker les sauvegardes
- Utilisation de l'instruction **BACKUP**
- Sauvegarde sur un **lecteur de bande**



# Création d'unités de sauvegarde

- Raisons justifiant la création d'unités de sauvegarde permanentes
  - Réutilisation de fichiers de sauvegarde
  - Automatisation des sauvegardes
- Utilisation de la procédure stockée système **sp\_addumpdevice**
  - Spécifier le nom logique
  - Les noms logiques et physiques sont stockées dans la table système **sysdevices** de la BD **master**

# Exemples

## Exemple 1 :

*USE master*

*EXEC sp\_addumpdevice 'disk',*

*'mybackupfile',*

*'C:\Buckup\MyBackupFile.bak'*

➔ création d'un fichier de sauvegarde sur le disque

## Exemple 2 :

*USE master*

*EXEC sp\_addumpdevice 'tape',*

*'mytape1', '\\.\tape0'*

➔ création d'une unité de sauvegarde sur une bande

# Création de fichiers de sauvegarde sans unité de sauvegarde

- Raisons justifiant la création d'unités de sauvegarde sans unité permanente
  - Exécution d'une seule sauvegarde
  - Test de l'opération de sauvegarde qu'on prévoit automatiser
- Utilisation de l'instruction BACKUP DATABASE
  - Spécifier un type de support (disque, bande ou canal nommé)
  - Spécifier un chemin d'accès et un nom de fichier complets

# Exemples

## Exemple :

*USE master*

*BACKUP DATABASE Northwind*

*TO DISK='C:\Temp\MyCustomers.bak'*

➔ création d'un fichier de sauvegarde temporaire sur le disque pour sauvegarder la BD master

# Utilisation de plusieurs fichiers de sauvegarde pour stocker les sauvegardes

- **Intérêt** : réduire la durée totale de sauvegarde de la BD
- Toutes les unités utilisées dans une opérations de sauvegarde doivent reposer sur le **même type** de support (disque ou bande)
- On peut combiner des fichiers permanents et temporaires
- Si on reformate un fichier, les données contenues dans les autres fichiers ne sont plus valides

# Utilisation de l'option MEDIANAME

```
BACKUP DATABASE {bd | @var_nom_bd}
```

```
TO <unité_sauvegarde> [,...]
```

```
[WITH
```

```
  [MEDIANAME={support | @var_nom_support}]]
```

➔ Spécifie le nom d'un jeu de supports de sauvegarde complet

➔ Associe les différents fichiers au sein d'un même jeu de supports de sauvegarde

➔ Une fois le jeu de support créé et nommé, on peut l'utiliser dans des opérations de sauvegarde ultérieures

# Utilisation des options INIT, NOINIT et FORMAT

- L'option **NOINIT** ajoute les sauvegardes à un fichier (prise par défaut)
- L'option **INIT** écrase un fichier de sauvegarde
- L'option **FORMAT**
  - Ecrase le contenu d'un fichier de sauvegarde
  - Subdivise un jeu de sauvegardes réparti

# Utilisation des options INIT, NOINIT et FORMAT (suite)

L'opération de sauvegarde échoue et les données ne sont pas remplacées (**INIT**) dans les cas suivants :

- L'option **EXPIREDATE** spécifiée sur l'unité de sauvegarde n'est pas encore arrivée à l'expiration
- Les paramètres *jeu\_sauvegardes* spécifiés dans l'option **NAME** ne correspondent pas au même paramètre de l'unité de sauvegarde
- On tente de remplacer un membre de jeu de sauvegardes précédemment nommé (membre d'un autre jeu de sauvegarde)



# Sauvegarde sur un lecteur de bande

Bandes : support de stockage pratique car:

- Peu coûteuses
- Offrent un volume de stockage important
- Peuvent être conservées en dehors du site pour une meilleure sécurité des données

# Sauvegarde sur un lecteur de bande

- Rattachement local du lecteur de bandes à SQL Server
- Enregistrement des informations de sauvegarde sur l'étiquette de la bande (Nom de la BD, Heure et Date, Type de sauvegarde)
- Stockage de sauvegardes SQL Server et non-SQL Server (Format de sauvegarde standard : ***MS Tape Format***)

# Spécification des options de bande

Option	Description
UNLOAD (Par défaut)	Rembobine et extrait automatiquement la bande une fois la sauvegarde terminée
NOUNLOAD	Ne rembobine pas et n'extrait pas automatiquement la bande
BLOCKSIZE	Change la taille des blocs physiques exprimée en octets
FORMAT	Ecrit un en-tête sur les fichiers utilisés pour une sauvegarde
SKIP	Ignore les étiquettes de bande ANSI
NOSKIP (Par défaut)	Lit les étiquettes de bande ANSI
RESTART	Redémarre l'opération de sauvegarde à partir du point d'interruption

# 6. Sauvegarde de Base de Données

6.1 Protection contre les pertes de données

6.2 Définition et changement de mode de  
récupération de BD

6.3 Sauvegarde de SQL Server

6.4 Moment approprié pour sauvegarder des BD

6.5 Exécution de sauvegardes

**6.6 Types de méthodes de sauvegarde**

6.7 Planification d'une stratégie de sauvegarde

## 6.6 Types de méthodes de sauvegarde

- Sauvegarde de BD complète
- Sauvegarde différentielle
- Sauvegarde du journal des transactions
- Sauvegarde d'un fichier ou d'un groupe de fichiers de BD

# Sauvegarde de BD complète

- Sert de point de référence en cas de défaillance du système
- Sauvegarde les fichiers, objets et données d'origine
- Sauvegarde des parties du journal des transactions

# Sauvegarde de BD complète (suite)

- SQL Server sauvegarde toutes les activités survenues pendant la sauvegarde
- Il sauvegarde toutes les transactions non validées dans le journal des transactions

# Exemple 1

USE master

```
EXEC sp_addumpdevice 'disk', 'NwindBac',  
'C:\NwindBac.bak'
```

**BACKUP DATABASE** Northwind TO NwindBac

➔ Une unité de sauvegarde est créée et une sauvegarde complète est effectuée



## Exemple 2

**BACKUP DATABASE Northwind TO NwindBac  
WITH INIT**

➔ L'intégralité de la BD est sauvegardée dans le fichier *NwindBac* et toutes les sauvegardes précédentes dans le fichiers sont écrasées

## Exemple 3

**BACKUP DATABASE Northwind TO NwindBac  
WITH NOINIT**

➔ Une sauvegarde de la BD complète est ajoutée au fichier *NwindBac* et tout fichier de sauvegarde précédent est conservé

## Exemple 4

```
BACKUP DATABASE Northwind TO  
DISK='D:\MyTempBackup.bak'
```

➔ Crée un fichier de sauvegarde sur le disque et effectue une sauvegarde complète de cette BD dans ce fichier

# Sauvegarde de BD différentielle

- Convient aux BD souvent modifiées
- **Requiert** une sauvegarde de BD **complète**
- Sauvegarde les modifications effectuées **depuis la dernière** sauvegarde de BD complète
- **Réduit** la durée des processus de sauvegarde et de restauration

# Sauvegarde de BD différentielle (suite)

Lors d'une sauvegarde différentielle, SQL Server :

- Sauvegarde les parties de la BD modifiées depuis la dernière sauvegarde complète de la BD
- Sauvegarde toute activité survenue pendant la sauvegarde différentielle ainsi que toute transaction non validée dans le journal des transactions

# Sauvegarde de BD différentielle (suite)

Lors d'une sauvegarde différentielle :

- Si une ligne de la BD a été **modifiée plusieurs fois** depuis la dernière sauvegarde de BD complète, la sauvegarde différentielle contient uniquement le **dernier** ensemble de valeurs de cette ligne
- Alors que **dans une sauvegarde du journal des transactions**, le journal contient un **historique de toutes les modifications** apportée à cette ligne

# Sauvegarde de BD différentielle (suite)

- La **durée de sauvegarde** d'une BD est **réduite** car les jeux de sauvegardes sont plus petits que pour une sauvegarde complète
- La **durée de restauration** d'une BD est **réduite** car il n'est pas nécessaire d'appliquer une série de journaux de transactions
- On doit définir une **convention de nommage** pour les fichiers de sauvegarde contenant des sauvegardes différentielles pour les **distinguer** des fichiers contenant des sauvegardes complètes

# Syntaxe partielle et exemple

## Syntaxe partielle :

```
BACKUP DATABASE {BD | @Var_Nom_BD}  
TO <unité_sauvegarde> [,...n]  
[WITH [DIFFERENTIAL]]
```

## Exemple : BACKUP DATABASE Northwind

```
TO DISK='D:MyDiffBackup.bak'  
WITH DIFFERENTIAL
```

➔ Une sauvegarde différentielle est créée dans un fichier de sauvegarde temporaire



# Sauvegarde du journal des transactions

- **Requiert** une sauvegarde de BD **complète**
- Sauvegarde le journal des transactions **entre** la dernière instruction **BUCKUP LOG** exécutée correctement et la fin du journal de transactions en cours
- **Tronque** le journal des transactions

# Sauvegarde du journal des transactions (suite)

Lorsqu'on sauvegarde le journal des transactions,  
SQL Server :

- Sauvegarde le journal des transactions à partir de la dernière instruction BACKUP LOG exécutée avec succès et jusqu'à la fin du journal des transactions en cours
- Tronque le journal des transactions jusqu'au début de la partie active et **élimine la partie inactive**

# Syntaxe partielle et exemple

## Syntaxe partielle :

BUCKUP LOG {*BD* | @*Var\_Nom\_BD*}

TO <*unité\_sauvegarde*> [,...n]

[WITH [,] {INIT | NOINIT}]

## Exemple : USE master

EXEC sp\_addumpdevice 'disk', 'NwinBacLog',  
'D:\NwinBacLog.bak'

BACKUP LOG Northwind TO NwinBacLog

➔ Crée une unité de sauvegarde pour le journal et sauvegarde le journal des transactions de la BD Northwind

# Utilisation de l'option NO\_TRUNCATE

SQL Server :

- Enregistre le journal des transactions** dans son intégralité même si la BD est inaccessible
- Ne supprime pas les transactions validées** du journal des transactions
- Permet de récupérer les données jusqu'au moment où le système a subi la défaillance**

# Utilisation de l'option **NO\_TRUNCATE** (suite)

Lorsqu'on restaure la BD, on peut restaurer la sauvegarde de la BD et appliquer la sauvegarde du journal des transactions créée à l'aide de l'option **NO\_TRUNCATE** pour récupérer les données

# Vidage du journal des transactions

- Si plein**, on ne peut pas mettre à jour la BD ni restaurer entièrement la BD en cas de défaillance
- On **doit donc vider** le journal **soit** en effectuant une **sauvegarde complète** de la BD et en enregistrant les données, **soit** en **tronquant** le journal des transactions
- Par l'instruction **BUCKUP LOG** avec l'option **TRUNCATE\_ONLY** ou **NO\_LOG**

# Vidage du journal des transactions (suite)

- SQL Server supprime la partie inactive du journal sans en faire une copie de sauvegarde
- La partie active du journal des transactions constituée de **transactions non validées** n'est jamais tronquée

# Vidage du journal des transactions (suite)

Lorsqu'on tronque le journal des transactions, on **doit tenir compte des instructions suivantes** :

- Vider le journal des transactions avant de sauvegarder la BD **réduit la taille** de la sauvegarde complète de la BD
- On ne peut pas récupérer les modifications enregistrées dans le journal des transactions. On doit exécuter l'instruction **BUCKUP DATABASE** immédiatement
- La troncature du journal des transactions n'est pas enregistrée



# Syntaxe partielle et exemples

## Syntaxe :

```
BUCKUP LOG {BD / @Var_Nom_BD}  
[WITH {TRUNCATE_ONLY | NO_LOG}]
```

## Exemple 1 :

```
BACKUP LOG Northwind WITH TRUNCATE_ONLY
```

➔ Suppression de la partie inactive d'un journal des transactions sans faire de copie de sauvegarde

## Exemple 2 :

```
BACKUP LOG Northwind WITH NO_LOG
```

➔ Suppression de la partie inactive d'un journal des transactions plein sans faire de copie de sauvegarde

# Définition de l'option *trunc. log on chkpt*

- On peut attribuer la valeur **true** à cette option pour **écrire toutes les transactions validées** dans le BD lors **d'un point de contrôle**
- Cette option **tronque automatiquement** le journal des transactions
- Elle est fournie uniquement pour assurer une **compatibilité ascendante**.
- Le mode de **récupération simple** la remplace

# Définition de l'option *trunc. log on chkpt*

- Si on attribue **true** à cette option, **on ne peut pas sauvegarder le journal des transactions** et l'utiliser pour restaurer la BD en cas de défaillance du système
- Le journal des transactions **ne stocke plus** les modifications apportées à la BD depuis la dernière sauvegarde complète de cette BD

# Sauvegarde d'un fichier ou d'un groupe de fichiers de BD

- Utilisé lorsqu'on ne peut pas réaliser une sauvegarde complète (BD très volumineuses)
- SQL Server effectue les tâches suivantes :
  - Sauvegarde uniquement les fichiers de BD spécifiés dans l'option FILE ou FILEGROUP
  - Vous permet de sauvegarder des fichiers de BD spécifiques et non toute la BD

# Sauvegarde d'un fichier ou d'un groupe de fichiers de BD (suite)

- Lorsqu'on effectue une telle sauvegarde :
  - On doit spécifier les fichiers ou les groupes de fichiers logiques
  - On doit effectuer des sauvegardes du journal des transactions pour assurer la cohérence des fichiers restaurés par rapport à la BD restaurée
  - Il est recommandé d'établir une stratégie de sauvegarde par rotation
  - On peut spécifier jusqu'à 16 fichiers ou groupes de fichiers

# Syntaxe partielle

```
BACKUP DATABASE {BD | @Var_Nom_BD}  
[<fichier_ou_groupefichiers>[,...n]  
TO <unité_sauvegarde> [,...n]]
```

Où <fichier\_ou\_groupefichiers> représente :

```
{FILE={nom_fichier_logique | @var_nom_fichier_logique}  
| FILEGROUP={nom_groupe_fichier_logique}  
}
```

# Exemple

**BACKUP DATABASE** PhoneOrders

**FILE=Orders2 TO** OrderBackup2

**BUCKUP LOG** PhoneOrders to OrderLog

➔ Le fichier Orders2 d'un groupe de fichiers de BD est sauvegardé et le journal des transactions est stocké dans le fichier OrderLog

# 6. Sauvegarde de Base de Données

6.1 Protection contre les pertes de données

6.2 Définition et changement de mode de récupération de BD

6.3 Sauvegarde de SQL Server

6.4 Moment approprié pour sauvegarder des BD

6.5 Exécution de sauvegardes

6.6 Types de méthodes de sauvegarde

**6.7 Planification d'une stratégie de sauvegarde**



## 6.7 Planification d'une stratégie de sauvegarde

- Nécessité d'une stratégie de sauvegarde
- Choisir la ou les méthodes adaptées à l'environnement de l'entreprise
- Tenir compte du processus de restauration
- Tenir compte des exigences liées à chaque stratégie

# Types de stratégie de sauvegarde

- Stratégie de sauvegarde complète de BD
- Stratégie de sauvegarde complète de BD et de sauvegarde du journal des transactions
- Stratégie de sauvegarde différentielle
- Stratégie de sauvegarde de fichiers ou de groupes de fichiers de BD

# Stratégie de sauvegarde complète

- La taille de la BD et la fréquence des modifications déterminent le temps et les ressources nécessaires
- Utile dans le cas des BD petites
  - ➔ le temps de sauvegarde est raisonnable

# Stratégie de sauvegarde complète de BD et de sauvegarde du journal des transactions

- Sauvegarde complète
- En plus, sauvegarde du journal des transactions pour enregistrer toutes les activités de BD survenues entre deux sauvegardes complètes
- Stratégie de sauvegarde couramment utilisée

# Stratégie de sauvegarde différentielle

- Sauvegarde complète
- Des sauvegardes du journal des transactions
- Sauvegarde différentielle
- Il suffit de restaurer la dernière sauvegarde différentielle pour récupérer une BD
- Cette dernière sauvegarde contient toutes les modifications apportées à la BD depuis la dernière sauvegarde complète

# Stratégie de sauvegarde de fichiers ou de groupes de fichiers de BD

- A utiliser pour des BD très volumineuses partitionnées sur plusieurs fichiers
- A sauvegarder aussi le journal des transactions

# Conseils pratiques



**Définissez une stratégie de sauvegarde**



**Sauvegardez les bases de données système après les avoir modifiées**



**Planifiez des opérations de sauvegarde à un moment où l'activité de la base de données est faible**



**Créez des unités de sauvegarde**



**Testez votre stratégie de sauvegarde**

# 7. Restauration de Base de Données

## 7.1 Processus de récupération de SQL Server

7.2 Préparation de la restauration d'une BD

7.3 Restauration de sauvegardes

7.4 Restauration de BD à partir de différents  
types de sauvegardes

7.5 Restauration de BD système endommagées



# 7.1 Processus de récupération de SQL Server

- SQL Server examine le journal des transactions depuis le **dernier CHECKPOINT** jusqu'au moment de la **panne**
- Si le journal des transactions contient des **transactions validées** qui n'ont pas encore été écrites dans la BD, SQL Server les transmet en **appliquant les modifications** dans la BD
- Si le journal des transactions contient des **transactions non validées**, SQL Server les **annule**

# Activités exécutées par SQL Server pendant le processus de restauration

- SQL Server effectue un contrôle de sécurité et ne restaure pas si :
  - BD existe déjà
  - Fichiers de BD sont différents
  - Fichiers de BD sont incomplets
- Lors d'une restauration de BD à partir d'une sauvegarde complète, SQL Server reconstitue la BD et tous ses fichiers (schéma complet de la BD)

# 7. Restauration de Base de Données

7.1 Processus de récupération de SQL Server

**7.2 Préparation de la restauration d'une BD**

7.3 Restauration de sauvegardes

7.4 Restauration de BD à partir de différents  
types de sauvegardes

7.5 Restauration de BD système endommagées

## 7.2 Préparation de la restauration d'une BD

- On doit vérifier les sauvegardes pour s'assurer qu'on restaure les données et les objets voulus et que la sauvegarde contient les informations valides
- On doit exécuter certaines tâches nécessaires au lancement du processus de restauration

# Vérification des sauvegardes

Instruction **RESTORE HEADERONLY** :

- Renvoie les **informations d'en-tête d'un fichier de sauvegarde ou d'un jeu de sauvegardes** :  
nom et description de fichier de sauvegarde,  
type de support, méthode de sauvegarde, date  
et heure de la sauvegarde, taille de la  
sauvegarde

# Vérification des sauvegardes (suite)

## Instruction **RESTORE FILELISTONLY** :

- Renvoie des **informations sur les fichiers de BD ou de journal des transactions d'origine** : noms logiques et physiques des fichiers de BD et de journal des transactions, type de fichier (de BD ou de journal des transactions), appartenance à un groupe de fichiers, taille du jeu de sauvegarde en MO, taille de fichier maximale autorisée en MO

# Vérification des sauvegardes (suite)

Instruction **RESTORE LABELONLY** :

- Renvoie des informations sur le support de sauvegarde contenant le fichier de sauvegarde

Instruction **RESTORE VERIFYONLY** :

- Vérifie que les fichiers constituant le jeu de sauvegardes sont complets et que toutes les sauvegardes sont lisibles

# Exécution de tâches spécifiques avant la restauration de sauvegardes

- Restriction de l'accès à la BD
  - Limitez l'accès aux membres du rôle **db\_owner**, **dbcreator** ou **sysadmin**
- Sauvegarde du journal des transactions
  - Assure la cohérence de la BD
  - Capture les modifications effectuées entre la dernière sauvegarde du journal des transactions et le moment auquel la BD a été déconnectée



# 7. Restauration de Base de Données

7.1 Processus de récupération de SQL Server

7.2 Préparation de la restauration d'une BD

**7.3 Restauration de sauvegardes**

7.4 Restauration de BD à partir de différents  
types de sauvegardes

7.5 Restauration de BD système endommagées

## 7.3 Restauration de sauvegardes

- Par SQL Server Enterprise Manager ou l'instruction **RESTORE**
- On peut spécifier des **options** propres au type de sauvegarde qu'on veut restaurer
- On peut aussi décider si on désire lancer le processus de récupération après chaque opération de restauration

# Utilisation de l'instruction RESTORE

- On n'a pas à supprimer la BD endommagée
- SQL Server crée automatiquement les fichiers et les objets de BD
- Les options de restauration permettent de détailler la façon de restaurer les sauvegardes

# Syntaxe partielle

```
RESTORE DATABASE {BD | @var_nom_bd}  
[FROM <unité_sauvegarde> [,...n]]  
[WITH  
    [FILE=numérofichier]  
    [[,]MOVE 'nom_fichier_logique' TO 'nom_fichier_SE']  
    [[,]REPLACE]  
    [[,] {NORECOVERY | RECOVERY | STANDBY=  
        nom_fichier_annulation}]]  
    [[,]RESTART]]
```

Où <unité\_sauvegarde> représente :

```
{{unité_sauvegarde | @var_nom_unité_sauvegarde}}  
{DISK | TAPE | PIPE}='unité_sauvegarde_temporaire' |  
@var_nom_unité_sauvegarde_temporaire}}
```

# Exemple

USE master

**RESTORE DATABASE** Northwind

FROM NwinBac

➔ Restauration de la BD Northwind à partir  
d'un fichier de sauvegarde permanent

# Lancement du processus de récupération

Utilisation de l'option **RECOVERY** (par défaut)

- **A utiliser lors** de la restauration du **dernier journal des transactions** ou dans le cadre d'une restauration complète de BD
- **A ne pas utiliser si** d'autres journaux de transactions ou d'autres sauvegardes différentielles doivent être restaurées

# Lancement du processus de récupération (suite)

Utilisation de l'option **RECOVERY** (suite)

- SQL Server **annule** les transactions **non validées** dans le journal des transactions et **transmet** les transactions **validées**
- Permet **d'accéder** à la BD

# Lancement du processus de récupération (suite)

Utilisation de l'option **NORECOVERY**

- A utiliser** pour restaurer toutes les sauvegardes **sauf la dernière**
- SQL Server **n'annule pas** les transactions **non validées** dans le journal des transactions et **ne transmet pas** les transactions **validées**
- La BD ne peut pas être utilisée tant qu'elle n'est pas récupérée



# Définition des options de restauration

Option RESTORE	Description
<b>FILE</b>	<ul style="list-style-type: none"><li>- Restaure une sauvegarde particulière</li><li>- On doit indiquer un numéro de fichier</li></ul>
<b>RESTART</b>	Reprend une opération de récupération interrompue
<b>MOVE ... TO</b>	<ul style="list-style-type: none"><li>- Indique un emplacement de restauration des fichiers de sauvegarde</li><li>- A utiliser pour effectuer une restauration sur un <b>autre disque</b>, serveur ou ordinateur SQL Server de secours</li></ul>
<b>REPLACE</b>	Remplace une BD existante SQL Server n'effectue pas de contrôle de sécurité

# 7. Restauration de Base de Données

7.1 Processus de récupération de SQL Server

7.2 Préparation de la restauration d'une BD

7.3 Restauration de sauvegardes

**7.4 Restauration de BD à partir de différents  
types de sauvegardes**

7.5 Restauration de BD système endommagées

## 7.4 Restauration de BD à partir de différents types de sauvegardes

- Restauration à partir d'une sauvegarde complète de BD
- Restauration à partir d'une sauvegarde différentielle
- Restauration à partir d'une sauvegarde du journal des transactions
- Restauration à partir d'une sauvegarde de fichier ou de groupe de fichiers

# Restauration à partir d'une sauvegarde complète de BD

On utilise ce type de restauration dans les cas suivants :

- Le **disque** physique est **endommagé**
- **Toute la BD** est **endommagée**, détériorée ou supprimée
- Pour conserver une **copie** identique de la BD sur un **autre ordinateur** SQL Server

# Restauration à partir d'une sauvegarde complète de BD (suite)

- Démarrer le processus de récupération avec l'option **RECOVERY** si on ne dispose **d'aucun journal des transactions ni d'aucune sauvegarde différentielle**
- Retarder le processus de récupération avec l'option **NORECOVERY** si on dispose de **sauvegarde de journal des transactions ou de sauvegardes différentielles**

# Exemple

USE master

RESTORE DATABASE Northwind

FROM NwinBac

WITH FILE=2, RECOVERY

➔ Sauvegarde complète stockée dans le fichier de sauvegarde **NwinBac** et 2 sauvegardes ont été ajoutées à ce fichier

➔ La BD **Northwind** est entièrement remplacée par la 2<sup>ème</sup> sauvegarde figurant dans l'unité de sauvegarde **NwinBac**

➔ Enfin, le processus de récupération rétablit la cohérence de la BD (transmet les modifications validées et annule celles non validées)

# Restauration à partir d'une sauvegarde différentielle

- Restaure uniquement les parties de la BD qui ont été **modifiées depuis la dernière sauvegarde complète** de BD
- Rétablit la BD à son état initial au moment où la sauvegarde différentielle a été effectuée
- Dure **moins longtemps** que l'application d'une série de journaux de transactions

# Éléments à prendre en compte pour la restauration de sauvegardes différentielles

- Restaurer la sauvegarde complète de BD **avant** la sauvegarde différentielle
- Spécifier le **fichier de sauvegarde** contenant la sauvegarde différentielle
- Spécifier l'option **NORECOVERY** si d'autres journaux de transactions doivent être restaurés; sinon, utiliser l'option RECOVERY



# Exemple

USE master

**RESTORE DATABASE** Northwind

FROM NwinBacDiff

WITH **NORECOVERY**

➔ Restaure une sauvegarde différentielle **sans récupérer** la BD

➔ Le fichier **NwinBacDiff** contient une sauvegarde différentielle

➔ SQL Server vous permet de **restaurer des journaux** de transactions avant de rétablir la cohérence de la BD **Northwind** en spécifiant l'option **NORECOVERY**

# Restauration à partir d'une sauvegarde du journal des transactions

- Restaure les modifications apportées à la BD et qui sont enregistrées dans le journal des transactions
- Généralement utilisée pour permettre l'application des modifications apportées à la BD depuis la dernière sauvegarde complète ou différentielle
- Joue également un rôle essentiel lorsqu'il s'agit de récupérer une BD **jusqu'à un certain point dans le temps**

# Éléments à prendre en compte pour la restauration des journaux de transactions

- **Restaurer** la sauvegarde **complète** avant de restaurer les **journaux** de transactions
- **Restaurer** les sauvegardes des **journaux** de transactions créées **après** une sauvegarde **différentielle** pour préserver la cohérence des données
- Spécifier l'option **NORECOVERY** pour tous les journaux sauf le dernier
- SQL Server interrompt le processus de récupération jusqu'à ce que le dernier journal soit restauré

# Exemple

- Une sauvegarde complète de BD a été effectuée et enregistrée dans un fichier de sauvegarde
  - Deux sauvegardes du journal des transactions ont été créées dans un autre fichier de sauvegardes
- 3 opérations de restaurations** distinctes doivent être effectuées pour garantir la cohérence de la BD:

# Exemple (suite)

1-     *USE master*  
       *RESTORE DATABASE Northwind*  
       *FROM NwindBac*  
       *WITH NORECOVERY*

➔ **restauration à partir d'une sauvegarde complète** sans récupérer la BD

2-     *USE master*  
       *RESTORE LOG Northwind*  
       *FROM NwindBacLog*  
       *WITH FILE=1, STATS, NORECOVERY*

➔ **restauration du 1<sup>er</sup> journal des transactions** sans récupérer la BD en affichant la progression du processus de restauration

# Exemple (suite)

3- *USE master*

*RESTORE LOG Northwind*

*FROM NwindBacLog*

*WITH FILE=2, RECOVERY*

➔ restaure le 2<sup>ème</sup> journal des transactions, transmet les transactions validées et annule les transactions non validées

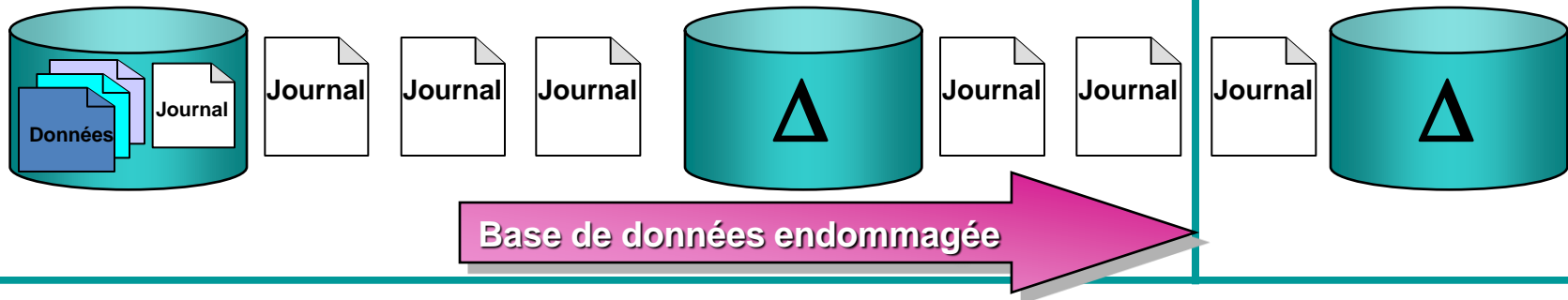
# Restauration à partir d'une sauvegarde du journal des transactions (suite)

## Sauvegardes de la base de données Northwind

Base de données complète

Différentielle

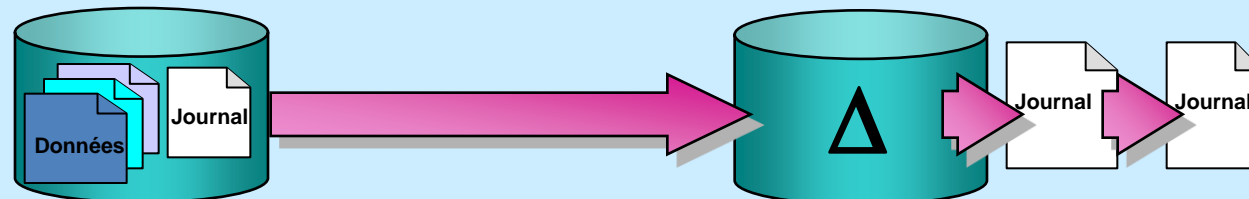
Différentielle



## Restauration de la base de données Northwind

Base de données complète

Différentielle



```
USE master
RESTORE LOG Northwind
FROM NwindBacLog
WITH FILE = 2, RECOVERY
```

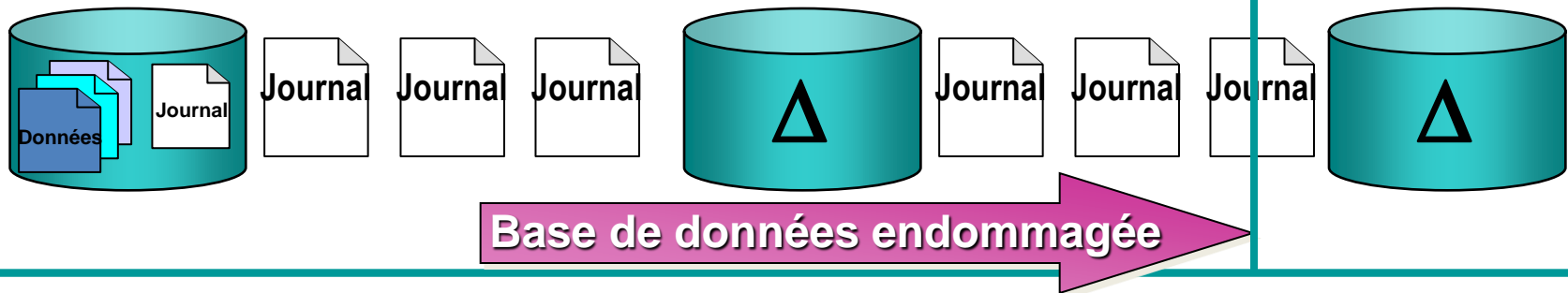
# Spécification d'un point dans le temps (exemple)

## Sauvegardes de la base de données Northwind

Base de données complète

Différentielle

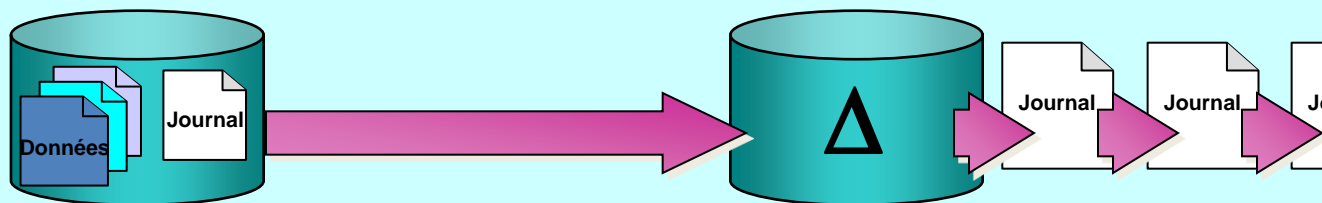
Differentielle



## Restauration de la base de données Northwind

Base de données complète

Différentielle





# Spécification d'un point dans le temps (exemple)

1-      *USE master*  
         *RESTORE DATABASE Northwind*  
         *FROM NwindBac*  
         *WITH NORECOVERY*

➔ restauration à partir d'une sauvegarde complète sans récupérer la BD

2-      *USE master*  
         *RESTORE LOG Northwind*  
         *FROM NwindBacLog*  
         *WITH FILE=1, NORECOVERY*

➔ restauration du 1<sup>er</sup> journal des transactions sans récupérer la BD

# Spécification d'un point dans le temps (exemple-suite)

```
3-      USE master  
  
      RESTORE LOG Northwind  
  
      FROM NwindBacLog  
  
      WITH FILE=2, RECOVERY, STOPAT='3 janvier 2000, 01:00'
```

➔ restaure le 2<sup>ème</sup> journal des transactions, applique les modifications apportées avant le 3 janvier 2000 à 01:00 et récupère la BD

# Restauration à partir d'une sauvegarde de fichier ou de groupe de fichiers

- Rappel de l'intérêt des sauvegardes sur fichiers et groupes de fichiers :
  - **Réduire le temps nécessaire de restauration** de BD très volumineuses
  - Récupérer des données lorsqu'un **fichier particulier a été endommagé** ou supprimé accidentellement
- SQL Server permet de restaurer un fichier de BD à partir d'une sauvegarde complète de BD ou d'une sauvegarde de fichier individuel

# Restauration à partir d'une sauvegarde de fichier ou de groupe de fichiers (suite)

- Appliquer tous les journaux de transactions **créés depuis la sauvegarde** du fichier afin de rétablir la cohérence du fichier ou du groupe de fichiers restauré par rapport au reste de la BD
- Restaurer les sauvegardes de groupes de **fichiers conjointement** si une table et ses index associés sont sur deux groupes de fichiers **différents**

# Syntaxe partielle

```
RESTORE DATABASE {BD | @var_nom_BD}  
<fichier_ou_groupe_fichiers> [...m]  
[FROM <unité_sauveragdre> [...n]]
```

Où <fichier\_ou\_groupe\_fichiers> représente:

{**FILE**=nom\_fichier\_logique |

**FILEGROUP**=nom\_groupe\_fichier\_logique}

# Exemple

- La BD utilise **3 fichiers** distincts : Nwind1, Nwind2 et Nwind3
- Le fichier de BD Nwind2 contient une seule table et ses index
- Il a été sauvegardé sur le fichier de sauvegarde Nwind2Bac
- Une sauvegarde du journal des transactions a été effectuée depuis la dernière sauvegarde de Nwind2
- Le support physique contenant **Nwind2** a été **endommagé**

# Exemple (suite)

**Etape 1** : on restaure la sauvegarde du fichier de BD Nwind2 **sans transmettre** les transactions les transactions validées ni annuler celles non validées

USE master

RESTORE DATABASE Northwind

FILE=Nwind2

WITH NORECOVERY

# Exemple (suite)

**Etape 2** : on restaure la sauvegarde du journal des transactions, on transmet les transactions validées et on annule celles non validées

USE master

RESTORE LOG Northwind

FILE=NwindBacLog

WITH RECOVERY



# 7. Restauration de Base de Données

7.1 Processus de récupération de SQL Server

7.2 Préparation de la restauration d'une BD

7.3 Restauration de sauvegardes

7.4 Restauration de BD à partir de différents types de sauvegardes

**7.5 Restauration de BD système endommagées**

## 7.5 Restauration de BD système endommagées

- **Obligatoire** si le support physique contenant les **BD système** est **endommagé**
- **Si on peut démarrer SQL Server**, on doit utiliser l'instruction **RESTORE DATABASE** ou SQL Server Enterprise Manager pour restaurer les **BD système**

## 7.5 Restauration de BD système endommagées (suite)

- Si la BD master est endommagée et on **ne peut pas démarrer SQL Server** alors :
  - Reconstruire les BD système à l'aide de l'utilitaire de ligne de commande **Rebuildm.exe** situé dans le dossier 80\Tools\Bin
  - Redémarrer le service SQL Server
  - Restaurer les sauvegardes des BD système : **master, msdb, model** à partir de sauvegarde

# Rattachement ou restauration de BD utilisateur

- Si la BD **master** a été restaurée à partir d'une sauvegarde valide, elle contient des références à chaque à chaque BD utilisateur. **Aucune autre action n'est requise**
- Si la BD **master** a été **reconstruite** et qu'aucune sauvegarde valide n'a été appliquée, alors :
  - **Soit restaurer les BD utilisateurs** à partir d'une sauvegarde
  - **Soit rattacher les fichiers** de BD utilisateur existants à la nouvelle BD master par le procédure stockée système **sp\_attach\_db** ou **sp\_attach\_single\_file\_db**

# Example

```
USE master
```






```
EXEC sp_attach_single_file_db
```

```
@dbname='Northwind',
```

```
@physname='Mssql\Data\Northwind.mdf'
```

➔ Rattachement de la BD Northwind à la BD master

# Conseils pratiques

-  Rassemblez des informations sur les sauvegardes que vous envisagez de restaurer
-  Utilisez l'option NORECOVERY si vous avez d'autres sauvegardes à restaurer
-  Utilisez l'option RECOVERY sur la dernière sauvegarde pour rétablir la cohérence de la base de données
-  Ajouter une marque dans le journal afin que la base de données soit restaurée jusqu'à un point antérieur au début de l'opération
-  Testez régulièrement vos fichiers de sauvegarde à l'aide de l'instruction RESTORE VERIFYONLY