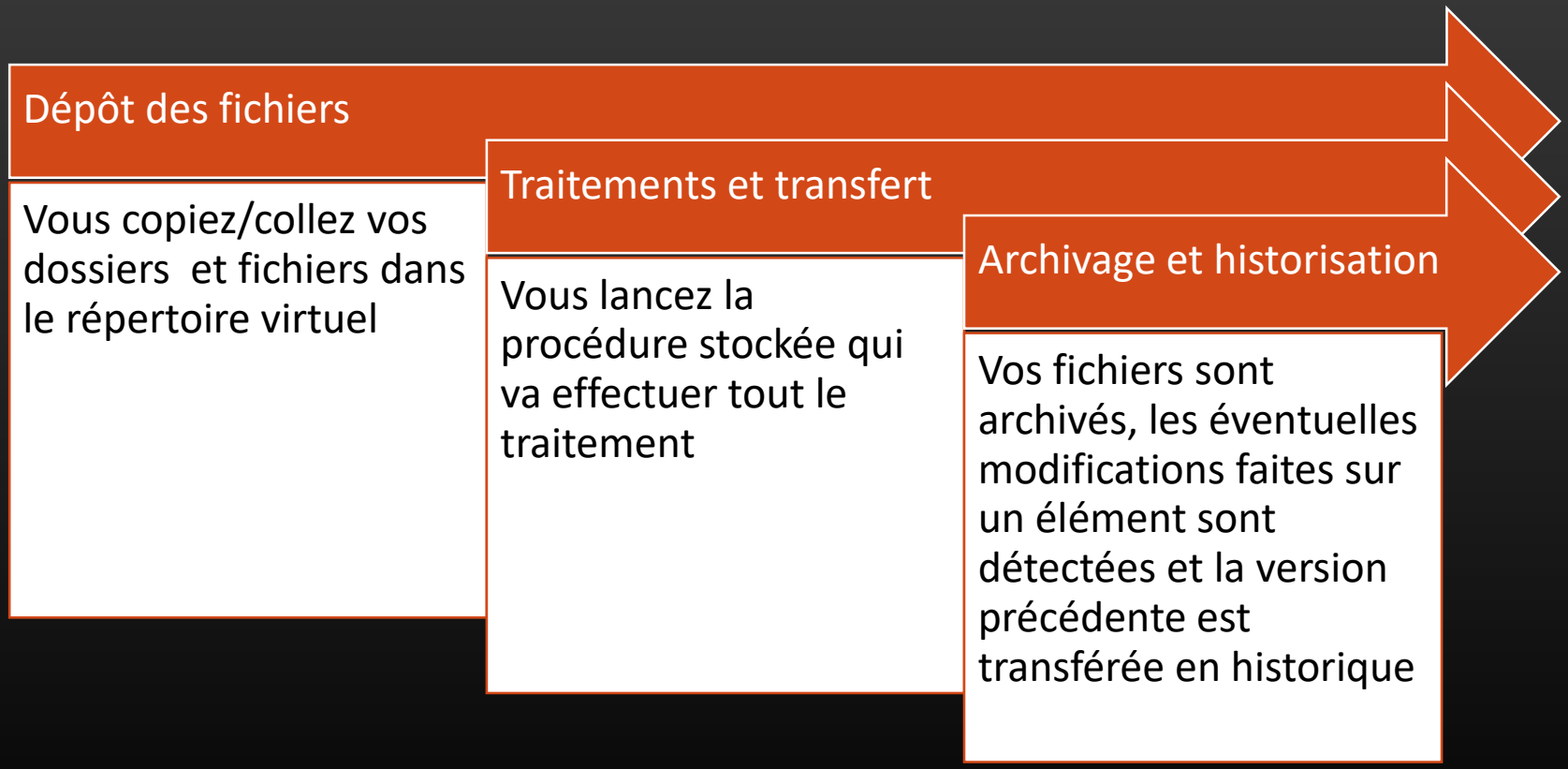
DepotFox

Archivage et historique sécurisé

DepotFox

- DepotFox est un utilitaire d'archivage et d'historique sécurisé pour tous les fichiers, intégrant la gestion des tuples.
- DepotFox crée un répertoire réseau virtuel et vous copiez/collez vos dossiers dans ce répertoire.
- DepotFox transfère les fichiers depuis ce répertoire vers une table SQL Server avec les informations d'arborescence.
- À chaque transfert vers les tables d'archivage, DepotFox détecte les éventuelles modifications faites sur un élément et dans ce cas archive automatiquement la version précédente.

Résumé des différentes étapes



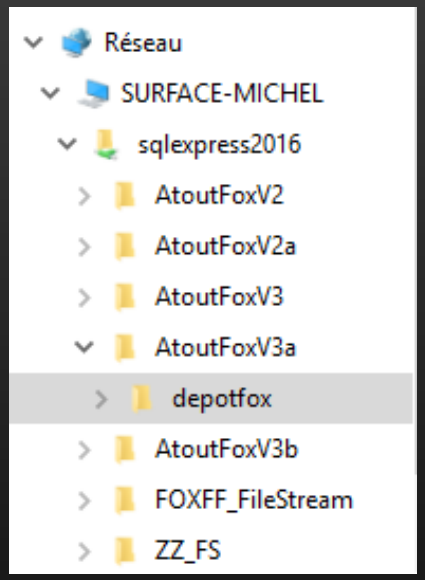
Depotfox est une application client-serveur

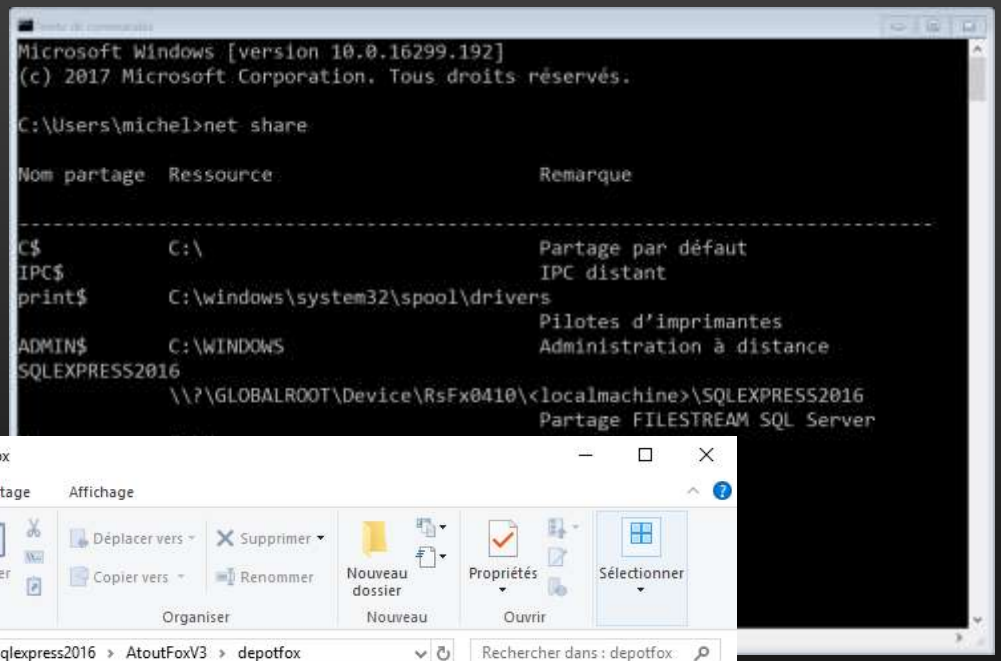
- Tout le code « métier » est codé sur SQL Server
- Il y a un module client en VFP
- Un prg VFP permet d'installer le module serveur

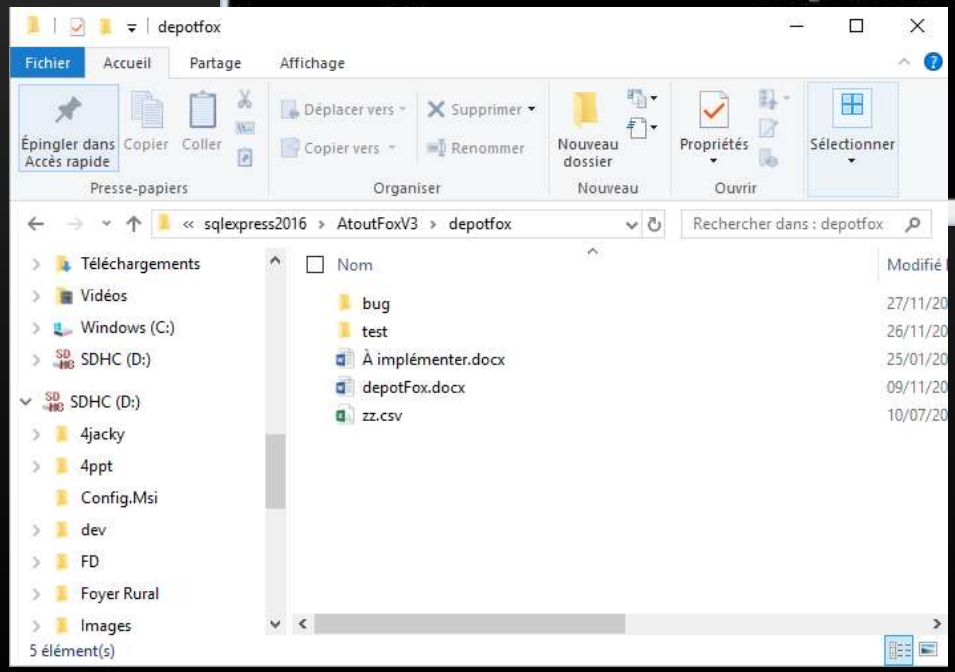
Le code serveur Depotfox repose sur 4 structures SQL

- Le répertoire de dépôt (une FileTable)
- Les procédures de transfert
- Les tables d’historique
- Le trigger d’évenement Database

Le répertoire de dépot







Le répertoire de dépôt…

est un répertoire virtuel.

C’est en réalité une table SQL Server dont la structure est prédéfinie par SQL,

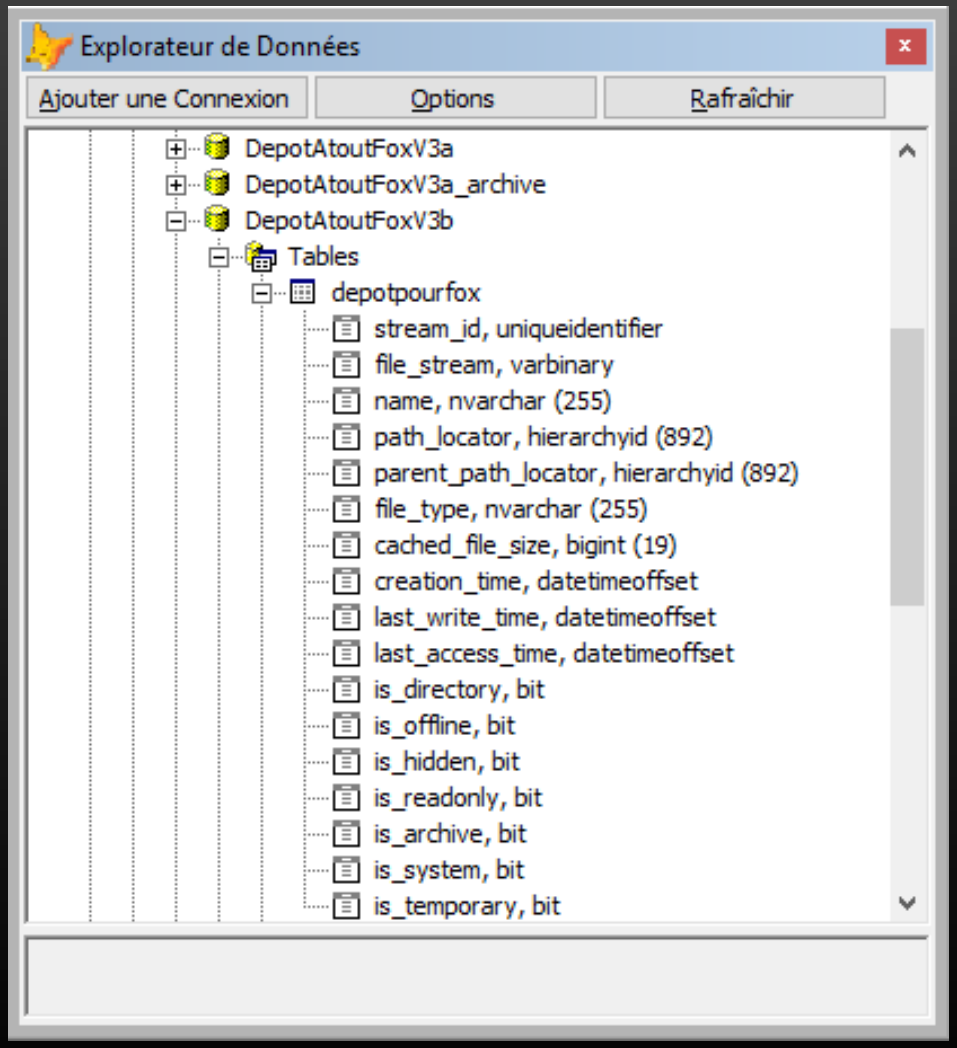
une **FileTable**

Le répertoire de dépôt…

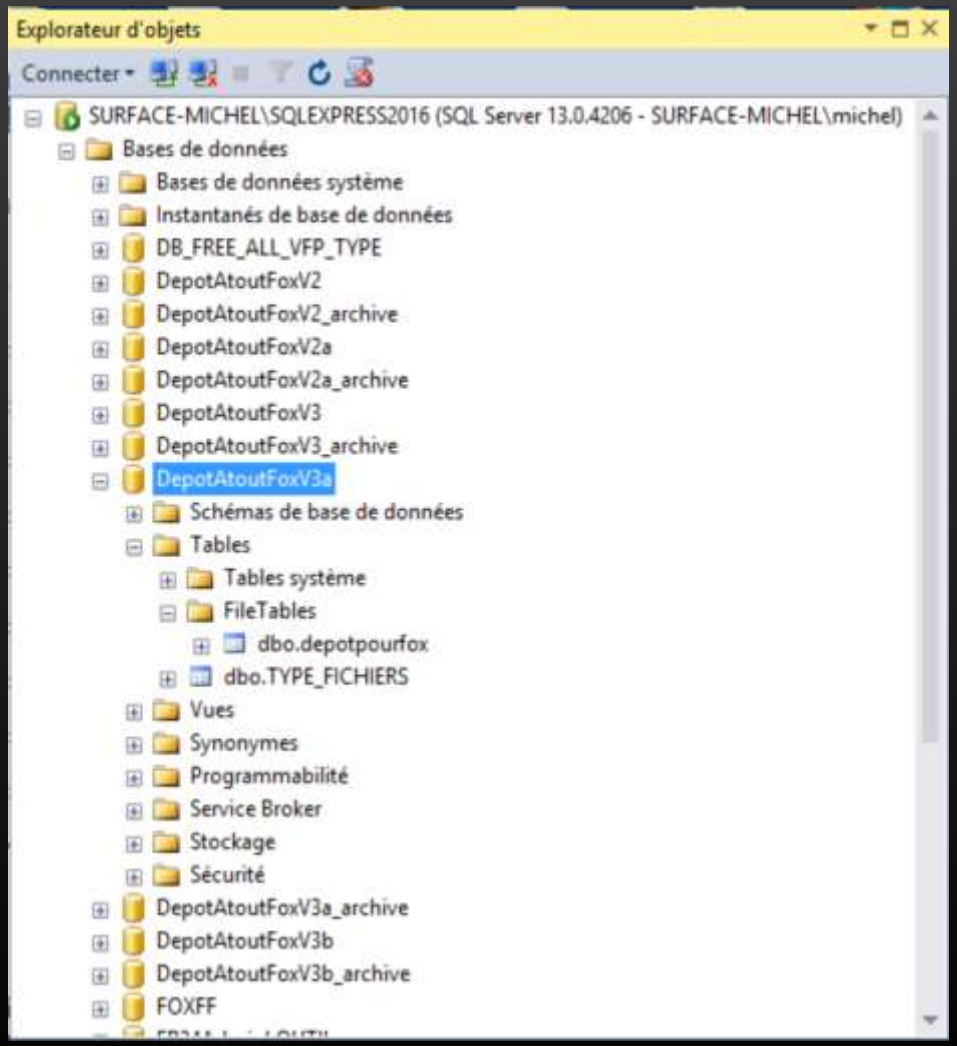
une **FileTable:**

```
CREATE TABLE [dbo].[depotpourfox] AS FILETABLE
   ON [PRIMARY]
   FILESTREAM_ON [FileStreamDepotAtoutFoxV3a]
   WITH
      (
         FILETABLE_DIRECTORY = N'depotfox',
         FILETABLE_COLLATE_FILENAME = French_CI_AS
      )
```

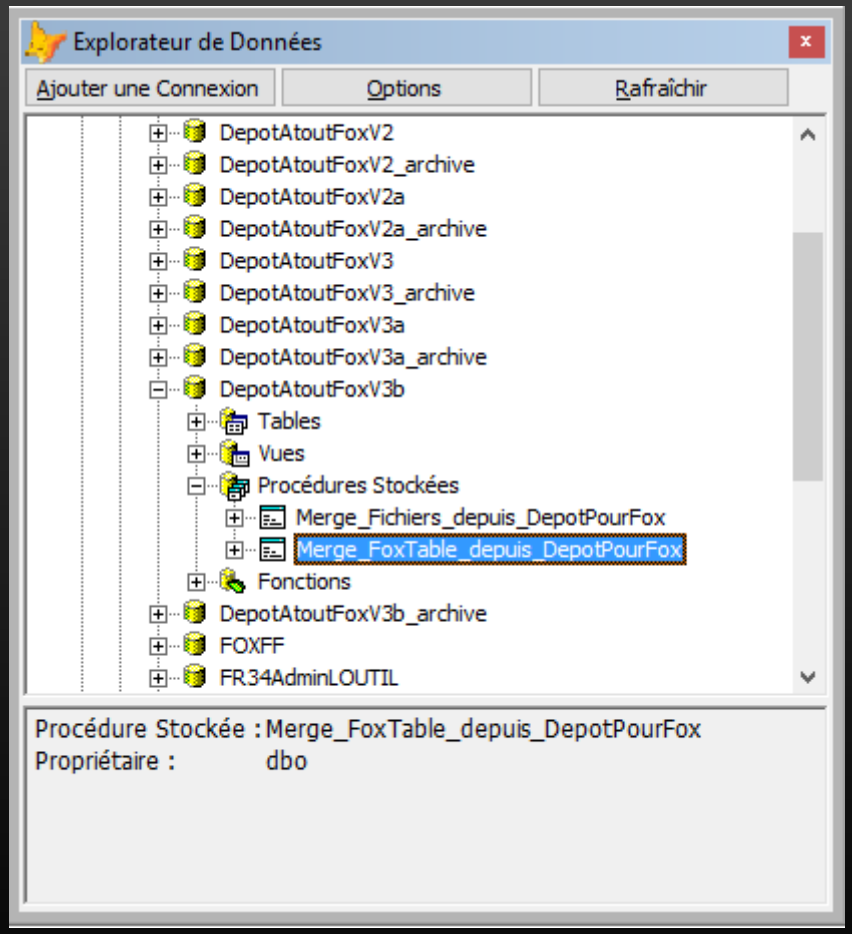
FileTable


Explorateur de Données
Ajouter une Connexion
Options
Rafraîchir
DepotAtoutFoxV3a
DepotAtoutFoxV3a_archive
DepotAtoutFoxV3b
Tables
depotpourfox
stream_id, uniqueidentifier
file_stream, varbinary
name, nvarchar (255)
path_locator, hierarchyid (892)
parent_path_locator, hierarchyid (892)
file_type, nvarchar (255)
cached_file_size, bigint (19)
creation_time, datetimeoffset
last_write_time, datetimeoffset
last_access_time, datetimeoffset
is_directory, bit
is_offline, bit
is_hidden, bit
is_readonly, bit
is_archive, bit
is_system, bit
is_temporary, bit

FileTable


Explorateur d'objets
Connecter
SURFACE-MICHEL\SQLEXPRESS2016 (SQL Server 13.0.4206 - SURFACE-MICHEL\michel)
Bases de données
Bases de données système
Instantanés de base de données
DB_FREE_ALL_VFP_TYPE
DepotAtoutFoxV2
DepotAtoutFoxV2_archive
DepotAtoutFoxV2a
DepotAtoutFoxV2a_archive
DepotAtoutFoxV3
DepotAtoutFoxV3_archive
DepotAtoutFoxV3a
Schémas de base de données
Tables
Tables système
FileTables
dbo.depotpourfox
dbo.TYPE_FICHIERS
Vues
Synonymes
Programmabilité
Service Broker
Stockage
Sécurité
DepotAtoutFoxV3a_archive
DepotAtoutFoxV3b
DepotAtoutFoxV3b_archive
FOXFF

Les procédures de transfert



Les procédures de transfert

Elles sélectionnent les fichiers concernés dans la FileTable (fichiers fox ou fichiers dont l'extension est mentionnée dans la table de paramétrage)

Elles assemblent ces fichiers en tuples

Elles envoient les fichiers vers les tables de la database d'archivage

Les procédures de transfert

- des CTE
- des PIVOT
- et un MERGE

CTE

- C’est une syntaxe SQL normalisée, qui permet de préparer des jeux de données intermédiaires (un peu comme des curseurs FoxPro)

CTE

```
-- extraction des données nécessaires
;
WITH
CTEprepare
    AS (
    SELECT
        REVERSE( SUBSTRING… ) ) AS Chemin,
        REVERSE(LEFT(REVERSE([Name]),…)-1  )) AS Fox_Type_Table,
        REVERSE(SUBSTRING(…)+1,255)) AS Fox_Nom_Table,
        File_Stream,
        File_Type
      FROM
          Dbo.Depotpourfox
      WHERE Is_Directory = 0),
```

CTE

```
,
CTEfinale
   AS (
      SELECT P.Chemin                    AS fichier_chemin,
        P.Nom_Complet                    AS fichier_nom,
        P.Contenu_Fichier                AS fichier_data,
        T.TYP_nom                        AS fichier_type ,
        S.Nom_Complet_Signature          AS signature_nom ,
        S.Contenu_Signature              AS signature_data ,
        S.Oksign
       FROM
          Cteprepare P
          JOIN Dbo.Type_Fichiers T
                ON P.File_Type = T.Typ_Ext AND T.Typ_Actif = 1
          LEFT OUTER JOIN Ctesignaturespossibles S
                ON P.Nom_Complet = S.Nom_Fichier AND P.Chemin  =S.Chemin
```

PIVOT

- C’est une syntaxe spécifique à SQL Server (non normalisée) qui permet de passer des lignes en colonnes

PIVOT

```
-- les fichiers de données
CTEdbf
    AS (
    SELECT Chemin,
          Fox_Type_Table,
          Fox_Nom_Table,
          [Dbf] AS "Fox_Data",
          [Cdx] AS "Fox_Index",
          [Fpt] AS "Fox_Memo"
        FROM
            CTEprepare
                        PIVOT (MAX(File_Stream) FOR File_Type IN
                                    ([Dbf],
                                     [Cdx],
                                     [Fpt]))
                              AS Pvt),
```

MERGE

- C’est une syntaxe SQL normalisée qui permet de réaliser un UPDATE ou un INSERT en une seule instruction
- MERGE destination USING source ON condition de jointure
  WHEN MATCHED instruction n°1
  WHEN NOT MATCHED BY TARGET instruction n°2

MERGE

```
-- on envoie les données résultantes vers la table d'historisation
        MERGE [DepotAtoutFoxV3b_archive].dbo.FoxTable AS T
        USING CTEfinale AS S
                ON ( T.Fox_nom_table = S.Fox_nom_table AND
                     T.fox_chemin = S.chemin
                    )
```

MERGE

Les données existent déjà en archive

```
WHEN MATCHED AND
(           (     T.Fox_data <> S.Fox_data
              OR (T.Fox_data IS NOT NULL AND S.Fox_data IS NULL)
              OR (T.Fox_data IS NULL AND S.Fox_data IS NOT NULL))
OR  (     T.Fox_index <> S.Fox_index
              OR (T.Fox_index IS NOT NULL AND S.Fox_index IS NULL)
              OR (T.Fox_index IS NULL AND S.Fox_index IS NOT NULL))
OR  (     T.Fox_memo <> S.Fox_memo
              OR (T.Fox_memo IS NOT NULL AND S.Fox_memo IS NULL)
              OR (T.Fox_memo IS NULL AND S.Fox_memo IS NOT NULL)))

THEN UPDATE SET T.fox_data = S.fox_data,
                T.fox_index = S.fox_index,
                T.fox_memo = S.fox_memo
```

On les met à jour

MERGE

Les données ne sont pas déjà en archive

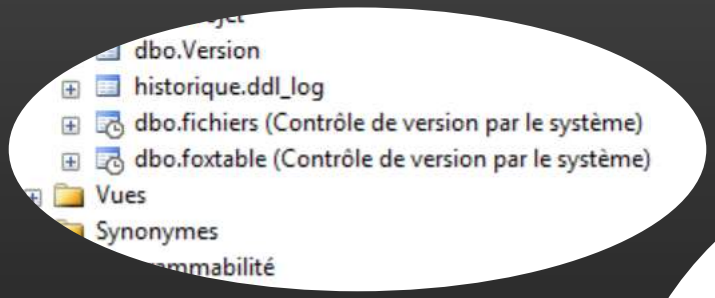
Alors on les ajoute

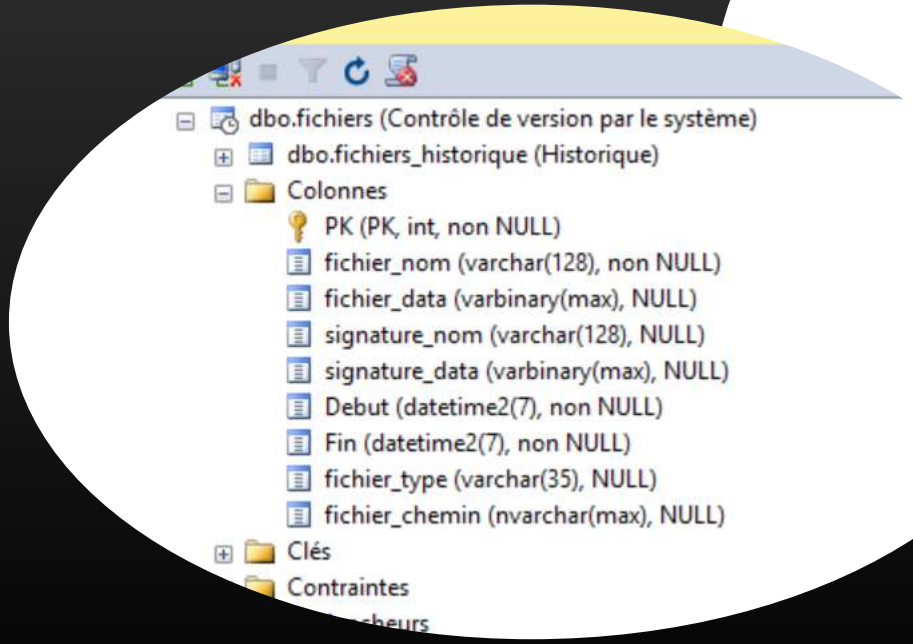
```
WHEN NOT MATCHED BY TARGET
        THEN
         INSERT (Fox_Chemin,
                              Fox_Typetable,
                              Fox_Nom_Table,
                              Fox_Data,
                              Fox_Index,
                              Fox_Memo)
          VALUES
                              (S.Chemin,
                               S.Fox_Typetable,
                               S.Fox_Nom_Table,
                               S.Fox_Data,
                               S.Fox_Index,
                               S.Fox_Memo
                              )
       OUTPUT $action ;
```

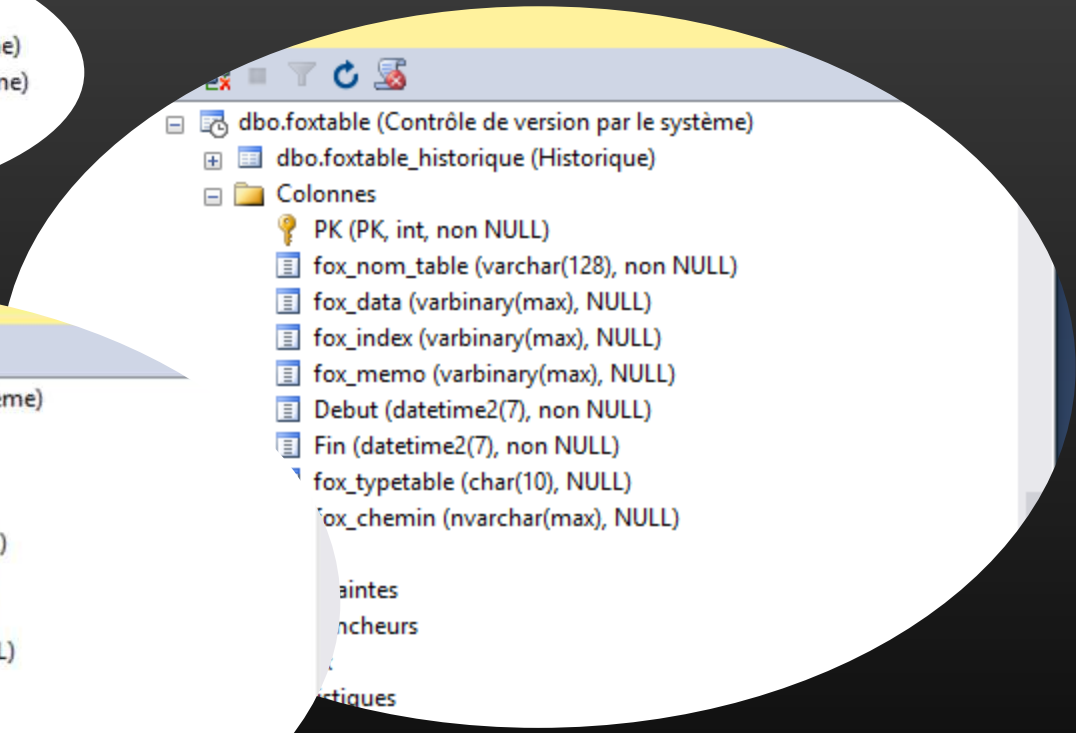
Les tables d’historique

- C’est une structure SQL normalisée
- Elles contiennent le contenu horodaté des fichiers
- Leur contenu est inaccessible depuis l’OS (pas de FileStream)
- Toute modification du contenu déclenche automatiquement la création d’une ligne d’archive dans la sous-table d’archivage
- La sous-table d’archive ne peut pas être modifiée ou supprimée

Les tables d'historique







Le trigger d'évenement Database

- C'est une structure SQL normalisée
- Le code du trigger est exécuté automatiquement quand un évenement ALTER TABLE ou DROP TABLE est demandé sur la base de données
- Toute action de « purge » de l'historique est donc enregistrée dans une table protégée

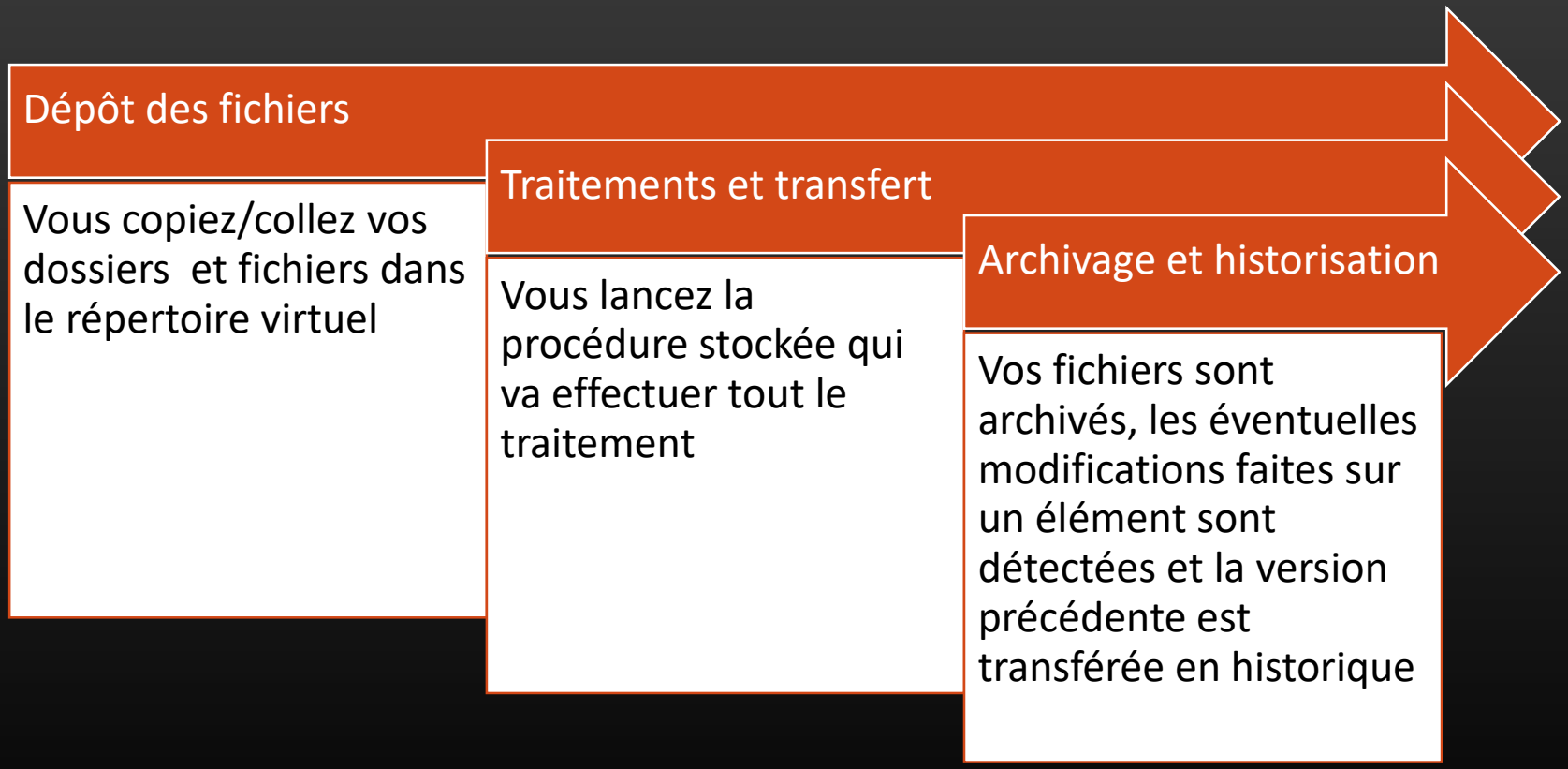
Le trigger d'évenement Database

```
CREATE TRIGGER [table_alter_drop_securite] ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
    DECLARE @Data XML;
    SET @Data = EVENTDATA();

    DECLARE @Evenement NVARCHAR(100),
            @Nomtable  NVARCHAR(100),
            @TSQL      NVARCHAR(2000),
            @Login     NVARCHAR(100),
            @origine   NVARCHAR(100);

    SET @Evenement  = @Data.value('(/EVENT_INSTANCE/EventType)[1]', 'nvarchar(100)');
    SET @Nomtable   = @Data.value('(/EVENT_INSTANCE/ObjectName)[1]', 'nvarchar(100)');
    SET @TSQL       = @Data.value('(/EVENT_INSTANCE/TSQLCommand)[1]', 'nvarchar(2000)');
    SET @Login      = @Data.value('(/EVENT_INSTANCE/LoginName)[1]', 'nvarchar(100)');
    SET @origine    = ORIGINAL_LOGIN()
```

Le trigger d’évenement Database

```
IF @Nomtable IN ('foxtable','fichiers')
        BEGIN
            INSERT INTO Historique.Ddl_Log
               (Posttime,
                Db_User,
                Event,
                TSQL
               )
            VALUES
               (GETDATE(),
                @origine + ' || '+   @Login + ' || ' + CONVERT(NVARCHAR(100), CURRENT_USER)  ,
                @Evenement,
                @TSQL
               );
        END;
```

Résumé des différentes étapes



DepotFox

- Permet également d'attribuer des numéros de version personnalisée à vos fichiers stockés, en fonction de l'horodatage
- Gère des projets-mots clés organisés en arborescence, auxquels vous pouvez rattacher vos fichiers

DepotFox, côté client…

- Un programme d’installation du serveur
- Une bibliothèque de classes

DepotFox, côté client…

- Un programme d'installation du serveur

installdepotfox.prg *

```
*!* Programme / Procédure : installDepotFox.prg
*!* Auteur : Michel LÉVY
*!* Date: 25/07/2017 14:51:21
*!*
*!* Description: création de la base de données SQL Server de dépot historisé des binaires fox
*!*
*!*Paramètres attendus :
* la chaine de connexion vers l'instance SQL Server
* le nom de la base de données que vous voulez créer dans cette instance SQL
* le nom du dossier FileStream pour l'accès Windows
* (c'est dans ce dossier que vous trouverez le sous-dossier "depotfox" dans lequel
*  vous copierez les fichiers fox à archiver pour historique)
*
```

Vue de document

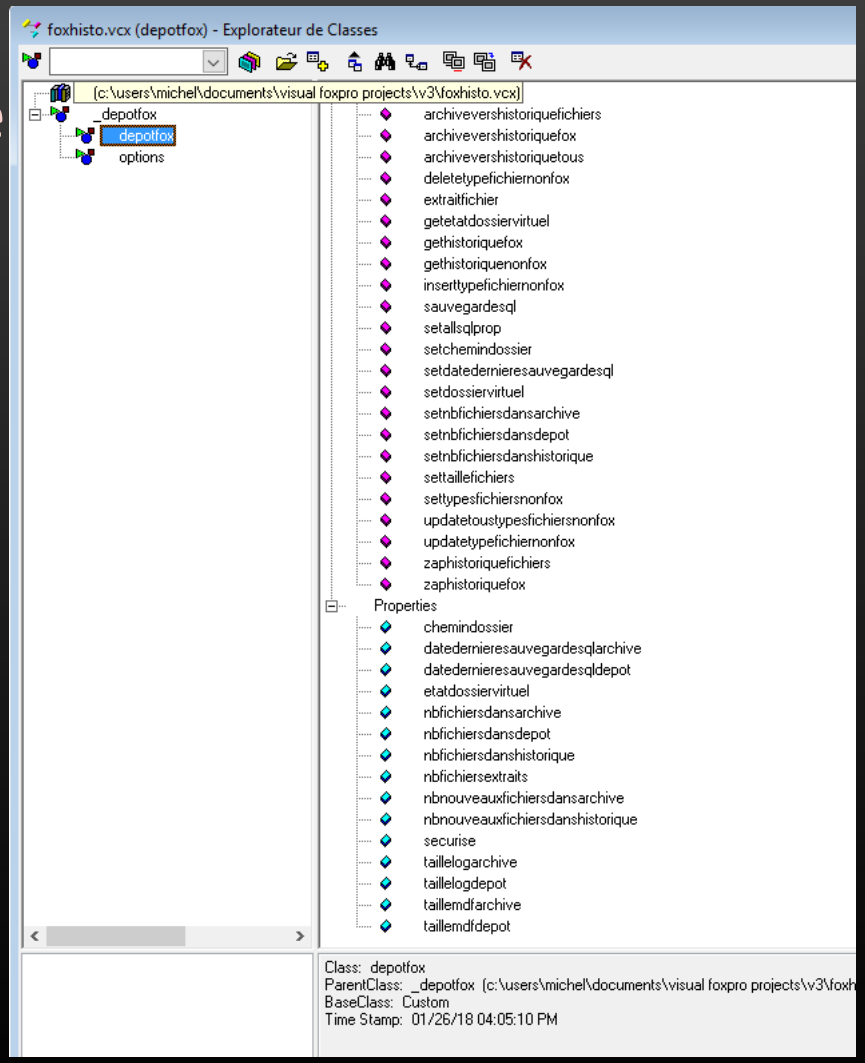
- CreateDatabase
- CreateFK
- CreateIndexes
- CreateProc
- CreateTables
- CreateViews
- GetFileStreamLevel
- GetNumVersion
- VerifPrerequis

Prog1 *

```
lcCnx="Driver={SQL Server Native Client 11.0};Server= SURFACE-MICHEL\SQLexpress2016;Database =master;Trusted_Connection=yes;"
lcDb="DepotAtoutFoxV3b"
lcPath="AtoutFoxV3b"
DO "installdepotfox.PRG" WITH m.lccnx,m.lcdb,m.lcpath
```

DepotFox, côté client…

- Une bibliothè



DepotFox

- gratuit, open source, libre de droits
- évolutif