

Agenda

- DTU -Introduction
- DTU - A detailed view
 - DTU-Walkthrough
 - Special Processing (Slots, Hierachies, Units)
- DTU - Additonal topics and frequent problems

Part 1: DTU - Introduction

Import via DTU

- One-way-transfer from external data into ETWeb
- Import is not integrated into ETWeb, but extra application (DTU.exe)
- Configuration and Logging via ETWeb-Frontend

Datafiles: CSV, XLS

Database

DTU.exe



Where to get the DTU.exe, sample configurations and further information?

- ETWeb Wiki
 - DTU and TP: Get Latest Version - Download various DTU-versions and get information about fixes and changes:
<https://devwikis.lumesse.com/ETWebWiki/dtu%20get%20latest%20version.ashx>
 - Search Wiki for DTU for introductions and various articles about DTU issues and versions.
- Partner Site

Configuration and Requirements

- DTU.exe requires .NET-Framework (3.5, some older versions 2.0)
- Configuration in DTU.exe.config (especially Database-Connection)
- DTU.exe needs access to ETWeb-Database
- Regular run is configured via ETWeb task processor or as schedules windows task.
- DTU.exe can run on a machine which is different from the SQL-server
- For Excel-Import, Microsoft.Jet.OLEDB.4.0 need to be accessible.

DTU Versions

- DTU versions are different from ETWeb versions
- Version history
 - DTU 2
 - Still in use with many older projects for 10.x nowadays.
 - DTU 3
 - Published with ETWeb 10.6, but can be also used with older versions of ETWeb from ETWeb 10.3 onwards (however unofficially).
 - Language-dependent lookups have been changed.
 - Added real SQL-transactions.
 - Compatibility-mode with DTU 2.
 - Only data changes will be imported.
 - DTU 3.5.x
 - performance issues for huge amounts of import data have been solved
 - Can be used for ETWeb 10.6 onwards.

Migration of DTU version

- Migration to a new main version (i.e. 2 -> 3 or 3 -> 3.5)
 - Apply of a DTU migration updater for changing to a new main version (Wiki page DTU and TP: Get Latest Version:
<https://devwikis.lumesse.com/ETWebWiki/DTU%20Get%20Latest%20Version.ashx>)
 - Check the differences of the DTU versions and if they affect your DTU configuration. Important issues with migrating 2 -> 3 or 2 -> 3.5 may be:
 - Transaction issues
 - Order of execution
 - Language dependence of lookups
 - Proper testing of the DTU with the new version is required.

Part 2: DTU - A detailed view

Import Steps

Pre-Staging

Staging



Raw Data Files (CSV,
XLS)



tDTU...



tDTU...._Typed
tDTU..._Untyped



Pre-Staging-Step

- Step 1: Raw data from the datasources is written into the untyped pre-staging table. Data is stored in NText-Fields.
- Step 2: Untyped data is casted into typed fields:
 - Validation of data-formats.
 - Optional validation of required fields.
 - Optional validation that a field value(s) is/are unique within the records.
- Configuration in DTU-section **Data sources**

Staging-Step

- Transfer the data from the pre-staging table to the staging table.
- Resolving foreign-key-fields via configured lookups.
- Applying further T-SQL-statements.
- Updating and Inserting the data into the ETWeb-Tables (also called „Basetable“ or „Ausgangstabelle“ in DTU) via the configured staging table-basetable mapping (puzzle piece).
- Configuration in DTU section **Import Transactions**

Namings of Pre-Staging Tables/Staging Tables

- Pre-Staging tables and Staging tables are generated due to the DTU-configuration
- Pre-Staging and Staging tables are named with prefix tDTU
- Untyped pre-staging tables: postfix `_untyped`
- Typed pre-staging tables: postfix `_typed`
- Examples:
 - DTU-Configuration: Pre-Staging-Table **Employee** will generate database tables
 - `tDTUEmployee_Untyped`
 - `tDTUEmployee_Typed`
 - DTU-Configuration: Staging-Table **Employee** will generate database tables
 - `tDTUEmployee`

Pre-Staging Tables/Staging Tables during DTU-processing

- Pre-Staging and Staging tables are created in the build-step of the DTU.
- Pre-Staging-Tables and Staging-Tables can be used in SQL-statements executed during the DTU-run. (take care with ID-fields, see later)
- It is common to configure the DTU to remove pre-staging and staging tables after import. For debugging and verification it is possible to keep the DTU-table-data after the import. (Setting: Empty staging tables after import)

- DTU-version 3 onwards: The content of pre-staging and staging tables are archived for the latest DTU runs. (Setting: Maximum number of logs)

Adding Pre-Staging Table

- Add new pre-staging table in Administration.
- Just name of pre-staging is booked here. The fields can just be configured when the pre-staging is linked to a data source.

Configuration of Pre-Staging

- Link raw data source to pre-staging table
- **BE CAREFUL!** The pre-staging table including the field definitions may be shared among various data sources.

Configuration of Pre-Staging fields

- Mapping of raw data fields to (typed) pre-staging fields
- **BE CAREFUL!** The pre-staging table including the field definitions may be shared among various data sources.

Adding of a staging table

- Add staging table in administration. The staging table can then be added to any DTU in your system.
- Just the staging table name can be configured here. The field mapping to basetables (=ETWeb tables) is configured within a DTU configuration.

Adding of a Basetable (=link to ETWeb table)

- A basetable (=Ausgangstabelle in German) is a link to an ETWeb table which should be a destination of the import.
- Dropdown provides all ETWeb tables
- **NOTE!** If the ETWeb-table is a 1:1-sub-table where entries are generated via a trigger, never forget the check 1:1 table.

Configure staging table fields (1)

- Updates/Inserts from staging table data into the basetable is defined via a mapping of staging table fields and base table fields.
- **NOTE!** The field names need to be the names of the fields in the basetable. The field types also refer to the basetable. **You cannot have additional fields in the staging table that do not refer to a basetable field!**

Configure staging table fields (2)

- **BE VERY CAREFUL!** A staging table can be shared among various DTUs. If you make changes to a staging table please check in advance if it has a side effect on other DTUs. To avoid side effects, you may need to have staging tables that are just used in your DTU.

Multiple purposes of staging table mappings

- Purpose 1: Import data
 - Definition of the fields that are transferred to the basetables
 - Definition of the rule how to identify imported records with records that may already exist in ETWeb and just need to be updated.
- Purpose 2: Defining lookup rules for filling foreign key fields
 - Definition of the rule how to lookup foreign key via imported fields.
- **Note!** A staging table can also be used just for one purpose.
- How do we identify imported records and fields with existing data in ETWeb?....**Let's talk about PKs and ECKs.**

ECK-Field (= External Combined Key)

- Imported data and data in the basetable is identified via one or more fields that are configured as ECK-field(s) (ECK = External Combined Key)
- Note that also multiple fields can build an ECK.
- If one record in the staging table and basetable matches due to ECK, there will be an UPDATE.
- If no corresponding record is found in the basetable, we have an INSERT.
- If there are multiple records in the staging table with the same ECK, just the first record will be imported. For DTU 3 or higher this behaviour can be change via configuration (`<add key="InvalidateDuplicateECK" value="true"/>` in DTU.exe.config).
- If one record of the staging table matches to multiple records in the basetable due to the ECK, just one record in the basetable will be updated and no error is reported (verified for DTU 3.5.3). **(The DTU may be too generous here. Take care!)**

PK-Field (= Primary Key)

- PK-field (PK = primary key) of the staging table corresponds to ID-field (or primary key field) in the basetables.
- Staging Table defines a mapping between a PK-field (ID in ETWeb) and the ECK-field which can be also used for ID-Lookups for foreign keys
- The PK-fields of the staging tables are filled during processing the import of the basetable if a record could be identified via ECK. **(Be Save! Avoid using PK-fields of staging tables in DTU-SQL-statements. In some DTU-versions (i.e. 3.5) the PK-field is still NULL when you SQL-statement is executed.)**

Lookup of foreign keys

- Staging tables define mapping of a PK (=primary key) to an ECK (external combined key) which can be used to lookups of foreign keys. In this example, you specify a lookup via the staging table **StatusEmp**.
- If we check “Auto add missing lookups” and have an ECK that does not exist in the basetable of StatusEmp, the ECK will be added to the basetable.

Lookup of foreign keys - Example

- Our previous example:
 - Staging tables: **Employee**
 - Foreign-Key-Field: **StatusEmpID**
 - Staging-table for lookup: **StatusEmp**
 - ECK: **Code**
 - PK: **StatusEmpID**
- DTU will generate two fields for the lookup in staging table **tDTUEmployee**:
 - **StatusEmpID**: will be set by the DTU via ECK-PK-lookup of staging-table **StatusEmp**.
 - **StatusEmpID_Code**: Additional lookup field. Need to be filled with ECK from the imported data.
- In general, the additional lookup-field has the name **[Fieldname]_[ECK-Fieldname]**. If we have a ECK with multiple fields, we have multiple lookup fields.
- **NOTE!** A missing foreign-key-lookup will just result into a warning. The record is still imported.

Cascading lookups - Example

- ECK can also contain a lookup field
- Example: Staging table Slot
 - PK: SlotID
 - ECK fields: PID, PosID
 - PosID is lookup field via staging table Position (PK: PosID, ECK: GlobalPosCode)
 - PID is lookup field via staging table Employee (PK: PID, ECK: GlobalPosCode)
 - If we use the staging table Slot for a lookup of a foreign key field (say ParentSlotID), we will have the following fields for lookup:
 - ParentSlotID_PID_GlobalEmpCode: need to be filled with the GlobalEmpCode
 - ParentSlotID_PosID_GlobalPosCode: need to be filled with the GlobalPosCode
 - ParentSlotID: is resolved by DTU.

Staging table - SysLID field

- Special field for staging table to indicate that the table contains multi-language content (often code tables)
- ECK is the Code plus the SysLID which is set due to the DTU-language setting.
- DTU 2: Fallbacks to the default language are not included in the lookup. The code need to exist in the DTU-language.
- DTU 3.x.x: Fallbacks to the default language are included in the lookup.
- **NOTE! Never change the language setting of a DTU that is in use already. Especially when you use the AutoAdd-feature for missing lookup values this can create inconsistent data in code tables.**

Staging table with fields from different basetables

- One staging-table can map to various basetables when they have a 1:1-relationship due to the PK-field plus SysLID-field.

Filling of staging tables

- The transfer of data from the pre-staging to the staging-table should be defined in the SQL-tab of the staging table for DTU 3.x.x.
- Use placeholder `{{StagingTable}}` which will be exchanged by the StagingTable on runtime.
- Use placeholder `{{SysLID}}`. `{{SysLID}}` will be filled due to the DTU-language setting on runtime.
- For DTU 2, the staging tables are filled via an SQL-step.

Process Order

- **DTU 3.x.x:** The steps within a transaction are really executed in a transaction with a commit or a rollback on error. In DTU 2, the transaction was not really transactions.
- **Most important to know and most confusing!** The SQL-steps in the tabs of a staging table are always executed first and before the transaction.

Process Order - Example

- **BE CAREFUL!** Look at transaction Employee: SQL-statements of the puzzle pieces StatusEmp, Employee_SAP and PersonManagementLevel are executed first and before all other steps of transaction “Employee”. This is not what you would expect from the ordering in the tree.

Fields for errors and warnings

- Additional fields of pre-staging or staging table for errors and warnings:
 - DTUIsGarbage: indicates that a record has an error and is not imported.
 - DTUGarbageText: error or warning message
- Fields DTUIsGarbage, DTUGarbageText can set and filled via SQL-statements in the DTU, however you should be aware of the process order in order to set the DTUIsGarbage-Flags early enough.

Validation steps

- Available for DTU 3.x.x
- Compose a validation SQL-statement out of the staging table. The resulting records will set as garbage (DTUlsGarbage=1) if the checkbox “Mark these rows as invalid” is ticked.
- Row message will turn out to be the DTUGarbageText.
- Validation step need to be before the staging step (=puzzle piece).

Slot Assignment via DTU

Employee - Position - Assignment (Slot)

- Constraints:
 - Each employee/position is assigned to one or more slots.
 - Each employee has exactly one primary position within the positions he/she is assigned to.
 - Each position has exactly one primary employee within the position's incumbents.
- Special position/employee for vacant positions/unassigned employees:
 - PosID = -1: artificial position "Unassigned"
 - PID = -1: artificial employee "Vacant"
- Triggers of tSlot preserve the constraints

DTU-SlotHandling

- Check “Process as: Slot table”: The import of the staging table is done by a special SP which is pzDTUSlotHandling per default.
- pzDTUSlotHandling additionally write internal work history entries for unassignments if “Create Internal Work History record” is checked.
- Triggers of tSlot are disabled during the operation of pzDTUSlotHandling. The SlotHandling cares about the constraints (primary flags) of tSlot-data.

DTU-SlotHandling: pzDTUSlotHandling

- pzDTUSlotHandling assumes a staging table tDTUSlot with the following fields which need to be filled from the pre-staging table.
 - **PID:** configured as a lookup field on staging table Employee (with generated ECK-lookup-field PID_GlobalEmpCode for the standard DTU).
 - **PosID:** configured as a lookup field on staging table Position (with generated ECK-lookup-field PosID_GlobalPosCode).
 - **IsPrimaryEmp:** employee is primary incumbent of the position.
 - **IsPrimaryPos:** position is primary position of the employee.
 - **DTUUnassign:** if set to 1, the employee is unassigned from old positions if he/she is assigned to a new position.
 - **DTUHierID:** if set to -1 (=direct hierarchy), the supervisor slot of the direct hierarchy is set corresponding to the primary incumbent of the supervisor position.

pzDTUSlotHandling: What happens?

- Imports slots from tDTUSlot into tSlot.
- Unassign employee from old assignments that are not in the import if DTUUnassign = 1 is set.
 - Write entry in internal work history if „Create Internal Work History Records“ is checked.
- Adjust the flags IsPrimaryEmp and IsPrimaryPos due to the constraints (one primary employee per position, one primary position per employee).
 - If there is a conflict due to multiple primary employees/positions from the imported data, the latest slot gets the primary flag.
- Synchronize the slot-slot-hierarchy due to the pos-pos-hierarchy for imported slots. The supervisor slot is the primary incumbent of the supervisor position (see later).

DTU-SlotHandling for DTU 2.0 and DTU 3.x

- **DTU 2.0**

- DTU-SlotHandling-SP is always pzDTUSlotHandling.

- **DTU 3.x**

- Default DTU-SlotHandling-SP is pzDTUSlotHandling, but other SlotHandling-SPs can be configured in DTU.exe.config:

- `<add key="SlotHandlingSP" value="pzMyDTUSlotHandling"/>`

- Delievered DTU-SlotHandling-SPs for the samples of DTU3.x:

- pzDTUOnDemand_SlotHandling: uses staging table tDTUOnDemand_Slot instead of tDTUSlot.

- pzDTUStandard_SlotHandling: copy of default pzDTUSlotHandling to be used with the standard DTU-sample.

Hierarchy import via DTU

Hierarchies in ETWeb

- ETWeb allows multiple hierarchies. In all hierarchy tables, we have a field PosHierID to distinguish different hierarchies.
- Default hierarchies:
 - Direct hierarchy: PosHierID = -1
 - Functional hierarchy: PosHierID = -2

Pos-Pos- vs. Slot-Slot-Hierarchy - Functional Concept

- **Pos-Pos-Hierarchy:**

- Each position has one (or none) supervisor position as a parent position.
- The supervisor of a position's incumbent is the primary incumbent of the supervisor position.
- For most companies, a pos-pos-hierarchy is sufficient and the easiest to maintain.
- Only one incumbent per position means always Pos-Pos.

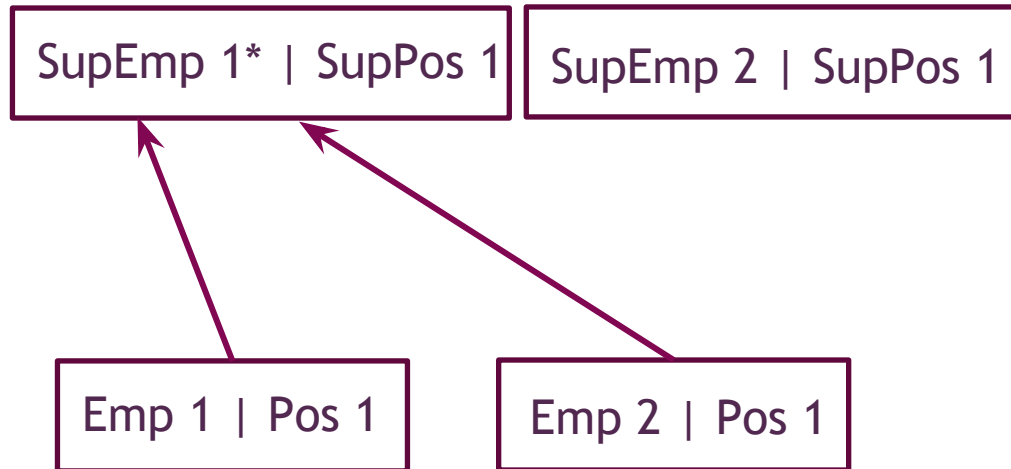
- **Slot-Slot-Hierarchy:**

- Each slot has one (or none) supervisor slot as a parent slot.
- The slot-slot-hierarchy is more flexible. The incumbents of one position can have different supervisors on different positions. It is not required that the supervisor is always the primary incumbent of a position.
- The slot-slot-hierarchy is far more complex and difficult to maintain. If you assign a person to a position, the supervisor and supervisor position needs to be assigned manually.

- **IMPORTANT! A consultant must clarify which hierarchy should be used. It is not possible to mix up the hierarchy concepts in one ETWeb.**

Examples: Pos-Pos vs. Slot-Slot

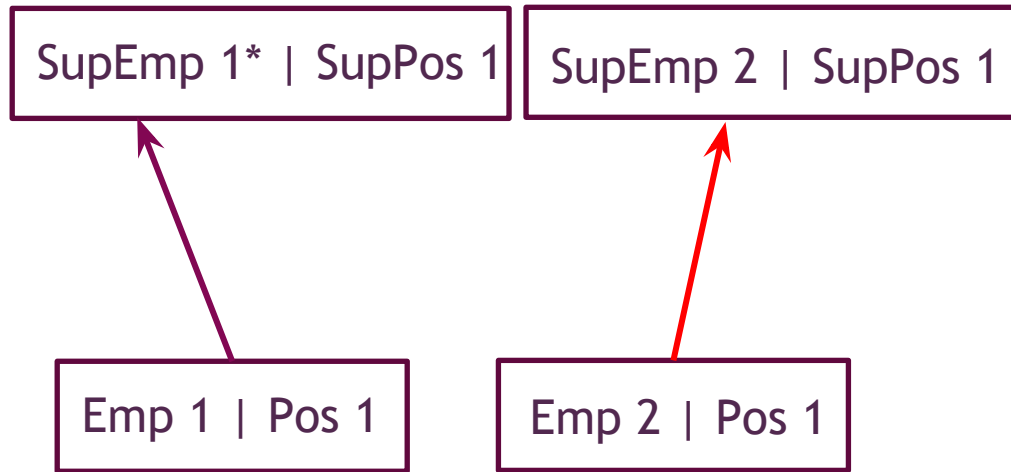
* Primary Incumbent of SupPos 1



OK: Pos-Pos fulfilled

Examples: Pos-Pos vs. Slot-Slot

*Primary Incumbent of SupPos 1

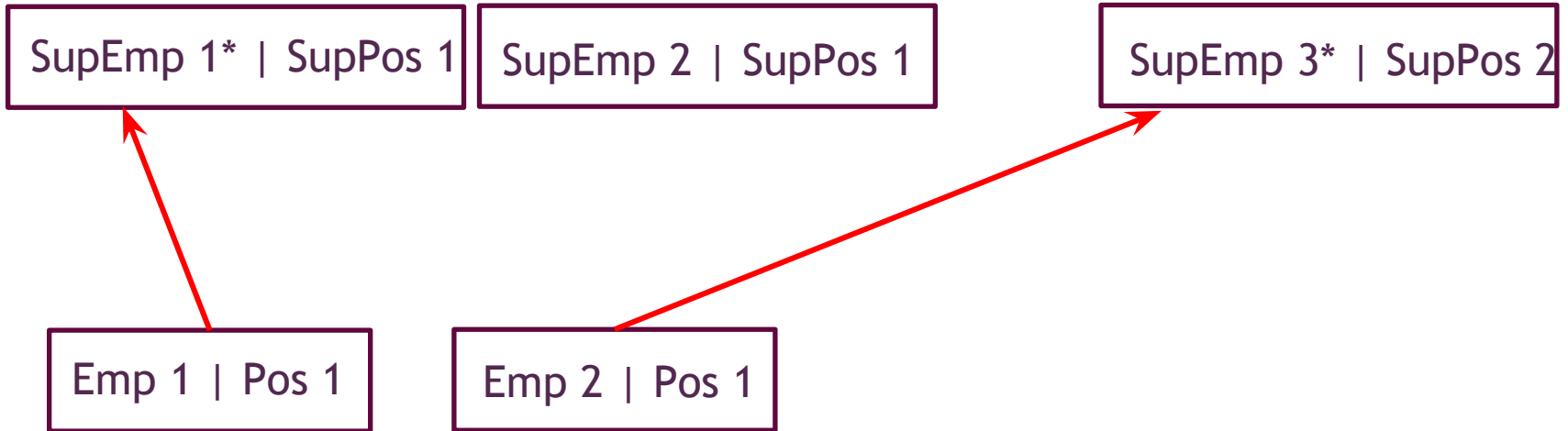


Pos-Pos not fulfilled: Supervisor of Emp 2 on Pos1 is not primary incumbent.

Examples: Pos-Pos vs. Slot-Slot

*Primary Incumbent of SupPos 1

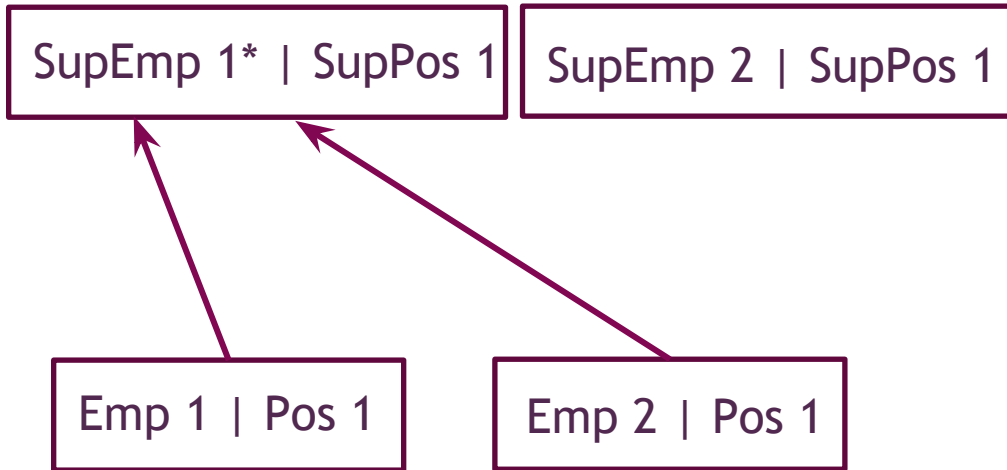
*Primary Incumbent of SupPos 2



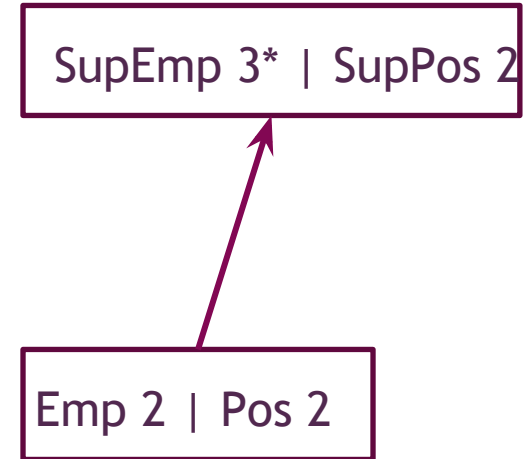
Pos-Pos not fulfilled: Pos 1 has not a unique supervisor position.

Examples: Pos-Pos vs. Slot-Slot

*Primary Incumbent of SupPos 1



*Primary Incumbent of SupPos 2



OK: Pos-Pos fulfilled. Note that an employee can have more than one supervisor (Emp 2). For each position a different one.

Pos-Pos- vs. Slot-Slot-Hierarchy - Frontends

- Show/hide the right frontends to maintain hierarchies:
 - **Pos-Pos-Hierarchy:**
 - PosSupervisor.asp
 - PosSubordinate.asp
 - **Slot-Slot-Hierarchy:**
 - SlotSupervisor.asp
 - SlotSubordinate.asp
- **NOTE!** In ETWeb 10.x, the ASP-pages for maintaining the pos-pos-hierarchy are available in the position controller. In ETWeb 11.x, the pages has been move to the HR-Explorer which is a slot controller and are activated per default!

Pos-Pos- vs. Slot-Slot-Hierarchy - Tables

- Two hierarchy tables exist in ETWeb:
 - **tPosRelation:** PosID, ParentPosID, PosHierID
 - **tSlotPosRelation:** SlotID, ParentSlotID, PosHierID
- **Pos-Pos-hierarchy:**
 - **Both table are used.** If tPosRelation is updated or positions are assigned/unassigned, the entries in tSlotPosRelation are kept synchronized due to the pos-pos-relation. (Supervisor is the primary incumbent of the parent position.)
- **Slot-Slot-hierarchy:**
 - **Just tSlotPosRelation is used.** tPosRelation will be empty.
- **NOTE!** If you need to retrieve a supervisor, do not use tPosRelation, but tSlotPosRelation as this will allow to extend pos-pos to slot-slot later on.

DTU-Import of hierarchies

- The DTU samples (Standard, OnDemand) import pos-pos-hierarchies.
- Pos-Pos-Hierarchy: Check “Position relation table” on the PosRelation-staging step:
 - The import and special processing is then done a PosRelation-handling-SP **pzDTUPosRelation** (default). **Do not forget this check as the hierarchy data will not be processed properly otherwise.**
- Imports of slot-slot-hierarchies requires extensions and modifications of the standard DTU.

pzDTUPosRelation: What happens?

- Imports from tDTUPosRelation into tPosRelation.
- Synchronizes tSlotPosRelation with tPosRelation due to pos-pos (supervisor is primary incumbent of parent position in tPosRelation). However, existing relations in tSlotPosRelation that deviate from the pos-pos are left untouched.
- Required fields in tDTUPosRelation:
 - PosID: configured as a lookup field on staging table Position(with generated ECK-lookup-field PosID_GlobalPosCode for the standard DTU).
 - ParentPosID: configured as a lookup field on staging table Position(with generated ECK-lookup-field ParentPosID_GlobalPosCode for the standard DTU).
 - PosHierID: hierarchy (direct or functional usually)
 - PeerOrder
 - Comment

SPs to rebuild tSlotPosRelation due to Pos-Pos-hierarchy

- It is also possible to synchronize tSlotPosRelation with tPosRelation due to pos-pos by applying the following SPs in SQL-steps within the DTU:
 - **pSlotPosRelDTU_Rebuild:** synchronize the entries in tSlotPosRelation for the imported records in staging table tDTUPosRelation.
 - **pSlotPosRel_Rebuild:** Complete synchronization of tSlotPosRelation with tPosRelation due to Pos-Pos.

DTU-PosRelation-Handling for DTU 2.0 and DTU 3.x

- **DTU 2.0**

- DTU-PosRelation-SP is always pzDTUPosRelation.

- **DTU 3.x**

- Default DTU-SlotHandling-SP is pzDTUPosRelation, but other PosRelation-SPs can be configured in DTU.exe.config:

- `<add key="PosRelationSP" value="pzMyDTUPosRelation"/>`

- Delivered DTU-PosRelation-SPs for the samples of DTU3.x:

- pzDTUOnDemand_PosRelation: uses staging table tDTUOnDemand_PosRelation instead of tDTUPosRelation
- pzDTUStandard_SlotHandling: copy of default pzDTUSlotHandling to be used with the standard DTU-sample
- pzDTUPosRelation3: improved version of pzDTUPosRelation; obsolete transaction has been removed.

OrgUnit-Assignment via DTU

DTU-Import of Position-OrgUnit-Assignment

- Check „Unit Assignment table“ on the staging table for Position-Unit-Assignment which imports into tPosUnitTypeSlot.
- Triggers on tPosUnitTypeSlot and tUnit/tUnitText need to be disabled. The special unit-processing of the DTU replaces the functionality of the triggers which are synchronizing txPosUnit with tPosUnitTypeSlot.

Lookup-Fields for Units

- The unit fields of the staging table for Pos-Unit-Assignment usually have the name **UnitX** (X= Unit Type Number) and are filled with names of units. (The reference language for the names is the configured language of the DTU.)
- Select the unit type of the unit from the dropdown and check „Process as unit assignment“
- If you import an assignment to a sub-unit, you also need to have the assignments to the parent units in the same staging table.
- Missing units will be added if „Auto add missing unit“ is checked.

Summary - Part 2

- Pre-Staging-Step
- Staging-Step
- PK/ECK for identifying records
- Loopup-fields
- SysLID-field
- Process order
- Validations, Errors, Warnings, Logging
- Special Processing
 - SlotHandling
 - PosRelation-Handling
 - Assignment of organisation units to positions

Part 3 -DTU - Additional Topics And frequent problems

Running DTU and DTU steps

- Typical execution command for DTU:

```
DTU.EXE /DTUCfgID=1 /build /import /execute
```

- Parameters:

- `/DTUCfgID=`: ID of the DTU to be executed.
- `/build`: Staging and Pre-Staging tables are created.
- `/import`: Pre-Staging-Tables are populated from data sources.
- `/execute`: Import transactions of the DTU are executed.
(Transferring data to staging tables, import of staging tables, additional SQL-steps, validations)
- It is possible to run just single steps for analyzing purposes.

Connection with ETWeb database

- Connection is specified in DTU.exe.config:

- ```
<add key="Connection" value="Initial Catalog=ETWeb;Data Source=DE-2K3DB-EXT05\CON3_2K8R2;Integrated Security=SSPI;Application Name='ETWeb';/>
```

(SSPI: Security Support Provider Interface)

- Connection with Integrated Security:

- Windows user who executes the DTU.exe needs access to ETWeb-database.

- Connection with SQL-server security

- NOTE! For old versions (DTU 2), UserID and Password are not specified in the connection string, but added as run parameters: /uid, /pwd

```
DTU.EXE /DTUCfgID=1 /build /import /execute /uid=sa /pwd=subscribe
```

## BE CAREFUL: Allow NULL overwrite

- **Take care with the setting Allow NULL overwrite!**  
This setting can be useful, but if you miss to populate the field in the staging table, the data field will be overridden by NULL!
- **RULE:** If you do not populate a field with data, it is important to deactivate the field for the import. **DO NOT FORGET AND CHECK PROPERLY!**

# BE CAREFUL: Auto add lookup values and language setting

- May be useful and time-saving as it is not required to add the values of code-tables manually. Errors due to missing code values can be avoided.
- **Take care to adjust the language setting properly!** You will arrive at multiple code table entries with the same functional meaning otherwise.
- You can just import in **one** language if using this feature. Never simply change the language setting of a DTU that is already running even. If a consultant may ask for it, explain the problems that this will cause.

# BE CAREFUL: Multiple use of pre-staging/staging tables

- Take care that pre-staging tables or staging tables can be used multiple times.
- If you want to change pre-staging tables or stagings tables just for your DTU, you need to create a copy. Copy-SQLs are available via Wiki:
  - <https://devwikis.lumesse.com/ETWebWiki/copy%20a%20staging%20table.ashx>
  - <https://devwikis.lumesse.com/ETWebWiki/copy%20a%20pre-staging%20table.ashx>
- You need to make sure that all SQL-statements in the DTU refer to the copied table. Also think about the Slot-Handler-SP or PosRelation-Handler-SP.

# Take care about transactions and processing order

- Note that DTU-transactions-steps in DTU 3.x are executed within a SQL-transactions. If an error occurs, the whole steps embedded in a DTU-transaction are rolled back. (In version 2.0, SQL-transactions was not implemented.)
- Note that the SQL-statements in the tab of the staging tables (puzzle pieces) are run **before** the DTU-transaction and are not included in the DTU-transaction.
- Take care if you call a stored procedure. If the stored procedure includes a transaction which is rolled back due to an error, the whole DTU-transaction will be rolled back.
- Bad example from the real world: A stored procedure call is embedded into BEGIN TRY....END TRY with empty CATCH.
  - An error occurred in the stored procedure. Due to BEGIN TRY ... END TRY the error did not appear in the DTU-log.
  - The whole DTU-transactions-step was rolled back due to the rollback in the stored procedure. Later it turned out that the data was incorrect.



## BE AWARE: Required field setting in pre-staging table

- Required-setting for pre-staging-fields has two meanings at once:
  - Field need to occur in the data source.
  - Field need to be filled.
- **NOTE!** If a pre-staging-field is not set as required and the column is missing in the import file, the field will be generated but left empty. Just a warning in the log occurs.
- Check log warnings about missing and additional fields carefully and regularly and advice the consultant to forward to the customer.

# Update DTU-Audit when changing data via SQL-steps

- DTU-Audit in Import Summary
  - Displays the numbers of records in the pre-staging tables/staging tables which are imported with the number of records with warnings/errors.
  - Displays the number of imported/updated records in basetables.
- All staging-records which are set as garbage before the staging step are included in the error/warning count automatically.
- DTU-Audit is updated just by the Built-In-DTU-steps (PreStaging-Step, Staging-Step, Validation-step). If you apply changes on data in SQL-steps which may affect the counts in the audit, you need to consider to update the DTU-audit via SQL yourself:
  - tzDTUAuditPreStagingData
  - tzDTUAuditStagingData
  - tzDTUAuditTableData

# Using staging tables just for ECK-PK-lookups

- Staging tables can be used for ECK-PK-Lookups without filling them with data. These staging tables just use the existing data in ETWeb for ECK-PK-lookups.
  - Pure lookup-staging tables do not occur as staging-step in the DTU-configuration or the staging-step should not be marked as active.
  - A warning occurs that the staging table is used for lookup, but is not imported.
  - If you want to check the definition of a staging table which is just used for ECK-PK-lookup, add the staging step to your DTU and mark the step as inactive.
- Take care when changing the definition of staging tables for ECK-PK-lookups as these may be used by multiple DTUs.

# Finalizing step of the DTU: pzDTUFinalize

- **pzDTUFinalize** is always executed at the end of the execute step of the DTU also if errors occur that will abort the DTU-process.
- **pzDTUFinalize** ensures data consistency and that triggers/constraints to preserve data consistency are re-enabled if a disabling during DTU-run is required.
- What happens in **pzDTUFinalize**?
  - Triggers/constraints are enabled.
  - Ensure that each position/employee is assigned to a slot.
  - Dummy slot (SlotID=-1, PID=-1, PosID=-1) is ensured.
  - Update of table txPosUnit due to tPosUnitTypeSlot.
  - Set EndDate of DTU-run in tzDTULog.
- **NOTE!** Never keep triggers that are not needed or working in the disabled mode as they may be enabled via the DTU-finalize.
- Customize your own final processing if required (take care with DTU-migrations as customizings may be lost.).

# Known problem with 1:1-basetables

- Bug in versions 3.5.1/3.5.2: Data fields of 1:1-tables are left empty when a new record is inserted in the DTU-run. They are filled on the second run though (workaround: let DTU run two times.)
- Bug is fixed in 3.5.3
- You have the same effect when you miss the 1:1-setting on basetables referring to 1:1 tables. This also occurs occasionally and causes confusion about fields which remain empty. **Please check carefully!**

# Issues on validations

- DTU 3.x: it is recommended to use new validation steps.
- Validation in SQL-steps:
  - Add customized validations to staging-table records and not pre-staging-table records.
  - Records with errors: Set DTUIsGarbage=1 plus the DTUGarbageText for records which should not be imported in the staging step.
  - Records with warnings: Set DTUGarbageText for records which should still be imported in the staging step, but have a warning in the DTU-log.
  - It is obvious that you need to validate the records before the staging step and after populating the staging table. **Remember the processing order here:** If you implement the filling of the staging table in the SQL-Tab of the staging table (puzzle piece), this step will be executed before any other step of the corresponding DTU-transaction.

# DTU.exe.config for DTU 3.x

- Configure SP for SlotHandling
- Configure SP for PosRelation-Handling
- Configure Log-Levels (Critical, Error, Warning, etc.) for DTU-logs.  
DTU-Logs are written into:
  - DTU-Log-tables in database
  - DTU-Console
  - Textfile DTU.log
- Configure timeout for populating pre-staging tables.
- Configure notify mail about DTU-executions with success or error message.

## DTU.exe.config: Additional settings for 3.5.3

- `<add key="InvalidateNotUnique" value="true" />`
  - true: records in pre-staging which are not unique due to the configured unique fields are marked as garbage and not imported.
  - false: the first record within the non-unique records is imported; others are marked as garbage and are not imported (default before 3.5.3).
- `<add key="InvalidateDuplicateECK" value="true" />`
  - true: records in the staging table which are not unique due to the ECK are marked as garbage and are not imported.
  - false: the first record of the non-unique is imported; others are marked as garbage and are not imported (default before 3.5.3).
- **NOTE!** The InvalidateDuplicateECK-check just refer to imported records. If a staging table record would refers to multiple records in the ETWeb-basetable via ECK, just the first basetable record will be updated. **Check your ECK-setting very properly as you will not be notified about multiple records that fit!**
- `<add key="TreatNullAsText" value="true" />`
  - A text value "NULL" is not interpreted as database-NULL. (default: false)



# Issues to clarify

- Format of the imported data:
  - CSV or XLS (always add extra time with excel as it is more troublesome)
  - Field names
  - Encoding of the text files
  - Field separators
  - Text qualifiers (if delimiter should be contained in the text), escape character for text qualifier
  - Formatting of date values and decimal values
- Import language: Especially important if Auto-Adds of codes and org. units should be used.
- It need to be absolutely clear if the customer uses pos-pos-hierarchies or slot-slot-hierarchies.
- Lookups of codes/units: Auto-Add is activated or not? **Do not decide that yourself!**
- ECKs need to be clear.
- Required fields need to be clear.
- Sample files with filled fields are available on project start?
- Responsibility for additional technical support (support of technical consultant available?).
- Execution of DTU via task processor, manual or by windows task processor? (support of technical consultant available?)

# Why are DTU issues sometimes detected so late?

- People just focus on this symbol in the log and assume everything is in order when it appears.
- This symbol just means that the DTU has execute all steps without abortion. **BUT it does not mean that everything is in order! The DTU-log need to be checked additionally on each DTU-run:**
  - Check records with warnings and errors in pre-staging and staging tables.
  - Check warnings indicating missing and additional fields in the import data.
  - Check warnings for those staging tables which are not populated but used for lookup. Is that ok?
- QA DTU before delivery: Verify the imported data field by field.
- Make very sure that the file paths for the data sources point to the right file if the customer/consultant are coming up with issues. (Yes, such things may occur...)

# DTU in ETWeb-Migrations

## How does it work?

- Define data sources of type **Other** for the pre-staging tables.
- Fill the typed pre-staging tables from the source database.
- Use IDs of source database as ECKs. You need to add these ID-fields to the destination database tables.
- Take care about different collations from source database and destination databases when comparing strings. (Use COLLATE)

Source DB

Destination DB



# DTU in ETWeb-Migrations

## Advantages

- No need to program foreign-key-lookups yourself.
- DTU-processing for multiple slots, hierarchies, unit-assignments.
- Migration can be run multiple times and update records. Corrections are possible and the migration process can be smooth.
- Migration of data of DTU versions with very different data models (9 -> 11).

## Disadvantages

- More programming effort than a migration script. You always need to program two steps (filling of pre-staging table, filling of staging table)
- Consider issues due to multiple languages carefully (especially with the Auto-Add-feature of codes and units).

**Customers with DTU-migration:** Bayer, Kaufhof

# Summary of fequent mistakes

- Errors due to lack of awareness of the processing order, especially the filling of staging tables in DTU 3.x via the SQL-step in the staging-table-tabs causes confusion.
- Missing setting for 1:1-tables.
- SlotHandling/PosRelation handling is not configured properly and/or understood properly.
- Inadequate use of primary key fields of the staging tables for joining data. For inserted records, these fields remain Null for some versions. Better to be save to avoid using these fields.
- Adjustments of staging-tables/pre-staging tables are done without awareness that they may be used by other DTUs as well.
- DTUs are scripted partially. **NOTE that all DTUs need to be scripted for any DTU-delivery.**
- No clearness in the Pos-Pos/Slot-Slot-issue.
- Truncation of staging tables are missing.
- Problems due to wrong or changed language setting.
- Log details are ignored.
- ECK is not really uniquely identifying records or configured wrong.
- Fields that are not filled are configured as active in the staging table configuration (with Allow Null overwrite).

# Final Questions/Discussions