Artikel Import Dokumentation

Letzte Änderung: 2020-10-17

Geschrieben von Simon van Laak

Dies ist die Technische Dokumentation. Für das Anwender\*innen ist das [Handbuch](file:///C:\Users\simon.vanlaak\AppData\Roaming\Microsoft\Word\Artikel%20Import%20Handbuch.docx) gedacht. Hier geht es zum [Quellcode](file:///C:\Users\simon.vanlaak\AppData\Roaming\Microsoft\Word\Artikel%20Import\src).

Inhalt

[Problem 2](#_Toc49083545)

[Loesung 2](#_Toc49083546)

[Verwendung einer Zwischendatenbank 2](#_Toc49083547)

[Vorrausetzungen 2](#_Toc49083548)

[Datenbank 2](#_Toc49083549)

[CSV 2](#_Toc49083550)

[Fehlerbehebung 3](#_Toc49083551)

[Datenbank 3](#_Toc49083552)

[ArtikelImport\_Mappings 3](#_Toc49083553)

[ArtikelImport\_Discounts 3](#_Toc49083554)

[ArtikelImport\_Fields 4](#_Toc49083555)

[ArtikelImport\_Articles 5](#_Toc49083556)

[ArtikelImport\_Einkauf 5](#_Toc49083557)

[Klassen 5](#_Toc49083558)

[Backend 5](#_Toc49083559)

[Articles [Artikel] 5](#_Toc49083560)

[SQL 6](#_Toc49083561)

[Constants [Konstanten] 6](#_Toc49083562)

[Log 6](#_Toc49083563)

[Backend Objekte 6](#_Toc49083564)

[Discount [Rabatt] 6](#_Toc49083565)

[Field [Feld] 7](#_Toc49083566)

[Mapping 7](#_Toc49083567)

[Pair [Paar] 8](#_Toc49083568)

[SqlReport [SqlBericht] 8](#_Toc49083569)

[UpsertCommand [UpsertBefehl] 9](#_Toc49083570)

[Frontend 9](#_Toc49083571)

[ConfirmationPopUp 9](#_Toc49083572)

[EnterNamePopUp 9](#_Toc49083573)

[Form 9](#_Toc49083574)

[MessagePopUp 10](#_Toc49083575)

[Programm 10](#_Toc49083576)

[Tests 10](#_Toc49083577)

# Problem

Inha verkauft Artikel von Lieferanten an Kunden. Diese Lieferanten schicken regelmäßig CSV[[1]](#endnote-1) Dateien mit Preislisten an uns. Jeder Lieferant hat dort seine eigene Formatierung und Sortierung der Daten. Diese Preislisten müssen in die Echtzeitdatenbank eingespielt werden, damit alle Informationen in der Enventa[[2]](#endnote-2) richtig angezeigt und verwendet werden können. Artikel Import hat die Funktion den Import von diesen Preislisten für Mitarbeiter\*innen zu vereinfachen, sodass keine Hilfe vom IT-Bereich mehr notwendig ist.

# Lösung

Ziel war es, den Anwendern viel Freiheit zu geben, ohne an Übersicht einbüßen zu müssen.

Für jede Preisliste erstellen Nutzer\*innen einmalig ein [Mapping](#_Mapping). Mappings enthalten [Paare](#_Pair), die entscheiden welche Spalten der Preisliste in welche Spalte der Zwischendatenbank geladen wird. Welche Spalten die Zwischendatenbank enthält wird über die [Felder](#_Field) entschieden. Wenn alle Preislisten [importiert](#_Import) sind, kann die Zwischendatenbank in die Echtzeitdatenbank [exportiert](#_Export) werden. Felder beschreiben welche Spalte der Zwischendatenbank in welche Spalte der Echtzeitdatenbank geladen wird.

## Verwendung einer Zwischendatenbank

Die Verwendung einer Zwischendatenbank hat folgende Vorteile:

1. Im Falle eines Fehlers ist nicht gleich die Echtzeitdatenbank in Gefahr.
2. Die Echtzeitdatenbank wird in nur einem Schritt bearbeiten. Dies sorgt dafür, dass zu keinem Zeitpunkt Informationen fehlen oder fehlerhafte Informationen vorhanden sind.
3. Die Konvertierung von Werten ist sehr viel einfacher
4. Es gibt Artikel, die nur Werte anderer Artikel überschreiben sollen, die auch neu importiert werden

Die Verwendung einer Zwischendatenbank hat folgende Nachteile:

1. Alle Daten müssen doppelt verschoben werden.
2. Die Zwischendatenbank verbraucht zusätzlichen Speicherplatz

# Vorrausetzungen

## Datenbank

1. Die Artikelnummer muss einzigartig sein.
2. Die EAN muss einzigartig sein.

## CSV

1. Die Preisliste gespeichert als Typ „Unicode“ und umbenannt als .csv
2. Die erste Zeile muss die Spaltennamen enthalten.
3. Getrennt mit Semikolon [;]

# Fehlerbehebung

Beim Auftreten von Fehlern, bitte zunächst die Einträge zum Zeitpunkt des Fehlers im Log [[log.txt](file:///C:\Users\simon.vanlaak\AppData\Roaming\Microsoft\Word\log.txt)] lesen. Falls der Fehler oder Fragen nicht gelöst werden kann bitte eine E-Mail an [it@inha.de](mailto:it@inha.de) schreiben.

# Datenbank

## ArtikelImport\_Mappings

Hier werden die [Paare](#_Pair) gespeichert.

Spalten:

* NAME:
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Primärschlüssel
  + Kommentar: Name des Mappings [z.B. „99669 EDE“]
* PAIR\_TYPE:
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Paar Typ [0;1;2;3]
* PAIR\_TARGET\_FIELD:
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Sekundärschlüssel
  + Kommentar: Name des [Field](#_Field)
* PAIR\_SOURCE\_FIELD:
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Wert oder Name der CSV-Spalte [z.B. „ArtikelNr“]
* OVERWRITE:
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: True: nicht null Daten in der Zwischendatenbank sollen überschrieben werden [0;1]
* FACTOR:
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Faktor mit dem der Wert multipliziert werden soll, bei „1“ wird kein Faktor berechnet [z.B. „0,001“] oder Name der Rabatt CSV-Spalte [z.B.“RabattGruppe“]

## ArtikelImport\_Discounts

Hier werden die [Rabatte](#_Discount_[Rabatt]) gespeichert.

* Mapping
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Primärschlüssel
  + Kommentar: Name des Mappings zu dem Discount gehört
* Key
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Sekundärschlüssel
  + Kommentar: Name des Discounts -> muss mit RabattGruppen in CSV-Spalte übereinstimmen
* Discount
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Wert des Rabatts -> 51% Rabatt = 51
* SupplierID
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Die LieferantenNr des Datenlieferanten

## ArtikelImport\_Fields

Hier werden die [Felder](#_Field) gespeichert. Jede reihe in Fields steht für eine Spalte in Articles oder Einkauf.

* Name
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Primärschlüssel
  + Kommentar: Name des Fields.
* Target\_Field
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Sekundärschlüssel
  + Kommentar: Ziel Feld in der Echtzeitdatenbank [z.B. „Artikel.ArtikelNr“]
* Description
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Beschreibung des Felds
* Type
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Der Varibalentyp des Felds in [ARTICLES](#_ArtikelImport_Articles)
* Column\_Size
  + Typ: VARCHAR2
  + Größe: 255
  + Nicht Null: True
  + Kommentar: Die Spaltengröße in ARTICLES
* Is\_NVL
  + Typ: VARCHAR2
  + Größe: 1
  + Nicht Null: True
  + Kommentar: True: die Echtzeitdatenbank in diesem Feld soll mit NULL überschrieben werden [0;1]

## ArtikelImport\_Articles

Hier werden die Daten der Artikel gespeichert, die in ARTIKEL und PREISGRUPPEN auf der Echtzeitdatenbank kommen sollen.

Welche Spalten die Tabelle hat wird durch Fields gesteuert. D.h. Jede Reihe in Fields steht für eine Spalte in Articles (oder Einkauf).

## ArtikelImport\_Einkauf

Hier werden die Daten der Artikel gespeichert, die in EINKRABATT auf der Echtzeitdatenbank kommen sollen.

Welche Spalten die Tabelle hat wird durch Fields gesteuert. D.h. Jede Reihe in Fields steht für eine Spalte in Einkauf (oder Articles).

# Klassen

## Backend

### Articles [Artikel]

#### Clear

Leert die Zwischendatenbank.

#### Import

Importiert eine CSV Tabelle in die Zwischendatenbank, indem UpsertCommands erstellt und ausgeführt werden.

1. CSV Laden
2. Pro Zeile einen [UpsertCommand](#_UpsertCommand_[UpsertBefehl]) für [Articles](#_ArtikelImport_Articles) und [Einkauf](#_ArtikelImport_Einkauf) erstellen:
   1. Schlüssel Argumente Finden und in Befehle eintragen
   2. Für alle [Paare](#_Pair_[Paar]) im Mapping Werte über die erste Zeile finden ([GetValueFromArticle](#_TODO:_GetValueFromArticle():)) und als Argument in einen Befehl laden.
3. Befehle ausführen

#### Export

Lädt die Daten aus der Zwischendatenbank in die Echtzeitdatenbank.

1. Felder Laden
2. Alle [Felder](#_Field_[Feld]) aufteilen auf die jeweiligen Tabellen
3. Pro Tabelle jedes Feld als Argument in [UpsertCommand](#_UpsertCommand) eintragen
4. Befehle ausführen

### SQL

Klasse um mit der Oracle SQL [Datenbank](#_Datenbank) zu interagieren, indem Befehle und Anfragen ausgeführt werden. Diese Befehlen können zum Teil mit [UpsertCommand](#_UpsertCommand) generiert werden. Alle Befehle geben einen [SqlReport](#_SqlReport_[SqlBericht]) zurück.

#### ExecuteCommand(string commandStr)

Führt einen SQL Befehl aus.

connectionString: Enthält den Namen der Datenbank und das Passwort.

#### ExecuteCommands

Führt ExecuteCommand auf alle Befehle aus.

#### ExecuteQuery(string commandStr, int responseIndex)

Führt eine Abfrage aus. Der Response index ist die Spalte, welche ausgelesen werden soll (fängt bei 0 an).

#### ExecuteMultiLineQuery

Führt ExecuteQuery auf alle Queries aus.

#### PreventSQLInjection(string value)

In einem Wert[value] dürfen manche Zeichen nicht enthalten sein, da sie Teil der Syntax von SQL sind und dann der Befehl verändert werden könnte. Dies konnte im Falle des böswilligen Benutzens zur kompletten Zerstörung der Datenbank kommen, im Normalfall sorgt es aber für einen nicht ausführbaren Befehl.

### Constants [Konstanten]

Hier werden Konstanten, wie zum Beispiel Strings, gespeichert damit Tippfehler nicht auftreten.

Folgende Strings sind hier hinterlegt: Datenbank Passwort, Logfile Pfad, Tabnamen, Datenbank Tabellennamen

### Log

Nachrichten ausgeben und in die Logdatei [[log.txt](file:///C:\Users\simon.vanlaak\AppData\Roaming\Microsoft\Word\log.txt)] schreiben. Dies hat keinen Einfluss auf den\*die Nutzer\*in.

* Info
  + Alle Arten von Informationen, die nicht in die anderen Kategorien passen
* Event
  + Interaktionen mit der UI -> z.B. Knopfdruck
* Debug
  + Wird nur angezeigt, wenn in den Einstellungen isDebug aktiviert ist
* Error
  + Alle möglichen Fehler

## Backend Objekte

### Discount [Rabatt]

Bei einem [Paar](#_Pair) mit dem Typ DiscountField wird eine Spalte ausgewählt in der die Bezeichnungen von Rabattgruppen steht. Für jede dieser Rabattgruppen wird ein Discount erstellt. Hier wird die Bezeichnung, sowie der Rabatt (51% Rabatt -> 51) gespeichert. Wenn ein Discount erstellt oder gelöscht wird, wird dies aus sofort in der Echtzeitdatenbank in EINKRABATT übernommen, damit die Rabattgruppen auch in Enventa verwendet werden können.

Mapping: Name des Mappings zu dem der Discount gehört

Key: Name oder Kürzel der Rabattgruppe

Discount: Wert des Rabattes (51% Rabatt -> 51 -> price\*0.49)

SupplierID: Die Lieferantennummer, zu der das Mapping gehört. Dies ist relevant, um die Rabattgruppen in EinkRabatt einzutragen.

### Field [Feld]

Zeigt die Verbindung der Zwischendatenbank und der Echtzeitdatenbank an. Zusätzlich wird es benutzt um sicher zu gehen, dass die Werte in die Echtzeitdatenbank passen (Format, Größe, Typ).

Name: Der Name des Felds. Die Namen der Spalten in der Echtzeitdatenbank sind meistens schwer verständlich und teilweise in Englisch. Das Felder eigene Namen haben, soll einfaches Erstellen von Mappings garantieren, ohne Spaltennamen zu kennen. Der Name entspricht auch dem Spaltennamen in der Zwischendatenbank.

TargetInRuntime: Der Tabellenname und Spaltenname in der Echtzeitdatenbank in folgendem Format (TABELLENNAME.SPALTENNAME).

Description: Beschreibung der Spalte zum Nachlesen für Nutzer\*innen, falls der Name nicht eindeutig ist.

SqlType: Der Variablen Typ der Spalte in der Echtzeitdatenbank [NUMBER, VARCHAR2, DATE]. Wird benutzt um die Werte vom String zu konvertieren. Wird außerdem benutzt beim Erstellen der Spalte in ArtikelImport\_Articels/ArtikelImport\_Einkauf.

isNVL: Wenn True, wird die Spalte in der Echtzeitdatenbank nicht mit NULL überschrieben. Dies ist hilfreich bei zum Beispiel Bezeichnungen, die nicht veralten und nicht in allen Preislisten vorhanden sind.

Size: Die maximale Größe der Werte in der Spalte. Zum Beispiel beim String die Anzahl der Charaktere. Wenn Werte zu groß sind, werden sie am Ende abgeschnitten. Wird außerdem benutzt beim Erstellen der Spalte in ArtikelImport\_Articels/ArtikelImport\_Einkauf.

### Mapping

Ein Mapping stellt die Verbindung zwischen einer CSV von einem Lieferanten und der Zwischendatenbank. Für jede Art von Preisliste (meistens nur eine pro Lieferant) sollte es ein Mapping geben.

Name: Der Name des Mappings.

Pairs: Hier werden alle [Paare](#_Pair) des Mappings gespeichert.

Discounts: Hier werden alle [Rabatte](#_Discount) des Mappings gespeichert.

EssentialPairs: Die enthaltenen Paare werden automatisch zum Mapping hinzugefügt, wenn es neu erstellt wird. Beim Mapping verifizieren, wird außerdem untersucht ob das Mapping alle EssentialPairs enthält.

### Pair [Paar]

Ein Paar ist Teil eines [Mappings](#_Mapping). Es speichert, welche Daten in die Zwischendatenbank geladen werden. Dies können feste Werte oder Werte aus einer CSV-Spalte sein.

Mapping: Name des Mappings zu dem das Paar gehört.

TargetField: Name des Fields, in welches der zurückgegebene Wert geschrieben wird.

PairType: Der Typ des Paares, es gibt vier unterschiedliche Arten.

* 0 -> Value
  + gibt immer einen fixen Wert zurück.
  + SourceField: Der Wert der zurückgegeben wird
  + Factor: leer
* 1 -> Field
  + Gibt den Wert aus einer bestimmtem CSV-Spalte zurück, außerdem kann es einen Faktor enthalten, der dann mit dem Wert multipliziert wird.
  + SourceField: Name der CSV-Spalte
  + Factor: Der Wert aus der CSV-Spalte wird mit dem Factor multipliziert.
* 2 -> DiscountField
  + Gibt den Wert aus einer bestimmten CSV-Spalte zurück, mit einem angewendeten Rabatt multipliziert. Mit welchem Rabatt multipliziert werden soll, wird durch eine weitere CSV-Spalte ermittelt, in der ein Rabattname steht welcher dann an [Discount](#_Discount) gesendet wird. Discounts gibt dann den Rabattwert zurück.
  + SourceField: Name der Wert CSV-Spalte
  + Factor: Name der Rabatt CSV-Spalte
* 3 -> ChangingValue
  + Ähnlich zu Value, gibt aber vorerst keinen Wert zurück, sondern wird dem\*der Nutzer\*in im „Add Values“-Tab erneut angezeigt, wo dort dann ein Wert eingetragen wird.
  + SourceField: Der zuletzt eingetragene Wert
  + Factor: leer
  + Overwrite (bool): Ob Daten (die nicht NULL sind) in der Zwischendatenbank mit Überschrieben werden soll.

#### GetValueFromArticle():

Eine sehr wichtige Methode im Programm, welche ein Paar auf eine CSV anwendet. Abhängig von dem Paar-Typ werden unterschiedliche Wege gegangen, um einen Wert zurück zu geben. Wenn der Werte ausgewählt wurde, wird er durch das zugehörige Feld so angepasst, dass er von der Echtzeitdatenbank angenommen wird.

### SqlReport [SQL Bericht]

Zeigt den Erfolg von ausgeführten [SQL](#_SQL) Anweisungen. Generiert einen String.

initiatedCommands: Die Menge an zu auszuführenden Befehlen.

successfulCommands: Die Menge an erfolgreich ausgeführten Befehlen.

executionTimeSec: Die Zeit in Sekunden, die zum Ausführen benötigt wurde.

### UpsertCommand [UpsertBefehl]

Wird genutzt zum Erstellen vor [SQL](#_SQL) Befehlen. Upsert Befehlen Updaten Zeilen, die schon existieren oder erstellen neue Zeilen. Jedes Mal, wenn ein Field/Pair/Mapping/Discount erstellt oder bearbeitet wird, wird ein UpsertCommand genutzt.

Table: Die Tabelle in welche die Daten geschrieben werden sollen.

KeyArgument: Ob die Zeile schon existiert wird über die KeyArgumente gelöst, diese sind in der Tabelle einzigartig. Außerdem wird damit die Zeile ausgewählt, die bearbeitet wird.

Argument: Jedes Argument hat eine Spalte[Column] und einen Wert[Value] die Spalte der ausgewählten Zeile wird mit dem Wert gefüllt.

OnlyInsertArgument: Eine Spalte die nur beim Neuerstellen mit dem Wert gefüllt wird.

ToString(): gibt den erstellten Befehl zurück

ToStringUsingSecondTable(string table2): Erstellt den Befehl so, dass er genutzt werden kann, um Daten von der Tabelle in table2[Tabelle2] geschoben werden können. Hierfür müssen anstatt von den Werten die Spaltennamen bei den Argumenten eingetragen werden.

## Frontend

Hier ist alle Logik zu den Anwendungsoberflächen, sowie die Forms enthalten.

### ConfirmationPopUp

Ein Windowsform, welches aufpoppt um den\*ie Nutzer\*in eine Nachricht zu bestätigen. Stoppt den Mainthread.

Message: die Nachricht, die bestätigt werden soll

Confirmed: ob die Nachricht bestätigt wurde

### EnterNamePopUp

Ein Windowsform, welches aufpoppt damit der\*ie Nutzer\*in einen Mappingnamen eintragen kann. Stoppt den Mainthread.

Name: der Name, der eingetragen wurde

### Form

Das ist die Hauptoberfläche des Programms. Hier findet fast alle Nutzer\*innen-Interaktion statt.

#### Mappings

Hier wird das [Mapping](#_Mapping) ausgewählt und entweder bearbeitet oder zum Hochladen verwendet.

#### Edit

Hier wird das ausgewählt Mapping bearbeitet. Es können [Paare](#_Pair) und [Rabatte](#_Discount_[Rabatt]_1) erstellt/gelöscht werden.

#### Fields

Hier werden die [Felder](#_Field) bearbeitet oder neu erstellt. Wenn ein Feld gelöscht wird, werden auch alle Paare aller Mappings, die dieses Feld angesprochen haben entfernt.

#### Discounts

Kann nur geöffnet werden, wen ein Paar im Mapping vom Typ Discount ist. Hier könne Discounts bearbeitet/erstellt werden.

#### Add Values

Wenn das Mapping Paare vom Typ ChangingValue enthält, werden diese hier vor dem Hochladen der CSV angezeigt

#### Upload

Hier wird die CSV-Datei ausgewählt, verifiziert, [Befehle erstellt und hochgeladen](#_Import).

### MessagePopUp

Ein Windowsform, welches aufpoppt um dem\*der Nutzer\*in eine Nachricht anzuzeigen. Stoppt den Mainthread.

## Programm

Hier wird das Programm gestartet. Wenn isDebug aktiviert ist, werden hier außerdem vorher [Tests](#_Tests) gestartet.

## Tests

Hier werden zum Start des Programmes, wenn isDebug aktiviert ist, Tests durchgeführt um die Funktionalität des Programmes zu gewährleisten. Derzeit sind nur Unit Tests vorhanden, d.h. die Ausführung geht sehr schnell und stört nicht beim Debuggen. Es sollten aber noch Datenbank Tests durchgeführt werden, am besten diese nicht bei jedem Start ausführen. Falls neue Funktionen hinzugefügt werden, muss dafür gesorgt werden, dass alle Tests bestanden werden. Am besten auch für neue Funktionen Tests schreiben.

# TODO:

1. Namen in dieser Dokumentation muessen aktualisiert werden
2. Es muessen noch Pairtypes hinzugefuegt

1. CSV: Die Preisliste als CSV gespeichert. [↑](#endnote-ref-1)
2. Enventa: Das Warenwirtschaftssystem, welches zum Zeitpunkt dieses Programmes von INHA verwendet wird. [↑](#endnote-ref-2)