

## Anforderungen an eine Schnittstelle zum H&U Smart.KIS

### Grundsätzliches zur Schnittstellenanbindung:

Dieses Dokument beschreibt eine REST-Schnittstelle, die vom Betreiber der anzubindenden Warenwirtschaft erstellt wird, und auf die das smart.KIS von Hutter & Unger zugreift.

Es ist notwendig, dass das smart.KIS zu *jedem* Zeitpunkt Zugriff auf die Schnittstelle hat. Es muss Stammdaten / Kundendaten anhand einer eindeutigen ID sowohl lesen als auch schreiben können.

Zusätzlich müssen Stammdaten / Kundendaten anhand eines Änderungszeitstempels gelesen werden können. Bewegungsdaten müssen anhand einer eindeutigen ID und einem Änderungszeitstempel gelesen werden können.

Alle Leseanforderungen nach dem Änderungszeitstempel müssen minütlich erfolgen können und einen Abfragezeitraum mindestens in Minuten angeben können, also bspw. *Geändert seit 01.02.2021 19:05*

Insofern ein Löschen von Datensätzen in der Warenwirtschaft möglich ist, muss dies nachvollziehbar durch das smart.KIS sein. Bitte lesen Sie sich das Dokument sorgfältig durch und notieren Sie sich Fragen und auf Ihrer Seite fehlende Attribute in Stamm- und Bewegungsdaten.

## 1. Datentypen (MySQL / InnoDB Engine) im Smart.KIS

### 1.1. Stammdaten (Eine einzelne Person)

Anmerkungen:

- **Daten, die durch die Warenwirtschaft zur Verfügung gestellt werden sind grün und durch das Smart.KIS generierte Daten blau markiert**
- „Mandanten“ bezeichnet Kunden des Warenwirtschaftsentwicklers zur Abgrenzung von den Endkunden
- Insofern ein Wert nicht geliefert werden kann müssen hier ggf. Standardwerte definiert werden. Wenn bspw. alle Laufkundenumsätze auf die Kundennummer 0 gebucht werden, ist kein Kennzeichen für Laufkunden notwendig
- Die Reservierungsdaten für die digitale Neukundenregistrierung werden im Regelfall nach der Anlage eines Neukundenkreises festgelegt, mehr hierzu unter dem Gliederungspunkt 2.3

Feldname	Datentyp (Länge)	Beschreibung
foreign_guid	VARCHAR(191)	Die eindeutige ID des Kundendatensatzes
customer_identifier	VARCHAR(191)	Die Kundennummer. Sie sollte eindeutig sein, bzw muss eindeutig sein, wenn keine foreign_guid zur Verfügung gestellt werden kann.
language	VARCHAR(2)	ISO 639-1 Code der Sprache. Momentan werden nur Deutsch und Französisch geführt.

store_id	VARCHAR(191)	Die ID der Stamffiliale des Kunden
internal_customer_identifizier	VARCHAR(255)	Die interne Kundennummer, falls eine solche z.B. innerhalb einer Filiale genutzt wird.
card_number	VARCHAR(191)	Die Kartennummer
family_number	VARCHAR(191)	Die Familiennummer
casual_customer	TINYINT(1)	Gibt an, ob es sich um einen Laufkunden handelt (1) oder nicht (0).
is_employee	TINYINT(1)	Gibt an, ob es sich um eine/n Mitarbeiter/in handelt (1) oder nicht (0).
salutation	ENUM(n,h,c,f,m)	Die Anrede. n = Keine Angabe / Divers (Keine Darstellung) h = Familie c = Firma f = Frau m = Herr
title	VARCHAR(191)	Der Titel
first_name	VARCHAR(191)	Der Vorname
last_name	VARCHAR(191)	Der Nachname
address_1	VARCHAR(191)	Die Straße, insofern die Warenwirtschaft kein eigenes Feld für die Hausnummer hat, inkl. dieser.
street_number	VARCHAR(191)	Die Hausnummer, falls diese in einem separaten Feld geführt wird.
address_2	VARCHAR(191)	Ein Adresszusatz
zip	VARCHAR(191)	Die Postleitzahl
city	VARCHAR(191)	Die Stadt
country_iso	VARCHAR(191)	Das Land als ISO 3166 1 Alpha 2 Code
date_of_birth	DATE (YYYY-MM-DD)	Das Geburtsdatum
email	VARCHAR(191)	Die E-Mailadresse
app_email	VARCHAR(191)	Die E-Mailadresse, welche für die H&U-App verwendet wird.
phone_work	VARCHAR(191)	Telefonnummer Arbeit
phone_mobile	VARCHAR(191)	Mobiltelefonnummer
phone_home	VARCHAR(191)	Telefonnummer Privat
pin	VARCHAR(191)	Der Aktivierungscode für die H&U-App. Beim Import von Stammdaten wird dieser automatisch erzeugt. Mit diesem und der Kundennummer kann man die App für dieses Konto aktivieren.
last_updated_at	DATETIME (YYYY-MM-DD hh:mm:ss)	Die letzte Aktualisierung in der Warenwirtschaft
comment	TEXT(65535)	Ein Kommentartext
logins	UINT(10)	Die Anzahl der Kunden-Logins in die H&U-App(s) des Mandanten
last_login_at	DATETIME	Der Zeitstempel des letzten Logins
email_marketing_confirmed	TINYINT(1)	Gibt an, ob E-Mailmarketing erlaubt wurde.

email_marketing_confirmed_at	DATETIME	Gibt an, wann E-Mailmarketing erlaubt wurde.
app_email_marketing_confirmed	TINYINT(1)	Gibt an, ob E-Mailmarketing für die App-E-Mailadresse erlaubt wurde.
app_email_marketing_confirmed_at	DATETIME	Gibt an, wann E-Mailmarketing für die App-E-Mailadresse erlaubt wurde.
classic_marketing_confirmed	TINYINT(1)	Gibt an, ob postalisches Marketing erlaubt wurde.
classic_marketing_confirmed_at	DATETIME	Gibt an, wann postalisches Marketing erlaubt wurde.
phone_marketing_confirmed	TINYINT(1)	Gibt an, ob Telefonmarketing erlaubt wurde.
phone_marketing_confirmed_at	DATETIME	Gibt an, wann Telefonmarketing erlaubt wurde.
whatsapp_marketing_confirmed	TINYINT(1)	Gibt an, ob Whatsapp-Marketing erlaubt wurde.
whatsapp_marketing_confirmed_at	DATETIME	Gibt an, wann Whatsapp-Marketing erlaubt wurde.
app_marketing_confirmed	TINYINT(1)	Gibt an, ob Marketing durch die App erlaubt wurde.
app_marketing_confirmed_at	DATETIME	Gibt an, wann App-Marketing erlaubt wurde.
marketing_block_confirmed	TINYINT(1)	Gibt an, ob kein Marketing mehr erfolgen soll (Werbesperre)
marketing_block_confirmed_at	DATETIME	Gibt an, wann die Werbesperre verhängt wurde.
first_purchase_at	DATETIME	Datum des ersten Einkaufs
uses_app	TINYINT(1)	Gibt an, ob der Kunde die App aktiviert oder sich in dieser registriert hat.
uses_app_since	DATETIME	Gibt an, seit wann der Kunde die App aktiviert hat.
reserved_for_digital_registration	TINYINT(1)	Markiert den Datensatz als eine leere Kundenhülle für die Neukundenregistrierung.
reserved_for_paper_registration	TINYINT(1)	Markiert den Datensatz als eine leere Kundenhülle für Papier-Anträge zum Loyalty-Programm.
has_digital_registration	TINYINT(1)	Gibt an, ob eine digitale Registrierung erfolgt ist.
has_paper_registration	TINYINT(1)	Gibt an, ob eine Registrierung zum Loyalty-Programm über einen Papier-Antrag erfolgt ist.
is_disabled	TINYINT(1)	Gibt an, ob der Kunde für die App-Nutzung deaktiviert ist.
is_disabled_reason	TEXT(65535)	Gibt den Grund für die Deaktivierung an
is_disabled_since	DATETIME	Gibt an, wann der Kunde deaktiviert wurde
is_deleted	TINYINT(1)	Gibt an, ob der Kunde gelöscht wurde
is_deleted_reason	TEXT(65535)	Gibt den Grund für die Löschung an
is_deleted_since	DATETIME	Gibt an, wann der Kunde gelöscht wurde

## 1.2. Bewegungsdaten (Kaufdaten)

Anmerkungen:

- Bei Retouren müssen die gelieferten Preiswerte negativ sein. Ansonsten positiv
- Die Summe von *sales\_price\_2\_with\_vat \* quantity* aller Bewegungsdaten muss exakt dem Umsatz in der Warenwirtschaft entsprechen
- *Sales\_price\_1\_with\_vat* muss *sales\_price\_2\_with\_vat + reduction\_price\_with\_vat + discount\_price\_with\_vat* entsprechen
- Durch das smart.KIS werden nur Bewegungsdaten abgefragt. Es werden keine Bewegungsdaten gesendet oder intern verändert.

Feldname	Datentyp	Beschreibung
foreign_guid	VARCHAR(255)	Eine eindeutige ID des Bewegungsdatensatzes
customer_identifier	VARCHAR(191)	Die Kundennummer
receipt_number	VARCHAR(191)	Die Bonnummer
type	ENUM(sale, refund, selection)	Die Art der Bewegung Sale = Verkauf Refund = Retoure Selection = Auswahl
store_id	VARCHAR(191)	Die ID der Filiale
department	VARCHAR(191)	Die Abteilung in der die Ware gekauft wurde oder aus der diese stammt.
cash_register	VARCHAR(191)	Die ID der Kasse
employee	VARCHAR(191)	Die ID des Verkaufspersonals.
seller	VARCHAR(191)	Der Name des Verkaufspersonals.
sku	VARCHAR(191)	Die eindeutige Artikelnummer.
name	VARCHAR(191)	Der Name des Artikels.
name_supplemental	VARCHAR(191)	Ein Fortsatz des Artikelnamens.
cost_price	DECIMAL(15,4)	Der Netto-Einkaufspreis pro Einheit
cost_price_with_vat	DECIMAL(15,4)	Der Brutto-Einkaufspreis pro Einheit
sales_price_1	DECIMAL(15,4)	Der Netto-Verkaufspreis 1 pro Einheit
sales_price_1_with_vat	DECIMAL(15,4)	Der Brutto-Verkaufspreis 1 Einheit
discount_price	DECIMAL(15,4)	Der gewährte Netto-Rabatt pro Einheit
discount_price_with_vat	DECIMAL(15,4)	Der gewährte Brutto-Rabatt pro Einheit
reduction_price	DECIMAL(15,4)	Der gewährte Netto-Nachlass pro Einheit
reduction_price_with_vat	DECIMAL(15,4)	Der gewährte Brutto-Nachlass pro Einheit
sales_price_2	DECIMAL(15,4)	Der Netto-Verkaufspreis 2 pro Einheit.
sales_price_2_with_vat	DECIMAL(15,4)	Der Brutto-Verkaufspreis 2 pro Einheit. entsprechen.
vat_percentage	DECIMAL(4,2)	Der Umsatzsteuersatz (19% = 19,00)
quantity	UNSIGNED INT(10)	Die Menge oder Anzahl der Einheiten, stets positive Werte.
supplier	VARCHAR(191)	Der Lieferant

label	VARCHAR(191)	Die Marke
product_group_1	VARCHAR(191)	Die Hauptwaren / -produktgruppe
product_group_2	VARCHAR(191)	Die Unterwaren / -produktgruppe
size	VARCHAR(191)	Die Größe des Artikels (bspw. XL, 42, etc.)
goods_received_at	DATETIME	Das Wareneingangsdatum
sold_at	DATETIME	Das Verkaufsdatum
currency	VARCHAR(3)	Die Währung insofern nicht alle Preise in Euro sind. Werte in ISO 4217

## 2. Anforderungen und Funktionalitäten

### 2.1. Alle Daten betreffend

#### 2.1.1. Grundsätzliche Kommunikation

**Die Kommunikation erfolgt über einen JWT-Token, der über Nutzernamen und Passwort bei der Warenwirtschaft angefragt wird. (OAuth)**

**Die Kommunikation zwischen den Systemen erfolgt im JSON-Format nach ECMA 404.**

**Die Kommunikation muss HTTPS verschlüsselt sein. Ohne ein gültiges Zertifikat erfolgen keine Anfragen durch das smart.KIS. Gegebenenfalls entsteht durch ein ungültiges Zertifikat so ein Datenverlust.**

Daten, die die Datentypen DATETIME, DATE, ENUM, VARCHAR und TEXT verwenden, werden in den JSON-Typ *string* umgewandelt. Alle anderen in den Tabellen der Stamm- und Bewegungsdaten erwähnten Typen werden in *number* umgewandelt.

Die Schnittstelle muss bei Änderungsanfragen mindestens die Werte annehmen, die sie für die entsprechenden Felder auch ausgibt. Ein Stammdatensatz muss also exakt mit den Werten, die aus einer Abfrage stammen, auch wieder zurückgeschrieben werden können, ohne, dass sich bis auf das Änderungsdatum etwas am Datensatz verändert.

Kann es verschiedene Varianten / Versionen einer Schnittstelle geben, soll diese (zumindest innerhalb desselben Major-Release) abwärtskompatibel sein.

Es muss eine Möglichkeit geben die aktuellste Versionsnummer sowie ein Changelog einzusehen.

Grundsätzlich wird bei einer eigens für die Kommunikation mit dem smart.KIS geschaffenen Schnittstelle vorausgesetzt, dass es ohne die Kenntnisnahme von Hutter und Unger keine Änderungen an dieser gibt.

Die Abfragen für Stamm- und Bewegungsdaten sollen Pagination mit den Parametern Seitengröße und Seite ermöglichen. Es soll sich in der Response der Schnittstelle ein Wert befinden, der angibt, ob noch Seiten verbleiben.

Jeder Datentyp in der Schnittstelle (Stamm/Kundendatensatz, Bewegungsdatensatz, weitere) muss eine eindeutige ID haben, die stets übergeben wird und sich **niemals** verändert.

### 2.1.2. Änderungsabfragen

Es soll möglich sein, abzufragen, welche Daten in einem Zeitraum verändert wurden.

**Alle** Änderungen an den Datensätzen werden notiert, also auch Umbuchungen und ähnliches.

Der Start- und Endzeitraum muss *sekundengenau* definierbar sein für einen möglichst schnellen Datenaustausch.

### 2.1.3. Löschen von Daten (insofern möglich)

Es soll eine Liste mit gelöschten IDs abfragbar sein, sowohl für Bewegungsdaten als auch Stammdaten. Alternativ kann bei Stammdaten die Information übergeben werden ob und wann ein Stammdatensatz gelöscht worden ist. Die Stammdatensätze werden in der H&U Datenbank als gelöscht markiert und die Felder geleert, falls die Assoziation der Datensätze z.B. zu den Bewegungsdaten nicht verloren gehen darf.

Zu klärende Funktionalität: Können Kundennummern neu vergeben werden?

Der Start- und Endzeitraum muss *sekundengenau* definierbar sein für einen möglichst schnellen Datenaustausch.

## 2.2. Bewegungsdaten betreffend

### Behandlung von Auswahlen:

Zu klärende Funktionalität: Gibt es eine Flag für Rückgabe und Kauf, einen Zeitstempel oder weitere Daten dazu? Falls ja, wären auch diese für eine Implementierung von Auswahlen in den H&U Apps sehr nützlich.

## 2.3. Stammdaten betreffend

Mit der eindeutigen ID sollen sich Kundendaten ändern lassen. Es muss eine konkrete Response geben, ob alle übergebenen Felder verarbeitet wurden.

Ein Erstellen eines neuen Datensatzes ist nicht zwingend notwendig, da bei vielen Warenwirtschaften das konsistenteste Vorgehen folgendes ist:

- Einen Nummernkreis mit leeren Kundenhüllen in der Warenwirtschaft anlegen.
- Kundenhüllen werden in das Smart.KIS importiert und als solche markiert. Die Markierung erfolgt über das Backend des smart.KIS (`reserved_for_digital_registration = 1`)
- Bei Registrierung in der H&U App wird die nächste, leere Nummer ausgewählt und die Stammdaten in die Warenwirtschaft zurückgeschrieben. Nur, wenn die Übertragung erfolgreich war, werden geänderte Daten wieder aus der Warenwirtschaft übernommen. Bis dahin bleibt die Nummer gesperrt.

### 3. Schnittstellenlayout

Nachfolgend werden die Endpunkte, Requests und erwarteten Responses festgelegt, die von H&U bevorzugt werden.

Datumsangaben in DATETIME nach <https://dev.mysql.com/doc/refman/8.0/en/datetime.html>

Die Folge ... gibt an, dass hier im Realfall ggf. weitere Daten kommen würden.

Der Wert pages\_remaining in den Antworten gibt als boolean aus, ob bei der Pagination noch weitere Seiten vorhanden sind. Die erste Seite ist die Seite 0.

Sollte ein Fehler auftreten können, muss ein Verhalten für das smart.KIS vorgegeben werden. Bspw. ein HTTP-Fehler 500 ohne einen besonderen, zusätzlichen Fehlercode für automatisiertes Verhalten ist nicht akzeptabel.

#### 3.1. Abfrage aller geänderten Stammdaten seit einem Datum

Beispiel: Seit dem 01.Januar 2020 um 14 Uhr 0 Minuten und 0 Sekunden geändert mit einer Seitengröße von 1000 auf Seite 0. Die Stammdaten sind dabei nach Alter absteigend sortiert. Sollten also während der Pagination neue Datensätze geändert werden, werden diese an der letzten Seite „angehängt“.

##### Request

HTTP: [GET] /customers?ChangeSince=2020-01-01 14:00:00?PageSize=1000?Page=0

##### Response

Status-Code: 200 – OK

Body:

```
{
  "total_records":2000,
  "pages_remaining":true,
  "records":
  [
    {
      "customer_identifier": "303020",
      ...
    },
    {
      "customer_identifier": "303432",
      ...
    }
    ...
  ]
}
```

### 3.2. Abfrage aller geänderten Bewegungsdaten seit einem Datum

Beispiel: Seit dem 01.Januar 2020 um 14 Uhr 0 Minuten und 0 Sekunden geändert mit einer Seitengröße von 1000 auf Seite 0. Die Bewegungen sind dabei nach Alter absteigend sortiert. Sollten also während der Pagination neue Datensätze geändert werden, werden diese an der letzten Seite „angehängt“.

#### Request

HTTPS: [GET] /sales?ChangedSince=2020-01-01 14:00:00?PageSize=1000?Page=0

#### Response

Status-Code: 200 - OK

Body:

```
{
  "total_records":4000,
  "pages_remaining":true,
  "records":
  [
    {
      "foreign_guid":"2131244",
      "customer_identifier": "303020",
      ...
    },
    {
      "foreign_guid":"2131245",
      "customer_identifier": "303020",
      ...
    },
    {
      "foreign_guid":"2131246",
      "customer_identifier": "303432",
      ...
    },
    ...
  ]
}
```



### 3.3. Abfrage eines Stammdatensatzes anhand der Kundennummer

#### Request

HTTPS: [GET] /customer?customer\_identifier=213030

#### Response

Status-Code: 200 – OK

Body:

```
{
  "customer_identifier": "303020",
  "first_name": "Peter",
  "last_name": "Müller",
  ...
}
```

### 3.4. Abfrage eines Bewegungsdatensatzes anhand der ID

#### Request

HTTPS: [GET] /sale?foreign\_guid=2131244

#### Response

Status-Code: 200 – OK

Body:

```
{
  "foreign_guid": "2131244",
  "customer_identifier": "303020",
  ...
}
```

### 3.5. Änderung eines Stammdatensatzes anhand der Kundennummer

Hierbei müssen nur die zu ändernden Felder angegeben werden. Alle anderen Felder, bis auf das Änderungsdatum des Datensatzes bleiben unverändert.

#### Request

HTTPS: [POST] /customer?customer\_identifier=2131244

Body:

```
{
  "customer_identifier": "303020",
  "first_name": "Peter",
  "last_name": "Müller"
}
```

#### Response

Status-Code: 200 - OK

Body:

```
{
  "updated_fields": [
    "customer_identifier",
    "first_name",
    "last_name"
  ]
}
```

### 3.6. Stammdatenlöschliste (insofern notwendig wie in 2.1.3 beschrieben)

#### Request

HTTPS: [POST] /customers?DeletedSince=2020-01-01 14:00:00

#### Response

Status-Code: 200 - OK

Body:

```
{
  {
    "customer_identifier":"9312423429",
    "deleted-at":"2020-01-10 13:12:12",
    "reason":" ",
  },
  {
    "customer_identifier":"9312423423",
    "deleted-at":"2020-01-10 13:12:12",
    "reason":" ",
  }
}
```

### 3.7. Bewegungsdatenlöschliste (insofern notwendig wie in 2.1.3 beschrieben)

#### Request

HTTPS: [POST] /sales?DeletedSince=2020-01-01 14:00:00

#### Response

Status-Code: 200 - OK

Body:

```
{
  {
    "foreign_guid":"9312423429",
    "deleted-at":"2020-01-10 13:12:12",
    "reason":" ",
  },
  {
    "foreign_guid":"9312423423",
    "deleted-at":"2020-01-10 13:16:32",
    "reason":" ",
  }
}
```