

Squeeze

- [Mehr → Tabelle in Squeeze Stammdaten hinzufügen](#)
- [Mehr → Squeeze-Stammdaten in Tabelle laden](#)

Mehr ? Tabelle in Squeeze Stammdaten hinzufügen

Dieser Dialog geht in die andere Richtung: nicht in die Datenbank, sondern **nach Squeeze**. Gewählt wird eine SQL-Tabelle – dieselbe Auswahl wie bei *SQL Tabelle ändern* –, und daraus entsteht über die Squeeze-API die passende Stammdatentabelle:

- ein `POST` auf `api/v2/masterData/tables` für die Tabelle, der mit ihrer Id antwortet,
- danach ein `POST` je Spalte auf `.../tables/<id>/columns`.

Die SQL-Tabelle selbst wird dabei nur gelesen.

Die Stammdatentabelle in Squeeze

Feld	Bedeutung
Name in Squeeze	Vorgeschlagen wird <code>Breeze_<Tabellenname></code> . Wer ihn ändert, behält seine Eingabe auch beim nächsten Tabellenwechsel
Beschreibung	Freier Text, den Squeeze zur Tabelle anzeigt. Bleibt das Feld leer, wird der Name der SQL-Tabelle eingetragen

Felder

Die Felder heißen in Squeeze genauso wie die Spalten der Tabelle – so kommt eine Übertragung zwischen beiden ohne Zuordnungstabelle aus.

- **ID geht nie mit.** Eine Squeeze-Stammdatentabelle zählt ihre Zeilen selbst und bringt eine eigene Id mit; eine zweite daneben bedeutete dasselbe und sagte etwas anderes, sobald beide Seiten schreiben.
- **Licence bleibt im Angebot** – welchem Mandanten eine Zeile gehört, weiß Squeeze von sich aus nicht.
- **Zeichenvorrat:** Squeeze nimmt höchstens 45 Zeichen aus Buchstaben, Ziffern, Unterstrich und Bindestrich. Eine Spalte mit Punkt, Leerzeichen oder Umlaut bekommt darum ein Feld für einen **Ersatznamen**; der Rückweg in die Tabelle bleibt über `sourceColumn` erhalten, das die SQL-Spalte behält. Kurze Namen wie `de` oder `en` sind in Ordnung.

Übertragungsskript

Die frisch angelegte Stammdatentabelle ist **leer**. Der nächste Schritt steht deshalb gleich in der Rückmeldung – drei Haken:

Haken	Wirkung
Übertragungsskript erzeugen	legt ein Portalskript nach dem Vorbild von <code>Squeeze_UserExit_MasterData_TransferPrincipalsApi</code> an, mit einem <code>addColumnMap()</code> je Feld. Name: <code>Squeeze_UserExit_Export_<Tabelle></code> , Kategorie <code>Squeeze_UserExit</code> . Gibt es den Namen schon, kommt eine Rückfrage – der Quelltext könnte von Hand geändert sein
Stammdaten gleich übertragen	führt genau dieses Skript einmal aus. Dafür – und nur dafür – ist das erzeugte Skript <code>clientExecutable</code> , anders als die Vorlage
Täglich als Job ausführen	mit einer Uhrzeit daneben (<code>HH:MM</code>). Das Skript bekommt <code>RunAsJob</code> , den festen Zeitpunkt und die Uhrzeit; ohne den Haken läuft es nur, wenn es jemand startet

Die letzten beiden Haken hängen am ersten: was nicht angelegt wird, lässt sich weder ausführen noch einplanen. Zeitpunkt und Häufigkeit lassen sich danach im DOCUMENTS-Manager ändern.

Das `Breeze_` der Tabelle fällt im Skriptnamen weg, und „Export“ steht darin, weil es in diese Richtung geht: aus einer DEX-Tabelle **hinaus** nach Squeeze. Was von dort holt, heißt `Squeeze_UserExit_MasterData_*` und soll davon zu unterscheiden sein.

Wie viele Zeilen übertragen wurden, zählt das Skript nicht mit – das steht im Log unter `Logs/SqueezeImport`. Hat die SQL-Tabelle für diesen Mandanten keine Zeile, bleibt die Stammdatentabelle vorerst leer; das Skript überträgt beim nächsten Lauf, was dann da ist.

Die Gegenrichtung beschreibt [Squeeze-Stammdaten in Tabelle laden](#).

Mehr ? Squeeze-Stammdaten in Tabelle laden

Die Gegenrichtung zu [Tabelle in Squeeze Stammdaten hinzufügen](#): aus einer Squeeze-Stammdatentabelle in eine Tabelle einer DEX-Datenbank.

Erzeugt wird dafür ein Portalskript `Squeeze_UserExit_Import_<Zieltabelle>` nach dem Vorbild von `DEXPRO__UserExit_SqueezeMasterdata_FirmInformation` – mit `SqueezeImport`, einem `setTable()` für die Squeeze-Seite und einem `add()` je Feld.

Nur in eine Tabelle, die es gibt

Angelegt wird hier nichts und geändert auch nichts. Die Zieltabelle wird aus den vorhandenen gewählt, ihre Spalten werden gelesen, wie sie sind, und zugeordnet wird nur auf sie. Fehlt die Tabelle noch, ist [SQL Tabelle anlegen](#) der Schritt davor.

Feld	Bedeutung
Stammdatentabelle in Squeeze	die Tabelle, aus der gelesen wird. Ihre Felder stehen danach in der Zuordnung zur Wahl
Verbindung	die DEXPRO-Verbindung der Zieltabelle. Workflow, DOCUMENTS und Squeeze stehen nicht zur Wahl
Zieltabelle	die Tabelle, in die geschrieben wird

Zuordnung

Zugeordnet wird **spaltenweise von der Zieltabelle aus** und nicht von Squeeze her: was gefüllt werden *muss*, steht in der DEX-Tabelle. Eine Pflichtspalte ohne Wert lässt den ersten Lauf scheitern – der Dialog sagt es als Warnung, lehnt aber nicht ab, denn ein Standardwert kann es doch noch richten.

Je Zielspalte gibt es drei Möglichkeiten:

Auswahl	Bedeutung
ein Feld aus Squeeze	vorbelegt ist ein Squeeze-Feld gleichen Namens

Auswahl	Bedeutung
ein fester Wert	bei <code>Licence</code> vorbelegt mit dem Mandanten. <code>cLicence</code> geht dabei als Konstante in den Quelltext, alles andere als Text
- nicht füllen -	die Spalte bleibt leer

Die Spalte `ID` steht nur abgeblendet da: sie zählt sich selbst hoch.

Zieltabelle vor jedem Lauf leeren

“ **Die einzige Stelle im ganzen Editor, die Daten zerstört.** Mit diesem Haken beginnt jeder Lauf mit `TRUNCATE TABLE` auf der Zieltabelle. Was darin steht und nicht aus Squeeze kommt, ist danach weg.

So ist ein Stammdatenabgleich gemeint, und der Haken ist deshalb voreingestellt. Er lässt sich abwählen; das erzeugte Skript schreibt seinen Zustand ausdrücklich hin, statt sich auf den Vorgabewert der Bibliothek zu verlassen. Wer gleich ausführen lässt, bekommt vorher noch eine Rückfrage.

Anlegen, ausführen, einplanen

`Skript erzeugen` legt das Portalskript an; ein vorhandenes gleichen Namens wird nur nach Rückfrage überschrieben. Daneben stehen dieselben beiden Haken wie beim Export:

- **Stammdaten gleich laden** – führt das Skript einmal aus.
- **Täglich als Job ausführen** – mit Uhrzeit; ändern lässt sich das danach im DOCUMENTS-Manager.