Inhaltsverzeichnis

Hochschuleigene Laderoutinen

Übersicht

Eigene Laderoutinen sind öfter gewünscht, wenn bestimmte Daten von den bisherigen Modulen (noch) nicht übernommen werden.

Als Beispiele könnten sein
- erfasste Mathenote von Bewerbern
- neue Tabelle zu Lehrbeauftragten
- bestimmte Zusatzfelder in MBS.inst

Dabei geht es darum, die Daten zunächst aus dem Quellsystem zu entladen (CSV), sie dann in SuperX/BI einzuspielen und ggfs. noch zu transformieren.

Lade-Techniken

Folgende Lade-Techniken sind möglich:
- Shellscripte (DOQUERY, sx_upload_table.x)
- Kettle
- ETL-Sonderladeroutinen

Beispielanforderung Mathenote

Die Technische Universität Hamburg (TUHH) erfasst die Mathematiknoten, welche Bewerber auf ihrem bisherigen Bildungsweg erreichten. Diese Mathematiknoten sollen zukünftig in Auswertungen einbezogen werden.

Ziel der Laderoutine

Das Ziel der Laderoutine ist das Befüllen einer neuen Tabelle mit Bewerbernummer und Mathematiknote. Diese neue Tabelle wird über das Feld 'Weitere Tabellen' im "Bewerbungen und Zulassungen Datenblatt" erreichbar sein. Ggf. ist noch ein Zeitraum anzugeben.
Tabellenstruktur:

- Tabellenname: zul_bew_mathenote
- Felder:
  - Bewerbernummer (INT)
  - Mathenote (FLOAT)
- Zusätzlich zum Hochladen noch eine "_neu"-Tabelle, die quasi als Zwischenlager dient.

Implementierung der Laderoutine

Entscheidung über die Technik des Ladens

- Shellscripte (DOQUERY, sx_upload_table.x)
- Kettle
- BI Sonderladeroutine

Unload aus dem Vorsystem

Entladen der Tabelle **application_content**, im Ergebnis eine CSV Datei

Laden der CSV-Datei

Hochladen mit Kettle oder Shellscript in _neu-Tabelle

Transformation in finale Tabelle

Aus der _neu-Tabelle in die finale Tabelle schreiben, ggf. noch Tranformation, z.B. von Noten-Punkten zur Note Dezimal (z.B. 11="gut")

Automatisierung

Nächtliches Laden mit Shellscript oder Kettle

Einbinden in das Bewerbungen Datenblatt

- sx_tables: Einfügen der Tabellennamen
- sx_fields: Einfügen der Spaltennamen inkl. Relationen (Fremdschlüssel-Beziehung von Bewerber-Mathenote zu Bewerbung Datenblatt (zul_bew_mathenote.bewnr = zul_antr_aggr.bewnr)

ETL Sonderladeroutinen

Im ETL-Modul gibt es die Möglichkeit, eigene Sonderladeroutinen zu entwerfen. Diese können dann auch in der HISinOne-BI ab Version 2024.12 in der neuen Komponentenverwaltung genutzt werden.

Voraussetzung dafür ist, dass das ETL-Modul aus dem Downloadbereich installiert ist.

Eine Laderoutine kann Entladeschritte, Ladeschritte und Transformationsschritte (SQL) enthalten.

Die Installation einer Sonderladeroutine läuft über ein komplexes Script, dass mit Freemarker-Variablen gefüllt wird und dann später zu reinem SQL transformiert wird, der die Installation ausführt.

Voraussetzungen

etl_step_type

Kontrollieren Sie, ob die BI-Tabelle etl_step_type mit fünf Datensätzen gefüllt ist.

Falls diese bei Ihnen leer ist, fügen Sie folgende Step-Types ein

```
INSERT  INTO
etl_step_type
```

ETL-Makros zur Generierung

Um fertigen SQL zu generieren muss in der verwendeten Datenbank in der Tabelle fm_templates die ETL_MAKRO aus mind. ETL-Modul 0.5 installiert sein. Damit ist nur die Generierung von Installations-SQL gemeint, zur Verwendung der Steps ist dies nicht nötig.

Definition von Sonderladeroutinen

Ausgangspunkt ist die Definition einer (oder mehrerer) Sonderladeroutinen.

Als uniquename muss eine eindeutige Kennung gewählt werden, die auf "special" endet und bei systeminfo, die ID zu welchem Teilbereich (wie Finanzen,Personal oder Studierende) entsprechend der Tabelle systeminfo die Sonderladeroutine gehört und unter deren Hauptkonnektor sie erscheinen soll. etl_job_params können leer bleiben.

```
<#assign    etl_jobs  =  [
```

Beispiel Entladen/Laden-Schritte

In einem einfachen Fall will man bestimmte Zusatzfelder entladen. Dazu legt man zwei ETL-Steps an.

Das Attribut "etl_job" verweist auf den ETL-JOB ("fin_inst_special") zu dem die Steps gehören sollen.

Dann gibt man ihnen einen uniquename, einen Namen und einen Typ

- UNLOAD zum Entladenaus einer Quell-Datenbank
- LOAD zum Einspielen in die BI-Datenbank

```
<#assign  etl_steps = [
{"etl_job":"fin_inst_special",
```

Parameter für UNLOAD-Steps

Folgende Parameter müssen für einen UNLOAD-Step hinterlegt werden datasource,sql und unlFile.

Das attribut "datasource" gibt die Quelldatenbankverbindung in HisInOne an. (hier im Beispiel mbs).

Im Script darf es für alle step_properties nur eine Definition mit <#assign etl_step_properties= .. geben, weitere StepProperties müssen in dieser Aufzählung ergänzt werden.

```
<#assign
```

Die folgenden Parameter werden vom Script automatisch mit defaultwerten gefüllt, könnten bei Bedarf aber zusätzlich definiert werden

```
{"etl_step":"unload_fin_inst",
"prop_name":"active",
```

Parameter für LOAD-Steps

Folgende Parameter müssen für einen LOAD-Step hinterlegt werden.

Das Attribut "tableName" (hier im Beispiel "fin_inst_plus")gibt die Zieltabelle an, in die vorher entladenen Daten eingespielt werden sollen.

Im Script darf es für alle step_properties nur eine Definition mit <#assign etl_step_properties= .. geben, weitere StepProperties müssen in dieser Aufzählung ergänzt werden.

```
<#assign
etl_step_properties  =  [
```

Die folgenden Parameter werden vom Script automatisch mit defaultwerten gefüllt, könnten bei Bedarf aber zusätzlich definiert werden, insbesondere header true könnte interessant sein.

```
{"etl_step":"upload_fin_inst",
"prop_name":"database",
```

DOSQL-Step

Einen DOSQL-Step, der eine SQL-Datei ausführt legt man folgendermaßen an:

Innerhalb der etl_steps Definition mach man einen Eintrag mit dem etl_job und einem eindeutigen uniquename, der später die SQL-Datei referenziert, Typ ist "DOSQL".

```
<#assign  etl_steps = [
{"etl_job":"meinJob",
```

Innerhalb der etl_step_properties muss für den DOSQL-Step der folgende Eintrag definiert werden

```
{"etl_step":"update_fin_zusa
tzmerkmale","prop_name":"
```

Die folgenden Parameter werden vom Script automatisch mit defaultwerten gefüllt, könnten bei Bedarf aber zusätzlich definiert werden

```
{"etl_step":"update_fin_zusa
tzmerkmale","prop_name":"
```

DOQUERY-Step

Einen DOQUERY-Step, der einen einzelnen SQL-Befehli ausführt legt man folgendermaßen an:

Innerhalb der etl_steps Definition mach man einen Eintrag mit dem etl_job und einem eindeutigen uniquename, der später die SQL-Datei referenziert, Typ ist "DOQUERY".

```
<#assign  etl_steps = [
{"etl_job":"meinJob",
```

Innerhalb der etl_step_properties müssen für den DOSQL-Step der folgende Eintrag definiert werden

```
{"etl_step":"update_test","pr
op_name":"sql",
```

Die folgenden Parameter werden vom Script automatisch mit defaultwerten gefüllt, könnten bei Bedarf aber zusätzlich definiert werden

```
{"etl_step":"update_fin_zusa
tzmerkmale","prop_name":"
```

MSG-Step

wird von HisInOne 2024.12 noch nicht unterstützt

komplettes Beispiel

```
--Freemarker  Template
```

Fehlermeldungen FAQ

Cannot invoke "javax.sql.DataSource.unwrap(java.lang.Class)" because the return value of de.superx.spring.batch.reader.JdbcUnloadReader.getDataSource()" is null

bedeutet,dass bei einem UnloadStep als dataSource z.B. "mbs" angegeben wurde, aber in der databases.xml (oder Spezialversion davon) keine Datenquelle "mbs" definiert ist.

ERROR: null value in column "step_type_id" of relation "etl_step" violates not-null constraint

Kontrollieren Sie, ob die Tabelle etl_step_type gefüllt ist (s.o.).