Daten sichten, bereinigen und integrieren mit Apache Hop

Ein Arbeitsblatt für Interessierte und Studierende in Data Engineering und Data Science

1. Einleitung

Ziele

Dieses Arbeitsblatt zeigt, wie die Desktop-Anwendung Apache Hop verwendet werden kann, um

Daten zu lesen, zu bereinigen, zu filtern, zusammenzuführen und zu speichern.

Die Aufgaben bestehen aus realistischen Beispielen, die die Arbeit mit heterogenen Daten beinhalten. Es soll gezeigt werden, wie man mit diesen Daten umgeht, wenn man mit Problemen aus diesem Bereich konfrontiert wird.

Die Ziele dieses Arbeitsblattes sind:

- **Apache Hop** verstehen und bedienen können.
- Einfache Abläufe zur Datenverarbeitung in Apache Hop erstellen.
- Mithilfe von Apache Hop mehrere Datenquellen zusammenführen.
- Spaltenwerte transformieren und in einer neuen Spalte speichern.

Zeitplanung

Ungefähr eine Stunde für den Leseteil (ohne Aufgaben), zusätzlich etwas über eine Stunde für die Aufgaben - beide Angaben können je nach Wissensstand abweichen.

Voraussetzungen

Um das Arbeitsblatt durcharbeiten zu können, brauchen Sie folgende Dinge:

- Internetzugang zum Herunterladen von der benötigten Software und Daten.
- Java Version 17, verfügbar bei OpenJDK (Aufgrund von rechtlichen Unsicherheit raten öffentliche Stellen von der Nutzung von Oracle Java ab)
- Software: Apache Hop Version 2.12 (oder neuer), verfügbar für Windows, Mac und Linux, und zu installieren wie unten beschrieben.
- Daten: Datei "Daten_OpenRefine.zip" von OpenSchoolMaps

Folgende Themen können als Vorbereitung auf dieses Thema hilfreich sein:

- Grundlegendes Verständnis im Umgang mit Daten.
- Das Arbeitsblatt "Daten sichten, bereinigen und integrieren mit OpenRefine" auf OpenSchoolMaps.

Installation von Apache Hop

Apache Hop kann auf der gleichnamigen Website heruntergeladen werden. Wichtig ist anzumerken, dass **Java 17** benötigt wird, um die Applikation zu starten. Auf der Seite wird sowohl der Sourcecode als auch das fertige Programm (Binaries) angeboten. Zum Ausführen der Applikation werden nur die Binaries benötigt. Die heruntergeladene Datei enthält alles, um Apache Hop auf den gängigen Betriebssystemen zu starten.

Wenn Sie auf Probleme stossen sollten, gibt es GitHub Discussions, welche eine erste Anlaufstelle sein können.

Struktur des Arbeitsblatts

Das folgende Arbeitsblatt ist in die folgenden Kapitel eingeteilt:

- *Einführung*: Überblick über Apache Hop, dessen Funktionen und Anwendungsbereiche.
- *Benutzeroberfläche (GUI)*: Beschreibung des GUI und deren Elemente zur Erstellung von Pipelines und Workflows.
- *Apache Hop-Funktionen*: Detaillierte Erklärungen zu verschiedenen Transformationsmöglichkeiten und Datenoperationen.
- *Übungen*: Praktische Aufgaben zur Anwendung von Apache Hop, einschliesslich Datenintegration, Bereinigung und Transformation.
- *Abschluss*: Zusammenfassung der erlernten Konzepte und deren praktische Relevanz.
- *Was gelernt wurde*: Überblick über die wichtigsten Lektionen aus den Übungen.
- *Anhang*: Glossar: Die wichtigsten Begrifflichkeiten werden erklärt.

2. Apache Hop-Grundlagen

Apache Hop (**H**op **O**rchestration **P**latform) ist eine Open-Source-Daten-Orchestrierungsplattform, die für die Erstellung von Datenintegrationsprozessen verwendet wird. Die Applikation wurde in Java basierend auf einer plattformunabhängigen und modularen Architektur entwickelt. Somit kann die Applikation flexibel mit Plugins für einen bestimmten Anwendungsfall erweitert werden.

Hop setzt dabei auf eine visuelle Benutzeroberfläche, um die Abläufe möglichst verständlich zu veranschaulichen. Es können dabei sowohl lokale Dateien, als auch externe Datenbanken zum Auslesen und Speichern der Daten verwendet werden.

Apache Hop findet Verwendung bei z.B.:

- Datenintegration
- Datenmigration
- Automatisierung von Workflows und Datenprozesse
- Datenbereinigung

Die Hop Engine ist das Herzstück von Apache Hop. Es ist über drei Clients zugänglich: Hop GUI, Hop Run und Hop Server. Hop GUI ist die visuelle Entwicklungsumgebung, in der Datenteams Workflows und Pipelines entwickeln, testen, ausführen und debuggen. Hop Run ist die Kommandozeilenschnittstelle (CLI) zur Ausführung von Workflows und Pipelines. Hop Server ist ein schlanker Webserver zur Ausführung von Workflows und Pipelines als Webservices (HTTP REST API).

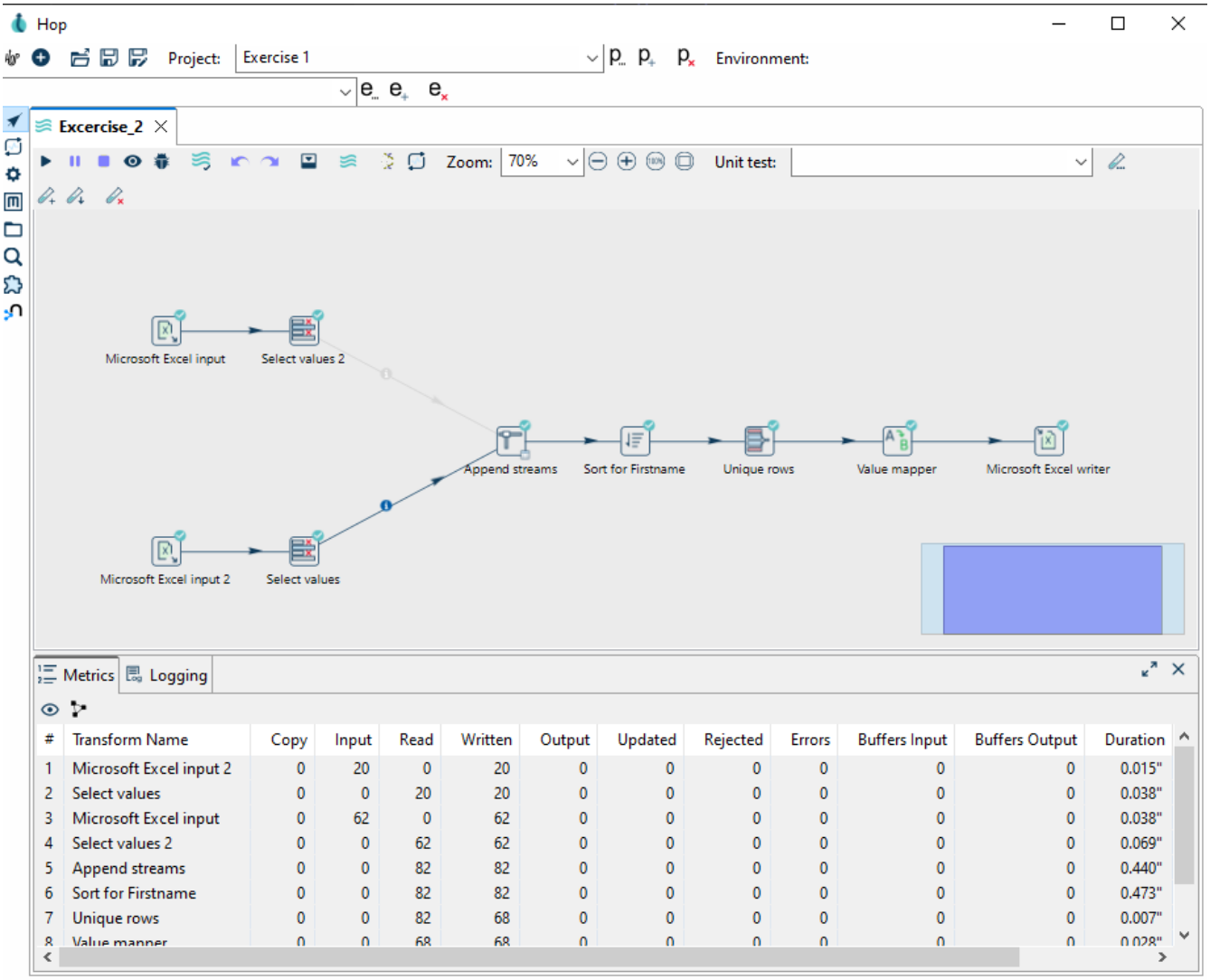
Die Benutzeroberfläche (GUI)

Apache Hop bietet eine grafische Benutzeroberfläche (GUI), in der die Prozesse erstellt und dargestellt werden. Sie besteht aus einem Hauptfenster, das als Arbeitsfläche dient. Hier wird das aktuelle Projekt und dessen Inhalt angezeigt.

In Apache Hop sind Transforms, Actions und Hops zentrale Konzepte zur Verarbeitung und Automatisierung von Daten.

- **Transforms** sind die einzelnen Verarbeitungsschritte innerhalb einer Pipeline. Sie übernehmen Aufgaben wie das Laden, Umwandeln oder Speichern von Daten.
- **Actions** sind die Bausteine eines Workflows und steuern Abläufe, z. B. das Starten einer Pipeline, das Senden einer Benachrichtigung oder den Zugriff auf externe Systeme.
- **Hops** verbinden Transforms und zeigen den Datenfluss in einer Pipeline an. In Workflows definieren Sie die Reihenfolge der Actions.

Diese drei Elemente ermöglichen eine flexible und visuelle Gestaltung von Datenverarbeitungs- und Automatisierungsprozessen in Apache Hop.



#	Transform Name	Copy	Input	Read	Written	Output	Updated	Rejected	Errors	Buffers Input	Buffers Output	Duration
1	Microsoft Excel input 2	0	20	0	20	0	0	0	0	0	0	0.015"
2	Select values	0	0	20	20	0	0	0	0	0	0	0.038"
3	Microsoft Excel input	0	62	0	62	0	0	0	0	0	0	0.038"
4	Select values 2	0	0	62	62	0	0	0	0	0	0	0.069"
5	Append streams	0	0	82	82	0	0	0	0	0	0	0.440"
6	Sort for Firstname	0	0	82	82	0	0	0	0	0	0	0.473"
7	Unique rows	0	0	82	68	0	0	0	0	0	0	0.007"
8	Value mapper	0	0	68	68	0	0	0	0	0	0	0.028"

Abbildung 1. Das Apache Hop GUI.

Transforms

Transforms sind mit der Wichtigste Baustein in Apache Hop. Im folgenden wird erklärt wie diese im Kontext der Apache Hop UI verwendet werden. In diesem Fall am Beispeil des `Generate Rows` -Transforms.

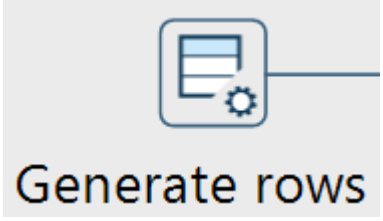


Abbildung 2. Das Generate Rows Transform.

Ein Transform hat immer drei Bereiche, welche angeklickt werden können.

1. Gesamtes Transform-Icon: Öffnet ein allgemeines Konfigrationsfenster, u.a. mit der Option, den gesamten Transform zu benennen, zu editieren oder zu löschen.
2. Transform-Name: Öffnet das für den Transformtyp spezifische Konfigurationsfenster. Der Name kann angepasst werden, muss aber für diese Pipeline einzigartig sein.
3. Vorschau-Mini-Icon: Zeigt die verfügbaren Ausgabezeilen ("Available output rows") als Vorschau an. Das Vorschau-Icon erscheint erst, nachdem der Transform das erste Mal erfolgreich ausgeführt wurde.

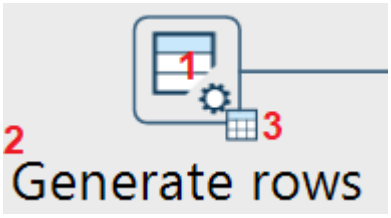


Abbildung 3. Visuelle Darstellung der Klickbaren Flächen eines Transforms.

Verhalten bei Fehlermeldungen von Transforms

Falls zur Laufzeit ein Fehler auftreten sollte, wird dies wie folgt dargestellt:

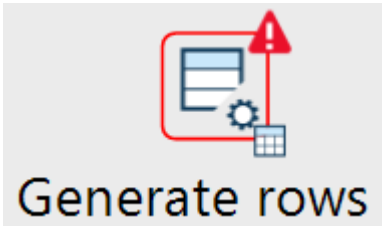
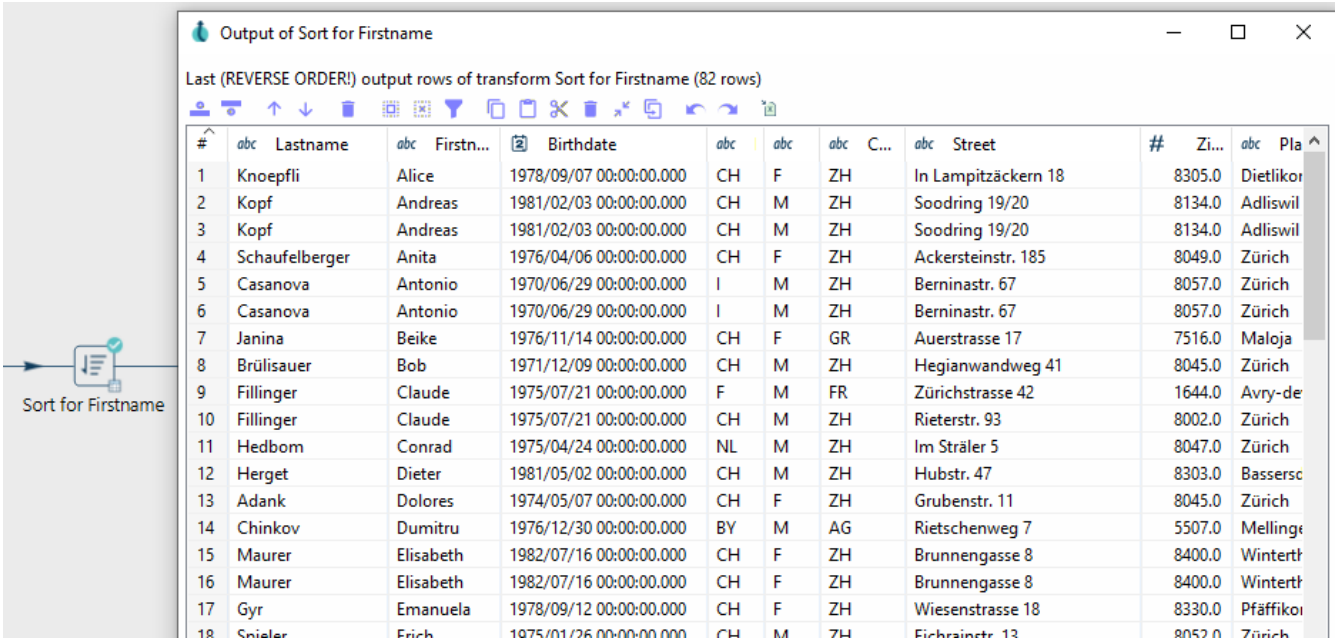


Abbildung 4. Warnsymbol zeigt, dass ein Fehler aufgetreten ist.

In der Toolbar ist die Funktion **[Verify this Pipeline]** zu finden. Wird dieses angeklickt, wird die Fehlermeldung der Transforms angezeigt. Alternativ kann auch das eingebaute Debuggtool verwendet werden. Jedoch sind die Fehlermeldungen, welche man erhält, nicht aussagekräftig. Darum empfiehlt es sich, die `Help-Pages` der Transforms genau durchzulesen.



#	Lastname	Firstn...	Birthdate			C...	Street	Zi...	Pla
1	Knoepfli	Alice	1978/09/07 00:00:00.000	CH	F	ZH	In Lampitzäckern 18	8305.0	Dietlikor
2	Kopf	Andreas	1981/02/03 00:00:00.000	CH	M	ZH	Soodring 19/20	8134.0	Adliswil
3	Kopf	Andreas	1981/02/03 00:00:00.000	CH	M	ZH	Soodring 19/20	8134.0	Adliswil
4	Schaufelberger	Anita	1976/04/06 00:00:00.000	CH	F	ZH	Ackersteinstr. 185	8049.0	Zürich
5	Casanova	Antonio	1970/06/29 00:00:00.000	I	M	ZH	Berninastr. 67	8057.0	Zürich
6	Casanova	Antonio	1970/06/29 00:00:00.000	I	M	ZH	Berninastr. 67	8057.0	Zürich
7	Janina	Beike	1976/11/14 00:00:00.000	CH	F	GR	Auerstrasse 17	7516.0	Maloja
8	Brülisauer	Bob	1971/12/09 00:00:00.000	CH	M	ZH	Hegianwandweg 41	8045.0	Zürich
9	Fillinger	Claude	1975/07/21 00:00:00.000	F	M	FR	Zürichstrasse 42	1644.0	Avry-de
10	Fillinger	Claude	1975/07/21 00:00:00.000	CH	M	ZH	Rieterstr. 93	8002.0	Zürich
11	Hedbom	Conrad	1975/04/24 00:00:00.000	NL	M	ZH	Im Sträler 5	8047.0	Zürich
12	Herget	Dieter	1981/05/02 00:00:00.000	CH	M	ZH	Hubstr. 47	8303.0	Bassersc
13	Adank	Dolores	1974/05/07 00:00:00.000	CH	F	ZH	Grubenstr. 11	8045.0	Zürich
14	Chinkov	Dumitru	1976/12/30 00:00:00.000	BY	M	AG	Rietschenweg 7	5507.0	Melling
15	Maurer	Elisabeth	1982/07/16 00:00:00.000	CH	F	ZH	Brunnengasse 8	8400.0	Winterth
16	Maurer	Elisabeth	1982/07/16 00:00:00.000	CH	F	ZH	Brunnengasse 8	8400.0	Winterth
17	Gyr	Emanuela	1978/09/12 00:00:00.000	CH	F	ZH	Wiesenstrasse 18	8330.0	Pfäffiko

Abbildung 5. Anzeige des Ergebnisses nachdem Apache Hop eine Pipeline ausgeführt hat.

Pipelines können auch mithilfe von **Workflows** zu grösseren Konstrukten zusammengeführt werden. Hier werden verschiedenen Pipelines, auch **Action** genannt, ebenfalls mit **Hops** verbunden. Diese stellen in diesem Fall jedoch die sequenzielle Ausführung der einzelnen Actions dar.

In Apache Hop unterschiedet man zwischen Workflows 

und Pipelines 

. In den Pipelines wird definiert, was gemacht werden soll. Diese Pipelines können dann wiederum zu grösseren Workflows zusammengestellt werden.

Projekte

Projekte sind in HOP die Grundlage für Aktionen jeglicher Art. In einem Projekt können auch Projektübergreifende **Environments** gesetzt werden.

Environments in Apache Hop sind vordefinierte Konfigurationen, die es ermöglichen, Projekte in verschiedenen Kontexten auszuführen, ohne manuelle Anpassungen vornehmen zu müssen. Sie enthalten spezifische Einstellungen wie Variablen, Verbindungsdetails oder Pfade und können beispielsweise für Entwicklungs-, Test- und Produktionsumgebungen genutzt werden. So kann eine Pipeline in der Entwicklungsumgebung mit lokalen Dateien arbeiten, während Sie in der Produktionsumgebung automatisch eine Datenbank als Quelle nutzt. Environments helfen dabei, Workflows flexibel und wiederverwendbar zu gestalten.

Zudem wird jedem Projekt ein eigener Projektordner zugewiesen, in welchem die Dateien des Projekts gespeichert werden. Apache Hop ändert in der Regel nicht die original Daten, ausser es wird in der Pipeline so konfiguriert.

3. Apache Hop-Funktionen

Apache Hop bietet eine Vielzahl leistungsstarker Funktionen zur effizienten Steuerung des gesamten Datenflusses. Dabei können Daten nicht nur einfach eingelesen und exportiert werden, sondern auch in komplexen Workflows transformiert, zusammengeführt und validiert werden. Die modulare und visuelle Oberfläche ermöglicht eine flexible Integration und Anpassung der Prozesse, was insbesondere beim Import von Daten, der Kombination von Datenströmen und der Bereinigung von Datensätzen entscheidende Vorteile bietet. Im Folgenden werden einige dieser Funktionen näher beleuchtet.

Ein- und Ausgaben

In Apache Hop gibt es eine grosse Auswahl an In- sowie `Output`-Transforms, welche zum Einlesen und Schreiben von Daten genutzt werden können. Im Folgenden wird auf ein paar ausgewählte Transforms eingegangen.

Excel Input

Mit dem `Microsoft Excel Input` können Excel Dateien direkt als Datenquelle in eine Pipeline eingebunden werden.

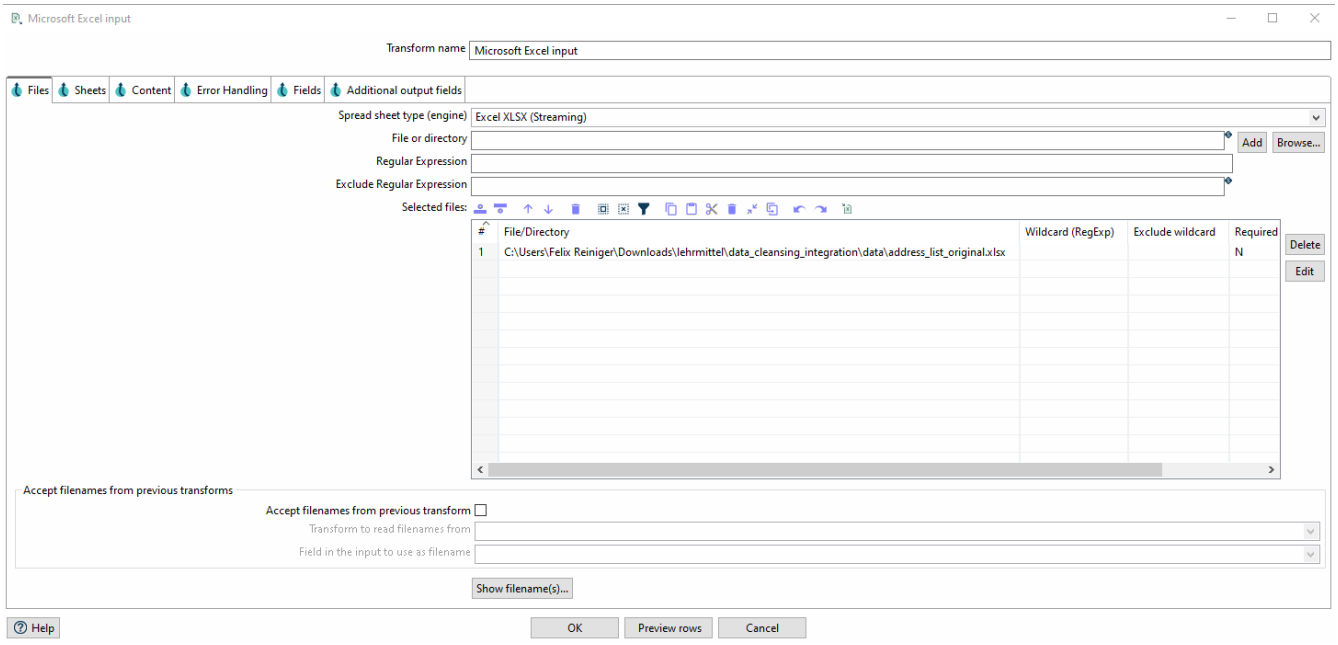


Abbildung 6. Das Parameter-Menu des Excel Input Transforms

Im Tab **Files › Browse..** muss zunächst die Datei referenziert werden und anschliessend mit einem Klick auf **[Add]** in die Liste der **Selected Files** hinzugefügt werden.

Im Anschluss kann im Tab **Sheets › List of sheets** unter **[Get sheetnames(s)...]** die entsprechenden `Sheets`, welche ausgelesen werden sollen, ausgewählt werden. Abschliessend kann im Tab **Fields › Define fields schema:** via **[Get fields from header row...]** der Header eingelesen werden. Nun können, wenn gewünscht die Datentypen der einzelnen Spalten angepasst werden.

Excel Writer

Mit dem `Microsoft Excel Writer` kann ein Datenfluss als Excel Datei (`.xlsx`, `.xls`) abgespeichert werden. In den Parametern muss entsprechend ein Ort zum Abspeichern der Datei definiert werden.

Transforms

Apache Hop ist besonders stark beim Transformieren von Daten. Dank der visuellen Darstellung lassen sich Pipelines effizient zusammenstellen und analysieren. Da Hop flexibel aufgebaut ist, lässt es sich auch gut in bestehende Abläufe und Infrastrukturen einbinden. Im Folgenden werden einige der verfügbaren Transformationen aufgelistet und erklärt.

Zusammenführung von Datensätzen aus verschiedenen Quellen

Apache Hop bietet verschiedene Funktionen zum Zusammenführern von verschiedenen Daten. Je nachdem was erreicht werden soll, können unterschiedliche Transforms zur Anwendung kommen.

Append Streams Transform

Der `Append Streams` Transform kann genutzt werden, um einen Datenfluss an einen anderen anzuhängen. In den Parametern ist die Rede von `Tail` und `Head`. Der `Tail` wird dabei am Ende von `Head` angesetzt. Das Schema, d.h. die Spalten der Tabellen und deren Datentypen, der beiden Quellen muss identisch sein, ansonsten kommt es zu einem Fehler.

Merge Join Transform

Der `Merge Join` Transform in Apache Hop ermöglicht das Verknüpfen von zwei Datenströmen anhand eines gemeinsamen Schlüssels. Dabei wird ein Join-Typ wie `INNER`, `LEFT OUTER` oder `FULL OUTER` gewählt, ähnlich wie bei SQL-Joins. Wichtig ist, dass die Eingabedaten vorab sortiert sind, da Merge Join keine automatische Sortierung durchführt. Nach der Verknüpfung werden die kombinierten Datensätze als ein einzelner Datenstrom weiterverarbeitet.

Validierung und Deduplizierung

Sorting Transform

Der `Sorting` Transform sortiert einen Datenfluss alphabetisch/nummerisch auf oder absteigend. In den Parametern müssen die entsprechenden Spalten ausgewählt werden. Es können auch mehrere Spalten ausgewählt werden. In diesem Fall ist die Reihenfolge, in welcher die Spalten angegeben werden, entscheidend.

Unique Rows Transform

Mit dem `Unique Rows` Transform können Duplikate entfernt werden. In den Parametern können für das Überprüfen relevanten Spalten definiert werden. Jedoch funktioniert dieser Transform nur bei sortierten Datenflüssen. Dementsprechend muss zuvor ein `Sorting` Transform angewendet werden.

Value Mapper Transform

Der `Value Mapper`-Transform wird verwendet, um Werte innerhalb eines Datenstroms umzuwandeln. Er ermöglicht es, bestimmte Eingabewerte durch definierte Zielwerte zu ersetzen.

Um ihn zu nutzen, füge in einer Pipeline einen `Value Mapper`-Transform hinzu. Öffne die Parameter und trage unter `Source value` den ursprünglichen Wert und unter `Target value` den gewünschten Zielwert ein. Nach dem Speichern werden die definierten Werte beim Durchlaufen der Pipeline automatisch ersetzt. Dies ist besonders nützlich für Datenbereinigungen oder die Standardisierung von Werten.

Übung 1: Generate Rows

In dieser Übung werden Sie eine erste Pipeline mit Apache Hop erstellen. Ziel ist es eine Tabelle mit 10 Tiernamen zu erstellen.

Schritt 1: Projekt erstellen

Um ein Projekt - den eigentlichen Arbeitsbereich, in dem mit den Daten gearbeitet wird - zu erstellen, müssen Sie folgendes tun:

1. Starten Sie Apache Hop auf Ihrer lokalen Maschine.
2. Drücken Sie in der Top-Bar auf **[Add a new Project]**.
3. In dem neuen Fenster müssen Sie dem Projekt einen Namen geben und einen `Home folder` definieren. Mit **[OK]** können Sie die Eingabe bestätigen.
4. Anschliessend erscheint ein neues Fenster, in welchem Sie Environment-Variablen festlegen können. Diese können Sie mit **[No]** wegklicken.

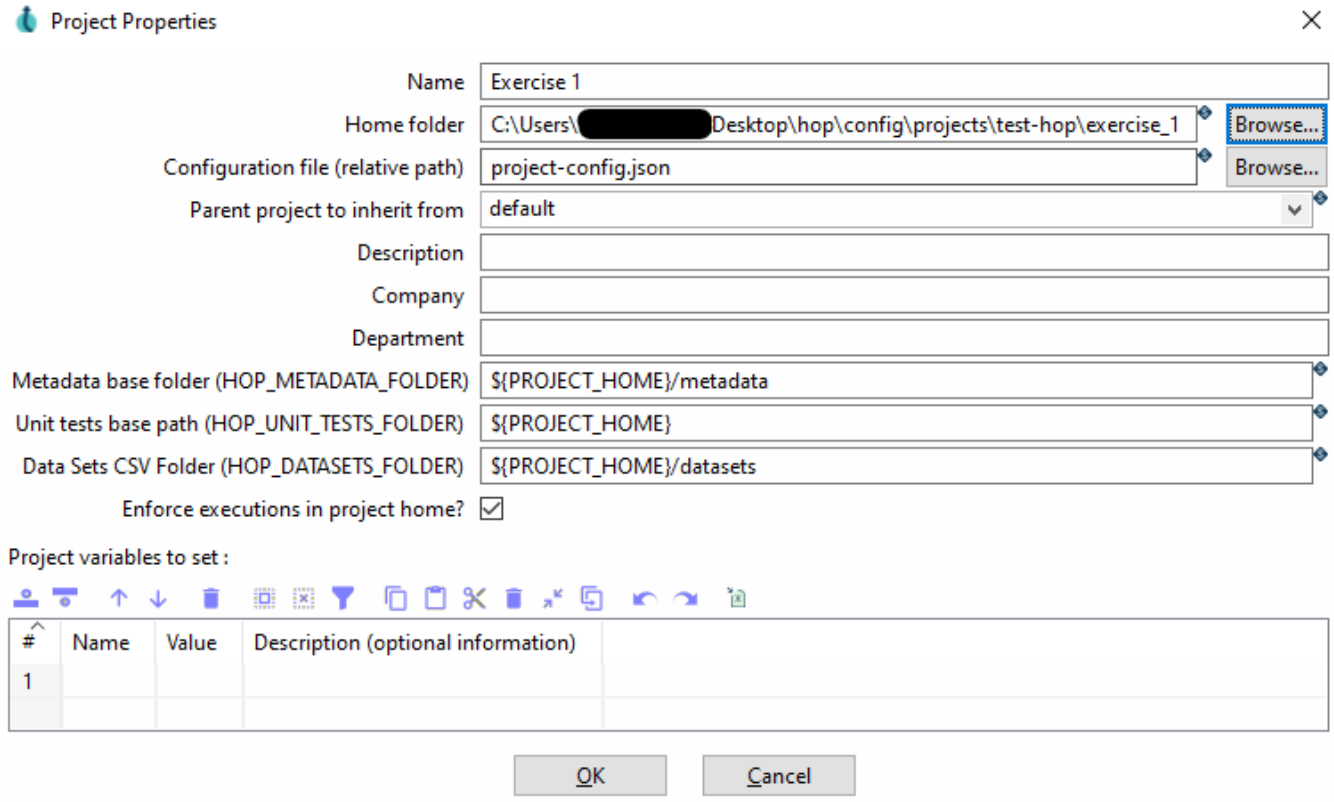


Abbildung 7. New Projects.

Schritt 2: Pipeline bauen

Hierzu kann der `Generate Rows`-Transform verwendet werden, welche anschliessend mit einem `Fake Data`-Transform mit Daten versehen werden kann.

Abbildung 8. Beide Transforms werden miteinander vebunden.

Anschliessend kann der `Fake Data`-Transform entsprechend konfiguriert werden.

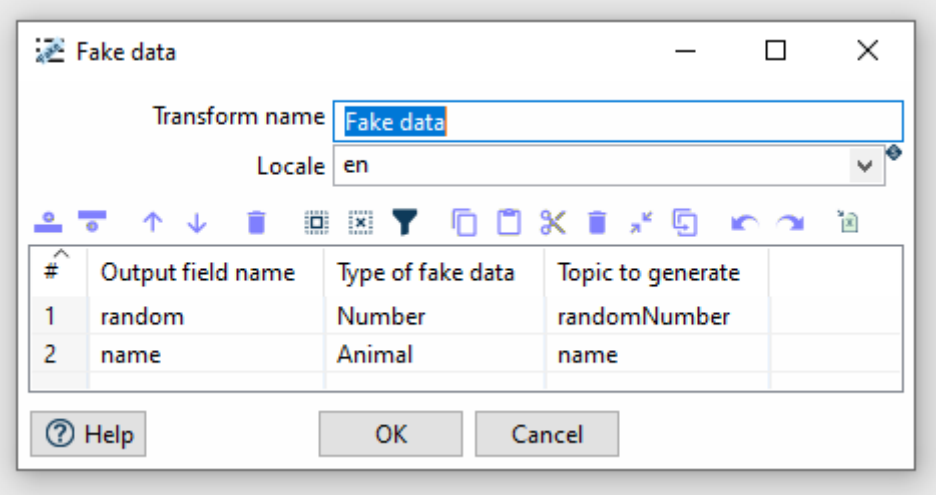


Abbildung 9. Konfiguration des Transforms "Fake Data" mit Zufallszahlen und Tiernamen.

Anschliessend kann die Pipeline ausgeführt werden und die Daten inspiziert werden. Die Aufgabe ist beendet, wenn die Tabelle 10 Zeilen mit einer zufälligen Zahl und Tiernahmen anzeigt.

Übung 2: Bestehende Daten Transformieren

In dieser Übung werden Sie eine simple Pipeline mit Apache Hop bauen. Eine Excel-Datei wird eingelesen, gefiltert, die Daten werden bereinigt und schliesslich wieder gespeichert.

Daten

Für diese Aufgabe werden die Daten aus der Datei `address_list_original.xlsx` verwendet. Diese ist Teil des Zip-Archivs `Daten_OpenRefine.zip`, das Sie von OpenSchoolMaps herunterladen können (gleicher Abschnitt wie dieses Arbeitsblatt).

Schritt 1: Pipeline erstellen

Wenn Sie alle Dateien heruntergeladen haben, erstellen Sie eine neue Pipeline, indem Sie auf das **[+]** in der Top-Bar klicken und **[Pipeline auswählen]**.

Schritt 2: Daten laden und überprüfen

Filtern Sie den Datensatz, um nur Kunden aus dem Kanton Zürich herauszufiltern.

1. Laden Sie die Daten aus der Datei `address_list_original.xlsx` mit Hilfe von `Microsoft Excel Input`.
2. Wenden Sie den Transform `Filter rows` auf die Spalte `Kanton` an, um nur Kunden zu filtern, die im Kanton `ZH` wohnen.
3. Nutzen Sie den gesonderten Transform `Standardize phone number`, um die Landesvorwahl hinzuzufügen und die Telefonnummer in ein einheitliches Format zu bringen.

`Standardize Phone Number` ist ein Transform, der beispielhaft für durch Plugins hinzugefügte Transforms ist. Alternativ kann man auch eine `String Operation` verwenden, um die Daten in die gewünschte Form zu bringen.

Schritt 3: Daten Exportieren

Nachdem Sie die oben genannten Aufgaben erledigt haben, können Sie einen Export zur Pipeline hinzufügen. Verwenden Sie hierfür den `Microsoft Excel Writer`. Wählen Sie in den Optionen `xlsx [Excel 2007 and above]` als Format aus und wählen Sie einen passenden Dateinamen.

Schritt 4: Speichern der Pipeline

Wenn die Pipeline erfolgreich ausgeführt wurde, speichern Sie diese unter `simple_data_transform.hpl` in Ihrem Projektordner.

Übung 3: Integrieren eines anderen Datensatzes

In dieser Übung integrieren Sie einen Datensatz in einen vorgegebenen Zieldatensatz. Einige Spaltennamen und Strukturen werden umgestellt, ansonsten bleibt der Datensatz unverändert. Solche Szenarien sind in der Praxis häufig, da Daten aus diversen Quellen zu einem umfassenden Datensatz zusammengeführt werden müssen. Mit Apache Hop setzen Sie verschiedene Techniken ein – etwa das Mapping von Attributen, das Zusammenführen und Aufteilen von Feldern sowie die Deduplizierung – um die unterschiedlichen Datenströme erfolgreich zu vereinheitlichen.

Daten

Für diese Aufgabe werden die Daten aus den Dateien `address_list_original.xlsx` & `address_list_scrambled.xlsx` verwendet.

Schritt 1: Pipeline erstellen

Wenn Sie beide Excel Dateien heruntergeladen haben, erstellen Sie eine neue Pipeline, indem Sie auf das **[+]** in der Top-Bar klicken und **[Pipeline auswählen]**.

Schritt 2: Integration der Daten

Um beide Daten in die Pipeline hereinzuladen, müssen zwei separate `Microsoft Excel Inputs` erstellt werden. Referenzieren Sie jeweils eine Excel-Datei pro Transform. Im Weiteren werden Sie die Daten zunächst in ein einheitliches Format bringen, bevor Sie die beiden Datenflüsse zusammenführen.

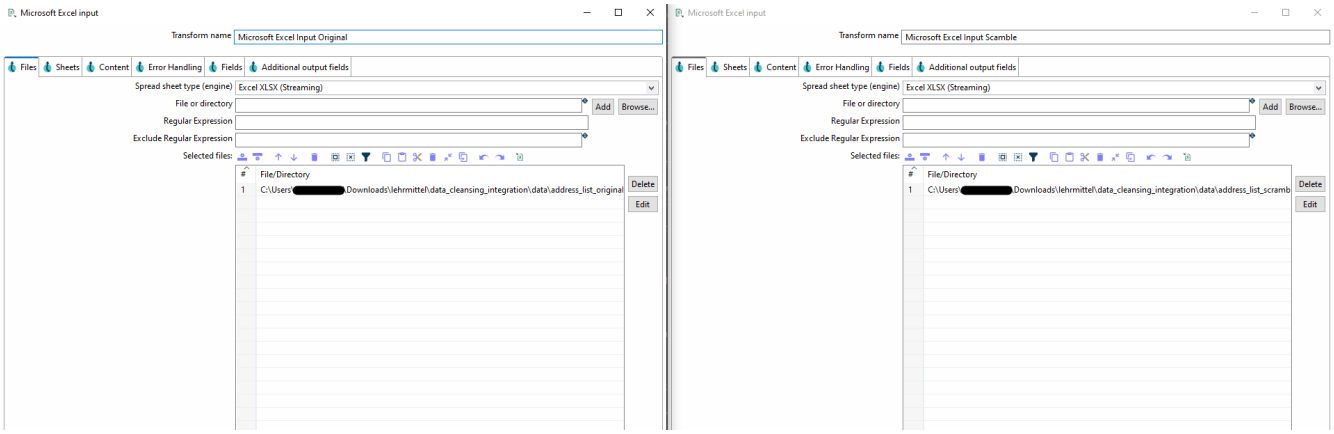


Abbildung 10. Parameter für beide Excel Inputs

Schritt 2.1 Zusammenführung und Standardisierung der Spalten aus beiden Dateien

Zunächst müssen Sie sich die Spalten beider Dateien anschauen und herausfinden, welche Spalten zueinander gehören.

Sobald Sie übereinstimmende Spalten gefunden haben, müssen diese mithilfe von `Select Values` zunächst aufeinander angepasst und anschliessend mit `Append Streams` zusammengeführt werden. `Append Streams` funktioniert nur mit Daten, welche einen identischen Aufbau aufweisen. Sobald die Tabellen zusammengeführt sind, können diese weiterverarbeitet werden.

In den Parametern für `Select Values` können Sie unter `rename` die Spalte umbenennen. Zudem können Sie in `Remove` Spalten wie z.B. `CustID` entfernen, welche nur in einer Tabelle vorkommen.

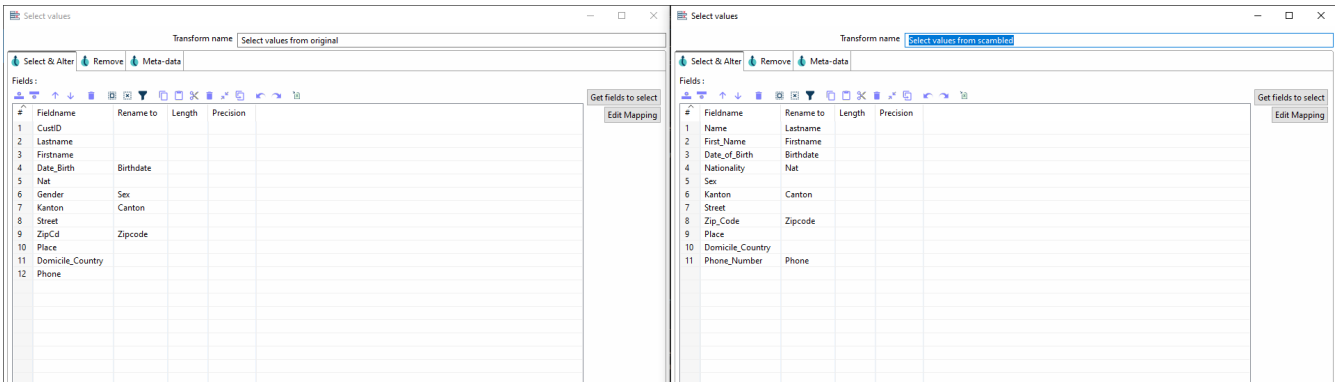


Abbildung 11. Select Values Parameter beider Streams.

Wenn beide Streams ein identisches Schema erzeugen, können sie mit `Append Streams` zusammengefügt werden. In den Parametern müssen Sie `head` und `tail` definieren. Die Reihenfolge spielt in diesem Beispiel jedoch keine Rolle.

Wenn Sie nun die Pipeline ausführen und ein Fehler auftritt, kann es gut sein, dass die beiden Streams noch kein identisches Schema haben.

Last (REVERSE ORDER!) output rows of transform Append streams (82 rows)

#	Lastname	Firstn...	Birthdate			C...	Street	Zi...	Place	Domicile_Co...	Phone
13	Ambühler	Urs	1977/07/29 00:00:00.000	CH	M	ZH	Gerechtigkeitsgasse 4	8002.0	Zürich	CH	+4144 327 13 82
14	Vlassidis	Stamatis	1970/12/22 00:00:00.000	GR	M	ZH	Brunnacherstr. 34	8174.0	Stadel b. Niederglatt	CH	+4144 790 05 07
15	Isler	Ruth	1979/11/21 00:00:00.000	CH	F	ZH	Weihermattstrasse 48	8902.0	Urdorf	CH	+4144 141 97 32
16	Baillie	Marianne	1980/08/11 00:00:00.000	F	F	TG	Zielweg 5	8580.0	Amriswil	CH	+4171 125 36 56
17	Fillinger	Claude	1975/07/21 00:00:00.000	F	M	FR	Zürichstrasse 42	1644.0	Avry-devant-Pont	FR	+4126 638 78545
18	Garcia	Noël	1940/04/20 00:00:00.000	CH	M	AG	Oberbodenstr. 10	5415.0	Nussbaumen AG	CH	+4156 126 14 33
19	Pabst	Swen	1968/07/16 00:00:00.000	CH	M	VS	Gerbiweg 67	3935.0	Bürchen	CH	+4127 939 45 63
20	Ragginger	Jean-Pierre	1983/09/29 00:00:00.000	CH	M	ZH	Saatlenzelg 24	8050.0	Zürich	CH	+4144 290 10 52
21	Dällenbach	Werner	1977/12/17 00:00:00.000	CH	M	ZH	Pflanzschulstr. 4	8400.0	Winterthur	CH	052 125 93 12
22	Meier	Pia	1976/01/13 00:00:00.000	CH	F	TG	Seestr. 23	8596.0	Scherzingen	CH	071 127 20 38
23	Spieler	Erich	1975/01/26 00:00:00.000	CH	M	ZH	Eichrainstr. 13	8052.0	Zürich	CH	044 753 01 77
24	Herget	Dieter	1981/05/02 00:00:00.000	CH	M	ZH	Hubstr. 47	8303.0	Bassersdorf	CH	044 586 36 92
25	Wernli	Thomas	1975/06/11 00:00:00.000	CH	M	AG	Ziegelrain 18	5000.0	Aarau	CH	062 126 28 47
26	Bosshard	Greti	1976/07/03 00:00:00.000	CH	F	ZH	Im Holzerhurd 11/172	8046.0	Zürich	CH	044 678 95 17
27	Brülisauer	Bob	1971/12/09 00:00:00.000	CH	M	ZH	Hegianwandweg 41	8045.0	Zürich	CH	044 419 72 07
28	Perron	Martin	1976/04/19 00:00:00.000	CH	M	ZH	Segantinistr. 215	8049.0	Zürich	CH	044 253 07 22
29	Schiesser	Rudolf	1971/06/16 00:00:00.000	CH	M	ZH	Hardstr. 29	8004.0	Zürich	CH	044 493 78 67
30	Habegger	Nicka	1978/02/08 00:00:00.000	CH	M	ZH	Ankengasse 3A	8623.0	Wetzikon	CH	044 697 46 82
31	Adank	Dolores	1974/05/07 00:00:00.000	CH	F	ZH	Grubenstr. 11	8045.0	Zürich	CH	044 512 30 32
32	Neff	Otto	1976/02/17 00:00:00.000	CH	M	ZH	Hammerstr. 42	8008.0	Zürich	CH	044 271 58 87
33	Sigot	Yvette	1976/03/15 00:00:00.000	CH	F	TG	Sportplatzstr. 5	8580.0	Amriswil	CH	071 125 43 63
34	Aeppli	Ernst	1973/02/21 00:00:00.000	CH	M	ZH	Auwiesenstr. 45	8050.0	Zürich	CH	044 771 53 42
35	Schaufelberger	Anita	1976/04/06 00:00:00.000	CH	F	ZH	Ackersteinstr. 185	8049.0	Zürich	CH	044 715 98 47
36	Gsell	Ernst	1970/01/04 00:00:00.000	CH	M	ZH	Röschibachstr. 77	8037.0	Zürich	CH	044 808 56 72
37	Walde	Hans L.	1975/12/04 00:00:00.000	CH	M	ZH	Buchholzstr. 110	8053.0	Zürich	CH	044 901 14 97

Abbildung 12. Beide Streams wurden erfolgreich zusammengeführt.

Schritt 2.2 Deduplizierung der Daten

Der nächste Schritt besteht darin, die Daten zu deduplizieren. Einige Daten aus der zweiten Datei sind auch in der ersten Datei vorhanden, was bedeutet, dass es sich um Duplikate handelt, die Sie entfernen und nur einen der Einträge behalten müssen. Es gibt auch andere Daten in der zweiten Datei, die in der ersten Datei nicht vorhanden sind (d.h. keine Duplikate), sodass Sie diese nicht entfernen müssen.

Damit ein Stream mithilfe von `Unique Rows` bereinigt werden kann, muss dieser als erstes sortiert werden. Zum Sortieren können Sie den `Sort`-Transform benutzten.

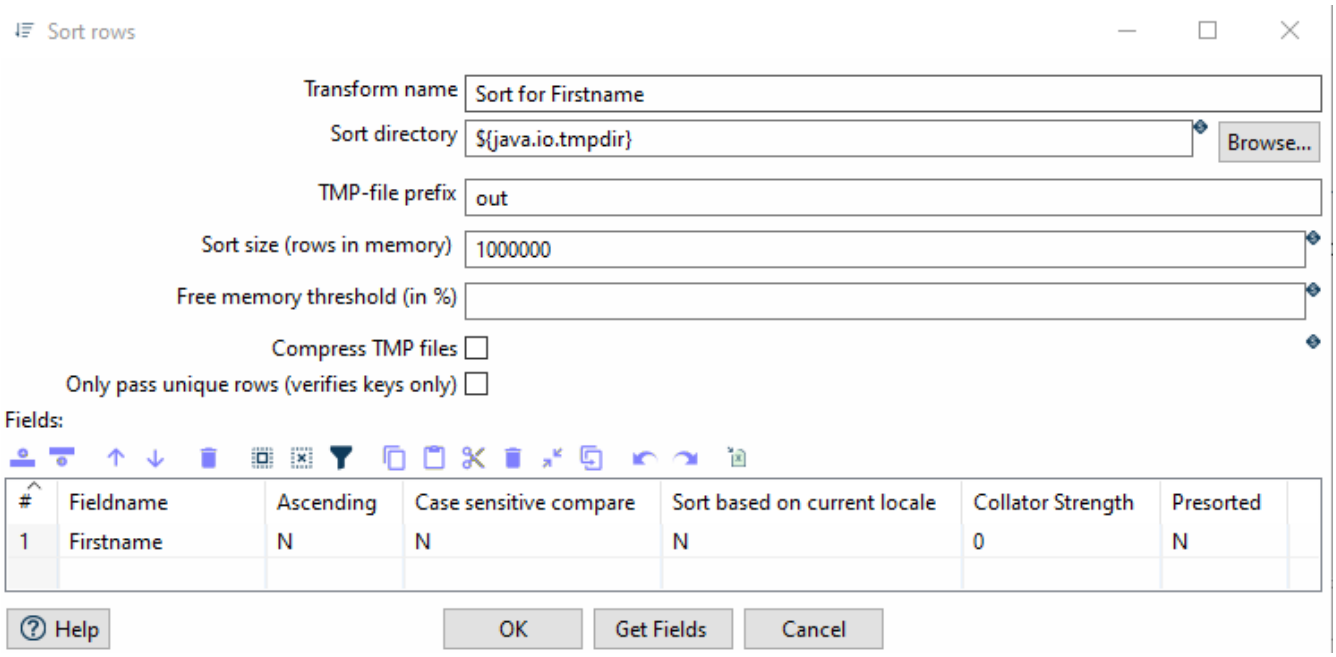


Abbildung 13. Sortierung nach der Spalte `Firstname`.

Hierbei ist wichtig, dass alle Zeilen entsprechend der Spalten, welche zum Identifizieren der doppelten Zeilen genutzt werden, sortiert werden. Dabei ist jedoch egal, ob das auf- oder absteigend ist.

Damit ein Datensatz als doppelt gilt, muss die folgende Bedingung erfüllt sein: Wenn der Vorname, der Nachname und das Geburtsdatum identisch sind, handelt es sich um dieselbe Person, d.h. um ein Duplikat.

Hierzu können wir den `Unique Rows` Transform verwenden, der Duplikate entfernt. Wenn in den Parametern keine bestimmten Spalten angegeben werden, werden alle Spalten überprüft.

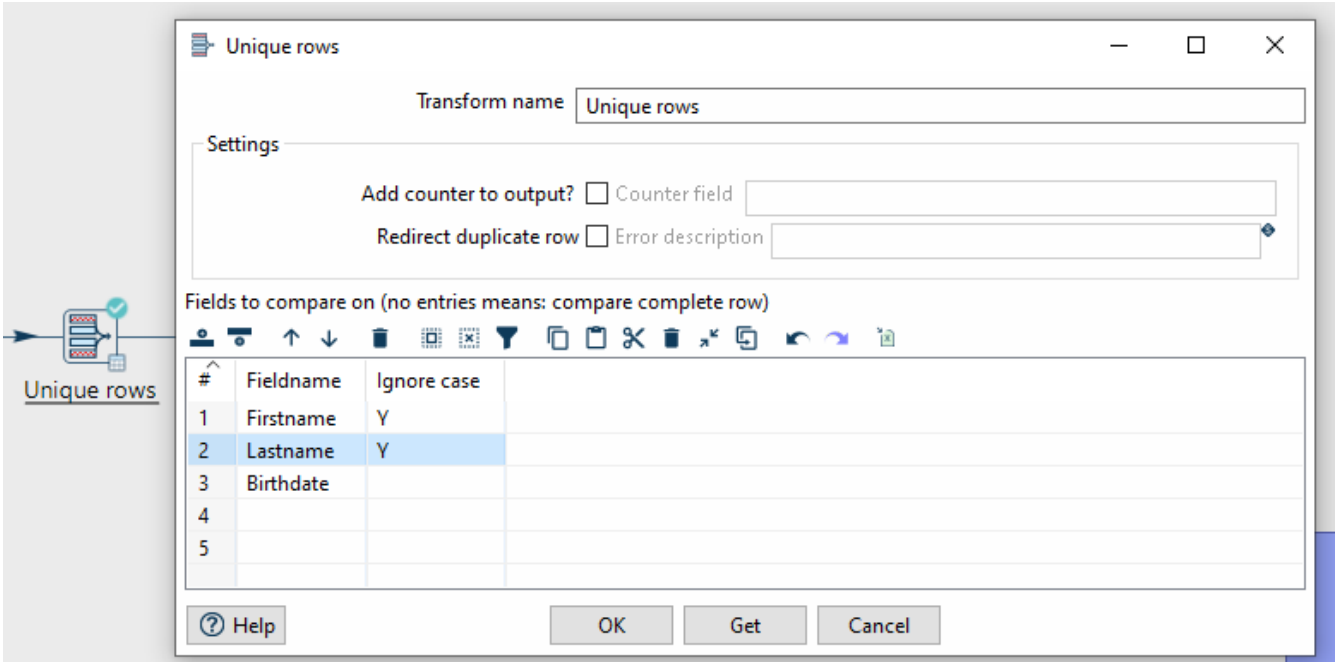


Abbildung 14. Deduplizieren mithilfe vom `Unique Rows` Transform.

Schritt 3: Fertigstellung

In der Spalte `Phone` sind die Telefonnummern in unterschiedlichen Formaten gespeichert. Wenden Sie das Gelernte aus Aufgabe 1 an. Stellen Sie jedoch sicher, dass auch internationale Telefonnummern die korrekte Vorwahl erhalten.

Nehmen Sie allenfalls andere Spalten zur Hilfe.

Nach Abschluss dieser letzten Aufgabe sind die Quelldaten erfolgreich in den Zieldatensatz integriert, validiert und dedupliziert.

Schritt 4: Exportieren der Daten

Fügen Sie einen `Export` Tranform zu Ihrer Pipeline hinzu, um die Dateien im Format *MS Excel 2007+ (.xslx)*, zu exportieren und diese Übung zu beenden.

Schritt 5: Pipeline speichern

Wenn die Pipeline erfolgreich ausgeführt wurde, speichern Sie diese unter `simple_data_merge.hpl` in Ihrem Projektordner.

Übung 4: Value Mapper Transform verwenden

Das Ziel dieser Übung ist es, anhand von den bestehenden Daten eine weitere Spalte zu ergänzen. In diesem Fall soll anhand der Spalte `Sex`/`Gender` eine neue Spalte namens `Salutation` entstehen, welche die entsprechende Anrede enthält.

Daten

Für diese Aufgabe können Sie entweder die Pipeline aus Aufgabe 2 verwenden oder `address_list_original.xlsx` verwenden.

Schritt 1: Projekt anlegen und Daten einlesen (Optional)

Falls Sie mit einer neuen Pipeline arbeiten, müssen Sie zunächst via dem **[+]** eine neue Pipeline anlegen und mithilfe einem `Excel Reader` die Daten aus der Excel-Datei auslesen,

Schritt 2: Value Mapper Transform erstellen

Nutzen Sie einen `Value-Mapper` Transform, um eine neue Spalte namens `Salutation` zu erstellen, welche je nachdem, welches Geschlecht angegeben wurde `M` → `Herr` oder `F` → `Frau` einfügt.

Schritt 3: Daten exportieren

Um diese Aufgabe abzuschliessen, müssen die Daten mit Hilfe eines "Excel Writer" Transform in das Format *MS Excel 2007+ (.xslx)* exportiert werden.

4. Abschluss

Anhand der besprochenen Punkte wurde aufgezeigt, dass Apache Hop ein leistungsstarkes Tool für die Verarbeitung und Aufbereitung von Daten ist. Mit der intuitiven und visuellen Umgebung lassen sich komplexe Abläufe einfach aufzeigen und nachvollziehen lassen.

Für die Arbeit mit Daten und die Umwandlung von Daten in etwas Sinnvolles und Nützliches ist eine Menge logisches, technisches und praktisches Wissen erforderlich. In den Übungen haben Sie hoffentlich gelernt, wie Sie mit unübersichtlichen Daten umgehen und welche Funktionen und Werkzeuge Sie bei verschiedenen Datenproblemen wählen können.

5. Was gelernt wurde

- Erstellen eines Projekts in Apache Hop.
- Transformieren eines Datensatzes mit Apache Hop, unter Verwendung von verschiedenen Transforms.
- Verwendung von Apache Hop zur Bereinigung/Duplizierung und Integration eines Datensatzes in einen anderen.
- Nutzung eines `Value Mapper` Transforms, um eine neue Spalte zu erstellen.

Anhang A: Glossar

Tools

Apache Hop bietet mehrere Tools, um Datenintegrationsprozesse zu modellieren, auszuführen und zu verwalten. Die wichtigsten sind:

- **Hop Gui**: Hauptbenutzeroberfläche zur grafischen Erstellung von Pipelines und Workflows. Bei einem Einstieg, das einzige Tool, das benötigt wird.
- **Hop Run**: Kommandozeilentool zum Ausführen von Workflows und Pipelines.
- **Hop Conf**: Konfigurationstool zur Verwaltung von Projekten und Umgebungen.
- **Hop Server**: Leichtgewichtiger Webserver zur Remote-Ausführung von Pipelines und Workflows.
- sowie **weitere Tools**, wie Hop Encrypt (Passwörter), Hop Search (Metadatenobjekte in Projekten), Hop Import (für Kettle)

Workflow

Eine strukturierte Abfolge einer oder mehrerer **Pipelines** und **Actions**. Wird zur Steuerung des **Datenflusses** verwendet, also für Ablauf- und Fehlerlogik, Zeitsteuerung und bedingte Ausführungen. Dient dem **Datenfluss-Management**. (Nicht verwendet in dieser Übung).

Pipeline

Strukturierter Ablauf zur Datenverarbeitung, bestehend aus einzelnen Transforms, verbunden durch **Hops**. Dient der eigentlichen **Datenverarbeitung**: Laden, Transformieren und Schreiben von Daten. Kann parallelisiert werden und läuft datensatzorientiert.

Transform

Ein Verarbeitungsschritt innerhalb einer Pipeline (z. B. Lesen, Filtern, Umwandeln von Daten). Führt Operationen wie Filtern, Umbenennen, Joinen oder Konvertieren durch. Arbeitet auf **tabellarischen Daten** und ändert ggf. Schema oder Format. Teil der **Datenverarbeitung**.

Hop (innerhalb Pipelines)

Verbindet **Transforms** in Piplines, **Actions** in Workflows und steuert den Fluss der Daten von einem Schritt zum nächsten.

Action

Eine Operation innerhalb eines Workflows – z. B. zum Starten einer Pipeline oder Senden einer E-Mail. Führt Aufgaben wie das Starten einer Pipeline, das Senden einer E-Mail oder das Warten auf eine Datei aus. Liefert **boolesche** Ergebniswerte (Success/Failure). Führt **keine eigene Datenverarbeitung** durch. (Nicht verwendet in dieser Übung).

Project

Eine logische Sammlung von Pipelines, Workflows, Metadaten und Konfigurationen. Dient der Strukturierung von Hop-Inhalten. Unterstützt **Projektspezifische Konfigurationen** wie Parameter (deklarative Variablen, können auch Standardwerte haben) oder einfache Variablen (Schlüssel-Wert-Paare, die zur Laufzeit verwendet werden).

Environment

Eine Sammlung von Konfigurationswerten, die zur Laufzeit auf ein **Project** angewendet werden. Erlaubt es, Pipelines und Workflows in verschiedenen Kontexten (z. B. Dev, Test, Prod) auszuführen.

Metadaten

Wiederverwendbare Objekte wie Datenbankverbindungen, Dateiformate, Tabellenbeschreibungen oder Benutzerdefinierte Werte. Werden zentral verwaltet und in Pipelines/Workflows referenziert.

Pipeline

Eine *Pipeline* ist ein Ablaufplan für die Verarbeitung von Daten. In Apache Hop beschreibt sie die Reihenfolge von Transformationsschritten (**Transforms**), mit denen Daten eingelesen, verändert, analysiert und gespeichert werden.

- Pipelines sind typischerweise zeilenbasiert: Jede Datenzeile durchläuft nacheinander die definierten Transforms.
- Sie werden oft für wiederholbare Datenverarbeitung genutzt – z. B. Import → Bereinigung → Export.

File Definition

Eine Metadatenstruktur, die beschreibt, wie eine strukturierte Datei (z. B. CSV) eingelesen oder geschrieben werden soll (z. B. Trennzeichen, Header, Zeichensatz).

Execution Engine

Apache Hop unterstützt mehrere Ausführungsengines, z. B. die lokale Engine oder Apache Beam für skalierbare und verteilte Ausführung.

Hop Web

(zukünftig) geplante Web-Oberfläche zur Steuerung und Modellierung. Derzeit nicht in der stabilen Version enthalten, aber in Entwicklung.

Metadata Injection

Dynamisches Einfügen von Metadaten in Pipelines zur Erstellung von flexiblen, parametrisierbaren Prozessen – z. B. basierend auf Excel- oder Datenbankdefinitionen.

Row

Grundeinheit der Datenverarbeitung in einer Pipeline. Eine Zeile (row) kann aus beliebig vielen Feldern (fields, z.B. Attribut wie `customer_id`, `amount`, `timestamp`) bestehen.

Daten- und Dateitypen

Hier sind die wichtigsten Datei sowie Datentypen aufgelistetm welche in Apache Hop Verwendet werden.

File Ending	Meaning
.hpl	Hop Pipeline (json)
.hwf	Hop Workflow
.json	Configuration files for Projects, environments and Metadata
.env	File for environment variables
.ktr/.kjb	Old Pentaho-Formats

Quelle: Apache Hop Seite

Data types	Description
BigNumber	An arbitrary unlimited precision number
Binary	An array of bytes that contain any type of binary data
Boolean	A boolean value true or false
Date	A date-time-value with milisecond precision
Integer	A signed log 64-bit integer
Internet Adress	InetAddress An Internet Protocol (IP) address
Number	A double precision floating point value
String	A variable ulimited length text encoded in UTF-8 (Unicode)
Timestamp	Allows the specification of fractional seconds to a precision of nanoseconds

Apache Hop enthält ausserdem eine Reihe zusätzlicher komplexer Datentypen (z. B. JSON, Graph, Avro, Geometry), die keine direkte Zuordnung zu Java-Datentypen

> haben. Diese Datentypen funktionieren nur mit bestimmten Transformationen und können nicht in Allzweck-Transformationen verwendet werden. Eine typische Ordnerstruktur für ein Apache Hop Projekt

Anhang B: Best Practices

In diesem Anhang werden einige Best Practices von Apache Hop beschrieben.

Installation von Hop in beschreibbaren Verzeichnissen

Beachten Sie, dass Apache Hop in einem beschreibbaren Verzeichnis installiert werden muss, da Hop unter anderem Logs dorthin schreibt.

Namenskonventionen verwenden

Verwenden Sie einheitliche Namenskonventionen für Dateien, Pipelines, Felder und Variablen, um die Projektverwaltung zu verbessern.

Absolute Pfade vermeiden

Bei Dateinamen absolute Pfade vermeiden und stattdessen relative Pfade mit der Projektvariablen `${PROJECT_HOME}` verwenden. Das verbessert das Verwalten und Teilen der Projektdaten.

Sample-Anzahl bei der Transformation "CSV file input"

Beim Importieren von CSV-Dateien mit der Transformation `CSV file input` wählen Sie im gleichnamigen Dialog unter `Get Fields` eine grössere Anzahl an Samples als nur die vorgegebenen 100 - am besten so viele wie die Datei enthält oder mehr. Hop berechnet damit die Datentypen und z.B. die String-Längen. Im `Get Fields` Dialog kann dies kontrolliert werden.

Bei CSV String-Längen beachten

Achten Sie beim Import von CSV-Dateien darauf, dass die Länge der String-Felder ausreichend gross ist, da ansonsten längere Strings auf die automatisch erkannte Länge gekürzt werden, was zu Datenverlusten führen kann. Eine Faustregel hierfür ist, die maximale Anzahl der Zeichen auf 60 zu setzen.

Formate vom Datentyp Number überprüfen

Bei numerischen Inhalten ist darauf zu achten, dass das richtige Zahlenformat zugewiesen wird. Andernfalls kann es zu Fehlern kommen. Koordinaten müssen z.B. mit 2 Vorkommastellen und 7 Nachkommastellen angegeben werden.

Datentypen beachten

Datentypen sind für die Datenintegration wichtig. Verwenden Sie ggf. explizite Typkonvertierungen, z.B. mit der Transformation `Select values` (im Tab `Metadaten`).

Felder-Zuordnung beachten

Felder immer explizit zuordnen, da nicht zugeordnete Felder später in der Pipeline zu "Field not found"-Fehlern führen können.

Bei der Transformation "Microsoft Excel Writer" auf `<null>`-Werte achten

Bei der Transformation `Microsoft Excel Writer` sicherstellen, dass numerische Felder keine `<null>`

-Werte enthalten, da diese zu einem Laufzeitfehler führen. (Bug in Hop Version 2.12.0)

Parallele Verarbeitung aktivieren

Manchmal lohnt es sich, die Parallelverarbeitung zu konfigurieren, da sie nicht bei allen Transformationen standardmässig aktiviert ist.

Logging für Fehlerdiagnose aktivieren

Zur besseren Fehlerdiagnose kann die Transformation “Pipeline Logging” erstellt und konfiguriert werden. Die Log-Datei `hopui.log` befindet sich im Verzeichnis `audit`.

Transformationen zur Fehlerbehandlung verwenden

Es gibt Transformationen zur Fehlerbehandlung, um Fehler gezielt abzufangen, statt bei jedem Fehler die gesamte Pipeline zu stoppen: siehe Apache Hop Dokumentation.

Noch Fragen? Sehen Sie auch "Kontakt" auf OpenSchoolMaps!

Frei verwendbar unter CC0 1.0