

# Dokumentation des Datenformats .hej und Anleitung zur Konvertierung in CSV

## Inhalt

Dokumentation des Datenformats .hej und Anleitung zur Konvertierung in CSV .....	1
1. Einführung .....	1
2. Beschreibung des ursprünglichen Datenformats .....	2
2.1 Signatur .....	2
2.1.1 Externe Signatur .....	2
2.1.2 Interne Signatur .....	2
2.2 Dateistruktur .....	2
2.3 Feldtrennzeichen .....	3
2.4 Zeichensatz .....	4
2.5 Datensatzstruktur .....	4
2.5.1 Personendaten .....	4
3. Schritte zur Konvertierung in CSV .....	5
3.1 Ersetzen der Steuerzeichen .....	5
3.2 Einfügen der Kopfzeilen .....	5
3.3 Speichern als CSV-Datei .....	5
4. Datenbankimport und Datentypen pro Spalte .....	5
4.1 Empfohlene Datentypen .....	5
5. Python Script zur Konvertierung nach CSV .....	7

---

## 1. Einführung

Diese Dokumentation beschreibt ein historisches Datenformat mit der Dateiendung .hej, das genealogische Informationen enthält. Ziel ist es, das ursprüngliche Datenformat zu erläutern, die notwendigen Schritte zur Konvertierung in ein CSV-Format darzustellen und Empfehlungen für den Import in eine Datenbank, einschließlich der Datentypen pro Spalte, zu geben. Diese Dokumentation soll zukünftigen Konvertierungen dienen und kann auch für PRONOM bereitgestellt werden.

---

3. **Adop (adop)**: Unbekannt, da in der verfügbaren .hej Datei keine Werte enthalten waren.
4. **Ortsverzeichnis ('ortv')**: Liste der Orte, die in den Personendaten referenziert werden.

```

6705 4047;4046;;;;;;;;;;Eheschliessung;;;;;;;;;;CRLF
6706 4048;4049;;;;;;;;;;Eheschliessung;;;;;;;;;;CRLF
6707 4049;4048;;;;;;;;;;Eheschliessung;;;;;;;;;;CRLF
6708 4050;4051;;;;;;;;;;Eheschliessung;;;;;;;;;;CRLF
6709 4051;4050;;;;;;;;;;Eheschliessung;;;;;;;;;;CRLF
6710 adopCRLF
6711 ortvCRLF
6712 (unbekannt);;;;;;;;;;CRLF
6713 Absam;;;;;;;;;;CRLF
6714 Abtei im Enneberg (I);;;;;;;;;;CRLF
6715 Adliswil (CH);;;;;;;;;;CRLF
6716 Affikon (CH);;;;;;;;;;CRLF
6717 Agram (CRO);;;;;;;;;;CRLF
6718 Alberschwende;;;;;;;;;;CRLF
6719 Altach;;;;;;;;;;CRLF
6720 Altenstadt;;;;;;;;;;CRLF
6721 Altstätten (CH);;;;;;;;;;CRLF
6722 Amerika;;;;;;;;;;CRLF
6723 Amstetten;;;;;;;;;;CRLF
6724 Appenzell (CH);;;;;;;;;;CRLF

```

5. **Quellenverzeichnis ('quellv'):** Liste der Quellen, die für die Daten verwendet wurden.

```

7034 Wolfurt;;;;;;;;;;CRLF
7035 Wollerau (CH);;;;;;;;;;CRLF
7036 Wr. Neustadt;;;;;;;;;;CRLF
7037 Württemberg (D);;;;;;;;;;CRLF
7038 Zell (CH);;;;;;;;;;CRLF
7039 Zell an der Ybbs;;;;;;;;;;CRLF
7040 Zug (CH);;;;;;;;;;CRLF
7041 Zürich (CH);;;;;;;;;;CRLF
7042 Übersaxen;;;;;;;;;;CRLF
7043 quellvCRLF
7044 09;;;;;;;;;;CRLF
7045 31;;;;;;;;;;CRLF
7046 05;;;;;;;;;;CRLF
7047 02;;;;;;;;;;CRLF
7048 25;;;;;;;;;;CRLF
7049 16;;;;;;;;;;CRLF
7050 13;;;;;;;;;;CRLF
7051 08;;;;;;;;;;CRLF

```

## 2.3 Feldtrennzeichen

Die Daten verwenden spezielle Steuerzeichen als Trennzeichen:

- **Feldtrennzeichen:** (ASCII-Code 15, 'Shift In')
- **Notiztrennzeichen:** (ASCII-Code 16, 'Data Link Escape')

Diese Steuerzeichen trennen die einzelnen Felder innerhalb eines Datensatzes bzw. markieren Anmerkungen oder zusätzliche Informationen.

## 2.4 Zeichensatz

ANSI (Windows-1252) oder Latin-1

## 2.5 Datensatzstruktur

### 2.5.1 Personendaten

Ein typischer Personendatensatz sieht wie folgt aus:

100GabrielJohannmrkBahnwächter21051851(unbekannt)03011936FrastanzVerkalkungSohn von Martin Gabriel aus Nenzing u. Katharinae geb. Häusle aus Schlins. - 1 Kind (\* 1888) stirbt klein.84 J 7 M 13 T

Daraus ergeben sich folgende Vermutungen für die enthaltenen Spalten und ihren Inhalt. Achtung es handelt sich um eine Einschätzung/Benennung durch das Liechtensteinische Landesarchiv, die Originale Verwendung Benennung ist unbekannt, da die ursprüngliche Software welche diese Dateien benutzte, nicht verfügbar war:

Datensatz\_ID;Vater\_ID;Mutter\_ID;Nachname;Vorname;Geschlecht;Religion;Beruf;Geburts\_Tag;Geburts\_Monat;Geburts\_Jahr;Geburtsort;Unbekannt1;Unbekannt2;Unbekannt3;Unbekannt4;Unbekannt5;Wohnort;Sterbe\_Tag;Sterbe\_Monat;Sterbe\_Jahr;Sterbeort;Todesursache;Unbekannt6;Unbekannt7;Unbekannt8;Unbekannt9;Unbekannt10;Unbekannt11;Unbekannt12;Unbekannt13;Kommentar;Unbekannt14;Unbekannt15;Unbekannt16;Unbekannt17;Spitznamen;Unbekannt18;Unbekannt19;Unbekannt20;Unbekannt21;Unbekannt22;Unbekannt23;Unbekannt24;Unbekannt25;Unbekannt26

Weiterer Vermutungen:

Unbekannt14 könnte "n" für "nein" bedeuten, aber was genau verneint wird, ist unklar.  
Mögliche Bedeutungen:

Familienstand: "n" für "ledig" oder "nicht verheiratet".

Migration: "n" für "nicht ausgewandert".

Militärdienst: "n" für "nicht gedient".

Besondere Ereignisse: "n" als Indikator für das Fehlen eines bestimmten Ereignisses.

Unbekannt22 Unbekannt23 Unbekannt24

Geburtsjahr, Geburtsmonat, Geburtstag ? Es sind aber weitere Werte wie 0 und ca. enthalten...

Unbekannt25 und Unbekannt26 vermutlich Alter bei Tod – aber warum in zwei Spalten?

---

### 3. Schritte zur Konvertierung in CSV

#### 3.1 Ersetzen der Steuerzeichen

Um die Daten in ein CSV-Format zu konvertieren, müssen die Steuerzeichen durch Standardtrennzeichen ersetzt werden.

1. **Öffnen der Datei in einem Texteditor:** Verwenden Sie einen Editor wie Notepad++, der die Anzeige und das Ersetzen von Steuerzeichen unterstützt.
2. **Ersetzen des Feldtrennzeichens:**
  - **Suchen nach:** Kopieren Sie das Steuerzeichen (ASCII 15).
  - **Ersetzen durch:** Semikolon ; (oder ein anderes geeignetes Trennzeichen).
3. **Ersetzen des Notiztrennzeichens:**
  - **Suchen nach:** Kopieren Sie das Steuerzeichen (ASCII 16).
  - **Ersetzen durch:** Doppelte Anführungszeichen ".

#### 3.2 Einfügen der Kopfzeilen

Fügen Sie am Anfang der Datei eine Kopfzeile mit den Feldnamen ein. Beispiel für die Personendaten:

Datensatz\_ID;Vater\_ID;Mutter\_ID;Nachname;Vorname;Geschlecht;Religion;Beruf;Geburts\_Tag;Geburts\_Monat;Geburts\_Jahr;Geburtsort;Unbekannt1;Unbekannt2;Unbekannt3;Unbekannt4;Unbekannt5;Wohnort;Sterbe\_Tag;Sterbe\_Monat;Sterbe\_Jahr;Sterbeort;Todesursache;Unbekannt6;Unbekannt7;Unbekannt8;Unbekannt9;Unbekannt10;Unbekannt11;Unbekannt12;Unbekannt13;Kommentar;Unbekannt14;Unbekannt15;Unbekannt16;Unbekannt17;Spitznamen;Unbekannt18;Unbekannt19;Unbekannt20;Unbekannt21;Unbekannt22;Unbekannt23;Unbekannt24;Unbekannt25;Unbekannt26

#### 3.3 Speichern als CSV-Datei

1. **Speichern der Datei:** Speichern Sie die bearbeitete Datei mit der Erweiterung .csv (z. B. personendaten.csv).
2. **Import in Excel oder ein anderes Tabellenkalkulationsprogramm:**
  - **Trennzeichen:** Wählen Sie das Semikolon ; als Trennzeichen.
  - **Textqualifizierer:** Stellen Sie sicher, dass Anführungszeichen " als Textqualifizierer verwendet werden.

---

### 4. Datenbankimport und Datentypen pro Spalte

#### 4.1 Empfohlene Datentypen

Bei der Übernahme der Daten in eine Datenbank sollten geeignete Datentypen gewählt werden. Hier sind Empfehlungen für die Personendaten:

<b>Feldname</b>	<b>Datentyp</b>	<b>Beschreibung</b>
Datensatz_ID	INTEGER	Eindeutige Kennung des Datensatzes
Vater_ID	INTEGER	Datensatz_ID des Vaters (NULL, wenn unbekannt)
Mutter_ID	INTEGER	Datensatz_ID der Mutter (NULL, wenn unbekannt)
Nachname	VARCHAR(255)	Familiennamen der Person
Vorname	VARCHAR(255)	Vorname(n) der Person
Geschlecht	CHAR(1)	'm' für männlich, 'w' für weiblich
Religion	VARCHAR(50)	Religion, z. B. 'rk' für römisch-katholisch
Beruf	VARCHAR(255)	Beruf oder Tätigkeit
Geburts_Tag	INTEGER	Tag der Geburt (1-31, NULL, wenn unbekannt)
Geburts_Monat	INTEGER	Monat der Geburt (1-12, NULL, wenn unbekannt)
Geburts_Jahr	INTEGER	Jahr der Geburt (NULL, wenn unbekannt)
Geburtsort	VARCHAR(255)	Ort der Geburt
Unbekannt1	VARCHAR(255)	Zweck noch zu klären
Unbekannt2	VARCHAR(255)	Zweck noch zu klären
Unbekannt3	VARCHAR(255)	Zweck noch zu klären
Unbekannt4	VARCHAR(255)	Zweck noch zu klären
Unbekannt5	VARCHAR(255)	Zweck noch zu klären
Wohnort	VARCHAR(255)	Aktueller Wohnort oder weiterer Wohnort
Sterbe_Tag	INTEGER	Tag des Todes (1-31, NULL, wenn unbekannt)
Sterbe_Monat	INTEGER	Monat des Todes (1-12, NULL, wenn unbekannt)
Sterbe_Jahr	INTEGER	Jahr des Todes (NULL, wenn unbekannt)
Sterbeort	VARCHAR(255)	Ort des Todes
Todesursache	VARCHAR(255)	Ursache des Todes
Unbekannt6	VARCHAR(255)	Zweck noch zu klären
Unbekannt7	VARCHAR(255)	Zweck noch zu klären
Unbekannt8	VARCHAR(255)	Zweck noch zu klären
Unbekannt9	VARCHAR(255)	Zweck noch zu klären
Unbekannt10	VARCHAR(255)	Zweck noch zu klären
Unbekannt11	VARCHAR(255)	Zweck noch zu klären
Unbekannt12	VARCHAR(255)	Zweck noch zu klären
Unbekannt13	VARCHAR(255)	Zweck noch zu klären
Kommentar	TEXT	Zusätzliche Informationen oder Notizen
Unbekannt14	VARCHAR(255)	Zweck noch zu klären
Unbekannt15	VARCHAR(255)	Zweck noch zu klären
Unbekannt16	VARCHAR(255)	Zweck noch zu klären
Unbekannt17	VARCHAR(255)	Zweck noch zu klären
Spitznamen	VARCHAR(255)	Eventuelle Spitznamen
Unbekannt18	VARCHAR(255)	Zweck noch zu klären
Unbekannt19	VARCHAR(255)	Zweck noch zu klären

Feldname	Datentyp	Beschreibung
Unbekannt20	VARCHAR(255)	Zweck noch zu klären
Unbekannt21	VARCHAR(255)	Zweck noch zu klären
Unbekannt22	VARCHAR(255)	Zweck noch zu klären
Unbekannt23	VARCHAR(255)	Zweck noch zu klären
Unbekannt24	VARCHAR(255)	Zweck noch zu klären
Unbekannt25	VARCHAR(255)	Zweck noch zu klären
Unbekannt26	VARCHAR(255)	Zweck noch zu klären

## 5. Python Script zur Konvertierung nach CSV

```
# This scripts reads .hej files containing genealogic data and converts them to csv. 4 csvs are generated
personen, mrg, ortv, quellv. 'adop' seems to be another potential part of these files, but no data was in-
cluded in the investigates .hej file, so no conclusions/conversions can be made for now.
```

```
import csv
import re
import os

# Data folder where input and output files are located
data_folder = 'c:/choose/your/folder'

# Filename containing the raw data
data_file = os.path.join(data_folder, "filename.hej")

# Define control characters
field_delimiter = '\x0f' # Corresponds to character "
note_delimiter = '\x10' # Corresponds to character "

# Lists to store the records
personen_data = []
mrg_data = []
ortv_data = []
quellv_data = []

# Current record type
current_type = 'personen'

# Open the file and read it line by line
```

```

with open(data_file, 'r', encoding='cp1252') as file:
    for line in file:
        line = line.strip()
        if not line:
            continue # Skip empty lines

        # Check for section change
        if line == 'mrg':
            current_type = 'mrg'
            continue
        elif line == 'ortv':
            current_type = 'ortv'
            continue
        elif line == 'quellv':
            current_type = 'quellv'
            continue

        # Process data according to the current type
        if current_type == 'personen':
            # Extract notes and replace them with a placeholder
            notes = re.findall(f'{note_delimiter}(.*?){note_delimiter}', line)
            line_without_notes = re.sub(f'{note_delimiter}.*?{note_delimiter}', '<<NOTE>>', line)

            # Split the record into fields
            fields = line_without_notes.split(field_delimiter)

            # Replace the placeholder with empty strings in fields
            fields = [field.replace('<<NOTE>>', '') for field in fields]

            # Add notes at the correct position if available
            if notes:
                kommentar = ' '.join(notes)
            else:
                kommentar = ""

            # Ensure the number of fields matches the header
            while len(fields) < 49: # Adjust the number as per your header
                fields.append("")

            # Insert the 'Kommentar' field at the correct index (31)
            fields[31] = kommentar

            # Append only the required number of fields
            personen_data.append(fields[:49])

        elif current_type == 'mrg':
            fields = line.split(field_delimiter)
            mrg_data.append(fields)
        elif current_type == 'ortv':
            fields = line.split(field_delimiter)

```



```

        ortv_data.append(fields)
    elif current_type == 'quellv':
        fields = line.split(field_delimiter)
        quellv_data.append(fields)

# Header fields for the CSV files (keep in German)
personen_header = [
    'Datensatz_ID', 'Vater_ID', 'Mutter_ID', 'Nachname', 'Vorname', 'Geschlecht', 'Religion', 'Beruf',
    'Geburts_Tag', 'Geburts_Monat', 'Geburts_Jahr', 'Geburtsort', 'Unbekannt1', 'Unbekannt2', 'Unbe-
kannt3',
    'Unbekannt4', 'Unbekannt5', 'Wohnort', 'Sterbe_Tag', 'Sterbe_Monat', 'Sterbe_Jahr', 'Sterbeort',
    'Todesursache', 'Unbekannt6', 'Unbekannt7', 'Unbekannt8', 'Unbekannt9', 'Unbekannt10', 'Unbe-
kannt11',
    'Unbekannt12', 'Unbekannt13', 'Kommentar', 'Unbekannt14', 'Unbekannt15', 'Unbekannt16', 'Unbe-
kannt17',
    'Spitznamen', 'Unbekannt18', 'Unbekannt19', 'Unbekannt20', 'Unbekannt21', 'Unbekannt22', 'Unbe-
kannt23',
    'Unbekannt24', 'Unbekannt25', 'Unbekannt26', '"', '"', '"
]

# Ensure the number of headers matches the number of fields
assert len(personen_header) == 49, "The number of header fields does not match the number of data
fields."

mrg_header = [
    'Person1_ID', 'Person2_ID', 'Heirats_Tag', 'Heirats_Monat', 'Heirats_Jahr', 'Heiratsort',
    'Feld7', 'Feld8', 'Feld9', 'Feld10', 'Feld11', 'Feld12', 'Ereignisart', 'Feld14', 'Feld15',
    'Feld16', 'Feld17', 'Feld18', 'Feld19', 'Feld20'
]

ortv_header = [
    'Ortsname', 'Feld2', 'Feld3', 'Feld4', 'Feld5', 'Feld6', 'Feld7', 'Feld8', 'Feld9', 'Feld10', 'Feld11'
]

quellv_header = [
    'Quellen_ID', 'Feld2', 'Feld3', 'Feld4', 'Feld5', 'Feld6', 'Feld7', 'Feld8', 'Feld9', 'Feld10'
]

# Function to write data to CSV files
def write_csv(filename, header, data):
    filepath = os.path.join(data_folder, filename)
    with open(filepath, 'w', newline="", encoding='utf-8-sig') as csvfile:
        writer = csv.writer(csvfile, delimiter=';', quotechar='"', quoting=csv.QUOTE_MINIMAL)
        writer.writerow(header)
        for row in data:
            # If the row has fewer fields than the header, fill with empty strings
            if len(row) < len(header):
                row.extend([""] * (len(header) - len(row)))
            writer.writerow(row)

```

```
# Save CSV files
write_csv('personen.csv', personen_header, personen_data)
write_csv('mrg.csv', mrg_header, mrg_data)
write_csv('ortv.csv', ortv_header, ortv_data)
write_csv('quellv.csv', quellv_header, quellv_data)

print("Data has been successfully converted to CSV files.")
```