Datenverarbeitung

Martin Raden

Einführung von

- `cat, head, tail, less, sort, uniq, wc` in Ein erster Blick
- `>, >>, <, /dev/null` in Piping und Streams
- `cut, paste, join` in Datenspalten verarbeiten

Um zu sehen, wo es hingehen soll, gibts die folgenden beiden Videos.

Video: Bashino - #05 Daten sortieren (sort uniq sort) [5 min]

Video: (en) Pedagogy - working with data files | cut command | paste command | join command | colrm command in linux

---

Ein erster Blick

Dateiinhalte betrachten

- `cat` = **gibt** den **gesamten Inhalt** einer Datei **aus** oder mehrerer nacheinander
- `head` = gibt die **ersten Zeilen** aus
  - "`-n XXX`" - Ausgabe bis zur XXX-ten Zeile (die ersten XXX)
- `tail` = gibt die **letzten Zeilen** aus
  - z.B. "`tail -n+2`" - alle Zeilen ab der 2-ten (ersten Zeilen werden ausgelassen)
- `less` = **interaktiver Textbetrachter** der nicht in pipes (oder nur am Ende) verwendet werden kann
  - Taste `h` - Hilfe mit Kommandoübersicht (help)
  - Taste `q` - schliessen des Betrachters (quit)
- `wc` = Zeichen-/Wort-/Zeilenstatistik (word count)
  - "`wc -l`" - **Anzahl Zeilen** (lines) einer Datei

---

erste Dateioperationen

- `sort` = gibt den Inhalt (lexikographisch) **sortiert** aus
  - "`-n`" = Zahlensortierung (numeric)
  - "`-r`" = umgekehrte Reihenfolge (reverse)
  - "`-k`" = Spaltennummer für Sortierung (key) !!! Spaltentrennzeichen mittels "`-t`" beachten/einstellen
  - check man page für weitere Sortieroptionen
  - Beispiele
- `uniq` = **eliminiert doppelte** (nacheinanderfolgende) **Zeilen**
  - i.d.R. in Kombination mit sort verwendet
  - "`uniq -c`" liefert auch Anzahl der Wiederholungen
  - Beispiele

Piping und Streams

- Konsolenausgabe **in Datei umleiten**
  - "`>`" leitet die Ausgabe in eine (danach benannte) Datei um (und *überschreibt diese*!)
  - "`>>`" leitet die Ausgabe in eine Datei um und *hängt die Ausgabe ans Ende* an
  - die Ausgabe von Programmen wird in einem sogenannten "stream" (Datenstrom) geliefert (und i.d.R. direkt in der Konsole angezeigt)
    - dieser Ausgabestrom wird **"standard output stream" (stdout)** genannt
    - genau dieser stdout wird über die obigen Umleitungen in eine Datei geschrieben/angehangen
- Konsolenausgabe **an nächsten Befehl übergeben**
  - Konsolenbefehle sind i.d.R. in der Lage den zu verarbeitenden Inhalt direkt einzulesen (ohne diese explizit aus einer Datei auszulesen)
  - dies ist der **"standard input stream" (stdin)**
  - "`|`" leitet die Ausgabe als neue Eingabe in den darauffolgenden Befehl um (== piping)
- **stdin**-Konsoleneingabe **aus Datei**
  - manche Programme können keine Datei direkt öffnen und auslesen, sondern erwarten die Eingabe via "standard input stream" (stdin)
  - es gibt zwei Möglichkeiten
    - Via `cat` mit Piping: "`cat DATEI | wildesProgramm`"
    - Via `<` Operator: "`wildesProgramm < DATEI`"
- **Beachten:** Ausgabenumleitung gilt erst einmal "nur" für den "standard output stream" der Ausgabe (stdout) in dem normalerweise Ergebnisse geliefert werden. Es gibt aber auch noch einen **"standard error stream" (stderr)**, in welchem Fehlermeldungen ausgegeben werden. Diese sind beim piping weiterhin auf der Konsole sichtbar, es sei den sie werden mittels "`2>`" oder "`2>>`" in eine Datei umgeleitet.
  - *Ggf. irgenwann mal von Interesse:* es ist auch möglich beide streams in den jeweils anderen umzuleiten, um alle Ausgaben auf einmal abzufangen.
- Unerwünschte Ausgabe kann ins "**Datennirvana**" in das dummy file "`/dev/null`" umgeleitet werden, z.B.
  - Standardausgabe (stdout) ignorieren (weil z.B. nur Statusmeldungen) : "`> /dev/null`"
  - Fehlerausgabe (stderr) ignorieren (wenn z.B. "nur" Warnungen etc.) : "`2> /dev/null`"

> Tutorials <

Im Anschluss empfehlen wir das folgende

- Tutorial zu Pipes and Filters von swcarpentry

Datenspalten verarbeiten

- `cut` = **Ausgabe bestimmter Spalten**/Felder/Teile **pro Zeile** = Spaltenextraktion
  - Standardspaltentrennzeichen = Tabulator
  - "`-d`" = Spaltentrennzeichen (delimiter)
  - "`-f`" = Spaltennummer (field)
  - "`-c`" = Buchstabennummer (character) für festbreitenformatierte Dateien
  - Beispiele
- `paste` = **spaltenweises** (horizontales) **Zusammenführen** mehrerer Dateien
  - "`-d`" = Trennzeichen der beiden Dateiinhalte pro Zeile (delimiter); Standardtrennzeichen ist wieder Tabulator

  - i.d.R. sinnvoll, wenn gleiche Anzahl von Zeilen in allen Dateien
  - Beispiele
- `join` = **schlüsselbasiertes Zusammenführen** zweier Dateien
  - d.h. nur Zeilen mit gleichem Schlüsselwert (in der entsprechenden Spalte) werden zusammengeführt
  - "`-t`" = Spaltentrennzeichen (Standard = Leerzeichen)
  - eine Eingabe kann auch von stdin (z.B. via piping kommen); dann muss als entsprechender "Dateiname" ein - angegeben werden

---