What are the Cowboyest Names in Switzerland?

Data Management and Integration Assignment

Marius Furter and Morley J Weston

2022-11-23

Introduction

For our data integration project we want to determine which Swiss family names are the cowboyest. For this we gathered data that allows us to link familiy name frequency with the number of cattle present in a commune (Gemeinde). We made use of the following data sources:

gemeinde.csv

Provides us with the

- commune identification number
- commune name
- commune population

Gathered from the Swiss federal geodata portal at data.geo.admin.ch, and converted from an ESRI shapefile into a CSV in QGIS.

cattle-map-commune.csv

Provides us with

- commune id
- cattle population

Gathered from identitas animal statistics here.

commune-languageregion.csv

Provides us with

- commune id
- commune language region

Gathered from BFS here.

familiy-names-commune.csv

Lists the 100 most frequent family names in each commune Provides us with

- commune id
- family name
- family name count in commune
- family name rank in commune

Gathered from BFS here.

cattle-NamesFemaleCalves.csv and cattle-NamesMaleCalves

Lists most popular names for named calves by language region. Provides us with

- calf name
- name count
- name rank
- language region

Gathered from identitas here.

Database organization

Using the data above we organize our database according to the following ER diagram.

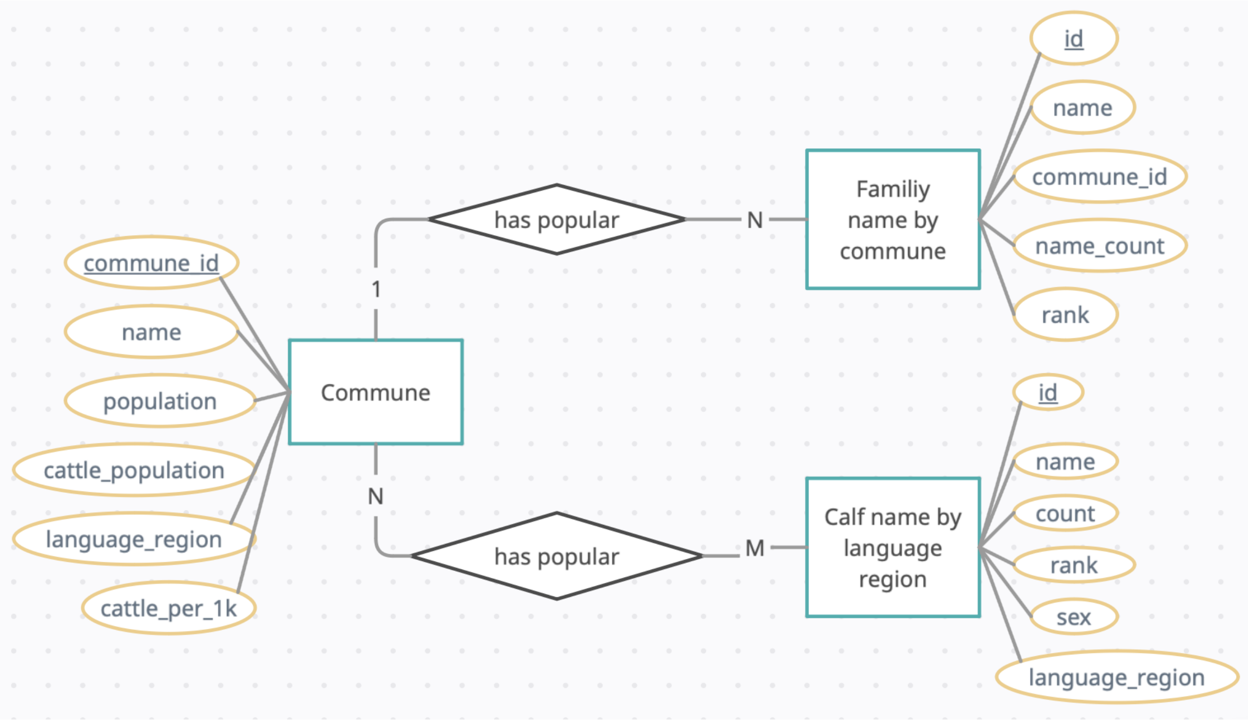

Figure 1: ER diagram of communes and cattle

Each commune is related to the top 100 family names in that commune. Moreover, each commune is related to the top calf names in the language region(s) it belongs to. In reality, no commune is assigned to multiple language regions although this sould be reasonable in principle.

Implementing the database

We start by instantiating a database in SQLite.

```
library(tidyverse)
library(DBI)
library(knitr)
```

```
conn <- dbConnect(RSQLite::SQLite(), "cowboys.db")
```

Commune table

The data in the commune table requires integration from three sources. First we load the commune data from `gemeinde.csv`.

```
communes <- read_csv("input_data/gemeinde.csv")
communes <- communes %>%
  filter(!is.na(EINWOHNERZ)) %>%
  select(BFS_NUMMER, EINWOHNERZ, NAME) %>%
  rename(commune_id = BFS_NUMMER, population = EINWOHNERZ, name = NAME)

communes %>% sample_n(10) %>% kable()
```

commune_id	population	name
382	894	Büetigen
4146	3343	Unterkulm
3215	9646	Rorschach
3862	207	Furna
1	2014	Aeugst am Albis
4228	2923	Boswil
5315	42	Linescio
5464	3348	Vully-les-Lacs
157	4857	Oetwil am See
6643	34898	Vernier

Next we extract the cattle population per commune from `cattle-map-commune.csv`.

```
cattle_population <- read_delim("input_data/cattle-map-commune.csv",
    delim = ";", escape_double = FALSE, trim_ws = TRUE)
cattle_population <- cattle_population %>%
  select(MunicipalityNumber, count) %>%
  rename(commune_id = MunicipalityNumber, cattle_population = count)

cattle_population %>% sample_n(10) %>% kable()
```

commune_id	cattle_population
951	1457
3339	468
4031	130
21	342
3101	2251
5430	439
4536	320
6248	115
1346	2051
5635	0

We then join the cattle population with the commune data using the `commune_id`.

```
communes <- left_join(communes, cattle_population)

communes%>% sample_n(10) %>% kable()
```

commune_id	population	name	cattle_population
2128	317	Châtel-sur-Montsalvens	126
4002	1588	Biberstein	58
2832	721	Ramlinsburg	131
5702	2916	Arzier-Le Muids	400
576	3800	Grindelwald	1859
5654	542	Vullierens	119
3231	7984	Au (SG)	142
5799	2078	Servion	176
2087	1062	Mézières (FR)	1255
497	559	Lüscherz	271

Finally, we extract the language region of each commune from `commune-languageregion.csv`.

```r
commune_language <- read_delim("input_data/commune-languageregion.csv",
    delim = ",", escape_double = FALSE, trim_ws = TRUE) %>%
      select(`Regions-ID`, Sprachgebiete) %>%
      rename(commune_id = `Regions-ID`, language_region = Sprachgebiete)

commune_language$language_region <- recode( commune_language$language_region,
                                        `Deutsches Sprachgebiet` = "de",
                                        `Französisches Sprachgebiet` = "fr",
                                        `Italienisches Sprachgebiet` = "it",
                                        `Rätoromanisches Sprachgebiet` = "rt"
                                          )

commune_language %>% sample_n(10) %>% kable()
```

commune_id	language_region
5520	fr
4091	de
5562	fr
3006	de
6436	fr
2620	de
3105	de
3633	de
3712	de
5180	it

The result is joined to the commune data via `commune_id`.

```r
communes <- left_join(communes, commune_language)

communes%>% sample_n(10) %>% kable()
```

commune_id	population	name	cattle_population	language_region
995	2382	Wiedlisbach	397	de
3001	15649	Herisau	2115	de
959	540	Walterswil (BE)	1128	de
988	1559	Seeberg	1418	de

commune_id	population	name	cattle_population	language_region
7003	4683	Balzers	782	NA
2183	2794	Corminboeuf	444	fr
6778	645	Bure	367	fr
5642	16101	Morges	0	fr
2300	1010	Plasselb	408	de
3311	1876	Amden	1066	de

We calculate the number of cattle per 1000 inhabitants and store this as a new column

```
communes <- communes %>% mutate(cattle_per_1k = cattle_population/population *10^3)
```

We now created a table in the database for the communes with the unique `commune_id` as a primary key.

```
dbExecute(conn, "
  create table if not exists communes (
    commune_id int primary key,
    name text,
    population int,
    language_region text,
    cattle_population int,
    cattle_per_1k real
  )
")
```

```
## [1] 0
```

The joined commune data is then loaded into the database.

```
for (i in 1:nrow(communes)) {
  row = communes[i,]
  dbExecute(conn, "
            insert into communes (commune_id, name, population, language_region,
                                    cattle_population, cattle_per_1k)
            values (?, ?, ?, ?, ?,?)",
          params = c(row$commune_id, row$name,
                     row$population, row$language_region,
                     row$cattle_population, row$cattle_per_1k))
}
```

We check that this has worked properly by querying the `communes` table.

```
dbGetQuery(conn, "select * from communes limit 10") %>% kable()
```

commune_id	name	population	language_region	cattle_population	cattle_per_1k
3762	Scuol	4624	rt	2107	455.666090
1631	Glarus Süd	9480	de	4143	437.025316
3746	Zernez	1506	rt	962	638.778220
3543	Surses	2377	rt	1826	768.195204
6037	Val de Bagnes	10329	NA	2441	236.324910
3851	Davos	10832	de	1746	161.189069
3792	Bregaglia	1556	it	510	327.763496
6252	Anniviers	2742	fr	517	188.548505
6300	Zermatt	5820	de	54	9.278351
784	Innertkirchen	1072	de	765	713.619403

Family names table

We now turn to the family names. First we load the data from `family-names-commune.csv`, retaining only the top names in 2021.

```
family_names <-
  read_delim(
    "input_data/family-names-commune.csv",
    delim = ";",
    escape_double = FALSE,
    trim_ws = TRUE
  )
family_names <- family_names %>%
  filter(TIME_PERIOD == 2021) %>%
  filter(RANG_GDE <= 100) %>%
  select(LASTNAME, GDENR, RANG_GDE, VALUE) %>%
  rename(family_name = LASTNAME, commune_id = GDENR, name_rank = RANG_GDE, name_count = VALUE)

family_names %>% sample_n(10) %>% kable()
```

family_name	commune_id	name_rank	name_count
Truan	5764	3	23
Schneider	5719	97	3
Peguiron	5489	18	4
Studer	566	24	6
Bähni	2044	15	9
Bänninger	2901	13	7
Kubli	214	29	5
Guggiari	5205	6	6
Mahendrarajah	4198	51	4
Giudici	5050	58	7

Next, we create a table `family_names` in the database with an automatically generated `id` as primary key, and `commune_id` as foreign key linking to the `communes` table.

```
dbExecute(conn, "
  create table if not exists family_names (
    id integer primary key,
    name text,
    commune_id int,
    name_count int,
    rank int,
    foreign key(commune_id) references communes(commune_id)
  )
")
```

```
## [1] 0
```

Then we load the data into the created table.

```
pb = txtProgressBar(min = 1, max = nrow(family_names), initial = 1)

for (i in 1:nrow(family_names)) {
  row = family_names[i,]
  dbExecute(conn, "
            insert into family_names (name, commune_id, name_count, rank)
            values (?, ?, ?, ?)",
          params = c(row$family_name, row$commune_id, row$name_count, row$name_rank))

  if (i %% 100 == 0) {
    setTxtProgressBar(pb,i)}
}
```

```
## ===============================================================================
```

```
close(pb)
```

Finally, we check that it worked by querying.

```
dbGetQuery(conn, "select * from family_names limit 10") %>% kable()
```

id	name	commune_id	name_count	rank
1	Frei	1	14	1
2	Meier	1	14	1
3	Weiss	1	14	1
4	Keller	1	13	4
5	Steiner	1	12	5
6	Angst	1	11	6
7	Bachmann	1	11	6
8	Müller	1	11	6
9	Spinner	1	11	6
10	Graf	1	10	10

Calf names table

The last table contains the calf names. We load the data from `cattle-NamesFemaleCalves.csv` and `cattle-NamesMaleCalves.csv`.

```
cattle_NamesFemaleCalves <- read_delim("input_data/cattle-NamesFemaleCalves.csv",
    delim = ";", escape_double = FALSE, trim_ws = TRUE) %>%
  mutate(sex="F")

cattle_NamesMaleCalves <- read_delim("input_data/cattle-NamesMaleCalves.csv",
    delim = ";", escape_double = FALSE, trim_ws = TRUE) %>%
  mutate(sex="M")

cattle_names <- bind_rows(cattle_NamesFemaleCalves, cattle_NamesMaleCalves)
rm(cattle_NamesFemaleCalves, cattle_NamesMaleCalves)

cattle_names <- cattle_names %>%
  filter(year==2022) %>%
  filter(OwnerLanguage != "__all__") %>%
  select(Name, count, Rank, sex, OwnerLanguage) %>%
  rename(name = Name, rank = Rank, language_region = OwnerLanguage)

cattle_names %>% arrange(rank, sex, language_region) %>% head(10) %>% kable()
```

name	count	rank	sex	language_region
Bella	674	1	F	de
Tulipe	88	1	F	fr
Luna	12	1	F	it
Max	633	1	M	de
Pvv	133	1	M	fr
Rambo	7	1	M	it
Sina	518	2	F	de
Bella	77	2	F	fr
Gina	10	2	F	it
Leo	545	2	M	de

We create the database table `cattle_names` using and automatically generated `id` as primary key, and `language_region` as a foreign key linking to the `communes` table.

```
dbExecute(conn, "
  create table if not exists cattle_names (
    id integer primary key,
    name text,
    count int,
    rank int,
    sex text,
    language_region text,
    foreign key(language_region) references communes(language_region)
  )
")
```

```
## [1] 0
```

We then load the data into the database and query to check if it worked.

```
for (i in 1:nrow(cattle_names)) {
  row = cattle_names[i,]
  dbExecute(conn, "
            insert into cattle_names (name, count, rank, sex, language_region)
            values (?, ?, ?, ?, ?)",
          params = c(row$name, row$count, row$rank, row$sex, row$language_region))
  }
```

```
dbGetQuery(conn, "select * from cattle_names limit 10") %>% kable()
```

id	name	count	rank	sex	language_region
1	Bella	674	1	F	de
2	Sina	518	2	F	de
3	Anna	464	3	F	de
4	Fiona	455	4	F	de
5	Tina	449	5	F	de
6	Olivia	448	6	F	de
7	Nora	446	7	F	de
8	Bianca	430	8	F	de
9	Nina	426	9	F	de
10	Luna	398	10	F	de

Querying the database

We now query the database to see which family names are most popular in communes with the highest cattle density per population.

We can get the communes with the highest cattle density per person as follows.

```
dbGetQuery(conn, "
           select *
           from communes
           order by cattle_per_1k desc
           limit 10
           ") %>% kable()
```

commune_id	name	population	language_region	cattle_population	cattle_per_1k
708	Schelten	39	de	278	7128.205
709	Seehof	56	de	389	6946.429
715	Rebévelier	41	de	279	6804.878
6433	Brot-Plamboz	290	fr	1571	5417.241
6759	Soubey	131	fr	682	5206.107
6432	La Brévine	624	fr	2985	4783.654
437	Mont-Tramelan	111	de	525	4729.730
2549	Kammersrohr	32	de	147	4593.750
2067	Le Châtelard	349	fr	1478	4234.957
6758	Saint-Brais	227	fr	940	4140.969

We see that the Scheltenites better treat their cattle with care, since in case of cow mutiny they are outnumbered 7:1.

Lets now get all communes that have more than 3 cattle per person. These are the cowboy communes.

```
dbGetQuery(conn, "
           select *
           from communes
           where cattle_per_1k > 3000
           order by cattle_per_1k desc
           ") %>% head(n = 10) %>% kable()
```

commune_id	name	population	language_region	cattle_population	cattle_per_1k
708	Schelten	39	de	278	7128.205
709	Seehof	56	de	389	6946.429
715	Rebévelier	41	de	279	6804.878
6433	Brot-Plamboz	290	fr	1571	5417.241
6759	Soubey	131	fr	682	5206.107
6432	La Brévine	624	fr	2985	4783.654
437	Mont-Tramelan	111	de	525	4729.730
2549	Kammersrohr	32	de	147	4593.750
2067	Le Châtelard	349	fr	1478	4234.957
6758	Saint-Brais	227	fr	940	4140.969

We can now plug the resulting `commune_ids` of the above query into `family_names` to get a list of the most frequent family names in these high-cow regions. These are the cowboy names.

```
dbGetQuery(conn, "
           select *
           from family_names
           where commune_id in
             (select commune_id
             from communes
             where cattle_per_1k > 3000)

           ") %>% sample_n(10) %>% kable()
```

id	name	commune_id	name_count	rank
208110	Petitpierre	6432	7	16
208135	Blättler	6432	4	40
209753	Donzé	6511	3	44
208166	Richard	6433	6	7
208130	Othenin-Girard	6432	5	30
91134	Häfliger	2612	4	21
208104	Philipona	6432	9	9
68953	Ayer	2067	3	22
209717	Rosselet-Christ	6511	7	11
209747	Vonlaufen	6511	4	30

We can now aggregate this list by `name_count` to get the most frequent cowboy names. We print the top 30 cowboy names

```
dbGetQuery(conn, "
           select name, sum(name_count) name_count
           from family_names
           where commune_id in
             (select commune_id
             from communes
             where cattle_per_1k > 3000)
           group by name
           order by name_count desc
           limit 30
           ") %>% kable()
```

name	name_count
Amstutz	82
Gerber	67
Frésard	61
Froidevaux	55
Girardin	52
Boillat	51
Marchand	44
Rey	42
Jeannerat	40
Oberson	38
Maire	38
Theurillat	36
Queloz	36
Odiet	36

name	name_count
Roch	35
Piquerez	33
Uldry	32
Rosselet-Christ	31
Pellaton	31
Choulat	31
Cantieni	31
Beuret	30
Willemin	29
Maitre	29
Donzé	29
Grolimund	28
Michael	27
Aubry	27
Gyger	26
Cattin	26

We observe that many of the names are French. Queloz is incidentally the name of physics nobel laureate Didier Queloz whose research concerns the detection of earth-like exoplanets. Perhaps he is in part motivated to find new pastures for raising cattle.

If we wanted to find the top cattle name, assuming that they take the family names of their masters, we could need the following information: 1. The commune with the most cattle per capita per language region:

```
dbGetQuery(conn,"
            select
                commune_id,
                max(cattle_per_1k),
                language_region
            from
                communes
            group by
                language_region
")
```

```
##   commune_id max(cattle_per_1k) language_region
## 1       3715           3159.780            <NA>
## 2        708           7128.205              de
## 3       6433           5417.241              fr
## 4       5304           1269.231              it
## 5       3618           1717.936              rt
```

2. The most common name in that commune

```
dbGetQuery(conn,"
select
    language_region,
    family_name
from
    (
        (
            select
                commune_id,
                max(cattle_per_1k),
                language_region
            from
                communes
            group by
                language_region
        ) as top_cattle_communes
        left join (
            select
                name as family_name,
                commune_id
            from
                family_names
            where
                rank = 1
        ) as top_names
        on top_cattle_communes.commune_id == top_names.commune_id
    ) where language_region is not null
") %>% kable()
```

language_region	family_name
de	Roos
de	Spänhauer
fr	Maire
it	Tomamichel
rt	Derungs

3. The top cattle name per sex and language region.

```
dbGetQuery(conn,"
select
    name as first_name,
    sex,
    language_region
from
    cattle_names
where
    rank == 1
") %>% kable()
```

first_name	sex	language_region
Bella	F	de
Tulipe	F	fr
Luna	F	it
Max	M	de
Pvv	M	fr
Rambo	M	it

Sadly, the Bündnerromanisch cattle names are not included in the original dataset, so the names of those *Vacha* will be excluded.

The results can be retrieved as a single SQL query, as shown:

```
dbGetQuery(conn,"
select
    first_name,
    family_name,
    sex,
    top_cowboy_first_names.language_region
from
    (
        select
            language_region,
            family_name
        from
            (
                (
                    select
                        commune_id,
                        max(cattle_per_1k),
                        language_region
                    from
                        communes
                    group by
                        language_region
                ) as top_cattle_communes

                left join (
                    select
                        name as family_name,
                        commune_id
                    from
                        family_names
                    where
                        rank = 1
                ) as top_names
                on top_cattle_communes.commune_id == top_names.commune_id
            )
    ) as top_cowboy_family_names
    join (
        select
            name as first_name,
            sex,
            language_region
        from
            cattle_names
        where
            rank == 1
    ) as top_cowboy_first_names
on top_cowboy_family_names.language_region = top_cowboy_first_names.language_region;") %>%
  kable()
```

first_name	family_name	sex	language_region
Bella	Roos	F	de
Max	Roos	M	de
Bella	Spänhauer	F	de

first_name	family_name	sex	language_region
Max	Spänhauer	M	de
Pvv	Maire	M	fr
Tulipe	Maire	F	fr
Luna	Tomamichel	F	it
Rambo	Tomamichel	M	it

We can see here that there is a tie for the top cattle family name in German, between Roos and Spänhauer. From this, we can see that these names are perhaps the most iconic cattle names in Switzerland.