



Personal data handling in a modern data platform

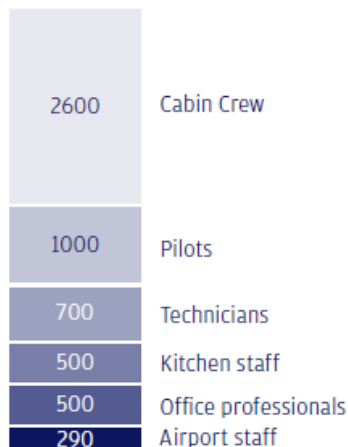
Jarkko Venna
Senior Data Engineer
Finnair



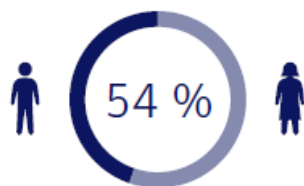
FINNAIR IN NUMBERS

PEOPLE

A breakdown of our largest employee groups



We also have data scientists, chefs and doctors



of our employees are women



DID YOU KNOW...

800 000

litres of blueberry juice is served on board each year

NETWORK

We fly the shortest route between Asia and Europe



8
destinations
in North America

100+
destinations
in Europe

19
destinations
in Asia

KEY FACTS



12 million
customers in 2017



40 000
customers per day



25 %
of them are transfer
passengers



100 000+
flights per year



300
flights per day



99 %
connection reliability
at Helsinki Airport

FLEET

We have one of the most modern fleets in Europe



We also operate

36
narrowbody aircraft

In addition, our partner Norra operates
24 embraers and ATRs for us

CARGO



We also carry
320 000 kilos
of cargo every day

Including precious cargo like
king crab, salmon and medicine

FINNAIR

New development at Finnair

- We are developing a new modern cloud based data platform
- Some problems with the current systems
 - Systems are siloed and they don't scale well
 - Personal data handling is inflexible and hard to maintain





A modern data platform: in a crossroads of requirements





Conflicting requirements

Pressure from usage needs

- Volume of data is increasing
- Data needs to be available for analysis faster
- Number of different types of data is increasing
- Data from multiple systems need to be combined
- Number of people using data is increasing

Pressure form privacy and security

- Access to data needs to be controlled more tightly (especially personal data)
- Customers have a right to ask for their data to be removed
- Personal data retention has to be defined
- Production personal data should not be used in development

How to solve the conflict

So that:

- We can keep storing large amounts of data
- We can control access to personal data
- The Data is easily available and usable
- New data can be added easily
- We can remove the personal data when requested
- We can remove personal data after a persons data retention period has expired

While:

- Keeping the removal process manageable
- Not causing too many problems for data usability





The strategy



Process personal data on ingestion

- Replace personal data with tokens in the ingestion flow.
- Create a mapping table to hold the tokens and real values

+

-

- | | |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none">• Personal data separated to a different location• Data except personal data available to all users• Data to be removed localized to one place• Can prevent already removed items from reappearing from source | <ul style="list-style-type: none">• Ingestion flow more complex• Data content needs to be known before ingesting• Getting the real values requires a join |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|





How to process data during ingestion?



The basic algorithm

Read the data

For each personal data field:

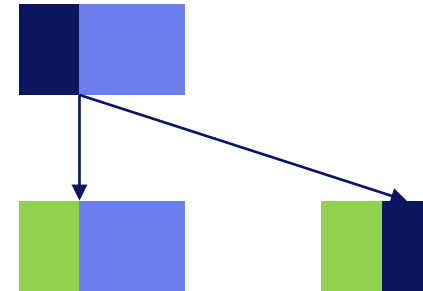
if the person id is already in the mapping table with a removed status:

- set the field value to “removed”

else:

- calculate the key to the mapping table
- replace the field value with the key
- write the real value to the mapping table with the key and person id

Write the data





Some requirements for mapping table

- The mapping table needs to hold enough additional information to make removal processing possible
- The mapping table should only store the person id - value -pair once (possibly for each dataset)

Example:

| Key | Dataset | Id_type | Id | Value_type | Value | Created_ts | Removed_ts | Removal_reason |
|----------|-----------|-------------|------|------------|---------|------------|------------|----------------|
| dsfa3243 | customers | Customer_id | 2382 | First_name | Jarkko | 2019-01-01 | | |
| da23ds | customers | Customer_id | 1234 | First_name | removed | 2018-01-01 | 2018-12-01 | request |



Basic removal algorithm

Get a list of all possible id, id-type -pairs related to the person whose data is to be removed

For each id id-type pair:

- Replace values in all matching rows in the mapping table with "removed"
- Add the removal timestamp and reason to the mapping table



Practical implementation considerations

- The process should be implemented as a parameterized component
 - Implement once – use multiple times
- It is a good idea to standardize the value before writing it to the mapping table
 - Lower case emails, upper case names etc.
- Removed values should match the type of the value. For example:
 - "removed" for string
 - "-1" for numbers
 - "1800-01-01 00:00:00" for timestamps



Enhancements





Use multiple mapping tables

- Create different versions of the same mapping
 - Standard – the normal mapping table
 - Development – replace values with scrambled values
 - Legal – For providing values after they are removed from standard mapping
- Separate mapping tables for different areas in the data
 - Customer – for the customer data
 - HR - for employee data



More flexibility to
handle different
use cases and
security
requirements




Provide analytical fields

- Create analytical fields from fields that have been mapped out from the data
 - The values in the analytical fields are fuzzy versions of the real values
 - Each analytical value should be fuzzy enough to cover large enough set of persons

Examples:

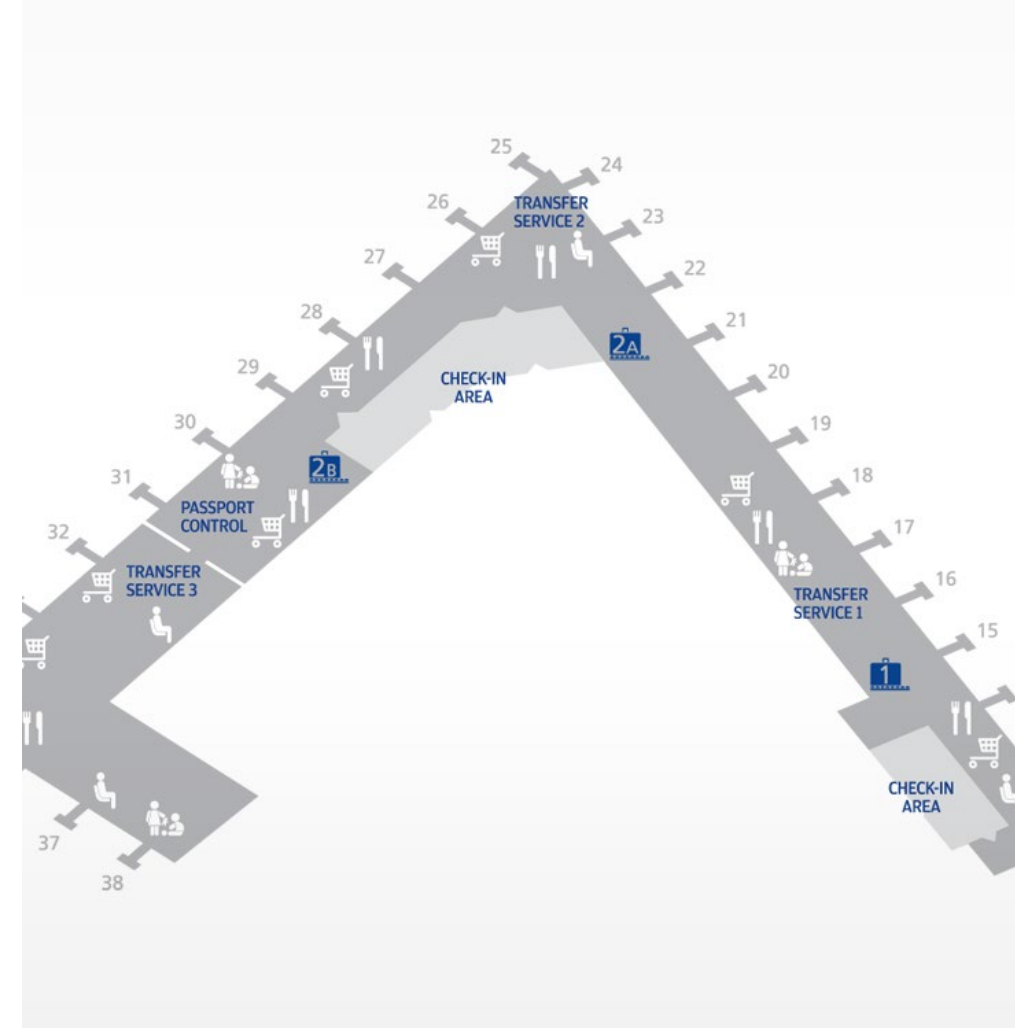
- Combine low population zip codes to large areas
- Categorize birth year to age group
- Provide only the first part of IP address



Provide usable data without need to use the real values



Considerations on Data Modelling





Data modelling principles

- If a personal data item is also a key to the data, replace the key with a masked version, and store the original value as an attribute
- Don't store real personal data values in tables that are incrementally updated
- You can store real personal data values in tables as long as they are fully refreshed often enough and you do not need to restrict access.



Key points

By handling personal data intelligently we can:

- Provide wide access to data without compromising personal data security
- Provide granular access to personal data
- Simplify the personal data removal process





Thank You

Jarkko Venna
jarkko.venna@finnair.com



FINNAIR