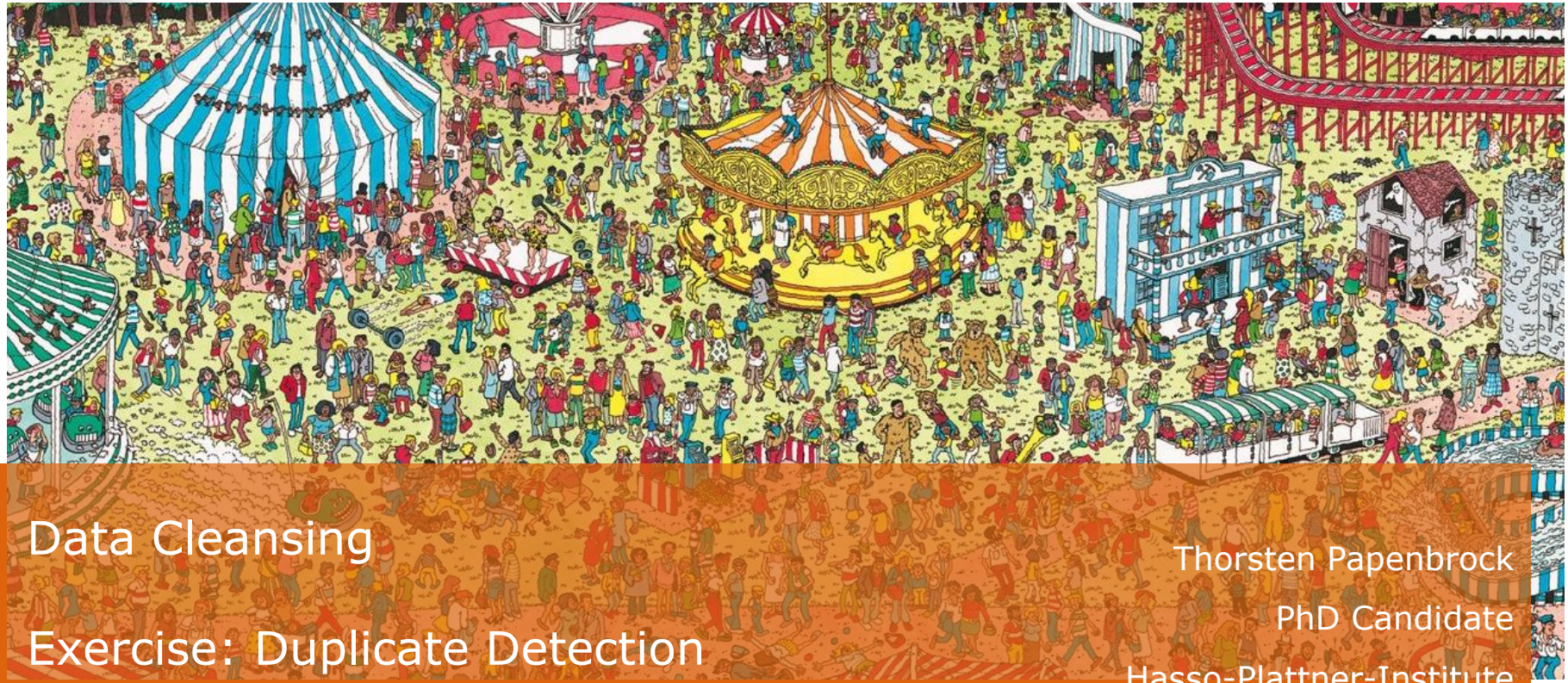
Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam



Data Cleansing

Exercise: Duplicate Detection

Thorsten Papenbrock
PhD Candidate
Hasso-Plattner-Institute

Advanced Profiling
Three important metadata

Hasso Plattner Institut

Unique Column Combinations

Key candidates

Name	Type	Equatorial diameter	Mass
Mercury	Terrestrial	0.382	0.06
Venus	Terrestrial	0.949	0.82
Earth	Terrestrial	1.000	1.00
Mars	Terrestrial	0.532	0.11
Jupiter	Giant	11.209	317.8
Saturn	Giant	9.449	95.2
Uranus	Giant	4.007	14.6
...	...	...	...

|Name|

Inclusion Dependencies

Foreign key candidates

Sign	Domicile
Aries	Mars
Taurus	Venus
Gemini	Mercury
Cancer	Moon
Leo	Sun
Virgo	Mercury
Libra	Venus
Scorpio	Pluto
Sagittarius	Jupiter
Capricorn	Saturn
Aquarius	Uranus
...	...

Name	Type
Mercury	Terrestrial
Venus	Terrestrial
Earth	Terrestrial
Mars	Terrestrial
Jupiter	Giant
Saturn	Giant
Uranus	Giant
...	...

Domicile ⊆ Name

Functional Dependencies

Normalization criterion

Name	Atmosphere	Rings
Mercury	minimal	no
Venus	CO_2, N_2	no
Earth	N_2, O_2, Ar	no
Mars	CO_2, N_2, Ar	no
Jupiter	H_2, He	yes
Saturn	H_2, He	yes
Uranus	H_2, He	yes
...	...	...

Atmosphere → Rings

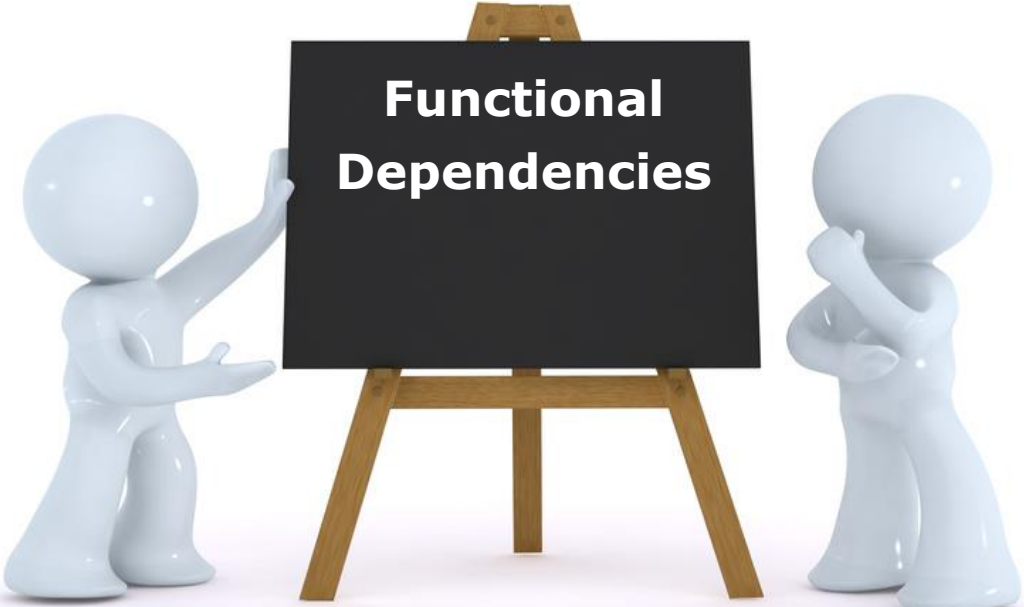
Data Profiling with Metanome

ThorstenPapenbrock,
PhD Candidate,
17th November, 2014
Chart **2**

Exercise 3
Discovery of functional dependencies

- All teams have passed the exercise:
  - 34 submissions
  - No duplicate algorithm names!
  - Still a few incorrect results (even after correction round)
  - No import errors in Metanome (apart from execution errors)

Exercise 3
Short presentations – Part 1



Exercise 3
Our evaluation

- DELL Optiplex 9010
  - CPU: Intel i5 3.2 GHz
  - RAM: 8 GB (2 GB for Metanome JVM)
  - OS: Debian 64-bit
  - JVM: Java 1.8

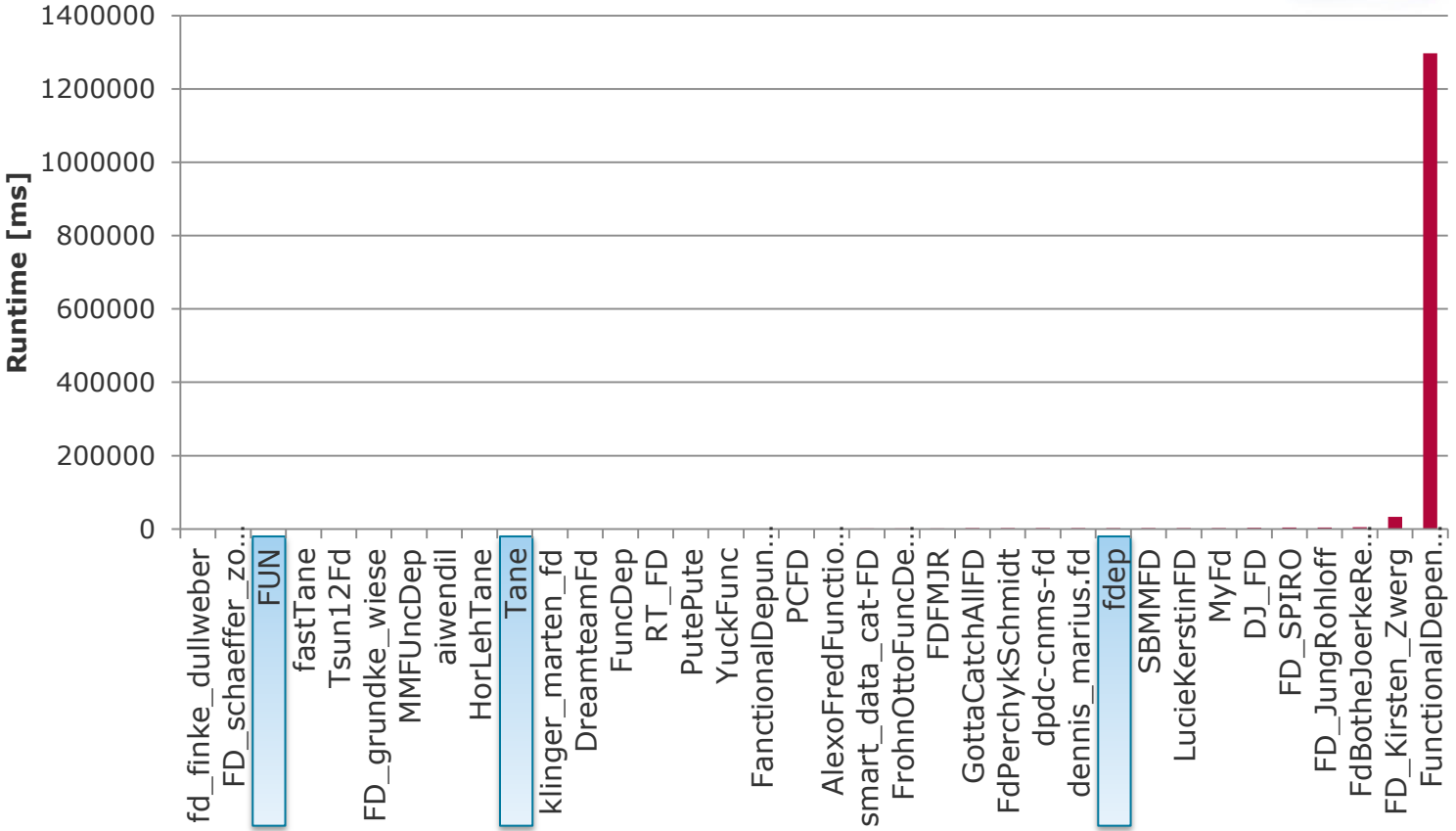
Exercise 3
Correctness for abalone.csv

aiwendil
AlexoFredFunctionals
dennis_marius.fd
DJ_FD
dpdc-cnms-fd
DreamteamFd
FanctionalDepundancy
fastTane
fd_finke_dullweber
FD_grundke_wiese
FD_JungRohloff
FD_Kirsten_Zwerg
FD_schaeffer_zoellner
FD_SPIRO
FdBotheJoerkeReissaus
FDFMJR
FdPerchykSchmidt
FrohnOttoFuncDep-LAME-TANE
FuncDep
FunctionalDependencyDetector
FunctionalDerpendency
GottaCatchAllFD
HorLehTane
klinger_marten_fd
LucieKerstinFD
MMFUncDep
MyFd
PCFD
PutePute
RT_FD
SBMMFD
smart_data_cat-FD
Tsun12Fd
YuckFunc

Exercise 3
Correctness for abalone.csv

- aiwendil
- AlexoFredFunctionals
- dennis_marius.fd
- DJ_FD
- dpdc-cnms-fd
- DreamteamFd
- FanctionalDepundancy
- fastTane
- fd_finke_dullweber
- FD_grundke_wiese
- FD_JungRohloff
- FD_Kirsten_Zwerg
- FD_schaeffer_zoellner
- FD_SPIRO
- FdBotheJoerkeReissaus
- FDFMJR
- FdPerchykSchmidt
- FrohnOttoFuncDep-LAME-TANE
- FuncDep
- FunctionalDependencyDetector
- FunctionalDerpendency **incorrect**
- GottaCatchAllFD
- HorLehTane
- klinger_marten_fd
- LucieKerstinFD
- MMFUncDep
- MyFd
- PCFD
- PutePute
- RT_FD
- SBMMFD
- smart_data_cat-FD
- Tsun12Fd
- YuckFunc

Exercise 3
Runtime for abalone.csv

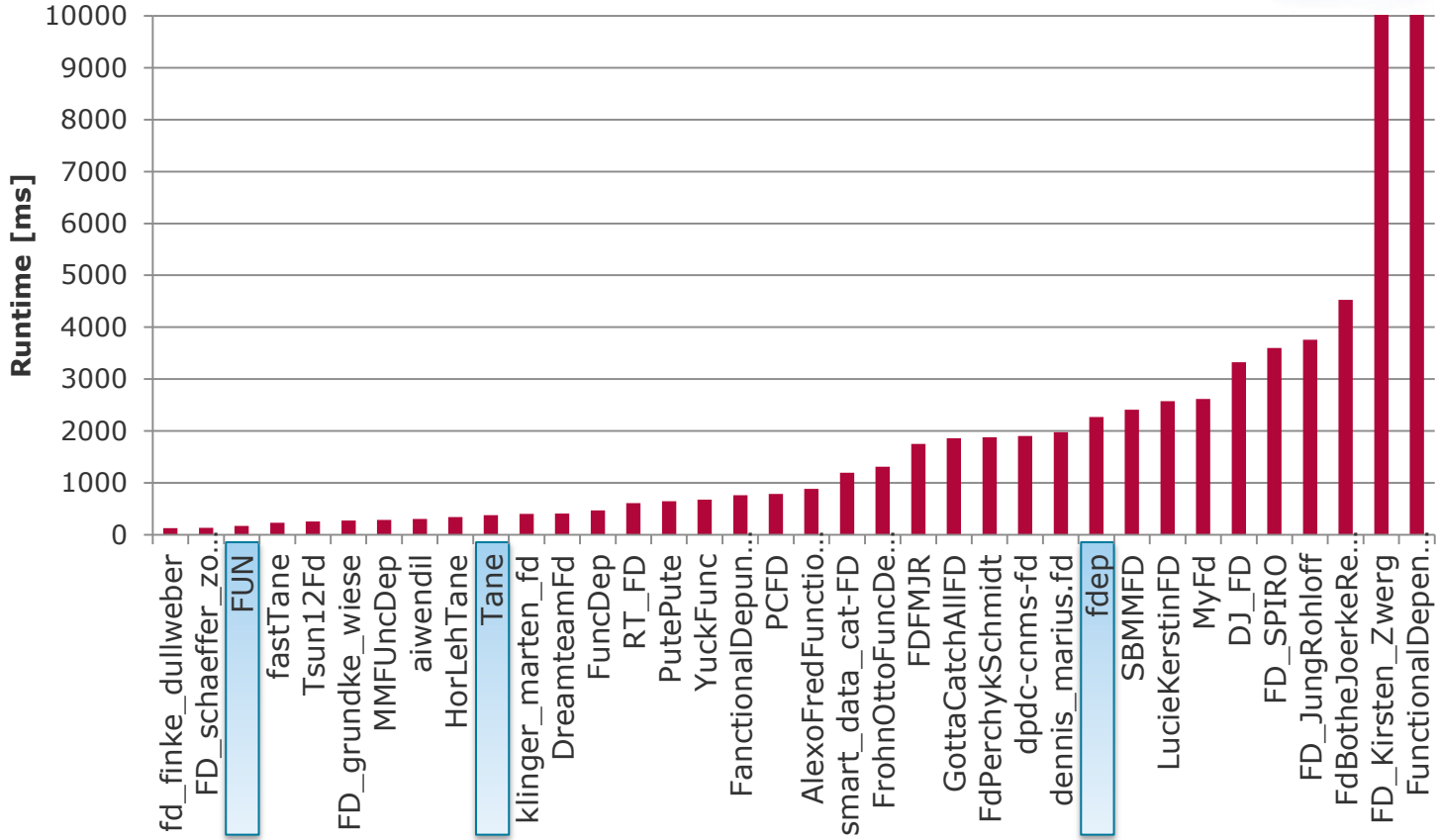


- Columns: **9**
- Rows: **4,177**
- FDs: **137**

Data Profiling with Metanome

ThorstenPapenbrock, PhD Candidate,
17th November, 2014
Chart **8**

Exercise 3
Runtime for abalone.csv (<10s)

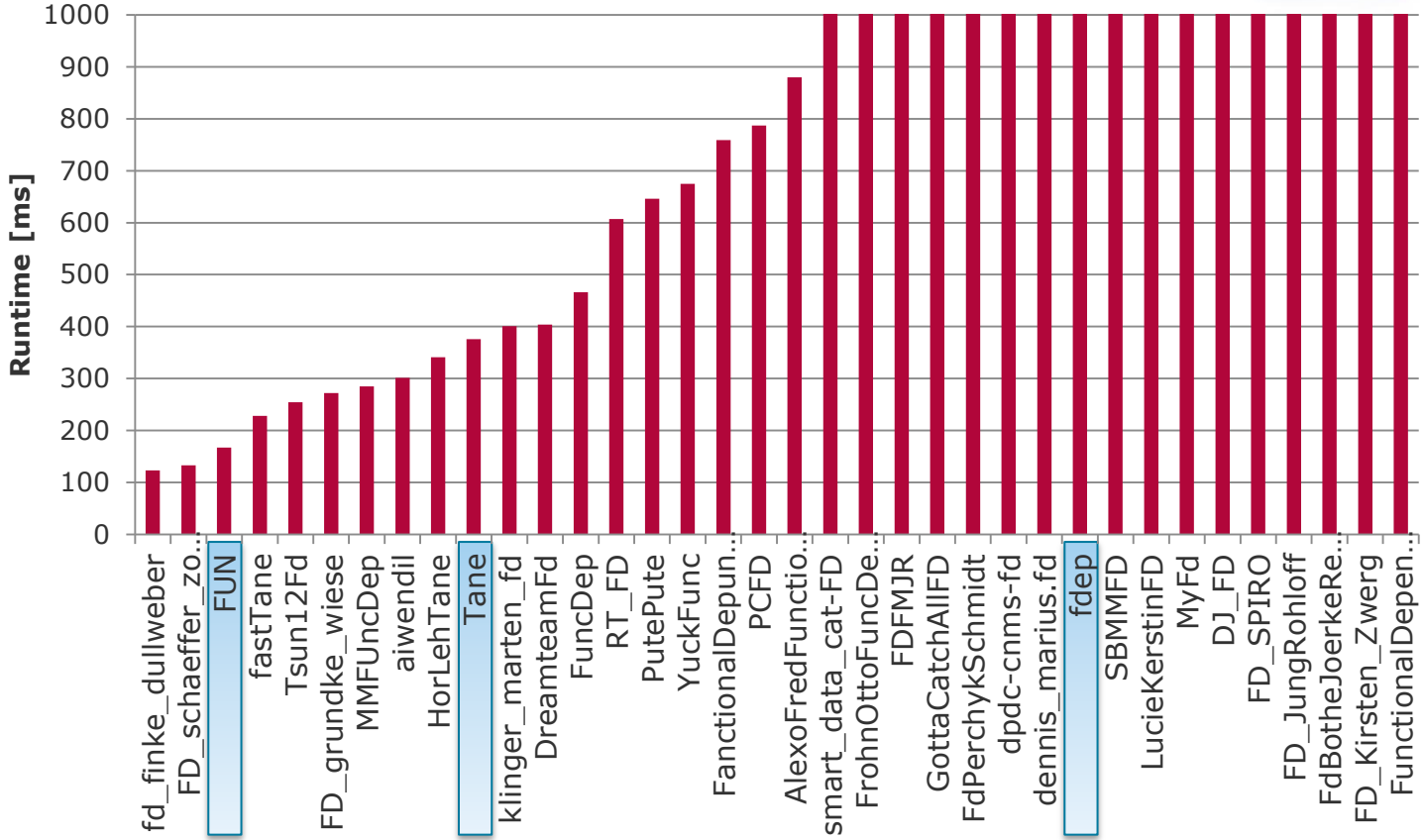


- Columns: **9**
- Rows: **4,177**
- FDs: **137**

Data Profiling with Metanome

ThorstenPapenbrock, PhD Candidate,
17th November, 2014
Chart **9**

Exercise 3
Runtime for abalone.csv (<1s)



- Columns: **9**
- Rows: **4,177**
- FDs: **137**

Data Profiling with Metanome

ThorstenPapenbrock, PhD Candidate,
17th November, 2014
Chart **10**

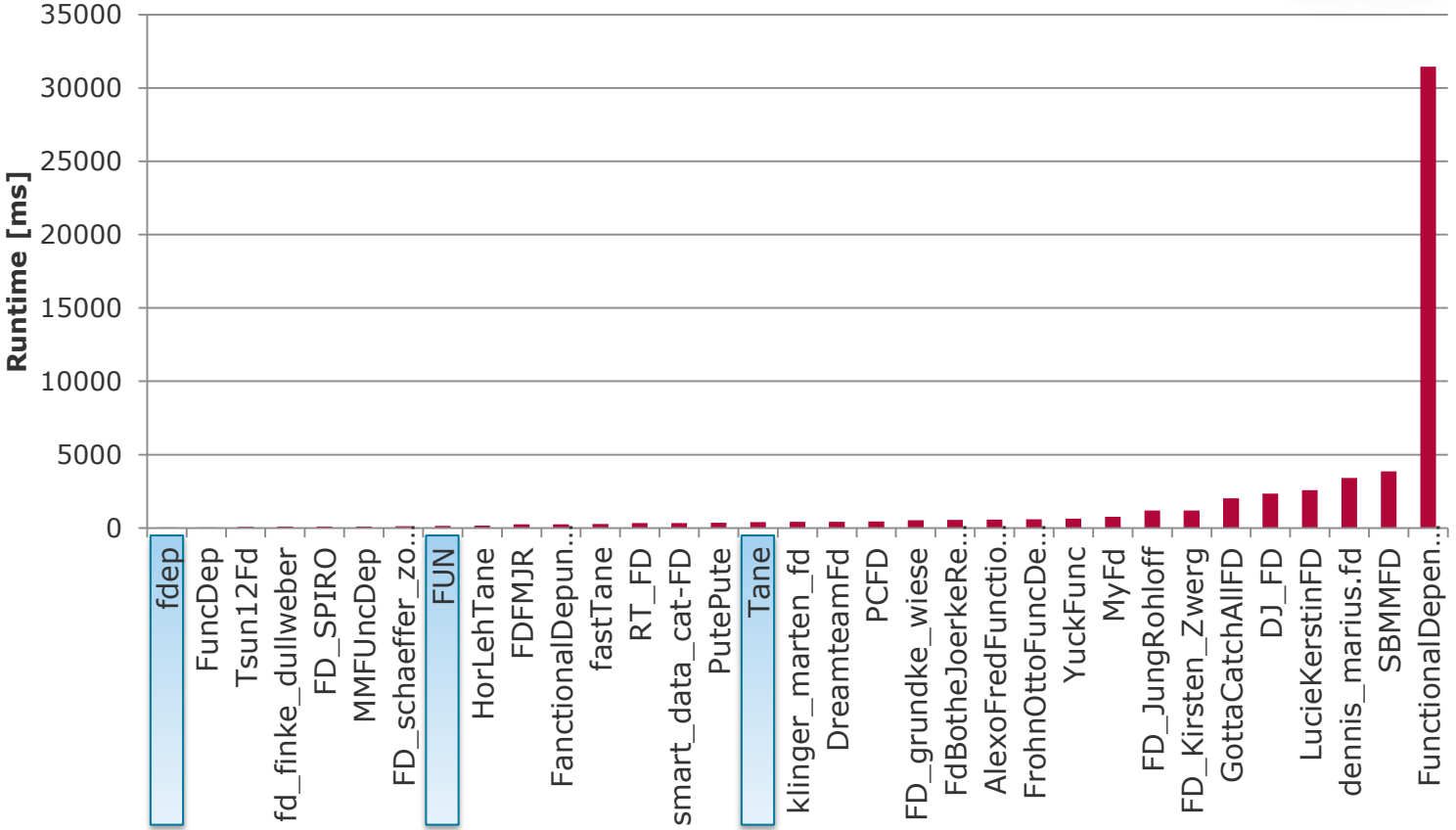
Exercise 3
Correctness for bridges.csv

aiwendil
AlexoFredFunctionals
dennis_marius.fd
DJ_FD
dpdc-cnms-fd
DreamteamFd
FanctionalDepundancy
fastTane
fd_finke_dullweber
FD_grundke_wiese
FD_JungRohloff
FD_Kirsten_Zwerg
FD_schaeffer_zoellner
FD_SPIRO
FdBotheJoerkeReissaus
FDFMJR
FdPerchykSchmidt
FrohnOttoFuncDep-LAME-TANE
FuncDep
FunctionalDependencyDetector
FunctionalDerpendency **incorrect**
GottaCatchAllFD
HorLehTane
klinger_marten_fd
LucieKerstinFD
MMFUncDep
MyFd
PCFD
PutePute
RT_FD
SBMMFD
smart_data_cat-FD
Tsun12Fd
YuckFunc

Exercise 3
Correctness for bridges.csv

aiwendil	**incorrect**	FrohnOttoFuncDep-LAME-TANE	
AlexoFredFunctionals		FuncDep	
dennis_marius.fd		FunctionalDependencyDetector	
DJ_FD		FunctionalDerpendency	**incorrect**
dpdc-cnms-fd	**SerializationError**	GottaCatchAllFD	
DreamteamFd		HorLehTane	
FanctionalDepundancy		klinger_marten_fd	
fastTane		LucieKerstinFD	
fd_finke_dullweber		MMFUncDep	
FD_grundke_wiese		MyFd	
FD_JungRohloff		PCFD	
FD_Kirsten_Zwerg		PutePute	
FD_schaeffer_zoellner		RT_FD	
FD_SPIRO		SBMMFD	
FdBotheJoerkeReissaus		smart_data_cat-FD	
FDFMJR		Tsun12Fd	
FdPerchykSchmidt	**SerializationError**	YuckFunc	

Exercise 3
Runtime for bridges.csv

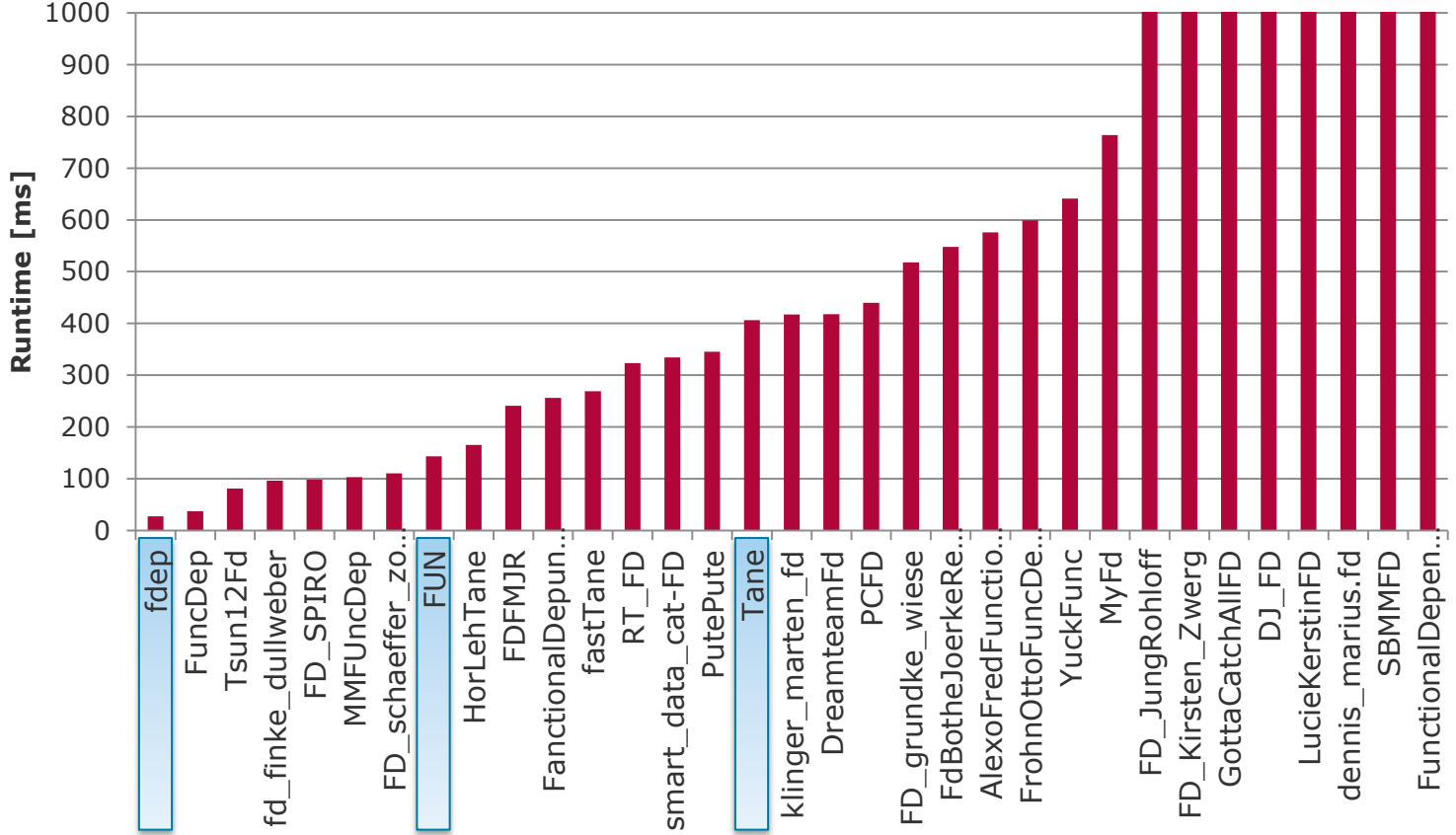


- Columns: **13**
- Rows: **108**
- FDs: **142**

Data Profiling with Metanome

ThorstenPapenbrock, PhD Candidate,
17th November, 2014
Chart **13**

Exercise 3
Runtime for bridges.csv (<1s)



- Columns: **13**
- Rows: **108**
- FDs: **142**

Data Profiling with Metanome

ThorstenPapenbrock, PhD Candidate, 17th November, 2014
Chart **14**

Exercise 3
Correctness for hepatitis.csv

✗ aiwendil	**incorrect**	
AlexoFredFunctionals		
dennis_marius.fd		
DJ_FD		
✗ dpdc-cnms-fd	**SerializationError**	
DreamteamFd		
FanctionalDepundancy		
fastTane		
fd_finke_dullweber		
FD_grundke_wiese		
FD_JungRohloff		
FD_Kirsten_Zwerg		
FD_schaeffer_zoellner		
FD_SPIRO		
FdBotheJoerkeReissaus		
FDFMJR		
✗ FdPerchykSchmidt	**SerializationError**	
FrohnOttoFuncDep-LAME-TANE		
FuncDep		
FunctionalDependencyDetector		
✗ FunctionalDerpendency	**incorrect**	
GottaCatchAllFD		
HorLehTane		
klinger_marten_fd		
LucieKerstinFD		
MMFUncDep		
MyFd		
PCFD		
PutePute		
RT_FD		
SBMMFD		
smart_data_cat-FD		
Tsun12Fd		
YuckFunc		

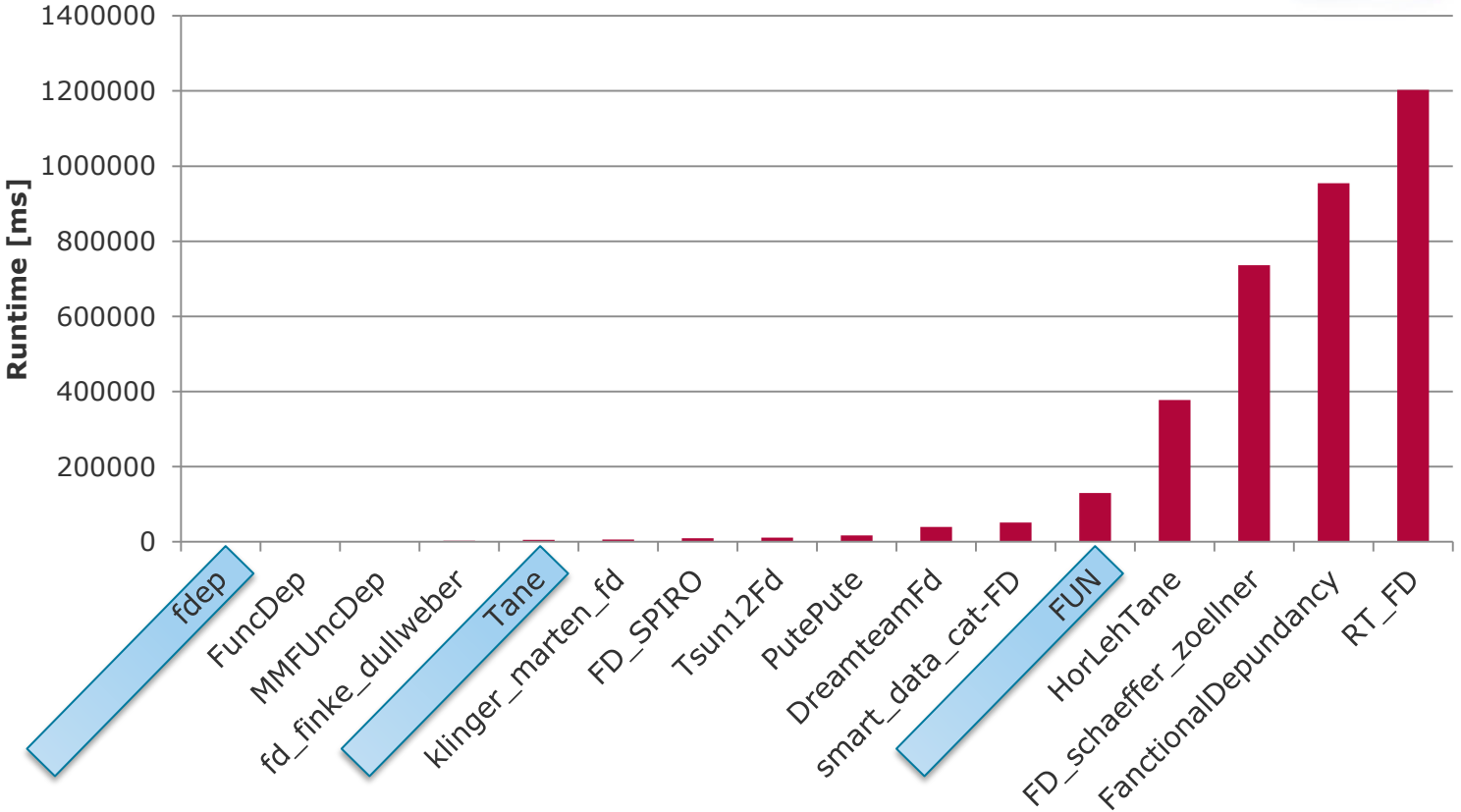
Exercise 3
Correctness for hepatitis.csv

✗ aiwendil	**incorrect**	FrohnOttoFuncDep-LAME-TANE	
AlexoFredFunctionals		FuncDep	
dennis_marius.fd		FunctionalDependencyDetector	
DJ_FD		✗ FunctionalDerpendency	**incorrect**
✗ dpdc-cnms-fd	**SerializationError**	GottaCatchAllFD	
DreamteamFd		HorLehTane	
FanctionalDepundancy		klinger_marten_fd	
✗ fastTane	**SerializationError**	LucieKerstinFD	
fd_finke_dullweber		MMFUncDep	
FD_grundke_wiese		MyFd	
FD_JungRohloff		PCFD	
FD_Kirsten_Zwerg		PutePute	
FD_schaeffer_zoellner		RT_FD	
FD_SPIRO		SBMMFD	
FdBotheJoerkeReissaus		smart_data_cat-FD	
FDFMJR		Tsun12Fd	
✗ FdPerchykSchmidt	**SerializationError**	YuckFunc	

Exercise 3
Correctness for hepatitis.csv

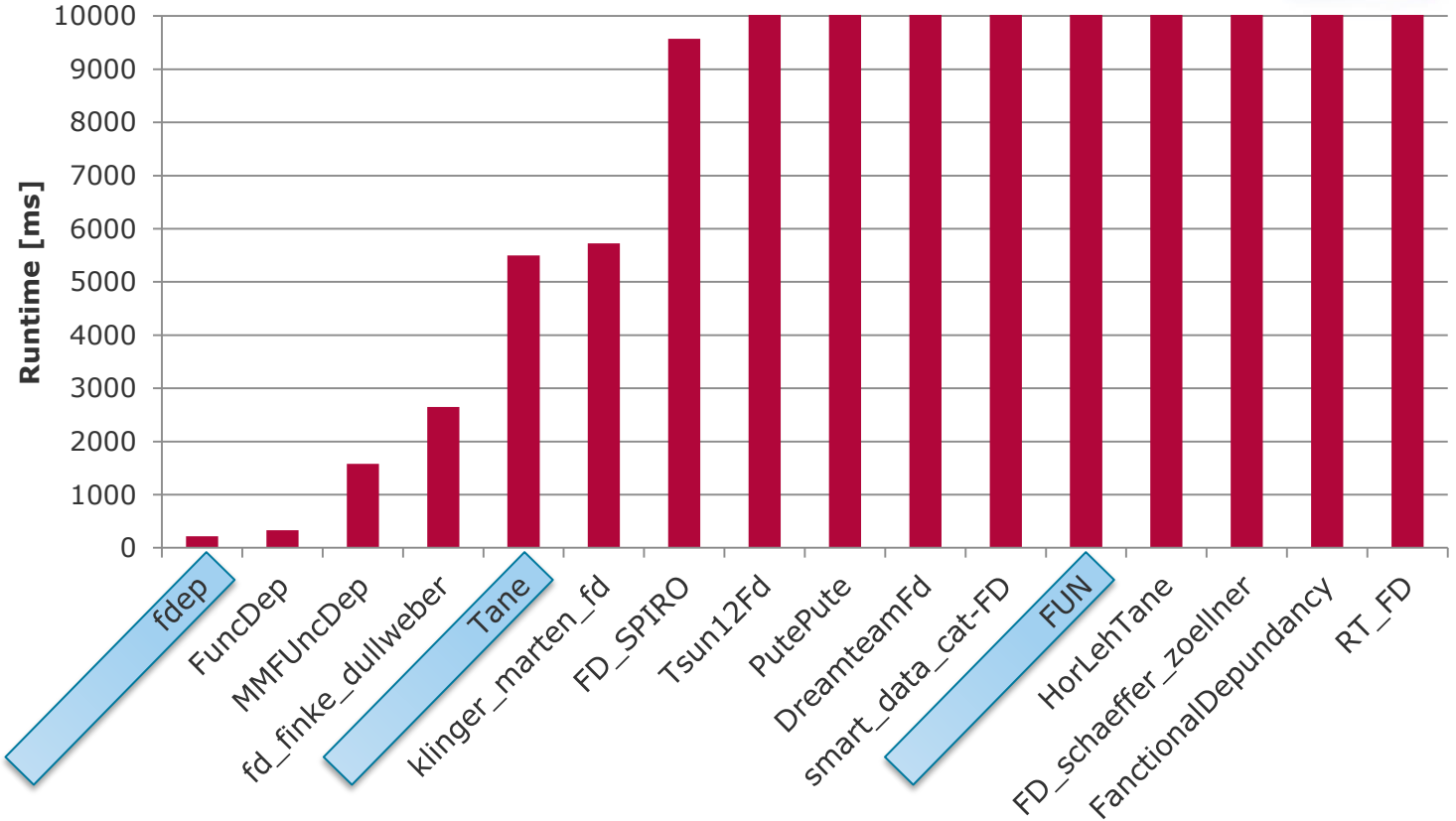
aiwendil	**incorrect**	FrohnOttoFuncDep-LAME-TANE	**> 1h ?**
AlexoFredFunctionals	**> 1h ?**	FuncDep	
dennis_marius.fd	**> 1h ?**	FunctionalDependencyDetector	**> 1h ?**
DJ_FD	**> 1h ?**	FunctionalDerpendency	**incorrect**
dpdc-cnms-fd	**SerializationError**	GottaCatchAllFD	**> 1h ?**
DreamteamFd		HorLehTane	
FanctionalDepundancy		klinger_marten_fd	
fastTane	**SerializationError**	LucieKerstinFD	**> 1h ?**
fd_finke_dullweber		MMFUncDep	
FD_grundke_wiese	**> 1h**	MyFd	**> 1h ?**
FD_JungRohloff	**> 1h ?**	PCFD	**> 1h**
FD_Kirsten_Zwerg	**> 1h ?**	PutePute	
FD_schaeffer_zoellner		RT_FD	
FD_SPIRO		SBMMFD	**> 1h ?**
FdBotheJoerkeReissaus	**> 1h ?**	smart_data_cat-FD	
FDFMJR	**> 1h**	Tsun12Fd	
FdPerchykSchmidt	**SerializationError**	YuckFunc	**> 1h**

Exercise 3
Runtime for hepatitis.csv



- Columns: **20**
- Rows: **155**
- FDs: **8,250**

Exercise 3
Runtime for hepatitis.csv (<10s)



- Columns: **20**
- Rows: **155**
- FDs: **8,250**

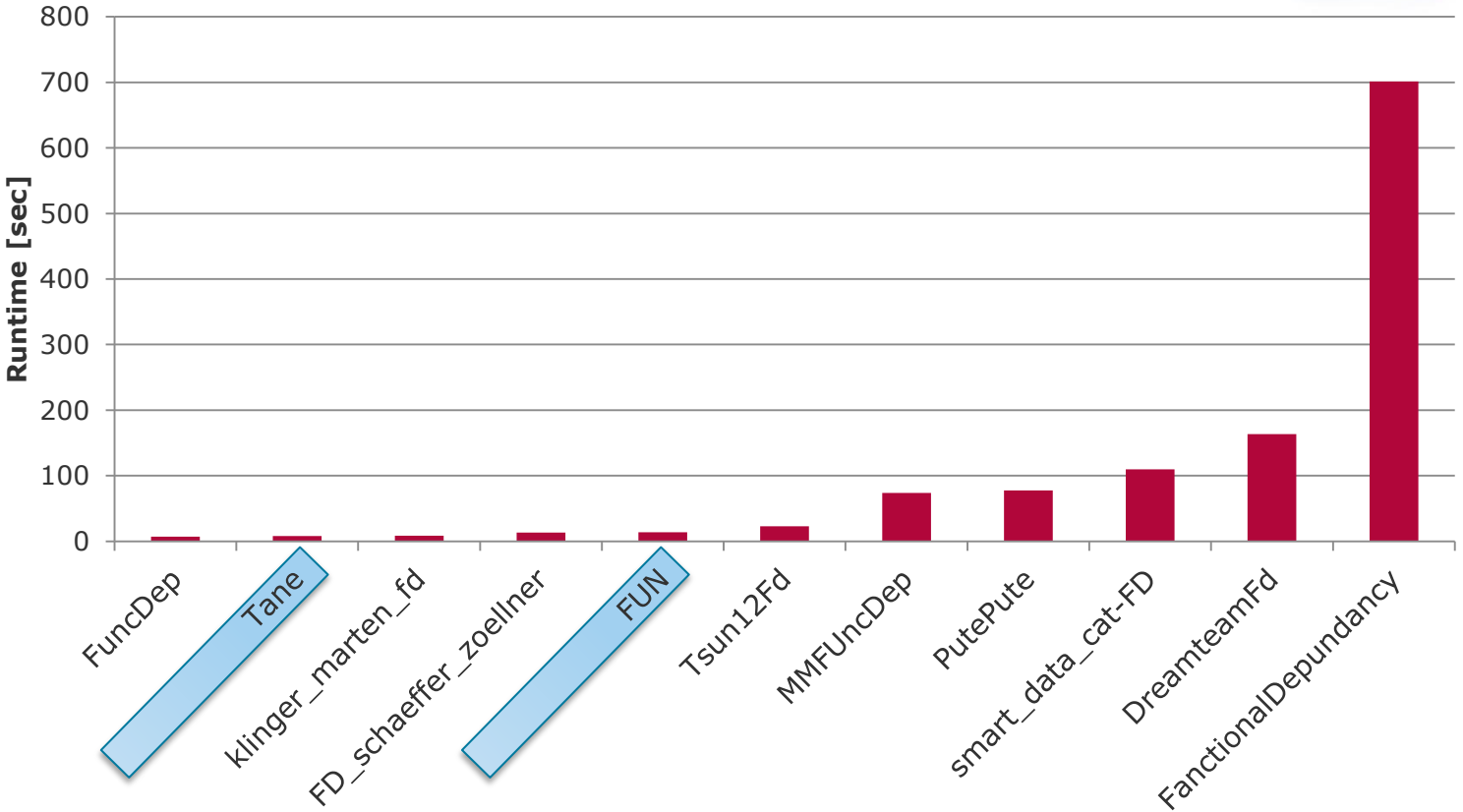
Exercise 3
Correctness for fd-reduced-15.csv

aiwendil	**incorrect**	FrohnOttoFuncDep-LAME-TANE	**> 1h ?**
AlexoFredFunctionals	**> 1h ?**	FuncDep	
dennis_marius.fd	**> 1h ?**	FunctionalDependencyDetector	**> 1h ?**
DJ_FD	**> 1h ?**	FunctionalDerpendency	**incorrect**
dpdc-cnms-fd	**SerializationError**	GottaCatchAllFD	**> 1h ?**
DreamteamFd		HorLehTane	
FanctionalDepundancy		klinger_marten_fd	
fastTane	**SerializationError**	LucieKerstinFD	**> 1h ?**
fd_finke_dullweber		MMFUncDep	
FD_grundke_wiese	**> 1h**	MyFd	**> 1h ?**
FD_JungRohloff	**> 1h ?**	PCFD	**> 1h**
FD_Kirsten_Zwerg	**> 1h ?**	PutePute	
FD_schaeffer_zoellner		RT_FD	
FD_SPIRO		SBMMFD	**> 1h ?**
FdBotheJoerkeReissaus	**> 1h ?**	smart_data_cat-FD	
FDFMJR	**> 1h**	Tsun12Fd	
FdPerchykSchmidt	**SerializationError**	YuckFunc	**> 1h**

Exercise 3
Correctness for fd-reduced-15.csv

aiwendil	**incorrect**	FrohnOttoFuncDep-LAME-TANE	**> 1h ?**
AlexoFredFunctionals	**> 1h ?**	FuncDep	
dennis_marius.fd	**> 1h ?**	FunctionalDependencyDetector	**> 1h ?**
DJ_FD	**> 1h ?**	FunctionalDerpendency	**incorrect**
dpdc-cnms-fd	**SerializationError**	GottaCatchAllFD	**> 1h ?**
DreamteamFd		HorLehTane	**> 30min**
FanctionalDepundancy		klinger_marten_fd	
fastTane	**SerializationError**	LucieKerstinFD	**> 1h ?**
fd_finke_dullweber	**incorrect**	MMFUncDep	
FD_grundke_wiese	**> 1h**	MyFd	**> 1h ?**
FD_JungRohloff	**> 1h ?**	PCFD	**> 1h**
FD_Kirsten_Zwerg	**> 1h ?**	PutePute	
FD_schaeffer_zoellner		RT_FD	**> 30min**
FD_SPIRO	**> 30min**	SBMMFD	**> 1h ?**
FdBotheJoerkeReissaus	**> 1h ?**	smart_data_cat-FD	
FDFMJR	**> 1h**	Tsun12Fd	
FdPerchykSchmidt	**SerializationError**	YuckFunc	**> 1h**

Exercise 3
Runtime for fd-reduced-15.csv

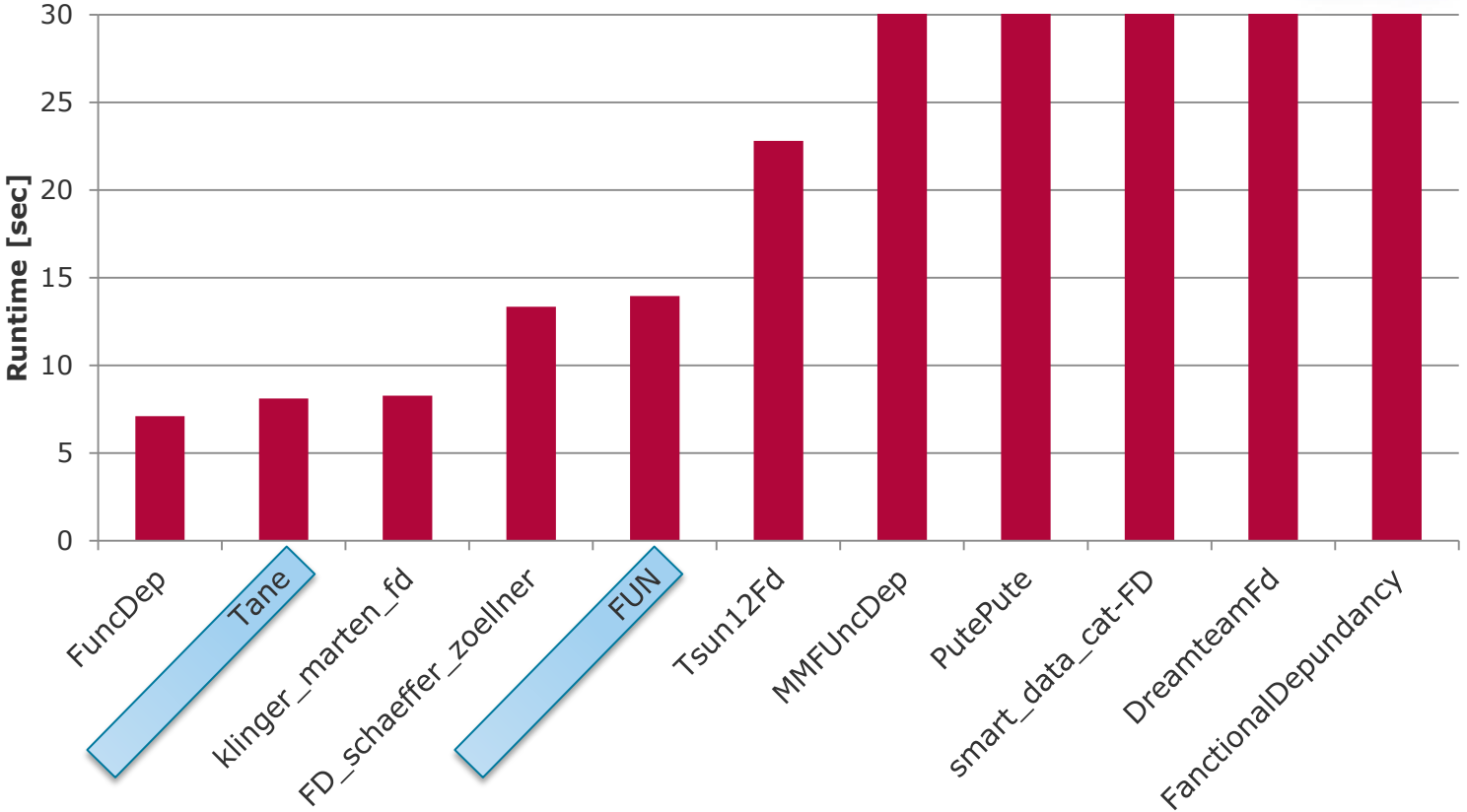


- Columns: **30**
- Rows: **250,000**
- FDs: **89,571**

Data Profiling with Metanome

ThorstenPapenbrock,
PhD Candidate,
17th November, 2014
Chart **22**

Exercise 3
Runtime for fd-reduced-15.csv (<30sec)



- Columns: **30**
- Rows: **250,000**
- FDs: **89,571**

Exercise 3
Correctness for plista1k.csv

aiwendil	**incorrect**
AlexoFredFunctionals	**> 1h ?**
dennis_marius.fd	**> 1h ?**
DJ_FD	**> 1h ?**
dpdc-cnms-fd	**SerializationError**
DreamteamFd	
FanctionalDepundancy	
fastTane	**SerializationError**
fd_finke_dullweber	**Incorrect**
FD_grundke_wiese	**> 1h**
FD_JungRohloff	**> 1h ?**
FD_Kirsten_Zwerg	**> 1h ?**
FD_schaeffer_zoellner	
FD_SPIRO	**> 30min**
FdBotheJoerkeReissaus	**> 1h ?**
FDFMJR	**> 1h**
FdPerchykSchmidt	**SerializationError**
FrohnOttoFuncDep-LAME-TANE	**> 1h ?**
FuncDep	
FunctionalDependencyDetector	**> 1h ?**
FunctionalDerpendency	**incorrect**
GottaCatchAllFD	**> 1h ?**
HorLehTane	**> 30min**
klinger_marten_fd	
LucieKerstinFD	**> 1h ?**
MMFUncDep	
MyFd	**> 1h ?**
PCFD	**> 1h**
PutePute	
RT_FD	**> 30min**
SBMMFD	**> 1h ?**
smart_data_cat-FD	
Tsun12Fd	
YuckFunc	**> 1h**

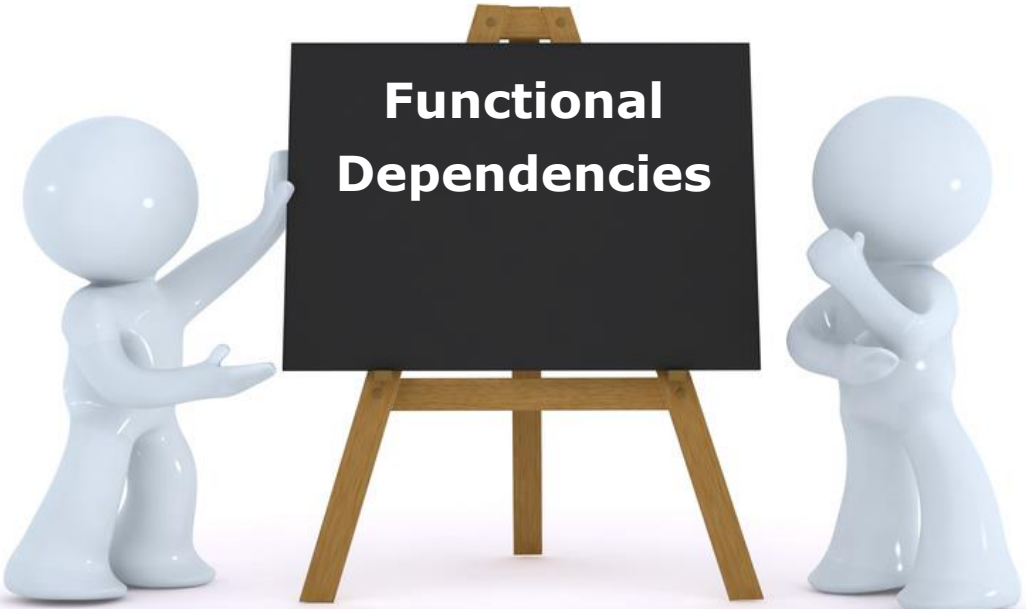
Exercise 3
Correctness for plista1k.csv

aiwendil	**incorrect**
AlexoFredFunctionals	**> 1h ?**
dennis_marius.fd	**> 1h ?**
DJ_FD	**> 1h ?**
dpdc-cnms-fd	**SerializationError**
DreamteamFd	**> 1h ?**
FanctionalDepundancy	**> 1h ?**
fastTane	**SerializationError**
fd_finke_dullweber	**Incorrect**
FD_grundke_wiese	**> 1h**
FD_JungRohloff	**> 1h ?**
FD_Kirsten_Zwerg	**> 1h ?**
FD_schaeffer_zoellner	**OutOfMemory**
FD_SPIRO	**> 30min**
FdBotheJoerkeReissaus	**> 1h ?**
FDFMJR	**> 1h**
FdPerchykSchmidt	**SerializationError**
FrohnOttoFuncDep-LAME-TANE	**> 1h ?**
FuncDep	
FunctionalDependencyDetector	**> 1h ?**
FunctionalDerpendency	**incorrect**
GottaCatchAllFD	**> 1h ?**
HorLehTane	**> 30min**
klinger_marten_fd	**OutOfMemory**
LucieKerstinFD	**> 1h ?**
MMFUncDep	**OutOfMemory**
MyFd	**> 1h ?**
PCFD	**> 1h**
PutePute	**ArrayIndexOutOfBounds**
RT_FD	**> 30min**
SBMMFD	**> 1h ?**
smart_data_cat-FD	**> 1h**
Tsun12Fd	**> 1h**
YuckFunc	**> 1h**

Exercise 3
Correctness for plista1k.csv

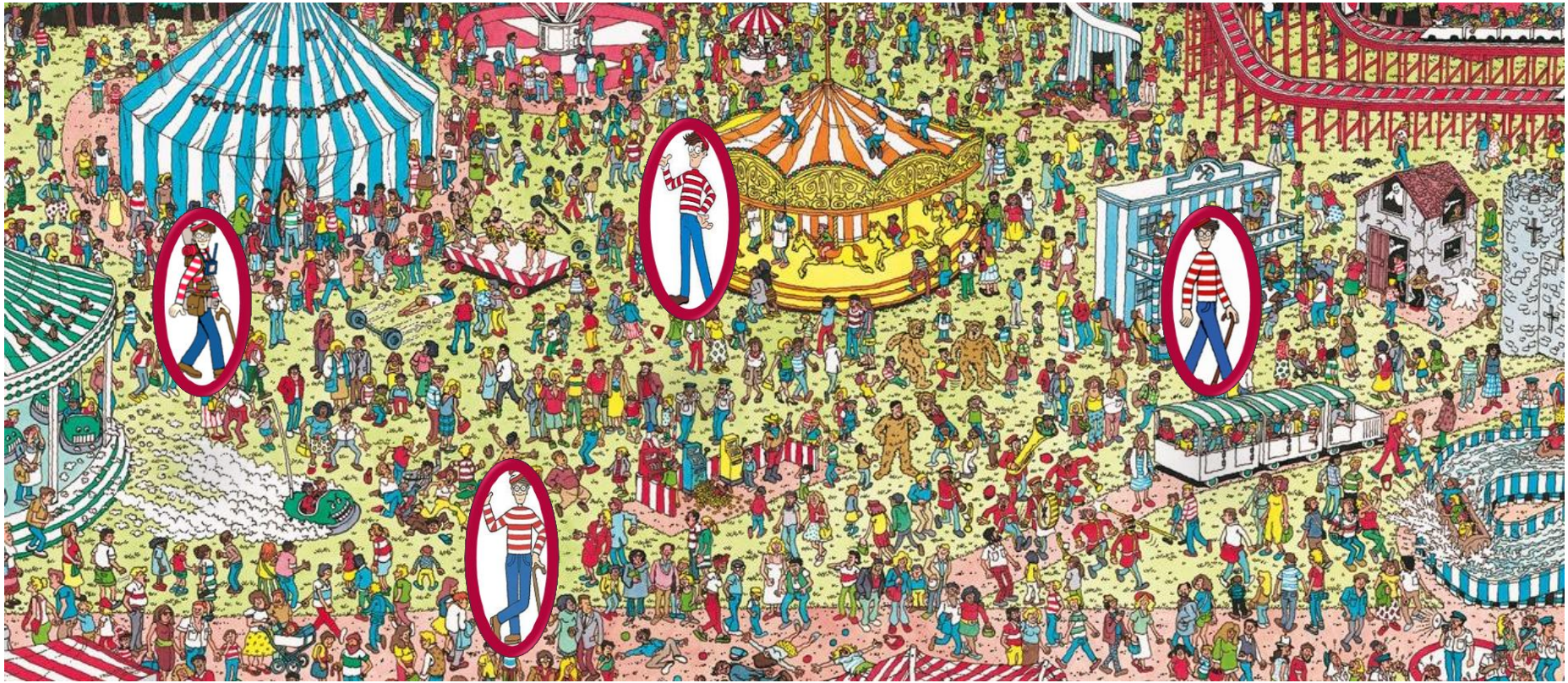
Algorithm	Runtime [ms]
FuncDep	7,043
fdep	18,492
TANE	OutOfMemory
FUN	OutOfMemory

Exercise 3
Short presentations – Part 2



Data Cleansing
Duplicate Detection

Hasso Plattner Institut



Hasso Plattner Institut

Exercise 4
Duplicate Detection

- Deadline: **Monday, 26.01.15**
- The admission to the exam requires *all* exercises to be solved.
- The exercises should be solved in teams of two students.
- The datasets and supplemental material can be found at network drive S:
  `\\fs3\bbs\DPDC`
- The submission system can be found at:
  `https://www.dcl.hpi.uni-potsdam.de/submit/`
- To solve this exercise, please submit a zip file containing the following items:
  - **<algorithm_name>.jar**: An executable Duplicate Detection algorithm.
  - **<algorithm_name>.zip**: The algorithm's source code.
  - **<algorithm_name>_docu.pdf**: Short documentation of the algorithm.
  - **<algorithm_name>_pres.pptx/ppt/pdf**: Two slides presentation of the algorithm.
  - **results.txt**: The results file listing all discovered duplicates.

Exercise 4
Duplicate Detection

Task 1: Duplicate Detection - A discovery algorithm

Use Case: A duplicate is a pair of two records that both represent the same real-world entity. Usually, duplicates constitute quality issues, which is why they need to be detected and corrected. In this exercise, we inspect an address dataset containing the names, streets, towns, and phone numbers of one million (fictional) persons. Our goal is to clean the address dataset from duplicate entries describing same persons.

Task: Write an algorithm that detects possibly many duplicates in a given dataset. At the same time, the algorithm should report possibly few false duplicates, i.e. record pairs that describe different real-world entities, because cleaning such records leads to data loss. The rules for your implementation are as follows:

Exercise 4
Duplicate Detection

a) Write your own algorithm and do not use existing duplicate detection frameworks. You may, however, use external libraries, e.g. for Levenshtein or Jaro-Winkler String comparisons, but we recommend to implement these on your own as well for exercise.

b) Your algorithm should take the path to the input dataset as command line parameter. Hence, the following command should execute your algorithm on the *addresses.csv*-file if this is located next to the jar-file:

```
java -jar <algorithm_name>.jar addresses.csv
```

c) The output of your algorithm is a file called *results.txt* that contains all discovered duplicates. Each line in this file is a comma-separated pair of two record IDs representing one duplicate each. If, for instance, the records 128 and 329, 245 and 453, and 353 and 972 are duplicates, the result file would look like this:

```
128,329
245,453
353,972
```

The ID of a record is the record's first value. Therefore, the following record has the ID 128:

```
"128","Herr","Jon","Pütz","Weserdeich","2","26919","Brake","60909"
```

Exercise 4
Duplicate Detection

Evaluation: In this exercise, we evaluate the precision and recall of the duplicates that you discover on the address dataset. It is, therefore, important to find a good similarity function for the record comparisons. To test your algorithm, you can use the cd and restaurant datasets. Both datasets can be found on the bbs share with their gold standards.

Hints: We do not evaluate the runtime of your algorithm, but you still need a good pair selection technique, because you do not have enough time to compare all records. To maximize precision and recall, you can also use clustering, parallelization, machine learning, or any other technique that supports your duplicate detection process. It could further help to calculate the transitive closure on your results, but note that most duplicates in the address dataset are pairwise duplicates, i.e. the cluster size is mostly two records.

Exercise 4
Duplicate Detection

Task 2: Documentation

Write a short (max one A4 page) documentation for your algorithm describing the algorithm that you implemented and the techniques that you used.

In the same document, answer the following questions:

a) How many duplicates did your algorithm find in the address dataset?

b) How long did the discovery take on the address dataset and what machine did you use?

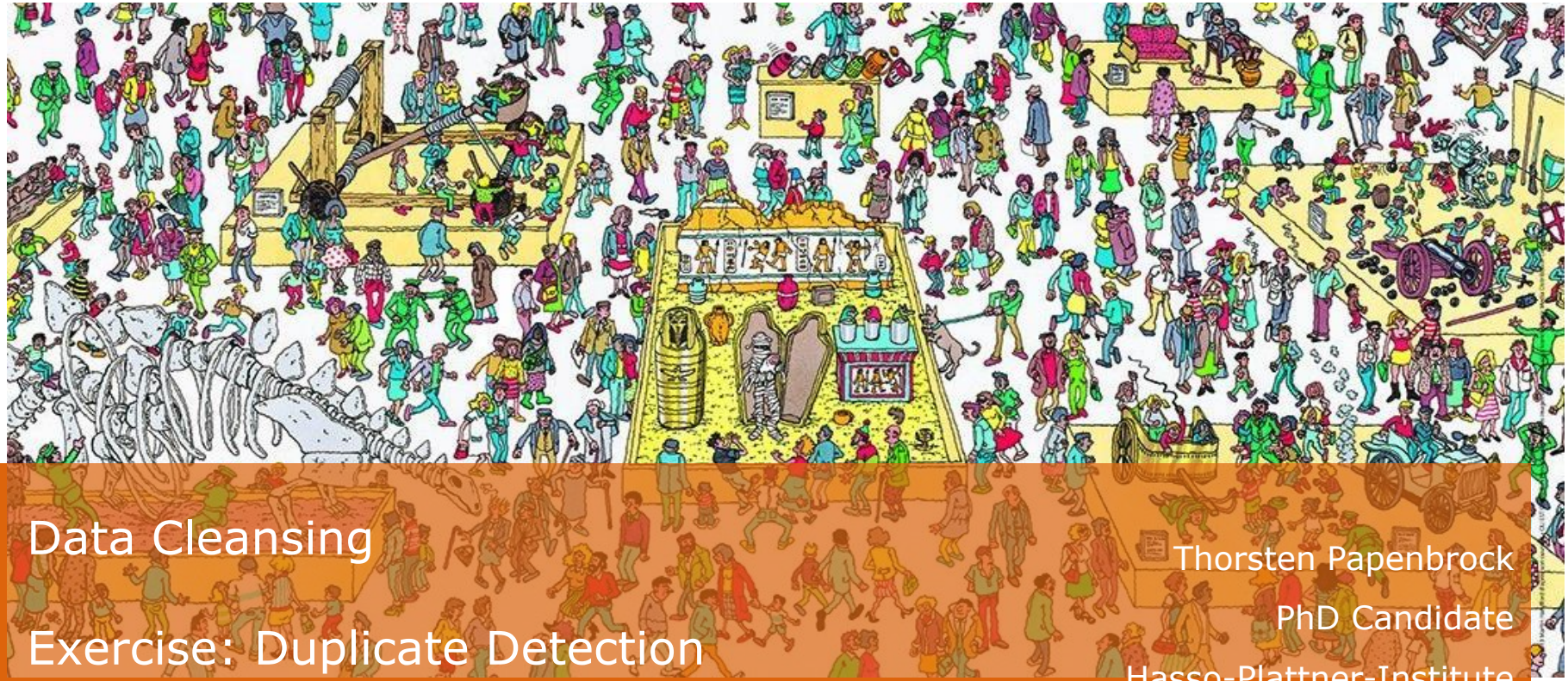
c) Did you discover any challenges of your approach, e.g. in runtime or memory consumption?

d) Analyze the results that your algorithm has found: List 3 record pairs (the complete tuples) that you think are *true* duplicates and 3 record pairs that you think are *false* duplicates.

Task 3: Presentation

Prepare two slides for a short, 5 min presentation of your algorithm in the lecture. One slide about the algorithm and one slide about the duplicates you discovered.

Note that each team will present its work once!



Data Cleansing

Exercise: Duplicate Detection

Thorsten Papenbrock
PhD Candidate
Hasso-Plattner-Institute