CS 6400 A

Database Systems Concepts and Design

Lecture 21

12/02/24

Announcements

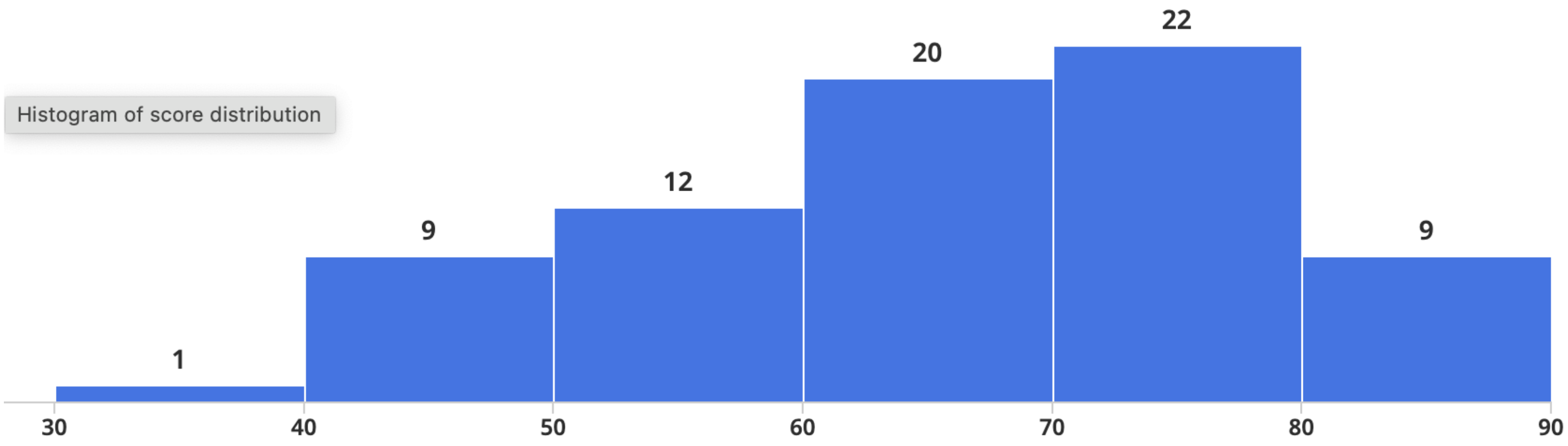
Exam 2 stats (will publish tonight):

- max: 86 (96%), median: 67.5 (75%), mean: 65.55 (72.8%), std: 11.89
- Solution posted on canvas (under Files/Midterm solution)
- Regrade request open until next Monday (Dec 9)

Assignment 3 grades published

Project presentation

- Video due night
- Demo: Dec 6



CIOS is open

Today’s class

1. Data Cleaning

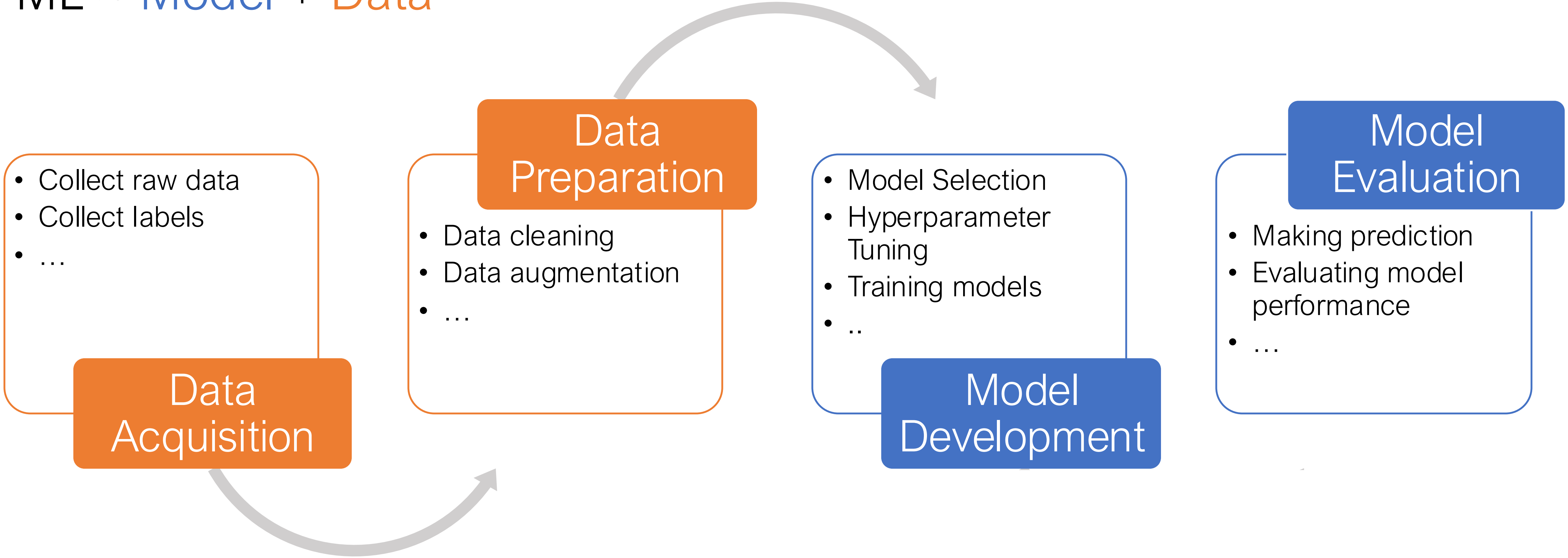
2. Data Labeling

3. Course Summary

The ML lifecycle in a bird's eye view

"Only a fraction of real-world ML systems is composed of ML code" [1]

ML ≈ Model + Data



[1] Sculley, David, et al. "Hidden technical debt in machine learning systems." NeurIPS 2015

Data is the Bottleneck in the lifecyle

ML ≈ Model + Data

Model is gradually commoditized

- Transformers for “all” tasks
- Out-of-the-box invocation of ML libraries gives decent results

Data remains the bottleneck

- Collecting and storing raw data is becoming cheaper
- Turning them into ML-ready datasets is not

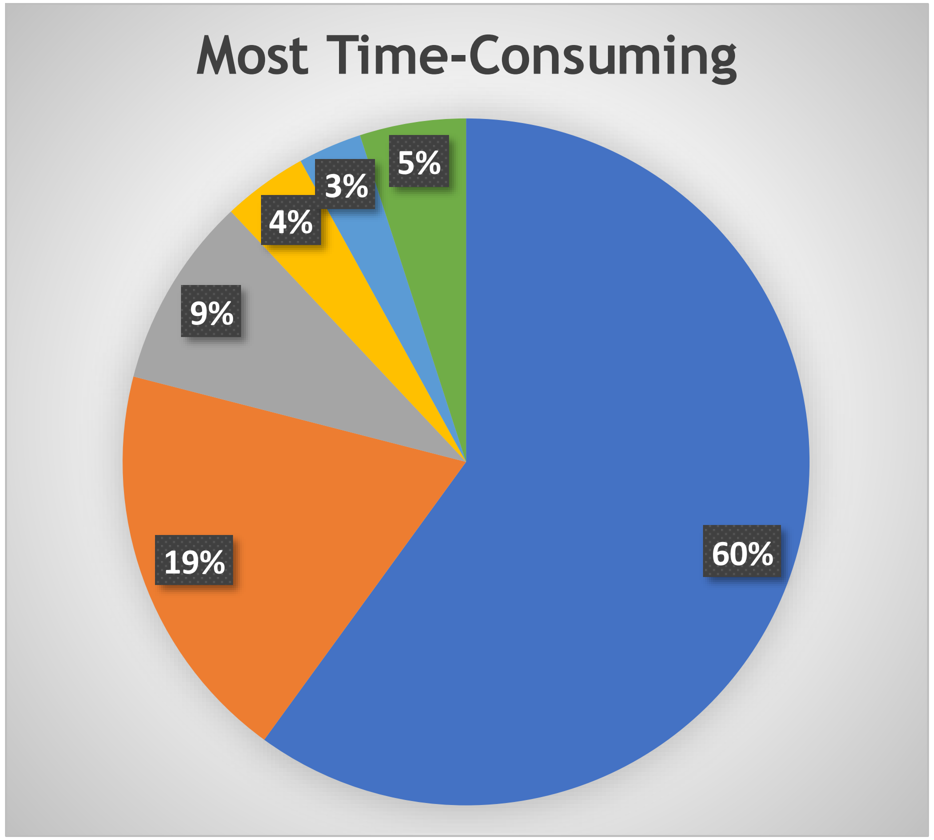
Sources:
https://www.techspot.com/news/98600-chatgpt-powered-hidden-army-contractors-making-15-hour.html

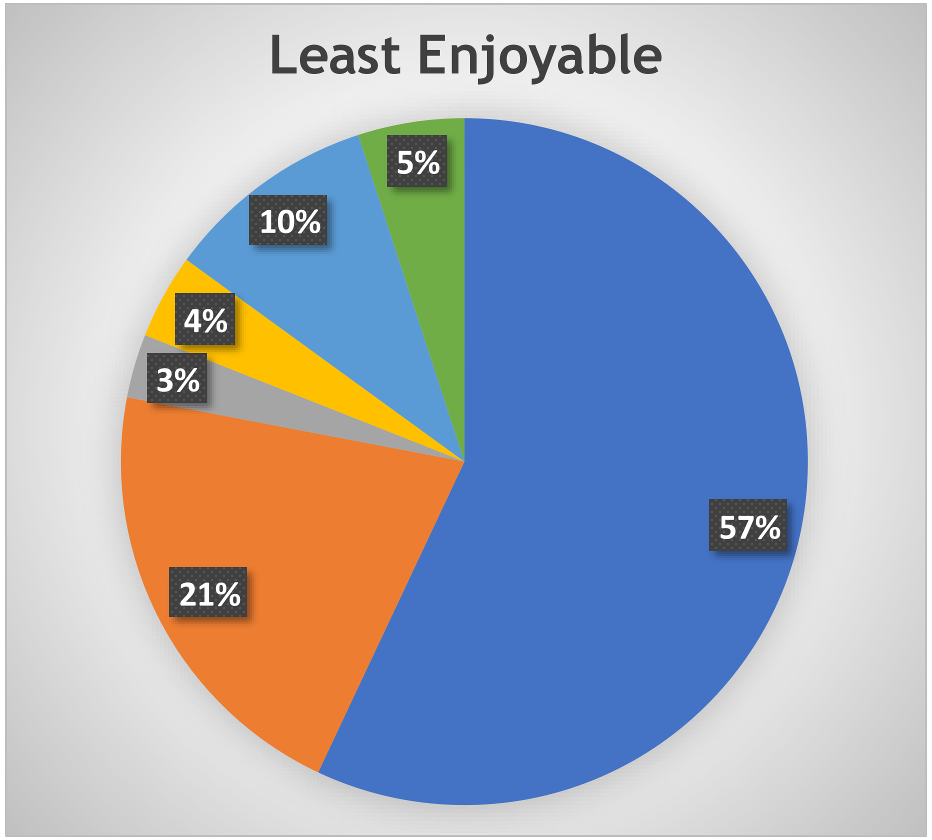
Q: How much time do you think data scientists spend on cleaning data?

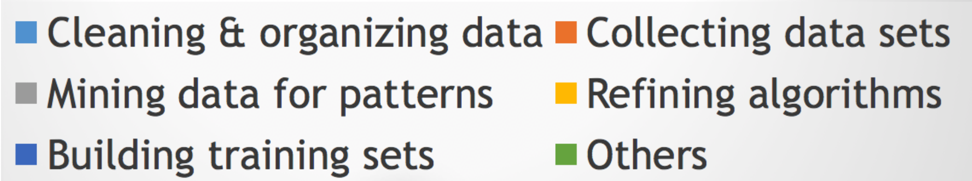
- A: <20%
- B: 20-50%
- C: 50-80%
- D: >80%

Cleaning Data: Most Time-Consuming, Least Enjoyable Data Science Task

Forbes, 2016







Common Data Problems

Incomplete

Country	UN R/P 10%[4]	UN R/P 20%[5]	World Bank Gini (%)[6]	WB Gini (year)	CIA R/P 10%[7]	Year	CIA Gini (%)[8]	CIA Gini (year)	GPI Gini (%)[9]
Seychelles			65.8	2007					
Comoros			64.3	2004					
Namibia	106.6	56.1	63.9	2004	129.0	2003	59.7	2010	
South Africa	33.1	17.9	63.1	2009	31.9	2000	65.0	2005	
Botswana	43.0	20.4	61.0	1994			63	1993	
Haiti	54.4	26.6	59.2	2001	68.1	2001	59.2	2001	
Angola			58.6	2000					62.0
Honduras	59.4	17.2	57.0	2009	35.2	2003	57.7	2007	

Adapted from Intro to Data Cleaning lecture from Xu Chu

Common Data Problems

Inconsistent

Financial

Employee	Salary
John	1000

Employee $\rightarrow$ Salary

Human Resources

Employee	Salary
John	2000
Mary	3000

Employee $\rightarrow$ Salary

Target Database

Employee	Salary
John	**1000**
John	**2000**
Mary	3000

Employee $\rightarrow$ Salary

Mapping

Financial(e,s) $\subseteq$ Global(e,s)

HumanRes(e,s) $\subseteq$ Global(e,s)

Common Data Problems

Inaccurate

Sheepdog or mop?

Poodle or fried chicken?

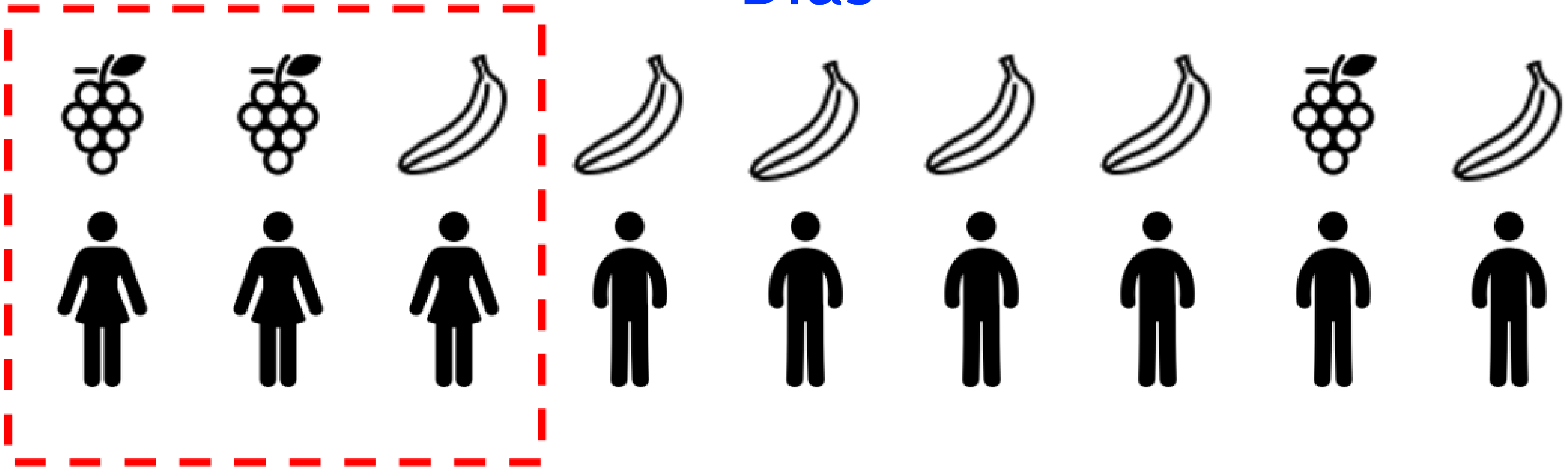
Fox or dog?

Common Data Problems

Outliers

Common Data Problems

Bias



Model amplifies data biases

Example: Buolamwini and Gebru (2018). Gender Shades

Gender Classifier	Darker Male	Darker Female	Lighter Male	Lighter Female	Largest Gap
Microsoft	94.0%	79.2%	100%	98.3%	20.8%
FACE++	99.3%	65.5%	99.2%	94.0%	33.8%
IBM	88.0%	65.3%	99.7%	92.9%	34.4%

Dirty Data is Costly

UNITED STATES POSTAL SERVICE

- **Address errors caused 6.8 billion undelivered mails in 2013**
- **Estimated $1.5 billion spent on processing**
- **At least $3.4 billion wasted postage**

Harvard Business Review

DATA

Bad Data Costs the U.S. $3 Trillion Per Year

by Thomas C. Redman

SEPTEMBER 22, 2016

1. Data Cleaning

Data Cleaning for Structured Data

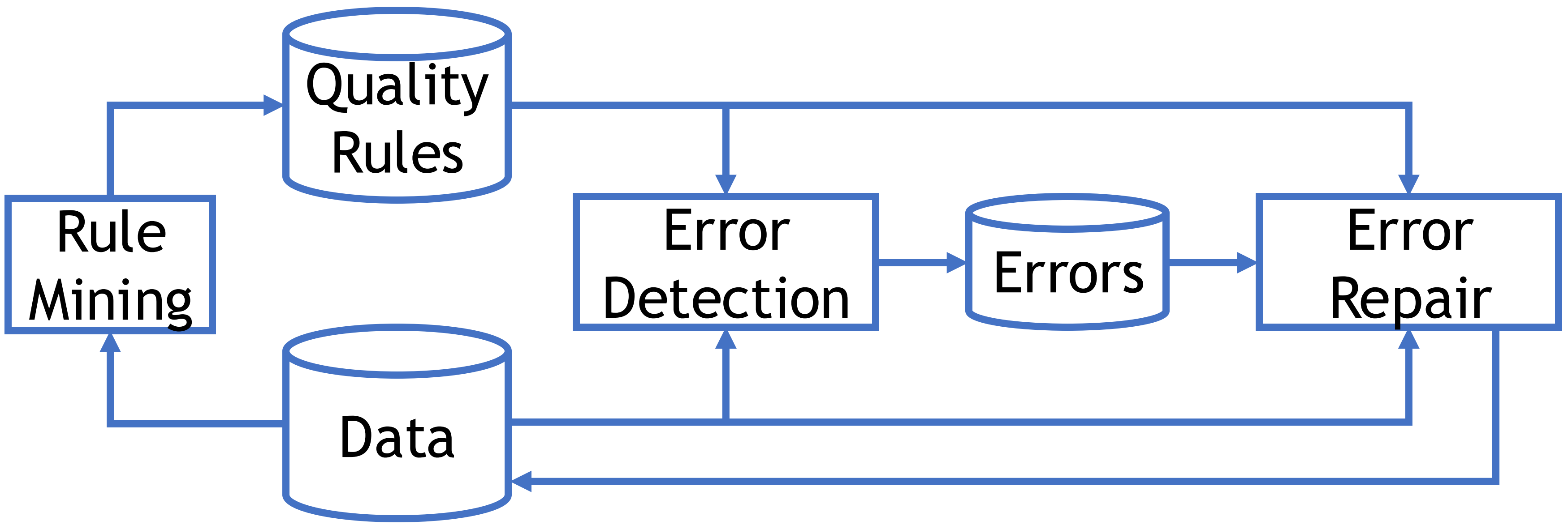
Detect and repair errors in a structured dataset

- [Discovering denial constraints](). [VLDB'13]
- [HoloClean: Holistic Data Repairs with Probabilistic Inference](). [VLDB'17]

Data cleaning and machine learning

- Cleaning before ML
- Cleaning for ML

Two tasks in data cleaning



- Detection: A minimal set of cells that cannot coexist together
- Repair: A set of cell updates to resolve the violations

Data Quality Rules

	Name	ID	LVL	ZIP	ST	SAL
t_1	Alice	ID1	5	10001	NM	90K
t_2	Bob	ID2	6	87101	NM	80K
t_3	Chris	ID3	4	10001	NY	80K
t_4	Dave	ID4	1	90057	CA	20K
t_5	Frank	ID5		90057	CA	50K

R1: Two persons with the same ZIP live in the same ST

Data Quality Rules

	Name	ID	LVL	ZIP	ST	SAL
t_1	Alice	ID1	5	10001	NM	90K
t_2	Bob	ID2	6	87101	NM	80K
t_3	Chris	ID3	4	10001	NY	80K
t_4	Dave	ID4	1	90057	CA	20K
t_5	Frank	ID5		90057	CA	50K

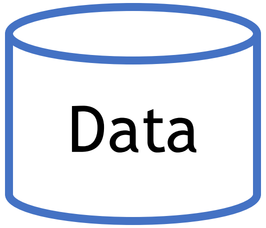
R2: LVL should not be empty

Data Quality Rules

	Name	ID	LVL	ZIP	ST	SAL
t_1	Alice	ID1	5	10001	NM	90K
t_2	Bob	ID2	6	87101	NM	80K
t_3	Chris	ID3	4	10001	NY	80K
t_4	Dave	ID4	1	90057	CA	20K
t_5	Frank	ID5		90057	CA	50K

R3: People with a higher LVL earn more SAL in the same ST

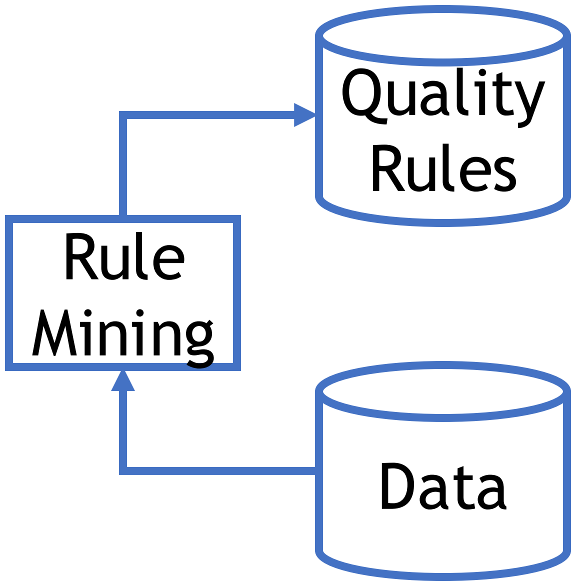
Rule-based Data Cleaning



Name	ZIP	ST
Alice	10001	NM
Bob	87101	NM
Chris	10001	NY

Adapted from Intro to Data Cleaning lecture from Xu Chu

Rule-based Data Cleaning

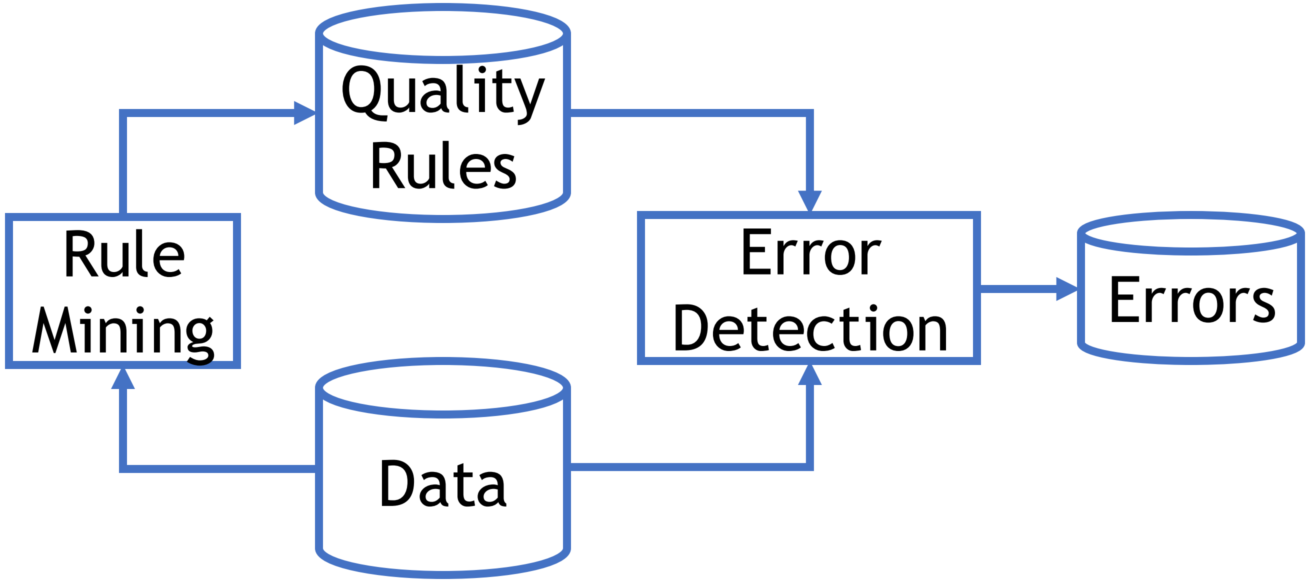


Name	ZIP	ST
Alice	10001	NM
Bob	87101	NM
Chris	10001	NY

Two persons with the same ZIP live in the same ST

Adapted from Intro to Data Cleaning lecture from Xu Chu

Rule-based Data Cleaning

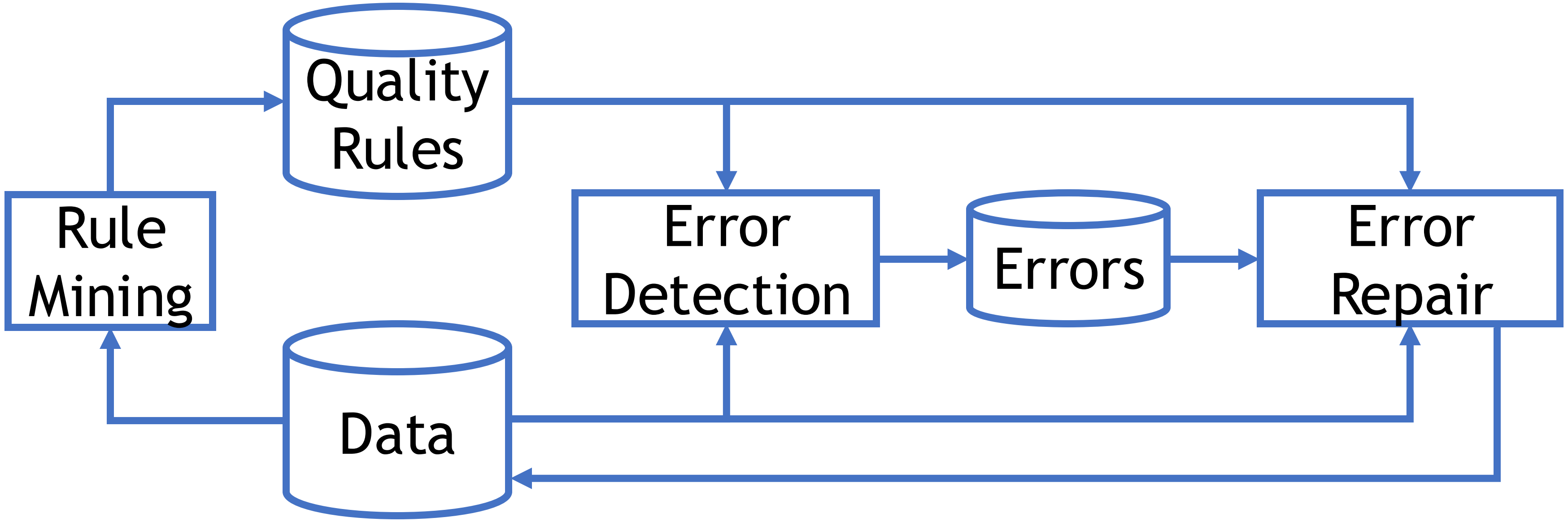


Name	ZIP	ST
Alice	10001	NM
Bob	87101	NM
Chris	10001	NY

Two persons with the same ZIP live in the same ST

Adapted from Intro to Data Cleaning lecture from Xu Chu

Rule-based Data Cleaning

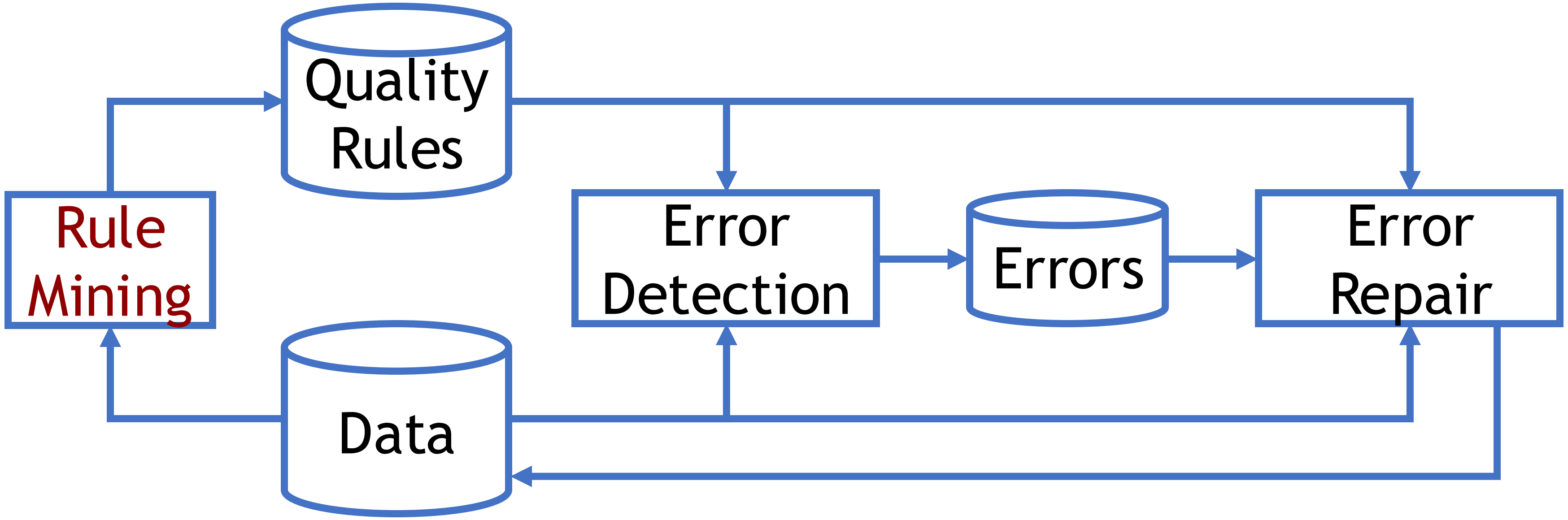


Name	ZIP	ST
Alice	10001	NY
Bob	87101	NM
Chris	10001	NY

Two persons with the same ZIP live in the same ST

Adapted from Intro to Data Cleaning lecture from Xu Chu

Discovering denial constraints [VLDB'13]



Can ask a domain expert, but takes too much time
Automatically discover quality rules in the form of Denial Constraints

R1: Two persons with the same ZIP live in the same ST

$$\forall t_\alpha, t_\beta \neg(t_\alpha.ZIP = t_\beta.ZIP \wedge t_\alpha.ST \neq t_\beta.ST)$$

Adapted from Intro to Data Cleaning lecture from Xu Chu

Examples of Discovered DCs

On a tax dataset

$$\forall t_\alpha \ \neg(t_\alpha.ST = \text{“FL”} \wedge t_\alpha.ZIP < 30397)$$

State Florida’s ZIP code cannot be lower than 30397.

$$\forall t_\alpha \ \neg(t_\alpha.MS \neq \text{“Single”} \wedge t_\alpha.STX \neq 0)$$

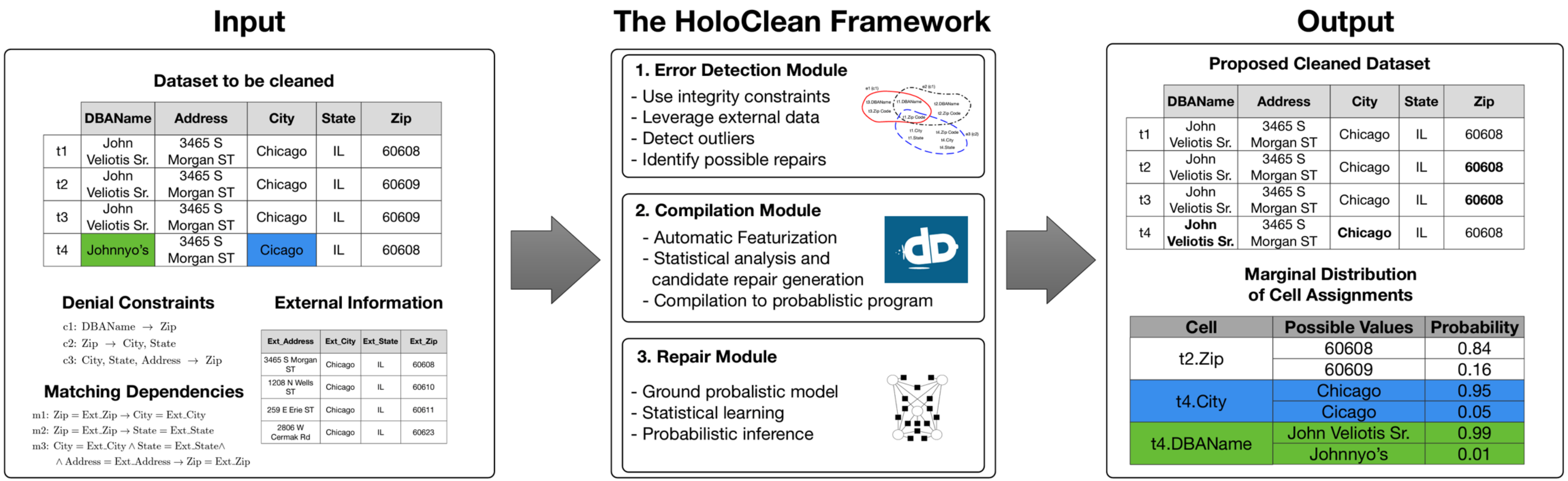
One has to be single to have any single tax exemption.

$$\forall t_\alpha, t_\beta \ \neg(t_\alpha.ST = t_\beta.ST \wedge t_\alpha.SAL < t_\beta.SAL \wedge t_\alpha.TR > t_\beta.TR)$$

There cannot exist two persons who live in the same state, but one person earns less salary and has higher tax rate at the same time.

Adapted from Intro to Data Cleaning lecture from Xu Chu

[HoloClean: Holistic Data Repairs with Probabilistic Inference](). [VLDB'17]



Probabilistic model that unifies different signals for repairing a dataset.

Adapted from Intro to Data Cleaning lecture from Prof. Xu Chu

Constraints and minimality

Functional dependencies

$$c1:\ \text{DBAName} \rightarrow \text{Zip}$$
$$c2:\ \text{Zip} \rightarrow \text{City, State}$$
$$c3:\ \text{City, State, Address} \rightarrow \text{Zip}$$

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	***Chicago***	IL	***60608***
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	***60609***
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	***60609***
t4	***Johnnyo's***	Johnnyo's	3465 S Morgan ST	***Cicago***	IL	60608

Bohannon et al., 2005, 2007; Kolahi and Lakshmanan , 2005; Bertossi et al., 2011; Chu et al., 2013; 2015 Fagin et al., 2015

Constraints and minimality

Functional dependencies

c1: DBAName $\rightarrow$ Zip

c2: Zip $\rightarrow$ City, State

c3: City, State, Address $\rightarrow$ Zip

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t4	**Johnnyo's**	Johnnyo's	3465 S Morgan ST	**Cicago**	IL	60608

Action: Fewer erroneous than correct cells; perform minimum number of changes to satisfy all constraints

Constraints and minimality

Functional dependencies

c1: DBAName $\rightarrow$ Zip
c2: Zip $\rightarrow$ City, State
c3: City, State, Address $\rightarrow$ Zip

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t4	**Johnnyo's**	Johnnyo's	3465 S Morgan ST	**Cicago**	IL	60608

Error; correct zip code is 60608

Does not fix errors and introduces new ones.

External Information

Matching dependencies

$$\text{m1: } \mathrm{Zip} = \mathrm{Ext\_Zip} \rightarrow \mathrm{City} = \mathrm{Ext\_City}$$

$$\text{m2: } \mathrm{Zip} = \mathrm{Ext\_Zip} \rightarrow \mathrm{State} = \mathrm{Ext\_State}$$

$$\text{m3: } \mathrm{City} = \mathrm{Ext\_City} \wedge \mathrm{State} = \mathrm{Ext\_State} \wedge \wedge \mathrm{Address} = \mathrm{Ext\_Address} \rightarrow \mathrm{Zip} = \mathrm{Ext\_Zip}$$

External list of addresses

Ext_Address	Ext_City	Ext_State	Ext_Zip
3465 S Morgan ST	Chicago	IL	60608
1208 N Wells ST	Chicago	IL	60610

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	***Chicago***	IL	***60608***
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	***60609***
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	***60609***
t4	***Johnnyo's***	Johnnyo's	3465 S Morgan ST	***Cicago***	IL	60608

Fan et al., 2009; Bertossi et al., 2010; Chu et al., 2015

External Information

Matching dependencies

$$\text{m1: } \text{Zip} = \text{Ext\_Zip} \rightarrow \text{City} = \text{Ext\_City}$$
$$\text{m2: } \text{Zip} = \text{Ext\_Zip} \rightarrow \text{State} = \text{Ext\_State}$$
$$\text{m3: } \text{City} = \text{Ext\_City} \wedge \text{State} = \text{Ext\_State} \wedge$$
$$\wedge \text{ Address} = \text{Ext\_Address} \rightarrow \text{Zip} = \text{Ext\_Zip}$$

External list of addresses

Ext_Address	Ext_City	Ext_State	Ext_Zip
3465 S Morgan ST	Chicago	IL	60608
1208 N Wells ST	Chicago	IL	60610

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60608
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60608**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60608**
t4	**Johnnyo's**	Johnnyo's	3465 S Morgan ST	**Chicago**	IL	60608

Action: Map external information to input dataset using matching dependencies and repair disagreements

External Information

Matching dependencies

$$\text{m1: } Zip = Ext\_Zip \rightarrow City = Ext\_City$$

$$\text{m2: } Zip = Ext\_Zip \rightarrow State = Ext\_State$$

$$\text{m3: } City = Ext\_City \wedge State = Ext\_State \wedge$$
$$\wedge\ Address = Ext\_Address \rightarrow Zip = Ext\_Zip$$

External list of addresses

Ext_Address	Ext_City	Ext_State	Ext_Zip
3465 S Morgan ST	Chicago	IL	60608
1208 N Wells ST	Chicago	IL	60610

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60608
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60608**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60608**
t4	**Johnnyo's**	Johnnyo's	3465 S Morgan ST	**Chicago**	IL	60608

External dictionaries may have limited coverage or not exist altogether

Quantitative Statistics

Reason about co-occurrence of values across cells in a tuple

Estimate the distribution governing each attribute

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	***Chicago***	IL	*60608*
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	***60609***
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	***60609***
t4	***Johnnyo's***	Johnnyo's	3465 S Morgan ST	***Cicago***	IL	60608

Example: Chicago co-occurs with IL

Hellerstein, 2008; Mayfield et al., 2010; Yakout et al., 2013

Quantitative Statistics

Reason about co-occurrence of values across cells in a tuple

Estimate the distribution governing each attribute

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60608
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60609
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60609
t4	**John Veliotis Sr.**	Johnnyo's	3465 S Morgan ST	**Chicago**	IL	60608

Again, fails to repair the wrong zip code

Combining Everything

Constraints and minimality

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t4	**Johnnyo's**	Johnnyo's	3465 S Morgan ST	**Cicago**	IL	60608

External data

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60608
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60608**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60608**
t4	**Johnnyo's**	Johnnyo's	3465 S Morgan ST	**Chicago**	IL	60608

Quantitative statistics

	DBAName	AKAName	Address	City	State	Zip
t1	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	60608
t2	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t3	John Veliotis Sr.	Johnnyo's	3465 S Morgan ST	Chicago	IL	**60609**
t4	**John Veliotis Sr.**	Johnnyo's	3465 S Morgan ST	**Chicago**	IL	60608

Different solutions suggest different repairs

HoloClean: a probabilistic model for data repairs

	Address	City	State	Zip
t1	3465 S Morgan ST	*Chicago*	IL	*60608*
t2	3465 S Morgan ST	Chicago	IL	*60609*
t3	3465 S Morgan ST	Chicago	IL	*60609*
t4	3465 S Morgan ST	*Cicago*	IL	*60608*

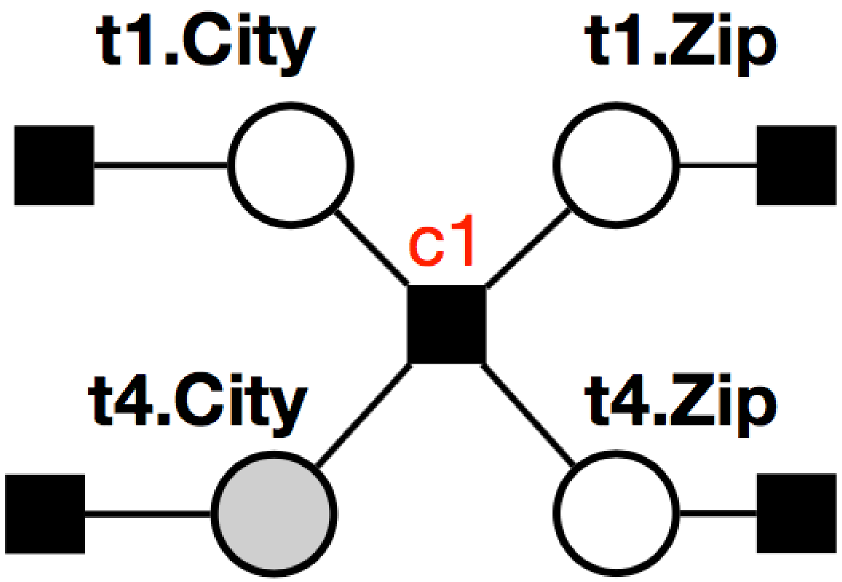
Each cell is a random variable

Value co-occurences capture data statistics

Constraints introduce correlations

$$\text{c1: Zip} \rightarrow \text{City}$$

"Address= 3465 S Morgan St"

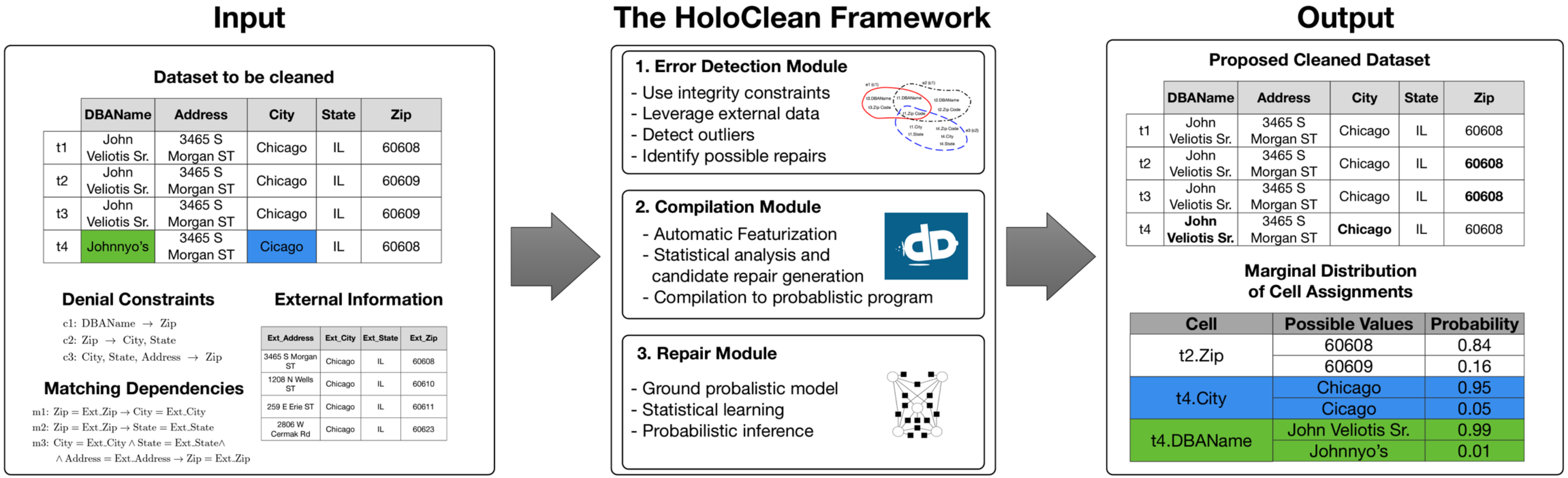


: Unknown (to be inferred) RV

: Observed (fixed) RV

: Factor (encodes correlations)

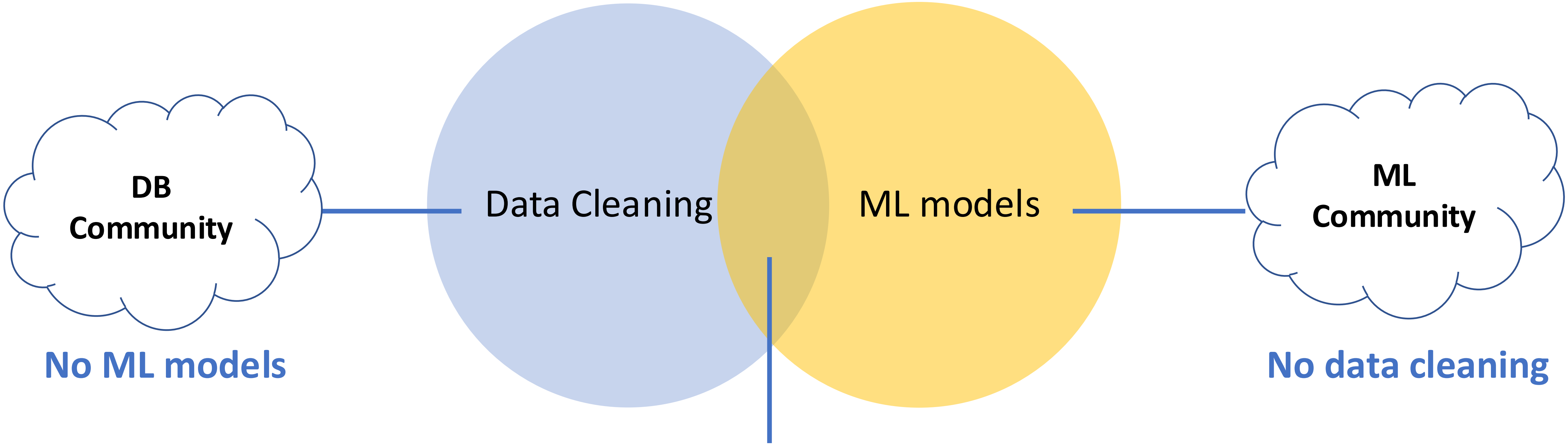
HoloClean: Holistic Data Repairs with Probabilistic Inference. [VLDB'17]



Probabilistic model that unifies different signals for repairing a dataset.

Adapted from Intro to Data Cleaning lecture from Prof. Xu Chu

Data cleaning and ML



The impact of data cleaning on downstream ML models?

Data cleaning and ML

Cleaning ”before” ML:

- Perform cleaning independently of the downstream ML applications; leverage user-specified signals or data-driven approaches
- Example: HoloClean: Holistic Data Repairs with Probabilistic Inference
  - Also an example of using ML for data cleaning

Reading: From Cleaning Before ML to Cleaning For ML

Data cleaning and ML

Cleaning “for” ML:

- Leverage the downstream ML model or application to define cleaning signals that incorporates high-level semantics
- Why is this a good idea?
  - Clean datasets that contain fully correct attributes are rarely available
  - Data cleaning can sometimes negatively impact the performance of ML models
    - CleanML: A Study for Evaluating the Impact of Data Cleaning on ML Classification Tasks
- Example: BoostClean: Automated Error Detection and Repair for Machine Learning

Reading: From Cleaning Before ML to Cleaning For ML

Data preprocessing

- Data preprocessing transforms raw data into a representation that is more suitable for the downstream ML model
- Data cleaning is usually performed as a part of the data preprocessing step
  - Missing value: mean/median imputation, frequent value imputation
  - Outlier removal: z-score, MAD, IQR…
- Also includes:
  - Normalization: min-max, standardization …
  - Discretization: uniform, quantile …
- Different from feature engineering, which creates new features from existing data

2. Data Labeling

Data Labeling

Data is the Bottleneck for ML

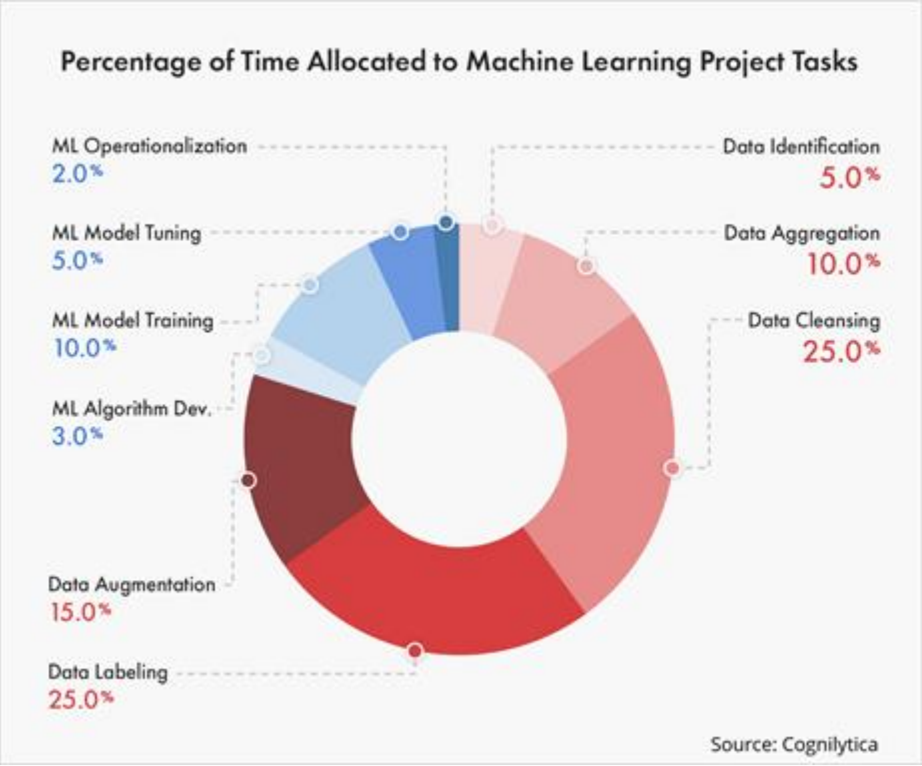
ML ≈ Model + Data

Model is gradually commoditized

- Out-of-the-box invocation of ML libraries gives decent results
- Transformers for “all” tasks

Data is the bottleneck

OpenAI has hired an army of contractors
to do what’s called “data labeling”



Sources:
https://www.semafor.com/article/01/27/2023/openai-has-hired-an-army-of-contractors-to-make-basic-coding-obsolete
https://www.datanami.com/2023/01/20/openai-outsourced-data-labeling-to-kenyan-workers-earning-less-than-2-per-hour-time-report/

Manual v.s. Programmatic Labeling

Labeling individual data points

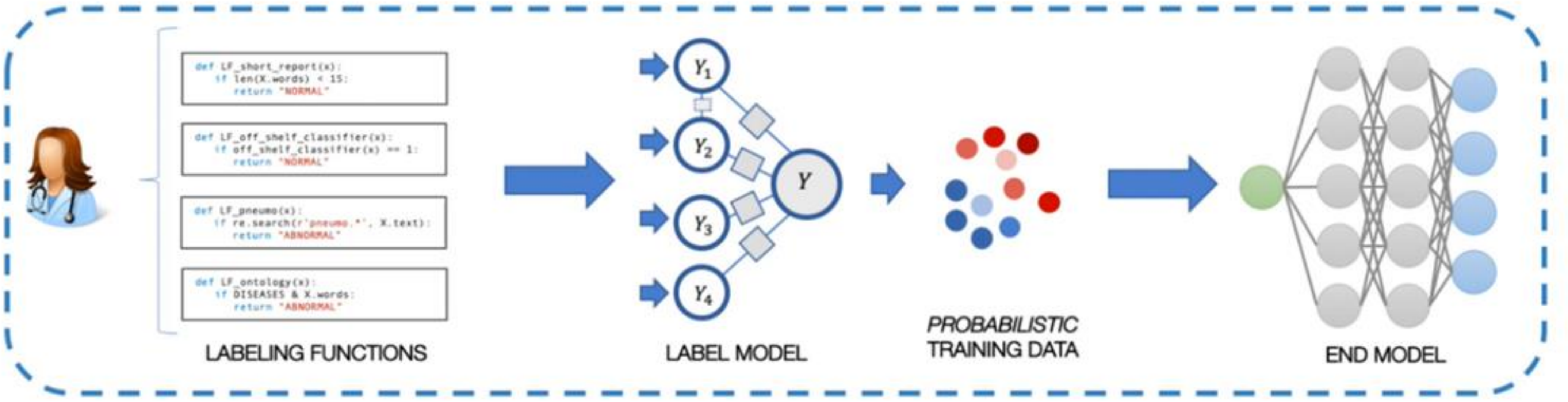
Writing Labeling Functions (LFs) where each LF abstracts a supervision source (e.g. heuristics, existing models, external KBs, …)

```
@labeling_function()
def lf_contains_link(x):
    # Return a label of SPAM if "http" in comment text, otherwise ABSTAIN
    return SPAM if "http" in x.text.lower() else ABSTAIN
```

Source: https://www.snorkel.org/use-cases/01-spam-tutorial#3-writing-more-labeling-functions

Programmatic Labeling Pipeline Overview

Credit: Snorkel Project



(1) Users **write labeling functions** to generate noisy labels

(2) A **label model combines noisy labels** to be probabilistic labels

(3) Using the **probabilistic labels to train** an end ML model

(1) Labeling Function

"Indication: Chest pain. Findings: Focal consolidation and pneumothorax.."

```
def LF_pneumothorax(c):
  if re.search(r'pneumo.*', c.report.text):
    return "ABNORMAL"
```

"Indication: Chest pain. Findings: No focal consolidation or pneumothorax..."

LFs can be noisy!

Source: https://ajratner.github.io/assets/papers/Snorkel_VLDB_2018_slides.pdf

Other Example LFs: Existing Knowledge

- Knowledge bases
  - Match the text inputs against the knowledge base (e.g., DBPedia) to search for known spouse relationships.
- Pretrained models
  - Pre-trained model with a different label space
- Thirty-party tools
  - TextBlob: Simplified Text Processing

How are LFs developed

- By domain experts
- Generate programmatically
  - Snuba: Automating Weak Supervision to Label Training Data. [VLDB'18]
  - Language Models Enable Simple Systems for Generating Structured Views of Heterogeneous Data Lakes

(2) Label Model

1: positive
-1: negative
0: abstain

	LF1	LF2	LF3	LFX
Data point 1	1	-1	0	…
Data point 2	0	1	0	…
Data point 3	1	1	-1	..
Data point 4	1	1	-1	…
Data point 5	1	1	-1	…
Data point x	…	…	…	…

Weak label matrix X

Label model $\longrightarrow$

y
1
1
-1
1
-1
…

Inferred ground-truth labels y

Example label model

Option 1: Majority voting

Q: What if some rules are more reliable than others?

Option 2: Evaluate the accuracy of each labeling function

Example: Dawid and Skene's method

1. Assume accuracies θ of each LF
2. Learn parameter θ with an Expectation and Maximization algorithm:
   a. Initialize y by majority vote
   b. Calculate accuracies θ for each LF
   c. Update y by maximizing $p(X|y, \theta)$

3. Course Summary

Course Summary

We learned…

1. How to query a database

Course Summary

We learned…

1. How to query a database
2. How to design a database

1. Intro

2-3. SQL

4. ER Diagrams

5-6. DB Design

7-9. Storage

11-13. QO

14-16. TXNs

17-21. Beyond RDBMS

Course Summary

We learned…

1. How to query a database

2. **How to design a database**

Course Summary

We learned…

1. How to query a database
2. How to design a database
3. How records are stored and indexed

Course Summary

We learned…

1. How to query a database
2. How to design a database
3. How records are stored and indexed
4. **How to optimize the performance of a database**

Course Summary

We learned…

1. How to query a database
2. How to design a database
3. How records is stored and indexed
4. How to optimize the performance of a database
5. **How to handle concurrent user requests and crashes/aborts**

Course Summary

We learned…

1. How to query a database
2. How to design a database
3. How records is stored and indexed
4. How to optimize the performance of a database
5. How to handle concurrent user requests and crashes/aborts
6. How RDBMS relates to OLAP, Distributed Query Processing etc.

1. Intro

2-3. SQL

4. ER Diagrams

5-6. DB Design

7-9. Storage

11-13. QO

14-16. TXNs

17-21. Beyond RDBMS

Relational databases => data-intensive systems

Most important computer applications must manage, update and query datasets

- Bank, store, search app…

Data quality, quantity & timeliness becoming even more important with AI

- Machine learning = algorithms that generalize from data

Relational databases => data-intensive systems

Relational databases are the most popular type of data-intensive system (MySQL, Oracle, etc)

Many other systems facing similar concerns: key-value stores, streaming systems, ML frameworks, your custom app?

Reliability in the face of crashes, bugs, bad user input, etc
Concurrency: access by multiple users
Performance: throughput, latency, etc
Access interface from many, changing apps
Security and data privacy (not covered in this course)

Beyond this class

Classes:

- CS 4420/6422: Database System Implementation
- CS 6220: Big Data Systems and Analytics
- CS 8803: Data-Centric Machine Learning

DB research at GT:

- [Data Systems and Analytics Group]