Synthesizing Relational Data with Differential Privacy

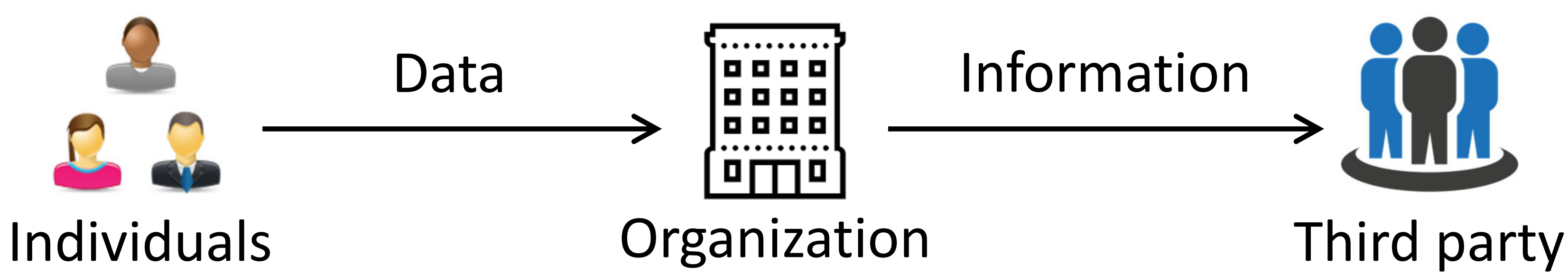
Xiaokui Xiao
School of Computing
National University of Singapore

Outline

- Statistical databases: what and why
- Existing solutions
- The road less travelled: synthetic data
- Conclusion and future work

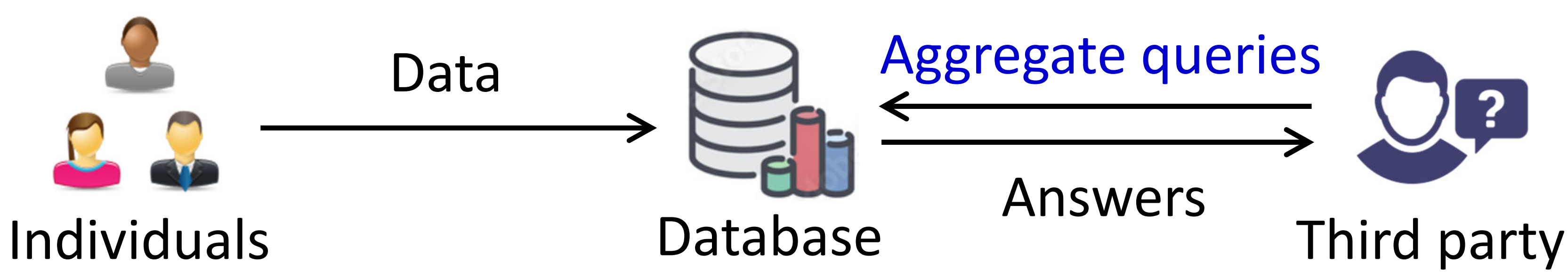
Introduction

- We live in an era where data is constantly being collected, analyzed, and shared
- Protecting privacy while sharing useful information is an important problem



Statistical Databases

- A database that answers only aggregate queries, for privacy protection
- Additional defence by
  - Returning noisy answers, and
  - Denying queries when necessary
- But still non-trivial to ensure privacy protection



Linear Program Reconstruction Attack

- A type of attacks that reconstruct a table *T* from noisy count query results
- Basic idea:
  - Formulate a linear program from the noisy count query results
  - Solve the linear program to infer the tuples in *T*
- How effective is this attack?
  - Even if each count has $o(\sqrt{n})$ noise, we could reconstruct a large portion of the input data, using $O(n \log^2 n)$ random queries
  - n: total number of possible tuples

Dinur and Nissim. Revealing Information while Preserving Privacy. PODS 2003

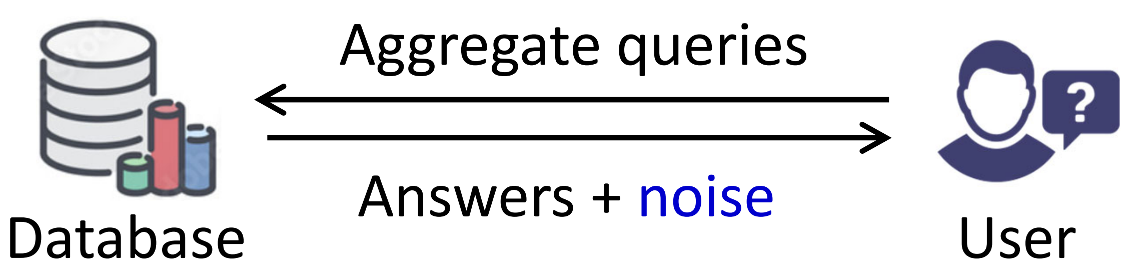
Database Reconstruction in Practice

- The US Census Bureau applied the linear program reconstruction attack on the census data released in 2010
- They were able to reidentified data from 17% of the US population

https://www.census.gov/content/dam/Census/newsroom/press-kits/2019/jsm/presentation-deploying-differential-privacy-for-the-2020-census-of-pop-and-housing.pdf

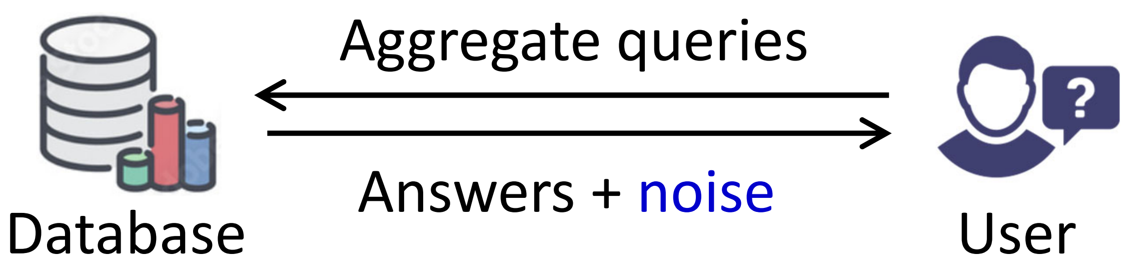
Statistical Database with Differential Privacy

- PINQ [SIGMOD 2009]
- wPINQ [VLDB 2014]
- FLEX [VLDB 2018]
- APEx [SIGMOD 2019]
- PrivateSQL [VLDB 2019]
- Chorus [EuroS&P 2020]
- …



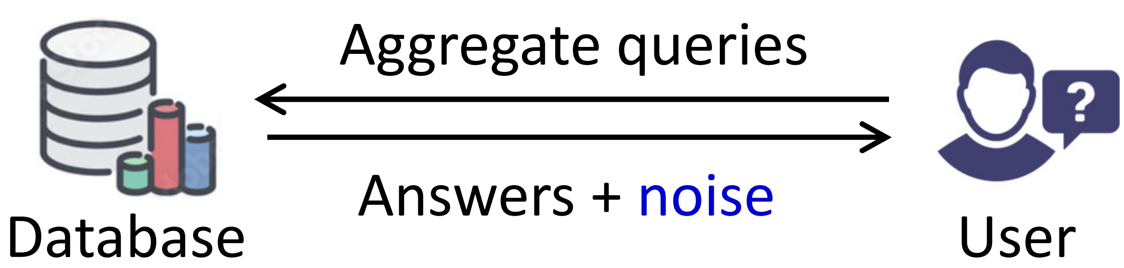
Statistical Database with Differential Privacy

- Basic idea:
  - Choose a total privacy budget ε_{tot}
  - For each query Q_i, compute the privacy budget ε_i consumed in the noisy answer
  - Stop when $\sum_i \varepsilon_i > \varepsilon_{tot}$
- Advantage: Strong privacy protection against attacks



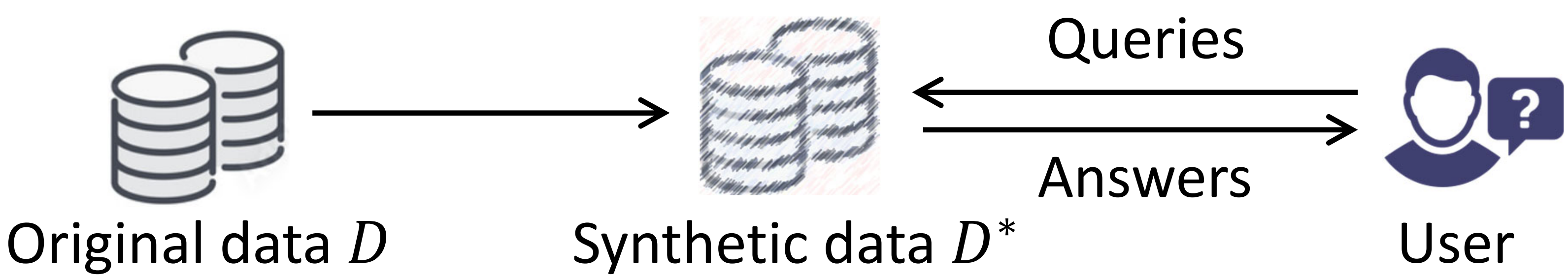
Statistical Database with Differential Privacy

- Common problem: the statistical database becomes unusable after the privacy budget is depleted
- To avoid this, we consider a different route: synthetic data

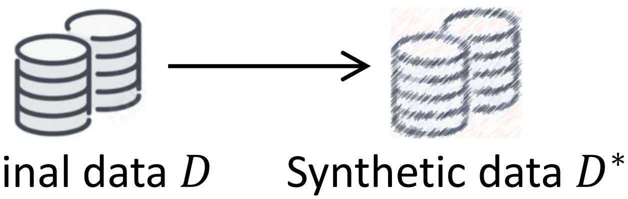


Synthetic Data with DP

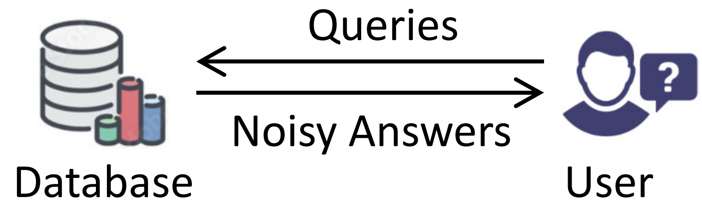
- Basic idea
  - Given the original dataset D, generate a synthetic dataset D^* that mimics D
  - Use D^* to answer queries
- Rationale
  - As long as D^* is generated with differential privacy, the query answers from D^* are "safe"



Synthetic Data vs. Noisy Answers



- Unlimited queries supported
- No change needed to the DBMS
- No additional query cost
- But no accuracy guarantee



- Limit on number of queries
- Considerable changes to the DBMS
- Additional computation cost per query
- Gives accuracy guarantees

Fine dining

Roadmap

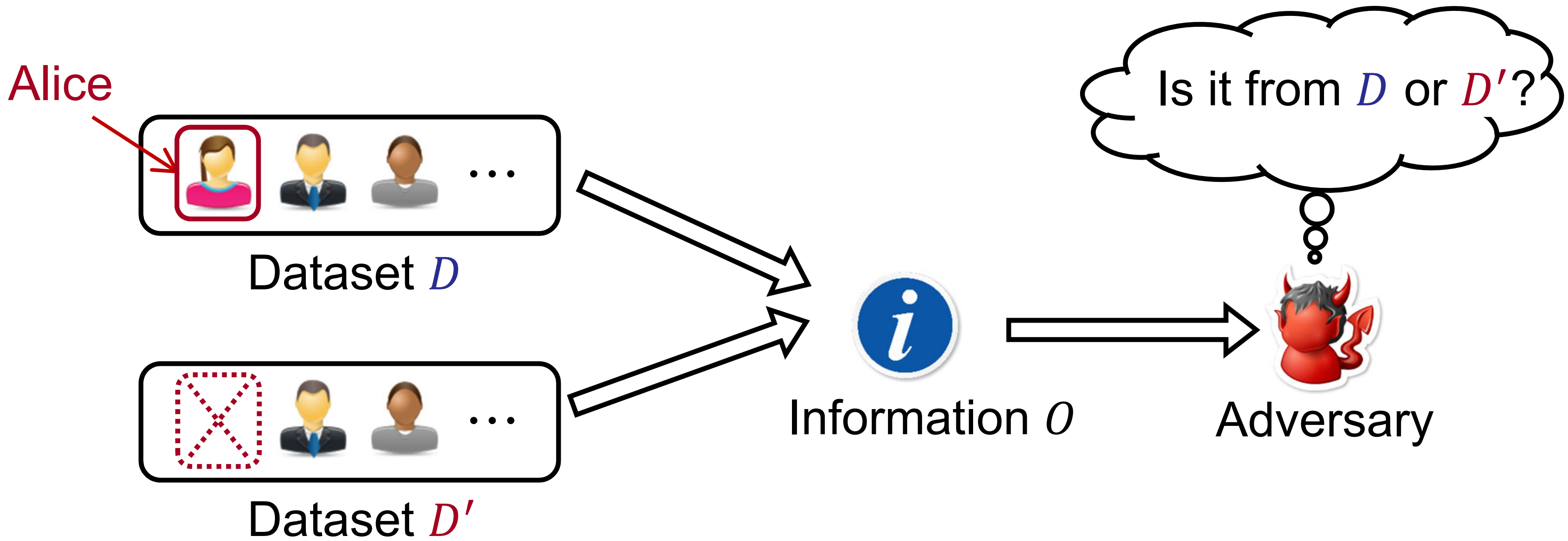
- Differential privacy (DP)
- Synthesizing relational data with DP
- Conclusion

Differential Privacy

- A notion of privacy proposed by theoreticians in 2006
  - Becomes popular over the years
  - Now adopted by Apple, US Census, etc.
- Its formulation borrows ideas from cryptography
  - Models privacy protection as a *game*

Dwork et al. Calibrating Noise to Sensitivity in Private Data Analysis. TCC 2006.

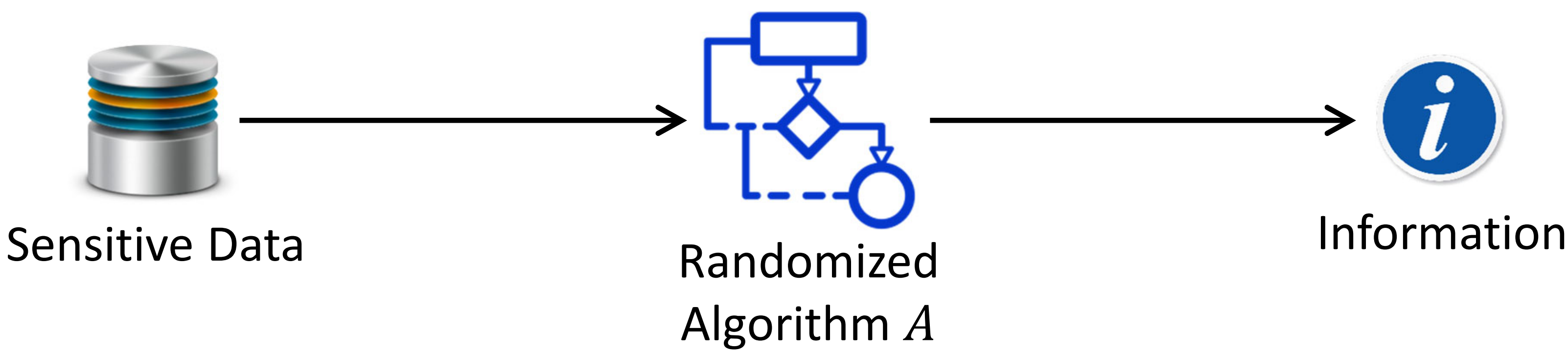
Differential Privacy: Rationale



- D' = D with Alice's information removed
- Intuition: If the adversary is unable to tell whether O is computed from D or D', then Alice's privacy is preserved

Differential Privacy: Details

- Differential privacy requires that any information to be shared should be generated using a *randomized algorithm* A

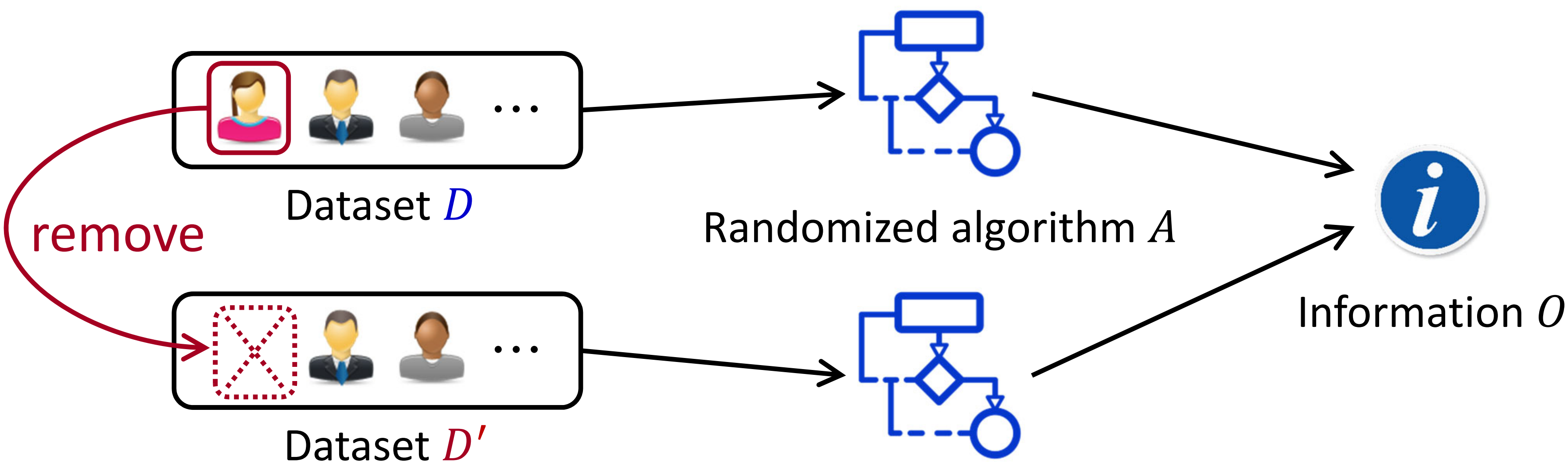


Differential Privacy: Details

- A randomized algorithm A satisfies ε-*differential privacy*, iff

$$\exp(-\varepsilon) \leq \frac{\Pr[A(D) = O]}{\Pr[A(D') = O]} \leq \exp(\varepsilon)$$

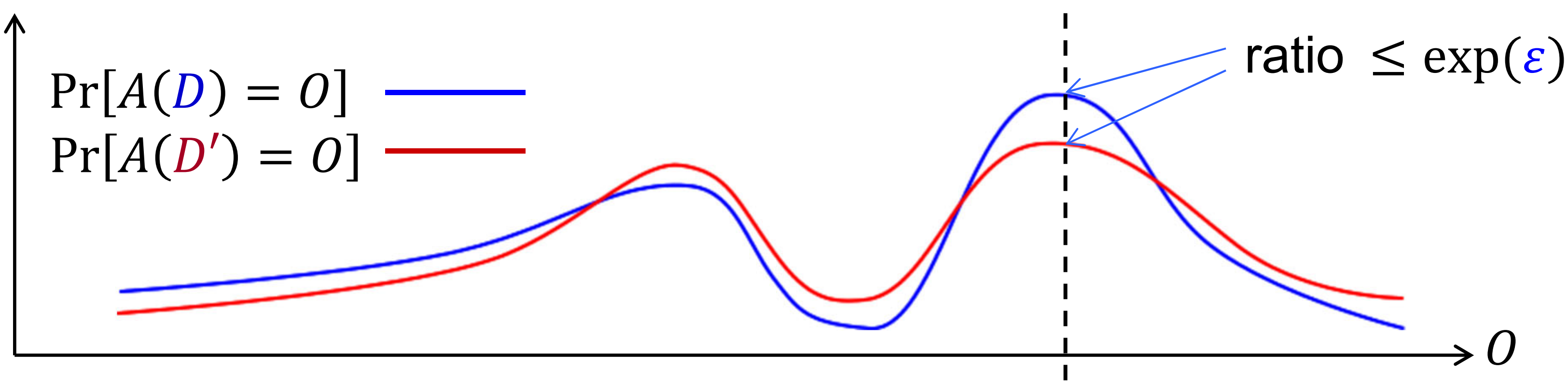
  for any two *neighboring* datasets D and D' and any output O of A



Differential Privacy: Illustration of Definition

$$\exp(-\varepsilon) \leq \frac{\Pr[A(D) = O]}{\Pr[A(D') = O]} \leq \exp(\varepsilon)$$

for any two *neighboring* datasets D and D' and any output O of A



Differential Privacy: Mechanisms

$$\exp(-\varepsilon) \leq \frac{\Pr[A(D) = O]}{\Pr[A(D') = O]} \leq \exp(\varepsilon)$$

- How can we achieve differential privacy?
- A canonical approach:
  - Take a non-private algorithm
  - Randomize it by injecting noise
- The amount and distribution of noise need to be carefully chosen
  - Details omitted

Roadmap

- Differential privacy (DP)
- Synthesizing relational data with DP
  - Single table synthesis
  - Multi-table synthesis
- Conclusion

Synthetic One Table with DP

- Problem definition:
  - Given a table T, release a synthetic version T^* in a way that satisfies ϵ-differential privacy
- Straightforward solution:
  - Convert T to a set of counts
  - Add noise to the counts
  - Map the noisy counts back to a synthetic table

Age	Has Diabetes?
< 30	Yes
< 30	Yes
30–39	No
40–49	No
…	…
50–59	No
$\geq$ 50	Yes

Synthetic Data with DP

- Step 1: Convert the data to a frequency matrix *M*

Age	Has Diabetes?
< 30	Yes
< 30	Yes
30–39	No
40–49	No
…	…
50–59	No
≥ 50	Yes

Table *T*

Age	Has Diabetes? Yes	Has Diabetes? No
< 30	2	0
30–39	4	15
40–49	5	22
50–59	8	49
≥ 50	12	87

Frequency Matrix *M*

Synthetic Data with DP

- Step 1: Convert the data to a frequency matrix *M*
- Step 2: Add noise into *M*

	Has Diabetes?	
Age	Yes	No
< 30	2	0
30–39	4	15
40–49	5	22
50–59	8	49
≥ 50	12	87

Frequency Matrix *M*

Synthetic Data with DP

- Step 1: Convert the data to a frequency matrix *M*
- Step 2: Add noise into *M*
- Step 3: map *M* back to a synthetic table

	Has Diabetes?	
Age	Yes	No
< 30	$2 + x_0$	$0 + x_5$
30–39	$4 + x_1$	$15 + x_6$
40–49	$5 + x_2$	$22 + x_7$
50–59	$8 + x_3$	$49 + x_8$
≥ 50	$12 + x_4$	$87 + x_9$

Frequency Matrix *M*

Synthetic Data with DP

- The good: simple and easy to implement
- The bad: it only works when M has a small number of entries
- But in practice, M could be large, especially when we have a sizable number d of attributes

Age	Has Diabetes? Yes	No
< 30	$2 + x_0$	$0 + x_5$
30–39	$4 + x_1$	$15 + x_6$
40–49	$5 + x_2$	$22 + x_7$
50–59	$8 + x_3$	$49 + x_8$
≥ 50	$12 + x_4$	$87 + x_9$

Frequency Matrix M

Synthetic Data with DP

- Suppose that we have n records, but M contains m cells with $m \gg n$
- The noise overwhelms the signal
  - We have m pieces of noise
  - But only $O(n)$ pieces of information
- This results in useless synthetic data

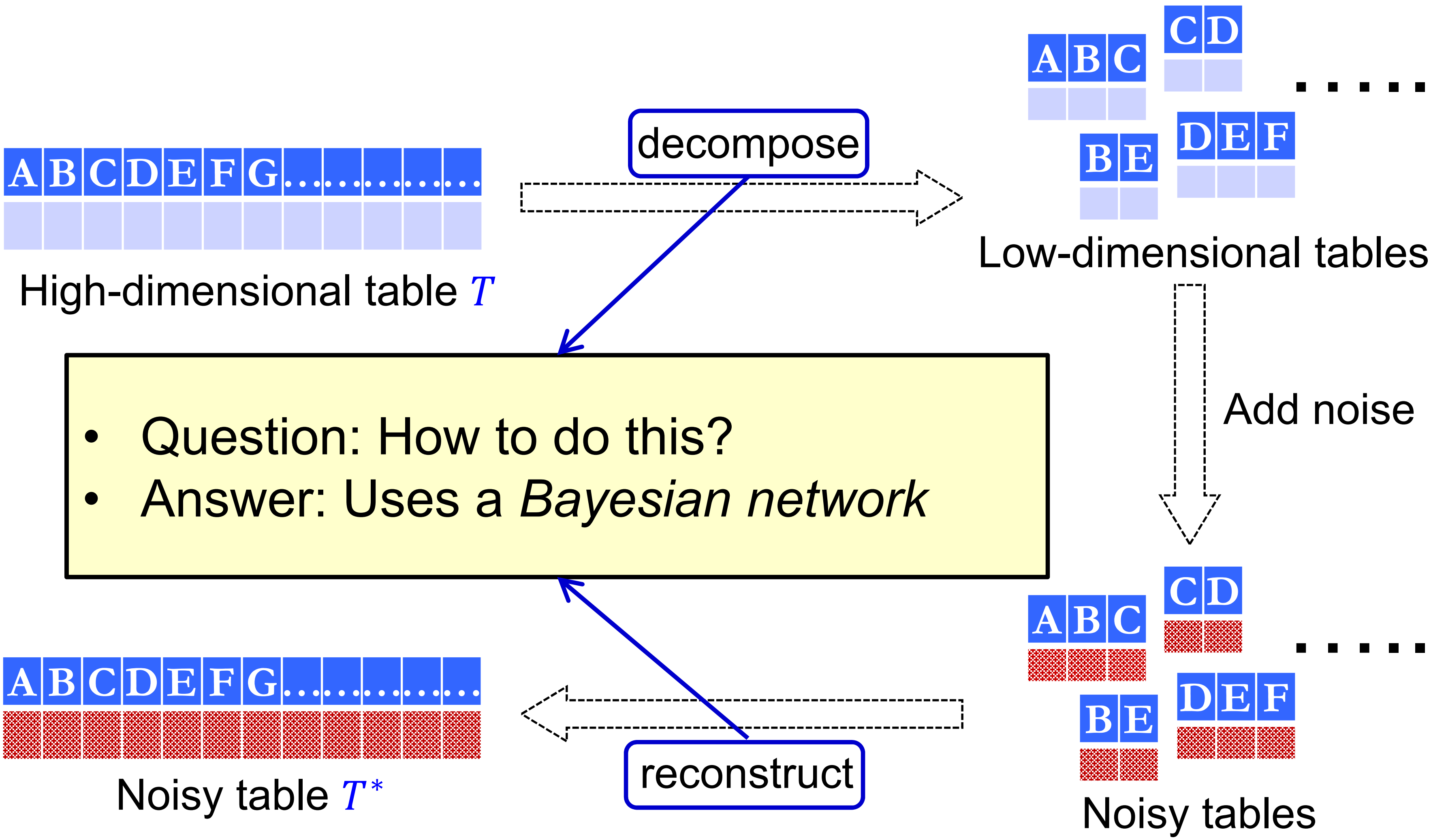
	Has Diabetes?	
Age	Yes	No
< 30	$2 + x_0$	$0 + x_5$
30–39	$4 + x_1$	$15 + x_6$
40–49	$5 + x_2$	$22 + x_7$
50–59	$8 + x_3$	$49 + x_8$
≥ 50	$12 + x_4$	$87 + x_9$

Frequency Matrix M

Towards a better solution

- Observation:
  - Attributes in datasets are often correlated
  - Even if a dataset has d dimensions, its *intrinsic dimensionality* could be much smaller than d
- Idea:
  - Exploit the correlations among attributes to mitigate the sparsity issue

Our Approach: PrivBayes



Zhang et al. PrivBayes: Private Data Release via Bayesian Networks. TODS 2017

Bayesian Network

- A graph that captures the correlations among the attributes
- Example: Table $T(A, B, C, D, E)$

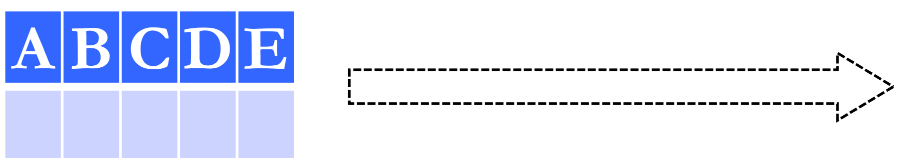
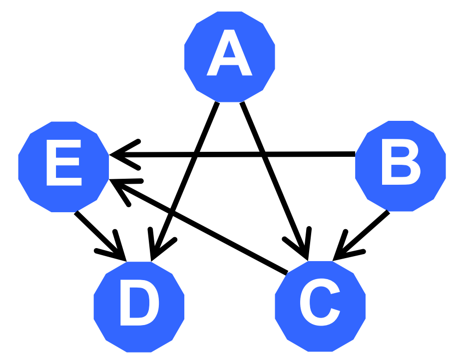


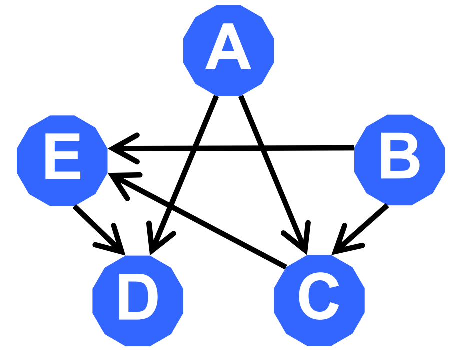
Table T



Bayesian network N

- Meaning:
  - $AB \dashrightarrow C; \quad BC \dashrightarrow E; \quad AE \dashrightarrow D$
- Decomposition:
  - $T_1(A, B, C), T_2(B, C, E), T_3(A, E, D)$

Bayesian Network



- Noisy tables:
  - $T_1^*(A, B, C), T_2^*(B, C, E), T_3^*(A, E, D)$
- Generation of synthetic tuple $t(a, b, c, d, e)$
  - Sample a, b, c based on $T_1^*(A, B, C)$
    - Result: $t(a, b, c, -, -)$
  - Sample e based on $T_2^*(B, C, E)$ and (b, c)
    - Result: $t(a, b, c, -, e)$
  - Sample d based on $T_3^*(A, E, D)$ and (a, e)
    - Result: $t(a, b, c, d, e)$

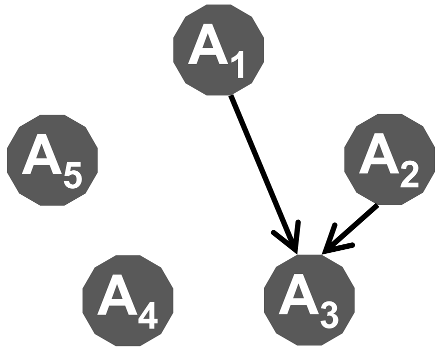
Bayesian Network with DP

- We need a way to construct the Bayesian network with differential privacy
  - Prior solutions were not designed with differential privacy in mind
- We devise our own solution based on a classic approach by Chow and Liu, with noise injected to achieve differential privacy

Chow and Liu. Approximating discrete probability distributions with dependence trees. TIT 1968

Variant of the Chow-Liu Approach

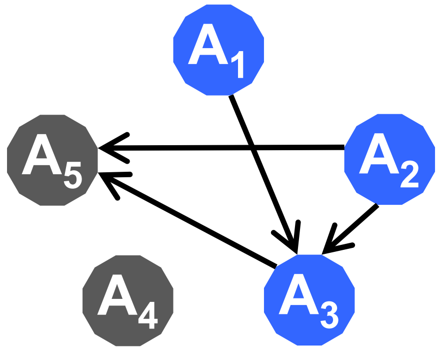
$k = 2$



- Input: d attributes $A_1, A_2, \dots, A_d$, a positive integer k
- Step 1: Initialize an empty Bayesian network N
- Step 2: Consider all possible $(k+1)$-attribute combinations $A_{i1}, A_{i2}, \dots, A_{ik}, A_j$, and evaluate $A_{i1}, A_{i2}, \dots, A_{ik} \dashrightarrow A_j$
  - Choose the combination that maximize the mutual information between $A_{i1} \times \cdots \times A_{ik}$ and A_j
- Step 3: Add the chosen $A_{i1}, A_{i2}, \dots, A_{ik} \dashrightarrow A_j$ into N

Variant of the Chow-Liu Approach

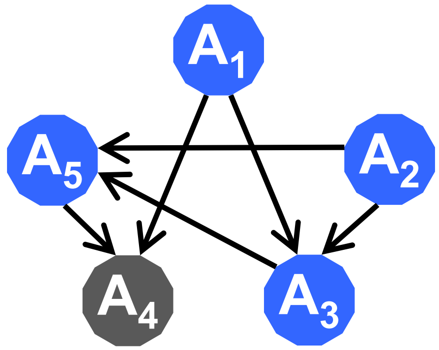
$k = 2$



- Input: d attributes $A_1, A_2, \ldots, A_d$, a positive integer k
- Step 1: Initialize an empty Bayesian network N
- Step 2: Consider all possible $(k+1)$-attribute combinations $A_{i1}, A_{i2}, \ldots, A_{ik}, A_j$, and evaluate $A_{i1}, A_{i2}, \ldots, A_{ik} \dashrightarrow A_j$
  - Choose the combination that maximize the mutual information between $A_{i1} \times \cdots \times A_{ik}$ and A_j
- Step 3: Add the chosen $A_{i1}, A_{i2}, \ldots, A_{ik} \dashrightarrow A_j$ into N
- Repeat Steps 2-3, but requiring $A_{i1}, A_{i2}, \ldots, A_{ik} \in N$ and $A_j \notin N$

Variant of the Chow-Liu Approach

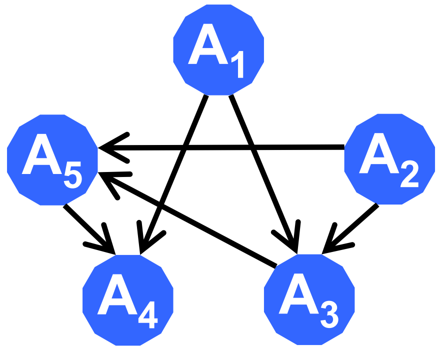
$k = 2$



- Input: d attributes $A_1, A_2, \dots, A_d$, a positive integer k
- Step 1: Initialize an empty Bayesian network N
- Step 2: Consider all possible $(k+1)$-attribute combinations $A_{i1}, A_{i2}, \dots, A_{ik}, A_j$, and evaluate $A_{i1}, A_{i2}, \dots, A_{ik} \dashrightarrow A_j$
  - Choose the combination that maximize the mutual information between $A_{i1} \times \cdots \times A_{ik}$ and A_j
- Step 3: Add the chosen $A_{i1}, A_{i2}, \dots, A_{ik} \dashrightarrow A_j$ into N
- Repeat Steps 2-3, but requiring $A_{i1}, A_{i2}, \dots, A_{ik} \in N$ and $A_j \notin N$

Variant of the Chow-Liu Approach

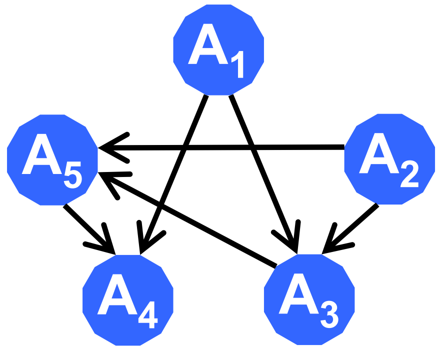
$k = 2$



- Input: d attributes $A_1, A_2, \ldots, A_d$, a positive integer k
- Step 1: Initialize an empty Bayesian network N
- Step 2: Consider all possible $(k+1)$-attribute combinations $A_{i1}, A_{i2}, \ldots, A_{ik}, A_j$, and evaluate $A_{i1}, A_{i2}, \ldots, A_{ik} \dashrightarrow A_j$
  - Choose the combination that maximize the mutual information between $A_{i1} \times \cdots \times A_{ik}$ and A_j
- Step 3: Add the chosen $A_{i1}, A_{i2}, \ldots, A_{ik} \dashrightarrow A_j$ into N
- Repeat Steps 2-3, but requiring $A_{i1}, A_{i2}, \ldots, A_{ik} \in N$ and $A_j \notin N$

Variant of the Chow-Liu Approach

$k = 2$

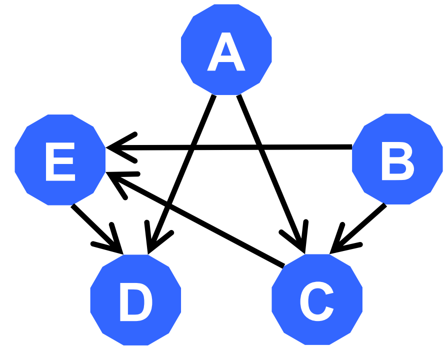


How to make it differentially private?

- Input: d attributes $A_1, A_2, \ldots, A_d$, a positive integer k
- Step 1: Initialize an empty Bayesian network N
- Step 2: Consider all possible $(k+1)$-attribute combinations $A_{i1}, A_{i2}, \ldots, A_{ik}, A_j$, and evaluate $A_{i1}, A_{i2}, \ldots, A_{ik} \dashrightarrow A_j$
  - Choose the combination that maximize the mutual information between $A_{i1} \times \cdots \times A_{ik}$ and A_j
  - Add noise into the mutual information before selecting the max
- Step 3: Add the chosen $A_{i1}, A_{i2}, \ldots, A_{ik} \dashrightarrow A_j$ into N
- Repeat Steps 2-3, but requiring $A_{i1}, A_{i2}, \ldots, A_{ik} \in N$ and $A_j \notin N$

Summary of PrivBayes

- Use a noisy version of the Chow-Liu approach to construct a Bayesian network N
- Obtain the low-dimensional tables corresponding to N
- Add noise into those tables
- Use them to generate synthetic data



Bayesian network N

$T_1(A, B, C)$, $T_2(B, C, E)$, $T_3(A, E, D)$

$T_1^*(A, B, C)$, $T_2^*(B, C, E)$, $T_3^*(A, E, D)$

T^*

Subsequent Improvement: PrivMRF

- Main idea: Use Markov random fields (MRF) instead of a Bayesian network
  - This provides more flexibility in terms of the choices of low-dimensional tables
- Result: much more accurate synthetic data
- It became the winning solution in the NIST 2020 Differential Privacy Temporal Map Challenge

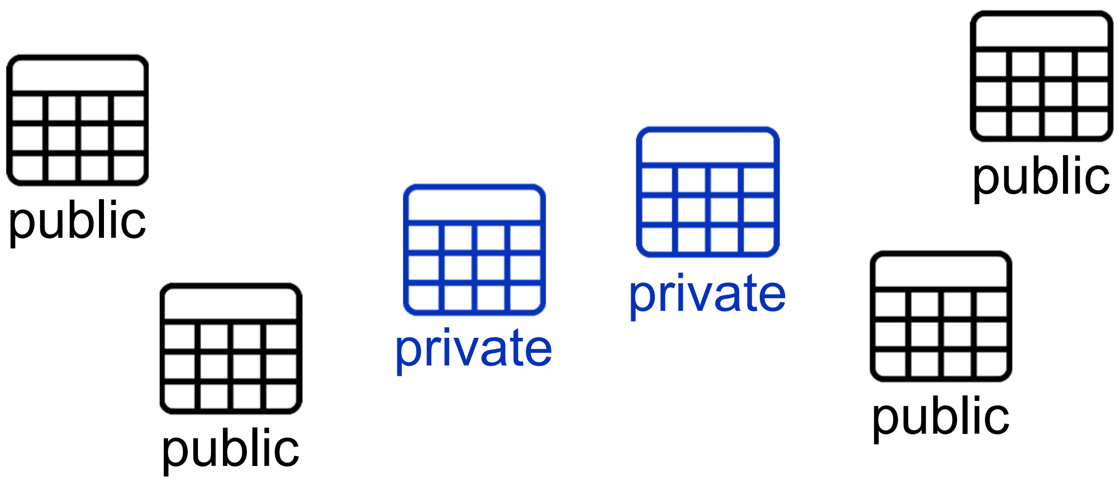
Cai et al. Data Synthesis via Differentially Private Markov Random Fields. PVLDB 2021

Roadmap

- Differential privacy (DP)
- Synthesizing relational data with DP
  - Single table synthesis: PrivBayes, PrivMRF
  - Multi-table synthesis
- Conclusion

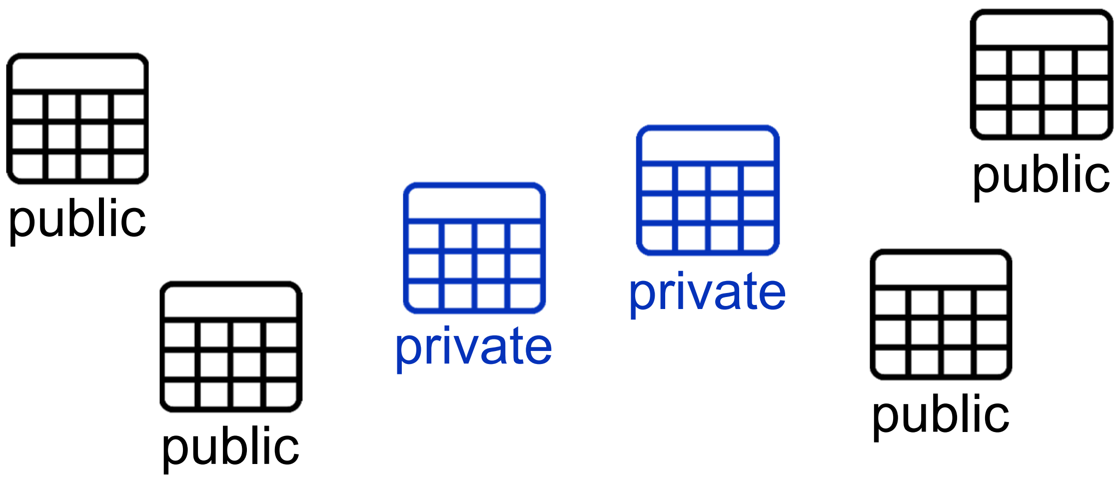
Multi-Table Synthesis

- Suppose that we have a database containing multiple tables
  - Some are private, some are public
- How can we synthesize the database?



Multi-Table Synthesis

- Straightforward solution:
  - Synthesize each private table separately (e.g., using PrivMRF)
- Problem:
  - It is unable to handle foreign keys



Multi-Table Synthesis

- Example from census data:
  - A table containing information about individuals
  - Another table containing household information

Age	Gender	…	H-ID
35	M	…	1
34	F	…	1
3	F	…	1
27	M	…	2
28	F	…	2
…	…	…	…

Individual Table T_I

H-ID	Ownership of Dwelling	…
1	Y	…
2	N	…
…	…	…

Household Table T_H

Multi-Table Synthesis

- If we synthesize these two tables separately:

Age	Gender	…	H-ID
35	M	…	1
34	F	…	1
3	F	…	1
27	M	…	2
28	F	…	2
…	…	…	…

Individual Table T_I

H-ID	Ownership of Dwelling	…
1	Y	…
2	N	…
…	…	…

Household Table T_H

Multi-Table Synthesis

- If we synthesize these two tables separately:
  - We have synthetic individuals, and synthetic households
  - How to assign individuals to households?

Age	Gender	…	H-ID
26	F	…	?
4	M	…	?
39	F	…	?
38	M	…	?
27	M	…	?
…	…	…	…

Individual Table T_I

H-ID	Ownership of Dwelling	…
1	N	…
2	Y	…
…	…	…

Household Table T_H

Multi-Table Synthesis

- What if we
  - Augment the household table with aggregate information of household members

Age	Gender	…	H-ID
26	F	…	?
4	M	…	?
39	F	…	?
38	M	…	?
27	M	…	?
…	…	…	…

Individual Table T_I

H-ID	Ownership of Dwelling	…
1	N	…
2	Y	…
…	…	…

Household Table T_H

Multi-Table Synthesis

- What if we
  - Augment the household table with aggregate information of household members
  - Synthesize the aggregate information, and use it to match individuals to households

Age	Gender	…	H-ID
26	F	…	?
4	M	…	?
39	F	…	?
38	M	…	?
27	M	…	?
…	…	…	…

Individual Table T_I

H-ID	Ownership of Dwelling	…	Size	Avg Age	…
1	N	…	…	…	…
2	Y	…	…	…	…
…	…	…	…	…	…

Household Table T_H

Multi-Table Synthesis

- Problem:
  - Too many augmented attributes needed
  - Matching individuals to household is non-trivial

Age	Gender	…	H-ID
26	F	…	?
4	M	…	?
39	F	…	?
38	M	…	?
27	M	…	?
…	…	…	…

Individual Table T_I

H-ID	Ownership of Dwelling	…	Size	Avg Age	…
1	N	…	…	…	…
2	Y	…	…	…	…
…	…	…	…	…	…

Household Table T_H

Multi-Table Synthesis

- Our idea:
  - Assume that there is some *latent variable* that decides the type of each household
  - Sample households and their members based on the latent variables

Age	Gender	…	H-ID
26	F	…	?
4	M	…	?
39	F	…	?
38	M	…	?
27	M	…	?
…	…	…	…

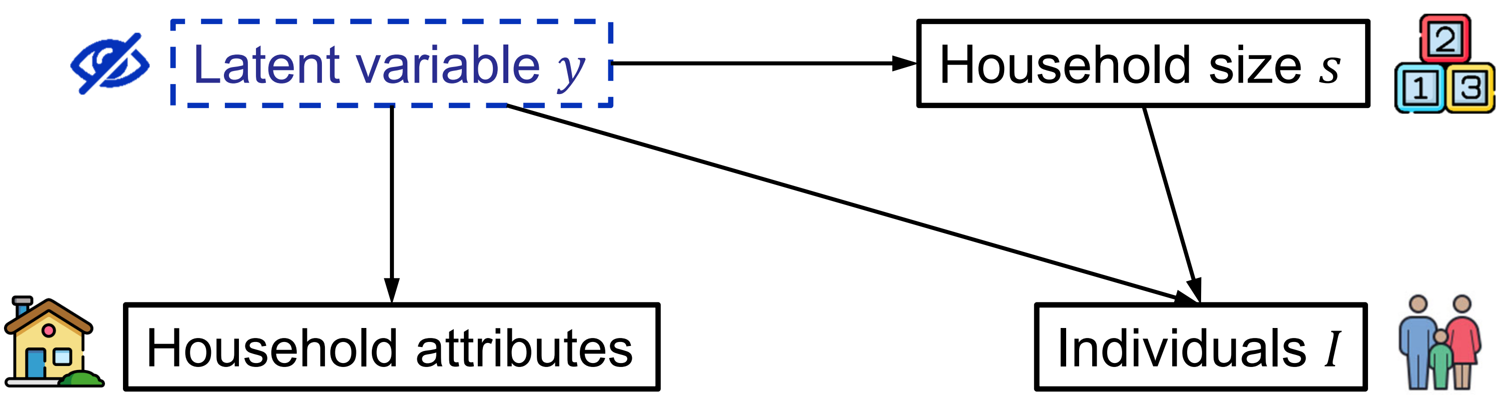
Individual Table T_I

H-ID	Ownership of Dwelling	…	Latent Variable	…
1	N	…	…	…
2	Y	…	…	…
…	…	…	…	…

Household Table T_H

Generative Process

- Sample the latent variable y
- Given y, sample the size s of the household and its attributes
- Given y and s, sample the attributes of s individuals

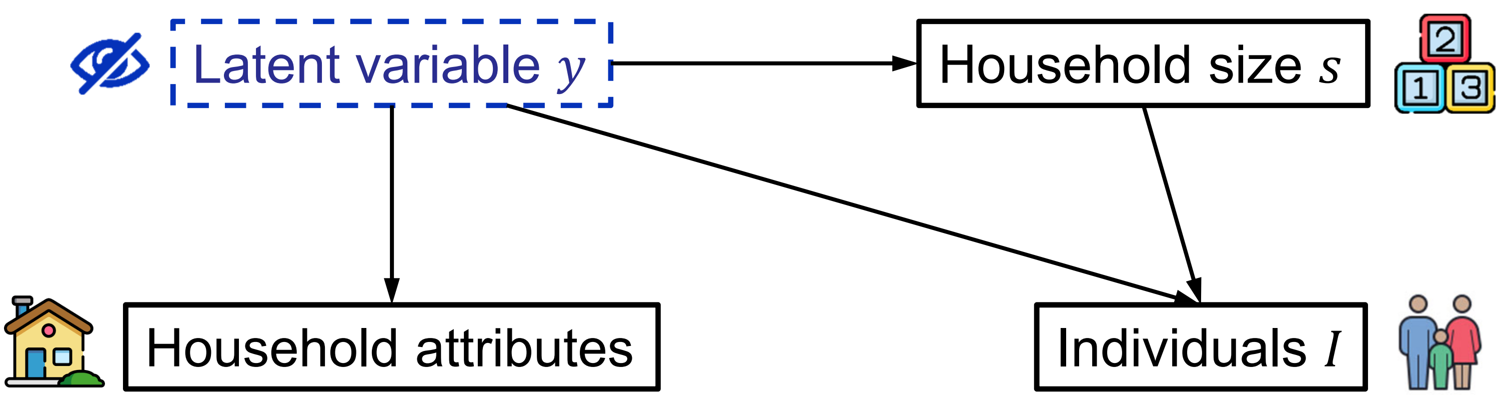


Model

- Likelihood of a household H with size s:

$$p(H) = \sum_{y \in Y} \left(p(y) \cdot p(s \mid y) \prod_{j=1}^{s} p(I_j \mid y) \right)$$

- Problem:
  - Given the observed households, estimate the distributions of y, s given y, and individuals given y



Model

- Likelihood of a household H with size s:

$$p(H) = \sum_{y \in Y} \left(p(y) \cdot p(s \mid y) \prod_{j=1}^{s} p(I_j \mid y) \right)$$

- Problem:
  - Given the observed households, estimate the distributions of y, s given y, and individuals given y
- Solution:
  - Use a graphical model with latent variables
  - Parameter estimation: use expectation maximization (EM)
    - With noise added to achieve differential privacy

Algorithm

- Given the two tables, we use EM + DP to obtain a model of individual + household type

Age	…
35	…
34	…
3	…
…	…

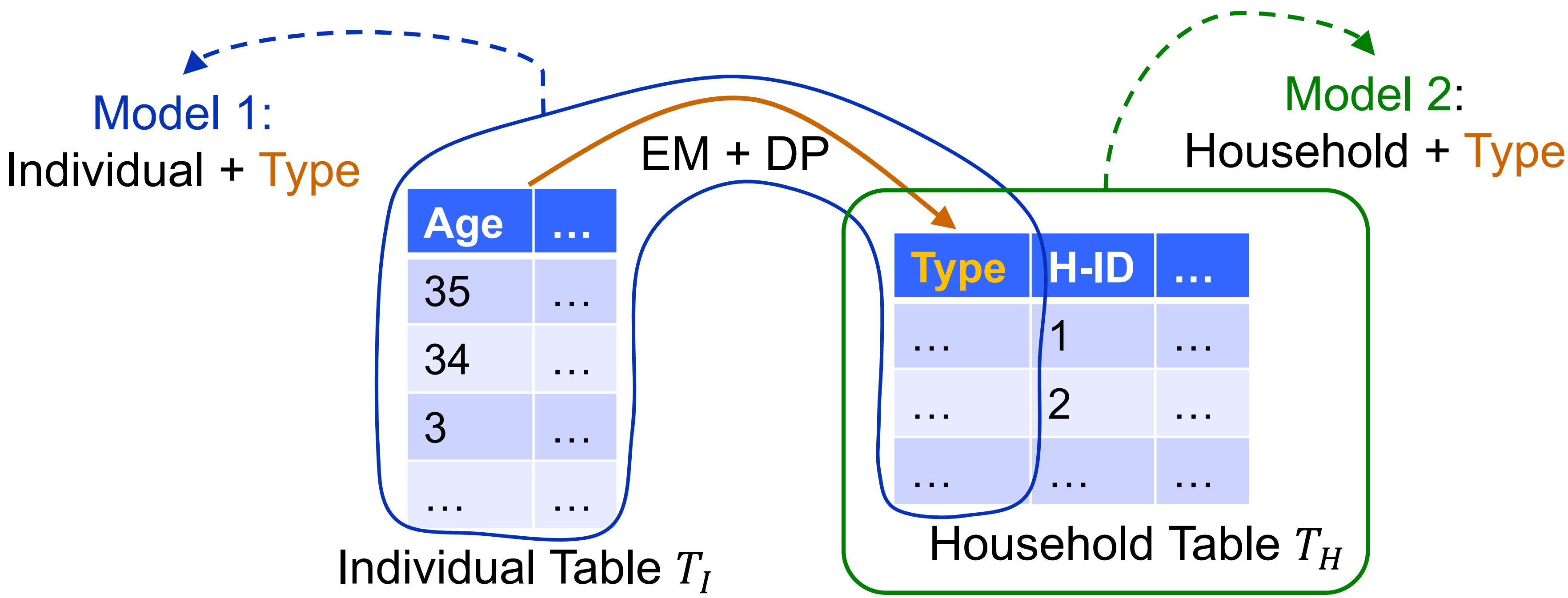
Individual Table T_I

H-ID	…
1	…
2	…
…	…

Household Table T_H

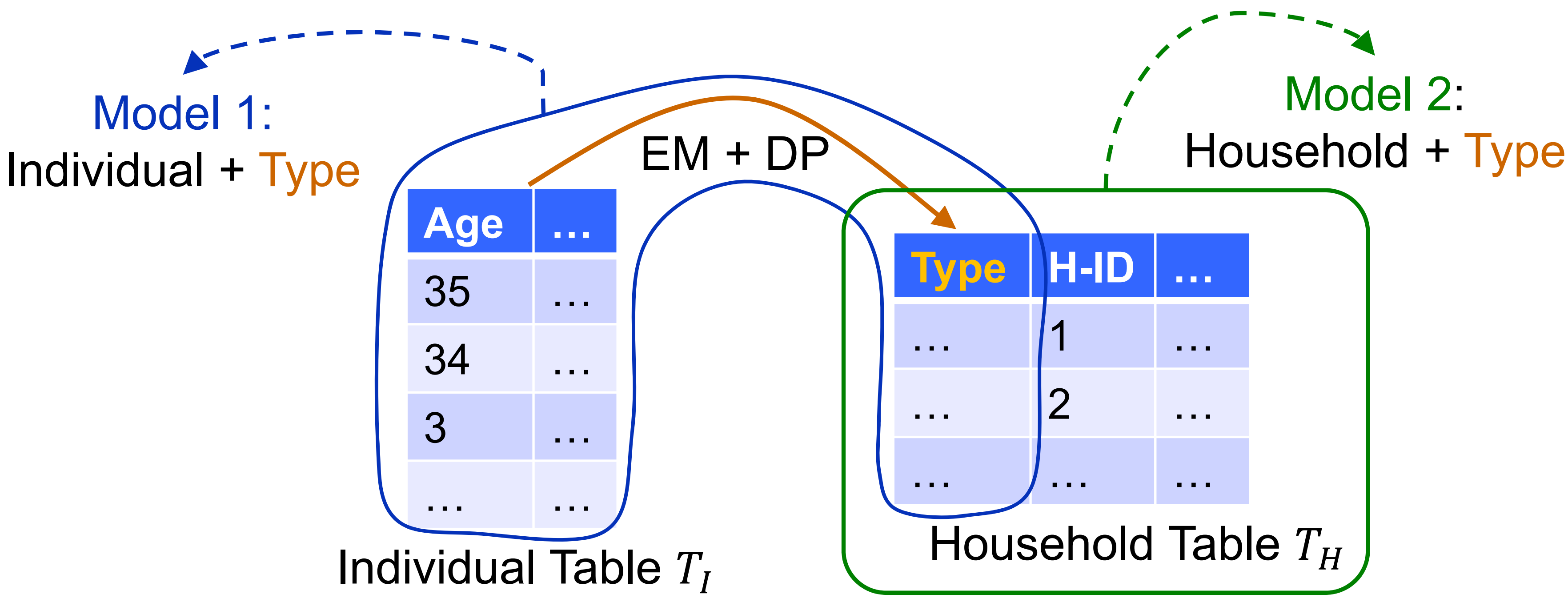
Algorithm

- Given the two tables, we use EM + DP to obtain a model of individual + household type
- And we use PrivMRF to obtain a model of household + household type



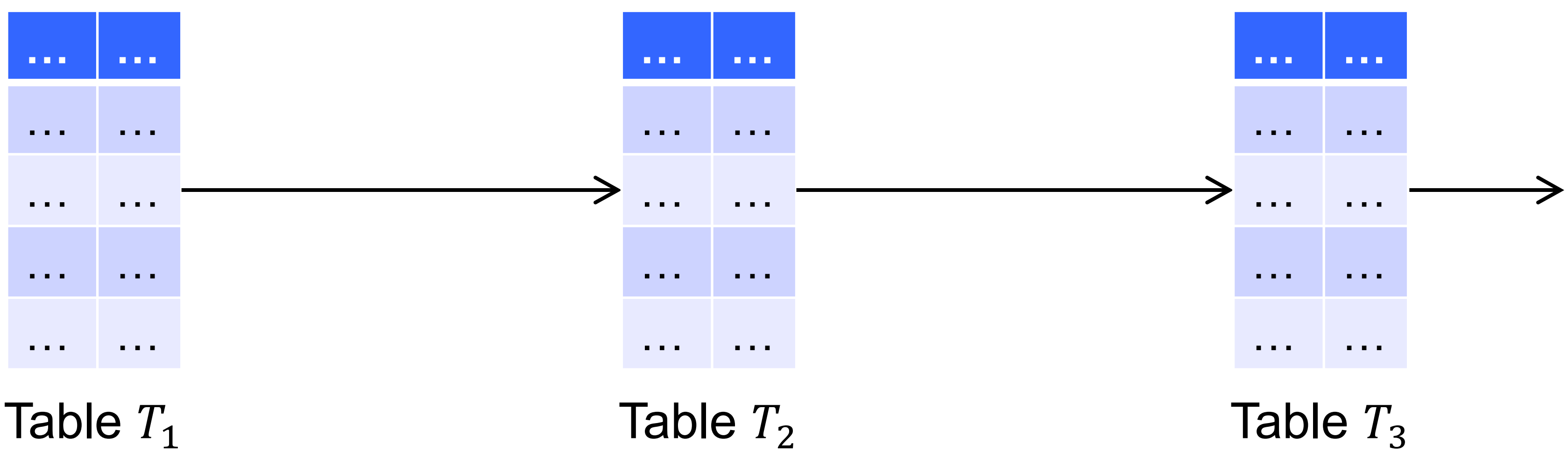
Algorithm

- This algorithm works for the case of two tables, and can be extended to more general cases



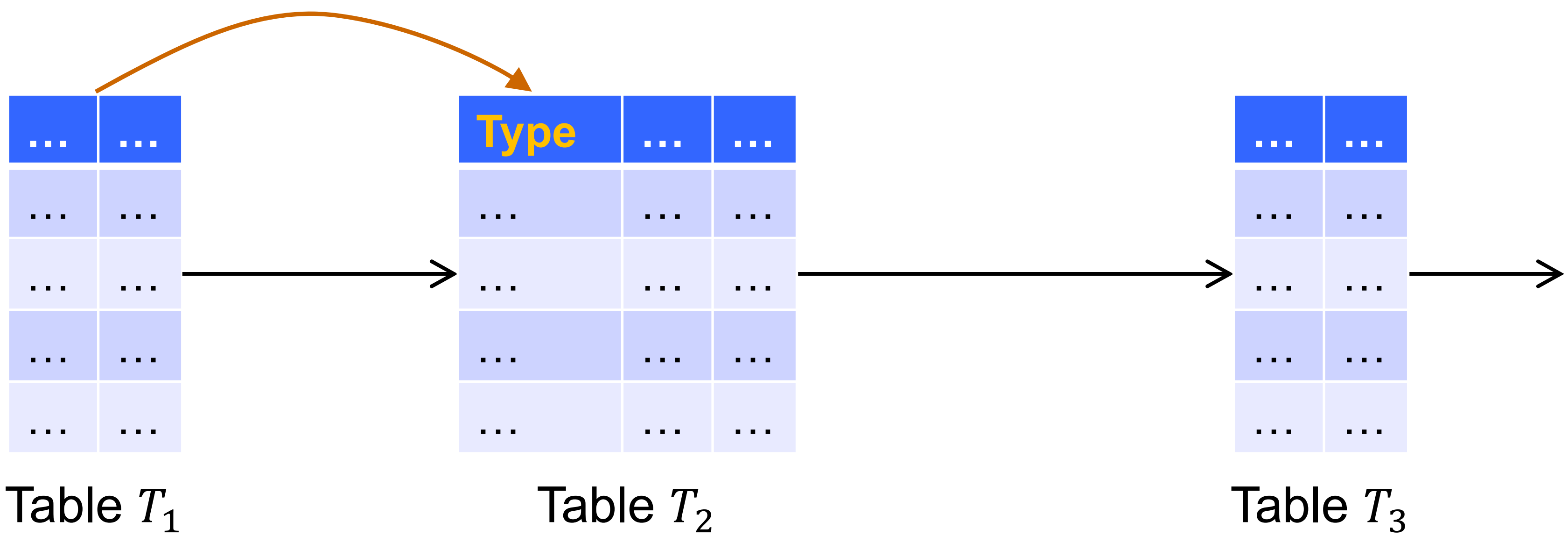
Extension: Foreign Key Chain

- For each foreign key, we consider latent variables in the table that it refers to
- We iteratively apply the two-table algorithm



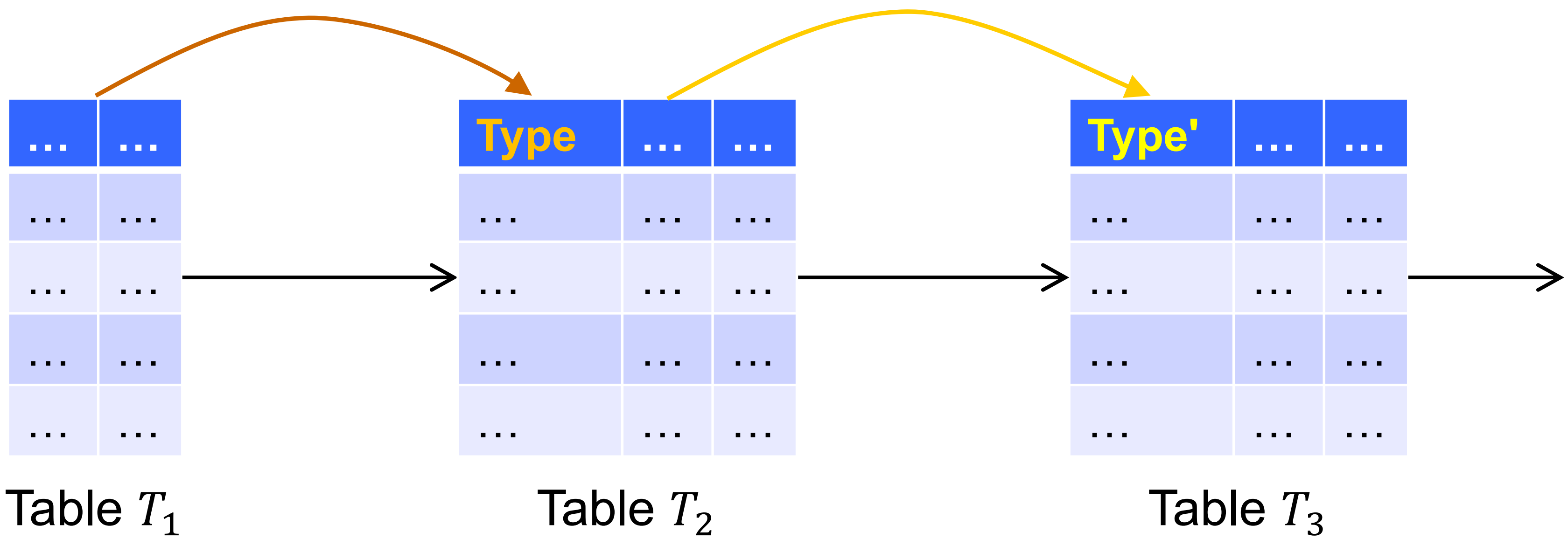
Extension: Foreign Key Chain

- For each foreign key, we consider latent variables in the table that it refers to
- We iteratively apply the two-table algorithm



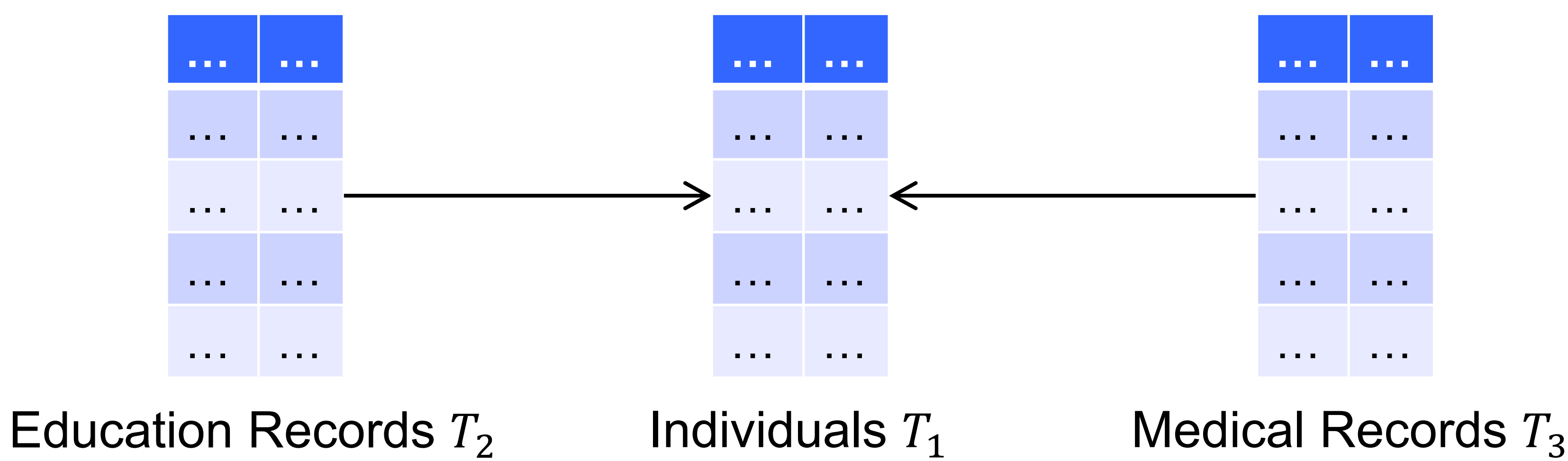
Extension: Foreign Key Chain

- For each foreign key, we consider latent variables in the table that it refers to
- We iteratively apply the two-table algorithm



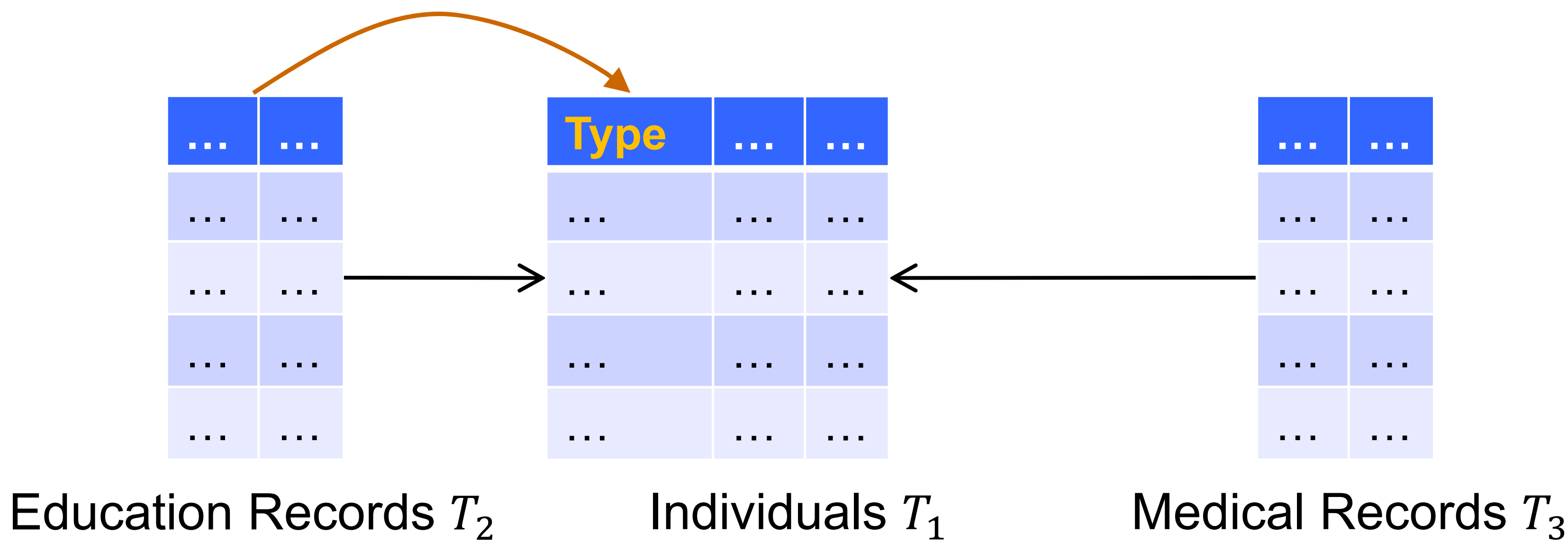
Extension: Reverse Star Schema

- For each foreign key, apply the two-table algorithm



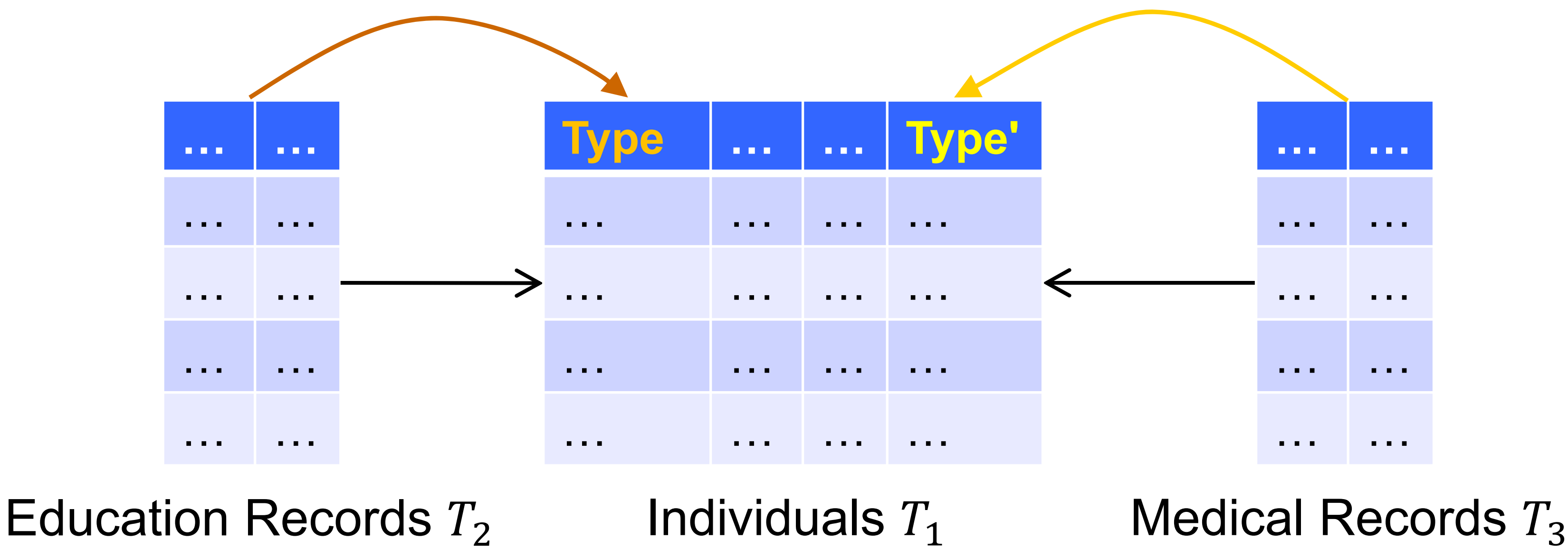
Extension: Reverse Star Schema

- For each foreign key, apply the two-table algorithm



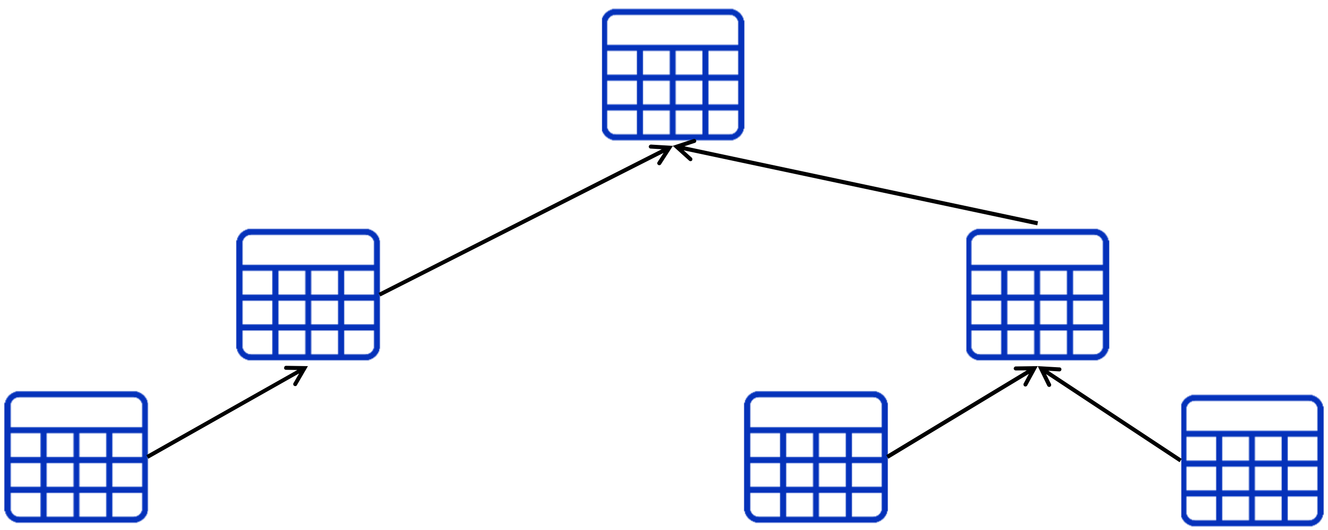
Extension: Reverse Star Schema

- For each foreign key, apply the two-table algorithm



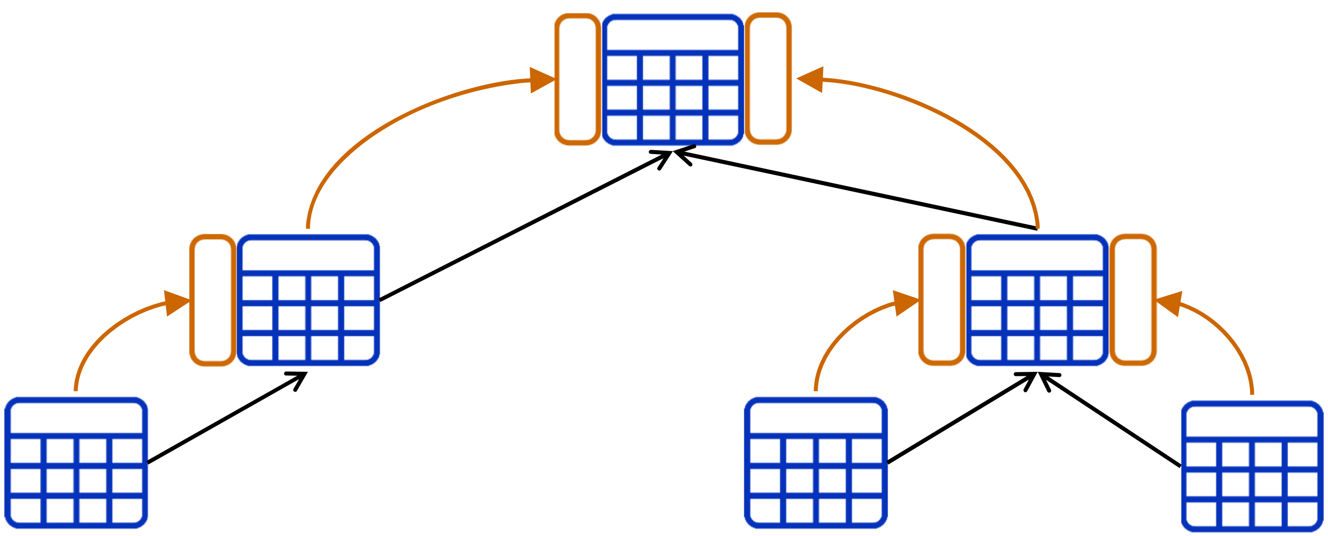
Extension: General Case

- In general, we can handle the case when
  - Each private table has at most one foreign key
  - There is no cycle in the key references



Extension: General Case

- Algorithm
  - Apply the two-table algorithm on each foreign key in a bottom up manner
  - Apply PrivMRF on the root(s)



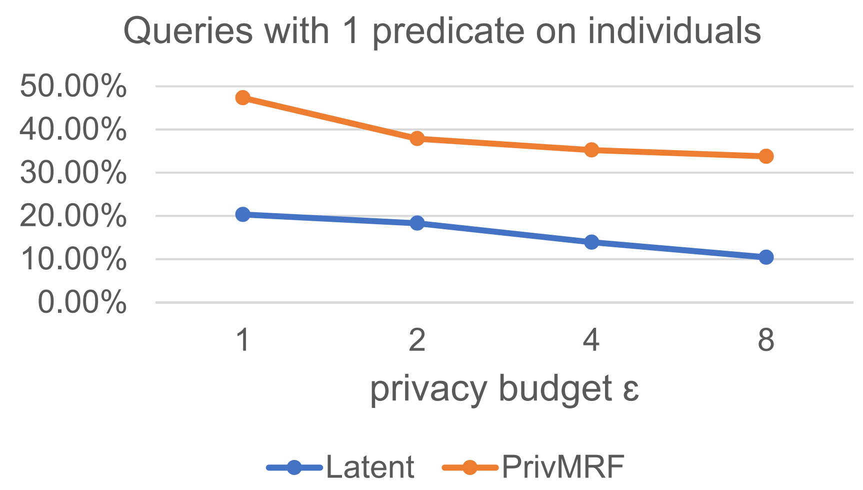
Experiments: Datasets

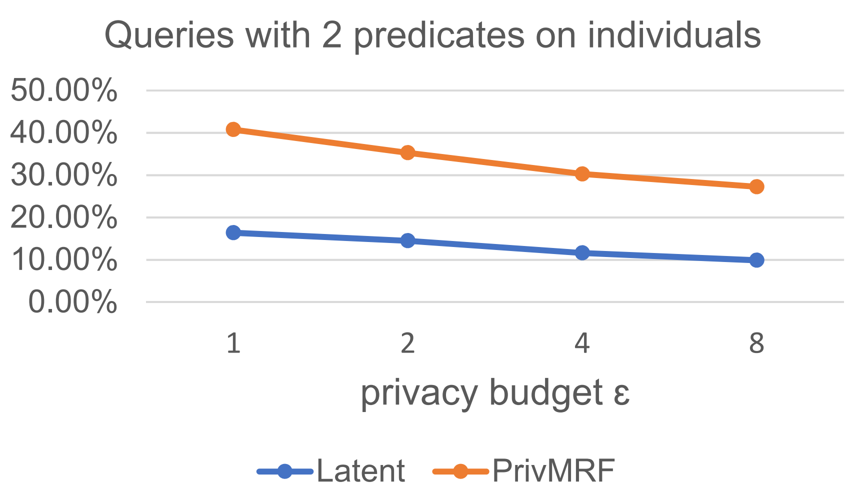
- Datasets: from the Integrated Public Use Microdata Series (www.ipums.org)

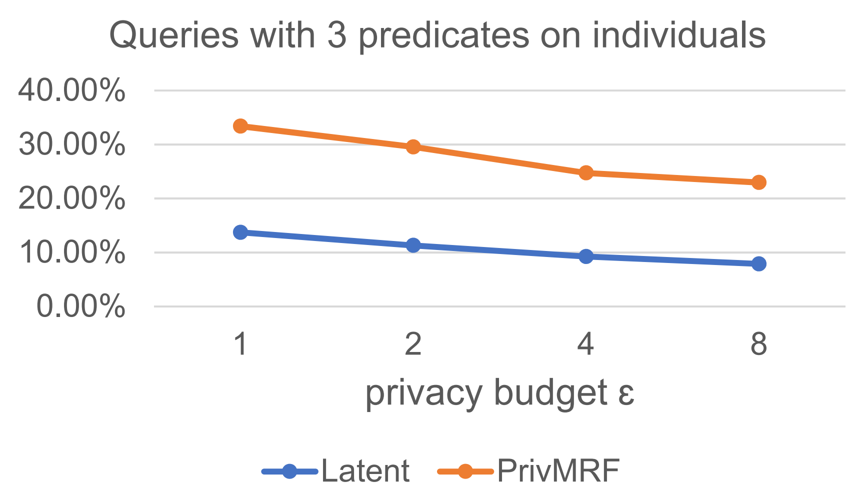
Dataset	# of Tuples	# of Attributes	Domain size
Person	561,046	16	$\approx 4.1 \times 10^{11}$
Household	251,364	9	$\approx 1.8 \times 10^{6}$

Experiments: Queries

- We consider count queries concerning both households and individuals
  - "How many households have annual income > x and at least one member with age > 30?"
- Query predicates are randomly generated:
  - 1 range predicate on a household attribute
  - *k* range predicates on individual attributes
- Error metric:
  - $$\frac{\text{absolute error of the query}}{\max\{\text{query result},\ 0.5\% \text{ of total population}\}}$$


Queries with 1 predicate on individuals
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
1
2
4
8
privacy budget ε
Latent
PrivMRF


Queries with 2 predicates on individuals
50.00%
40.00%
30.00%
20.00%
10.00%
0.00%
1
2
4
8
privacy budget ε
Latent
PrivMRF

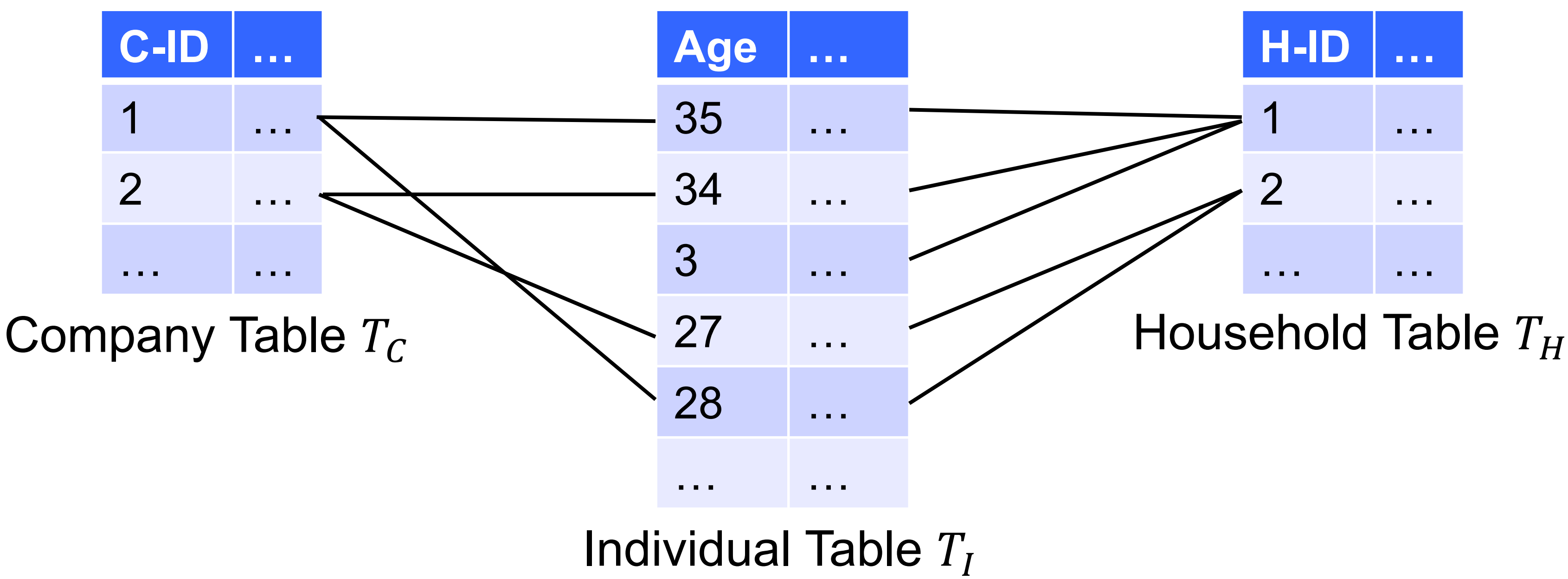

Queries with 3 predicates on individuals
40.00%
30.00%
20.00%
10.00%
0.00%
1
2
4
8
privacy budget ε
Latent
PrivMRF

Summary

- Synthetic relational data is a promising approach for statistical databases
  - Unlimited queries
  - No change to DBMS needed
- But handling foreign keys is a challenge
  - We have barely scratched the surface

Future Work

- Private tables with multiple foreign keys
- Main issue: Difficult to model the data



Future Work

- Private tables with self-relationships
- Main issue: how to capture the topology of the induced graph?

Age	...
35	...
34	...
3	...
27	...
28	...
...	...

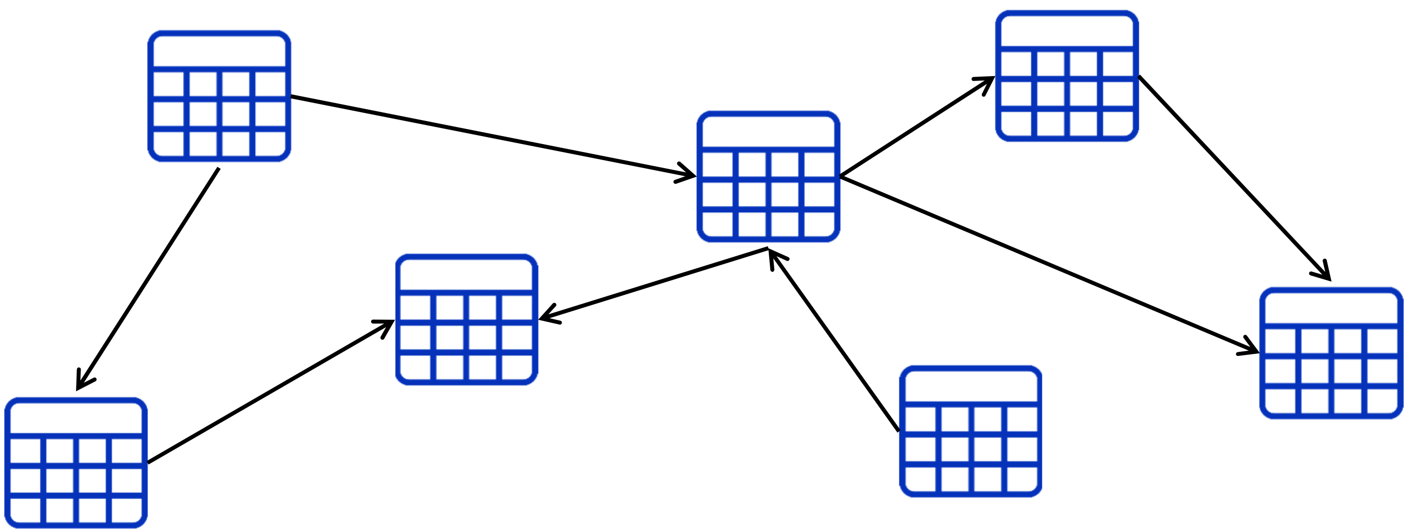
Individual Table T_I

ID1	ID2
...	...
...	...
...	...
...	...
...	...
...	...

Friends Table T_F

Future Work

- Arbitrary foreign keys



Future Work

- Beyond relational data
  - Time series
  - Trajectories
  - Transactions
  - …

Acknowledgement

- Graham Cormode, University of Warwick
- Kuntai Cai, NUS
- Xiaoyu Lei, U. of Connecticut
- Cecilia M. Procopiuc, Google
- Divesh Srivastava, AT&T Labs-Research
- Jianxin Wei, NUS
- Jun Zhang, Formerly NTU, Singapore