Schema Extraction

Divesh Srivastava (**AT&T Labs – Research)**

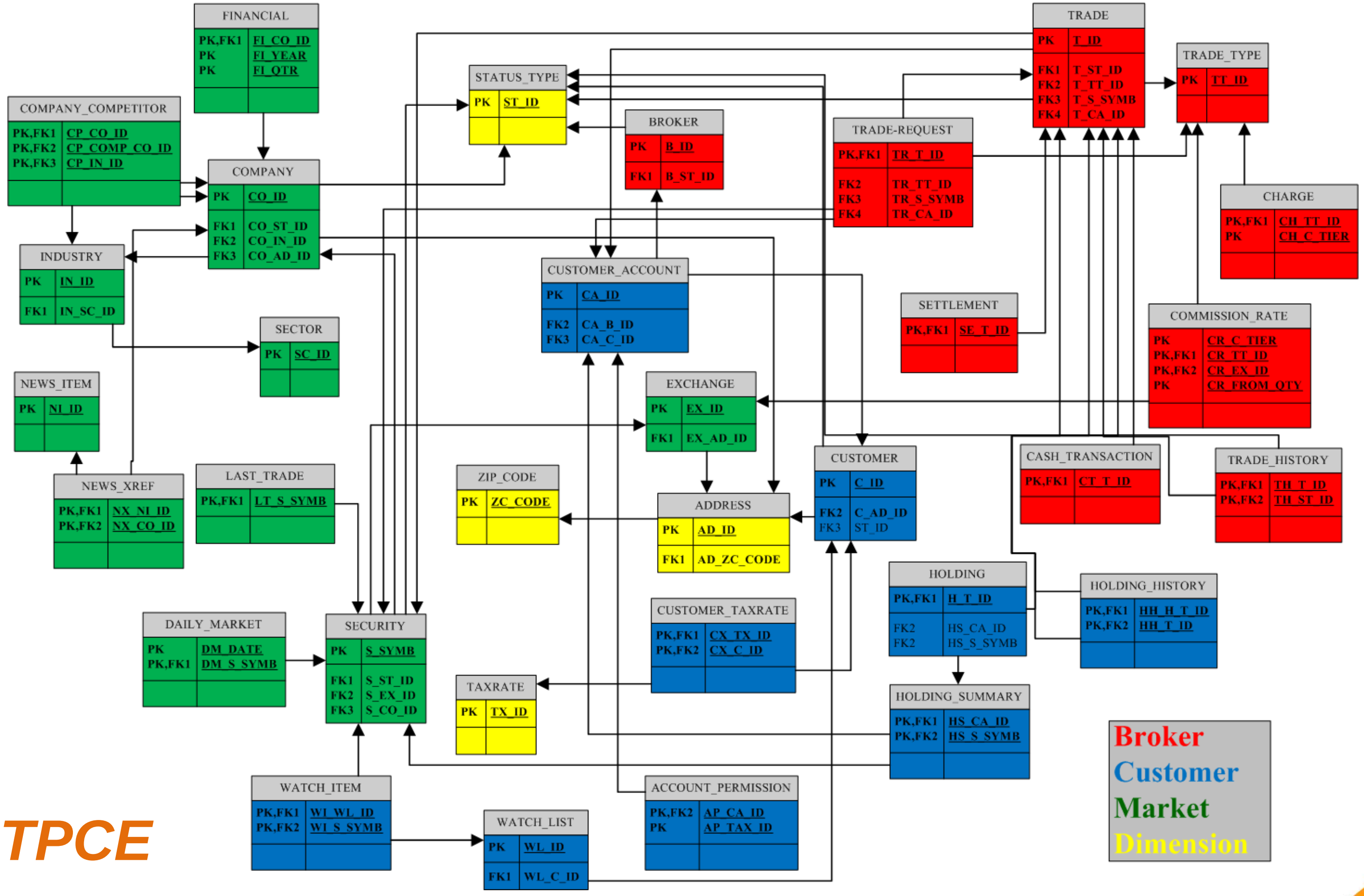
Joint work with Marios Hadjieleftheriou, Beng Chin Ooi, Cecilia M. Procopiuc, Xiaoyan Yang, Meihui Zhang

Motivation

- Information extraction
  - Extracting structure (e.g., tables) from unstructured data (e.g., text)

- Schema extraction
  - Extracting schema (σχήμα) from structured (e.g., tabular) data
  - Wealth of tabular data, e.g., spreadsheets, web tables, …
  - Schema includes keys, foreign keys, table spaces, …
  - Knowledge of database schema enables richer queries (e.g., joins), more sophisticated data analysis

Motivation: TPCE Schema Graph



FINANCIAL
PK,FK1 FI_CO_ID
PK FI_YEAR
PK FI_QTR

COMPANY_COMPETITOR
PK,FK1 CP_CO_ID
PK,FK2 CP_COMP_CO_ID
PK,FK3 CP_IN_ID

COMPANY
PK CO_ID
FK1 CO_ST_ID
FK2 CO_IN_ID
FK3 CO_AD_ID

INDUSTRY
PK IN_ID
FK1 IN_SC_ID

SECTOR
PK SC_ID

NEWS_ITEM
PK NI_ID

NEWS_XREF
PK,FK1 NX_NI_ID
PK,FK2 NX_CO_ID

LAST_TRADE
PK,FK1 LT_S_SYMB

DAILY_MARKET
PK DM_DATE
PK,FK1 DM_S_SYMB

SECURITY
PK S_SYMB
FK1 S_ST_ID
FK2 S_EX_ID
FK3 S_CO_ID

WATCH_ITEM
PK,FK1 WI_WL_ID
PK,FK2 WI_S_SYMB

WATCH_LIST
PK WL_ID
FK1 WL_C_ID

STATUS_TYPE
PK ST_ID

ZIP_CODE
PK ZC_CODE

TAXRATE
PK TX_ID

BROKER
PK B_ID
FK1 B_ST_ID

CUSTOMER_ACCOUNT
PK CA_ID
FK2 CA_B_ID
FK3 CA_C_ID

EXCHANGE
PK EX_ID
FK1 EX_AD_ID

ADDRESS
PK AD_ID
FK1 AD_ZC_CODE

CUSTOMER_TAXRATE
PK,FK1 CX_TX_ID
PK,FK2 CX_C_ID

ACCOUNT_PERMISSION
PK,FK2 AP_CA_ID
PK AP_TAX_ID

TRADE
PK T_ID
FK1 T_ST_ID
FK2 T_TT_ID
FK3 T_S_SYMB
FK4 T_CA_ID

TRADE_TYPE
PK TT_ID

TRADE-REQUEST
PK,FK1 TR_T_ID
FK2 TR_TT_ID
FK3 TR_S_SYMB
FK4 TR_CA_ID

CHARGE
PK,FK1 CH_TT_ID
PK CH_C_TIER

SETTLEMENT
PK,FK1 SE_T_ID

COMMISSION_RATE
PK CR_C_TIER
PK,FK1 CR_TT_ID
PK,FK2 CR_EX_ID
PK CR_FROM_QTY

CUSTOMER
PK C_ID
FK2 C_AD_ID
FK3 ST_ID

CASH_TRANSACTION
PK,FK1 CT_T_ID

TRADE_HISTORY
PK,FK1 TH_T_ID
PK,FK2 TH_ST_ID

HOLDING
PK,FK1 H_T_ID
FK2 HS_CA_ID
FK2 HS_S_SYMB

HOLDING_HISTORY
PK,FK1 HH_H_T_ID
PK,FK2 HH_T_ID

HOLDING_SUMMARY
PK,FK1 HS_CA_ID
PK,FK2 HS_S_SYMB

TPCE

Broker
Customer
Market
Dimension

Outline

- Motivation
  - Extracting schema from tabular data
- Discovering good foreign keys from tabular data
  - Schema graph = nodes (tables, attributes) + edges (foreign keys)
- Discovering good table spaces
  - Clustering tables by topic, identifying important tables

Discovering Foreign Keys: Motivation

- Foreign/primary key relationship is an important constraint in relational databases

- Knowing foreign keys is often a crucial step in understanding and analyzing the data

Discovering Foreign Keys: Motivation

- In practice, foreign keys are often NOT specified in the schema

- Reasons
  - Associations not known to DB designers but inherent in data
  - Implicit relationships across multiple databases
  - Data inconsistencies (data integration, database evolution, …)
  - …

Existing Work

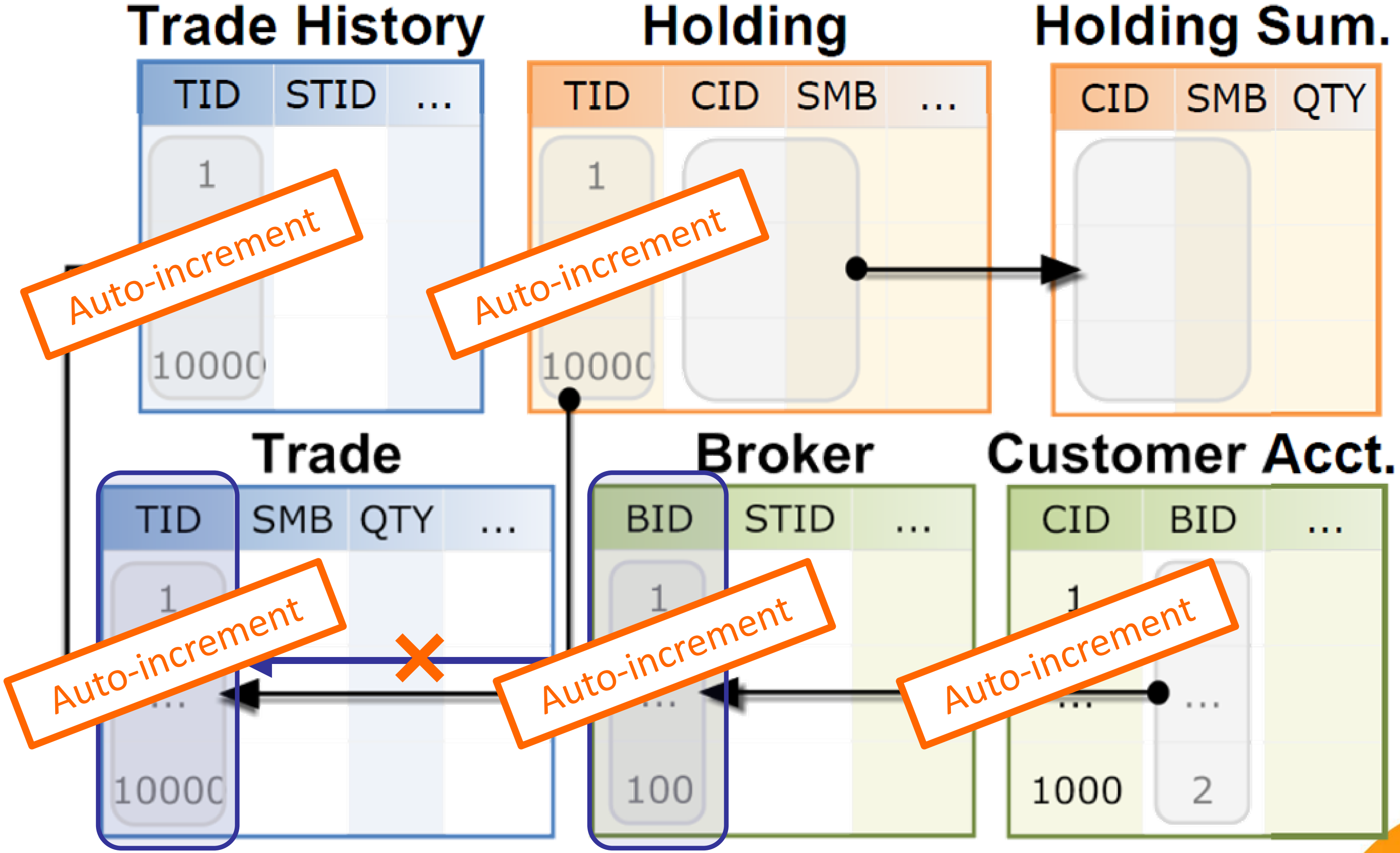
- ♦ Little previous work on discovering multi-column foreign keys

- ♦ Most focus mainly on identifying inclusion dependencies [1,2,3]
  - – The only formal requirement (a subset of primary key)
  - – Not enough

⟹ a large number of false positives

[1] J. Bauckmann, et al. Efficiently Detecting Inclusion Dependencies. ICDE 2007
[2] F. D. Marchi, et al. Unary and n-ary inclusion dependency discovery in relational databases.
[3] F. D. Marchi, et al. Zigzag: a new algorithm for mining large inclusion dependencies in database. ICDM 2003

Existing Work



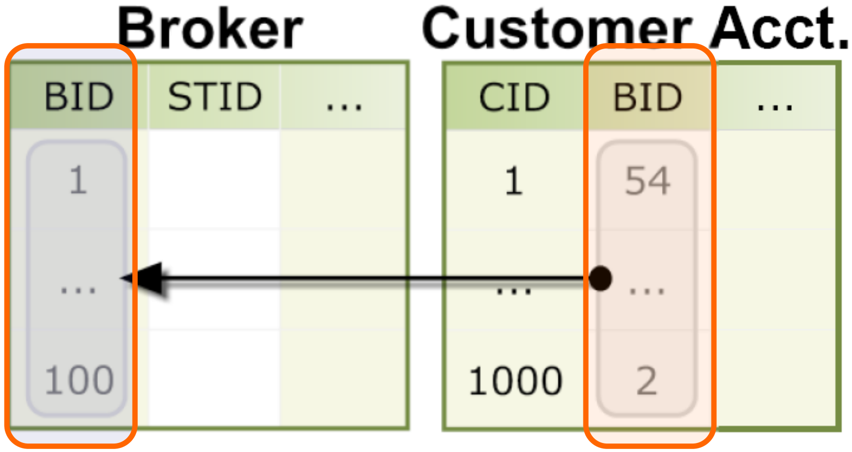
Existing Work

- Heuristic rules to reduce the number of false positives [4]
  - The column names of foreign/primary keys should be similar
  - A foreign key should have significant cardinality
  - A foreign key should have good coverage of the primary key
  - The primary key should have only a small percentage of values outside the range of the foreign key
  - The average length of the values in foreign/primary key columns should be similar (mostly for strings)
  - …
- Counter-examples exist for any rule!

[4] A. Rostin, et al. A Machine Learning Approach to Foreign Key Discovery. WebDB 2009

Our Approach

- Randomness
  - Measuring the likelihood that (F, P) is a useful FK/PK constraint
  - Thesis: values in F form a random sample of (ordered) values in P
  - No correlation between the semantics of the table with the foreign key and the way the primary keys are generated
  - In dynamic databases, the distributions change over time



Broker

BID	STID	…
1		
…		
100		

Customer Acct.

CID	BID	…
1	54	
…	…	
1000	2	

Outline

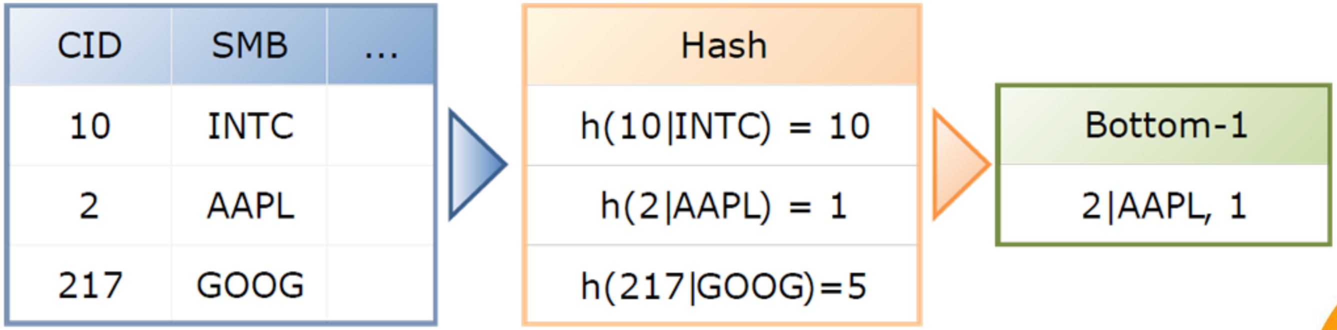
- Motivation
  - Extracting schema from tabular data
- Discovering good foreign keys from tabular data
  - Schema graph = nodes (tables, attributes) + edges (foreign keys)
  - Inclusion, randomness
- Discovering good table spaces
  - Clustering tables by topic, identifying important tables

Inclusion

- Partial inclusion $\sigma(F,P)$: user defined threshold
  - $\sigma(F,P) = |F \cap P|/|F|$
  - $\sigma(F,P) \geq 0.9$

- For efficiency
  - bottom-k sketch [5]

    [5] Edith Cohen, Haim Kaplan: Leveraging discarded samples for tighter estimation of multiple-set aggregates. SIGMETRICS/Performance 2009



CID	SMB	…
10	INTC	
2	AAPL	
217	GOOG	

Hash
h(10|INTC) = 10
h(2|AAPL) = 1
h(217|GOOG)=5

Bottom-1
2|AAPL, 1

Inclusion

- Partial inclusion $\sigma(F,P)$: user defined threshold
  - $\sigma(F,P) = |F \cap P| / |F|$
  - $\sigma(F,P) \geq 0.9$

- For efficiency
  - bottom-k sketch [5]

    [5] Edith Cohen, Haim Kaplan: Leveraging discarded samples for tighter estimation of multiple-set aggregates. SIGMETRICS/Performance 2009
  - SCS estimator [5]: Jaccard coefficient
  - $\sigma(F,P) = \mathrm{Jacc}(F,P)/\mathrm{Jacc}(F \cup P,F)$

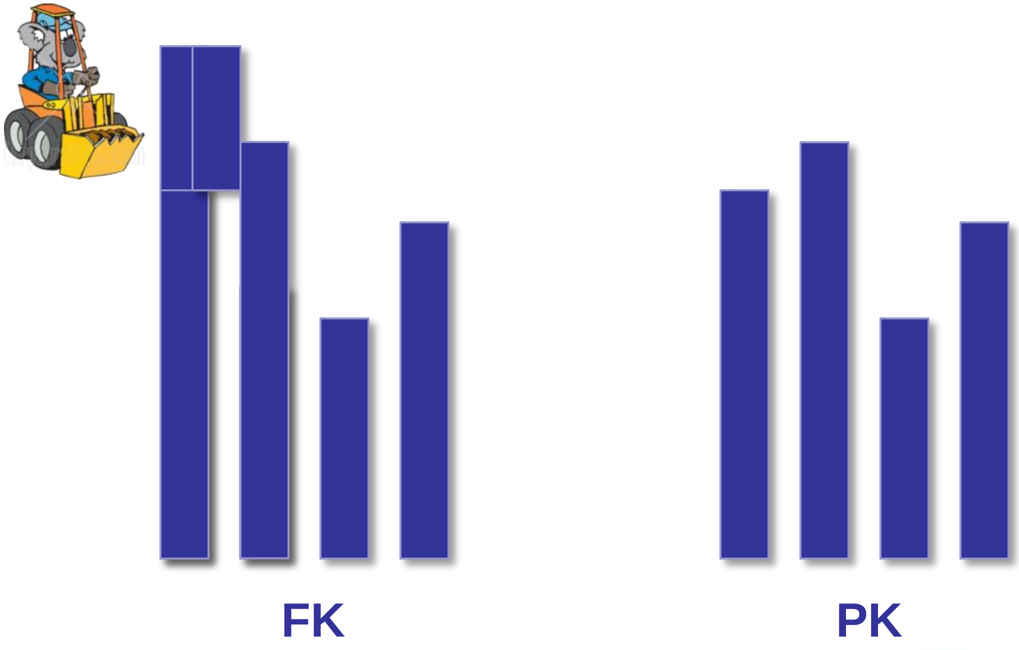
Randomness

- Randomness test
  - Given F and P, test if the distinct values (tuples) in F have the same underlying distribution as the values (tuples) in P

- Domain order
  - Numerical order: numeric values
  - Lexicographic order: strings

Randomness Measure

- Earth Mover’s Distance (EMD)
  - Standard distance measure between probability distributions
  - EMD measures the amount of work needed to convert 1st distribution into the 2nd



Distance Function

- Normalized distance between ranks
  - Independent of the actual values in any column

- Single-column
  - Absolute difference between the ranks in the underlying ordered space (PK column)
  - Normalize by the number of values

- Multi-column
  - Manhattan distance
  - Normalize by dimensionality

Probability Distribution

- Exact distribution
  - let each value in F (P) have a probability of 1/|F| (1/|P|)

Computing EMD is **too expensive** over large F (P)
(Hungarian algorithm has cubic complexity)

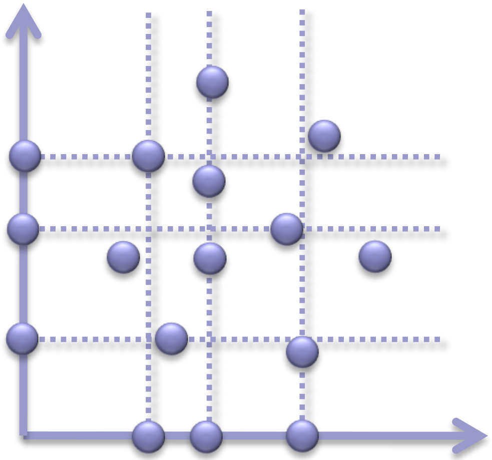
Probability Distribution

- Quantile histogram
  - ℓ-quantiles of PK

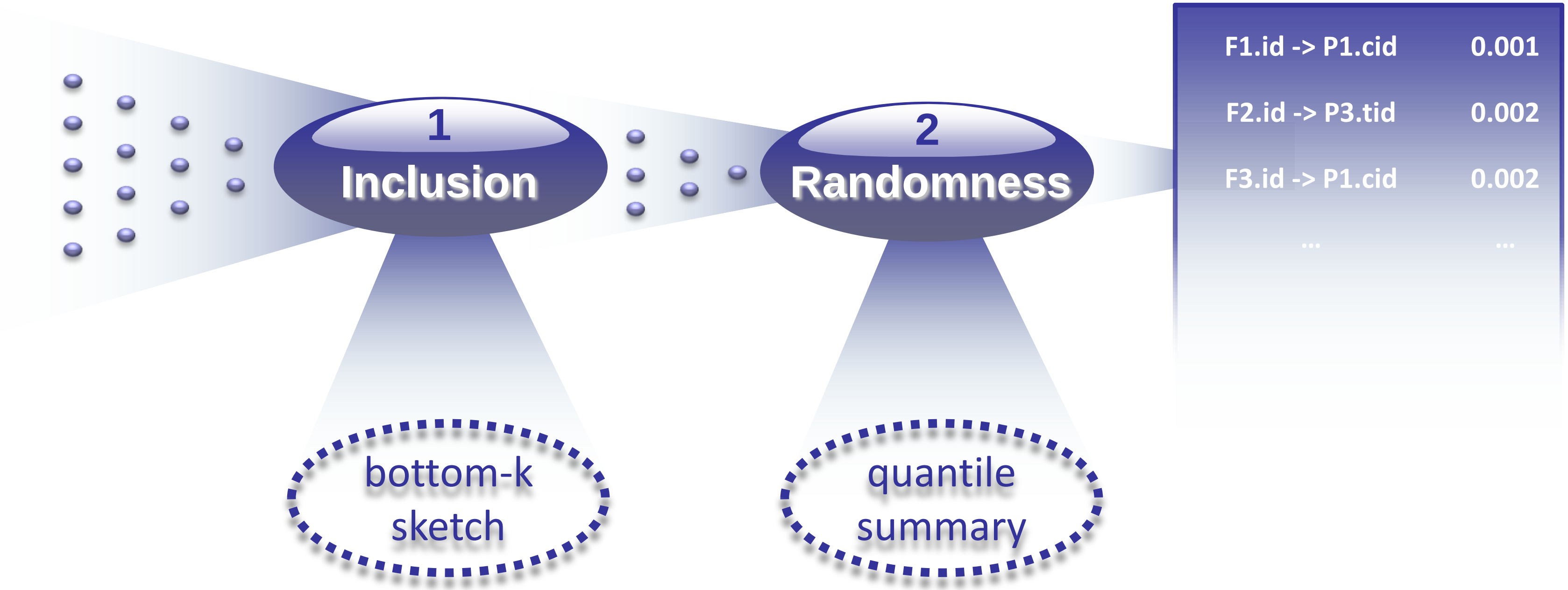
- One dimension
  - Equi-depth

Probability Distribution

- Quantile histogram
  - ℓ-quantiles of PK
  - Probability distribution of FK is defined w.r.t quantiles of PK
- Multi-dimension
  - Compute quantiles separately on each dimension
  - Construct a grid



Overall Algorithm



F1.id -> P1.cid	0.001
F2.id -> P3.tid	0.002
F3.id -> P1.cid	0.002
…	…

Outline

- Motivation
  - Extracting schema from tabular data
- Discovering good foreign keys from tabular data
  - Schema graph = nodes (tables, attributes) + edges (foreign keys)
  - Inclusion, randomness
  - Experimental results
- Discovering good table spaces
  - Clustering tables by topic, identifying important tables

Experiments

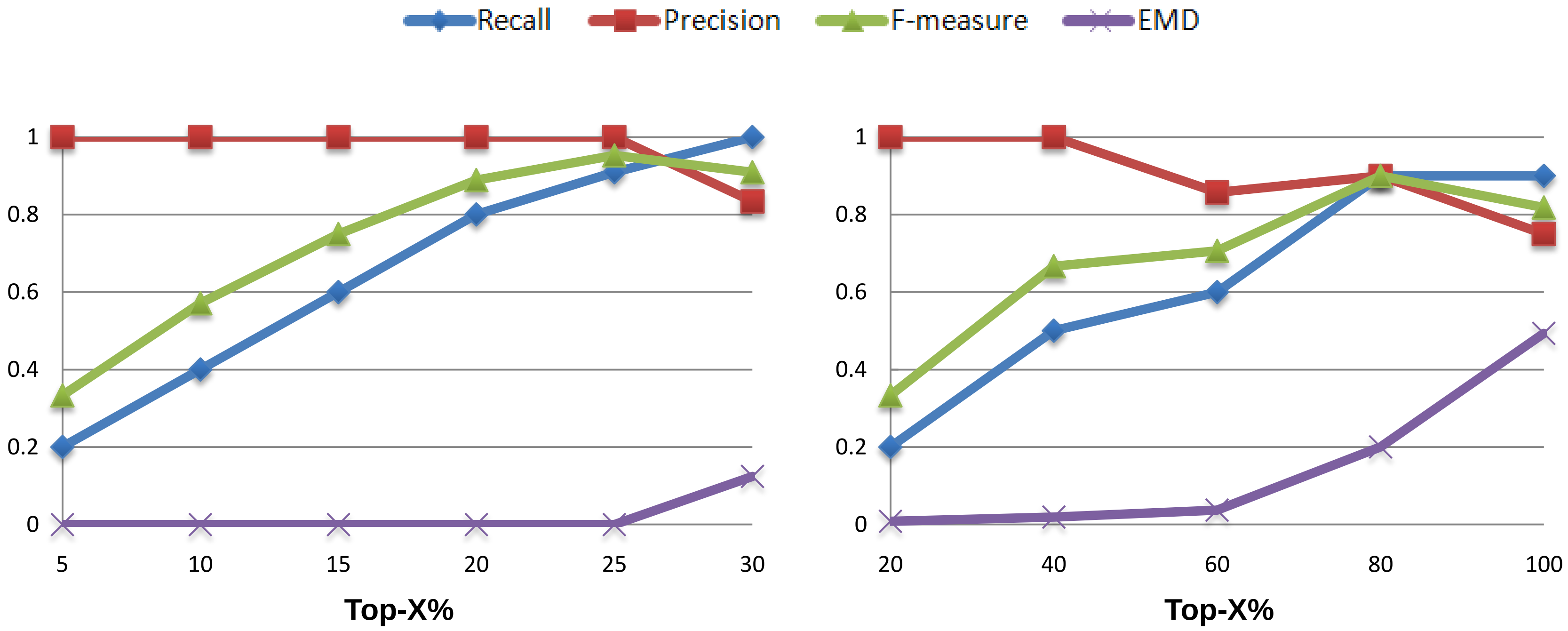
- Datasets
  - Benchmark databases: TPC-E, TPC-H
  - Real databases: Wikipedia, IMDB
- Evaluation
  - Accuracy
  - Scalability
  - Comparison

Experiments

- Number of candidates after inclusion test

Dataset	TPC-H		TPC-E		Wikipedia		IMDB	
SC-FK	9		44		10		8	
MC-FK	1		1		0		0	
θ =	0.9	1	0.9	1	0.9	1	0.9	1
SC-Candidates	38	34	304	214	12	8	24	24
MC-Candidates	1	1	4	3	0	0	0	0

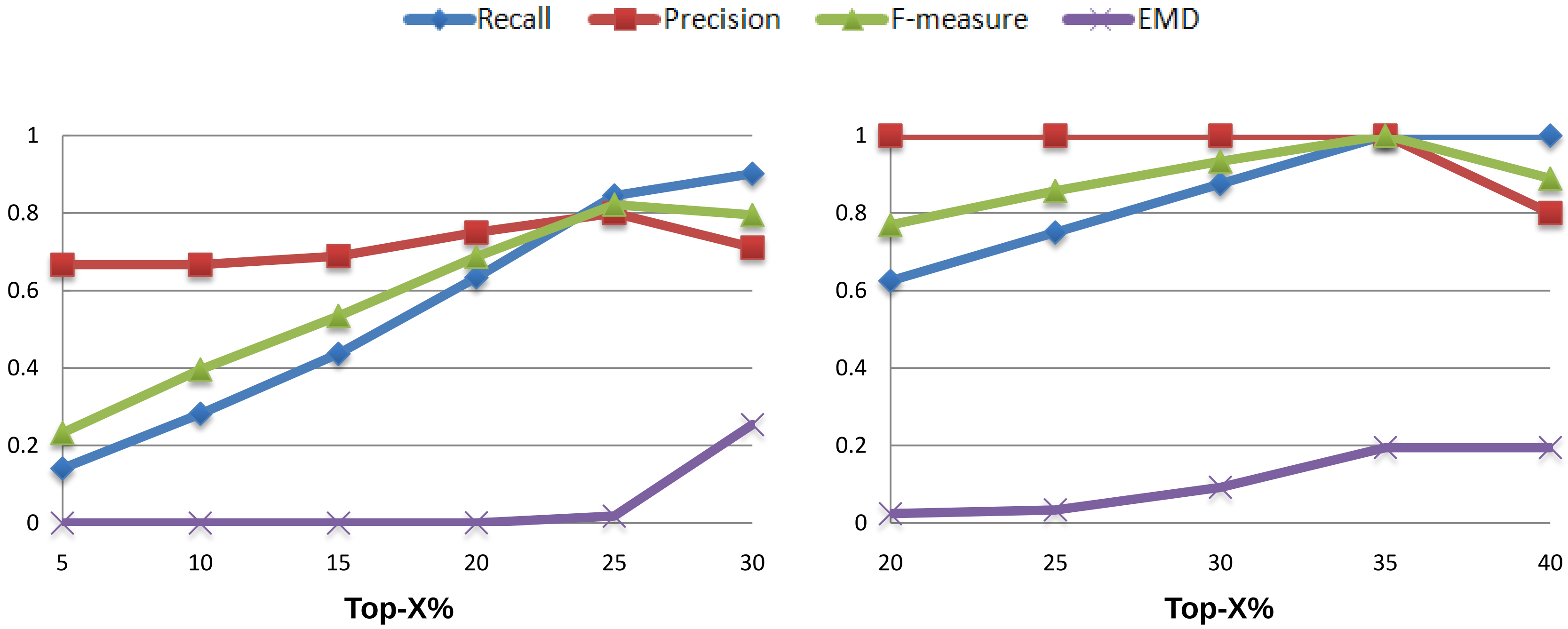
Accuracy



TPC-H

Wikipedia

Accuracy

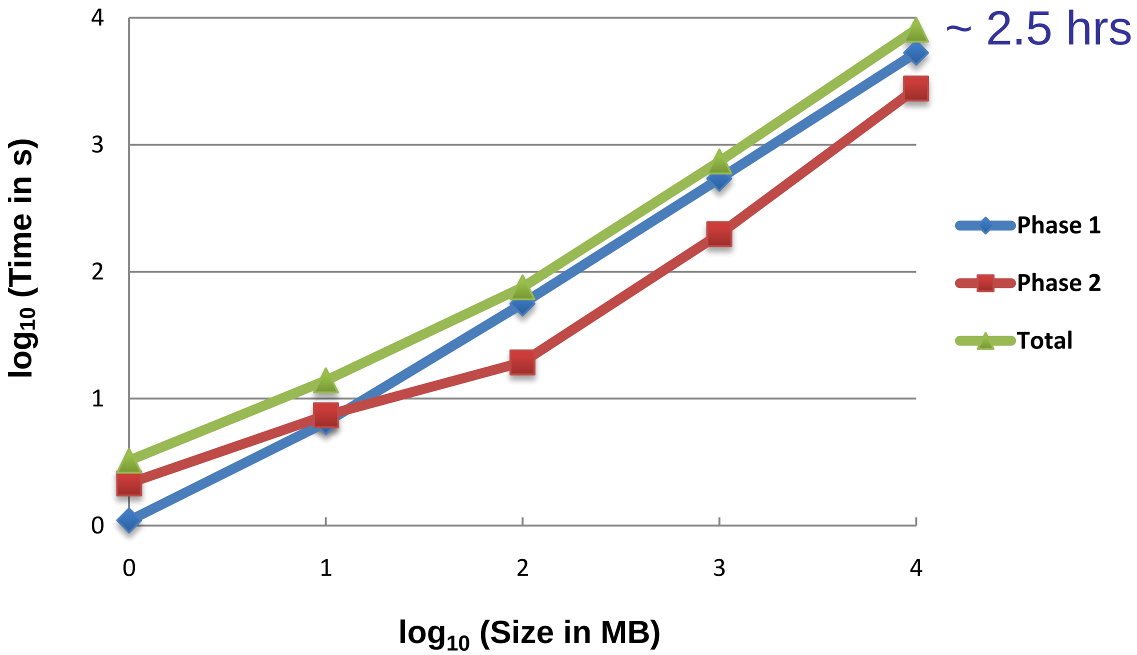


TPC-E

IMDB

Scalability

- TPC-H: 1M, 10M, 100M, 1G, 10G



Comparison

- ♦ Machine learning approach [4]
  - Use 7 heuristic rules
  - Need learning phase to train 4 classifiers
  - Need known foreign/primary key pairs for training
  - Discover single-column keys only
  - TPC-H: F-measure = 0.95 (best classifier J48)

    F-measure = 0.915 (average all classifiers)
- ♦ Our approach
  - TPC-H: F-measure = 0.95

[4] A. Rostin, et al. A Machine Learning Approach to Foreign Key Discovery. WebDB 2009

Summary

- Introduce randomness and show it can discover meaningful foreign keys, including multi-column foreign keys
- Provide efficient algorithm for evaluating randomness
- Present I/O efficient algorithm for discovering good foreign keys
- Experiments show the efficacy of our techniques

Outline

- Motivation
  - Extracting schema from tabular data

- Discovering good foreign keys from tabular data
  - Schema graph = nodes (tables, attributes) + edges (foreign keys)

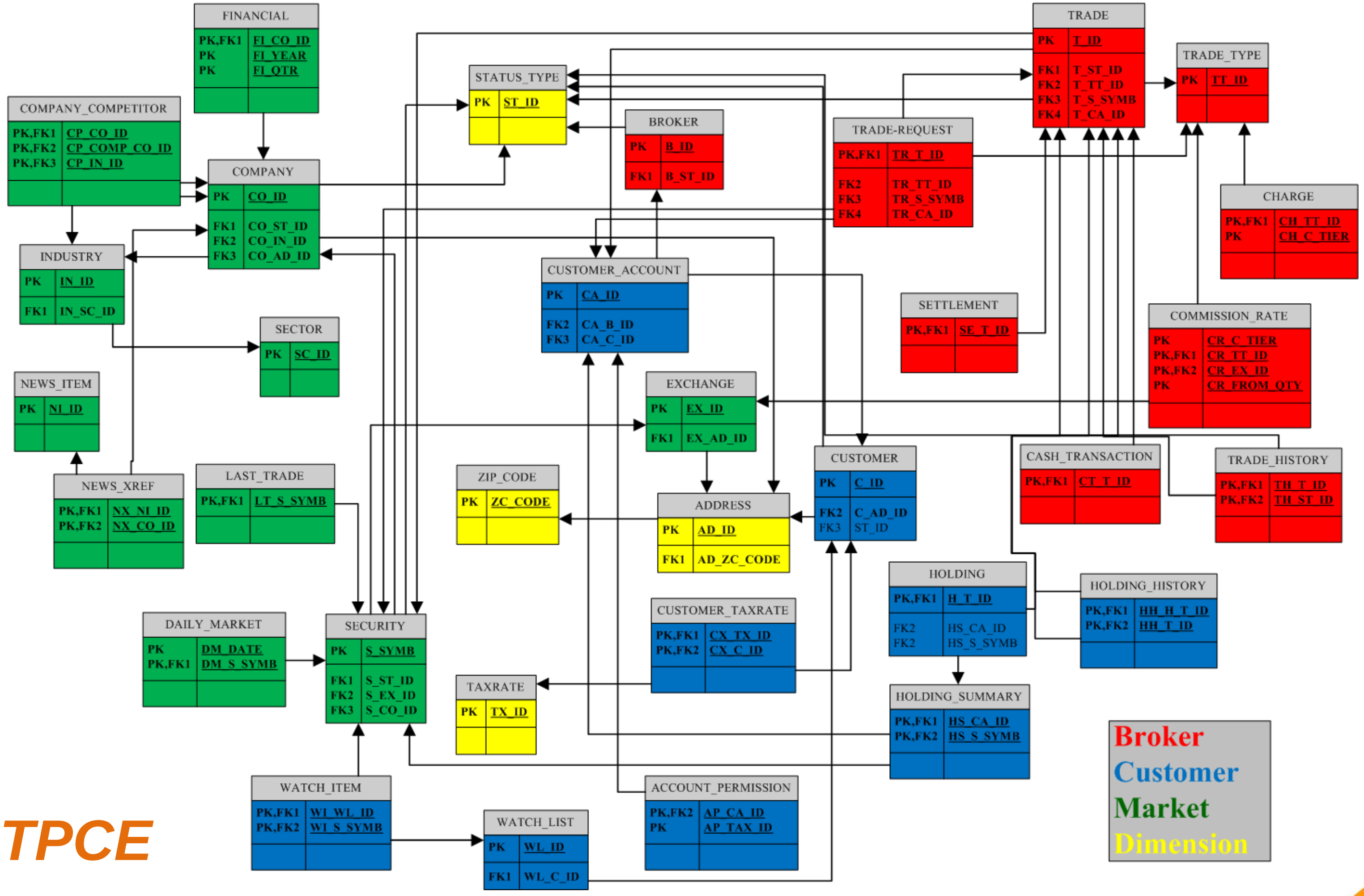
- Discovering good table spaces
  - Clustering tables by topic, identifying important tables

Discovering Table Spaces: Motivation

- Complex databases are challenging to explore and query
  - Consisting of hundreds of inter-linked tables
  - Users unfamiliar with the schema
  - Insufficient or unavailable schema information

- Propose a principled approach to discover table spaces
  - Cluster similar tables
  - Label each cluster by its most important table

Discovering Table Spaces: Motivation



TPCE

Outline

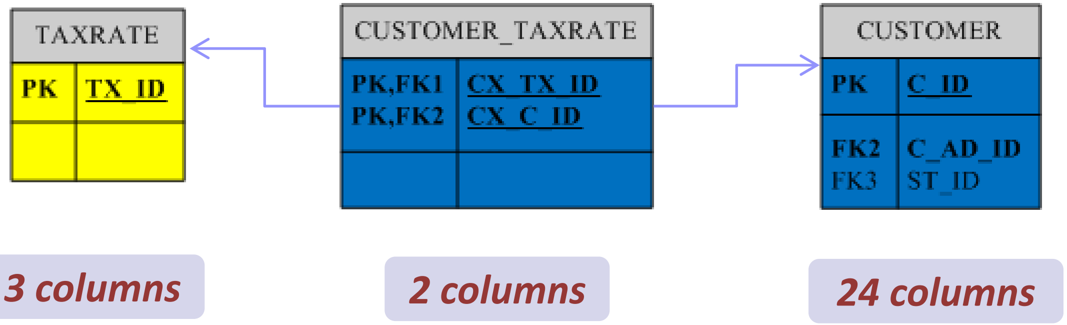
- Motivation
  - Extracting schema from tabular data

- Discovering good foreign keys from tabular data
  - Schema graph = nodes (tables, attributes) + edges (foreign keys)

- Discovering good table spaces
  - Clustering tables by topic, identifying important tables
  - Table importance, weighted k-center clustering

Table Importance

- Depends on
  - Internal information content



TAXRATE

PK TX_ID

CUSTOMER_TAXRATE

PK,FK1 CX_TX_ID

PK,FK2 CX_C_ID

CUSTOMER

PK C_ID

FK2 C_AD_ID

FK3 ST_ID

3 columns

2 columns

24 columns

  - External connectivity
    - Join behavior
    - Taxrate: 1 join
    - Customer: 5 joins

Table Importance (cont'd)

- Entropy of Attribute *A* in table *R* is defined as

$$H(R.A) = \sum_{i=1}^{k} p_i \log(1/p_i)$$

  - *R.A* = {*a*₁, …, *a*ₖ}
  - p_i is the fraction of tuples in *R* that have value a_i on attribute *A*

- The ***Information Content*** of a table *R* is defined as

$$\underline{IC(R)} = \boxed{\log|R|} + \sum_{R.A} H(R.A)$$

  - Create a primary key *R.Key* to table *R*
  - Add a self-loop *R.Key* – *R.Key*

***R.Key** consists of all attributes*

Table Importance (cont'd)

- *Entropy transfer matrix* $\prod$ associated with schema graph *G* is defined as:
  - For a join edge ***e = R.A – S.B***

$$Pr(R.A \to S.B) = \frac{\mathrm{H}(R.A)}{\log|R| + \sum_{R.A'} q_{A'} \cdot \mathrm{H}(R.A')}$$

$$= \frac{\mathrm{H}(R.A)}{\mathrm{IC}(R) + \sum_{R.A'} (q_{A'} - 1) \cdot \mathrm{H}(R.A')}$$

$q_{A'}$: number of join edges involving *R.A'* (including self-join)

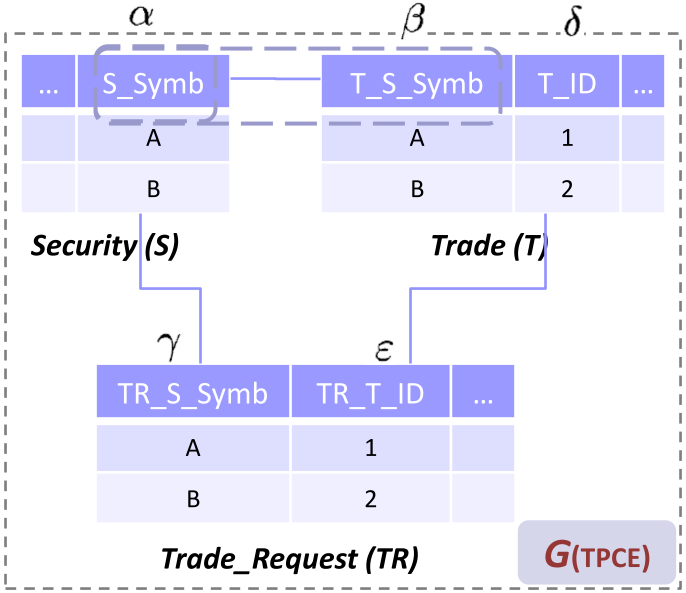
VE – *Variable entropy transfer* ***Model***

  - For a pair of tables *R* and *S*, define

$$\Pi[R,S] = \sum_{R.A-S.B} Pr(R.A \to S.B), \quad \Pi[R,R] = 1 - \sum_{S \neq R} \Pi[R,S]$$

Table Importance (cont'd)

- The ***importance*** of table *R* is defined as the stable-state value of a random walk on G, using probability matrix Π
  - Vector $\mathcal{I}$, s.t. $\mathcal{I} \times \Pi = \mathcal{I}$
  - Importance $\mathcal{I}(R)$, $R \in G$
- Example



Security (S)

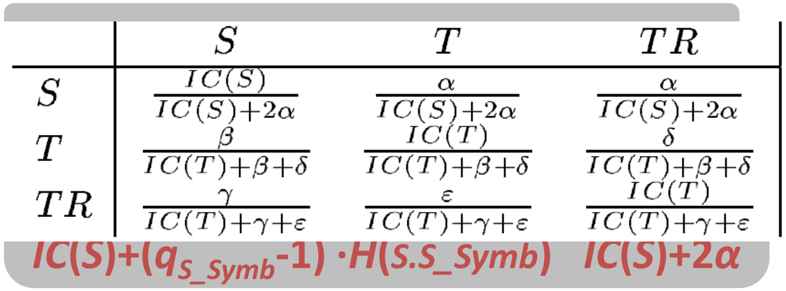
...	S_Symb
	A
	B

Trade (T)

T_S_Symb	T_ID	...
A	1	
B	2	

Trade_Request (TR)

TR_S_Symb	TR_T_ID	...
A	1	
B	2	

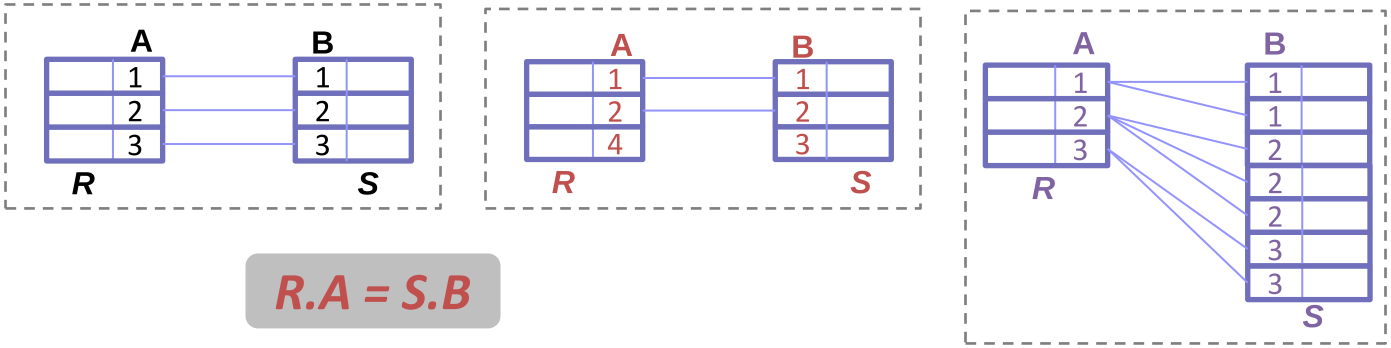


	S	T	TR
S	$\frac{IC(S)}{IC(S)+2\alpha}$	$\frac{\alpha}{IC(S)+2\alpha}$	$\frac{\alpha}{IC(S)+2\alpha}$
T	$\frac{\beta}{IC(T)+\beta+\delta}$	$\frac{IC(T)}{IC(T)+\beta+\delta}$	$\frac{\delta}{IC(T)+\beta+\delta}$
TR	$\frac{\gamma}{IC(T)+\gamma+\varepsilon}$	$\frac{\varepsilon}{IC(T)+\gamma+\varepsilon}$	$\frac{IC(T)}{IC(T)+\gamma+\varepsilon}$

$IC(S)+(q_{S\_Symb}-1) \cdot H(S.S\_Symb)$ $IC(S)+2\alpha$

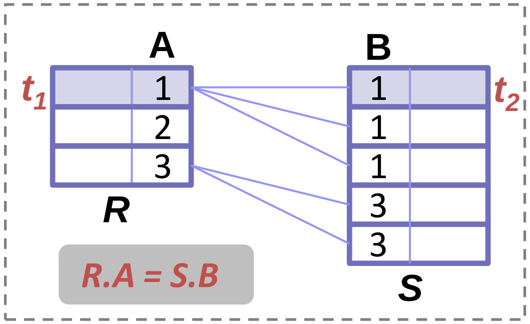
Table Similarity

- Distance = 1 - similarity
- Goal: define metric distance
  - Enables meaningful clustering over relational databases
- Table similarity depends on how *join edges* and *join paths* are instantiated



R.A = S.B

Table Similarity (cont'd)



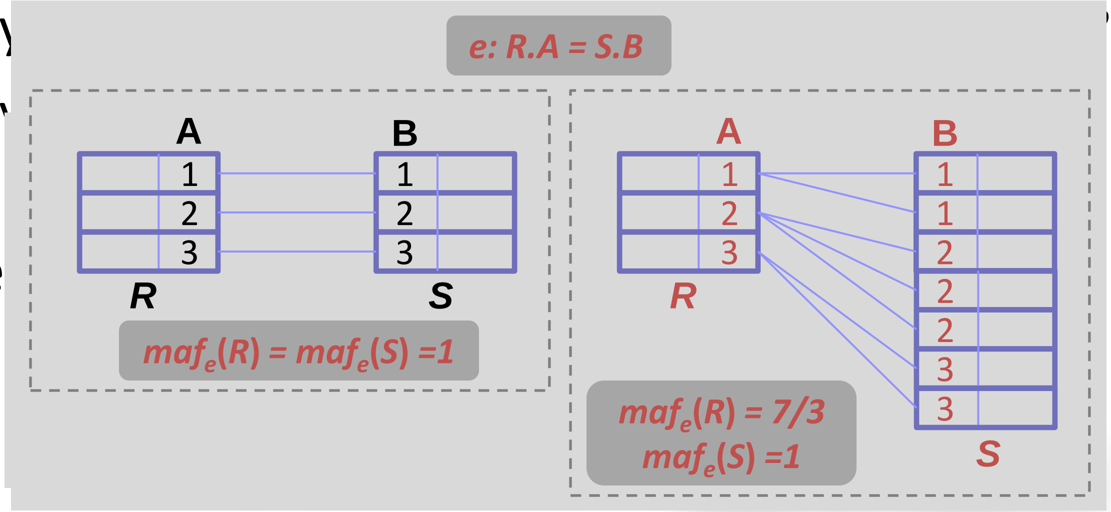
- Consider a join edge $e = R.A – S.B$
  - Tuples t_1, t_2 instantiate e
  - $fanout_e(t_i)$ is the fanout of t_i along e
    - $fanout_e(t_1) = 3$
  - Let q be the number of tuples in R s.t. $fanout_e(t_i) > 0$, define the **matching fraction** of R w.r.t. e as $f_e(R) = q/n$, $|R| = n$
    - $f_e(R) = 2/3 \leq 1$; $f_e(S) = 5/5 = 1 \leq 1$
  - Define the **matched average fanout** of R w.r.t. e as

$$maf_e(R) = \frac{\sum_{i=1}^{n} fanout_e(t_i)}{q}$$

    - $maf_e(R) = (3+2)/2 = 2.5 \geq 1$; $maf_e(S) = 5/5 = 1 \geq 1$

Table Similarity (cont'd)

- The similarity of tables *R* and *S* (w.r.t. $e_{(R,S)}$) must satisfy:
  - Property) and $f_e(S)$
  - Property ge fanouts $maf_e(R)$
- Define the



  - Property 2: Inverse proportional to the matched average fanouts $maf_e(R)$ and $maf_e(S)$

Table Similarity (cont'd)

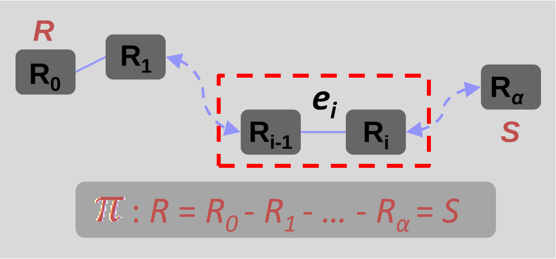
- Let $\pi : R = R_0 - R_1 - \ldots - R_\alpha = S$ be a path in *G*, define

$$Strength_\pi(R,S) = \prod_{i=1}^{\alpha} Strength_{e_i}(R_{i-1}, R_i)$$

- Table similarity (R, S):

$$\boxed{Strength(R,S) = \max_\pi Strength_\pi(R,S)}$$

- Distance (R, S)
  - $dist_s(R,S) = 1 - strength(R,S)$
  - $(R, dist_s)$ is a metric space



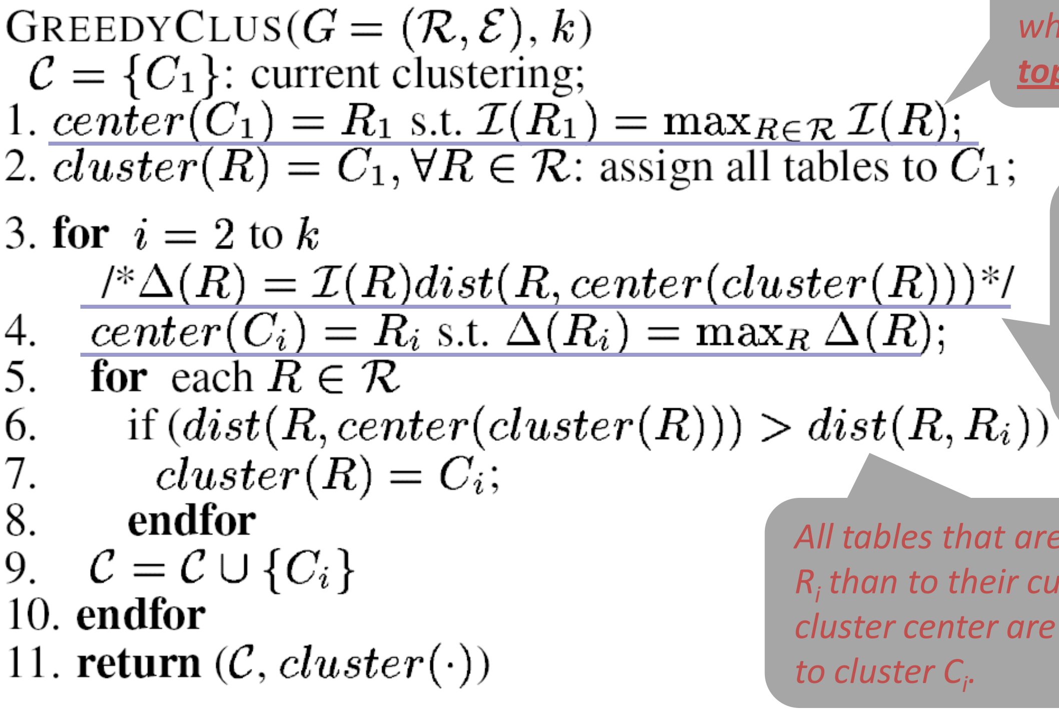
Clustering: Weighted *k*-Center

- Clustering Criteria:
  - Minimize the maximum ***distance*** between a cluster center and a table in that cluster
  - Take table ***importance*** into consideration, avoid grouping top important tables into one cluster
- Weighted *k*-Center clustering
  - Weights: table importance
  - Given *k* clusters $C = \{C_1, C_2, \ldots, C_k\}$, minimize

$$\mu(\mathcal{C}) = max_{i=1}^{k} \max_{R \in C_i} \mathcal{I}(R) dist(R, center(C_i))$$

  - NP-Hard

Weighted *k*-Center: Greedy Algorithm



```
GreedyClus($G = (\mathcal{R}, \mathcal{E}), k$)
 $\mathcal{C} = \{C_1\}$: current clustering;
 $center(C_1) = R_1$ s.t. $\mathcal{I}(R_1) = \max_{R \in \mathcal{R}} \mathcal{I}(R)$;
 $cluster(R) = C_1, \forall R \in \mathcal{R}$: assign all tables to $C_1$;
 for $i = 2$ to $k$
     /*$\Delta(R) = \mathcal{I}(R) dist(R, center(cluster(R)))$*/
     $center(C_i) = R_i$ s.t. $\Delta(R_i) = \max_R \Delta(R)$;
     for each $R \in \mathcal{R}$
         if ($dist(R, center(cluster(R))) > dist(R, R_i)$)
             $cluster(R) = C_i$;
     endfor
     $\mathcal{C} = \mathcal{C} \cup \{C_i\}$
 endfor
 return ($\mathcal{C}$, $cluster(\cdot)$)
```

*Start with one cluster, whose center is the **top-1 important table**.*

*Iteratively chooses the table R_i whose **weighted distance** from its cluster center is largest, and creates a new cluster with R_i as its center.*

All tables that are closer to R_i than to their current cluster center are reassigned to cluster C_i.

Outline

- Motivation
  - Extracting schema from tabular data
- Discovering good foreign keys from tabular data
  - Schema graph = nodes (tables, attributes) + edges (foreign keys)
- Discovering good table spaces
  - Clustering tables by topic, identifying important tables
  - Table importance, weighted k-center clustering
  - Experimental results

Experimental Results

- Validate the proposed three components in our approach
  - Model for table importance I_E *Entropy-based*
  - Distance function $dist_s$ *Strength-based*
  - Clustering: Weighted *k*-Center
- Other methods

Table Importance	Distance	Clustering
I_E	$dist_s$	**Weighted *k*-Center**
I_C [1]	$dist_c$ [1]	**Balanced-Summary**[1]
	$dist_p$ [2]	

I_c: Cardinality-initialized

$dist_c = 1 -$ *coverage*

$dist_p = 1 -$ *proximity*

[1] C.Yu and H.V.Jagadish. Schema summarization. VLDB 2006.
[2] H.Tong, C.Faloutsos and Y.Koren. Fast direction-aware proximity for graph mining. KDD 2007.

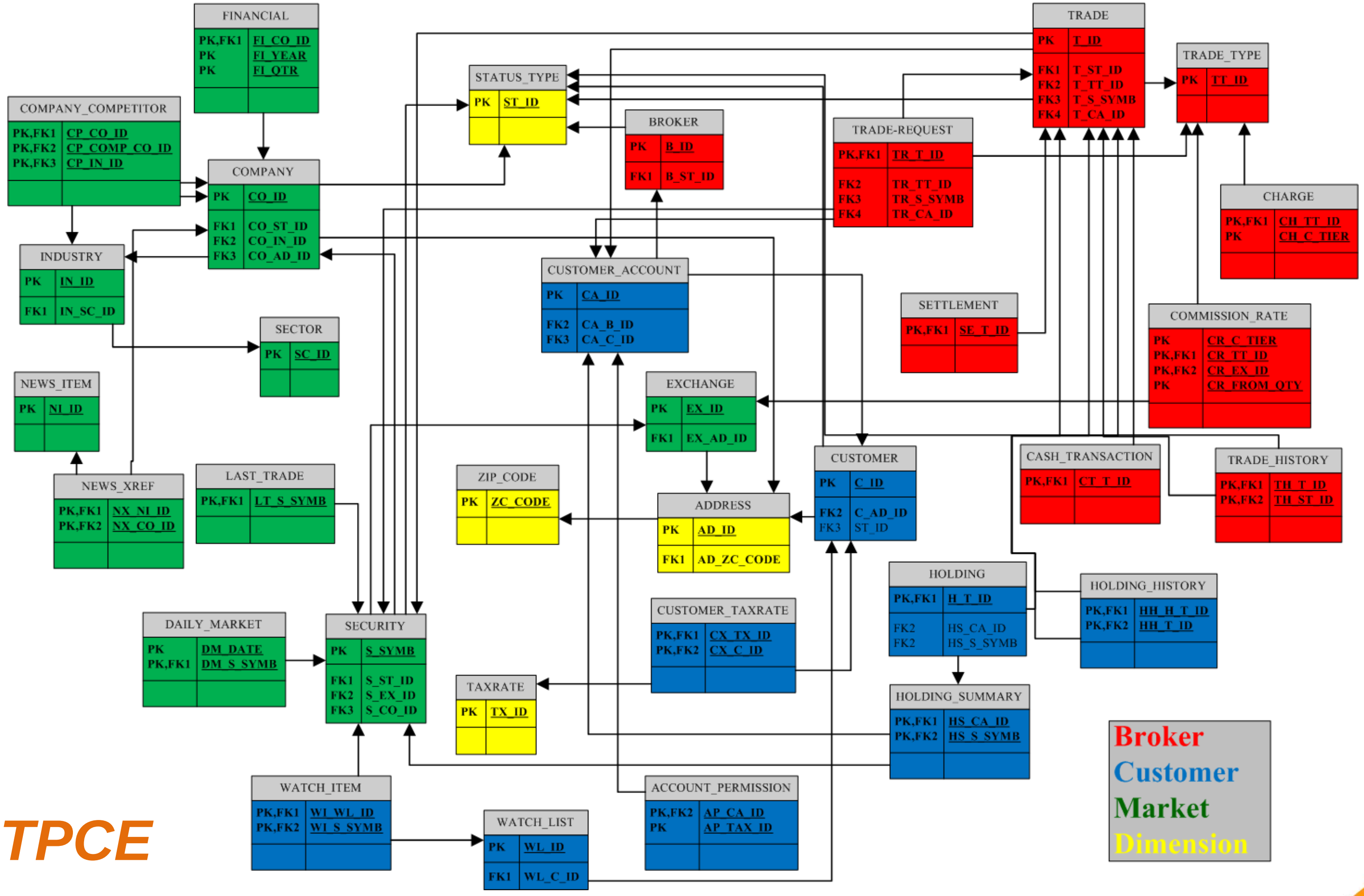
Experimental Results (cont'd)

- Data Sets: TPCE schema
  - Benchmark database simulating OLTP workload
  - 33 tables pre-classified into 4 categories
  - Two database instances: TPCE-1 / TPCE-2

Parameters	TPCE-1	TPCE-2
Number of customers	1,000	5,000
Initial Trade Days	10	10
Scale Factor	1,000	36,000

  - Affect the size of the majority of tables
  - Affect *Pr*(*R.A*→*S.B*), strength(R,S) for most pairs and maf_e for 1/3 of edges

TPCE Schema



FINANCIAL

PK,FK1	FI_CO_ID
PK	FI_YEAR
PK	FI_QTR

COMPANY_COMPETITOR

PK,FK1	CP_CO_ID
PK,FK2	CP_COMP_CO_ID
PK,FK3	CP_IN_ID

COMPANY

PK	CO_ID
FK1	CO_ST_ID
FK2	CO_IN_ID
FK3	CO_AD_ID

INDUSTRY

PK	IN_ID
FK1	IN_SC_ID

SECTOR

PK	SC_ID

NEWS_ITEM

PK	NI_ID

NEWS_XREF

PK,FK1	NX_NI_ID
PK,FK2	NX_CO_ID

LAST_TRADE

PK,FK1	LT_S_SYMB

DAILY_MARKET

PK	DM_DATE
PK,FK1	DM_S_SYMB

SECURITY

PK	S_SYMB
FK1	S_ST_ID
FK2	S_EX_ID
FK3	S_CO_ID

WATCH_ITEM

PK,FK1	WI_WL_ID
PK,FK2	WI_S_SYMB

WATCH_LIST

PK	WL_ID
FK1	WL_C_ID

STATUS_TYPE

PK	ST_ID

BROKER

PK	B_ID
FK1	B_ST_ID

CUSTOMER_ACCOUNT

PK	CA_ID
FK2	CA_B_ID
FK3	CA_C_ID

EXCHANGE

PK	EX_ID
FK1	EX_AD_ID

ZIP_CODE

PK	ZC_CODE

ADDRESS

PK	AD_ID
FK1	AD_ZC_CODE

CUSTOMER_TAXRATE

PK,FK1	CX_TX_ID
PK,FK2	CX_C_ID

TAXRATE

PK	TX_ID

ACCOUNT_PERMISSION

PK,FK2	AP_CA_ID
PK	AP_TAX_ID

TRADE

PK	T_ID
FK1	T_ST_ID
FK2	T_TT_ID
FK3	T_S_SYMB
FK4	T_CA_ID

TRADE_TYPE

PK	TT_ID

TRADE-REQUEST

PK,FK1	TR_T_ID
FK2	TR_TT_ID
FK3	TR_S_SYMB
FK4	TR_CA_ID

CHARGE

PK,FK1	CH_TT_ID
PK	CH_C_TIER

SETTLEMENT

PK,FK1	SE_T_ID

COMMISSION_RATE

PK	CR_C_TIER
PK,FK1	CR_TT_ID
PK,FK2	CR_EX_ID
PK	CR_FROM_QTY

CUSTOMER

PK	C_ID
FK2	C_AD_ID
FK3	ST_ID

CASH_TRANSACTION

PK,FK1	CT_T_ID

TRADE_HISTORY

PK,FK1	TH_T_ID
PK,FK2	TH_ST_ID

HOLDING

PK,FK1	H_T_ID
FK2	HS_CA_ID
FK2	HS_S_SYMB

HOLDING_HISTORY

PK,FK1	HH_H_T_ID
PK,FK2	HH_T_ID

HOLDING_SUMMARY

PK,FK1	HS_CA_ID
PK,FK2	HS_S_SYMB

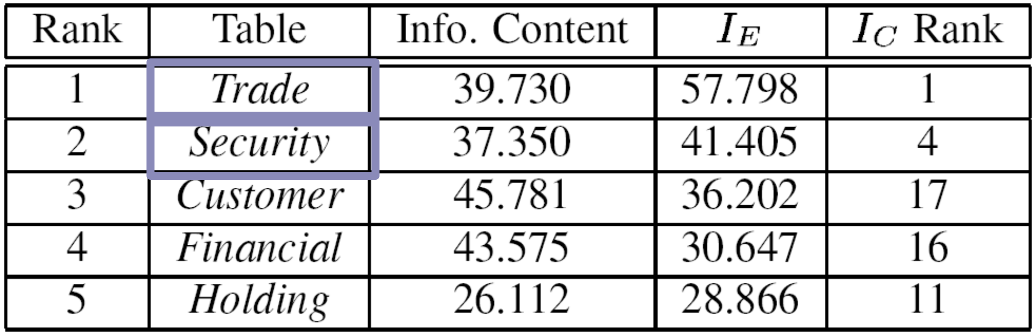
Broker
Customer
Market
Dimension

TPCE

Table Importance

- Comparison of I_E and I_C models

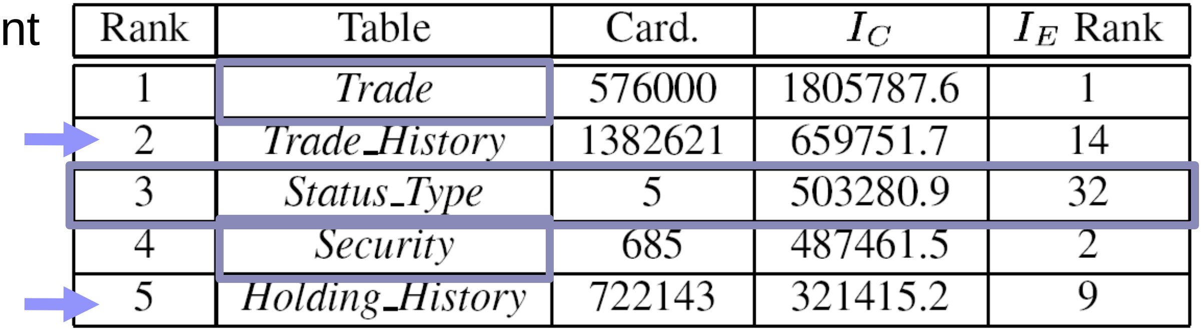
  - Top-5 Important Tables in I_E and their ranks in I_C



Rank	Table	Info. Content	I_E	I_C Rank
1	*Trade*	39.730	57.798	1
2	*Security*	37.350	41.405	4
3	*Customer*	45.781	36.202	17
4	*Financial*	43.575	30.647	16
5	*Holding*	26.112	28.866	11

I_E more accurate than I_C

  - Top-5 Important Tables in I_C and their ranks in I_E

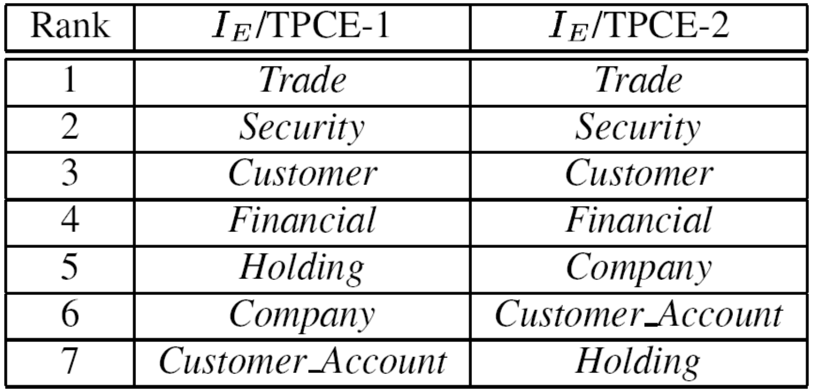


Rank	Table	Card.	I_C	I_E Rank
1	*Trade*	576000	1805787.6	1
2	*Trade_History*	1382621	659751.7	14
3	*Status_Type*	5	503280.9	32
4	*Security*	685	487461.5	2
5	*Holding_History*	722143	321415.2	9

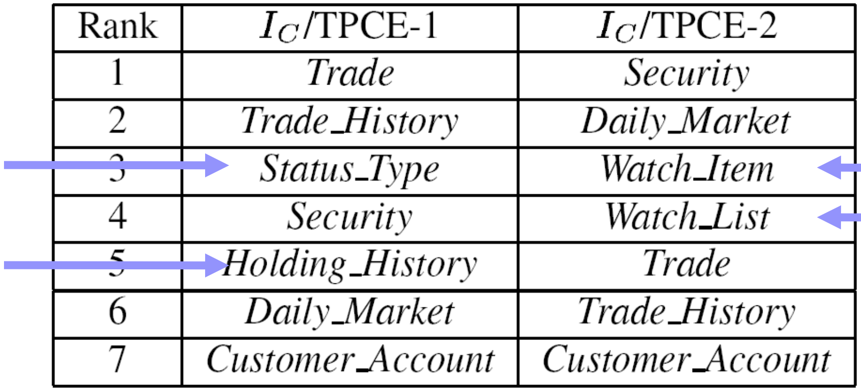
Table Importance (cont'd)

- Consistency of I_E and I_C models

  - Top-7 Important Tables in I_E and I_C for TPCE-1 and TPCE-2



Rank	I_E/TPCE-1	I_E/TPCE-2
1	*Trade*	*Trade*
2	*Security*	*Security*
3	*Customer*	*Customer*
4	*Financial*	*Financial*
5	*Holding*	*Company*
6	*Company*	*Customer_Account*
7	*Customer_Account*	*Holding*



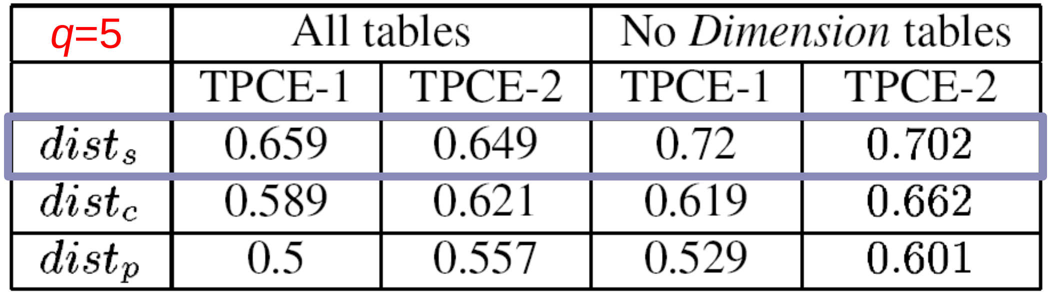
Rank	I_C/TPCE-1	I_C/TPCE-2
1	*Trade*	*Security*
2	*Trade_History*	*Daily_Market*
3	*Status_Type*	*Watch_Item*
4	*Security*	*Watch_List*
5	*Holding_History*	*Trade*
6	*Daily_Market*	*Trade_History*
7	*Customer_Account*	*Customer_Account*

I_E more consistent than I_C

Distance Between Tables

- Accuracy of distance functions
  - Observation: for each table *R*, its distances to tables within the same category (*pre-defined*) should be smaller than its distances to tables in different categories
  - *n*(*R*): # top-*q* nbrs (NN_R) of *R* under dist. *d* ($dist_S$, $dist_C$, $dist_P$)
  - *m*(*R*): # *tables* (∈ NN_R) in the same category as *R* under dist. *d*
  - Calculate:

$$acc(d) = \frac{\sum_R \frac{m(R)}{n(R)}}{n}$$



q=5	All tables		No *Dimension* tables	
	TPCE-1	TPCE-2	TPCE-1	TPCE-2
$dist_s$	0.659	0.649	0.72	0.702
$dist_c$	0.589	0.621	0.619	0.662
$dist_p$	0.5	0.557	0.529	0.601

***$dist_S$* most accurate**

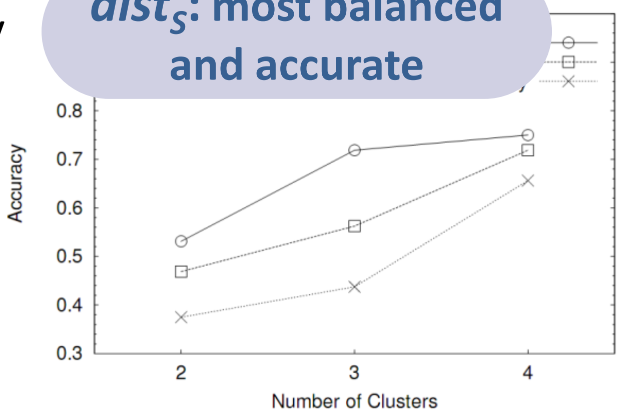
Table Space Discovery Algorithms

- Weighted *k*-Center over three distance functions

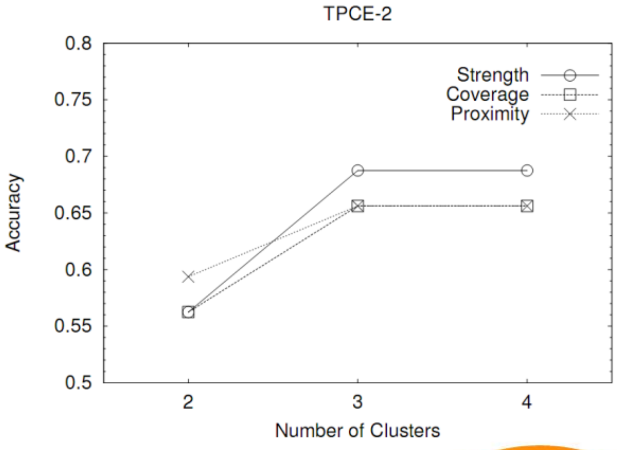
k	C_i	$dist_s$				$dist_c$				$dist_p$			
		center(C_i)	$n(C_i)$	$m(C_i)$	$acc(C_i)$	center(C_i)	$n(C_i)$	$m(C_i)$	$acc(C_i)$	center(C_i)	$n(C_i)$	$m(C_i)$	$acc(C_i)$
3	1	*Trade*	9	6	0.67	*Trade*	19	8	0.42	*Trade*	21	8	0.38
	2	*Security*	13	11	0.85	*Financial*	7	7	1.0	*Security*	6	4	0.67
	3	*Customer*	10	6	0.6	*Customer*	6	3	0.5	*Customer*	5	2	0.4
4	1	*Trade*	9	6	0.67	*Trade*	13	7	0.54	*Trade*	14	8	0.57
	2	*Security*	12	10	0.83	*Financial*	7	7	1.0	*Security*	6	4	0.67
	3	*Customer*	10	6	0.6	*Customer*	6	3	0.5	*Customer*	5	2	0.4
	4	*Financial*	1	1	1.0	*Security*	4	6	0.67	*Financial*	7	7	1.0

✓Summary Accuracy

$$acc(\mathcal{C}) = \frac{\sum_{i=1}^{k} m(C_i)}{n}$$



dist**$_S$**: most balanced and accurate



Summary

- Novel approach for discovering good table spaces
  - A new model for table importance
  - A metric distance over schema tables
  - A summarization algorithm

- Ongoing work
  - Summarizing schema graphs for at-a-glance understanding

Parting Thoughts

- Schema extraction is critical for automatically creating databases from collections of tables
  - We focused on discovering good foreign keys, tables spaces
- Other work on discovering good primary keys, good FDs:
  - Y. Sismanis, P. Brown, P. Haas, and B. Reinwald. Gordian: Efficient and scalable discovery of composite keys. VLDB 2006
  - P. Andritsos, R. Miller, and P. Tsaparas. Information-theoretic tools for mining database structure from large data sets. SIGMOD 2004
- Exciting research area with a lot of practical utility!

Discovering Foreign Keys: Motivation

- In practice, foreign keys are often NOT specified in the schema

- What if this happens in enterprise databases?
  - Thousands of tables
  - Tens of thousands of columns
  - Insufficient (missing/out-of-date) documentation

Objective

- To efficiently discover FK/PK relationships in relational databases
  - Single-column
  - Multi-column

Randomness

- Randomness measure
  - How close are the (multi-dimensional) distributions of F and P?

Our Approach

- Counter-examples exist for randomness rule as well
  - Table *P* contains all NUS graduate students
  - *SID* is generated according to the year, e.g. g10xxxxx
  - Table *F* references only the students who enrolled in NUS in 2010
  - *F.SID* is not a random sample of *P.SID*
  - Foreign key table is correlated to the way keys are generated

Our Approach

- Counter-examples exist for randomness rule as well
  - Table *P* contains all NUS graduate students
  - *SID* is generated according to the year, e.g. g10xxxxx
  - Table *F* references the students who come from China
  - *F.SID* is a random sample of *P.SID*

- No solution with 100% precision/recall

- Experiments on real databases show randomness rule can effectively eliminate false positives and achieve high recall!

Overall Algorithm

- Two passes over data

- Phase 1
  - Read all columns in table-wise order
  - Build bottom-k sketches for all single columns and all multi-column PKs
  - Build quantile summaries for all single/multi-column PKs
  - Evaluate single-column inclusions

Overall Algorithm

- Two passes over data

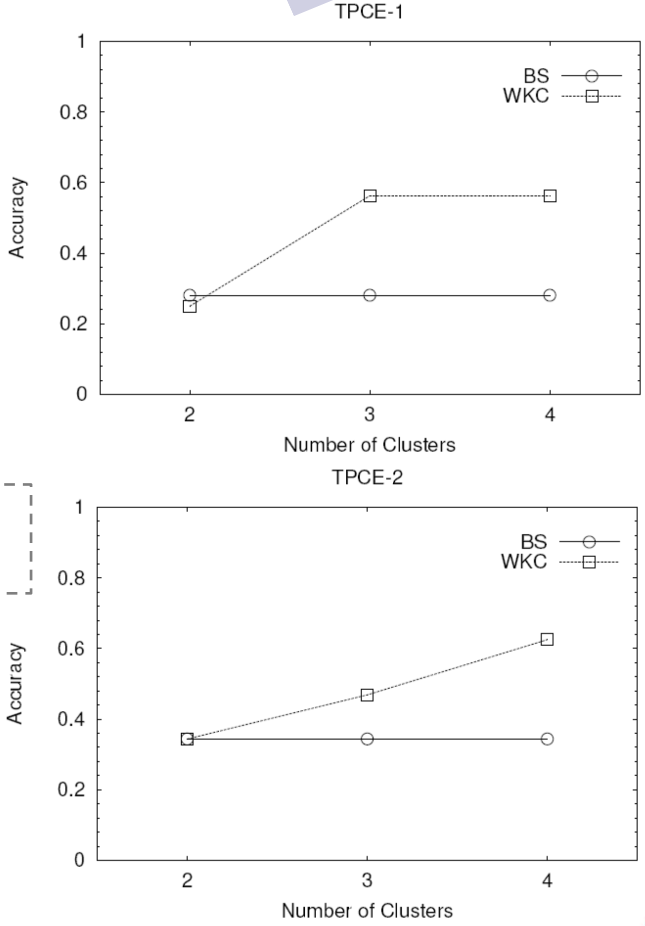
- Phase 2
  - Compute multi-column candidate FKs
  - For each single-column candidate FK, scan it, compute distribution histograms w.r.t all relevant PKs
  - For each multi-column candidate FK, scan it, compute bottom-k sketch and distribution histograms w.r.t all PKs
  - Evaluate randomness

Clustering Algorithms

- Accuracy of a summary
  - TPCE is pre-classified into 4 categories: ***Broker***, ***Customer***, ***Market*** and ***Dimension***
  - $m(C_i)$: # tables in C_i with the same category as *center*(C_i)
  - Given a summary C = {C_1, C_2, …, C_k}, calculate $acc(\mathcal{C}) = \frac{\sum_{i=1}^{k} m(C}{n}$
  - Balanced-Summary (BS) [1]
  - Weighted *k*-Center (WKC)

→ **WKC is more accurate**

Based on l_C and $dist_C$ (coverage)



Related Work

- C. Yu and H. V. Jagadish. *Schema summarization*. VLDB’06
- H.Tong, C.Faloutsos and Y.Koren. *Fast direction-aware proximity for graph mining*. KDD’07
- W. Wu, B. Reinwald, Y. Sismanis and R. Manjrekar. *Discovering topical structures of databases*. SIGMOD’08