Relational AI Systems: In Pursuit of Simplicity

Dan Olteanu April 24, 2023



My research agenda:

- Investigate the principles behind computational challenges for data processing
- Design simple and scalable solutions towards these challenges in both academia and industry

This talk: Two ideas in relational AI

My research agenda:

- Investigate the principles behind computational challenges for data processing
- Design simple and scalable solutions towards these challenges in both academia and industry

This talk: Two ideas in relational AI

• But first: Why relational?

Simple model rooted in logic, invented by Codd at IBM in 1969

- The most widely deployed software paradigm of any type Similar reach: zip, libpng, libjpeg
- \bullet > trillion SQLite active instances:
 - Android/iPhone/iOS devices
 - Mac/Windows10 machines
 - Firefox/Chrome/Safari browsers
 - Skype, iTunes, PhP, Python
 - smart TV sets
 - automotive multimedia systems



Why is the Relational Paradigm Ubiquitous?

It is not for lack of trying something else..

- Transactional databases were initially navigational
- Relational took over:

Oracle (née Relational Software)

Current Market Cap: \$250.4B

Ingres

(née Relational Technology)

Informix

(née Relational Databases)



Why is the Relational Paradigm Ubiquitous?

It is not for lack of trying something else..

- Analytic databases were initially multidimensional arrays (tensors)
- Relational took over:

Tableau Software Market Cap at sale: \$11.6B



Why is the Relational Paradigm Ubiquitous?

It is not for lack of trying something else..

- Big Data systems were initially MapReduce/Spark
- Relational took over:

Snowflake Software Current Market Cap: \$97.5B

Spark turned relational

Google BigQuery

AWS Cloud Databases



Relational Always Wins!

	IBM Sy	stem R	Teradata	44 + c	a b l e a v	relational <u>AI</u>		
		ORACLE	Micr	croStrategy Snowflake		k snowflake		
	1970	1980	1990	2000	2010	2020	2030	
DBMS		Client-Server Relational DBMS	C B Da	Client-Server ta Warehous	Cle e Wa	oud Data arehouse K	Relational nowledge Graph	s
Data Model		~3rd Normal Form	Sta Sno	r and wflake	Semi- Structure	ed	Graph	
	(Hie Navi Rel a	rarchical & Network igational → itional	<) Multi-dir Array/Te Relatior	nensional nsor → nal	MapReduc Relational	e → D P N Te R	ataFrame, rocedural, avigational, ∋nsor→ elational	23

(graphic courtesy of Molham Aref, RelationalAI CEO)

Relational Always Wins!

	IBM Sys	stem R	Teradata	‡‡‡ + o	b e a u		relational	AI
		ORACLE	Micro	Strategy	s r	nowflake		
	1970	1980	1990	2000	2010	2020	2030	\setminus
DBMS		Client-Server Relational DBMS	Cl S Data	ient-Server a Warehouse	Cloud Wareh	Data louse Ki	Relational nowledge Gra	aphs
Data Model		~3rd Normal Form	Star Snow	and /flake	Semi- Structured		Graph	_ /
	(Hier Navig Rela	archical & Networ gational → tional	k) Multi-dim Array/Ter Relation a	ensional ısor → al	MapReduce → Relational	Da Pr Na Te	ataFrame, ocedural, avigational, nsor→ elational	7

"Making the simple complicated is commonplace; making the complicated simple, that's creativity." – C. Mingus

(graphic courtesy of Molham Aref, RelationalAI CEO)

• Very simple data model rooted in logic

First principles then implementation

- Very simple data model rooted in logic
 First principles then implementation
- Separation of What from How

Relational systems are declarative

• Very simple data model rooted in logic

First principles then implementation

• Separation of What from How

Relational systems are declarative

• Automatic Programming

query optimization, memory mgt, parallelization, incrementalization

• Very simple data model rooted in logic

First principles then implementation

• Separation of What from How

Relational systems are declarative

• Automatic Programming

query optimization, memory mgt, parallelization, incrementalization

• Easy to understand by domain experts

Domain experts are cheaper and more plentiful than programmers

• Very simple data model rooted in logic

First principles then implementation

• Separation of What from How

Relational systems are declarative

• Automatic Programming

query optimization, memory mgt, parallelization, incrementalization

• Easy to understand by domain experts

Domain experts are cheaper and more plentiful than programmers

• Easy to implement in practice

Tables have rows, all row have the same columns :)

• ...

Achilles heel: Rigid data format that encourages redundancy

Redundancy in data begets redundancy in computation

- Redundancy hides the true computational complexity
- Key reason for lack of efficiency and scalability

Achilles heel: Rigid data format that encourages redundancy

Redundancy in data begets redundancy in computation

- Redundancy hides the true computational complexity
- Key reason for lack of efficiency and scalability

This talk looks at redundancy when:

- Reasoning under uncertainty
- Training machine learning models over relational data

Probabilistic Databases



Many Worlds Interpretation

Data may admit many interpretations or possible worlds

• Different runs of scientific and social experiments may have (slightly) different outcomes



Example: Manually Completed Census Forms

	Social Security Number: Name:	185 Smith	
	Marital Status:	(1) single ⊠ (2) married(3) divorced □ (4) widowed	•
1			
		102	
	Social Security Number:	185	
	Social Security Number: Name:	<u>185</u> Brown	
	Social Security Number: Name: Marital Status:	185 Brown (1) single □ (2) married	

Example: Manually Completed Census Forms



Several interpretations of the above simple forms are possible

- What is the marital status of Smith or Brown?
- What are their social security numbers? 185? 186? 785?
- Some interpretations more likely (probable) than others

SSN	Name	Status	Prob
185	Smith	Single	0.2
185	Brown	Single	0.2

SSN	Name	Status	Prob	SSN	Name	Status	Prob
185	Smith	Single	0.2	785	Smith	Single	0.3
185	Brown	Single	0.2	185	Brown	Single	0.2

SSN	Name	Status	Prob	SSN	Name	Status	Prob
185	Smith	Single	0.2	785	Smith	Single	0.3
185	Brown	Single	0.2	185	Brown	Single	0.2
SSN	Name	Status	Proh	SSN	Name	Status	Proh
SSN	Name	Status	Prob	SSN	Name	Status	Prob
SSN 185	Name Smith	Status Married	Prob 0.2	SSN 785	Name Smith	Status Married	Prob 0.3

SSN	Name	Status	Prob	SSN	Name	Status	Prob
185	Smith	Single	0.2	785	Smith	Single	0.3
185	Brown	Single	0.2	185	Brown	Single	0.2
SSN	Name	Status	Prob	SSN	Name	Status	Prob
SSN 185	Name Smith	Status Married	Prob 0.2	SSN 785	Name Smith	Status Married	Prob 0.3

.

for each interpretation for Smith, each possible interpretation for Brown

Total interpretations = 32: 4 (for Smith) \times 8 (for Brown)

- Very many
$$\approx 10^{10^6}$$
 worlds (in our experiments)

- Each world needs $\approx 1~Gigabyte~(\text{678,000 book pages})$
- Each world has a likelihood (probability) for being true

- Very many
$$\approx 10^{10^6}$$
 worlds (in our experiments)

- Each world needs $\approx 1~Gigabyte~(\text{678,000 book pages})$
- Each world has a likelihood (probability) for being true

Answer: Avoid redundancy in the representation

- Very many
$$\approx 10^{10^6}$$
 worlds (in our experiments)

- Each world needs $\approx 1~Gigabyte~(\text{678,000 book pages})$
- Each world has a likelihood (probability) for being true

Answer: Avoid redundancy in the representation

+ 10^{10^6} worlds need \approx 6 Gigabytes (in our experiments)

How to efficiently query all the worlds?

Computational Challenges in Probabilistic Databases

How to efficiently query all the worlds?

- Efficient \neq Query one world at a time
- $\bullet\,$ Ideal: Time to query all worlds \approx time to query one world

Computational Challenges in Probabilistic Databases

How to efficiently query all the worlds?

- Efficient \neq Query one world at a time
- Ideal: Time to query all worlds \approx time to query one world

Distinguish fast queries from slow queries

 Syntactic characterization of queries by their computational complexity ⇒ Dichotomy for query answering

Dichotomy for Query Answering



Dichotomy for Query Answering

Queries are

either easy and can be solved efficiently

or hard and cannot be solved efficiently



Dichotomies sound simple yet are very challenging to prove.

"Simple can be harder than complex." -Steve Jobs

Dichotomy for Query Answering

Queries are

either easy and can be solved efficiently

• Exact computation feasible

or hard and cannot be solved efficiently

• Approximate computation feasible



Dichotomies sound simple yet are very challenging to prove.

"Simple can be harder than complex." -Steve Jobs

How Do Hard Queries Look Like?



How Do Easy Queries Look Like?



Systems and Theory for Probabilistic Databases



Systems and Theory for Probabilistic Databases



Systems and Theory for Probabilistic Databases





Application: Probabilistic Google Search



Probabilistic Google Search with SPROUT²

<u></u>	sprout?									
File M	laintenance									
enlighte	enlightenment philosophers European Countries + Compose Query Query Result									
enlightenment philosophers Retrieve Google Square										
Key		Image	Description	Date Of Birth	Place Of Birth	Place Of Death	Died	Date Of Death		
Voltaire										
John Loc	cke	http://oregonstate	His writings influe	29 August 1632	Wrington, Somerse	Essex, England	1704-10-28	1704-10-28		
David Hu	ume	http://upload.wikim	David Hume (7 Ma	1711-04-26	Edinburgh, Scotland	Edinburgh, Scot Par	is, France high confic	lence)8-25		
Immanu	iel Kant	http://www.homodi	Immanuel Kant wa	April 22, 1724	Kaliningrad, Russia	Kaliningrad, Rus Fra	nce low confide	ence)2-12		
Charles	de Second	http://www.constit	Charles-Louis de	January 18, 1689	Bordeaux, France	Paris, France Par	is low confide	ance ary 10, 1755		
Denis Di	iderot	http://www.utm.ed	Denis Diderot (Oct	5 October 1713	Langres, France	Paris, France	1/84-07-31	1/84-07-31		
Roussea	au	http://upload.wikim	Jean-Jacques Rous	June 28, 1712		Malone, NY	7/2/1778	June 28, 1712		
Thomas	Hobbes	http://oregonstate	Enlightenment phil	April 5, 1588	Malmesbury, Wilts	Derbyshire, England	1679-12-04	December 4, 1679		
Adam Si	mith	http://upload.wikim	Adam Smith (bapti	1991-04-29	Kirkcaldy, Scotland	Edinburgh, Scotland	1790-07-17	1790-07-17		
Thomas	Paine	http://www.catscra	Thomas "Tom" Pai	1737-01-29	Thetford, England	New York City	8-Jun-1809	June 8, 1809		
Baruch !	Spinoza	http://www.phillwe	Dutch rationalist p	November 24, 1632	Amsterdam	Hague, Netherlands	1677-02-21	1677-02-21		
Rene De	escartes	http://oregonstate	René Descartes Fr	March 31, 1596	La Haye, France	Stockholm, Sweden	11 February 1650	February 11, 1650		
Gottfried	d Leibniz	http://www.gap-sv	Gottfried Wilhelm L	1 Iulv 1646	Leipzig. Germany	Hanover. Germany	14 November 1716	14 November 1716	•	
			9	how Database Querie	es for Caching of this	Table				

SPROUT									
File Maintenance									
enlightenment philosophers European C	ountries + Compose Query Query Re	sult							
European Countries		Retrieve G	oogle Fusion Table						
English short and formal names	Capital	Population	Area						
Albania - Republic of Albania	Tirana	2986952	28,748 km2(11,100 sq mi)						
Andorra - Principality of Andorra	Andorra la Vella	84525	468 km2(181 sq mi)						
Armenia - Republic of Armenia	Yerevan	2966802	29,743 km2(11,484 sq mi)[n 1]						
Austria - Republic of Austria	Vienna	8214160	83,871 km2(32,383 sq mi)						
Azerbaijan - Republic of Azerbaijan	Baku	8303512	86,600 km2(33,436 sq mi)[n 1]						
Belarus - Republic of Belarus	Minsk	9612632	207,600 km2(80,155 sq mi)						
Belgium - Kingdom of Belgium	Brussels	10423493	30,528 km2(11,787 sq mi)						
Bosnia and Herzegovina	Sarajevo	3843126	51,129 km2(19,741 sq mi)						
Bulgaria - Republic of Bulgaria	Sofia	7148785	110,879 km2(42,811 sq mi)						
Croatia - Republic of Croatia	Zagreb	4486881	56,594 km2(21,851 sq mi)						
Cyprus - Republic of Cyprus	Nicosia	1102677	9,251 km2(3,572 sq mi)[n 1]						
Czech Republic - Czech Republic	Prague	10201777	78,867 km2(30,451 sq mi)						
Denmark - Kingdom of Denmark	Copenhagen	5515575	43.094 km2(16.639 sa mi)(n 1)						

Probabilistic Google Search with SPROUT²

			SPROUT ²			0	
File Maintenance							
enlightenment philosophers European Co	untries + Comp	ose Qu	ery Query Result				
	enlightenment	Select	Probabilistic Model		European Countries	Select	
	Key	•	Certain		English short and		
	Image		Mutually exclusive		Capital		
	Description		Mutually exclusive		Population		
	Date Of Birth		Mutually exclusive	Ξ	Area		
	Place Of Birth		Independent				
	Place Of Death		Independent				
	Died		Mutually exclusive				
	Date Of Death		Mutually exclusive		J		
	Approximate C	onfider	nce (Relative Error)		= 0.001 🗸 Comp	oute Query	
			Currently configur	red j	oins		
Left		Jo	in		Right)
enlightenment philosophers.Place Of Birth		Ap	oproximate Equality Joi	n	enlightenmen	t philosophers.Place Of Death	
enlightenment philosophers.Place Of Death		Ap	oproximate Equality Joi	n	European Cou	untries. Capital	

🛃 SPROUT? 🛛 🗖 🗟 🛣									
File Maintenance									
enlightenment philosophers European Countries + Compose Query Query Result									
enlightenment philosophers.Key	enlightenment philosophers.Plac	enlightenment philosophers.Pla	967574.Capital	Confidence	ור				
Thomas	London	London	London	High Confidence					
Voltaire				High Confidence					
Denis Diderot	Langres, France	Paris, France	Paris	Medium Confidence					
Charles de Secondat, baron de	Bordeaux, France	Paris, France	Paris 0.61321735	Medium Confidence					
Wolfgang Amadeus Mozart	Salzburg, Austria	Vienna, Austria	Vienna	Low Confidence					
Voltaire	Paris, France	Paris	Paris	Low Confidence					
Baruch Spinoza	Amsterdam, Netherlands	Hague, Netherlands	Prague	Low Confidence					
Voltaire	Paris	Paris, France	Paris	Low Confidence					
Wolfgang Amadeus Mozart	Vienna, Austria	Vienna	Vienna	Low Confidence					
Voltaire	Paris, France	Paris, France	Amsterdam, The Hague	Low Confidence					
Baruch Spinoza	Amsterdam, Netherlands	Hague, Netherlands	Amsterdam, The Hague	Low Confidence					
Wolfgang Amadeus Mozart	Vienna, Austria	Vienna, Austria	Vienna	Low Confidence					
Denis Diderot	Langres, France	Paris, France	Amsterdam, The Hague	Low Confidence					
Charles de Secondat haron de	Rordeaux France	Darie France	Amsterdam The Haque	Low Confidence					
	Show	Database Query for Confidence Cor	mputation						

Factorized Databases

(2 * 100) + (3 * 100)

(2 * 100) + (3 * 100) = (2+3) * 100

(x and y) or (z and y) = (x or z) and y

where x, y, z are Boolean variables

$(R_1 \times S) \cup (R_2 \times S) = (R_1 \cup R_2) \times S$

where \times is Cartesian product and \bigcup is union; R_1, R_2, S are relations

Factorization in Relational Algebra

[BSc DB Course]



where \times is Cartesian product and \bigcup is union; R_1, R_2, S are relations

All previous identities are instances of the same distributivity law of an algebraic structure called the ring with sum-product operations:

Identity	Sum	Product	Domain
(a * b) + (c * b) = (a + c) * b	+	*	Reals
(x and y) or (z and y) = (x or z) and y	\vee	\wedge	Booleans
$(R_1 \times S) \cup (R_2 \times S) = (R_1 \cup R_2) \times S$	U	×	Relations

Factorization reduces redundant computation

Factorization reduces redundant computation

"The ability to simplify means to eliminate the unnecessary so that the necessary may speak." – Hans Hofmann

Key Advantage of Factorization



Key Advantage of Factorization



Factorized form (left):

- Lossless representation
- More compact
- Supports computation:
 - Database queries
 - Matrix computation
 - Model training

Key Advantage of Factorization



Factorized form (left):

- Lossless representation
- More compact
- Supports computation:
 - Database queries
 - Matrix computation
 - Model training

Example:

Compute Count($R \times S$) as Count(R) * Count(S)

State of Affairs in Learning over Relational Data



Factorized Learning over Relational Data



Factorized computation drastically improves the time and accuracy of model training over relational data

Factorization can Achieve 1000x Speedup



Relation	Size on Disk (CSV)
Inventory	2 GB
Items	129 KB
Stores	139 KB
Demographics	161 KB
Weather	33 MB
Join	23GB

Train a linear regression model to predict inventory given all features

	Time	Size	
Database	_	2.1 GB	
Join Relations	152.06 secs	23 GB	
Export Data	351.76 secs	23 GB	
Query batch	-	_	
Learn	12,738.31 secs	-	
Total time	13,242.13 secs		

Train a linear regression model to predict inventory given all features

	$PostgreSQL{+}TensorFlow$		Our system	
	Time	Size	Time	Size
Database	_	2.1 GB	_	2.1 GB
Join Relations	152.06 secs	23 GB	-	_
Export Data	351.76 secs	23 GB	-	_
Query batch	_	_	6.08 secs	37 KB
Learn	12,738.31 secs	_	0.05 secs	_
Total time	13,242.13 secs		6.13 secs	

 $2,160 \times$ faster while being more accurate (RMSE on 2% test data)

Similar Speedups Observed for other Datasets & Models

Factorization can lead to 1000x Better Numerical Accuracy

Problem: Decompose large matrices defined by relational data

- QR decomposition
- Singular Value Decomposition
- Principal Component Analysis
- Low-rank matrix decomposition



Factorization \Rightarrow less (redundant) computation

• fewer square roots, divisions, and multiplications

Why are Speedups & Numerical Accuracy Useful?

- Less energy to achieve the same task as competing systems
- Commodity machines can now perform the task previously done on more powerful machines or many more machines
- We can train more models within the same time budget
- Maintain prediction models fresh on a second/minute/hour basis instead of every day/week
- Numerically unstable algorithms are of no use for critical tasks that require precise computation



Systems and Theory for Factorized Computation

- Publicly available, open-source systems: LMFAO & F-IVM
 - Influenced the design of commercial system RelationalAI
- Impact in database theory: test-of-time award
 - We answered questions on the optimality and computational complexity of factorization
 - Influenced graph database design, static and dynamic query evaluation, provenance, factorized machine learning
- Summer of 2022: Workshop in Zurich dedicated to factorized computation

Going More Succinct than Factorization

- Subject of on-going work by several research groups
- More succinct \Rightarrow subsequent computation not efficient

Going More Succinct than Factorization

- Subject of on-going work by several research groups
- More succinct \Rightarrow subsequent computation not efficient

"Everything should be made as simple as possible, but not simpler." – Sessions paraphrasing Einstein

Acknowledgments

FDB team, in particular:



Ahmet Amir Dorde Haozhe Jakub Max Milos Nils Robert

LogicBlox & RelationalAI teams, in particular:



ElSeidy Henrik Hung Long Mahmoud Molham Niko

Thank you!