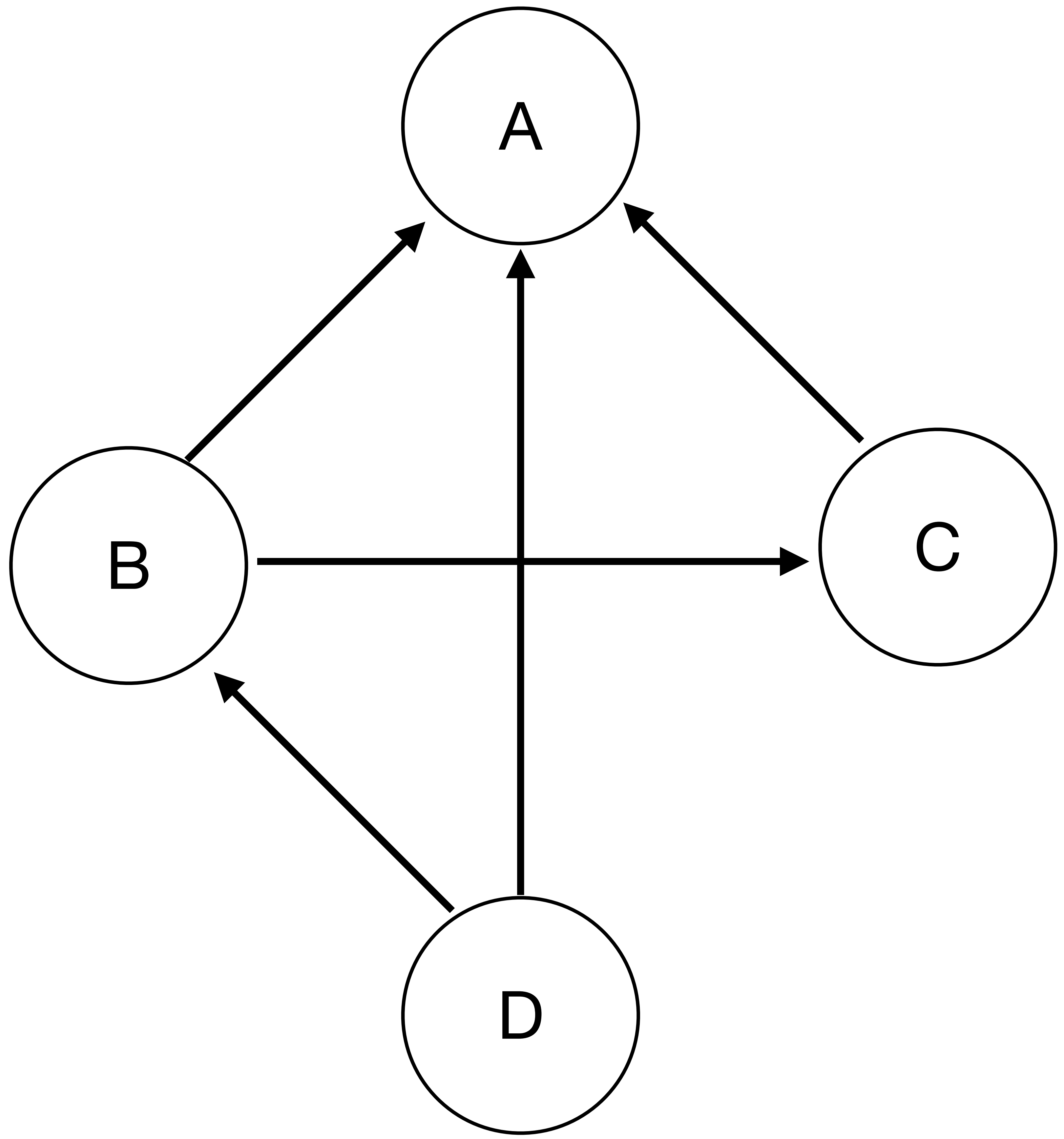
Advanced Data Management (CSCI 490/680)

Visualization and Databases

Dr. David Koop

Graphs

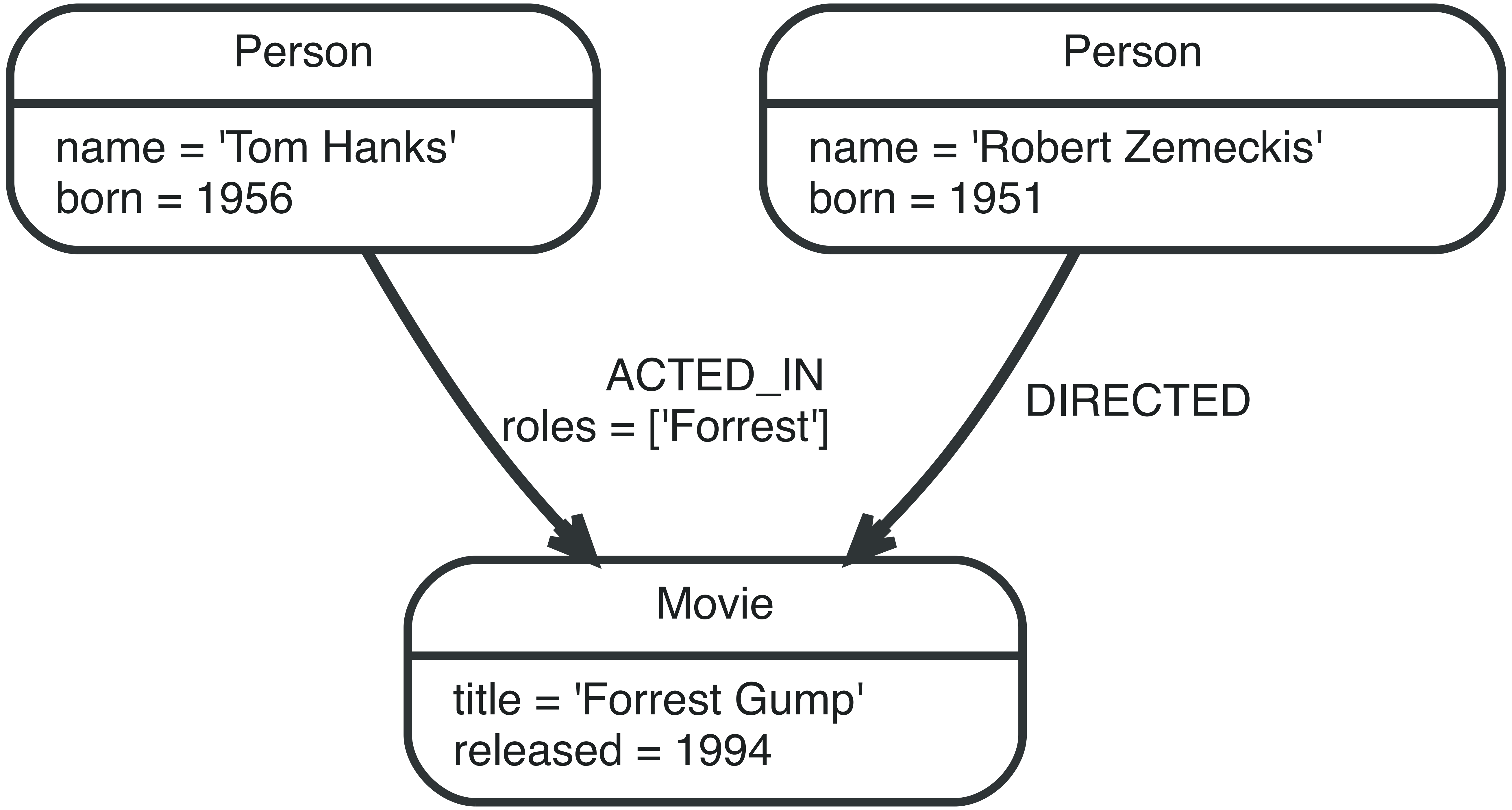


- In computing, a **graph** is an abstract **data structure** that represents set objects and their relationships as **vertices** and **edges/links**, and supports a number of graph-related **operations**
- Objects (nodes): `{A,B,C,D}`
- Relationships (edges): `{(D,B),(D,A),(B,C),(B,A),(C,A)}`
- Operation: shortest path from `D` to `A`

[K. Salama, 2016]

Graphs with Properties

- Each vertex or edge may have properties associated with it
- May include identifiers or classes



[neo4j]

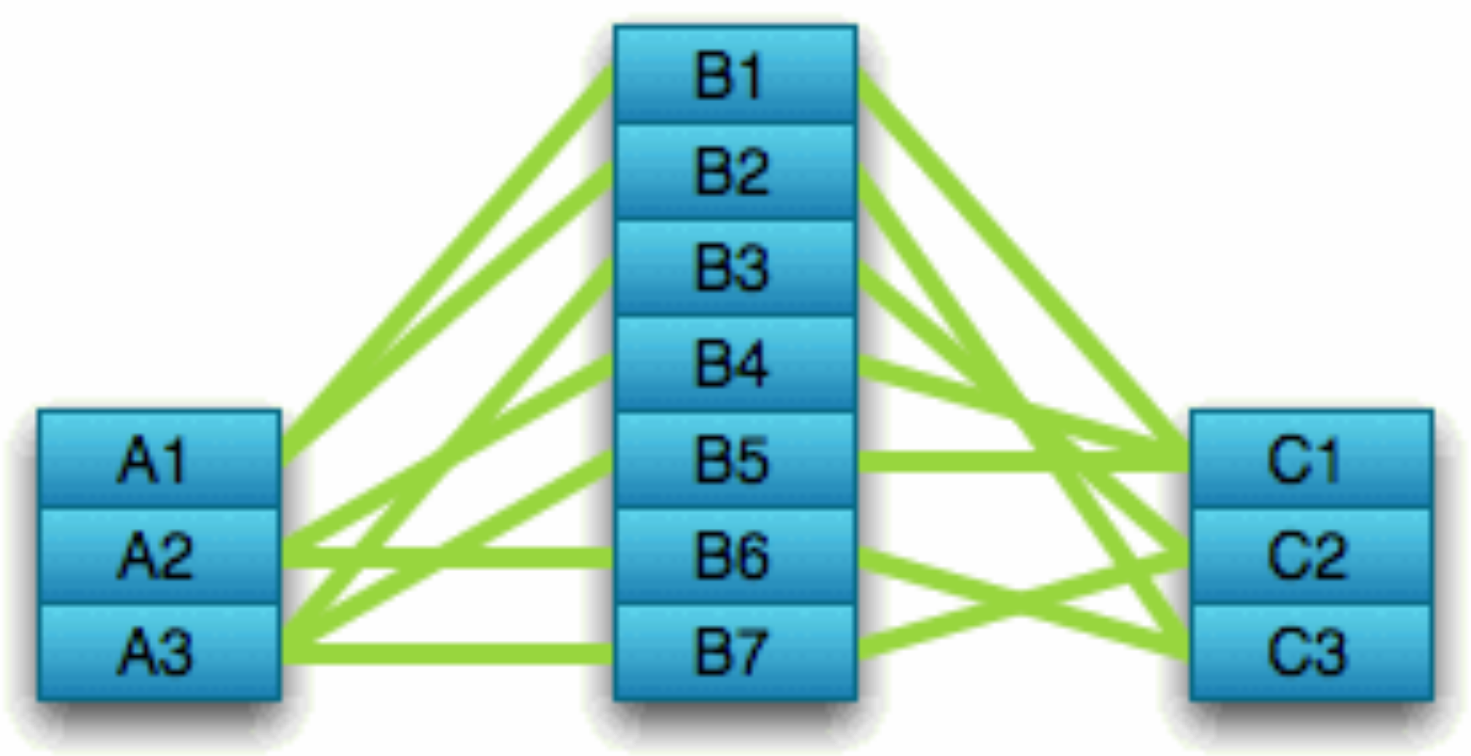
What is a Graph Database?

- A database with an explicit graph structure
- Each node knows its adjacent nodes
- As the number of nodes increases, the cost of a local step (or hop) remains the same
- Plus an Index for lookups

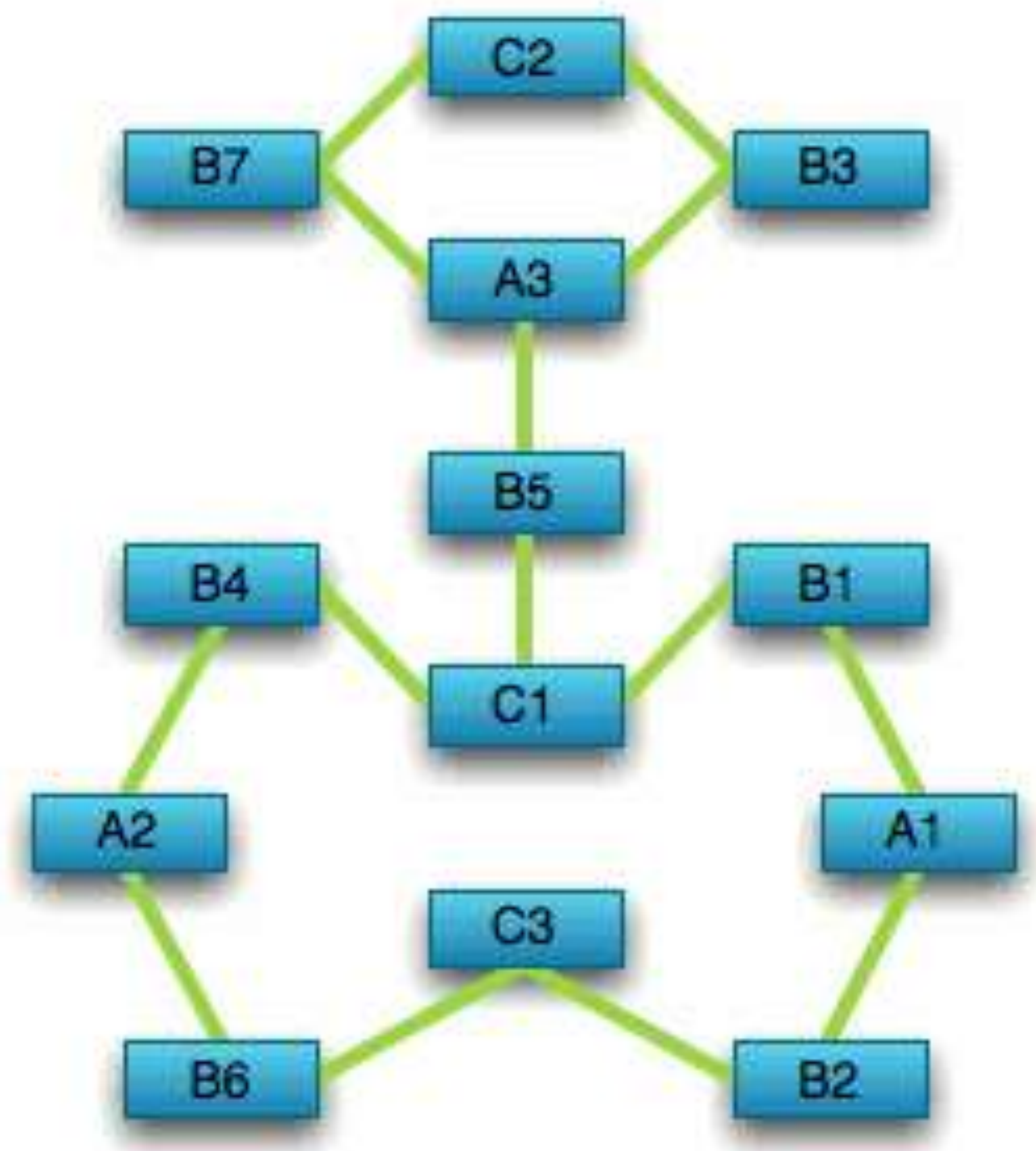
[M. De Marzi, 2012]

Graph Databases Compared to Relational Databases

Optimized for aggregation



Optimized for connections



[M. De Marzi, 2012]

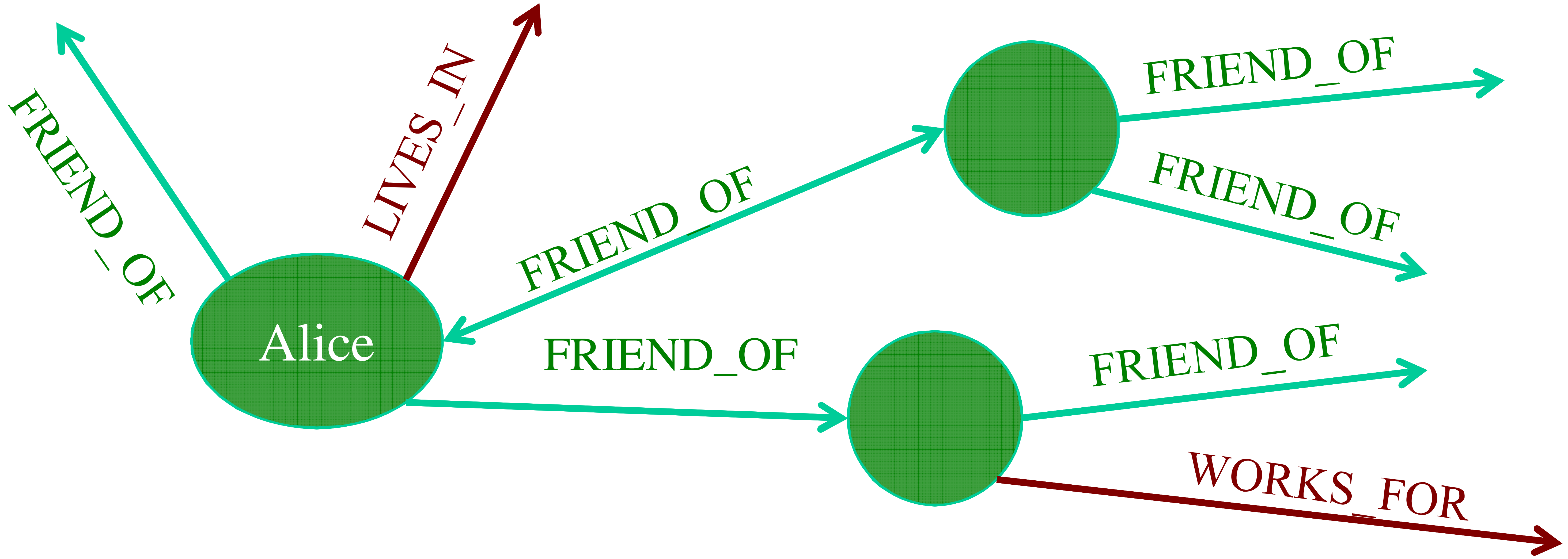
Graph Databases Compared to Relational Databases

- Relational Databases (querying is through joins)
  - In effect, the join operation forms a graph that is **dynamically constructed** as one table is linked to another table.
  - Must be inferred through a series of **index-intensive** operations
- Graph Databases (querying is through traversal paths)
  - There is no explicit join operation because vertices maintain **direct references** to their adjacent edges
  - Structures are "**hard-wired**", not computed at query time

[Rodriguez & Neubauer via Lembo & Rosati]

Example: Friend of Friends Query

- Relational:
- ```
SELECT p1.Person AS PERSON, p2.Person AS FRIEND_OF_FRIEND FROM
PersonFriend pf1 JOIN Person p1 ON
  pf1.PersonID = p1.ID JOIN PersonFriend pf2 ON
  pf2.PersonID = pf1.FriendID JOIN Person p2 ON
  pf2.FriendID = p2.ID
WHERE p1.Person = 'Alice' AND pf2.FriendID <> p1.ID
```
- Graph:

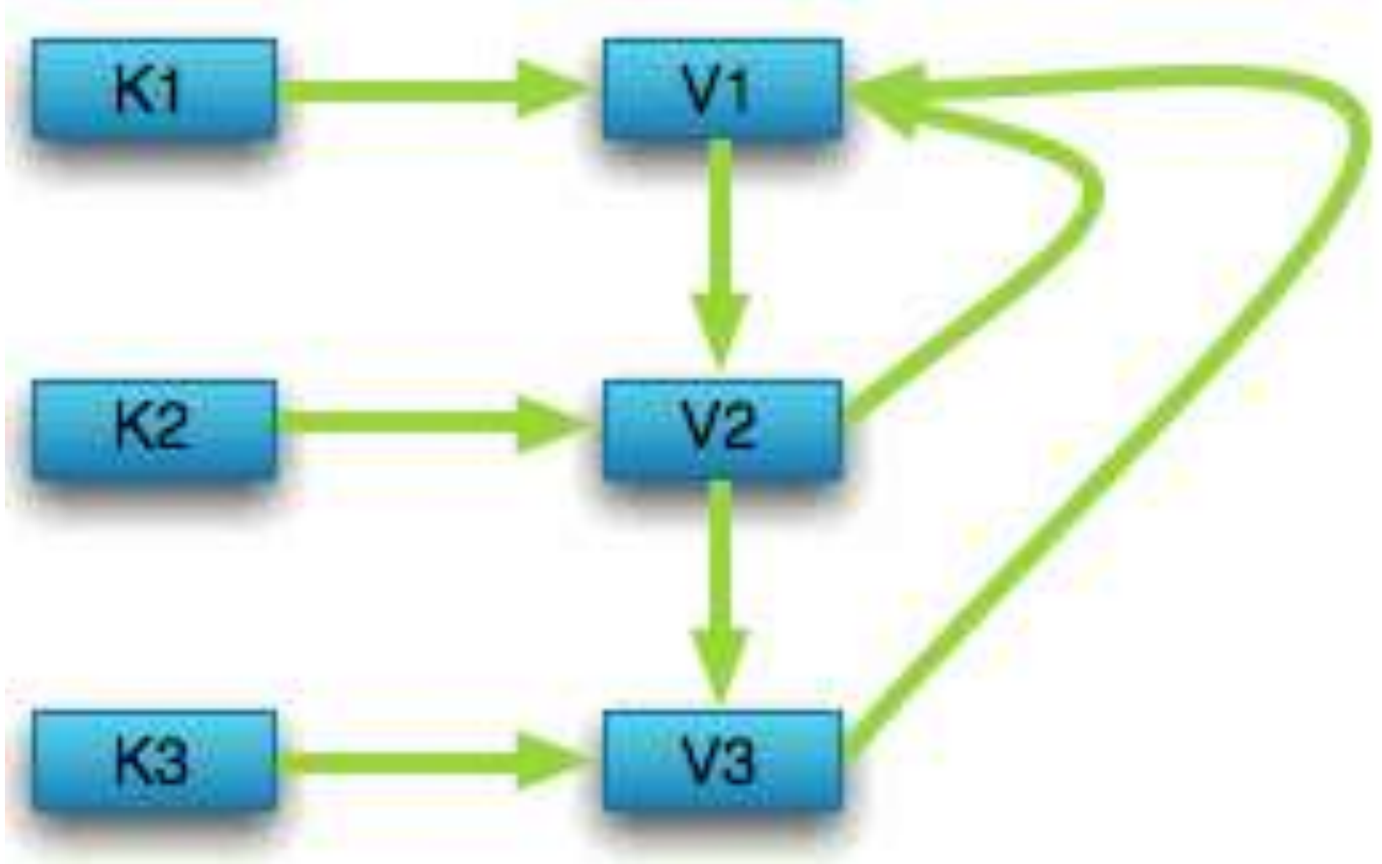



[Lembo & Rosati]
```

Graph Databases Compared to Key-Value Stores

Optimized for simple look-ups

Optimized for traversing connected data

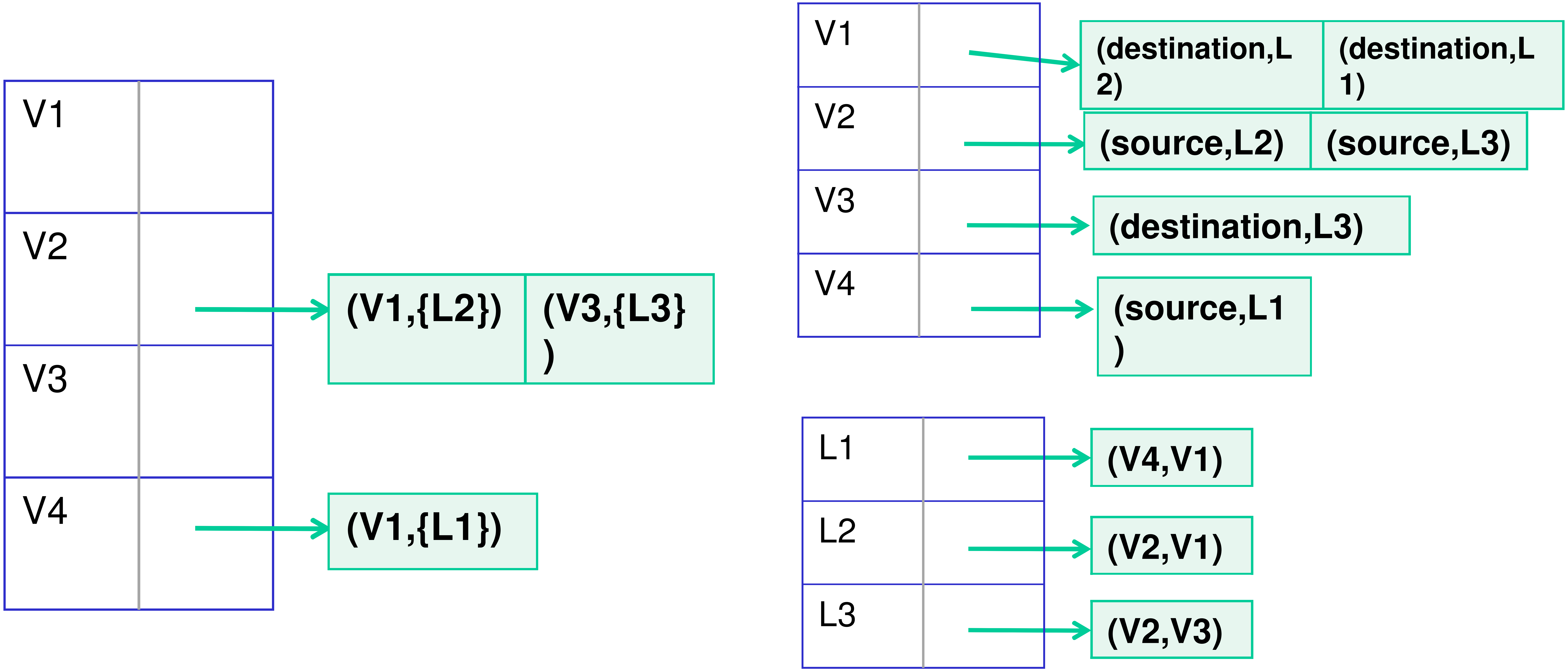


[M. De Marzi, 2012]

Storing and Traversing Graphs

- Storage:
  - Adjacency List: nodes store their neighbors
  - Incidence List: nodes store edges and edges store incident nodes
  - Adjacency Matrix: adjacency list in matrix form (rows & cols are nodes)
  - Incidence Matrix: rows are vertices, columns are **edges**
- Traversal:
  - Breadth-first Search
  - Depth-first Search

Adjacency List vs. Incidence List



[Acosta et al.]

Adjacency Matrix vs. Incidence Matrix

	V1	V2	V3	V4
V1				
V2	{L2}		{L3}	
V3				
V4	{L1}			

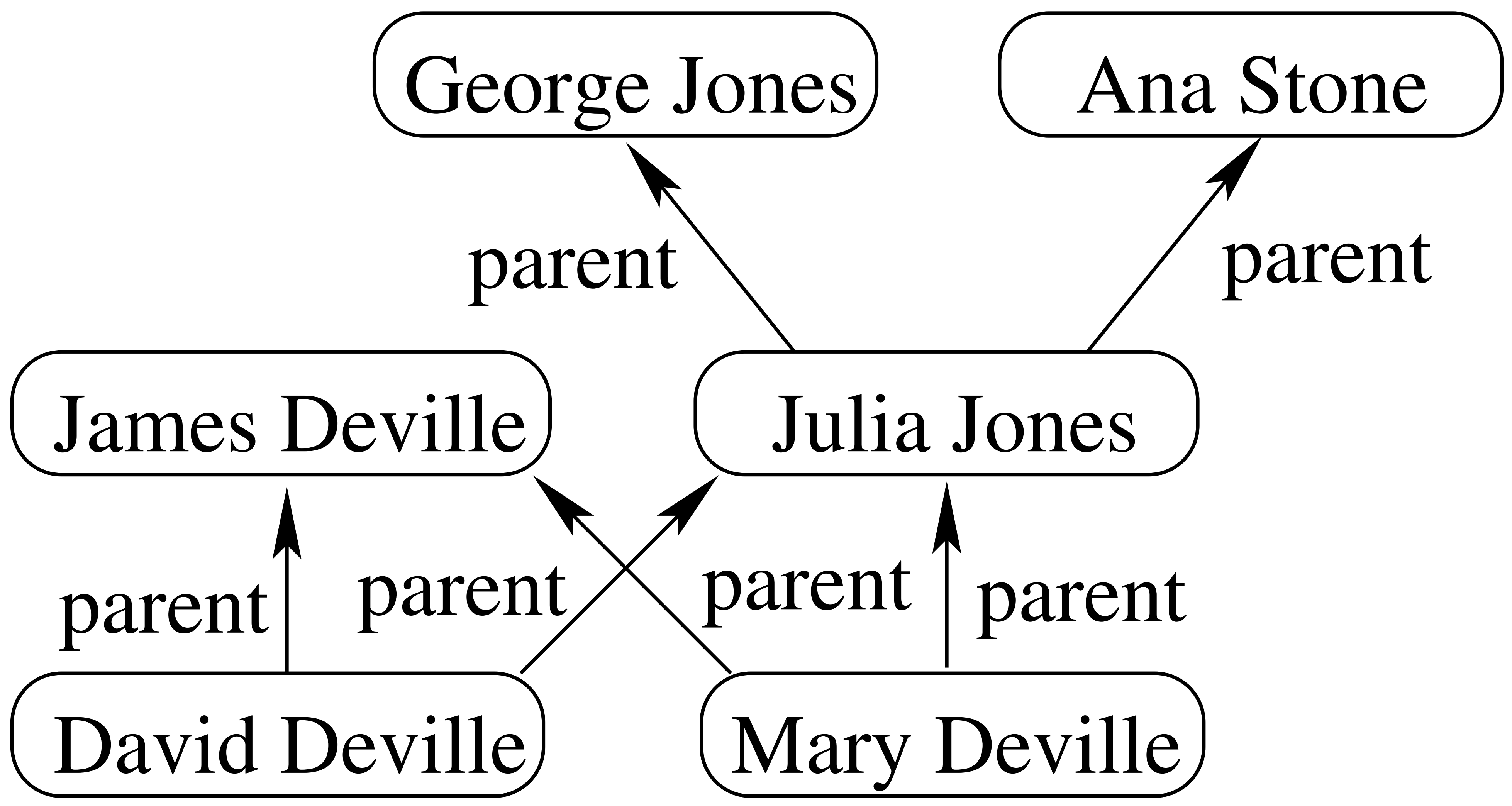
	L1	L2	L3
V1	destination	destination	
V2		source	source
V3			destination
V4	source		

[Acosta et al.]

Graph Models: Relational Model

NAME	LASTNAME
George	Jones
Ana	Stone
Julia	Jones
James	Deville
David	Deville
Mary	Deville

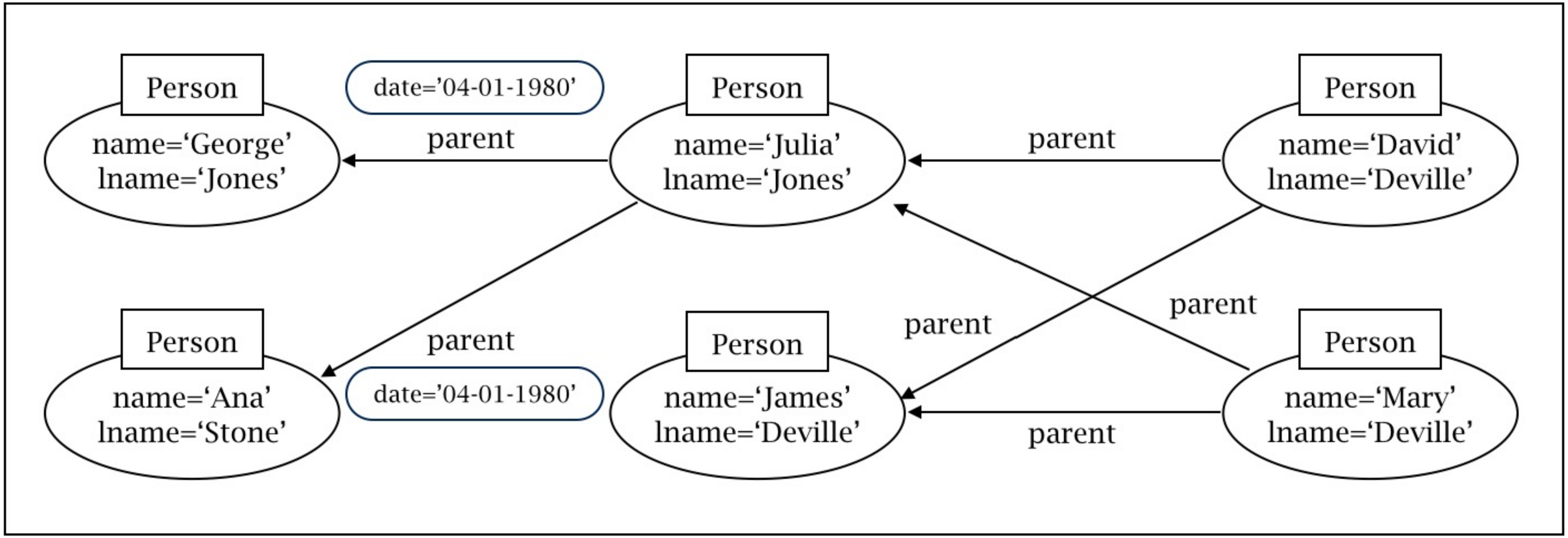
PERSON	PARENT
Julia	George
Julia	Ana
David	James
David	Julia
Mary	James
Mary	Julia



[R. Angles and C. Gutierrez, 2017]

Property Graph Model (Cypher in neo4j)

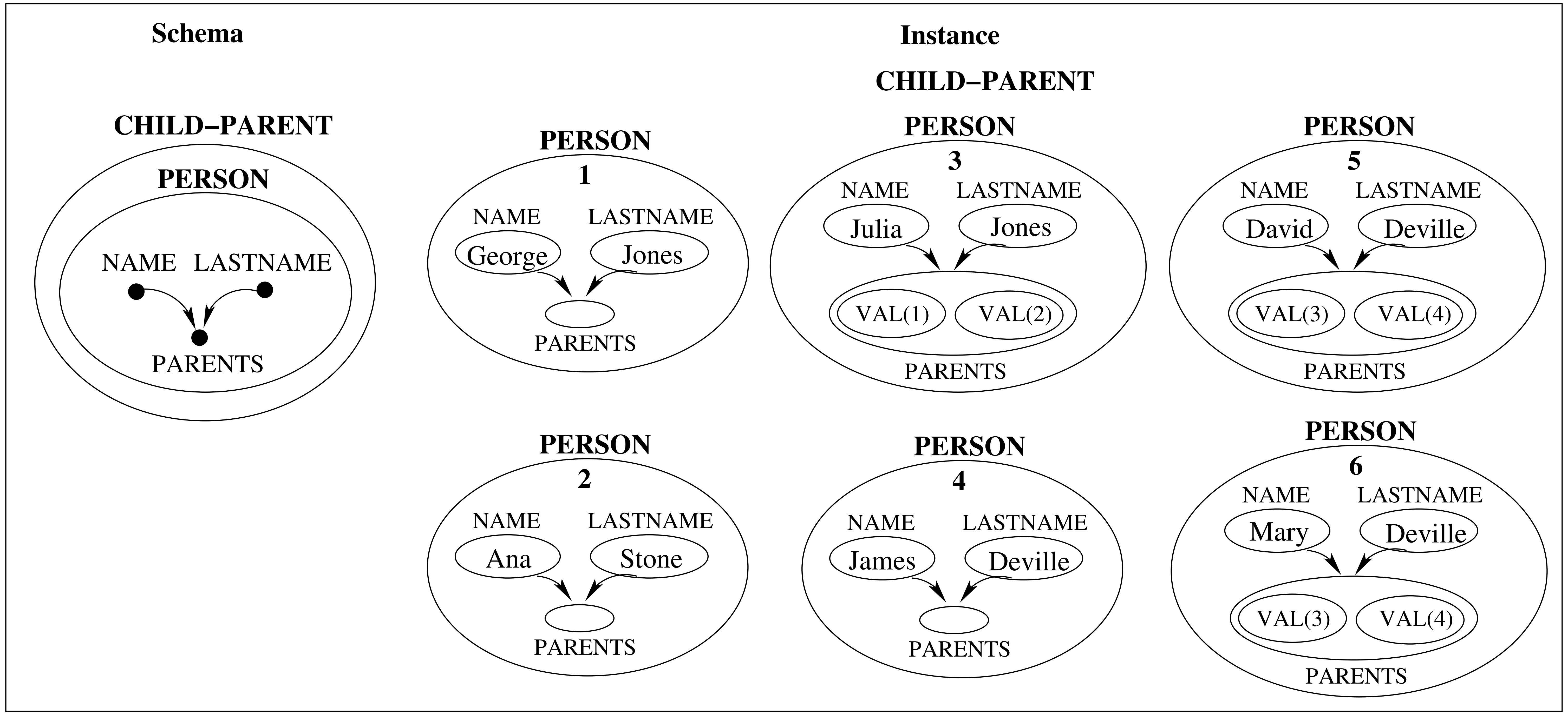
- Directed, labelled, attributed multigraph
- Properties are **key/value pairs** that represent metadata for nodes and edges



[R. Angles and C. Gutierrez, 2017]

Hypergraph Model (Groovy)

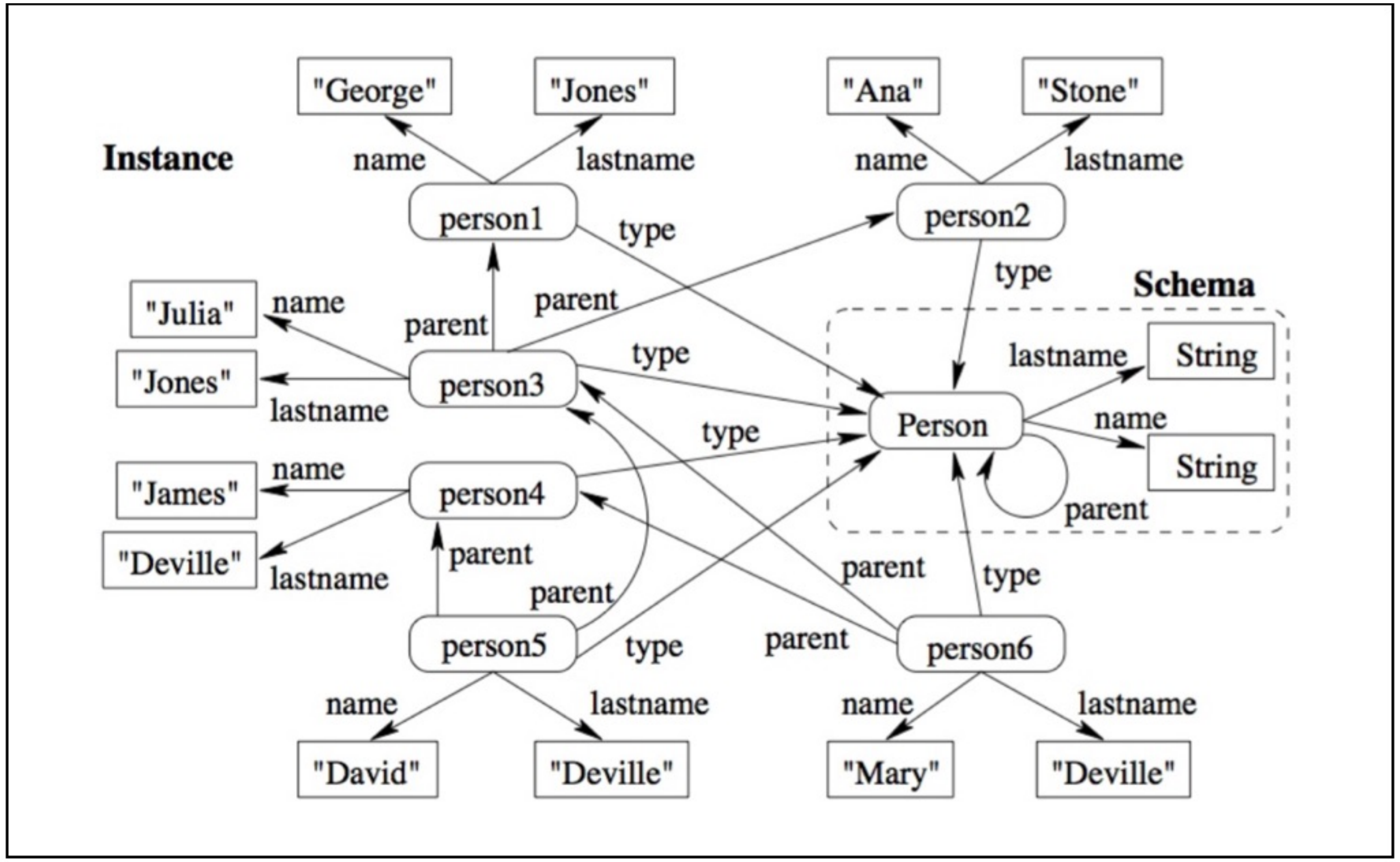
- Notion of edge is extended to **hyperedge**, which relates an arbitrary set of nodes
- Hypergraphs allow the definition of complex objects (undirected), functional dependencies (directed), object-ID and (multiple) structural inheritance



[R. Angles and C. Gutierrez, 2017]

RDF (Triple) Model

- Interconnect resources in an extensible way using graph-like structure for data
- Schema and instance are **mixed** together
- SPAQL to query
- Semantic web



[R. Angles and C. Gutierrez, 2017]

Graph Query Languages: Cypher

- Implemented by neo4j system
- Expresses reachability queries via path expressions
  - `p = (a)-[:knows*]->(b)`: nodes from `a` to `b` following `knows` edges
- ```
  START x=node:person(name="John")
  MATCH (x)-[:friend]->(y)
  RETURN y.name
  ```

[R. Angles and C. Gutierrez, 2017]
  ```

Graph Query Languages: SPARQL (RDF)

- Uses SELECT-FROM-WHERE pattern like SQL
-
```
SELECT ?N
FROM <http://example.org/data.rdf>
WHERE { ?X rdf:type voc:Person . ?X voc:name ?N }
```

[R. Angles and C. Gutierrez, 2017]

Graph DBMS Building Blocks

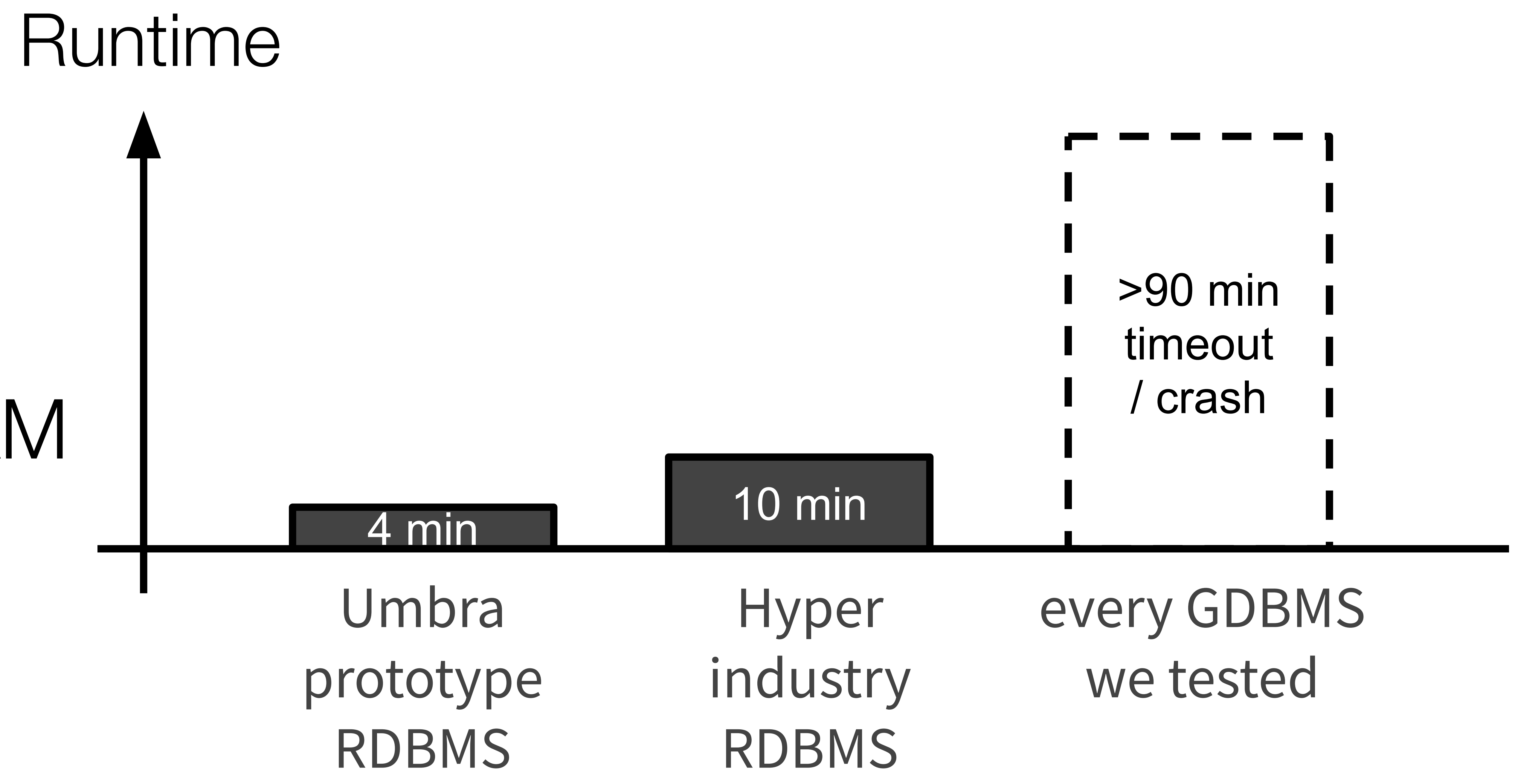
- property graph data model
- graph query language
- graph visualization

- subgraph matching
- relational queries
- path queries
- stored procedures

[P. Boncz, 2022]

Graph DBMS Problems

- performance
  - Slow loading speeds
  - Query speeds over magnitude slower than RDBMS
- scalability
  - Low datasize limit, typically << RAM
  - Little benefit from parallelism
- reliability
  - Loads never terminate
  - Query run out of memory or crash
  - Bugs



[P. Boncz, 2022]

Quiz Wednesday

- Read Nanocubes paper
- Quiz at the beginning of class

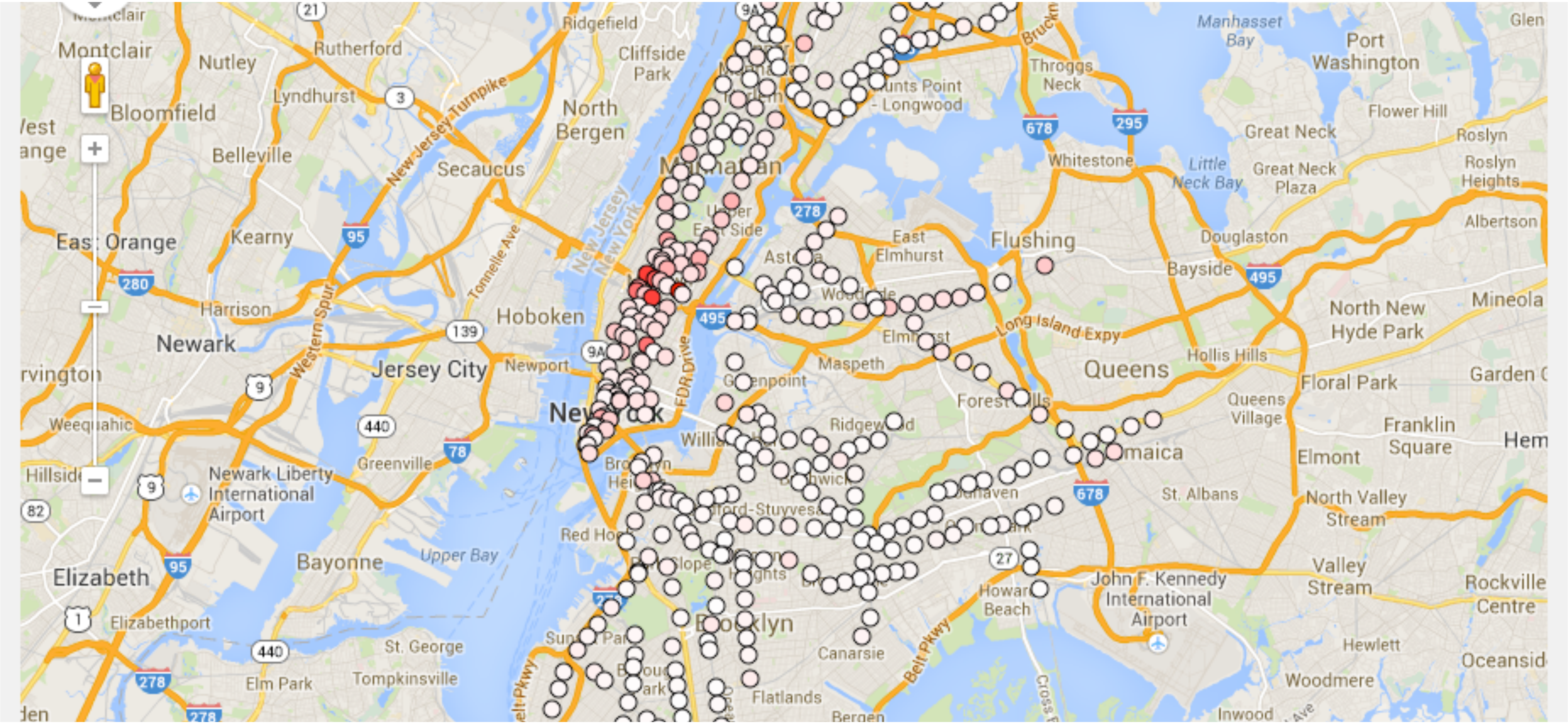
Data Exploration through Visualization

Transportation Data - NYC MTA

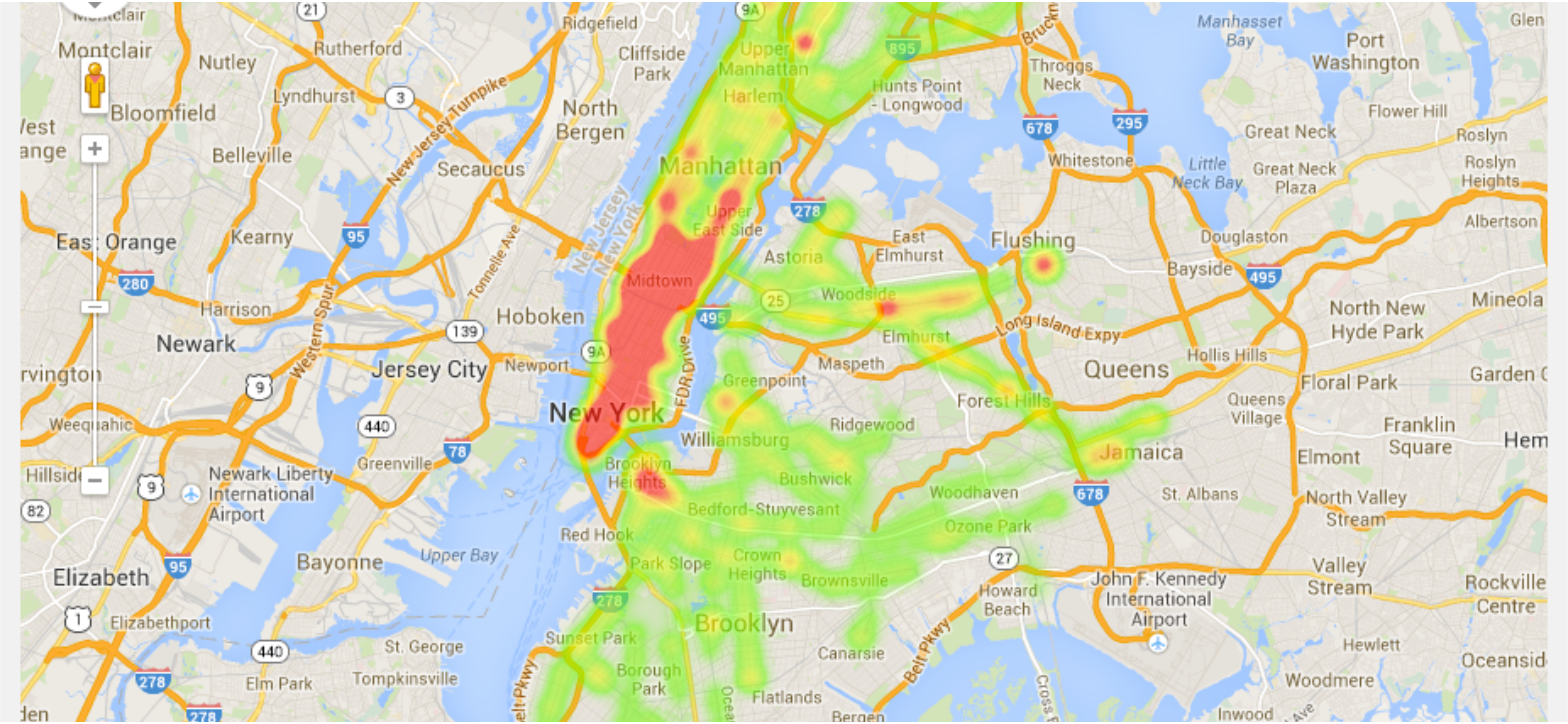
MTA Fare Data Exploration

	REMOTE	STATION	FF	SEN/DIS	7-D AFAS UNL	D AFAS/RMF I	JOINT RR TKT	7-D UNL	30-D UNL
1	R011	42ND STREET & 8TH AVENUE	00228985	00008471	00000441	00001455	00000134	00033341	00071255
2	R170	14TH STREET-UNION SQUARE	00224603	00011051	00000827	00003026	00000660	00089367	00199841
3	R046	42ND STREET & GRAND CENTRAL	00207758	00007908	00000323	00001183	00003001	00040759	00096613
4	R012	34TH STREET & 8TH AVENUE	00188311	00006490	00000498	00001279	00003622	00035527	00067483
5	R293	34TH STREET - PENN STATION	00168768	00006155	00000523	00001065	00005031	00030645	00054376
6	R033	42ND STREET/TIMES SQUARE	00159382	00005945	00000378	00001205	00000690	00058931	00078644
7	R022	34TH STREET & 6TH AVENUE	00156008	00006276	00000487	00001543	00000712	00058910	00110466
8	R084	59TH STREET/COLUMBUS CIRCLE	00155262	00009484	00000589	00002071	00000542	00053397	00113966
9	R020	47-50 STREETS/ROCKEFELLER	00143500	00006402	00000384	00001159	00000723	00037978	00090745
10	R179	86TH STREET-LEXINGTON AVE	00142169	00010367	00000470	00001839	00000271	00050328	00125250
11	R023	34TH STREET & 6TH AVENUE	00134052	00005005	00000348	00001112	00000649	00031531	00075040
12	R029	PARK PLACE	00121614	00004311	00000287	00000931	00000792	00025404	00065362
13	R047	42ND STREET & GRAND CENTRAL	00100742	00004273	00000185	00000704	00001241	00022808	00068216

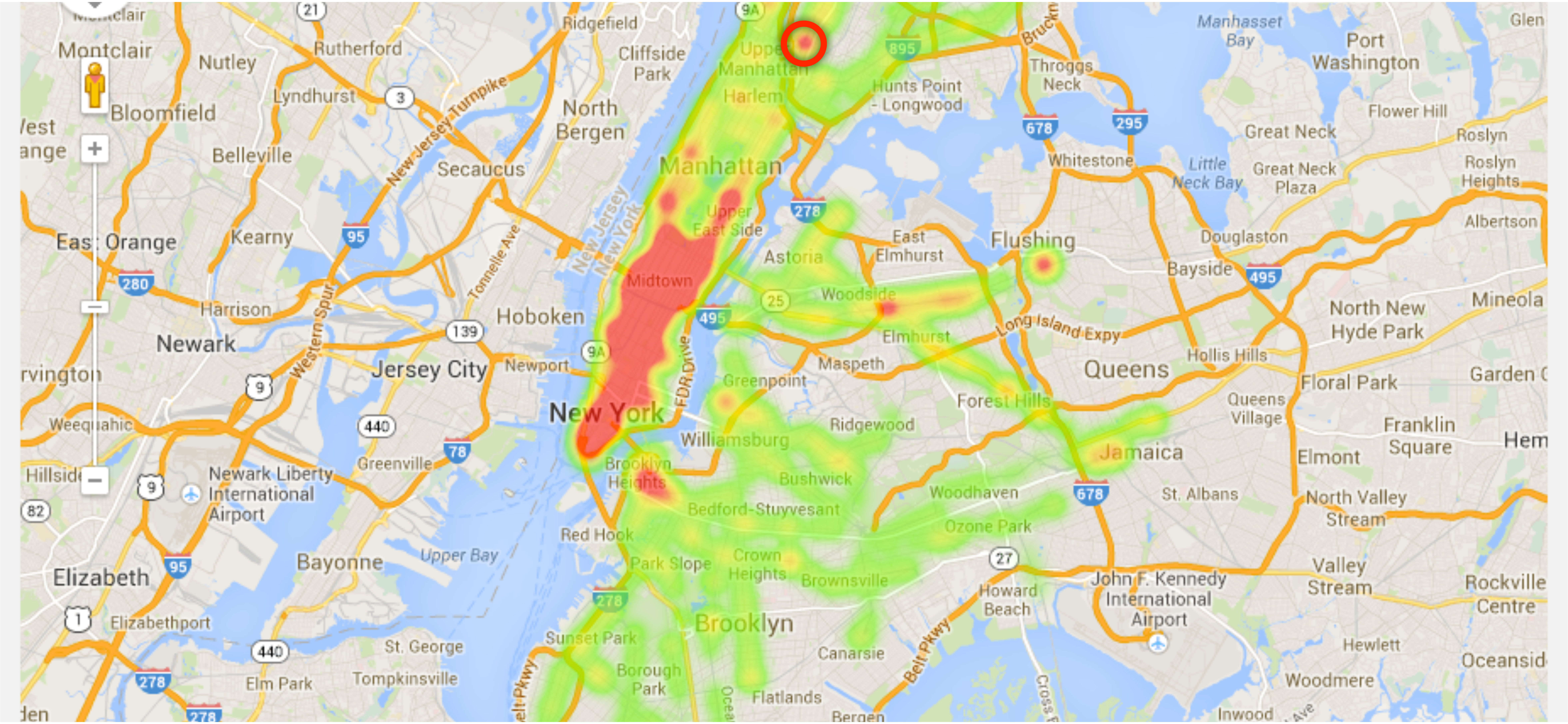
MTA Fare Data Exploration



MTA Fare Data Exploration

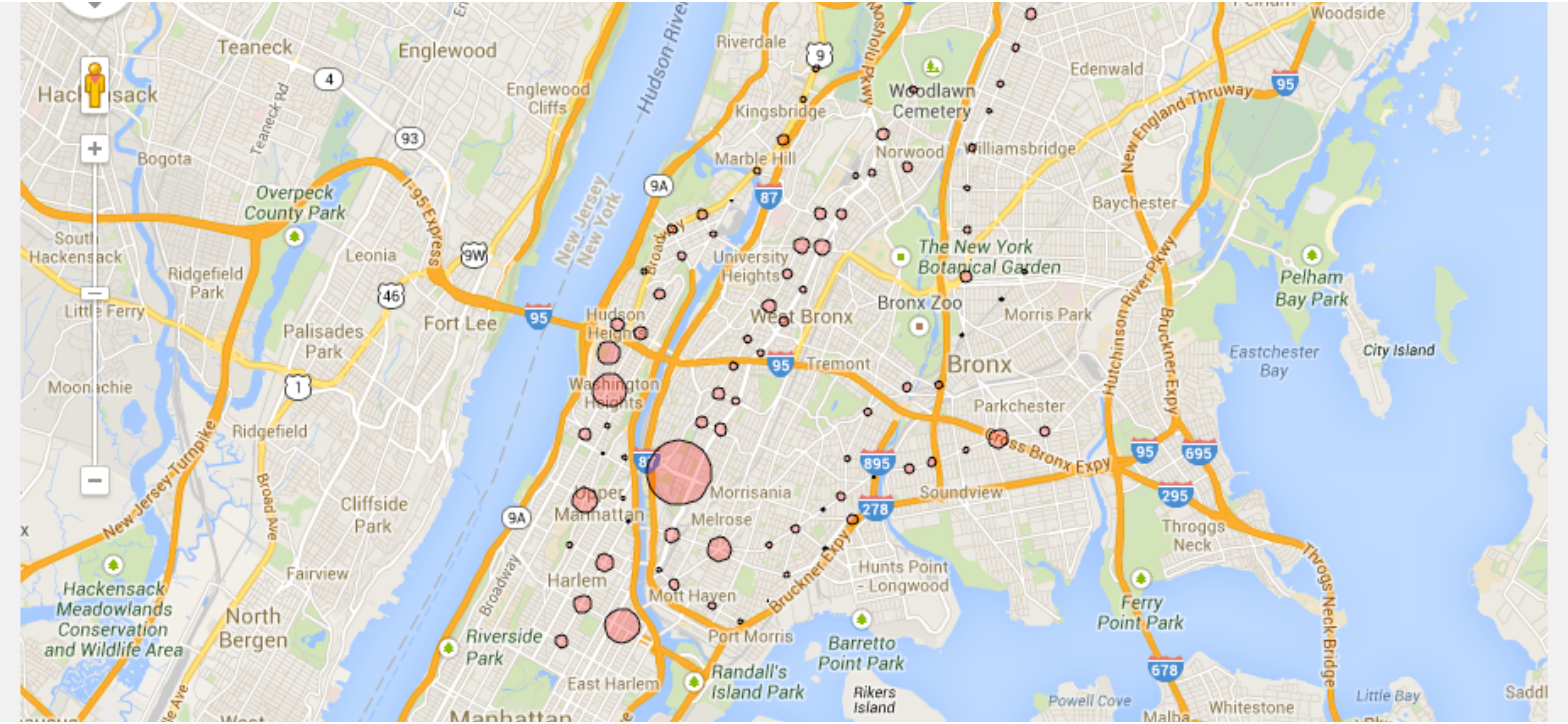


MTA Fare Data Exploration

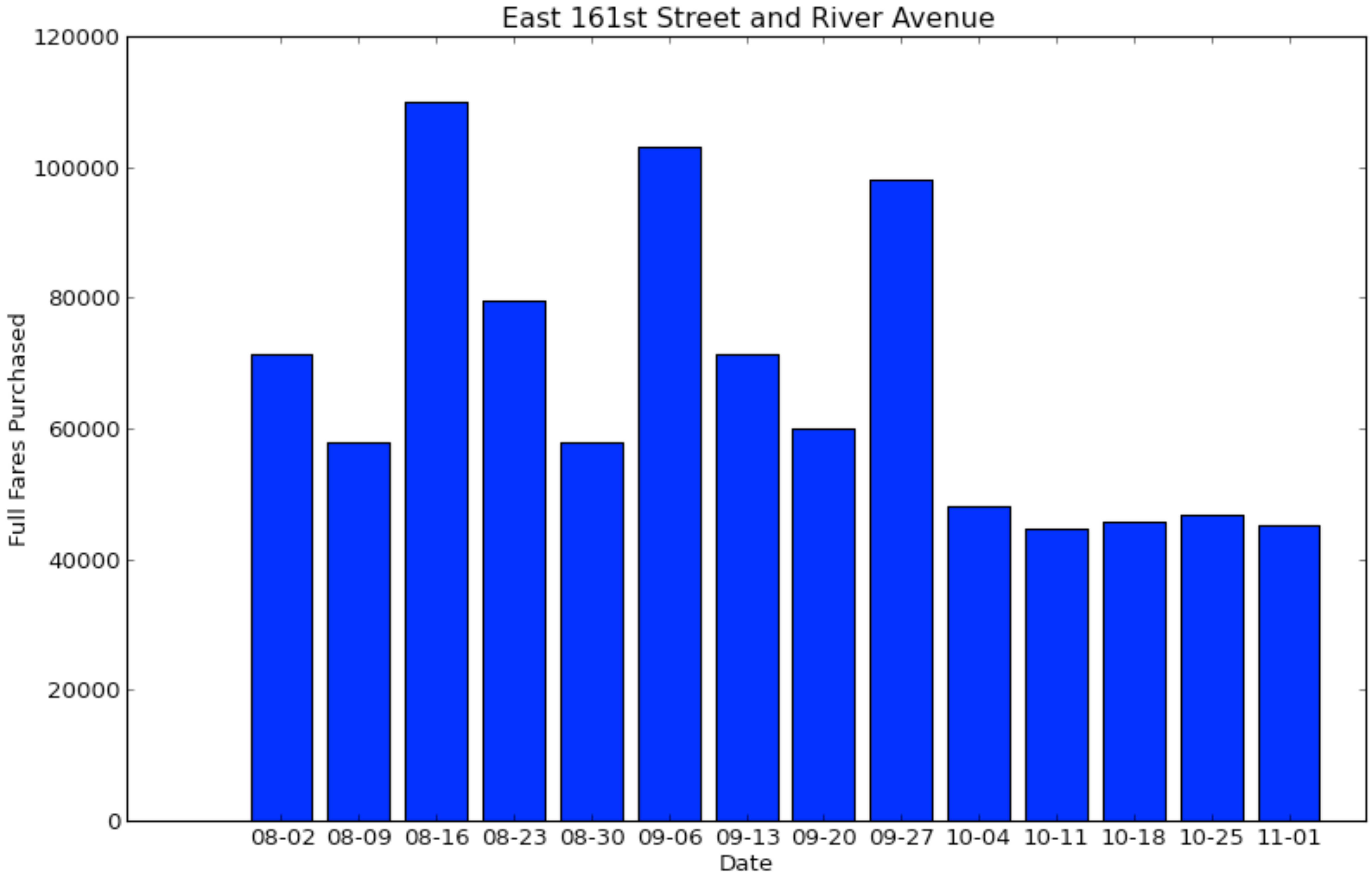


Northern Illinois University

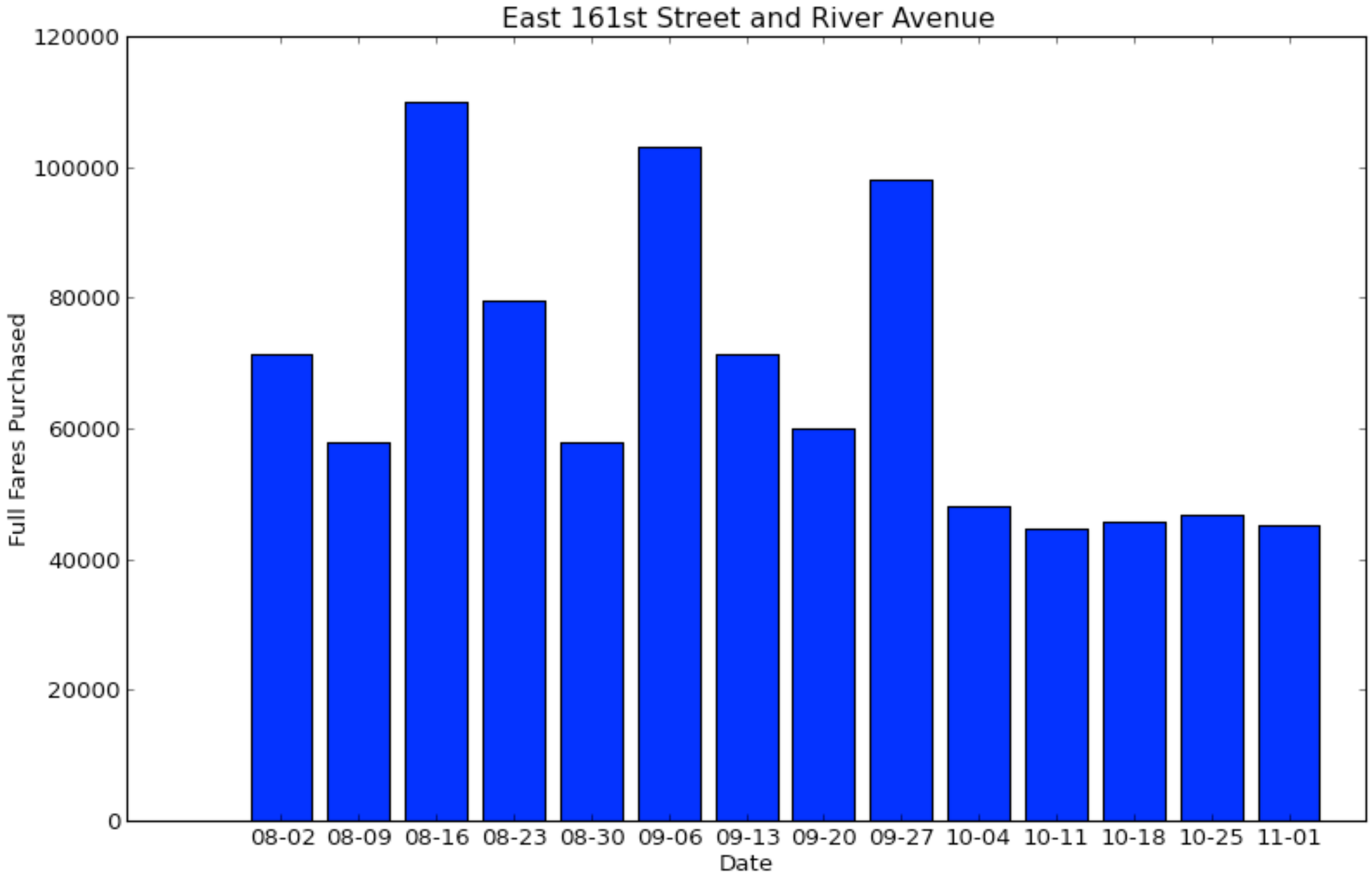
MTA Fare Data Exploration



MTA Fare Data Exploration



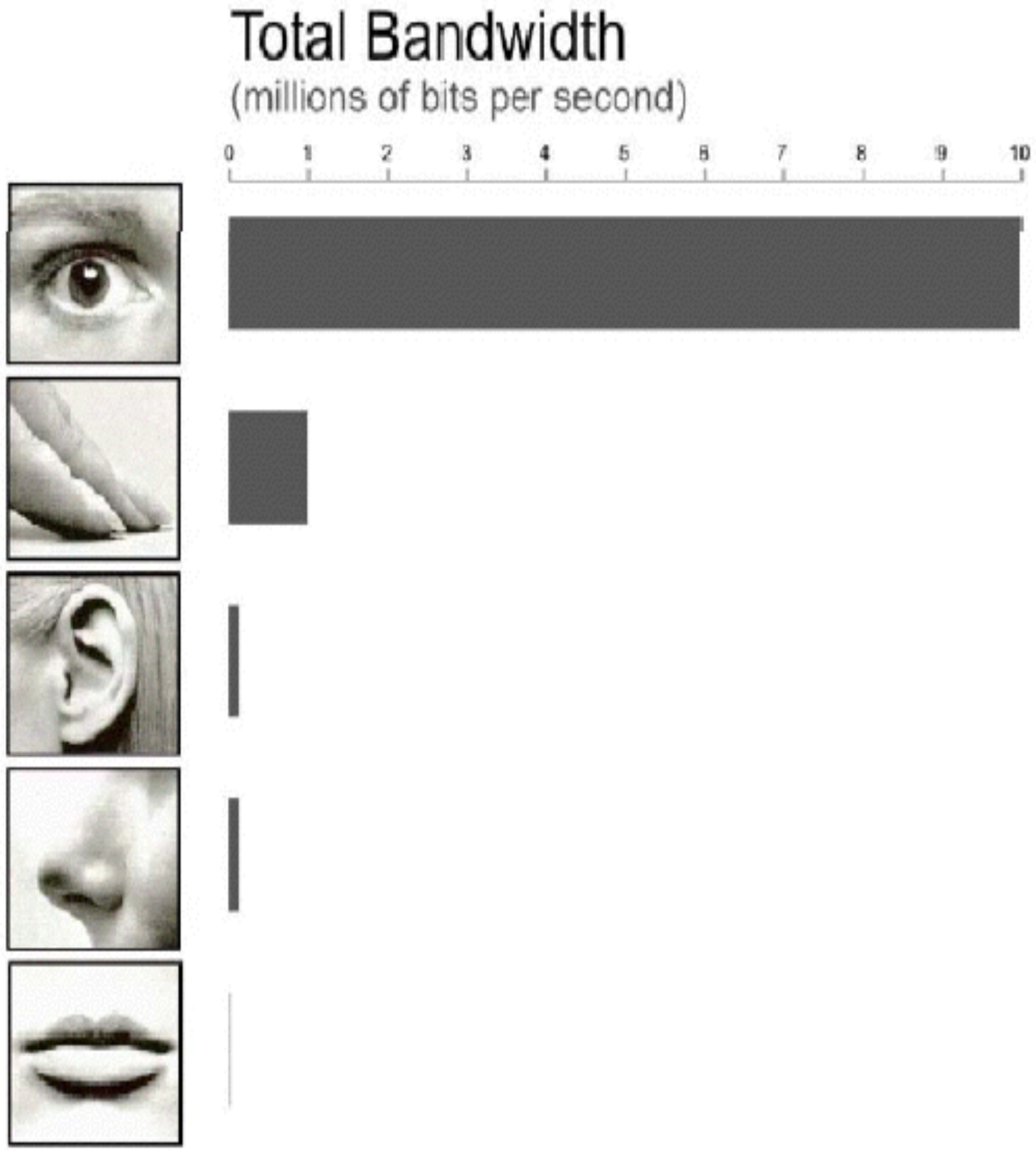
MTA Fare Data Exploration



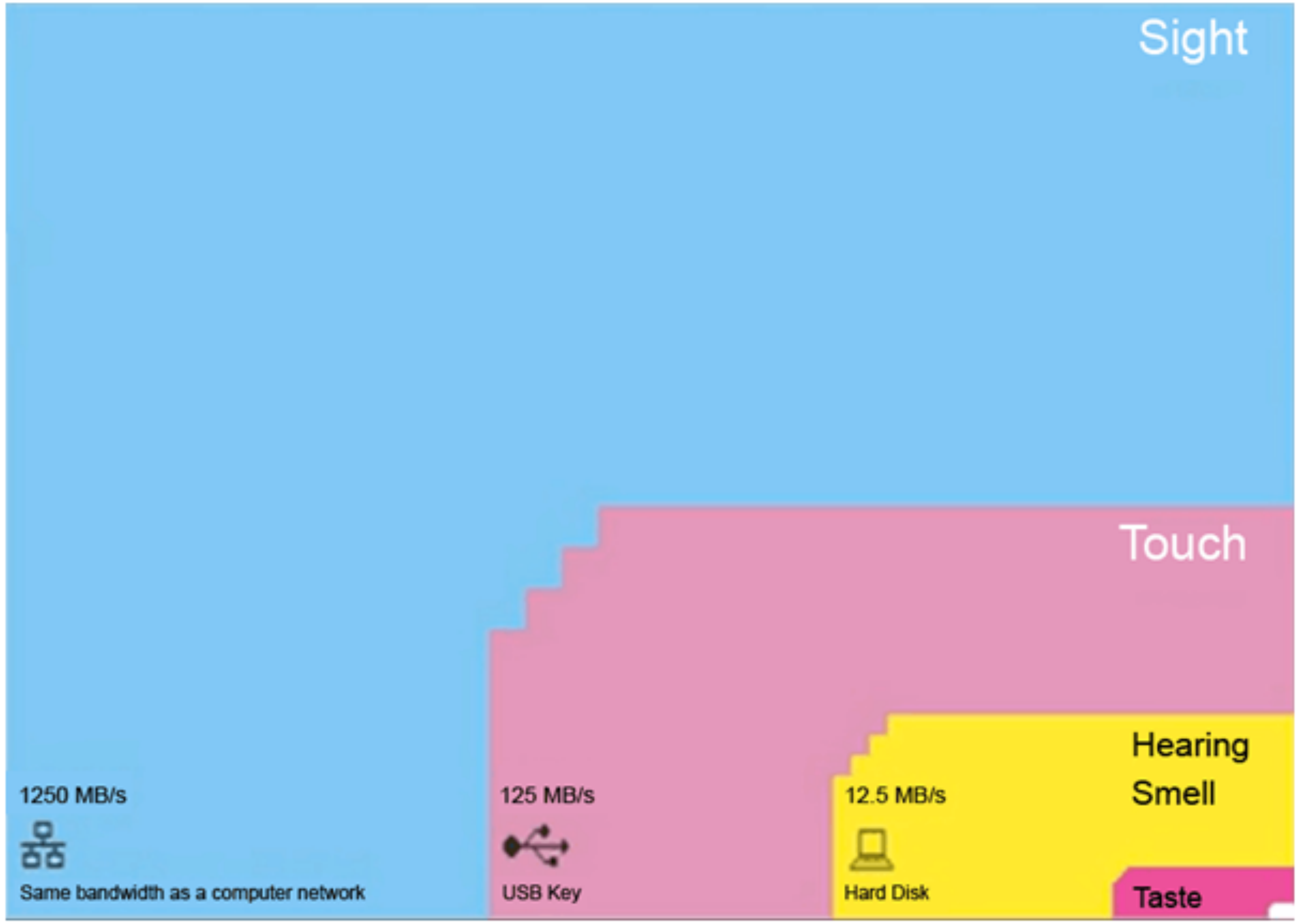
Definition of Visualization

"Computer-based visualization systems provide visual representations of datasets designed to help people carry out tasks more effectively" — T. Munzner

Why do we visualize data?



[via A. Lex]



[T. Nørretranders]

Why Visual?

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

[F. J. Anscombe]

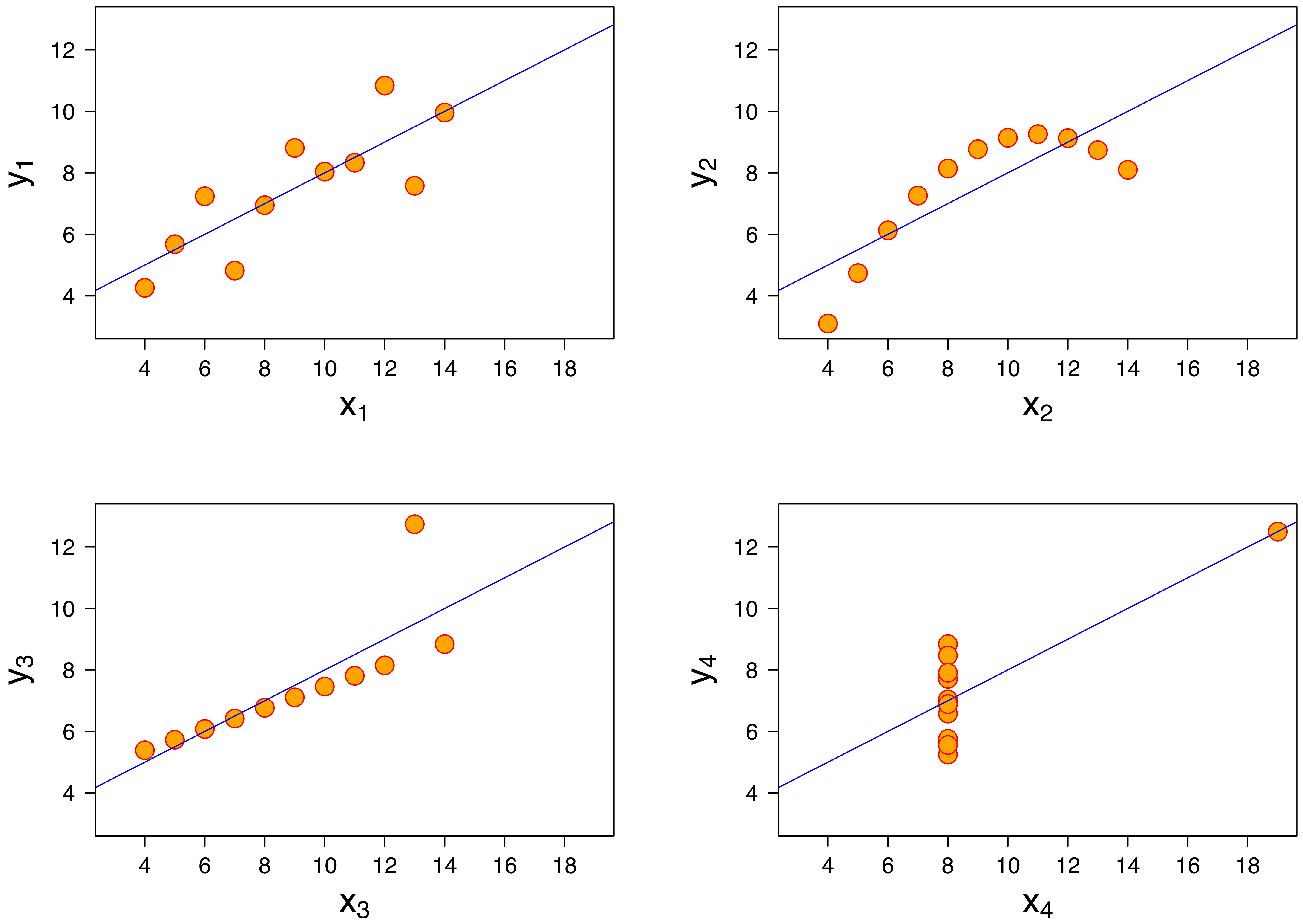
Why Visual?

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

Mean of x	9
Variance of x	11
Mean of y	7.50
Variance of y	4.122
Correlation	0.816

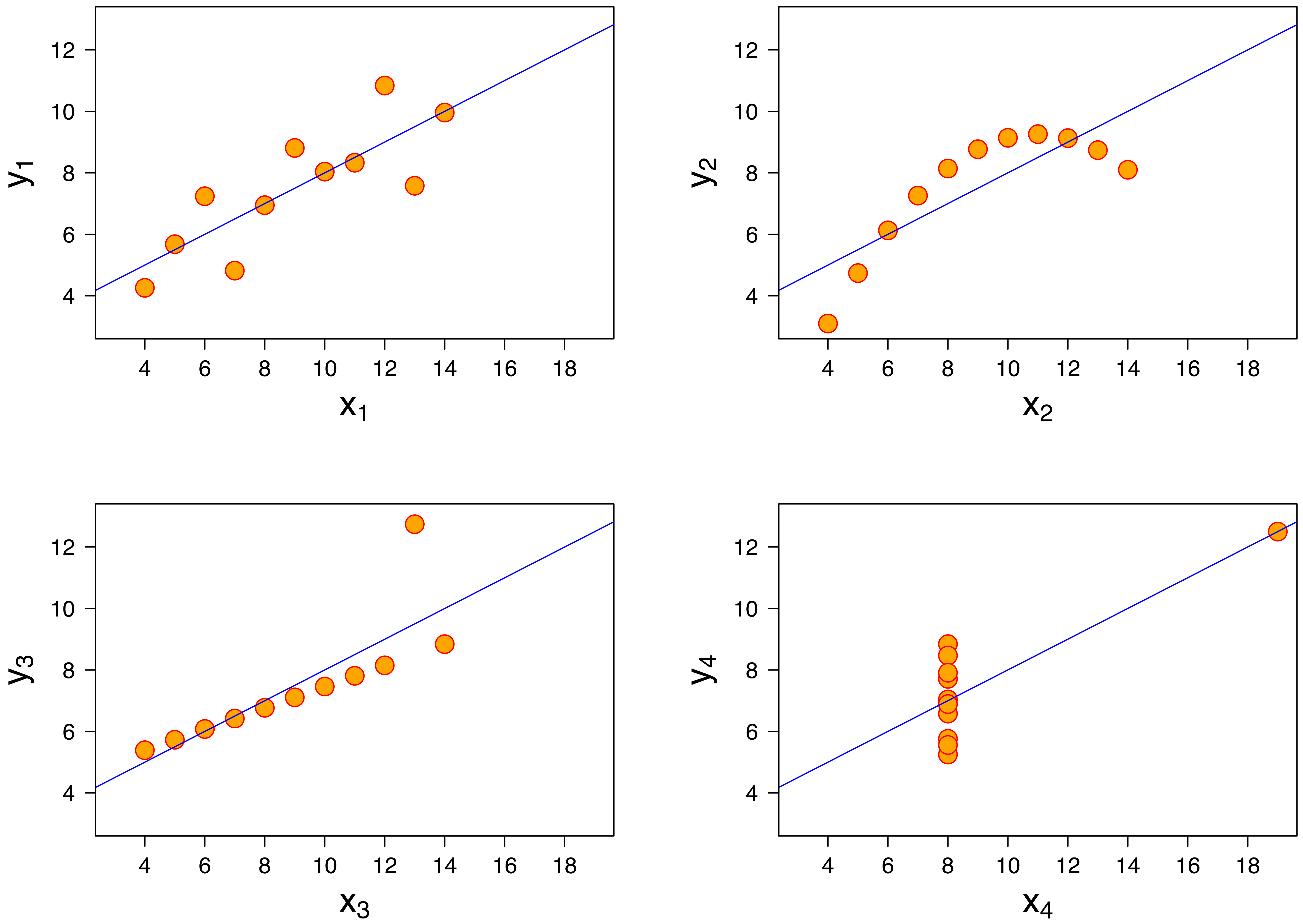
[F. J. Anscombe]

Why Visual?



[F. J. Anscombe]

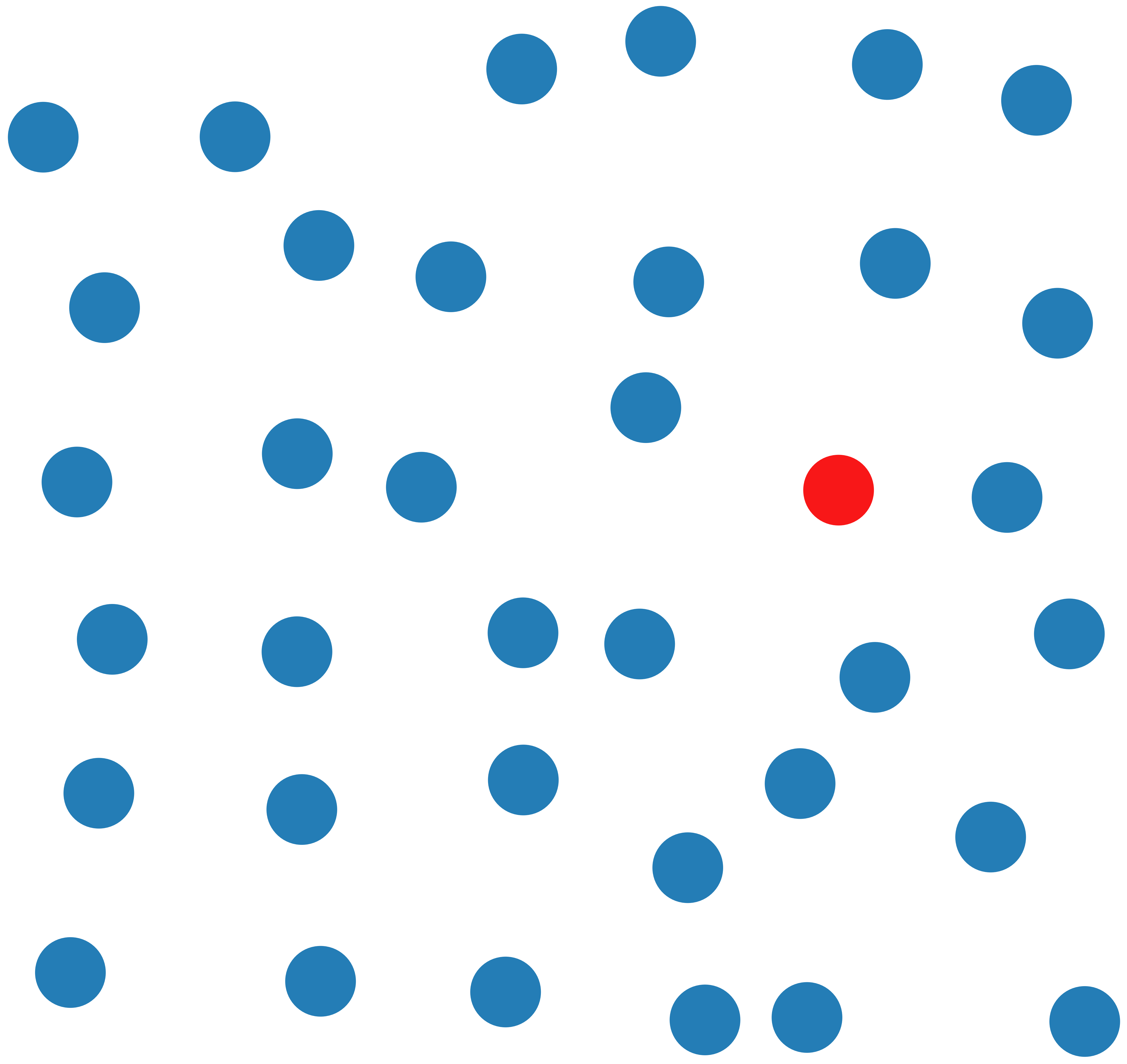
Why Visual?



Mean of x	9
Variance of x	11
Mean of y	7.50
Variance of y	4.122
Correlation	0.816

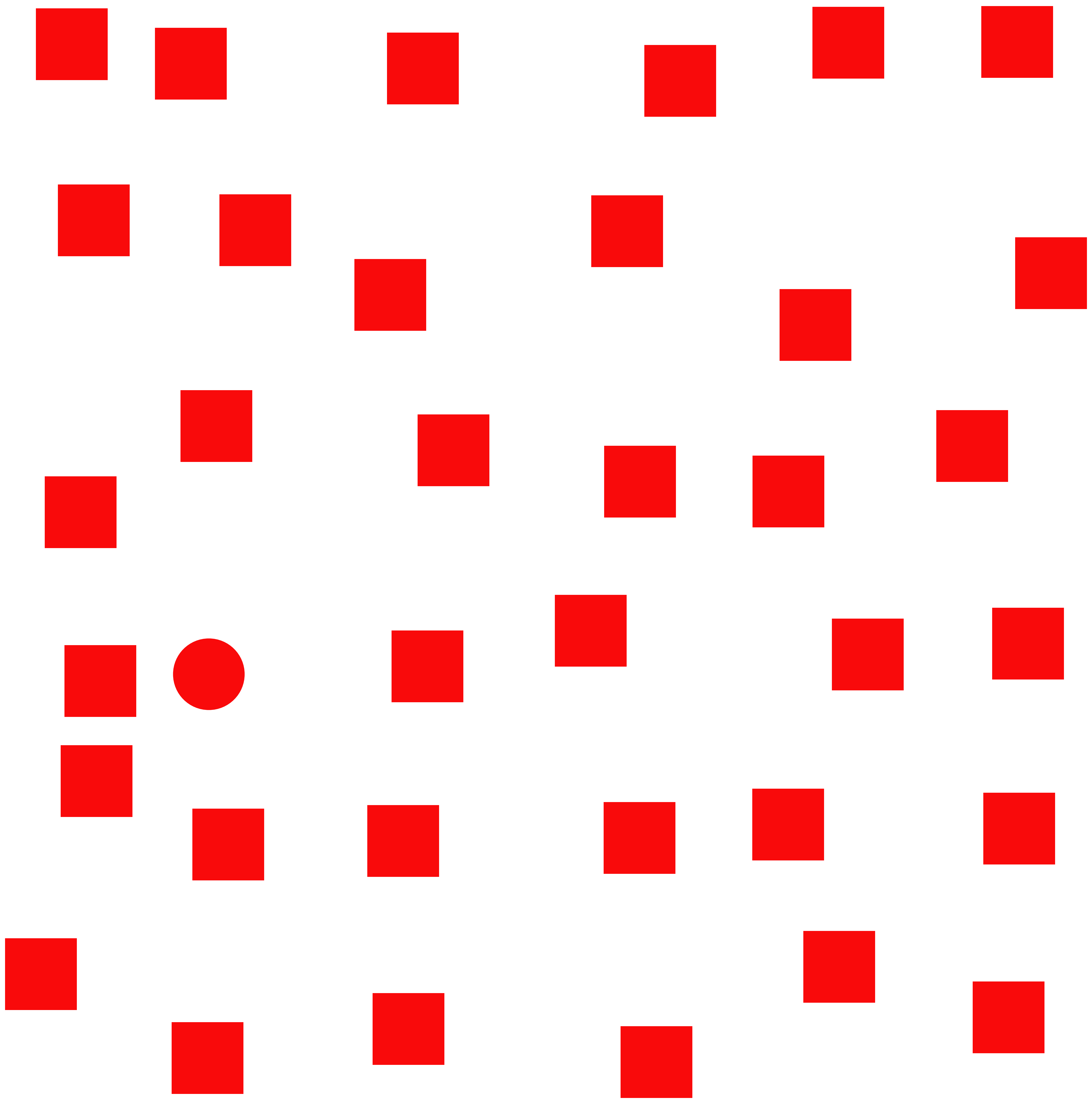
[F. J. Anscombe]

Visual Pop-out



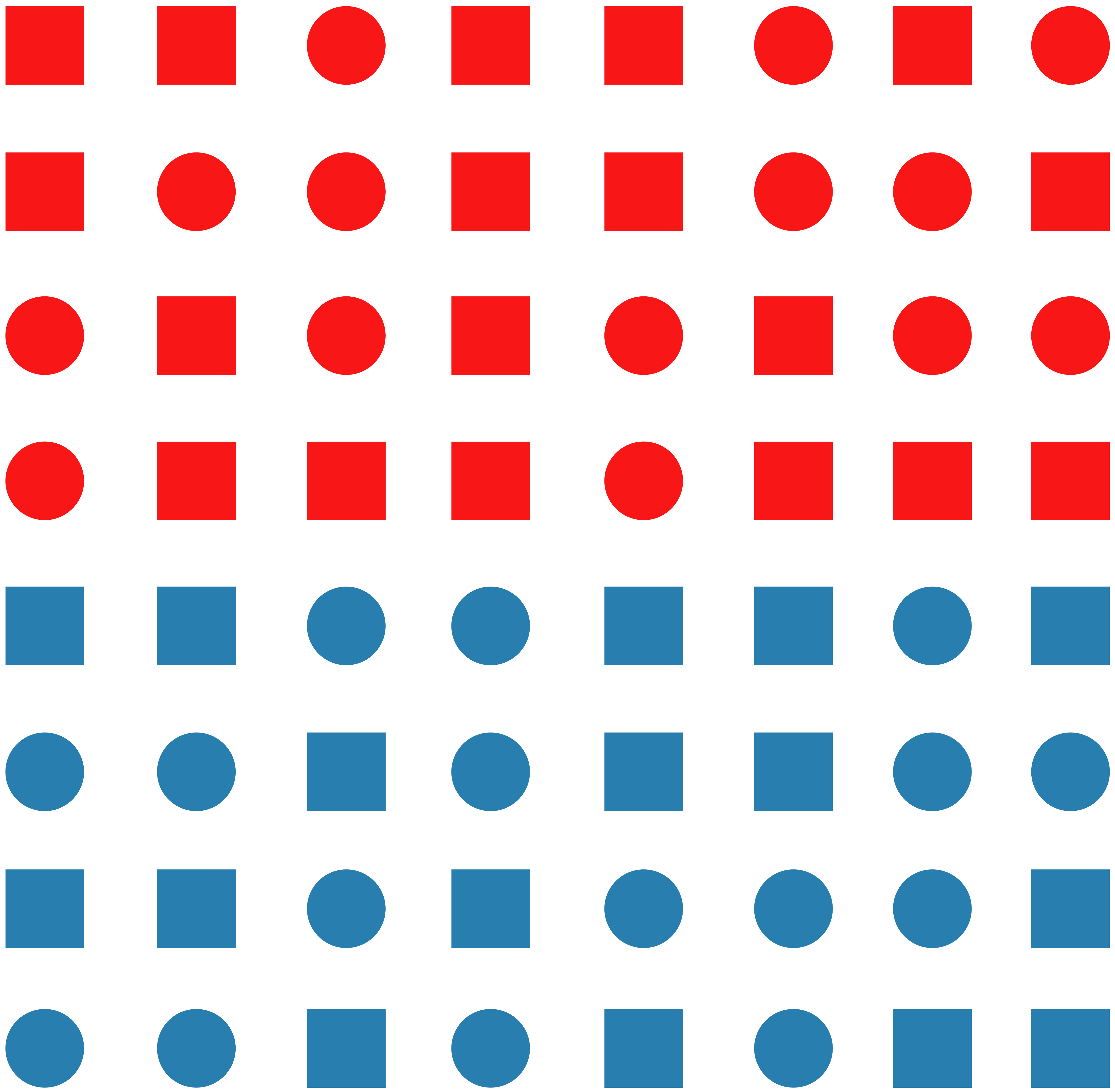
[C. G. Healey]

Visual Pop-out



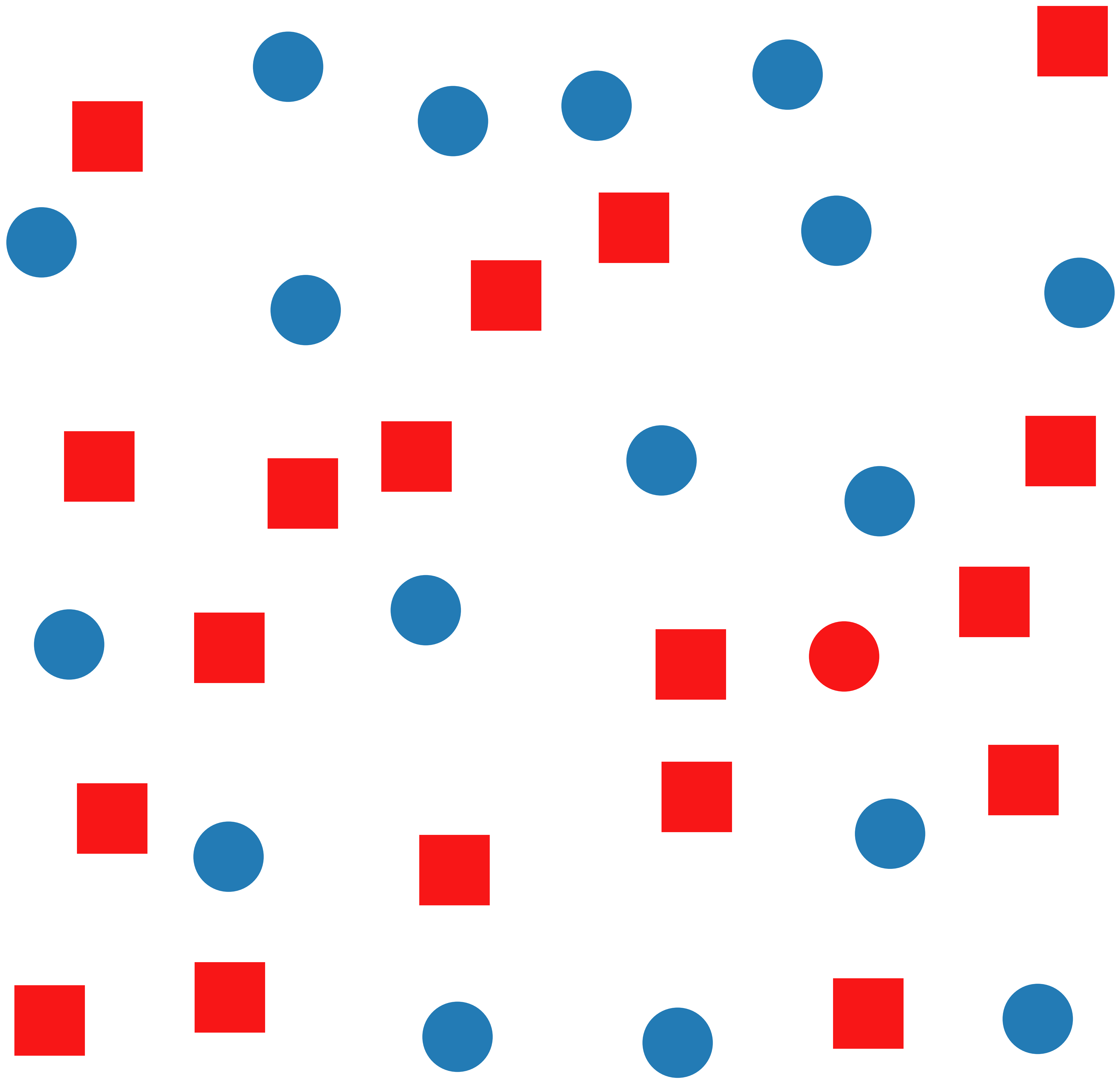
[C. G. Healey]

Visual Pop-out



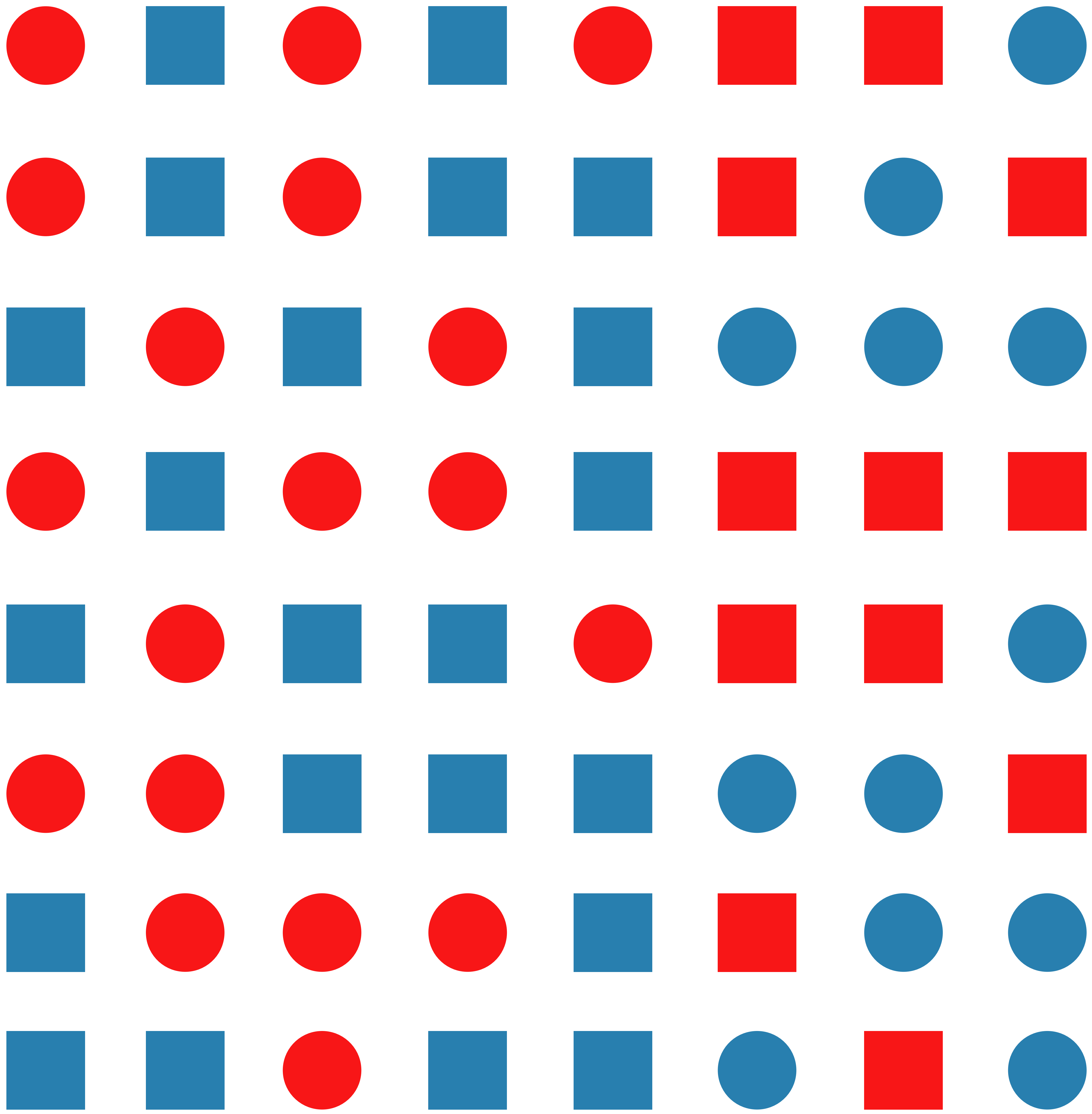
[C. G. Healey]

Visual Perception Limitations



[C. G. Healey]

Visual Perception Limitations



[C. G. Healey]

Databases and Visualization?

Scalable Visualization

J. Heer