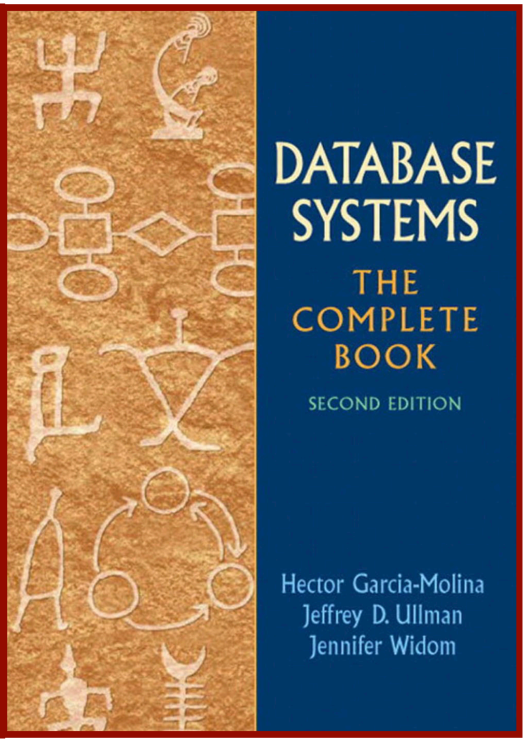
History and Types of Databases



DATABASE SYSTEMS

THE COMPLETE BOOK

SECOND EDITION

Hector Garcia-Molina
Jeffrey D. Ullman
Jennifer Widom

Alan G. Labouseur, Ph.D.
Alan.Labouseur@Marist.edu

Data?

What is data?

?

Data?

What is data?

```
1  007  42  21  12  90  125  86 75 30 9
```

Data?

What is data?

What does it mean?

```
1  007  42  21  12  90  125  86 75 30 9
```

Data?

What is data?

What does it mean?

```
1  007  42  21  12  90  125  86 75 30 9
```

We need context.

Data?

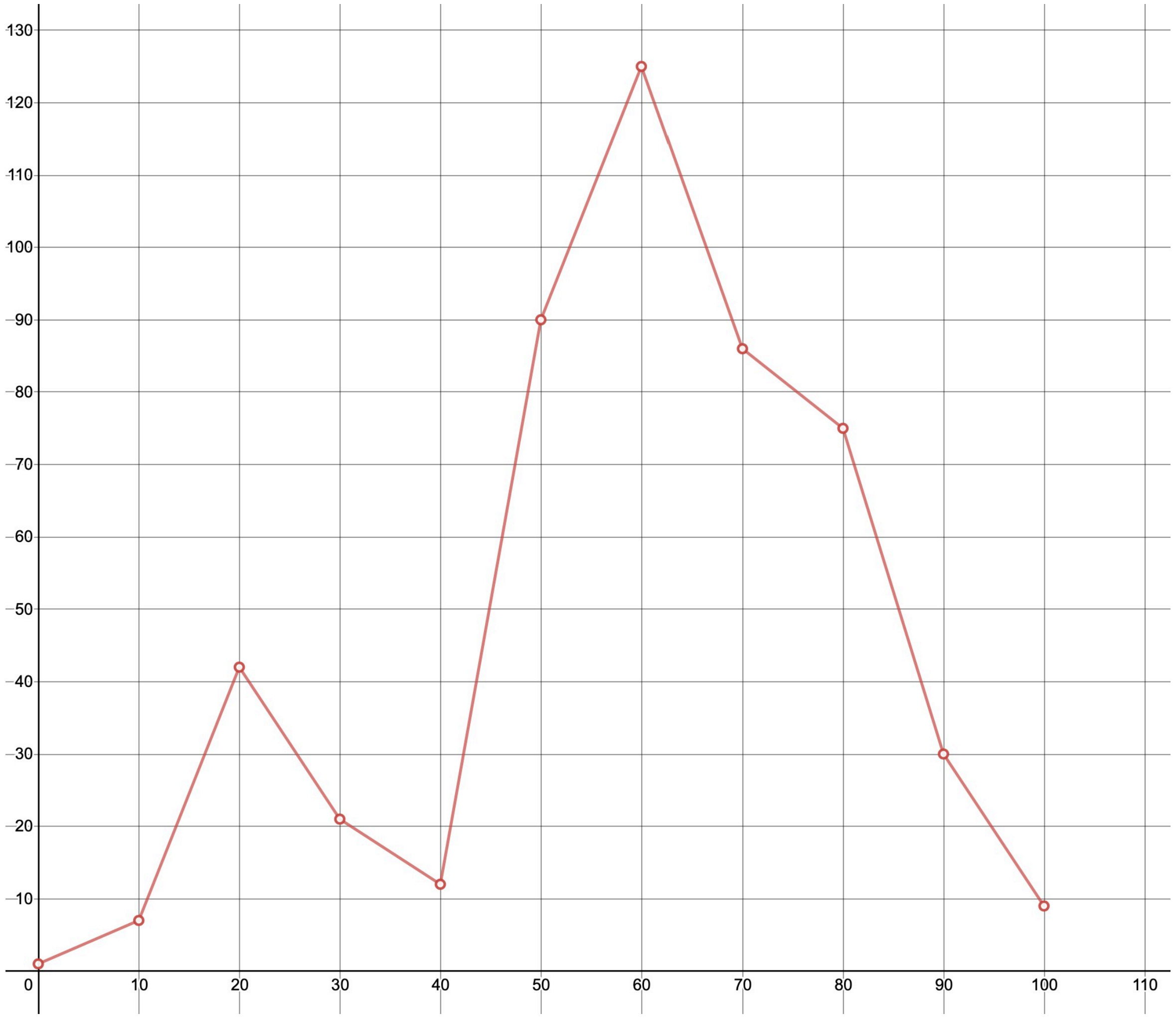
What is data?

What does it mean?

```
1  007  42  21  12  90  125  86 75 30 9
```

With context we can draw conclusions from data.

With context
we have **information**.



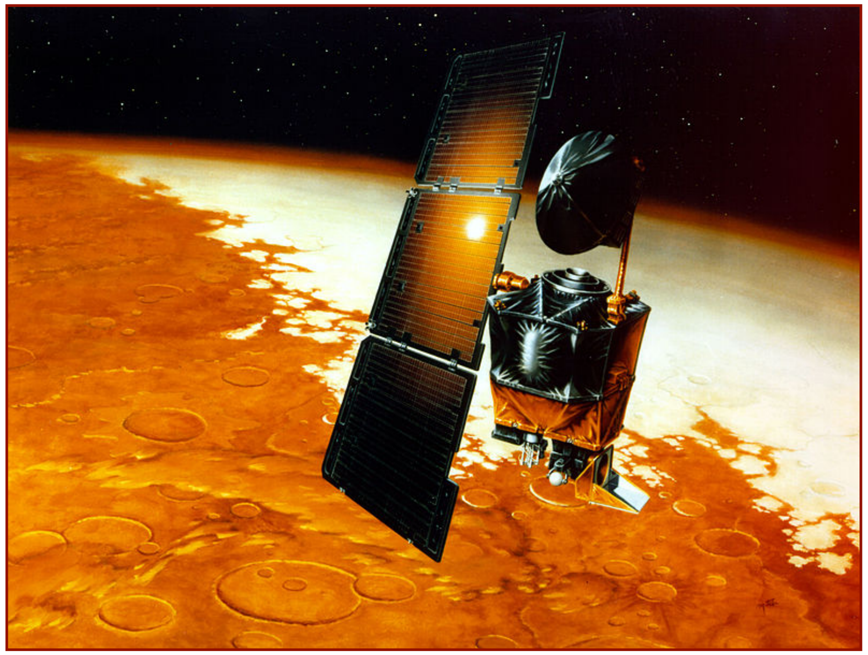
Data is Dangerous

What is data?

What does it mean?

```
1   007   42   21   12   90   125   86 75 30 9
```

What if we're wrong?



Data is Dangerous, Information is Valuable

Data + Context = Information

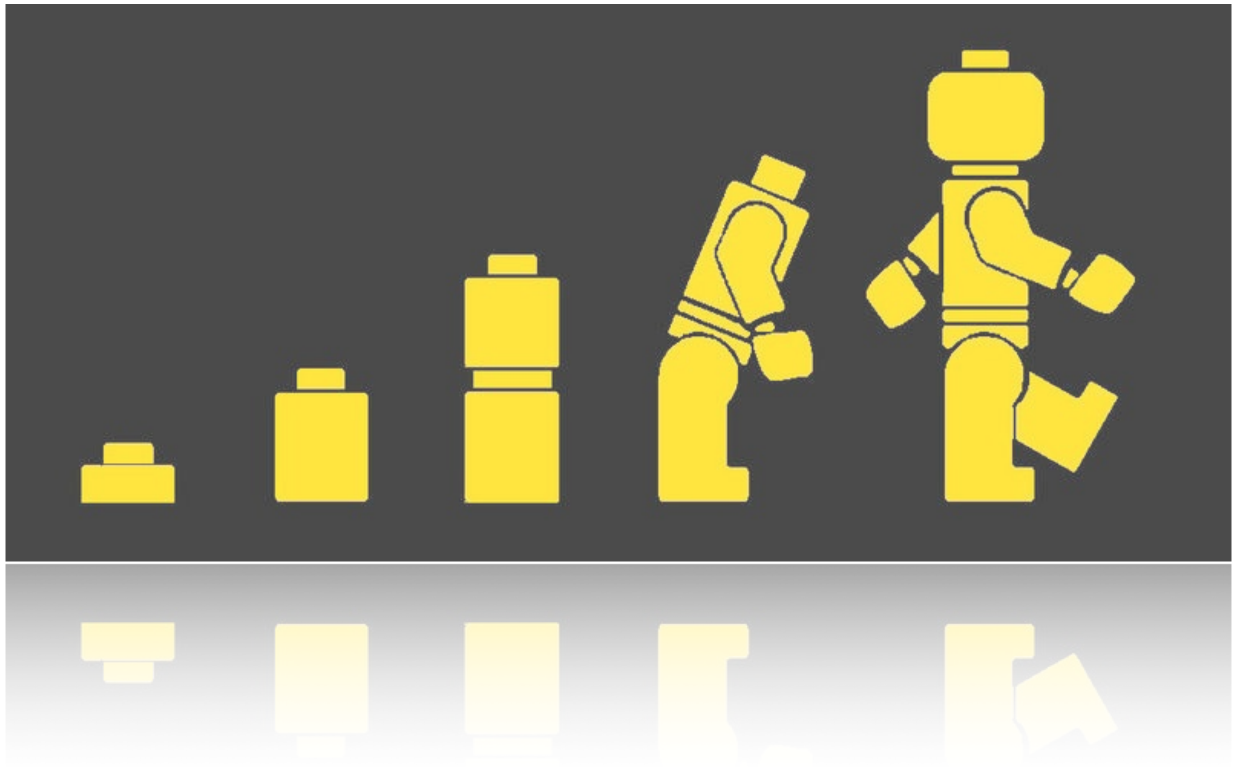
Information is valuable.
Information is difficult to obtain.
Information is what we want.

And to get it, we need to impose structure for context.

Evolution

Consider the evolution of Data Management

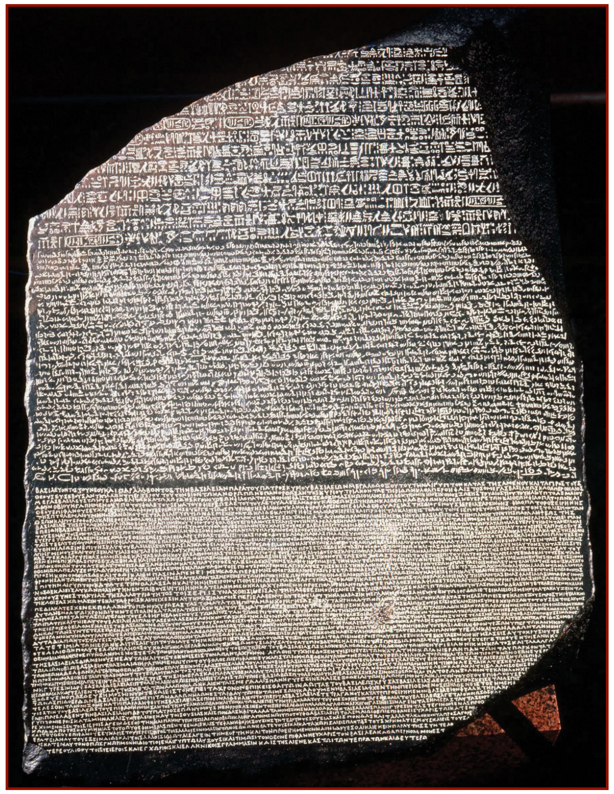
- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

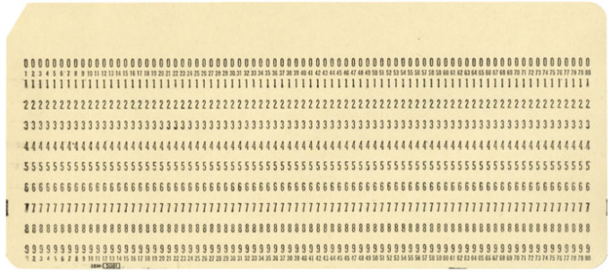


Heavy data

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



1890 Census

Big data

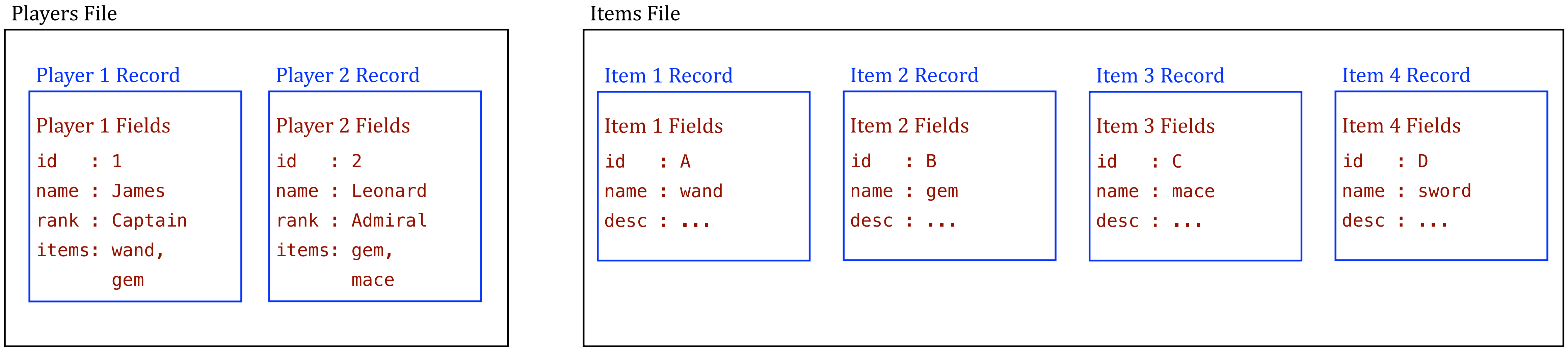
(Still heavy.)

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

- Files of Records of Fields for a D&D-type game



Players File

Player 1 Record

Player 1 Fields

```
id   : 1
name : James
rank : Captain
items: wand,
       gem
```

Player 2 Record

Player 2 Fields

```
id   : 2
name : Leonard
rank : Admiral
items: gem,
       mace
```

Items File

Item 1 Record

Item 1 Fields

```
id   : A
name : wand
desc : ...
```

Item 2 Record

Item 2 Fields

```
id   : B
name : gem
desc : ...
```

Item 3 Record

Item 3 Fields

```
id   : C
name : mace
desc : ...
```

Item 4 Record

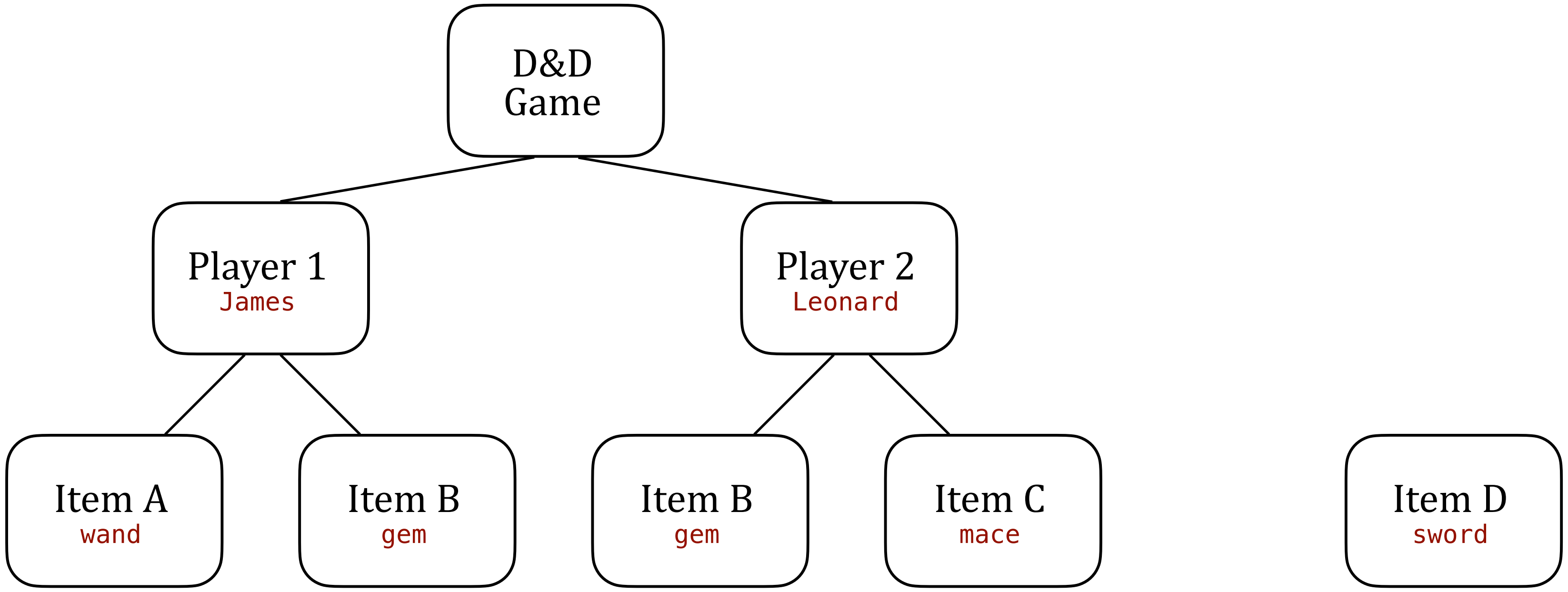
Item 4 Fields

```
id   : D
name : sword
desc : ...
```

Evolution

Consider the evolution of Data Management†

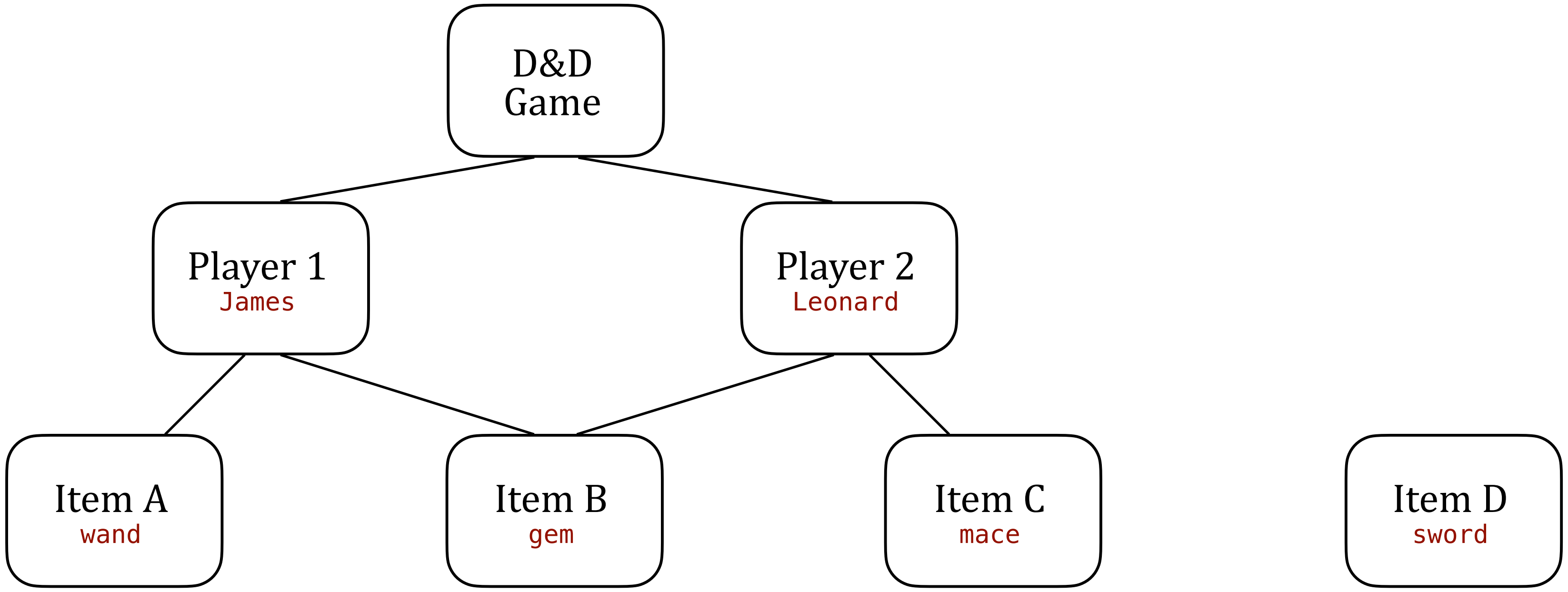
- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



Evolution

Consider the evolution of Data Management

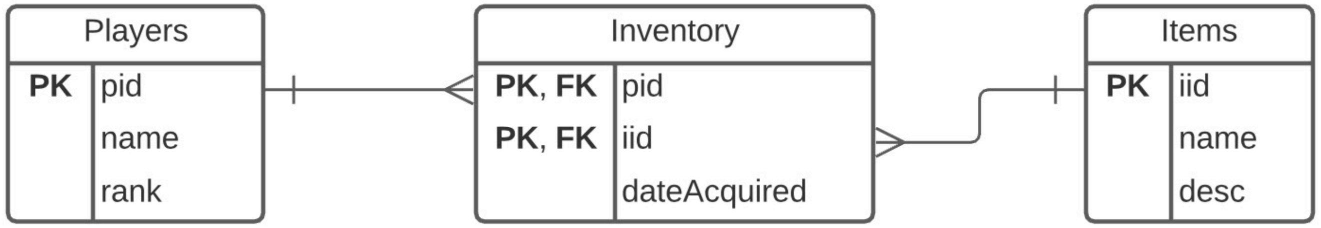
- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

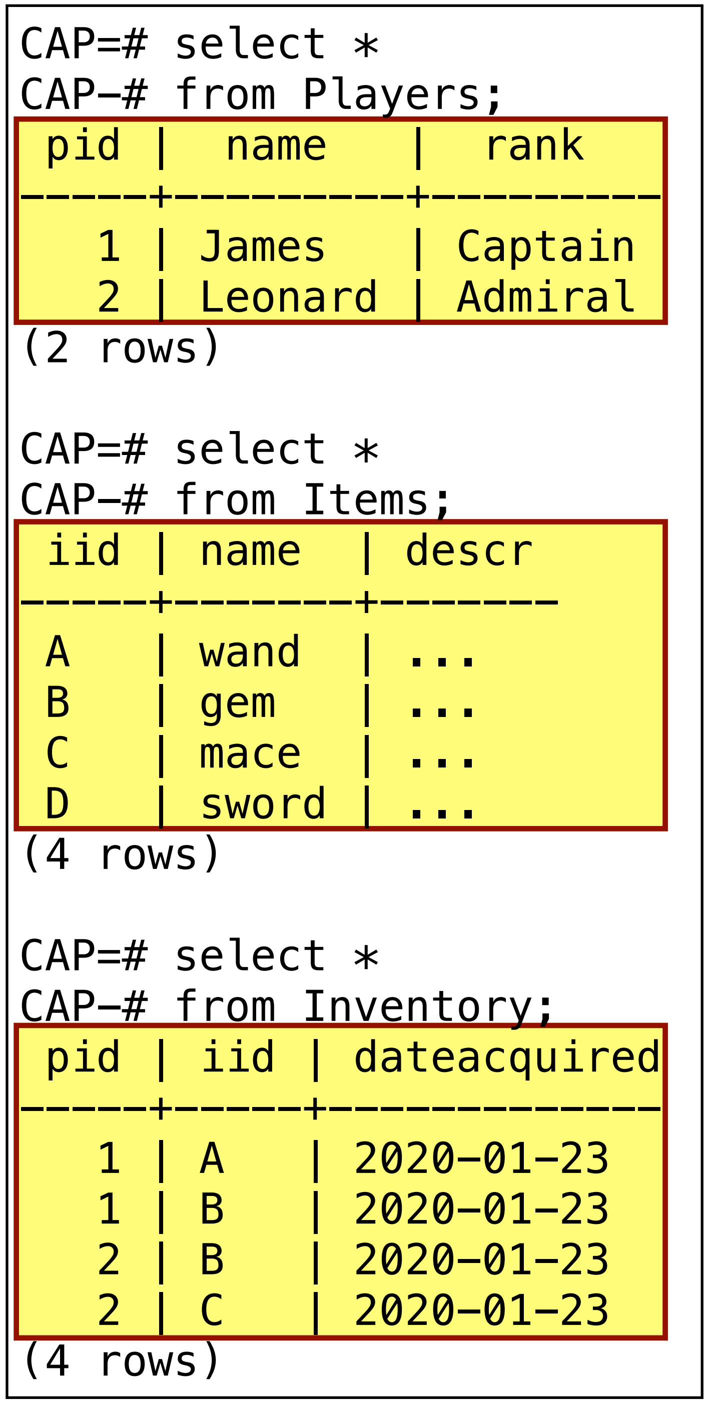


Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



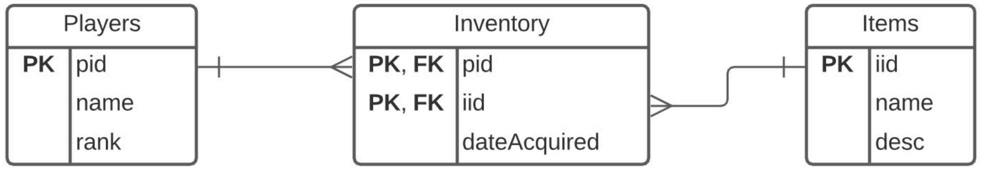


```
CAP=# select *
CAP-# from Players;
 pid |   name    |   rank
-----+-----------+----------
   1 | James     | Captain
   2 | Leonard   | Admiral
(2 rows)

CAP=# select *
CAP-# from Items;
 iid | name   | descr
-----+--------+-------
 A   | wand   | ...
 B   | gem    | ...
 C   | mace   | ...
 D   | sword  | ...
(4 rows)

CAP=# select *
CAP-# from Inventory;
 pid | iid | dateacquired
-----+-----+--------------
   1 | A   | 2020-01-23
   1 | B   | 2020-01-23
   2 | B   | 2020-01-23
   2 | C   | 2020-01-23
(4 rows)
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

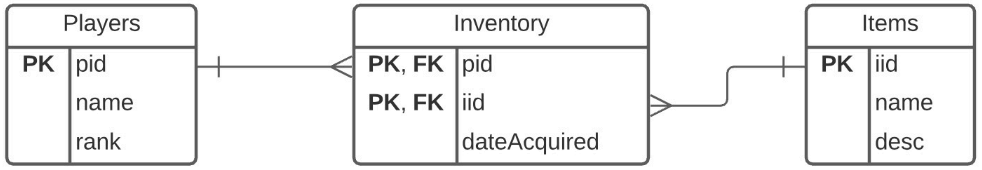


```
DB=# -- Players and their Items
DB=# select Players.name, Items.name
DB=# from Players inner join Inventory on Players.pid = Inventory.pid
DB=#              inner join Items on Inventory.iid = Items.iid
DB=# ;

  name    | name
---------+------
 James    | wand
 James    | gem
 Leonard | gem
 Leonard | mace
(4 rows)
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



Players
PK pid
name
rank

Inventory
PK, FK pid
PK, FK iid
dateAcquired

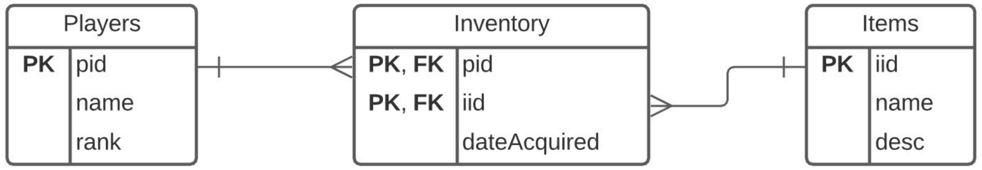
Items
PK iid
name
desc

```
DB=# -- Unused Items
DB=# select *
DB=# from Items
DB=# where iid not in (select iid
DB=#                   from Inventory);

 iid | name  | descr
-----+-------+-------
 D   | sword | ...
(1 row)
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

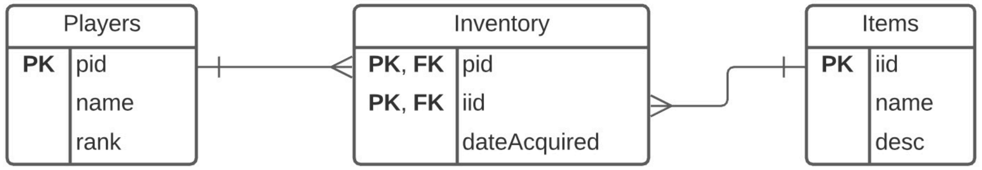


```
DB=# -- Item use count v1
DB=# select iid, count(iid)
DB=# from Inventory
DB=# group by iid
DB=# order by count(iid) DESC;

 iid | count
-----+-------
 B   |     2
 C   |     1
 A   |     1
(3 rows)
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



Players

PK	pid
	name
	rank

Inventory

PK, FK	pid
PK, FK	iid
	dateAcquired

Items

PK	iid
	name
	desc

```
DB=# -- Item use count v2, now with item names!
DB=# select Inventory.iid, Items.name, count(Inventory.iid)
DB=# from Inventory inner join Items on Inventory.iid = Items.iid
DB=# group by Inventory.iid, Items.name
DB=# order by count(Inventory.iid) DESC
DB=# ;

 iid | name | count
-----+------+-------
 B   | gem  |     2
 A   | wand |     1
 C   | mace |     1
(3 rows)
```

Evolution

SQL Script for
Player, Items, and Inventory tables
and a few queries

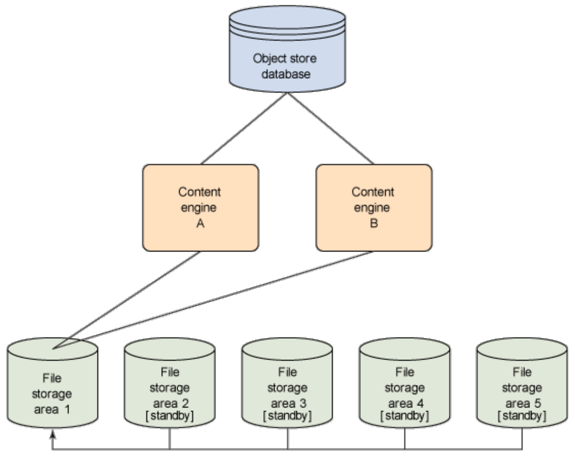
```
create table Players (
   pid   int not null,
   name  text,
   rank text,
 primary key (pid)
);

insert into Players(pid, name, rank)
values (1, 'James',   'Captain'),
       (2, 'Leonard', 'Admiral');

select *
from Players;


create table Items (
   iid   char(1) not null,
   name  text,
   descr text,
 primary key (iid)
);

insert into Items (iid, name, descr)
values ('A', 'wand',  '...'),
       ('B', 'gem',   '...'),
       ('C', 'mace',  '...'),
       ('D', 'sword', '...');

select *
from Items;
```

```
create table Inventory (
   pid          int     not null references Players(pid),
   iid          char(1) not null references Items(iid),
   dateAcquired date,
  primary key(pid, iid)
);

insert into Inventory (pid, iid, dateAcquired)
values (1, 'A', '2020-01-23'),
       (1, 'B', '2020-01-23'),
       (2, 'B', '2020-01-23'),
       (2, 'C', '2020-01-23');

select *
from Inventory;

-- Players and their Items
select Players.name, Items.name
from Players inner join Inventory on Players.pid = Inventory.pid
             inner join Items on Inventory.iid = Items.iid;

-- Unused Items
select *
from Items
where iid not in (select iid
                  from Inventory);

-- Item use count v1
select iid, count(iid)
from Inventory
group by iid
order by count(iid) DESC;

-- Item use count v2, now with item names!
select Inventory.iid, Items.name, count(Inventory.iid)
from Inventory inner join Items on Inventory.iid = Items.iid
group by Inventory.iid, Items.name
order by count(Inventory.iid) DESC;
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

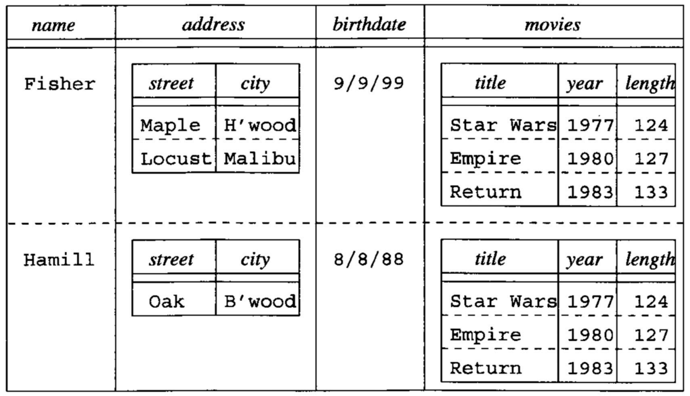

Object store database
Content engine A
Content engine B
File storage area 1
File storage area 2 [standby]
File storage area 3 [standby]
File storage area 4 [standby]
File storage area 5 [standby]

```
Players:
{
  {1, {James, Captain,{A,B}}}
  {2, {Leonard, Admiral,{B,C}}}
}

Items:
{
  {A, {wand, ..., {1}}}
  {B, {gem, ..., {1,2}}}
  {C, {mace, ..., {3}}}
  {D, {sword, ..., {}}}
}
```

Evolution

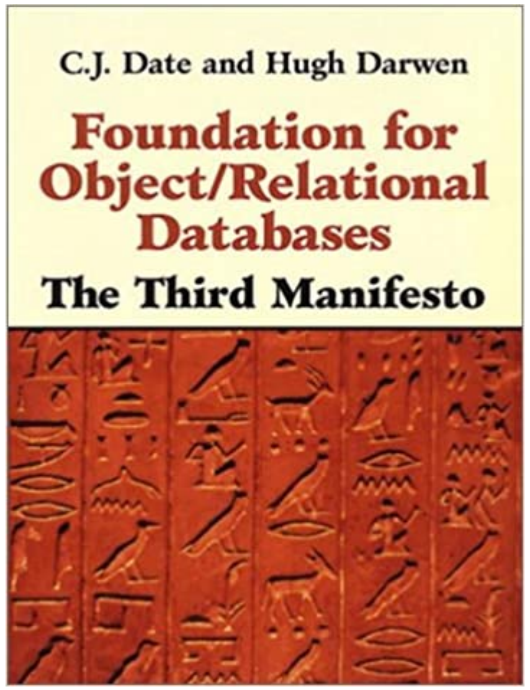
Consider the evolution of Data Management

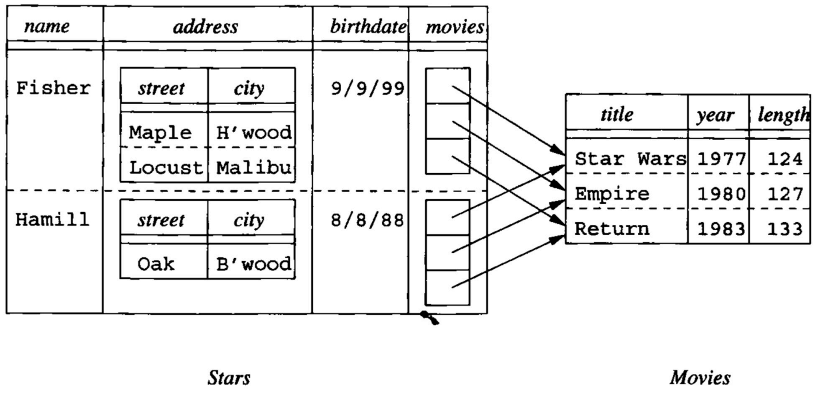
- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



name	address		birthdate	movies		
	street	city		title	year	length
Fisher	Maple	H'wood	9/9/99	Star Wars	1977	124
	Locust	Malibu		Empire	1980	127
				Return	1983	133
Hamill	Oak	B'wood	8/8/88	Star Wars	1977	124
				Empire	1980	127
				Return	1983	133

Figure 10.15: A nested relation for stars and their movies


C.J. Date and Hugh Darwen
Foundation for Object/Relational Databases
The Third Manifesto



name	address		birthdate	movies
	street	city		
Fisher	Maple	H'wood	9/9/99	
	Locust	Malibu		
Hamill	Oak	B'wood	8/8/88	

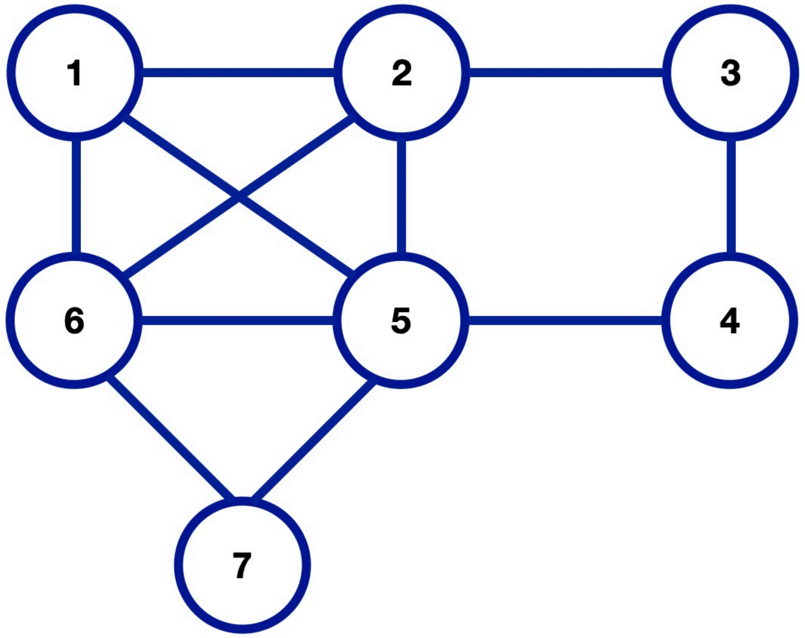
title	year	length
Star Wars	1977	124
Empire	1980	127
Return	1983	133

Figure 10.16: Sets of references as the value of an attribute

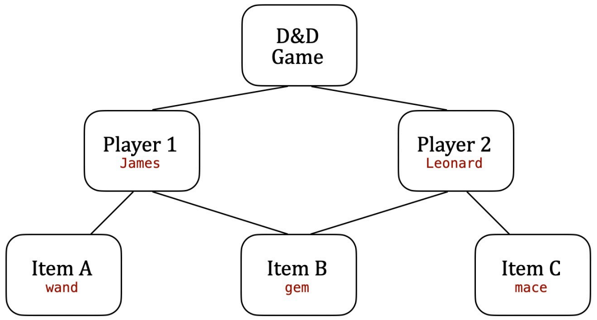
Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases



A **graph** is like a **network** in most ways.



D&D Game

Player 1 James

Player 2 Leonard

Item A wand

Item B gem

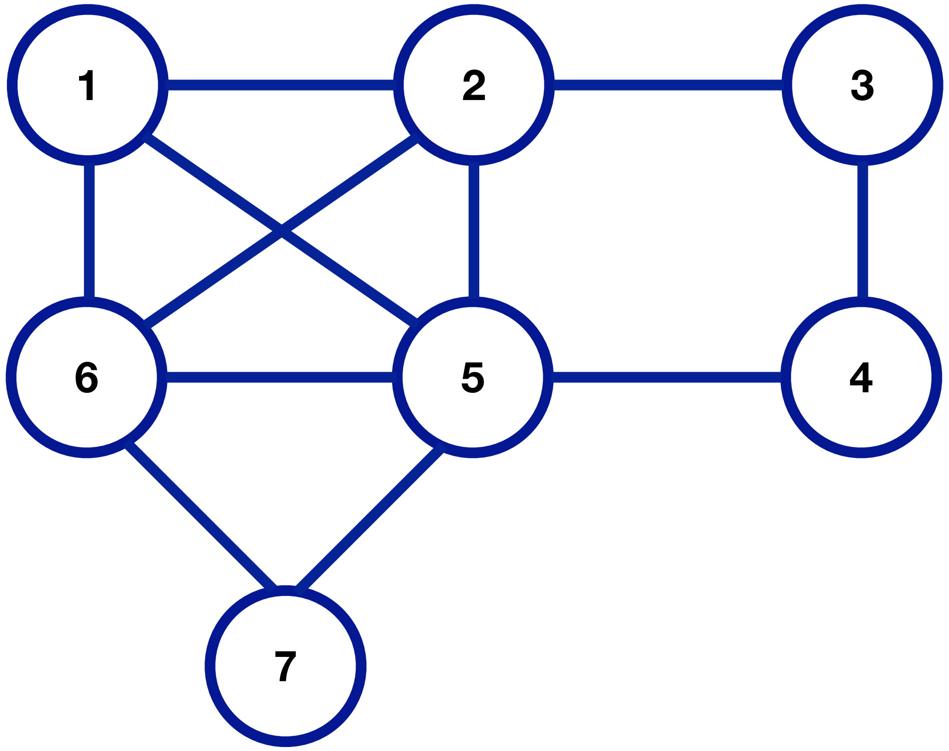
Item C mace

But graph databases are modern tools for managing them and gaining insight from the data pile.

Evolution

Consider the evolution of Data Management

Graph . . .



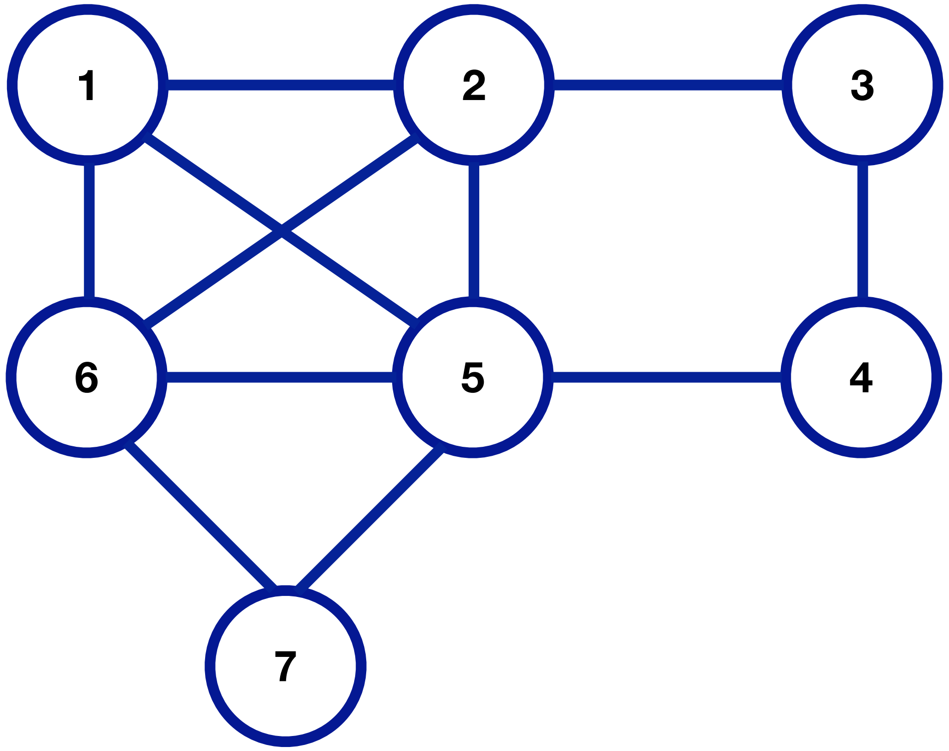
as Matrix

$$\begin{array}{cccccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
1 & . & 1 & . & . & 1 & 1 & . \\
2 & 1 & . & 1 & . & 1 & 1 & . \\
3 & . & 1 & . & 1 & . & . & . \\
4 & . & . & 1 & . & 1 & . & . \\
5 & 1 & 1 & . & 1 & . & 1 & 1 \\
6 & 1 & 1 & . & . & 1 & . & 1 \\
7 & . & . & . & . & 1 & 1 & .
\end{array}$$

Evolution

Consider the evolution of Data Management

Graph . . .



as Adjacency List

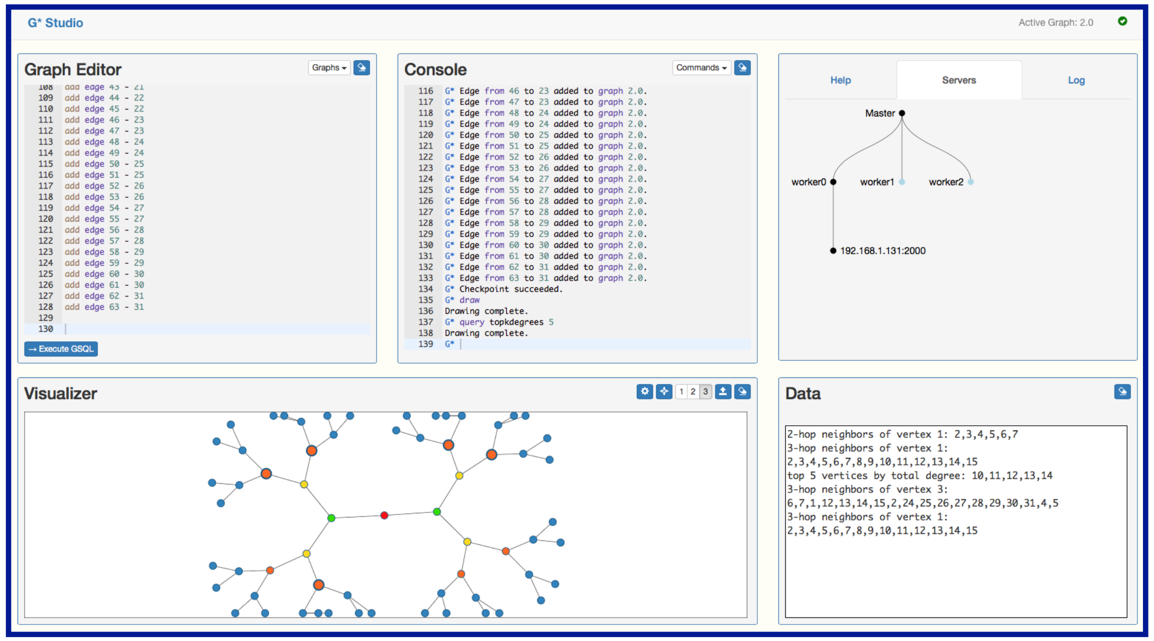
```
[1]  2  5  6
[2]  1  3  5  6
[3]  2  4
[4]  3  5
[5]  1  2  4  6  7
[6]  1  2  5  7
[7]  5  6
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

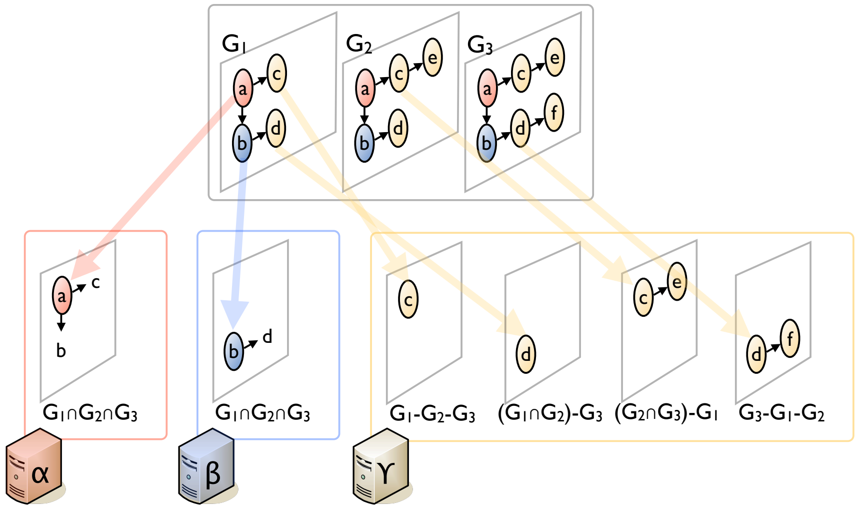
G*The Dynamic Graph Database



```
add edge 43 - 21
add edge 44 - 22
add edge 45 - 22
add edge 46 - 23
add edge 47 - 23
add edge 48 - 24
add edge 49 - 24
add edge 50 - 25
add edge 51 - 25
add edge 52 - 26
add edge 53 - 26
add edge 54 - 27
add edge 55 - 27
add edge 56 - 28
add edge 57 - 28
add edge 58 - 29
add edge 59 - 29
add edge 60 - 30
add edge 61 - 30
add edge 62 - 31
add edge 63 - 31
```

```
G* Edge from 46 to 23 added to graph 2.0.
G* Edge from 47 to 23 added to graph 2.0.
G* Edge from 48 to 24 added to graph 2.0.
G* Edge from 49 to 24 added to graph 2.0.
G* Edge from 50 to 25 added to graph 2.0.
G* Edge from 51 to 25 added to graph 2.0.
G* Edge from 52 to 26 added to graph 2.0.
G* Edge from 53 to 26 added to graph 2.0.
G* Edge from 54 to 27 added to graph 2.0.
G* Edge from 55 to 27 added to graph 2.0.
G* Edge from 56 to 28 added to graph 2.0.
G* Edge from 57 to 28 added to graph 2.0.
G* Edge from 58 to 29 added to graph 2.0.
G* Edge from 59 to 29 added to graph 2.0.
G* Edge from 60 to 30 added to graph 2.0.
G* Edge from 61 to 30 added to graph 2.0.
G* Edge from 62 to 31 added to graph 2.0.
G* Edge from 63 to 31 added to graph 2.0.
G* Checkpoint succeeded.
G* draw
Drawing complete.
G* query topkdegrees 5
Drawing complete.
G*
```

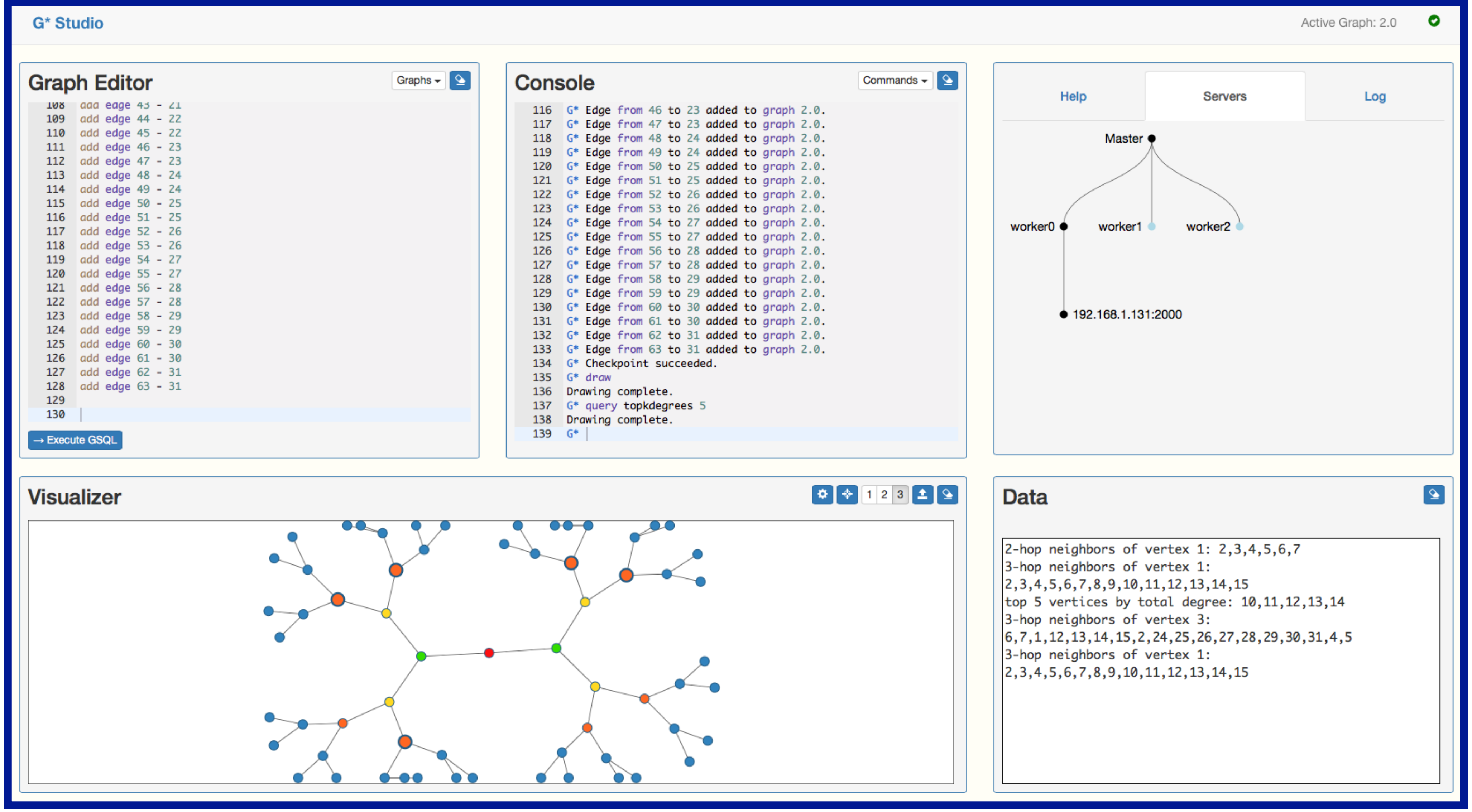
```
2-hop neighbors of vertex 1: 2,3,4,5,6,7
3-hop neighbors of vertex 1:
2,3,4,5,6,7,8,9,10,11,12,13,14,15
top 5 vertices by total degree: 10,11,12,13,14
3-hop neighbors of vertex 3:
6,7,1,12,13,14,15,2,24,25,26,27,28,29,30,31,4,5
3-hop neighbors of vertex 1:
2,3,4,5,6,7,8,9,10,11,12,13,14,15
```



Evolution

G* The Dynamic Graph Database

Browser Application



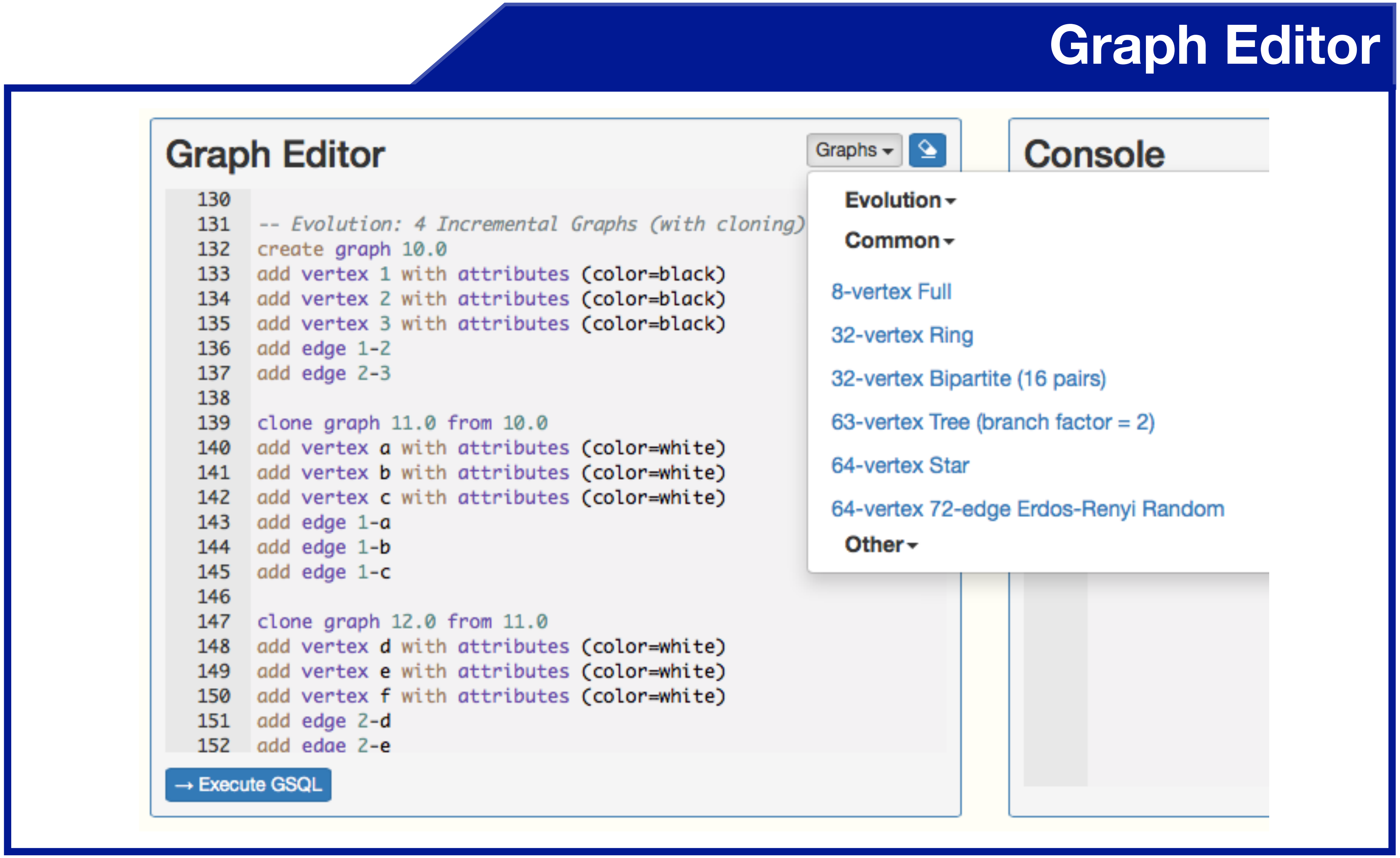
```
add edge 43 - 21
add edge 44 - 22
add edge 45 - 22
add edge 46 - 23
add edge 47 - 23
add edge 48 - 24
add edge 49 - 24
add edge 50 - 25
add edge 51 - 25
add edge 52 - 26
add edge 53 - 26
add edge 54 - 27
add edge 55 - 27
add edge 56 - 28
add edge 57 - 28
add edge 58 - 29
add edge 59 - 29
add edge 60 - 30
add edge 61 - 30
add edge 62 - 31
add edge 63 - 31
```

```
G* Edge from 46 to 23 added to graph 2.0.
G* Edge from 47 to 23 added to graph 2.0.
G* Edge from 48 to 24 added to graph 2.0.
G* Edge from 49 to 24 added to graph 2.0.
G* Edge from 50 to 25 added to graph 2.0.
G* Edge from 51 to 25 added to graph 2.0.
G* Edge from 52 to 26 added to graph 2.0.
G* Edge from 53 to 26 added to graph 2.0.
G* Edge from 54 to 27 added to graph 2.0.
G* Edge from 55 to 27 added to graph 2.0.
G* Edge from 56 to 28 added to graph 2.0.
G* Edge from 57 to 28 added to graph 2.0.
G* Edge from 58 to 29 added to graph 2.0.
G* Edge from 59 to 29 added to graph 2.0.
G* Edge from 60 to 30 added to graph 2.0.
G* Edge from 61 to 30 added to graph 2.0.
G* Edge from 62 to 31 added to graph 2.0.
G* Edge from 63 to 31 added to graph 2.0.
G* Checkpoint succeeded.
G* draw
Drawing complete.
G* query topkdegrees 5
Drawing complete.
G*
```

```
2-hop neighbors of vertex 1: 2,3,4,5,6,7
3-hop neighbors of vertex 1:
2,3,4,5,6,7,8,9,10,11,12,13,14,15
top 5 vertices by total degree: 10,11,12,13,14
3-hop neighbors of vertex 3:
6,7,1,12,13,14,15,2,24,25,26,27,28,29,30,31,4,5
3-hop neighbors of vertex 1:
2,3,4,5,6,7,8,9,10,11,12,13,14,15
```

Evolution

G*The Dynamic Graph Database

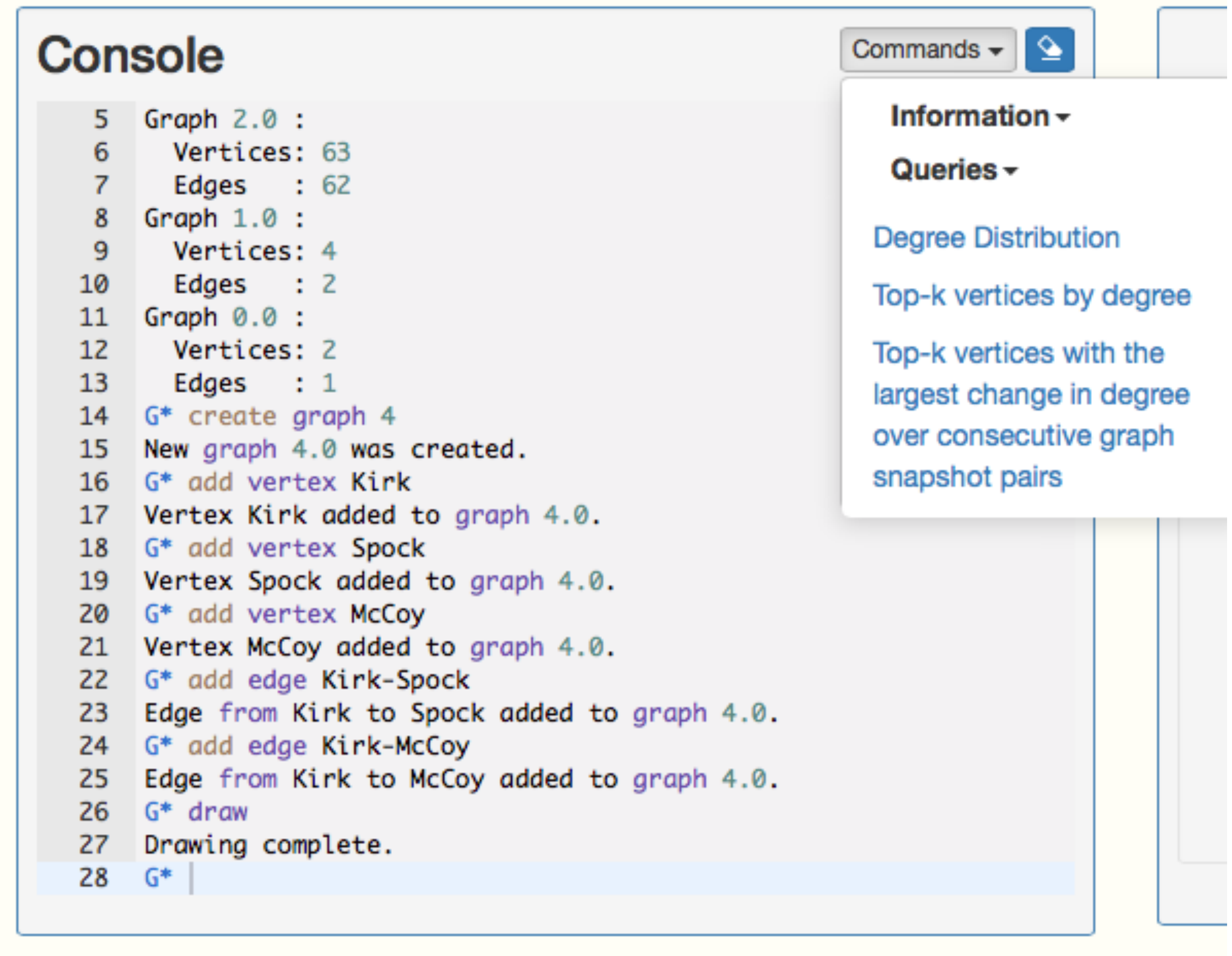


```

-- Evolution: 4 Incremental Graphs (with cloning)
create graph 10.0
add vertex 1 with attributes (color=black)
add vertex 2 with attributes (color=black)
add vertex 3 with attributes (color=black)
add edge 1-2
add edge 2-3

clone graph 11.0 from 10.0
add vertex a with attributes (color=white)
add vertex b with attributes (color=white)
add vertex c with attributes (color=white)
add edge 1-a
add edge 1-b
add edge 1-c

clone graph 12.0 from 11.0
add vertex d with attributes (color=white)
add vertex e with attributes (color=white)
add vertex f with attributes (color=white)
add edge 2-d
add edge 2-e
```

Evolution

G*The Dynamic Graph Database

Interactive Console



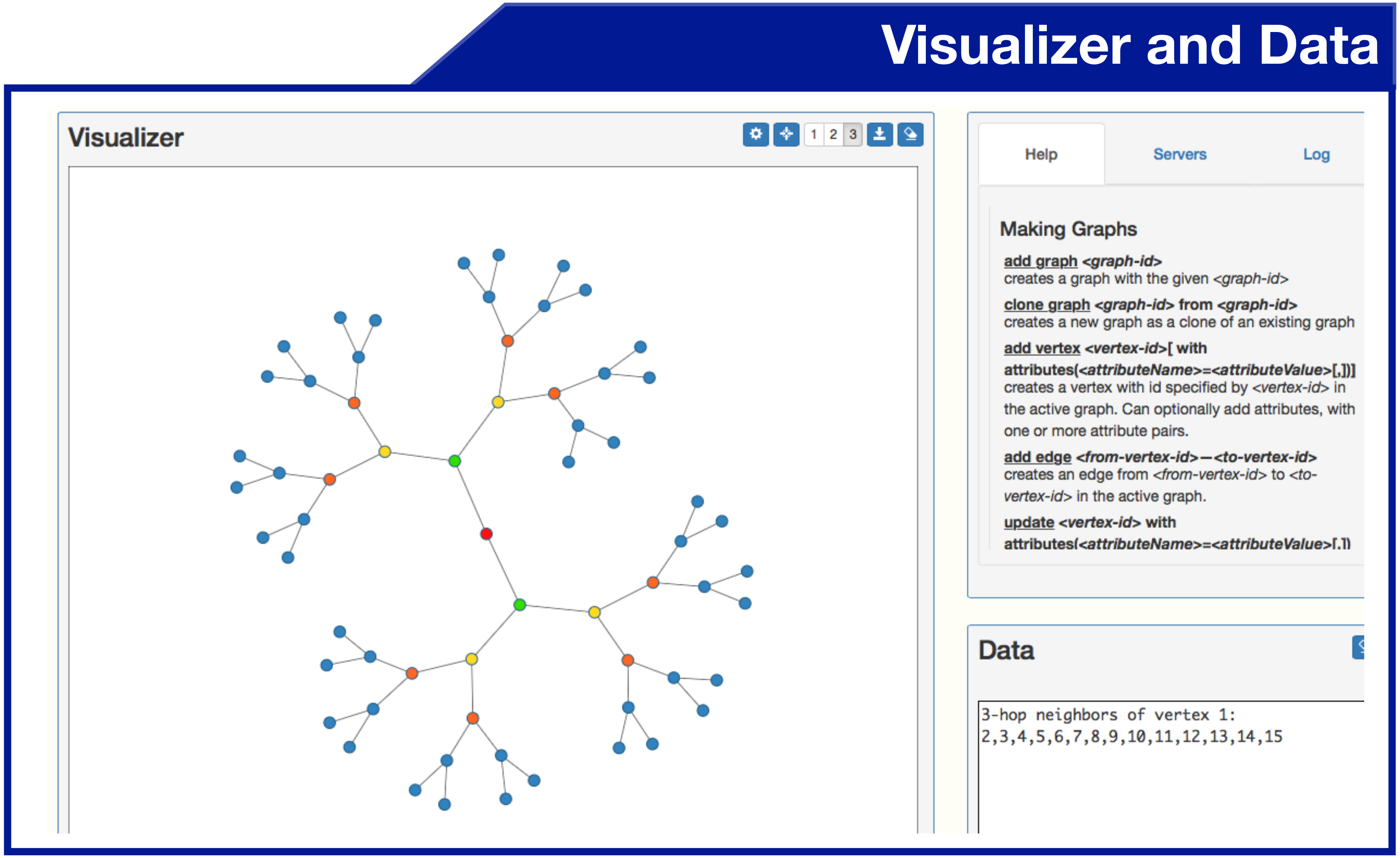
Console

```
Graph 2.0 :
  Vertices: 63
  Edges   : 62
Graph 1.0 :
  Vertices: 4
  Edges   : 2
Graph 0.0 :
  Vertices: 2
  Edges   : 1
G* create graph 4
New graph 4.0 was created.
G* add vertex Kirk
Vertex Kirk added to graph 4.0.
G* add vertex Spock
Vertex Spock added to graph 4.0.
G* add vertex McCoy
Vertex McCoy added to graph 4.0.
G* add edge Kirk-Spock
Edge from Kirk to Spock added to graph 4.0.
G* add edge Kirk-McCoy
Edge from Kirk to McCoy added to graph 4.0.
G* draw
Drawing complete.
G*
```

Commands
Information
Queries
Degree Distribution
Top-k vertices by degree
Top-k vertices with the largest change in degree over consecutive graph snapshot pairs

Evolution

G* The Dynamic Graph Database



Visualizer and Data

Making Graphs

add graph <graph-id>
creates a graph with the given <graph-id>

clone graph <graph-id> from <graph-id>
creates a new graph as a clone of an existing graph

add vertex <vertex-id>[with attributes(<attributeName>=<attributeValue>[,])]
creates a vertex with id specified by <vertex-id> in the active graph. Can optionally add attributes, with one or more attribute pairs.

add edge <from-vertex-id>—<to-vertex-id>
creates an edge from <from-vertex-id> to <to-vertex-id> in the active graph.

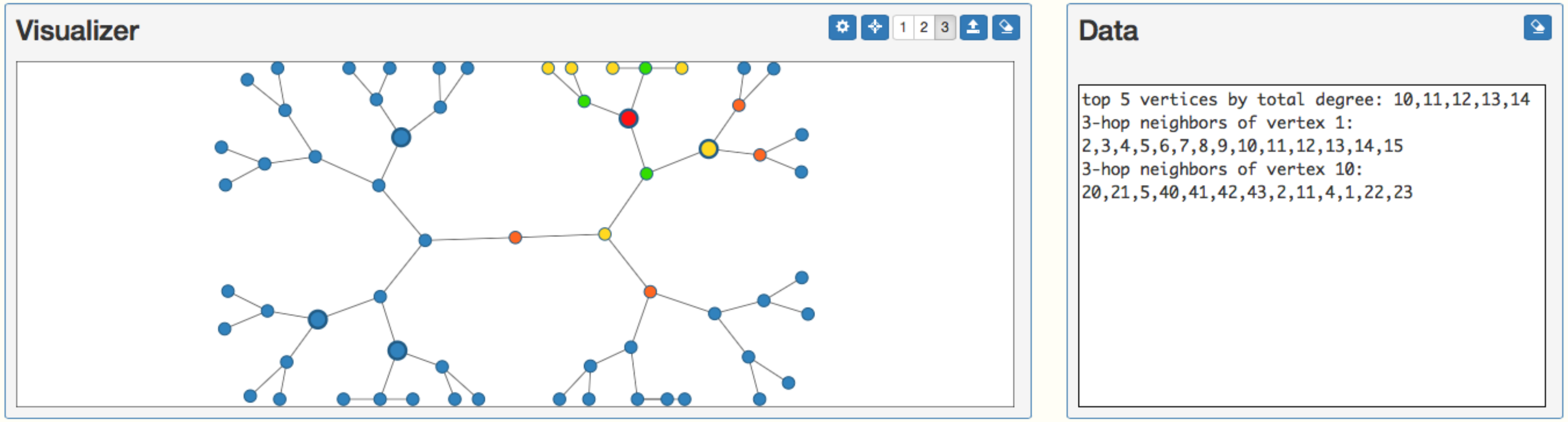
update <vertex-id> with attributes(<attributeName>=<attributeValue>[,])

```
3-hop neighbors of vertex 1:
2,3,4,5,6,7,8,9,10,11,12,13,14,15
```

Evolution

G*The Dynamic Graph Database

Top-k Query



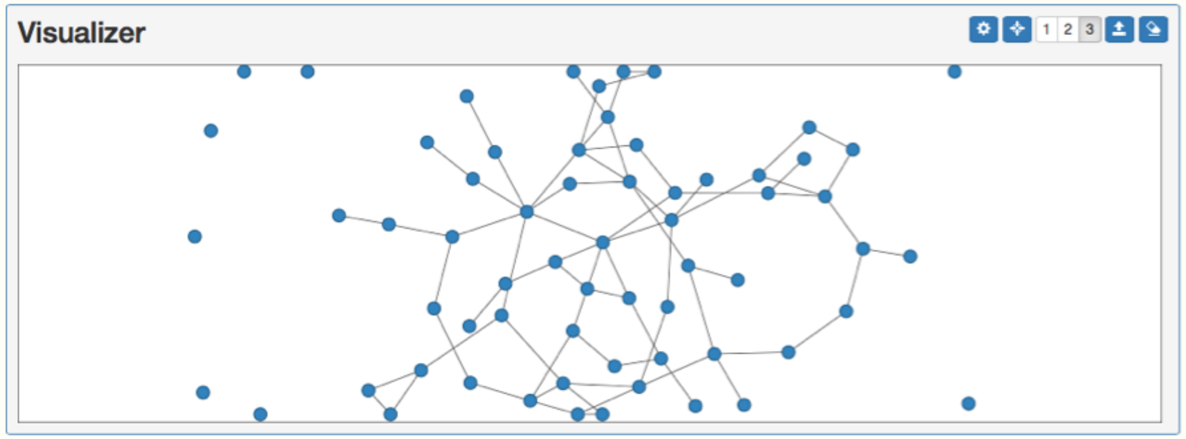
```
top 5 vertices by total degree: 10,11,12,13,14
3-hop neighbors of vertex 1:
2,3,4,5,6,7,8,9,10,11,12,13,14,15
3-hop neighbors of vertex 10:
20,21,5,40,41,42,43,2,11,4,1,22,23
```

Evolution

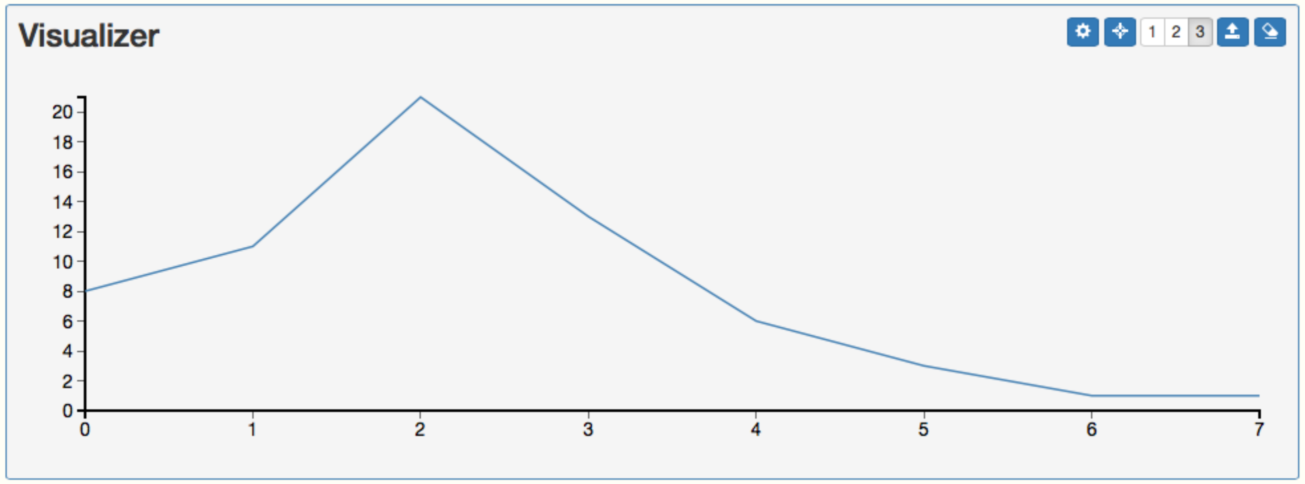
G*The Dynamic Graph Database

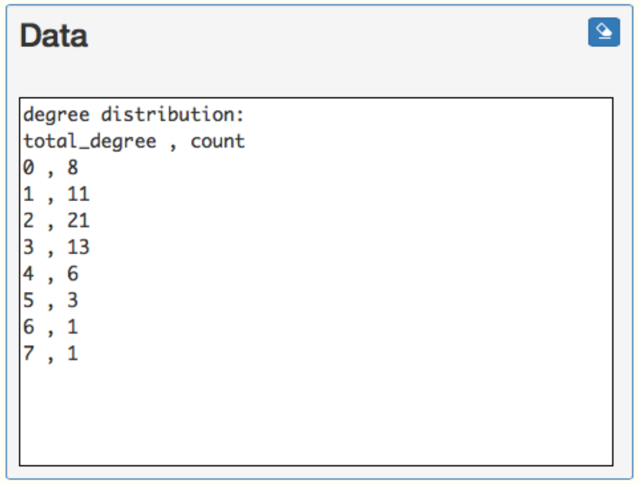
Degree Distribution Query

Erdős-Rényi random graph



Degree Distribution

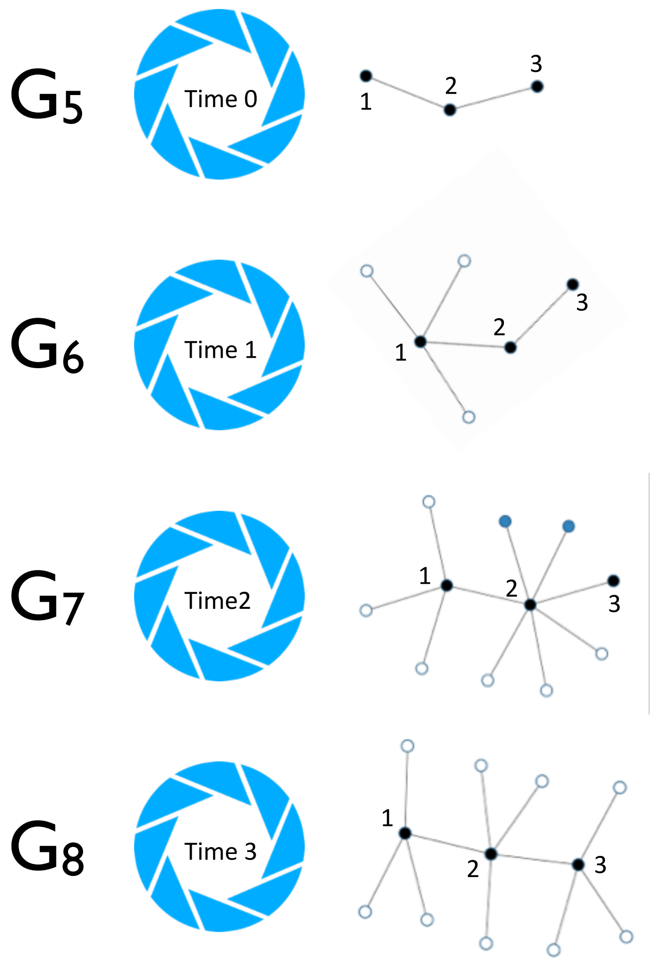


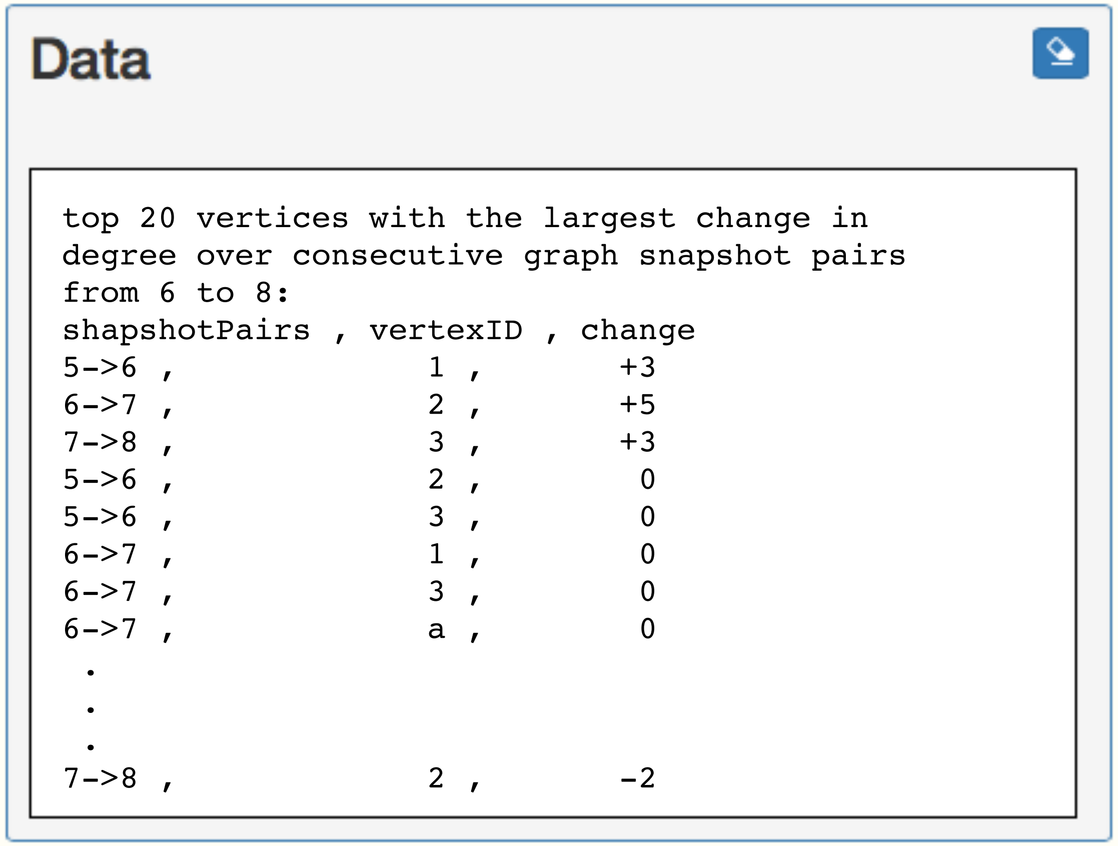


```
degree distribution:
total_degree , count
0 , 8
1 , 11
2 , 21
3 , 13
4 , 6
5 , 3
6 , 1
7 , 1
```

Evolution

G* The Dynamic Graph Database





```
top 20 vertices with the largest change in
degree over consecutive graph snapshot pairs
from 6 to 8:
shapshotPairs , vertexID , change
5->6 ,           1 ,      +3
6->7 ,           2 ,      +5
7->8 ,           3 ,      +3
5->6 ,           2 ,       0
5->6 ,           3 ,       0
6->7 ,           1 ,       0
6->7 ,           3 ,       0
6->7 ,           a ,       0
 .
 .
 .
7->8 ,           2 ,      -2
```

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- relational databases
- object stores
- object-relational databases (Third Manifesto?)
- graph databases

What should we concentrate on?
Where should we spend our time?

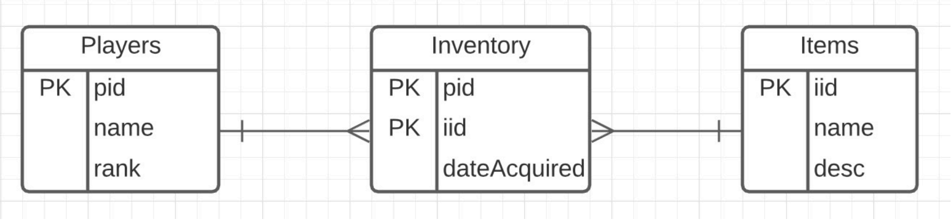
?

Evolution

Consider the evolution of Data Management

- stone tablets
- punched cards
- flat files on tape
- hierarchical databases on DASD
- network databases on disk
- **relational databases**
- object stores
- object-relational databases (Third Manifesto?)
- **graph databases**

We will spend most of our time on the Relational model and relational databases. And a little time on graphs.



Players
PK pid
name
rank

Inventory
PK pid
PK iid
dateAcquired

Items
PK iid
name
desc