

DATABASE INTRO

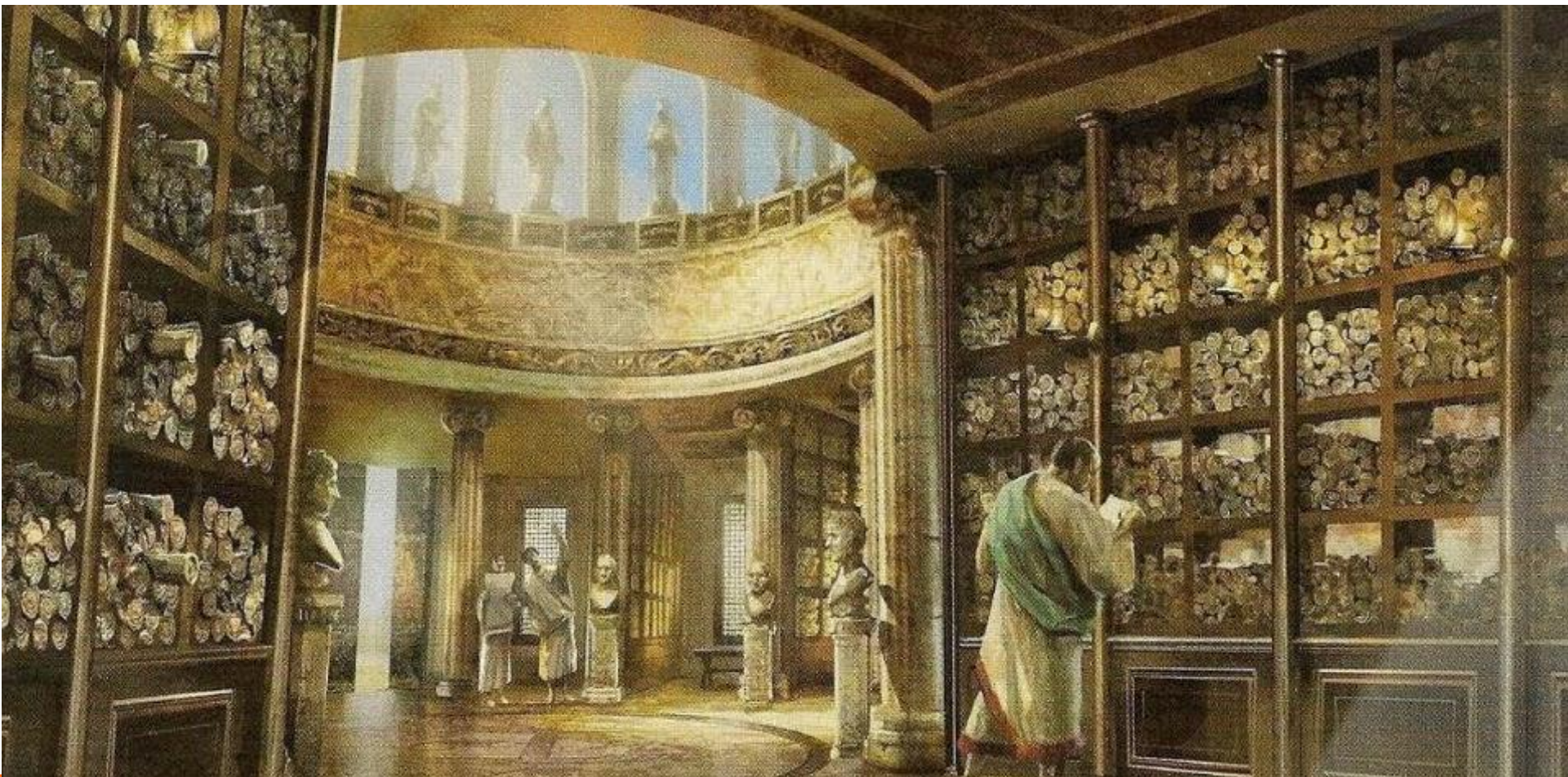
JOHN JERNIGAN 8/23/2022

So many databases...
Why?



Library of Alexandria

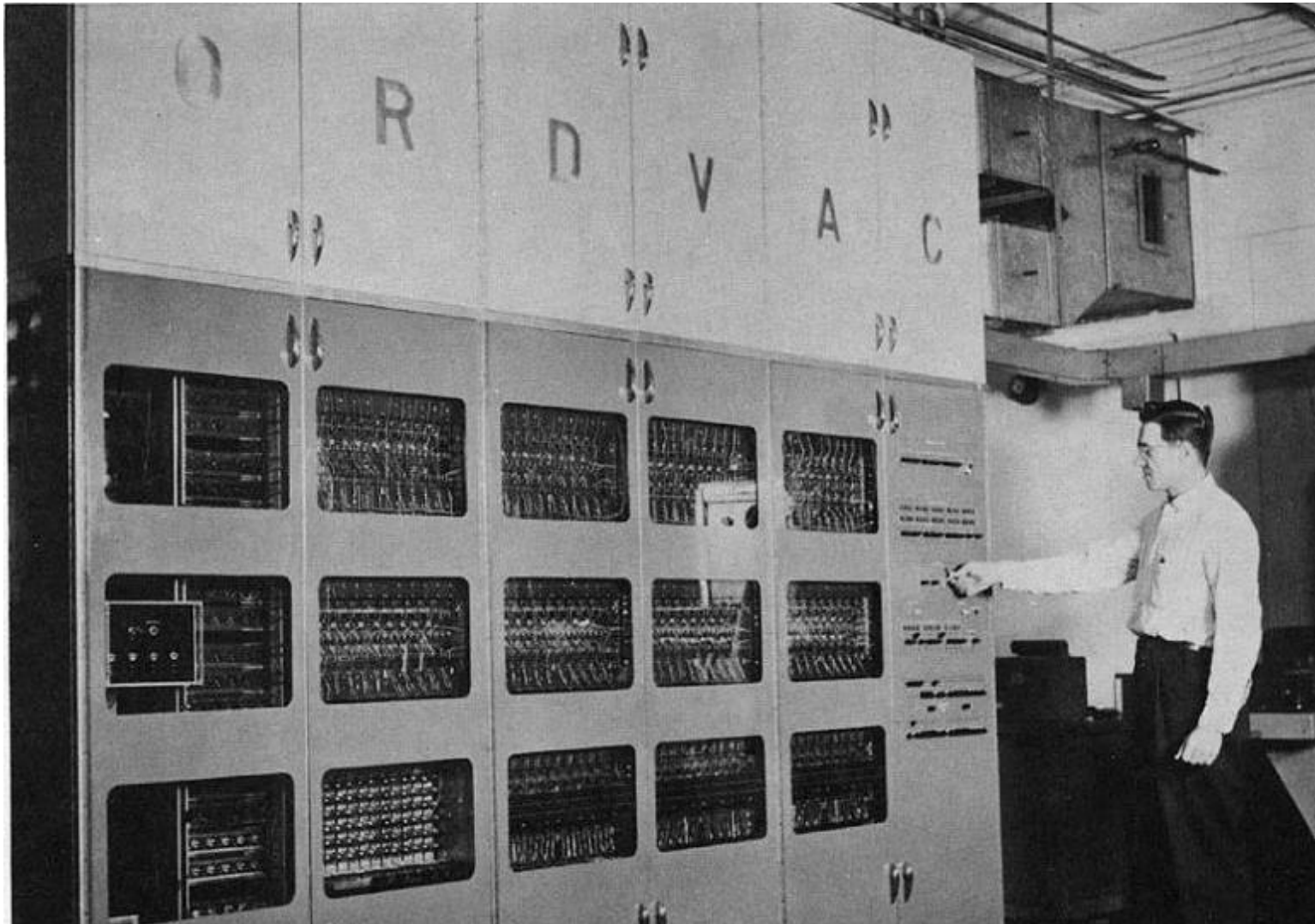
First use of alphabetical ordering?

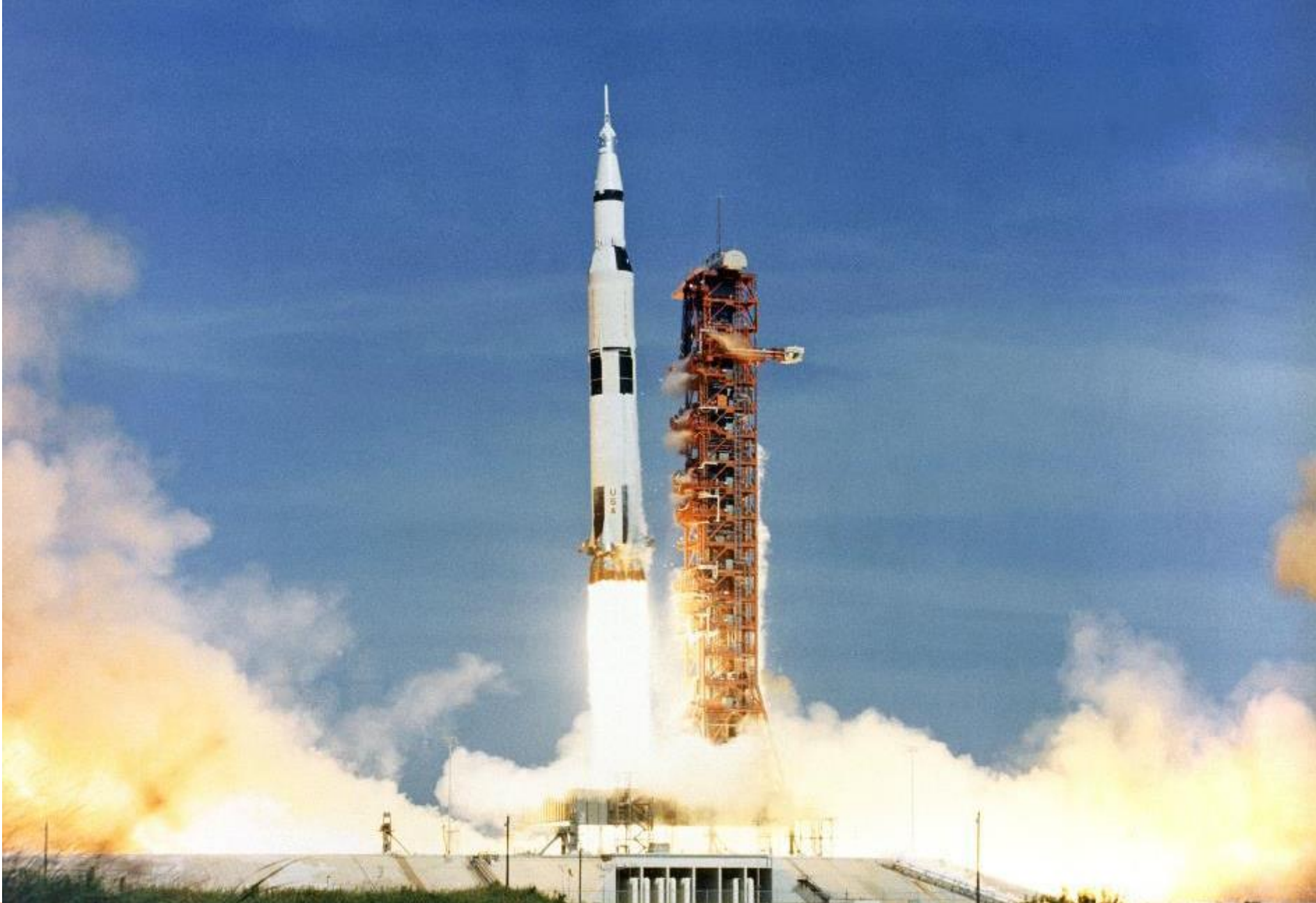


Casson, L. (2001). Libraries in the ancient world. ProQuest Ebook Central (pg 37)

108	Coucheur de la 1 ^{re} classe	M ^{re} Camille Vaforgue	Mise en gare de ...	Camionneur	108
		cave à Beautegord	Mise au canal de ...		
Conditions de Marché	Exemples	Retractions	Payements	Destination des Vins achetés	
1 ^{re} classe, 1 ^{re} rang		Date Quantité	Date Montant	Quantité	
2 ^{de} classe, 1 ^{er} rang					
3 ^{de} classe, 1 ^{er} rang					
4 ^{de} classe, 1 ^{er} rang					
5 ^{de} classe, 1 ^{er} rang					
6 ^{de} classe, 1 ^{er} rang					
7 ^{de} classe, 1 ^{er} rang					
8 ^{de} classe, 1 ^{er} rang					
9 ^{de} classe, 1 ^{er} rang					
10 ^{de} classe, 1 ^{er} rang					
11 ^{de} classe, 1 ^{er} rang					
12 ^{de} classe, 1 ^{er} rang					
13 ^{de} classe, 1 ^{er} rang					
14 ^{de} classe, 1 ^{er} rang					
15 ^{de} classe, 1 ^{er} rang					
16 ^{de} classe, 1 ^{er} rang					
17 ^{de} classe, 1 ^{er} rang					
18 ^{de} classe, 1 ^{er} rang					
19 ^{de} classe, 1 ^{er} rang					
20 ^{de} classe, 1 ^{er} rang					
21 ^{de} classe, 1 ^{er} rang					
22 ^{de} classe, 1 ^{er} rang					
23 ^{de} classe, 1 ^{er} rang					
24 ^{de} classe, 1 ^{er} rang					
25 ^{de} classe, 1 ^{er} rang					
26 ^{de} classe, 1 ^{er} rang					
27 ^{de} classe, 1 ^{er} rang					
28 ^{de} classe, 1 ^{er} rang					
29 ^{de} classe, 1 ^{er} rang					
30 ^{de} classe, 1 ^{er} rang					
31 ^{de} classe, 1 ^{er} rang					
32 ^{de} classe, 1 ^{er} rang					
33 ^{de} classe, 1 ^{er} rang					
34 ^{de} classe, 1 ^{er} rang					
35 ^{de} classe, 1 ^{er} rang					
36 ^{de} classe, 1 ^{er} rang					
37 ^{de} classe, 1 ^{er} rang					
38 ^{de} classe, 1 ^{er} rang					
39 ^{de} classe, 1 ^{er} rang					
40 ^{de} classe, 1 ^{er} rang					
41 ^{de} classe, 1 ^{er} rang					
42 ^{de} classe, 1 ^{er} rang					
43 ^{de} classe, 1 ^{er} rang					
44 ^{de} classe, 1 ^{er} rang					
45 ^{de} classe, 1 ^{er} rang					
46 ^{de} classe, 1 ^{er} rang					
47 ^{de} classe, 1 ^{er} rang					
48 ^{de} classe, 1 ^{er} rang					
49 ^{de} classe, 1 ^{er} rang					
50 ^{de} classe, 1 ^{er} rang					
51 ^{de} classe, 1 ^{er} rang					
52 ^{de} classe, 1 ^{er} rang					
53 ^{de} classe, 1 ^{er} rang					
54 ^{de} classe, 1 ^{er} rang					
55 ^{de} classe, 1 ^{er} rang					
56 ^{de} classe, 1 ^{er} rang					
57 ^{de} classe, 1 ^{er} rang					
58 ^{de} classe, 1 ^{er} rang					
59 ^{de} classe, 1 ^{er} rang					
60 ^{de} classe, 1 ^{er} rang					
61 ^{de} classe, 1 ^{er} rang					
62 ^{de} classe, 1 ^{er} rang					
63 ^{de} classe, 1 ^{er} rang					
64 ^{de} classe, 1 ^{er} rang					
65 ^{de} classe, 1 ^{er} rang					
66 ^{de} classe, 1 ^{er} rang					
67 ^{de} classe, 1 ^{er} rang					
68 ^{de} classe, 1 ^{er} rang					
69 ^{de} classe, 1 ^{er} rang					
70 ^{de} classe, 1 ^{er} rang					
71 ^{de} classe, 1 ^{er} rang					
72 ^{de} classe, 1 ^{er} rang					
73 ^{de} classe, 1 ^{er} rang					
74 ^{de} classe, 1 ^{er} rang					
75 ^{de} classe, 1 ^{er} rang					

Computers offered data storage and computation...
Could the spreadsheet be electrified?





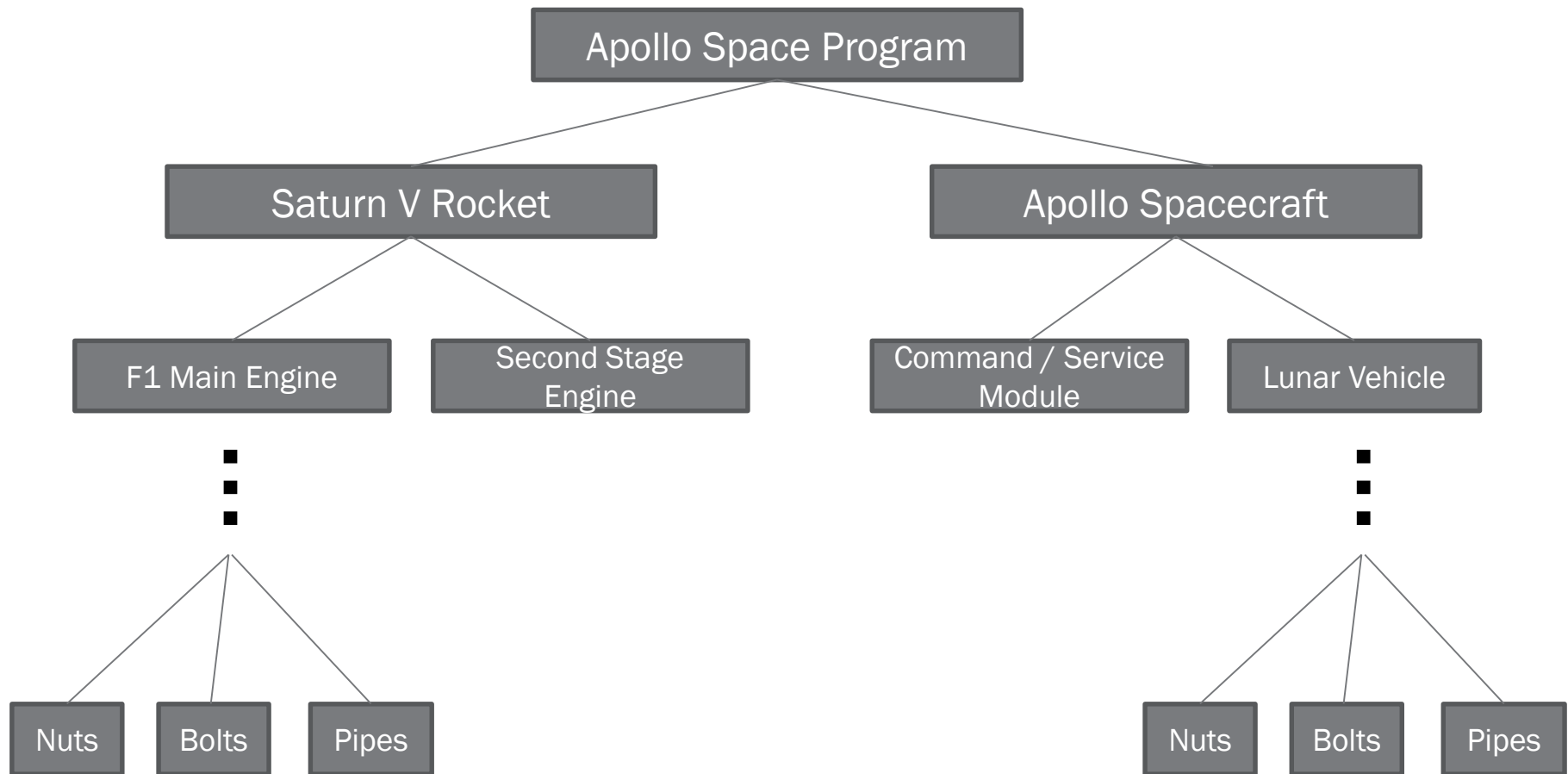


Information Management System

3 million parts
inventoried!

1966: First Database Management System (DBMS)

IMS had a hierarchical “tree” structure, internally



(This is not really how it was organized)

Tree structures are fast and efficient!

Hierarchical Database Disadvantages:

- Developers MUST understand internal database structure
- Which means you really don't want to fundamentally alter the database
- Because that will break your existing applications that use it

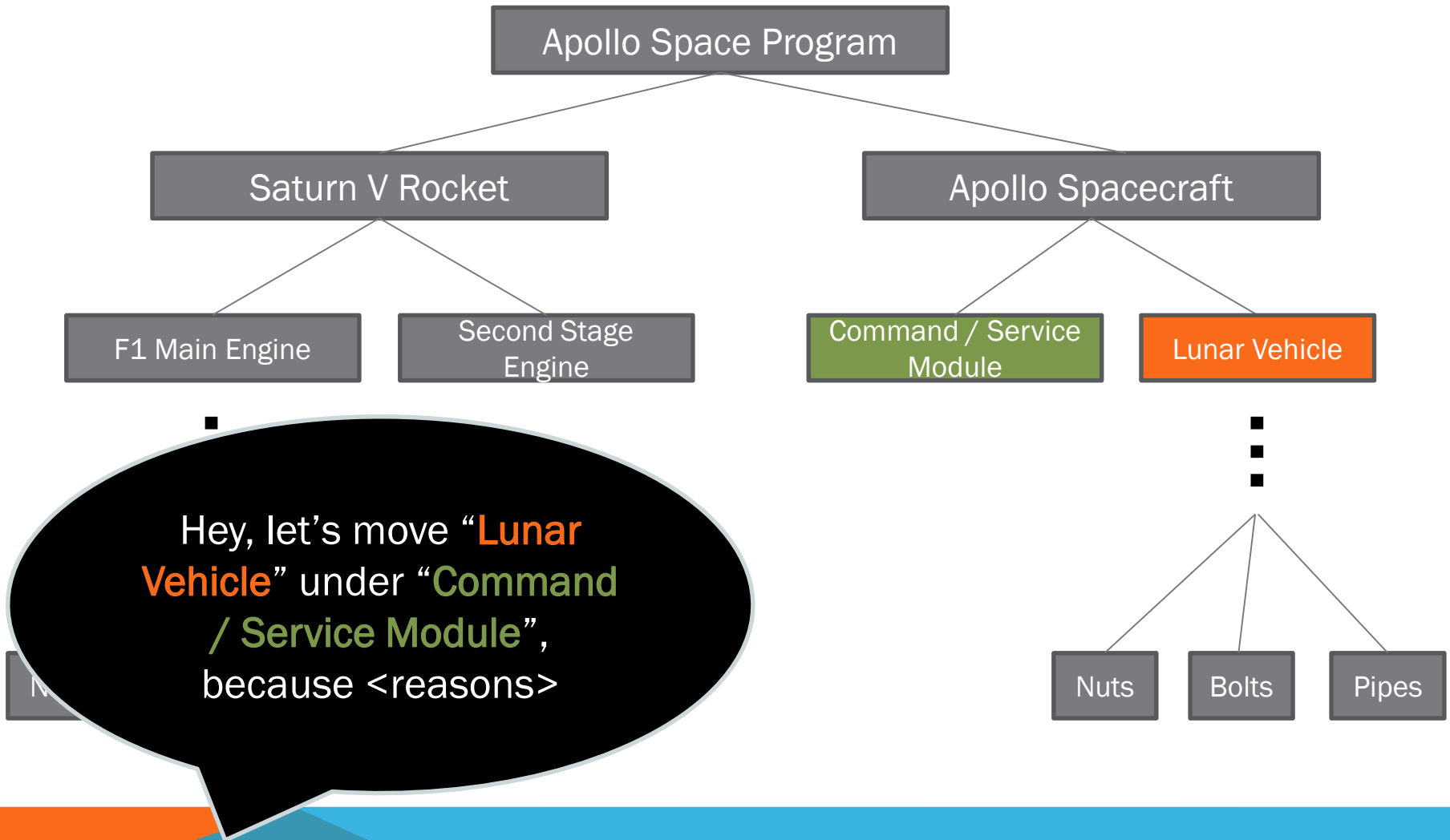
Hierarchical Database Disadvantages:

- Developers MUST understand internal database structure
- Which means you really don't want to fundamentally alter the database
- Because that will break your existing applications that use it

Hierarchical Database Disadvantages:

- Developers MUST understand internal database structure
- Which means you really don't want to fundamentally alter the database
- Because that will break your existing applications that already use it

You Really Don't Want to Fundamentally Alter a Hierarchical Database



But ***what if*** there were a database model...

But ***what if*** there were a database model...

...where expensive database
applications ***continued to work...***

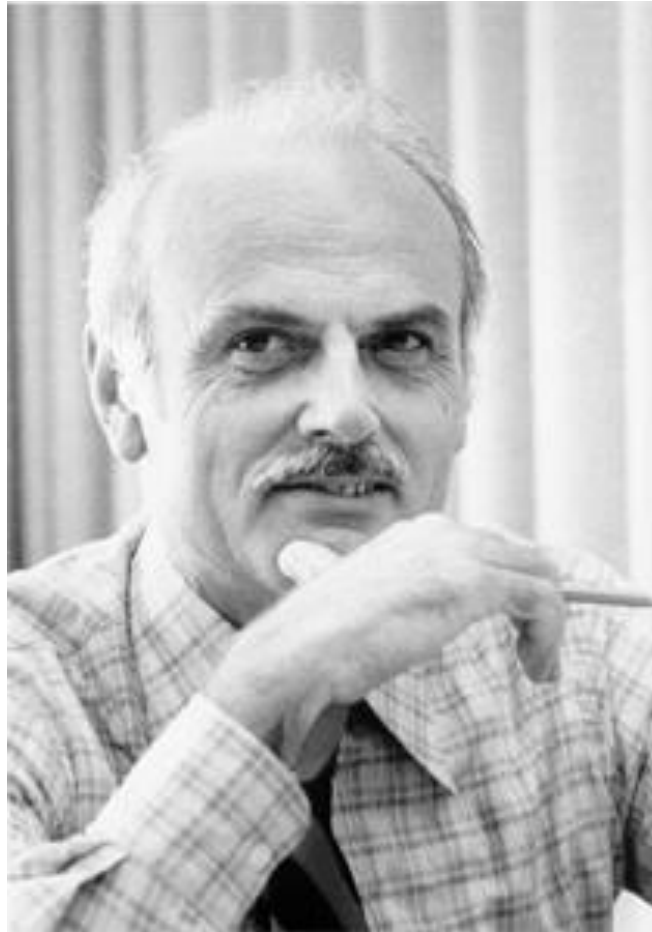
But ***what if*** there were a database model...

...where expensive database
applications ***continued to work...***

***...even if the internal database
structure were altered?***

The bottom of the slide features a decorative graphic consisting of several overlapping geometric shapes. On the left, there is a bright orange triangle pointing upwards. To its right is a light blue triangle pointing downwards. Further right, there is a darker blue triangle pointing upwards. The overall effect is a modern, abstract design at the bottom of the presentation slide.

In 1970, Dr. Edgar Codd invents:
“*Relational* model for database management”



A Relational Model of Data for Large Shared Data Banks

E. F. Codd

IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation

Relational Databases Use Table Structures, Not Trees

Schema

Attributes (Columns)

Unity ID	Name	Email	Phone
jajerni2	John	jajerni2@ncsu.edu	919.513.1666
bwbarbou	Brandon	bwbarbou@ncsu.edu	919.515.0706
<u>avillan</u>	Andrea	<u>avillan@ncsu.edu</u>	919.515.7106

Tuple (Row)

Use Structured Query Language (SQL) to Interface with Database

Unity ID	Name	Email	Phone
jajerni2	John	jajerni2@ncsu.edu	919.513.1666
bwbarbou	Brandon	bwbarbou@ncsu.edu	919.515.0706

```
mysql> SELECT Name,Email FROM staff WHERE 'Unity ID'='jajerni2';
```

```
+-----+-----+
| Name   | Email                               |
+-----+-----+
| John   | jajerni2@ncsu.edu |
+-----+-----+
```

Creating the First Relational Databases...



Creating the First Relational Databases...

Ed Oates

Bob Miner

Larry Ellison



Creating the First Relational Databases...

The word "ORACLE" is rendered in a large, pixelated, monospaced font, typical of early computer graphics. Each letter is composed of small squares, giving it a blocky, digital appearance.

ORACLE Database Management System

(c) Copyright Oracle Corporation, 1984.

All Rights Reserved.

This software has been provided under a license agreement
containing certain restrictions on use and disclosure.
Reverse engineering of object code is prohibited.


Press Any Key To Continue..._

Creating the First Relational Databases...


The Oracle logo is centered within a large red square. The word "ORACLE" is written in a white, bold, sans-serif font. A registered trademark symbol (®) is located at the top right of the letter "E".

ORACLE®


Some Top Relational Database Management Systems

- **Oracle**
 - Microsoft SQL Server
 - MySQL (free, open-source; see: MariaDB)
 - Postgres (free, open-source)
- 


Some Top Relational Database Management Systems

- Oracle
 - **Microsoft SQL Server**
 - MySQL (free, open-source; see: MariaDB)
 - Postgres (free, open-source)
- 

Some Top Relational Database Management Systems

- Oracle
 - Microsoft SQL Server
 - **MySQL (free, open-source; see: MariaDB)**
 - Postgres (free, open-source)
- 

Some Top Relational Database Management Systems

- Oracle
 - Microsoft SQL Server
 - MySQL (free, open-source; see: MariaDB)
 - **PostgreSQL (free, open-source)**
- 

Special Mention of Popular Database:



- Stripped-down
- Bare-essentials
- Still powerful (also: free)

What's Wrong with Relational Databases?



What's Wrong with Relational Databases?



- Not much

What's Wrong with Relational Databases?



- Not much
- Until you start dealing with Big Data

What's Wrong with a Honda Fit?



- Not much
- Until you start dealing with Big Cargo

What's Wrong with a Honda Fit?



- Not much
- Until you start dealing with Big Cargo

CLASSIC “BIG DATA” DEFINITION



What Can Go Wrong With Relational Databases:
Big Data

Database Too Slow?

Scale Resources **Vertically**



What Can Go Wrong With Relational Databases:
Poor Performance

Database Too Slow?

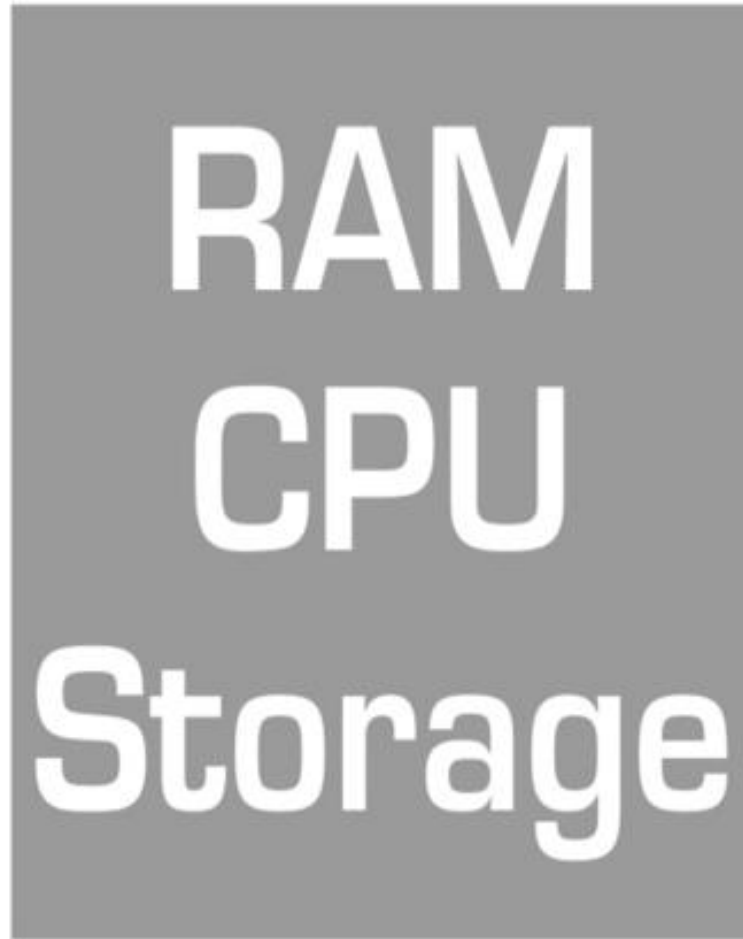
Scale Resources Vertically



What Can Go Wrong With Relational Databases:
Poor Performance

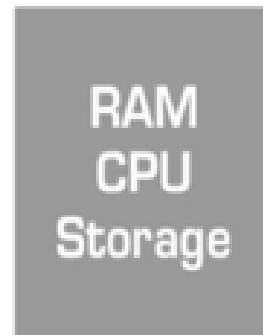
Database Too Slow?

Scale Resources Vertically



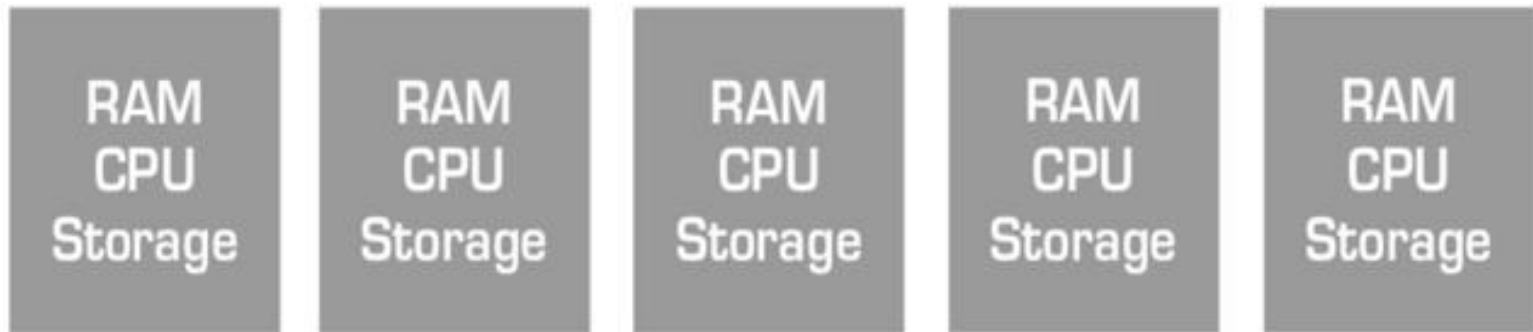
What Can Go Wrong With Relational Databases:
Poor Performance

Or ... **Scale** Resources **Horizontally**



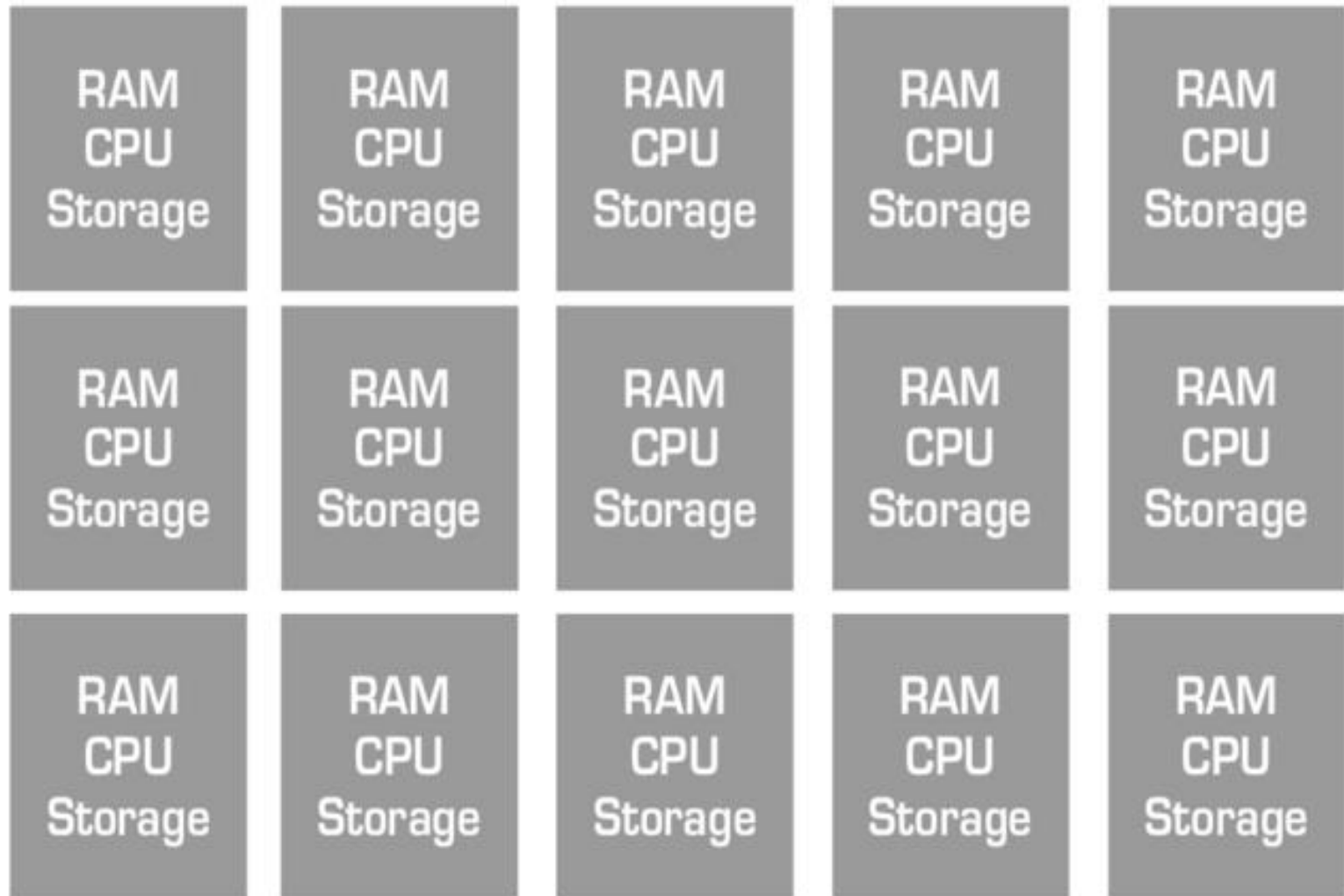
What Can Go Wrong With Relational Databases:
Poor Performance

Or ... **Scale** Resources **Horizontally**



What Can Go Wrong With Relational Databases:
Poor Performance

Or ... **Scale Resources Horizontally**



What Can Go Wrong With Relational Databases:
Poor Performance

Scaling Vertically: Limited by physics, and
state-of-the-art

Scaling Horizontally: Works! But ...
presents new problems

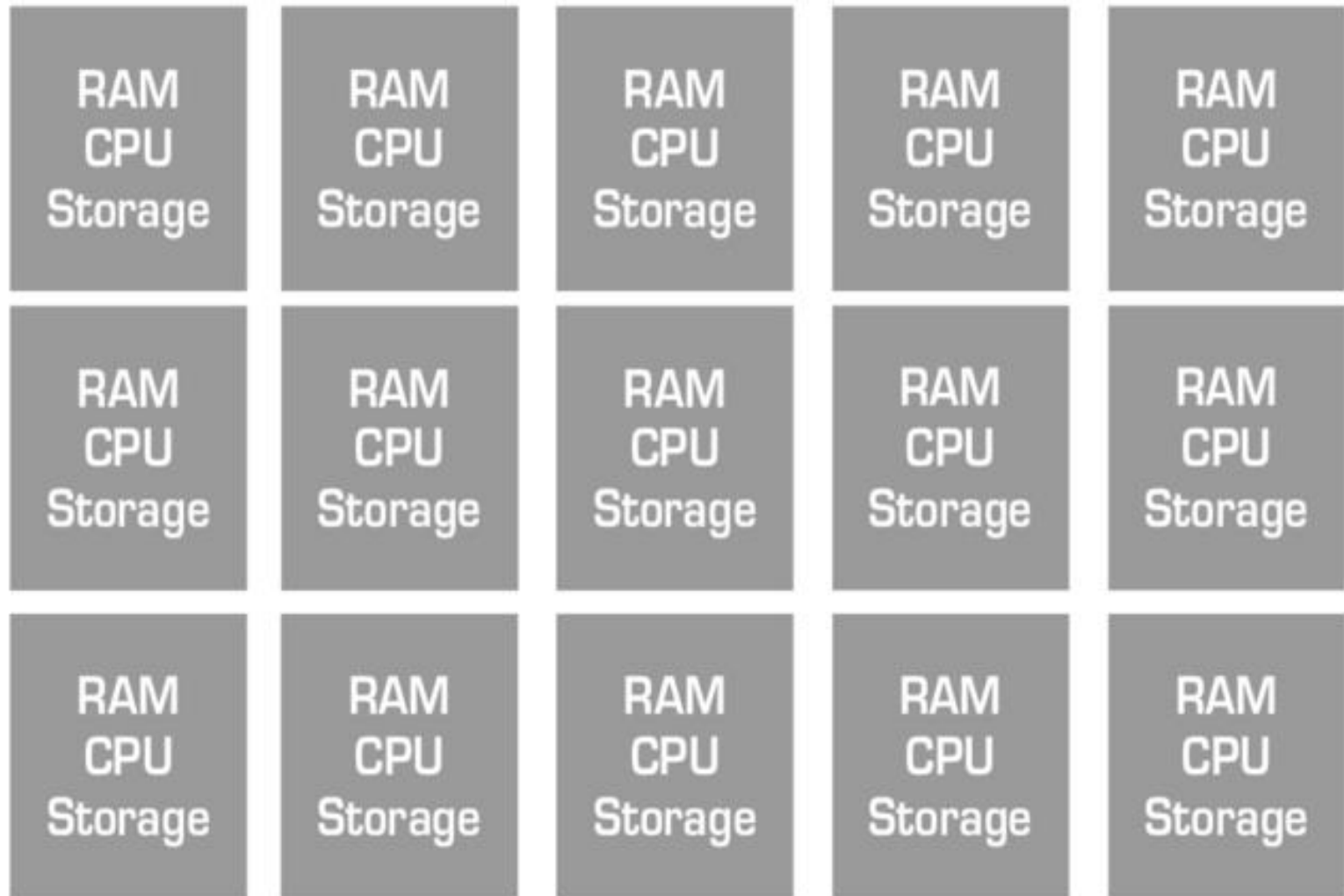
What Can Go Wrong With Relational Databases:
Poor Performance

Scaling Vertically: Limited by physics, and
state-of-the-art

Scaling Horizontally: Works! But ...
presents new problems

What Can Go Wrong With Relational Databases:
Poor Performance

Consider the synchronization issues when updating the database simultaneously on multiple nodes



What Can Go Wrong With Relational Databases:
Poor Performance

Engineers attempted to build
better performing databases.
They were called...



NoSQL - A misleading term

Have you heard the term NoSQL?



NoSQL - A misleading term

NoSQL?

It *seems* to mean
“not a relational database”

NoSQL - A misleading term

A better way of interpreting it:



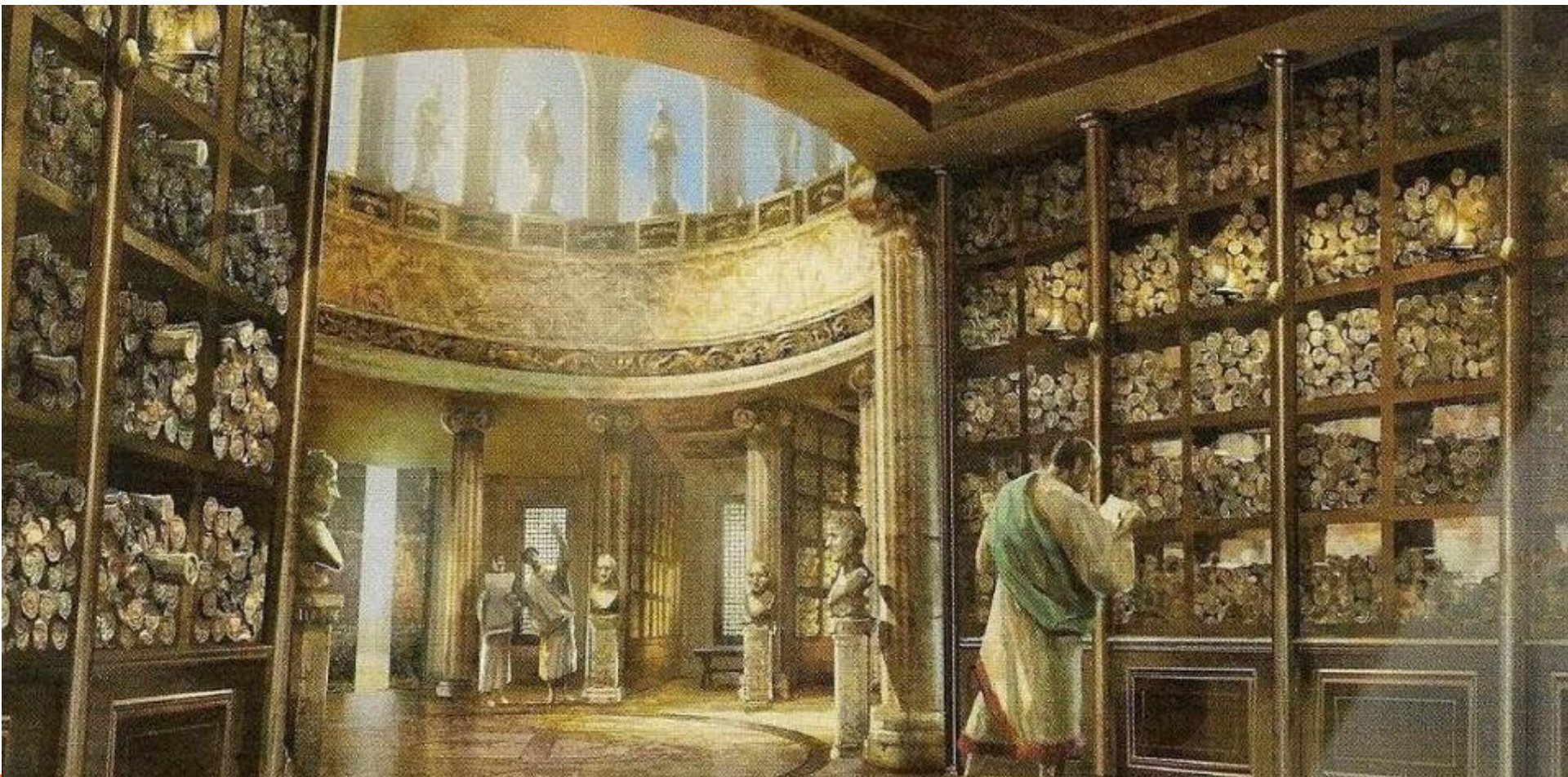
Not Only SQL

So many databases...
Why?

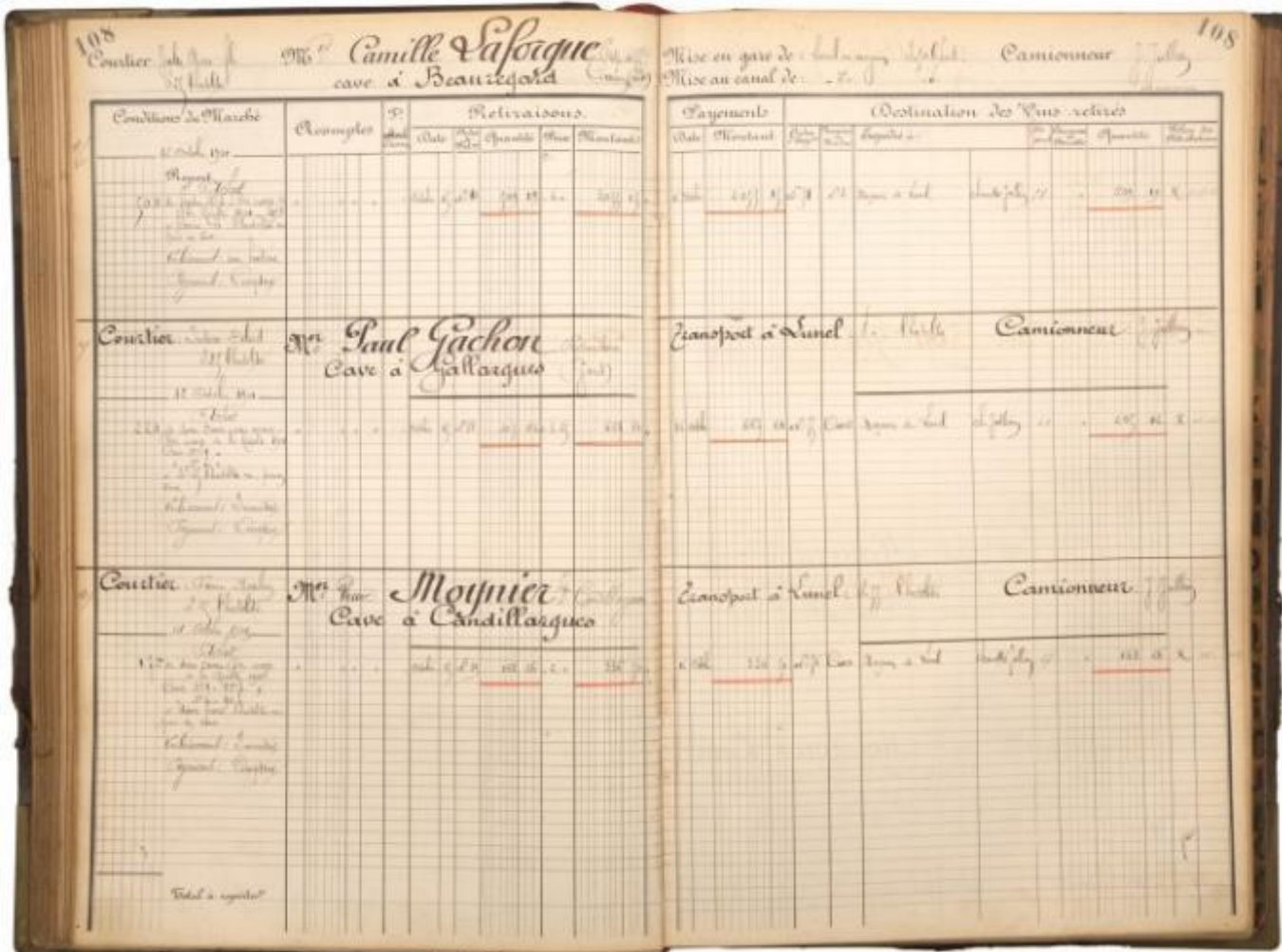


Library of Alexandria

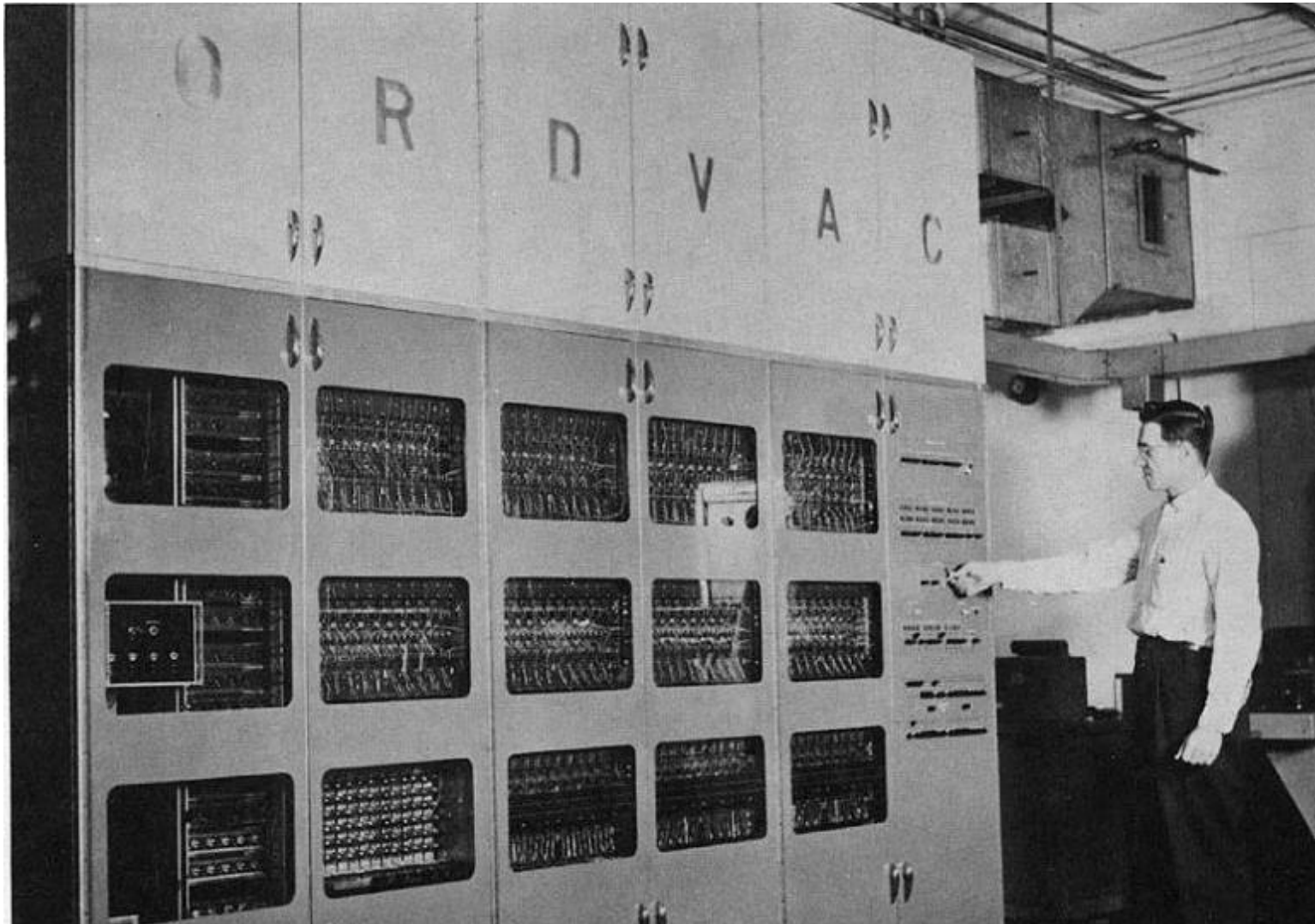
First use of alphabetical ordering?



Casson, L. (2001). Libraries in the ancient world. ProQuest Ebook Central (pg 37)

[illegible]

Computers offered data storage and computation...
Could the spreadsheet be electrified?



First Relational Database Management System (ca. 1979)



ORACLE

ORACLE Database Management System

(c) Copyright Oracle Corporation, 1984.

All Rights Reserved.

This software has been provided under a license agreement
containing certain restrictions on use and disclosure.
Reverse engineering of object code is prohibited.

Press Any Key To Continue..._

What's Wrong with Relational Databases?



- Not much
- Until you start dealing with Big Data

NoSQL - A misleading term

A better way of interpreting it:



Not Only SQL

NoSQL - A misleading term

Think: Solving Big Data database challenges
with *application-specific solutions*.



NoSQL - A misleading term

Think: Solving Big Data database challenges
with *application-specific solutions*.

Working with big JSON data?
Try:



NoSQL - A misleading term

Think: Solving Big Data database challenges
with *application-specific solutions*.

Working with big key-value pairs?
Try:



NoSQL - A misleading term

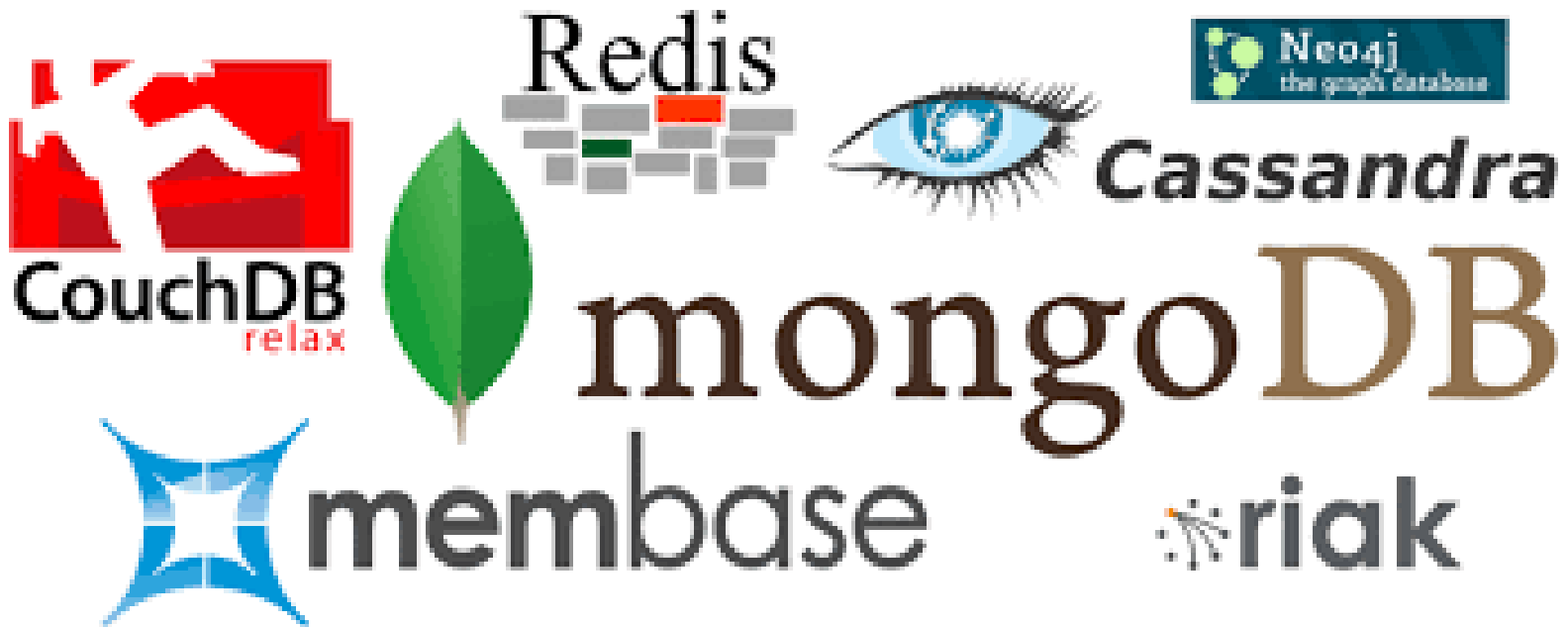
Think: Solving Big Data database challenges
with *application-specific solutions*.

Working with big graph data?
Try:



NoSQL - A misleading term

Think: Solving Big Data database challenges
with *application-specific solutions*.

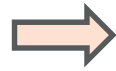


Recap: The Big Picture of Relational Databases and NoSQL

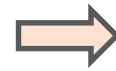


Recap: The Big Picture of Relational Databases and NoSQL

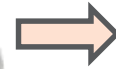
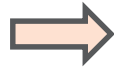
IBM IMS



Relational Database
Management Systems
(RDBMS)



NoSQL Database
Management Systems



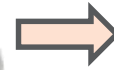
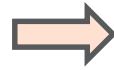
Recap: The Big Picture of Relational Databases and NoSQL

Gartner “Strategic Planning Assumption”:

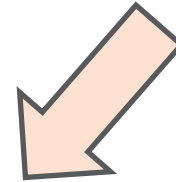
By 2017, all leading operational DBMSs will offer multiple data models, relational and NoSQL, in a single DBMS platform.



RDBMS



NoSQL

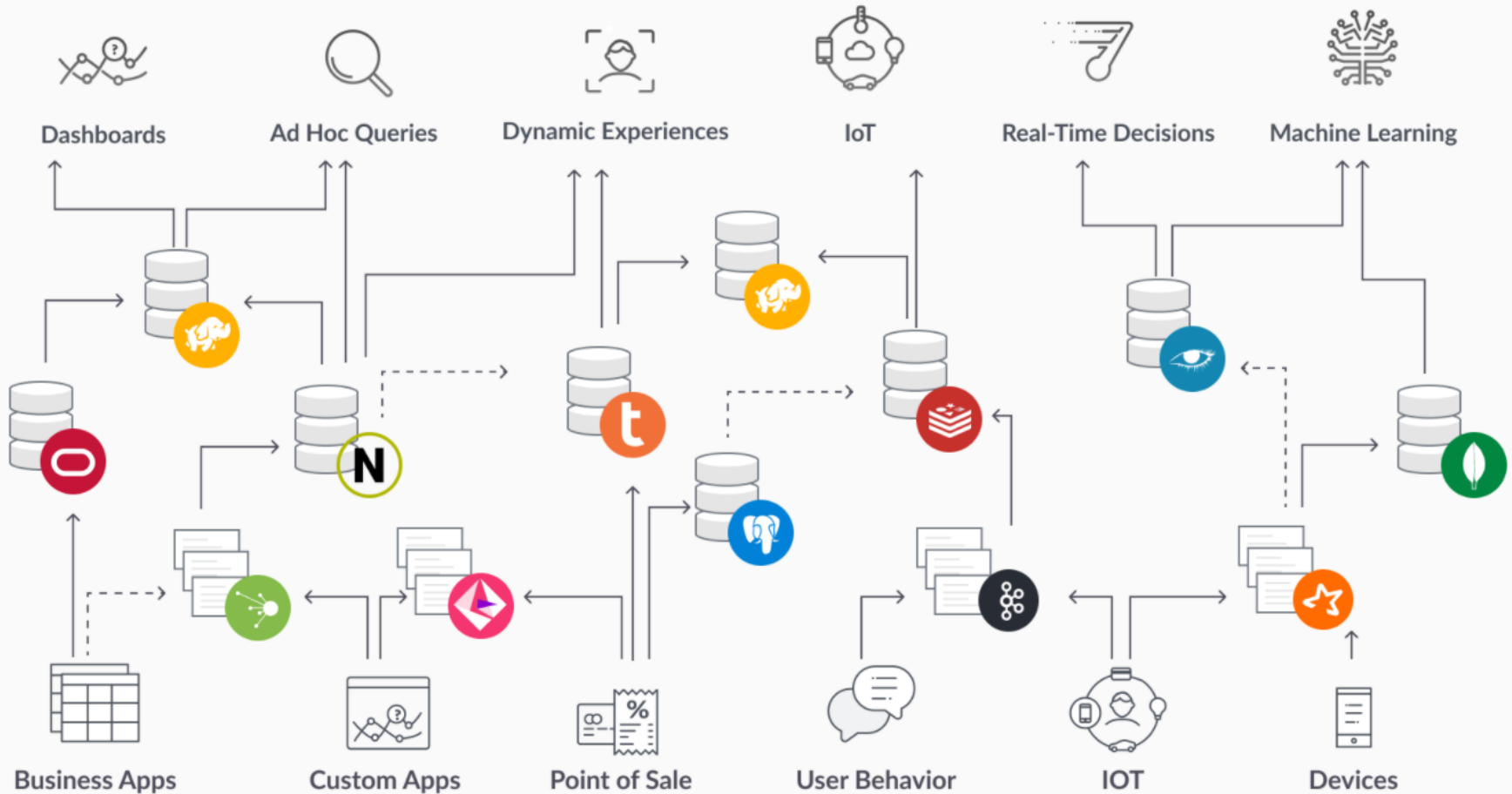


RDBMS + NoSQL =
Flying Car



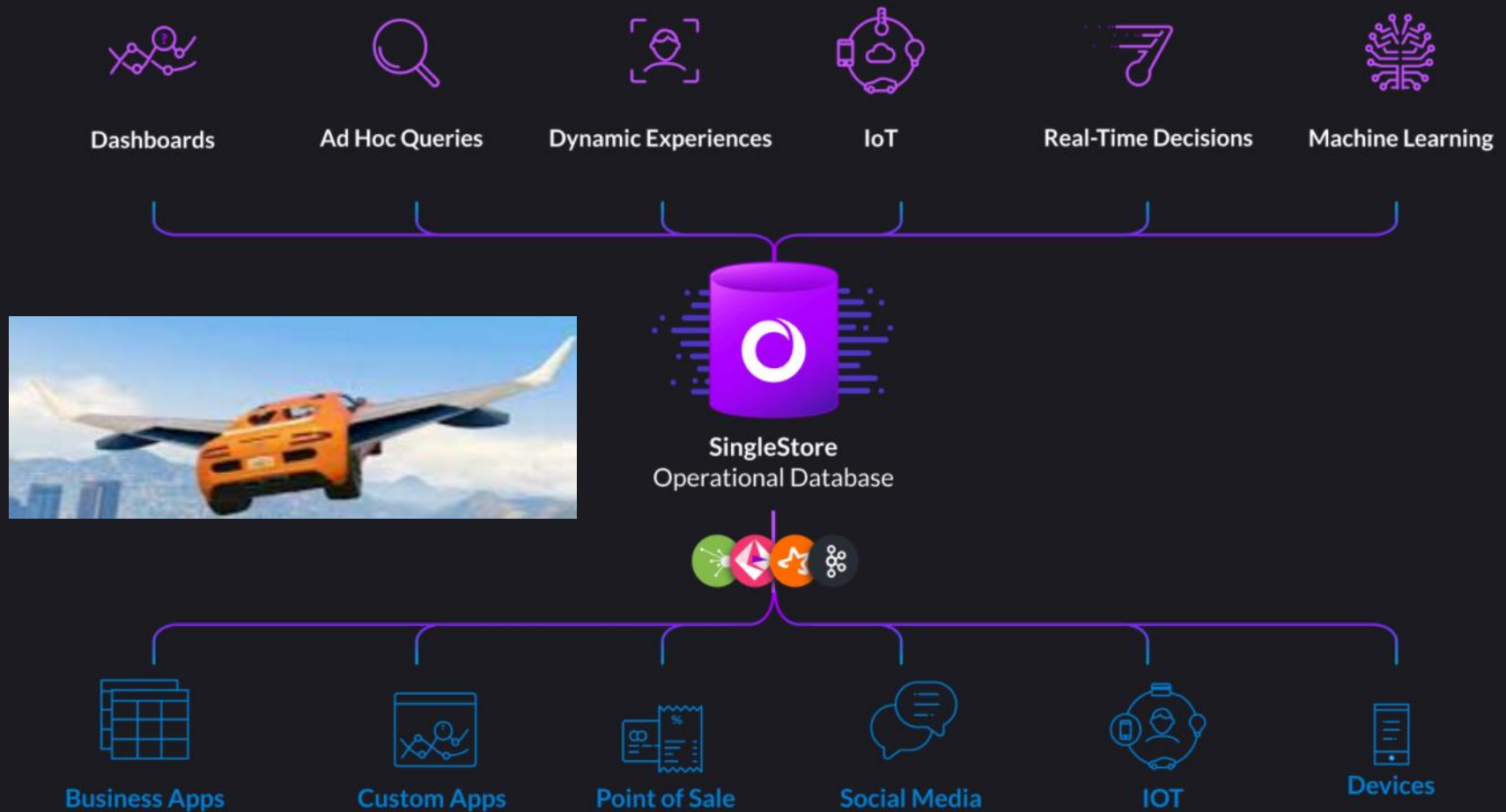
Trend is moving away from all these different database solutions...

Legacy Architecture



Instead, seeing data storage solutions try to “do it all”

Modern Architecture



PRACTICUM TIP: USE INDEXING

If you have a SAS7BDAT dataset...



PRACTICUM TIP: USE INDEXING

If you have a SAS7BDAT dataset...

Or your data is inside MySQL / Postgres / MSSQL...



PRACTICUM TIP: USE INDEXING

If you have a SAS7BDAT dataset...

Or your data is inside MySQL / Postgres / MSSQL...

**You NEED to CONSIDER where to
create indexes for performance purposes.**

PRACTICUM TIP: USE INDEXING

✓ *Dramatically* decrease query times!

E.g. 1 hour → 2.4 seconds

I have seen this... that is a 1500x speedup.

PRACTICUM TIP: USE INDEXING

- ✓ *Dramatically* decrease query times!
- ✓ *Speed up* time to sort data!

E.g. 1 hour → 2.4 seconds

I have seen this... that is a 1500x speedup.

PRACTICUM TIP: USE INDEXING

- Indexes are created per-column (*attribute*)

e.g.

- Employee Number
- Purchase Price
- Transaction Date

Index them all!

PRACTICUM TIP: USE INDEXING

- Indexes are created per-column (*attribute*)
- Indexes can hugely speed up your queries!

PRACTICUM TIP: USE INDEXING

- Indexes are created per-column (*attribute*)
- Indexes can hugely speed up your queries!
- Index updates are performance overhead when data is added to database which is a tradeoff to consider.

MSA Students:

Don't worry about this!

Real-World Production Systems: Worry about this!

Consider Which Columns You are Querying Against

E.g. will you be searching the table by Name?
Unity ID? All columns?

Schema

Attributes (Columns)

Unity ID	Name	Email	Phone
jajerni2	John	jajerni2@ncsu.edu	919.513.1666
bwbarbou	Brandon	bwbarbou@ncsu.edu	919.515.0706
<u>avillan</u>	Andrea	avillan@ncsu.edu	919.515.7106

Tuple (Row)

Understand Database Indexing



Understand Database Indexing

With an index present, queries CAN dramatically improve.

Understand Database Indexing

Indexes help with “finding a needle in a haystack”.

They do not help (and do not hurt) if you're going to extract most of the database anyway:

```
SELECT COUNT(Unity_ID) FROM directory;
```

(this returns the number of NCSU students in a directory)

Index Your Database Columns!

As a data scientist...

... if you use SQL queries in a database...

As a data scientist...

... if you use SQL queries in a database...

... which includes proc sql on a SAS7BDAT file ...

... or SQL with PostgreSQL ...

Index Your Database Columns!

As a data scientist...

... if you use SQL queries in a database...

... which includes proc sql on a SAS7BDAT file ...

... or SQL with PostgreSQL ...

... learn how to turn on indexing for any relevant data columns.

Index Your Database Columns!

As a data scientist...

- ... if you use SQL queries in a database...

 - ... which includes proc sql on a SAS7BDAT file ...

 - ... or SQL with PostgreSQL ...

... learn how to turn on indexing for any relevant data columns.

(e.g. index “date” columns for transactional data)

Index Your Database Columns!

As a data scientist...

... if you use SQL queries in a database...

... which includes proc sql on a SAS7BDAT file ...

... or SQL with PostgreSQL ...

... learn how to turn on indexing for any relevant data columns.

(e.g. index “date” columns for transactional data)

This will save you time whenever you’re looking for a small bit of data in a large dataset...

... but it’s up to YOU to turn on indexing.

For more info on indexing...

- **SAS Indexes:** <http://www2.sas.com/proceedings/sugi29/123-29.pdf>
Also: SAS Programming 3 book, section 3-5
- **PostgreSQL Indexes:**
<https://www.postgresql.org/docs/current/static/indexes.html>
- **General Info:** https://en.wikipedia.org/wiki/Database_index

Questions



Bonus: PostgreSQL Demo

Never had any hands-on experience writing SQL code against a simple database?

Try this browser-based Postgres database with some sample data loaded:

- <https://www.crunchydata.com/developers/playground?sql=https://gist.githubusecontent.com/eccentric-j/11cce77331a82bcab52563d1f63a9c46/raw/create-births-table.sql>
- List the tables available:
 - \dt (think: display tables)

```
postgres=# \dt
          List of relations
 Schema |          Name          | Type  | Owner
-----+-----+-----+-----
 public | us_births_20002014_ssa | table | postgres
(1 row)
```

This table contains US number of births for each day of the year between 2000 – 2014, provided by the Social Security Administration.

Bonus: PostgreSQL Demo

- Describe the table schema (i.e. tell me what columns it has)
 - `\d us_births_20002014_ssa` (think: `describe us_births_20002014_ssa`)

```
postgres=# \d us_births_20002014_ssa
Table "public.us_births_20002014_ssa"
  Column      | Type      | Collation | Nullable | Default
-----+-----+-----+-----+-----
 id           | integer   |           | not null | nextval('us_births_20002014_ssa_id_seq'::regclass)
 year         | integer   |           |          |
 month        | integer   |           |          |
 date_of_month | integer   |           |          |
 day_of_week   | integer   |           |          |
 births       | integer   |           |          |
Indexes:
    "us_births_20002014_ssa_pkey" PRIMARY KEY, btree (id)
```

Hint: use the Tab key to auto-complete the table name instead of typing it in every command. Type `"\d us"` and then hit Tab.

Bonus: PostgreSQL Demo

- Display the first 10 rows of data
 - `SELECT * FROM us_births_20002014_ssa LIMIT 10;`

```
postgres=# SELECT * FROM us_births_20002014_ssa LIMIT 10;
 id | year | month | date_of_month | day_of_week | births
-----+-----+-----+-----+-----+-----
  1 | 2000 |     1 |             1 |           6 |    9083
  2 | 2000 |     1 |             2 |           7 |    8006
  3 | 2000 |     1 |             3 |           1 |   11363
  4 | 2000 |     1 |             4 |           2 |   13032
  5 | 2000 |     1 |             5 |           3 |   12558
  6 | 2000 |     1 |             6 |           4 |   12466
  7 | 2000 |     1 |             7 |           5 |   12516
  8 | 2000 |     1 |             8 |           6 |    8934
  9 | 2000 |     1 |             9 |           7 |    7949
 10 | 2000 |     1 |            10 |           1 |   11668
(10 rows)
```

Looks like fewer births
on Saturday and
Sunday (day_of_week
6 and 7)...

Hint: use the Tab key to
auto-complete the table
name instead of typing it
in every command. Type
“us” and then hit Tab.

Bonus: PostgreSQL Demo

- **Some other ideas...**
 - Count how many rows of data there are:
 - `SELECT count(births) FROM us_births_20002014_ssa;`
 - Find out what year is the latest year in the data:
 - `SELECT MAX(year) FROM us_births_20002014_ssa;`
 - Find the average number of births on Sunday across all data:
 - `SELECT AVG(births) FROM us_births_20002014_ssa WHERE day_of_week='7';`
 - Find the average number of births on Monday across all data:
 - `SELECT AVG(births) FROM us_births_20002014_ssa WHERE day_of_week='1';`

Interesting...fewer births on the weekend consistently over 14 years...

Bonus: PostgreSQL Demo

- Display the first 10 rows of data
 - `SELECT * FROM us_births_20002014_ssa LIMIT 10;`

```
postgres=# SELECT * FROM us_births_20002014_ssa LIMIT 10;
 id | year | month | date_of_month | day_of_week | births
-----+-----+-----+-----+-----+-----
  1 | 2000 |     1 |             1 |           6 |   9083
  2 | 2000 |     1 |             2 |           7 |   8006
  3 | 2000 |     1 |             3 |           1 |  11363
  4 | 2000 |     1 |             4 |           2 |  13032
  5 | 2000 |     1 |             5 |           3 |  12558
  6 | 2000 |     1 |             6 |           4 |  12466
  7 | 2000 |     1 |             7 |           5 |  12516
  8 | 2000 |     1 |             8 |           6 |   8934
  9 | 2000 |     1 |             9 |           7 |   7949
 10 | 2000 |     1 |            10 |           1 |  11668
(10 rows)
```

Looks like fewer births
on Saturday and
Sunday (day_of_week
6 and 7)...

Hint: use the Tab key to
auto-complete the table
name instead of typing it
in every command. Type
“us” and then hit Tab.