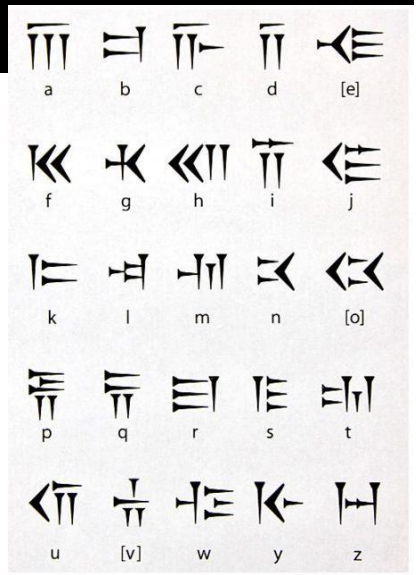
History and Evolution of Databases

CS 341 Database Systems



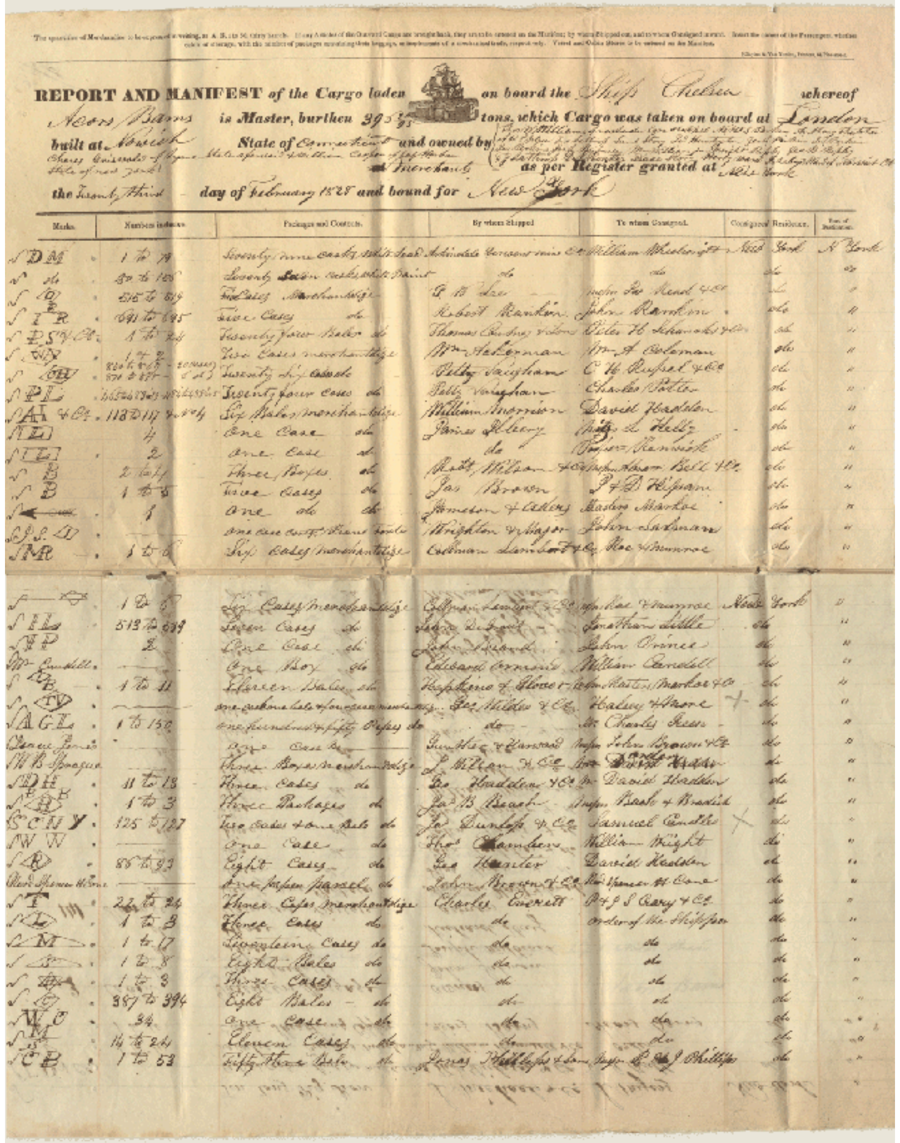
First Records

The first written records date back a little more than 5,000 years in Egypt and ancient Sumer.

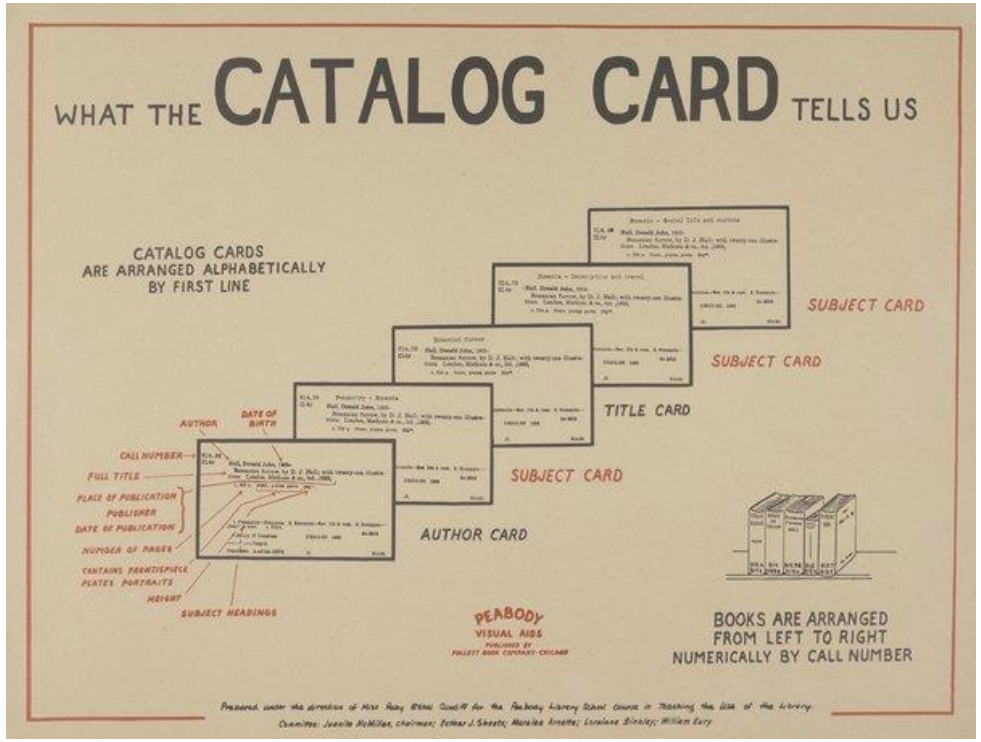
The earliest Sumerian records were made using reeds cut at an angle to make wedge-shaped (cuneiform) marks on clay, which was then baked hard.

The earliest records look like accounts: lists of property, cattle, sheep, and wheat.

1800s Ship Passengers Manifests



Card Catalogs



Card Catalogs at IBA

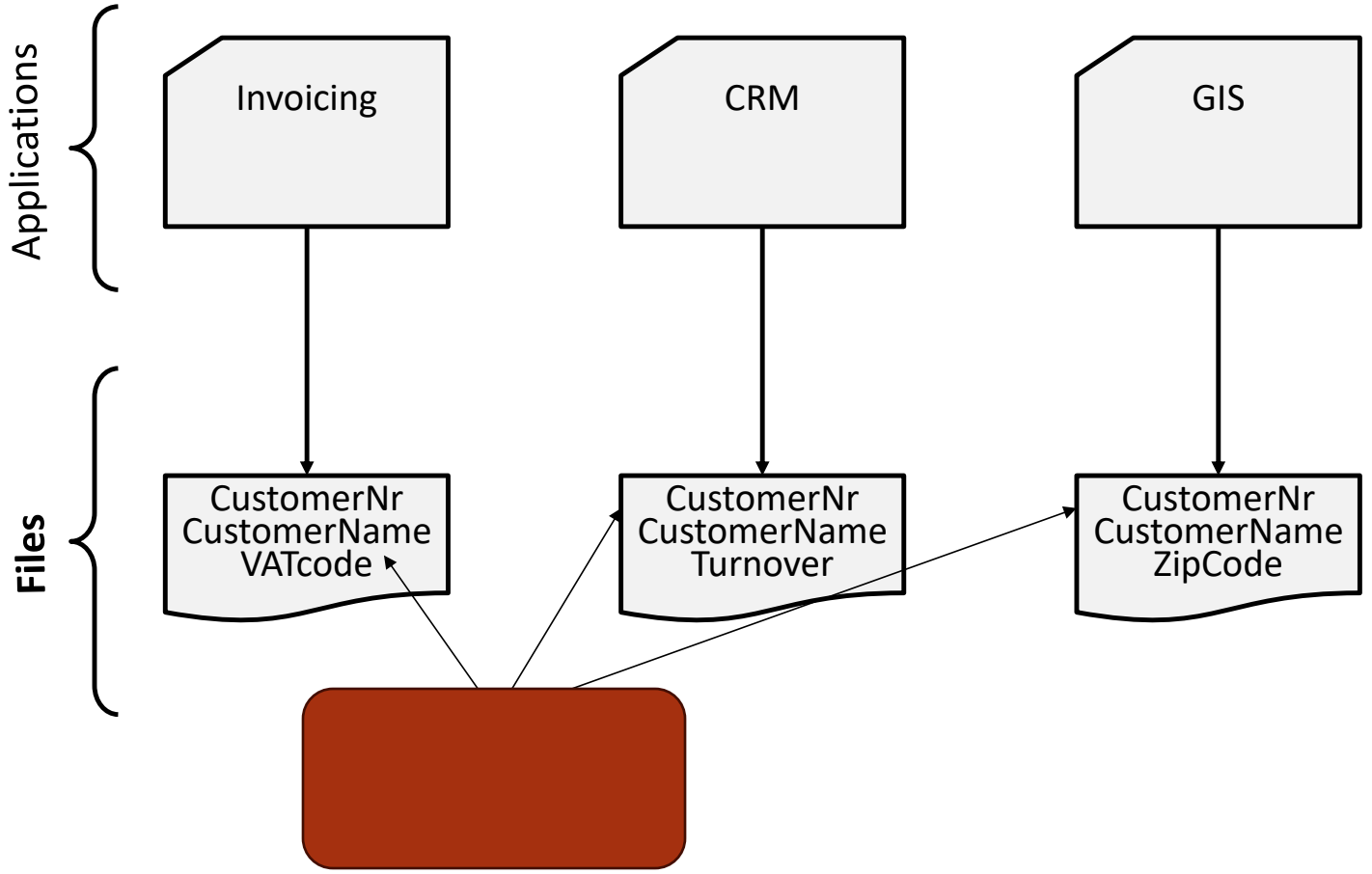
Predecessor to Databases

File-based Systems around 1960s

- Collection of application programs that perform services for the end-users (e.g., reports).

- Each program defines and manages its own data.

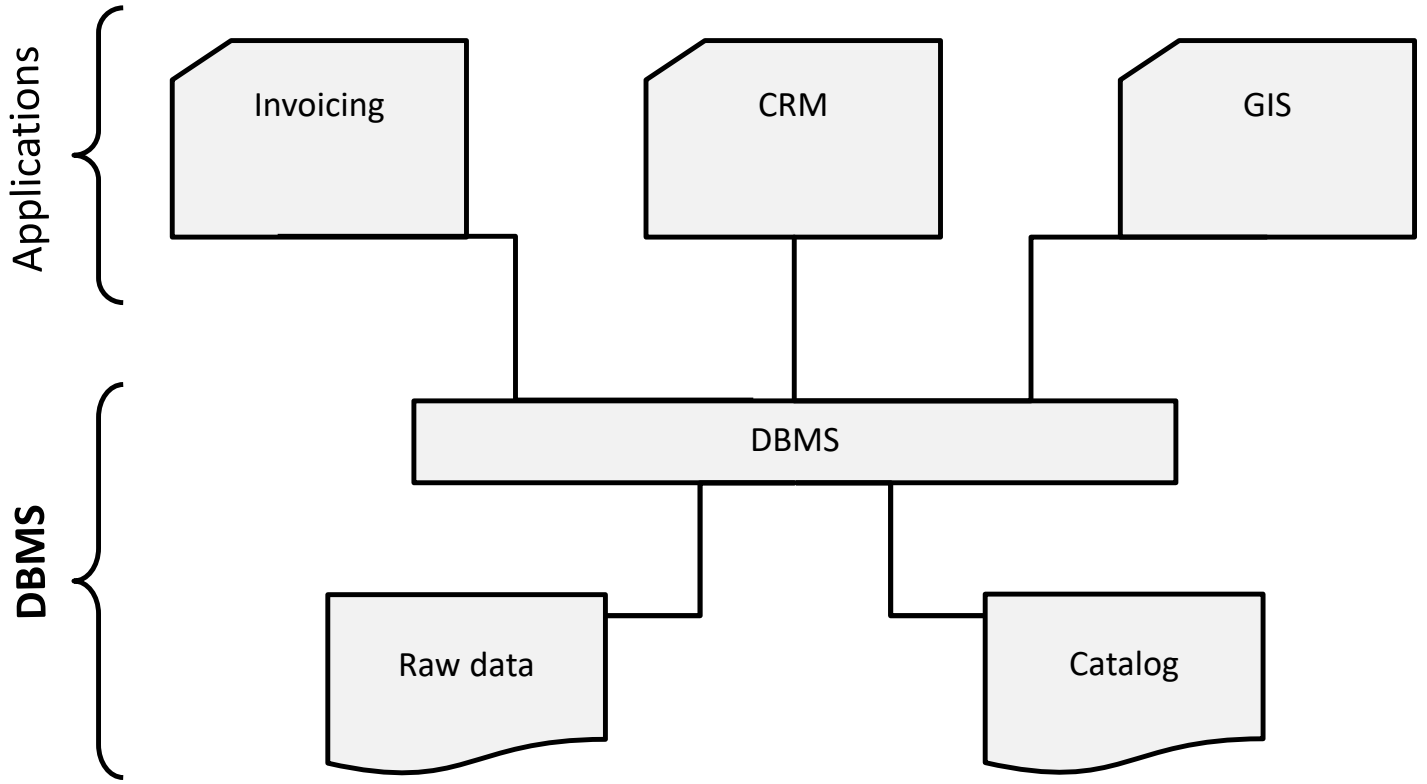
File based Approach to Data Management
(Example #1)



File versus Database Approach to Data Management

- **File Approach**
  - *duplicate* or *redundant* information will be stored
  - danger of *inconsistent* data
  - strong coupling between applications and data
  - hard to manage concurrency control
  - hard to integrate applications aimed at providing cross-company services

File versus Database Approach to Data Management



File versus Database Approach to Data Management

- **Database approach**
  - superior to the file approach in terms of *efficiency*, *consistency* and *maintenance*
  - loose coupling between applications and data
  - facilities provided for data querying and retrieval

File versus Database Approach to Data Management

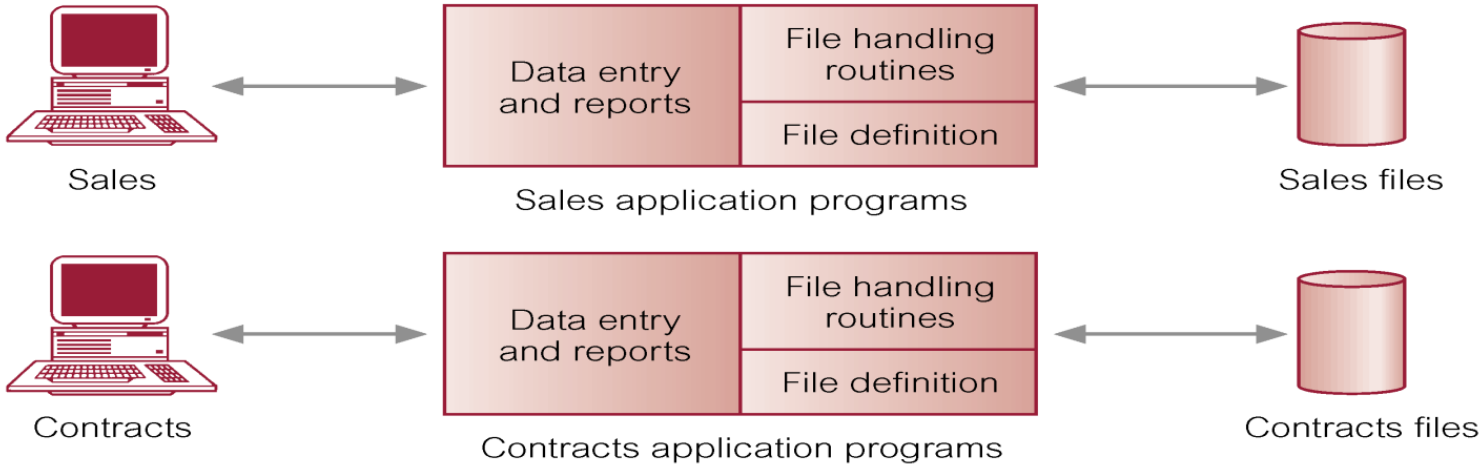
File approach

```
Procedure FindCustomer;
begin
     open file Customer.txt;
Read(Customer)
While not EOF(Customer)
If Customer.name='Bart' then
display(Customer);
EndIf
Read(Customer);
EndWhile;
End;
```

Database approach (SQL)

```
SELECT *
FROM Customer
WHERE
name = 'Bart'
```

File-based Processing



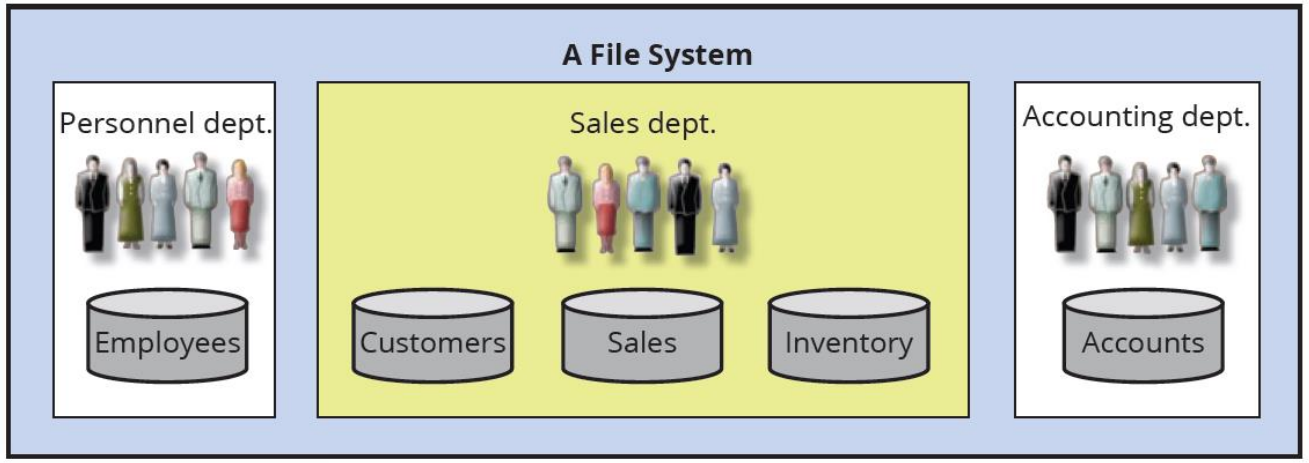
Sales Files

PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)

Contracts Files

Lease (leaseNo, propertyNo, clientNo, rent, paymentMethod, deposit, paid, rentStart, rentFinish, duration)
PropertyForRent (propertyNo, street, city, postcode, rent)
Client (clientNo, fName, lName, address, telNo)

Limitations of File-based Approach



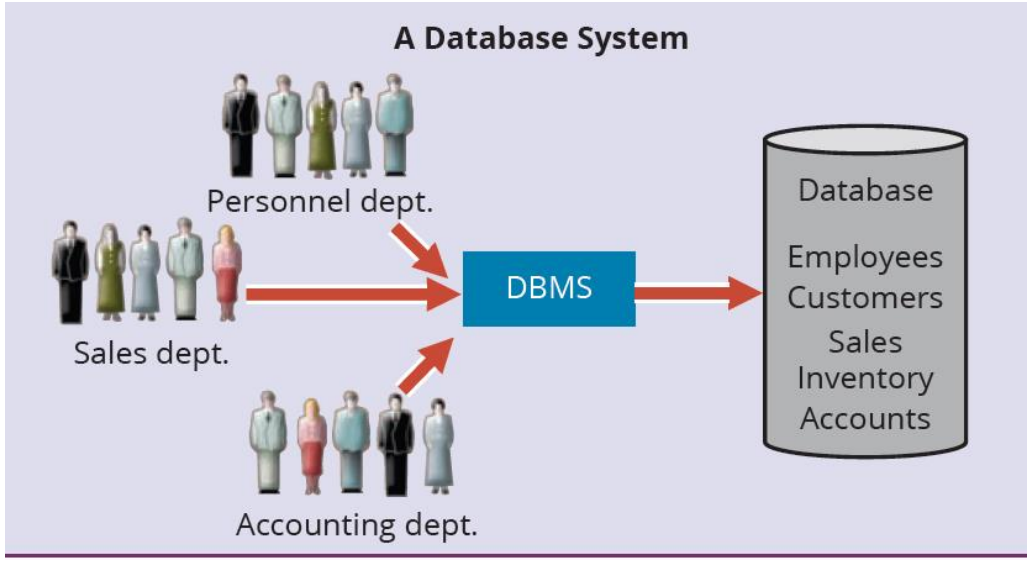
- **Separation and isolation of data**
  - Each program maintains its own set of data.
  - Users of one program may be unaware of potentially useful data held by other programs.
- **Duplication of data**
  - Same data is held by different programs.
  - Wasted space and potentially different values and/or different formats for the same item.

Limitations of File-based Approach

- **Data dependence**
  - File structure is defined in the program code.
- **Incompatible file formats**
  - Programs are written in different languages, and so cannot easily access each other's files.
- **Fixed Queries/Proliferation of application programs**
  - Programs are written to satisfy particular functions. Any new requirement needs a new program.

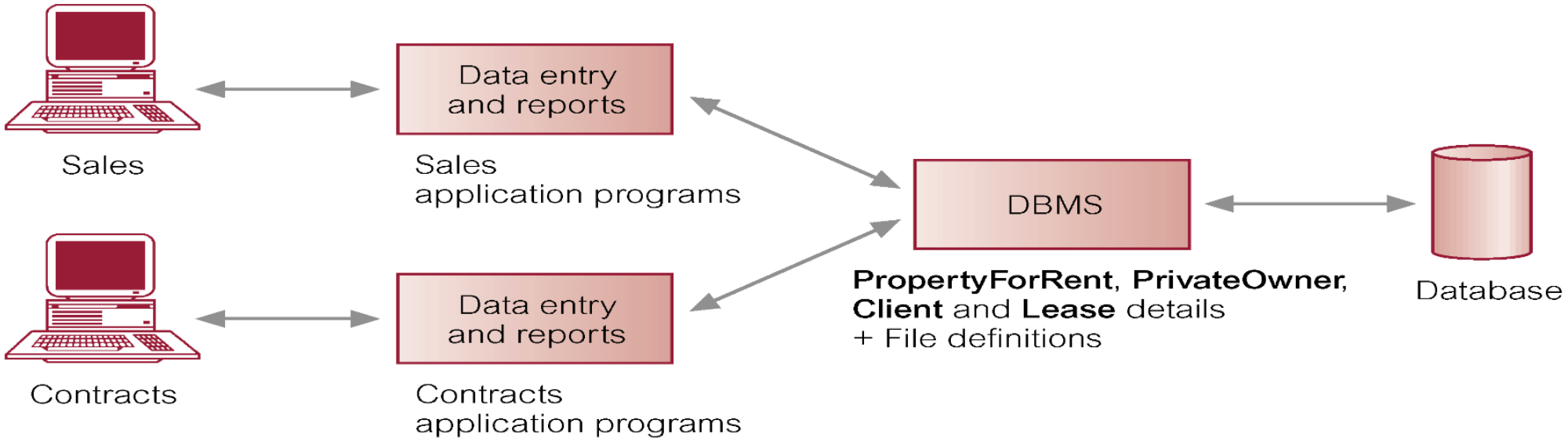
Database Approach

- **Arose because:**
  - Definition of data was embedded in application programs, rather than being stored separately and independently.
  - No control over access and manipulation of data beyond that imposed by application programs.
- **Result**
  - the database and Database Management System (DBMS).



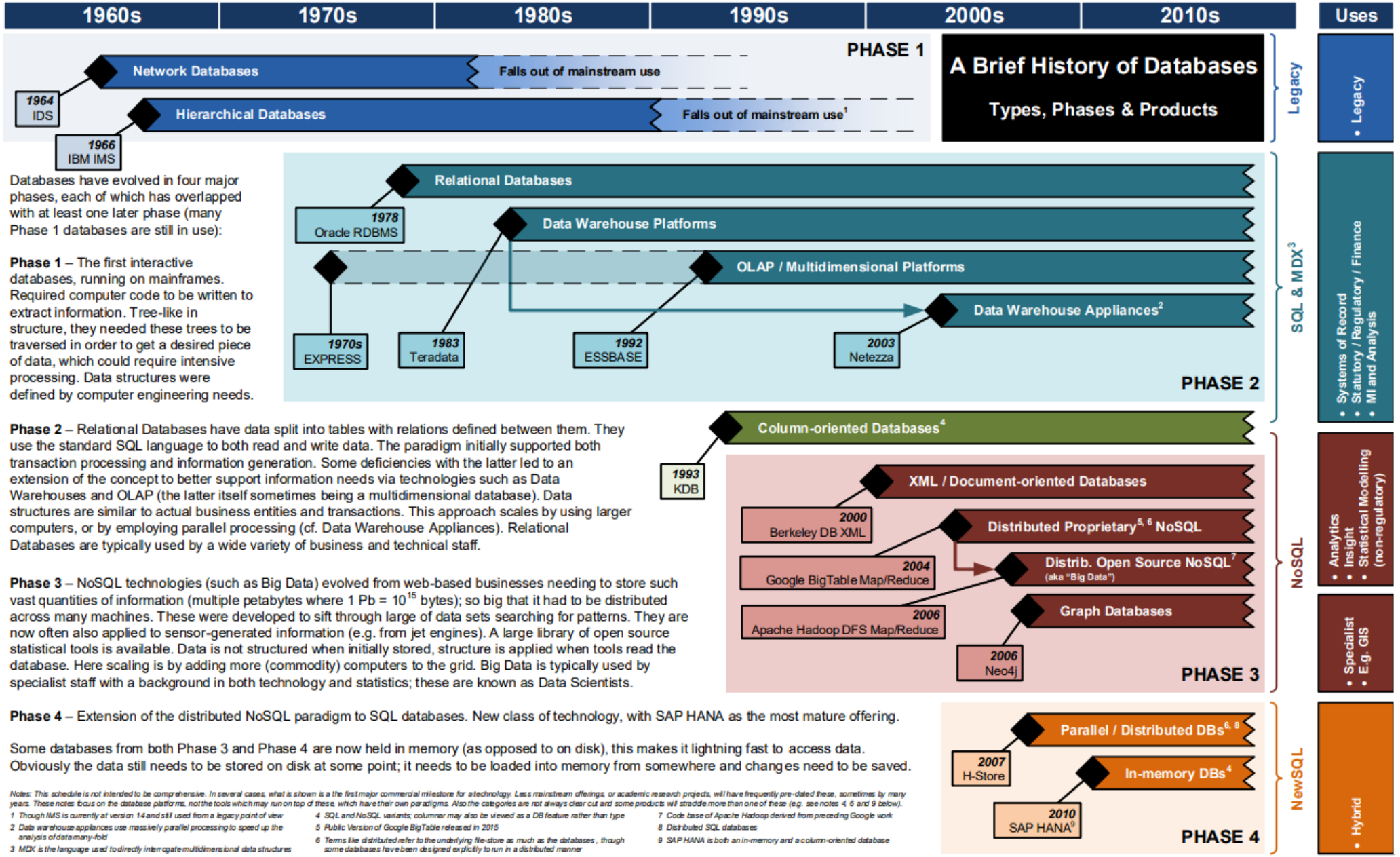
File versus Database Approach to Data Management

(Example #2 - Database Management System (DBMS)



PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)
Lease (leaseNo, propertyNo, clientNo, paymentMethod, deposit, paid, rentStart, rentFinish)

1960s – Era of Databases



Databases have evolved in four major phases, each of which has overlapped with at least one later phase (many Phase 1 databases are still in use):

Phase 1 – The first interactive databases, running on mainframes. Required computer code to be written to extract information. Tree-like in structure, they needed these trees to be traversed in order to get a desired piece of data, which could require intensive processing. Data structures were defined by computer engineering needs.

Phase 2 – Relational Databases have data split into tables with relations defined between them. They use the standard SQL language to both read and write data. The paradigm initially supported both transaction processing and information generation. Some deficiencies with the latter led to an extension of the concept to better support information needs via technologies such as Data Warehouses and OLAP (the latter itself sometimes being a multidimensional database). Data structures are similar to actual business entities and transactions. This approach scales by using larger computers, or by employing parallel processing (cf. Data Warehouse Appliances). Relational Databases are typically used by a wide variety of business and technical staff.

Phase 3 – NoSQL technologies (such as Big Data) evolved from web-based businesses needing to store such vast quantities of information (multiple petabytes where 1 Pb = 10^{15} bytes); so big that it had to be distributed across many machines. These were developed to sift through large of data sets searching for patterns. They are now often also applied to sensor-generated information (e.g. from jet engines). A large library of open source statistical tools is available. Data is not structured when initially stored, structure is applied when tools read the database. Here scaling is by adding more (commodity) computers to the grid. Big Data is typically used by specialist staff with a background in both technology and statistics; these are known as Data Scientists.

Phase 4 – Extension of the distributed NoSQL paradigm to SQL databases. New class of technology, with SAP HANA as the most mature offering.

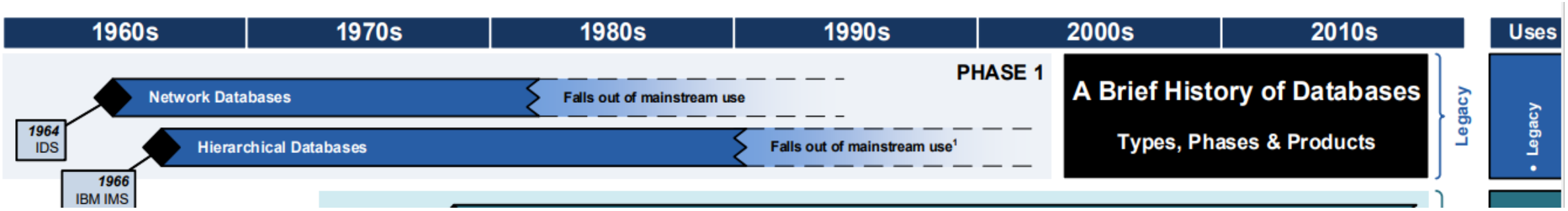
Some databases from both Phase 3 and Phase 4 are now held in memory (as opposed to on disk), this makes it lightning fast to access data. Obviously the data still needs to be stored on disk at some point; it needs to be loaded into memory from somewhere and changes need to be saved.

Notes: This schedule is not intended to be comprehensive. In several cases, what is shown is a the first major commercial milestone for a technology. Less mainstream offerings, or academic research projects, will have frequently pre-dated these, sometimes by many years. These notes focus on the database platforms, not the tools which may run on top of these, which have their own paradigms. Also the categories are not always clear cut and some products will straddle more than one of these (e.g. see notes 4, 6 and 9 below).

1 *Though IMS is currently at version 14 and still used from a legacy point of view*
2 *Data warehouse appliances use massively parallel processing to speed up the analysis of data many-fold*
3 *MDX is the language used to directly interrogate multidimensional data structures*
4 *SQL and NoSQL variants; columnar may also be viewed as a DB feature rather than type*
5 *Public Version of Google BigTable released in 2015*
6 *Terms like distributed refer to the underlying file-store as much as the databases, though some databases have been designed explicitly to run in a distributed manner*
7 *Code base of Apache Hadoop derived from preceding Google work*
8 *Distributed SQL databases*
9 *SAP HANA is both an in-memory and a column-oriented database*

Acknowledgements:
a. Neil Raden (@neilraden) from Hired Brains (hiredbrains.com) – For both reviewing earlier drafts and providing significant input

Evolution of Databases – Phase 1

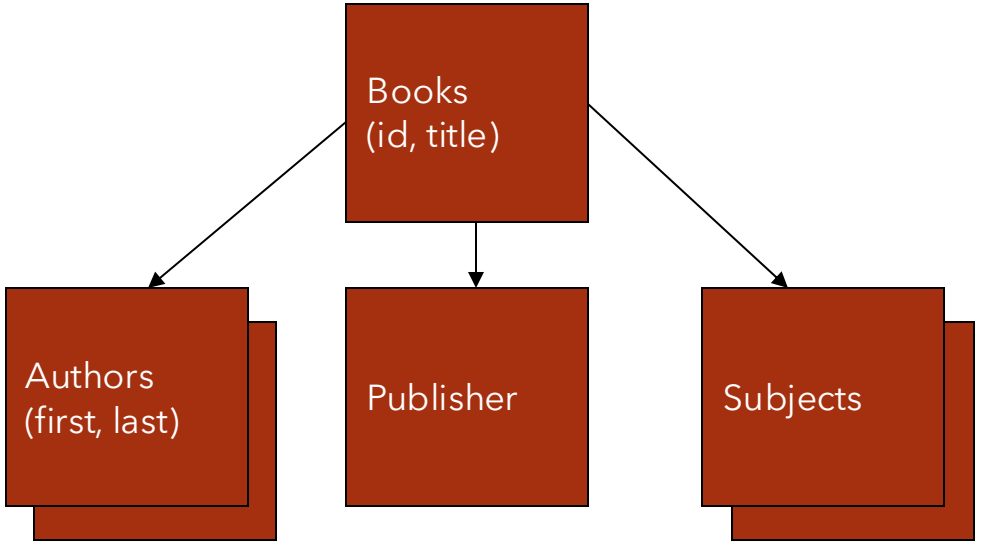


Databases have evolved in four major phases, each of which has overlapped with at least one later phase (many Phase 1 databases are still in use):

Phase 1 – The first interactive databases, running on mainframes. Required computer code to be written to extract information. Tree-like in structure, they needed these trees to be traversed in order to get a desired piece of data, which could require intensive processing. Data structures were defined by computer engineering needs.

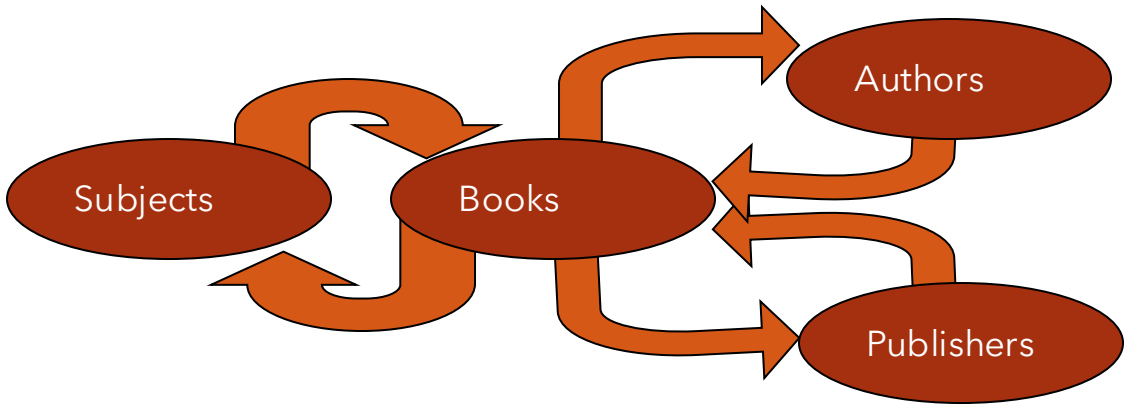
Hierarchical Model

- 1960s – IBM IMS
- Tree-like structure
- **Problems:**
  - Duplicate data – repetitive storage in different entities
  - Sequential searching – top to bottom search of the model – slow query processing

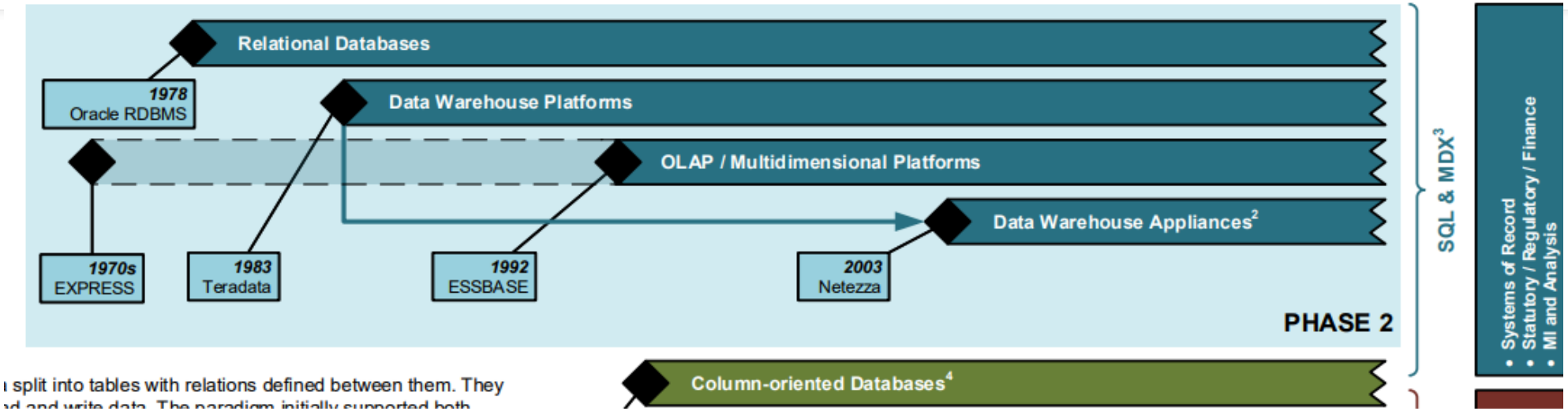


Network Model

- 1970s – CODASYL
- Charles Bachman
- Graph like structure
- **Problems:**
  - Complicated
  - All records are maintained by pointers
  - Difficult to maintain
  - Complex queries



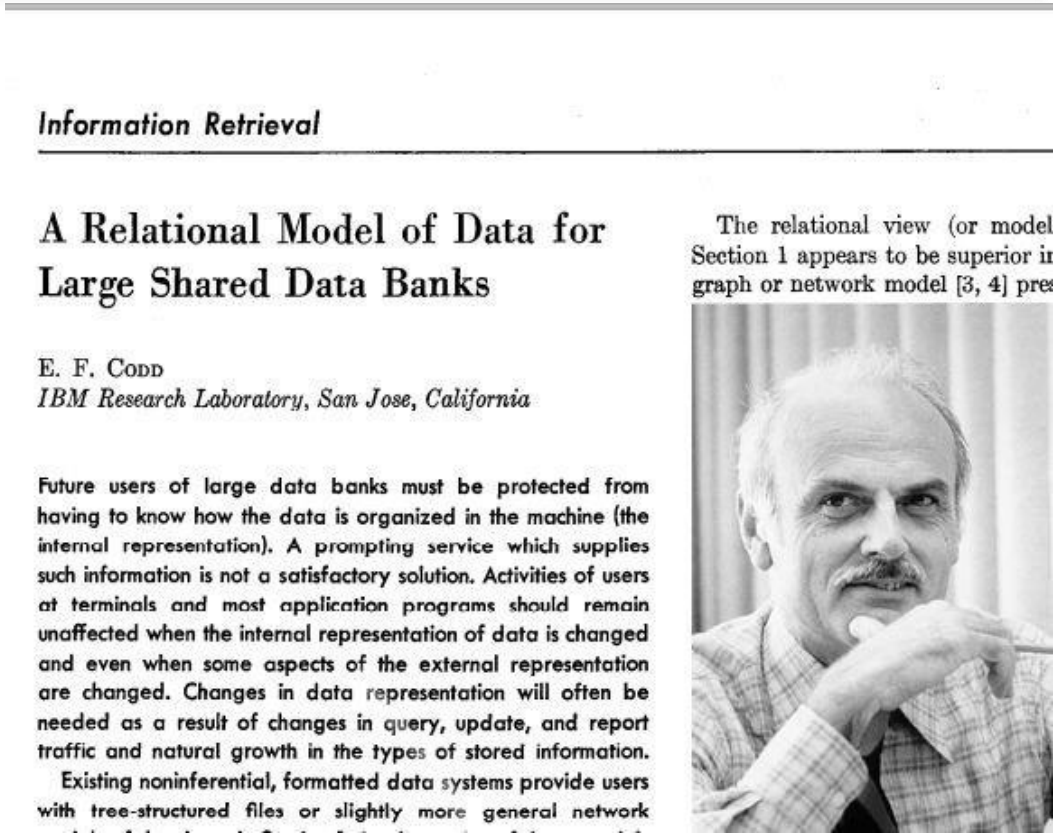
Evolution of Databases – Phase 2



Phase 2 – Relational Databases have data split into tables with relations defined between them. They use the standard SQL language to both read and write data. The paradigm initially supported both transaction processing and information generation. Some deficiencies with the latter led to an extension of the concept to better support information needs via technologies such as Data Warehouses and OLAP (the latter itself sometimes being a multidimensional database). Data structures are similar to actual business entities and transactions. This approach scales by using larger computers, or by employing parallel processing (cf. Data Warehouse Appliances). Relational Databases are typically used by a wide variety of business and technical staff.

Relational Database

- Edgar Codd at IBM
- Witnessed rewriting programs for IMS and CODASYL when database schema changed.
- Proposed the relational data model in 1970. Data arranged in relations or tables
- No pointers to maintain as tables are connected by *matching* fields.
- Easy to access, merge and modify data.



Information Retrieval

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 are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

Existing noninferential, formatted data systems provide users with tree-structured files or slightly more general network

The relational view (or model
Section 1 appears to be superior ir
graph or network model [3, 4] pre

Relational Data Model

- CJ Date – IBM instructor and author also convinced with the relational model.
- Along with Codd, lectured and published papers.
- IBM IMS (hierarchical) was a more profitable product hence, IBM support remained slow.

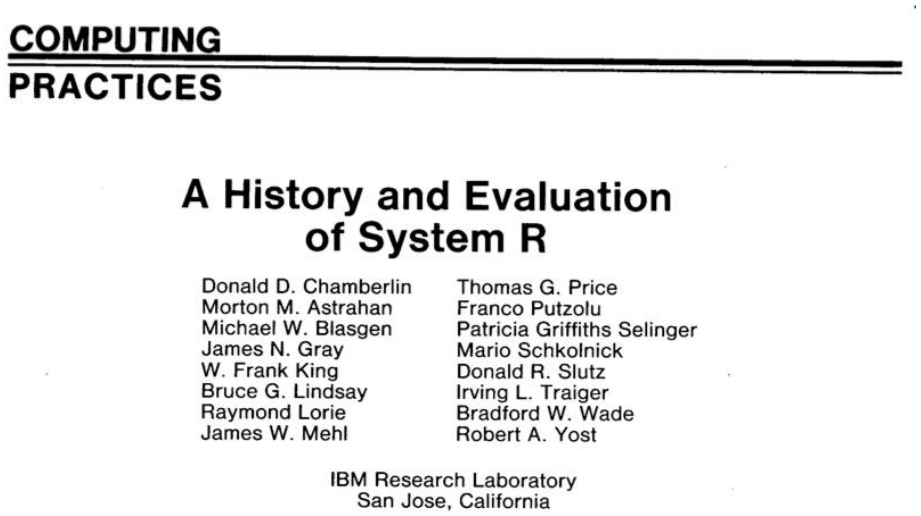
Relational DBMS

Ingres

- 1973 – Micheal Stonebraker – UC Berkely team
- Ingres relational database
- **In**teractive **G**raphics and **Re**trieval **S**ystem (InGReS)

System R

- 1975 – IBM produced an experimental relational database
- Used a structured query language (SQL) developed by *Don Chamberlin and Raymond Boyce of IBM* - to search and modify data.



COMPUTING PRACTICES

A History and Evaluation of System R

Donald D. Chamberlin, Morton M. Astrahan, Michael W. Blasgen, James N. Gray, W. Frank King, Bruce G. Lindsay, Raymond Lorie, James W. Mehl, Thomas G. Price, Franco Putzolu, Patricia Griffiths Selinger, Mario Schkolnick, Donald R. Slutz, Irving L. Traiger, Bradford W. Wade, Robert A. Yost

IBM Research Laboratory
San Jose, California

ORACLE RDBMS - 1979

- 1977 - Larry Ellison + Bob Miner + Ed Oates
  - Software Development Laboratories (SDL)
  - Mission to develop and sell first commercially available relational database compatible with IBM System R
- 1979 – Oracle shipped
  - Company renamed to Relational Systems Inc (RSI)
  - First version on mini-computers.
- 1983 improved and reprogrammed to run on more systems
  - Company now named Oracle Systems Corporation

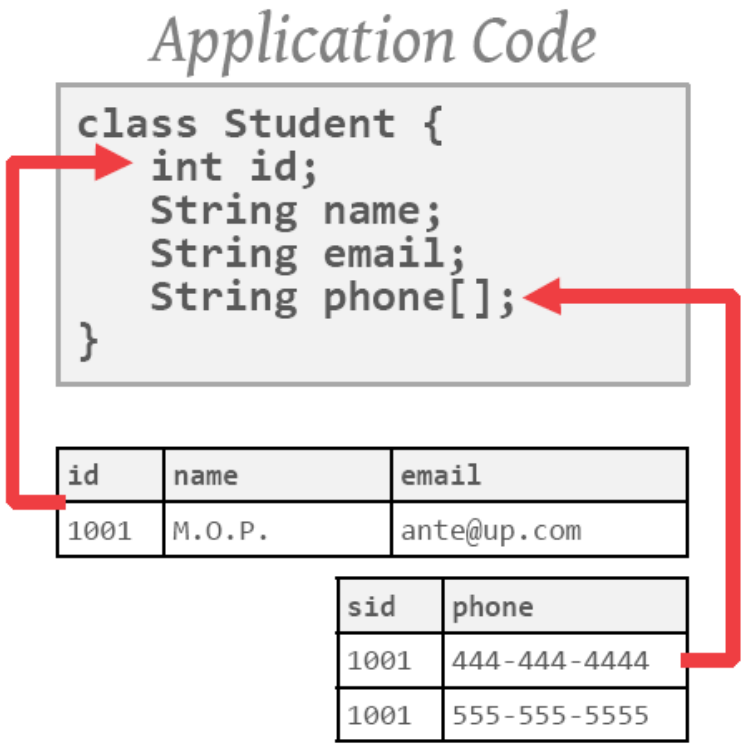
Other Relational DBMS

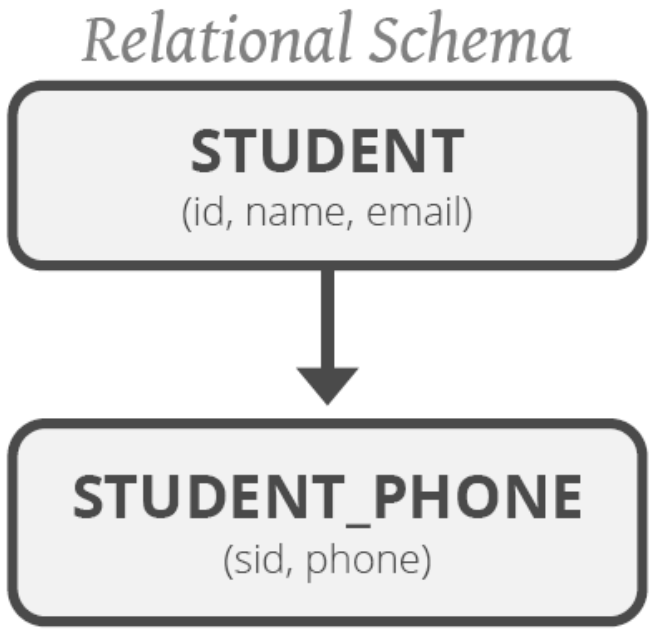
- IBM DB2 (1983) - Commercial relational database for mainframes
- Other new enterprise DBMS invented (Informix, Sybase, TeraData)
- Too late to dominate – Oracle already popular

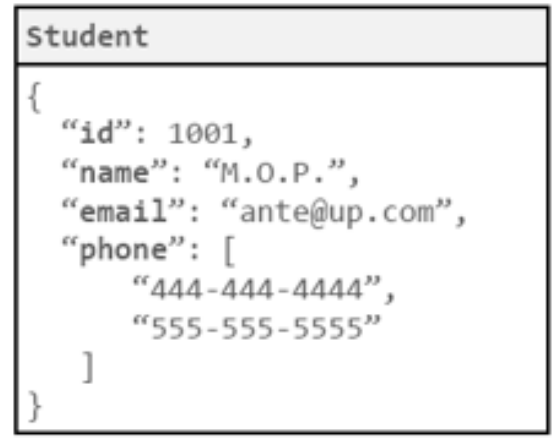
PostgreSQL

- 1986 by Micheal Stonebreaker
- Initially named POSTGRES in reference to older INGRES database developed at Berkely.
- New features for multiple data types

Object – Oriented Model







Object-Oriented Model

- Avoid "*relational-object impedance mismatch*" by tightly coupling objects and database.
- Few of these original DBMSs from the 1980s still exist today but many of the technologies exist in other forms (JSON, XML)
- **Problems:**
  - Complex queries
  - No Standard API or programming language

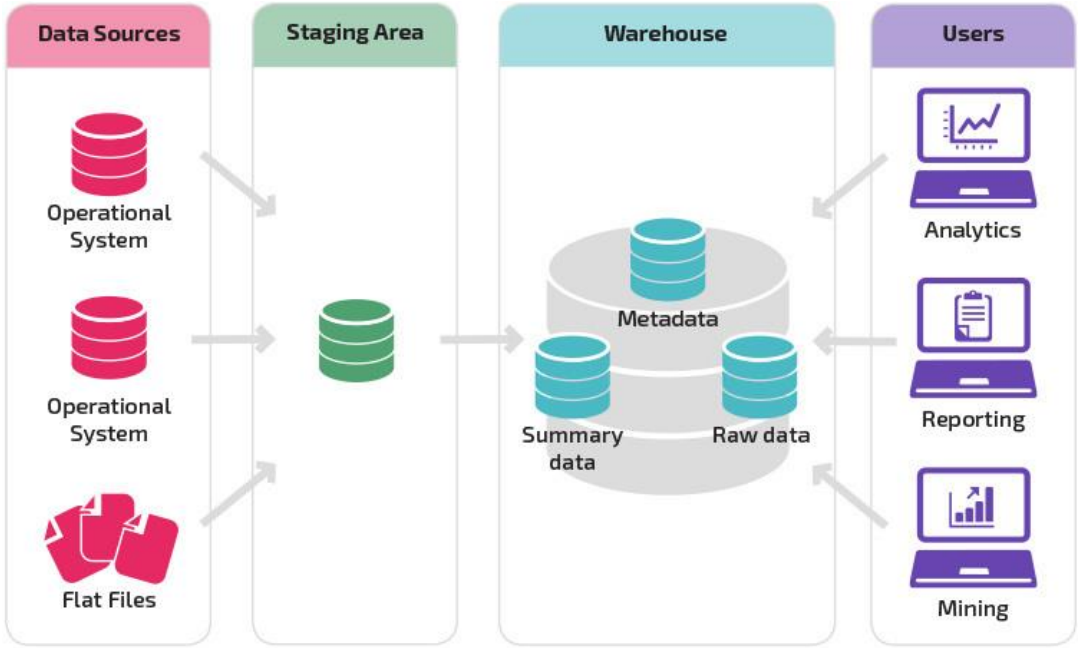
VERSANT ObjectStore. MarkLogic

1990s

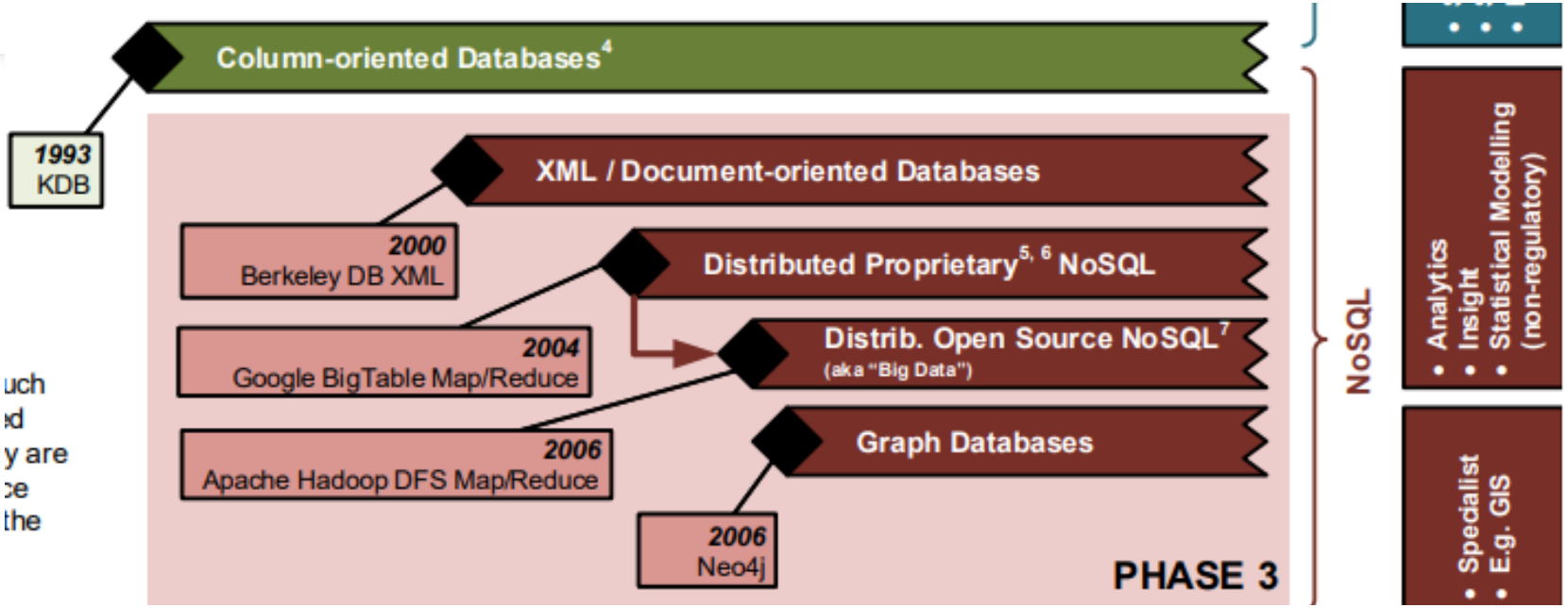
- Few advancements
- Microsoft SYBASE → SQL Server
- MySQL was written as a replacement for mSQL
- Illustra (commercial version of Postgres) bought by Informix. Berkeley graduate students take original academic Postgres code and adds support for SQL
- SQLite in early 2000s.

2000s - Data Warehouses

- Developed by businesses to consolidate the data from a variety of databases to help support strategic decision-making.



Evolution of Databases – Phase 3



Phase 3 – NoSQL technologies (such as Big Data) evolved from web-based businesses needing to store such vast quantities of information (multiple petabytes where 1 Pb = 10^{15} bytes); so big that it had to be distributed across many machines. These were developed to sift through large of data sets searching for patterns. They are now often also applied to sensor-generated information (e.g. from jet engines). A large library of open source statistical tools is available. Data is not structured when initially stored, structure is applied when tools read the database. Here scaling is by adding more (commodity) computers to the grid. Big Data is typically used by specialist staff with a background in both technology and statistics; these are known as Data Scientists.

XML

- 2000 – for managing semi-structured data
- XML documents and elements as compared to tables, records and fields in relational.
- Perform better in heavy document processing such as newspaper publishing, website services and management, etc.

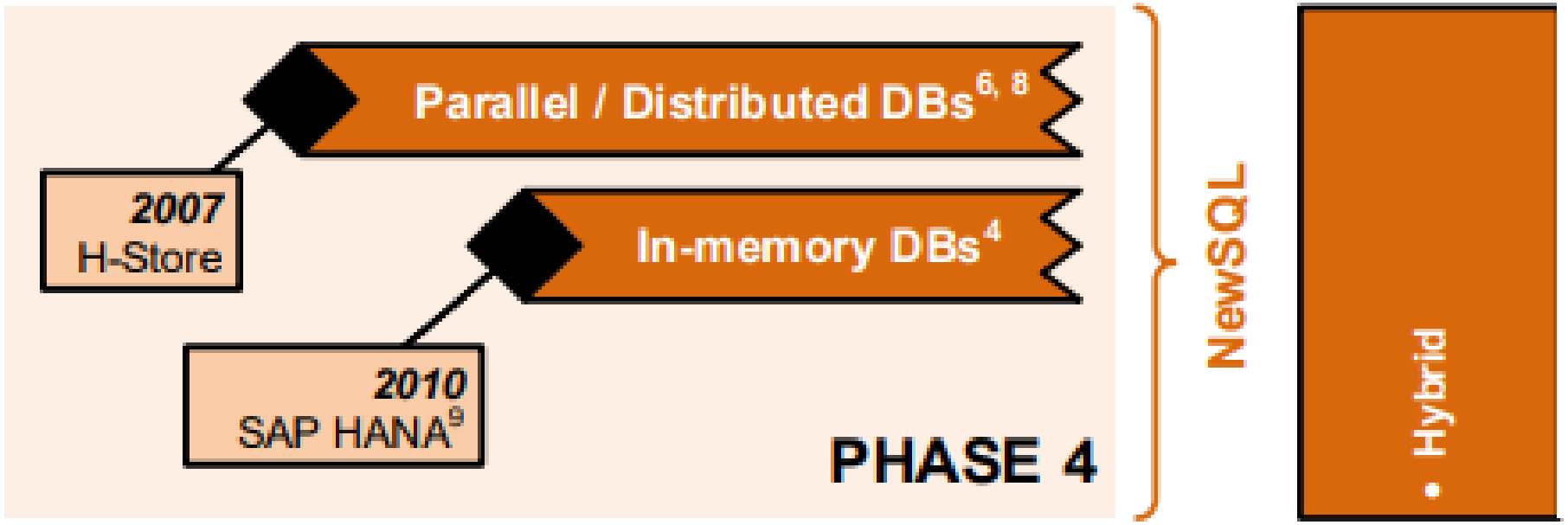
```
<!DOCTYPE html>          ← Tells version of HTML
<html>                   ← HTML Root Element

<head>                   ← Used to contain page HTML metadata
  <title>Page Title</title>   ← Title of HTML page
</head>

<body>                   ← Hold content of HTML
  <h2>Heading Content</h2>    ← HTML headling tag
  <p>Paragraph Content</p>    ← HTML paragraph tag
</body>

</html>
```

```
<?xml version="1.0" ?>
<!-- Bookstore with no DTD -->
- <Bookstore>
  - <Book ISBN="ISBN-0-13-713526-2" Price="85" Edition="3rd">
      <Title>A First Course in Database Systems</Title>
    - <Authors>
      - <Author>
          <First_Name>Jeffrey</First_Name>
          <Last_Name>Ullman</Last_Name>
        </Author>
      - <Author>
          <First_Name>Jennifer</First_Name>
          <Last_Name>Widom</Last_Name>
        </Author>
      </Authors>
    </Book>
  - <Book ISBN="ISBN-0-13-815504-6" Price="100">
      <Remark>Buy this book bundled with "A First Course" -- a great deal!</Remark>
      <Title>Database Systems: The Complete Book</Title>
    - <Authors>
```

NoSQL

- 2009 – Eric Evans and Johan Oskarsson described the non-relational databases as NoSQL.
  - Multiple types (document, key-value, etc)
  - Process unstructured data
  - Distributed database system
  - Quick and flexible
  - Schema – less
  - Custom API instead of SQL
  - No ACID transactions

Evolution of Databases – Phase 4



Phase 4 – Extension of the distributed NoSQL paradigm to SQL databases. New class of technology, with SAP HANA as the most mature offering.

Some databases from both Phase 3 and Phase 4 are now held in memory (as opposed to on disk), this makes it lightning fast to access data. Obviously the data still needs to be stored on disk at some point; it needs to be loaded into memory from somewhere and changes need to be saved.

NewSQL

- Provide same performance for OLTP workloads as NoSQL DBMSs without giving up ACID properties
  - Relational
  - Distributed System
  - Usually closed- source

Hybrid Systems

- **H**ybrid **T**ransactional-**A**nalytical **P**rocessing.
- Execute fast OLTP like a NewSQL system while also executing complex OLAP queries like a data warehouse system.
  - Distributed / Shared-Nothing
  - Relational / SQL
  - All closed-source (as of 2016).

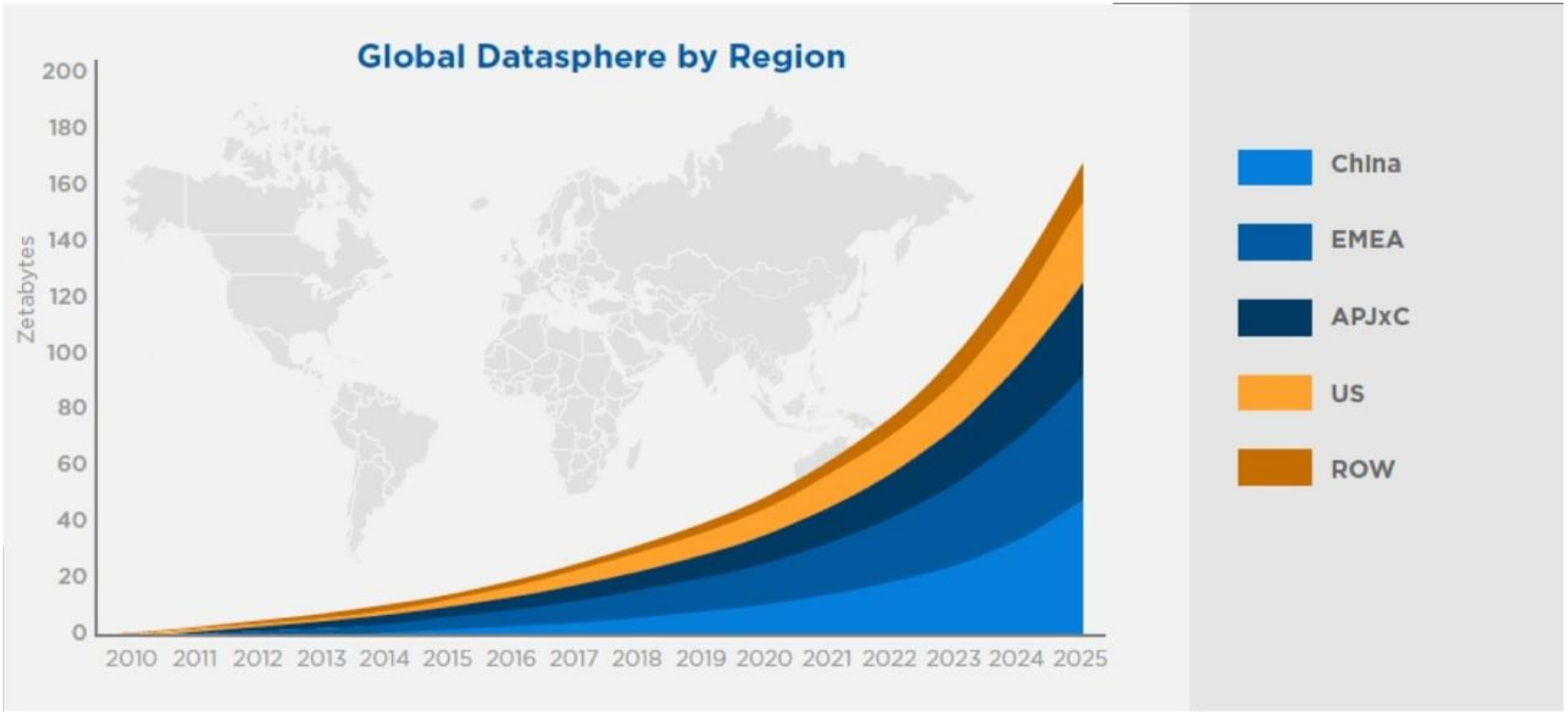
Cloud Systems

- First database-as-a-service (DBaaS) offerings were "containerized" versions of existing DBMSs.
- Some new DBMSs that are designed from scratch explicitly for running in a cloud environment.

Newer Advancements

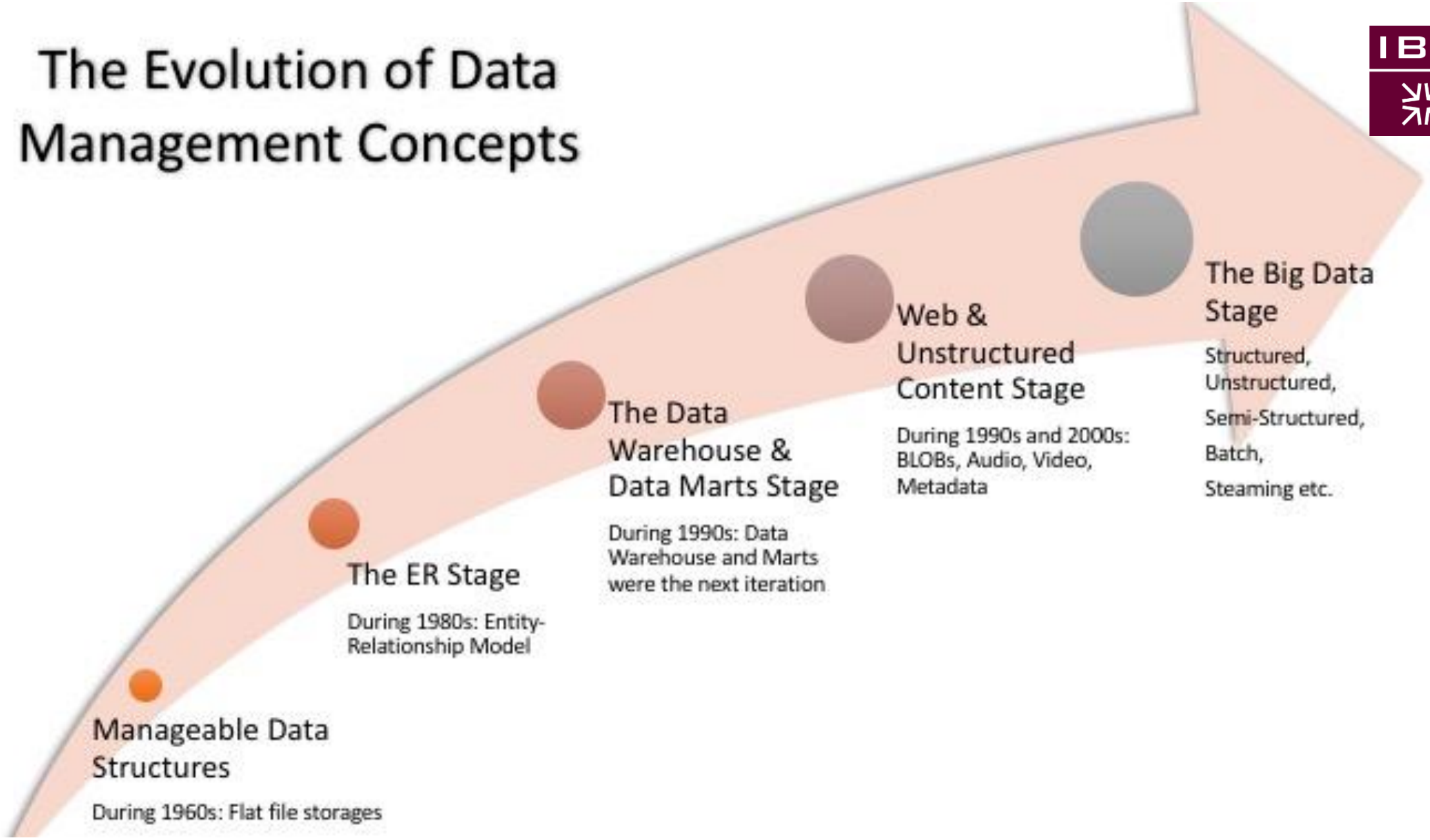
- Shared Disk Engines
- Graph Databases
- Time series systems
- Blockchain databases
- Vector databases

Datasphere Today



Source: IDC's Data Age 2025 study, sponsored by Seagate

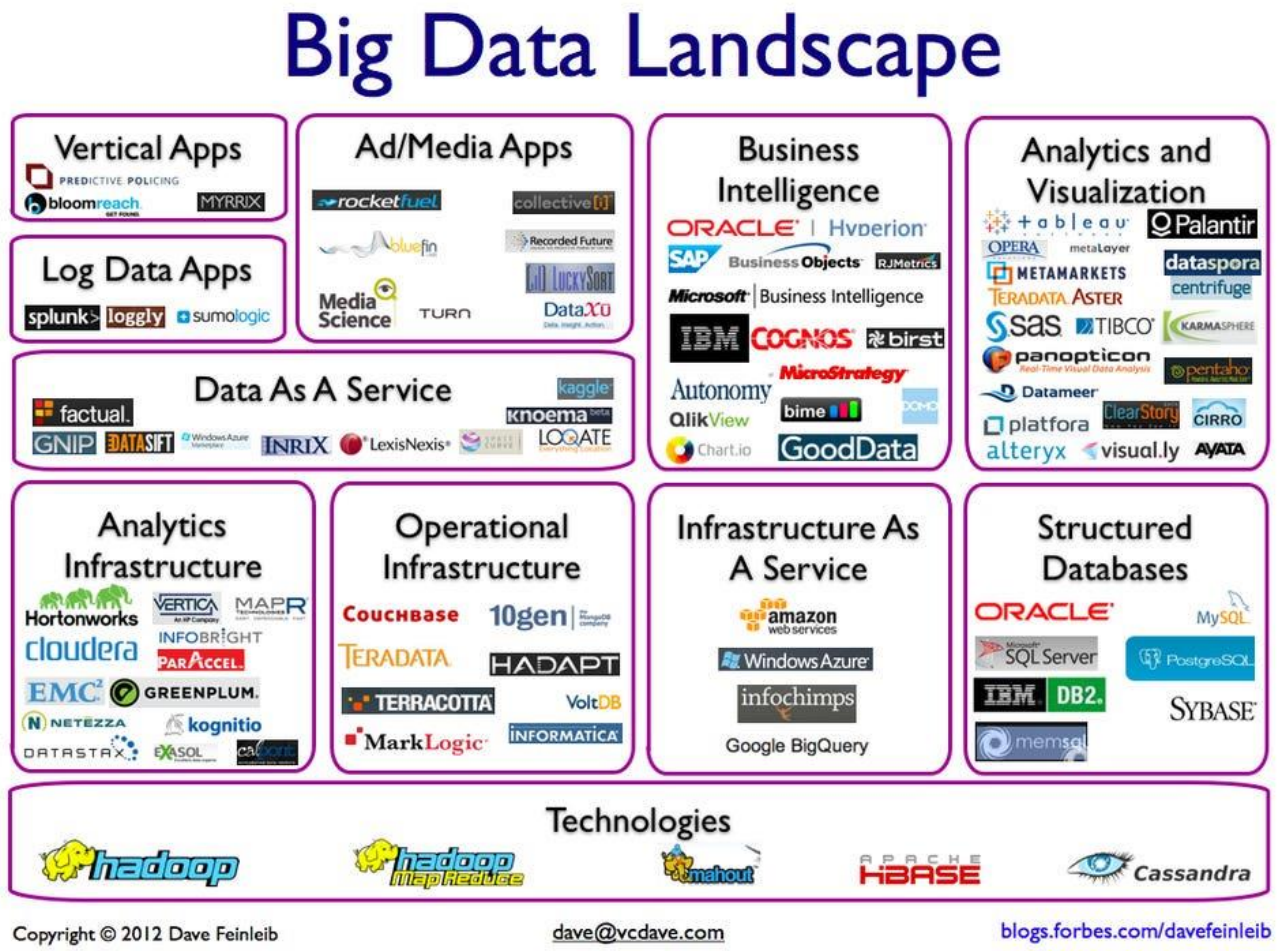
The Evolution of Data Management Concepts



Era of Big Data

3 Vs – Volume, Velocity, Variety

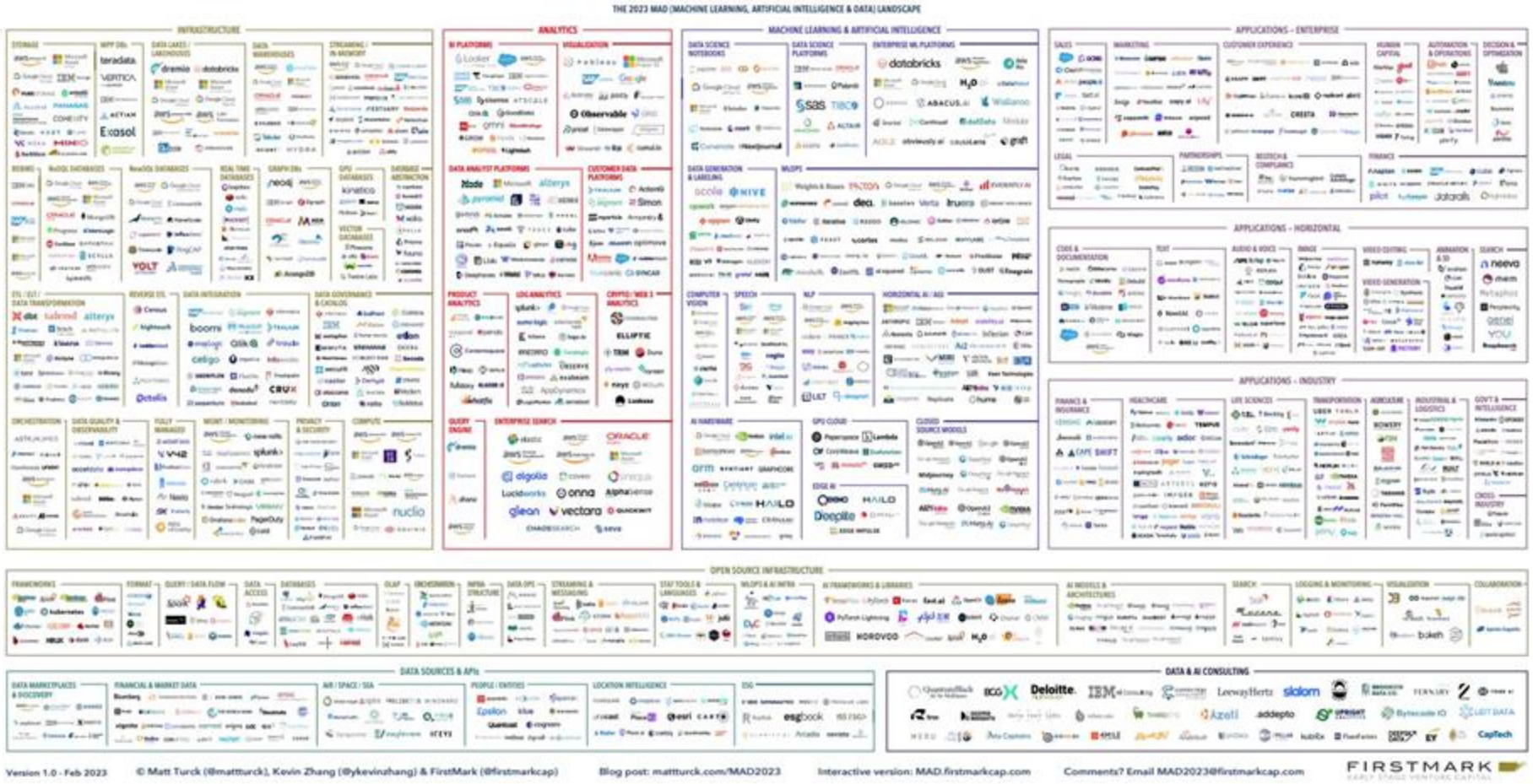
Big Data Landscape in 2012



MAD Landscape 2023

MACHINE LEARNING, ARTIFICIAL INTELLIGENCE & DATA

https://mattturck.com/landscape/mad2023.pdf



MAD Landscape 2024

FirstMark | 2024 MAD (ML/AI/Data) Landscape