A brief history of data and databases

Spanning thousands and thousands of years

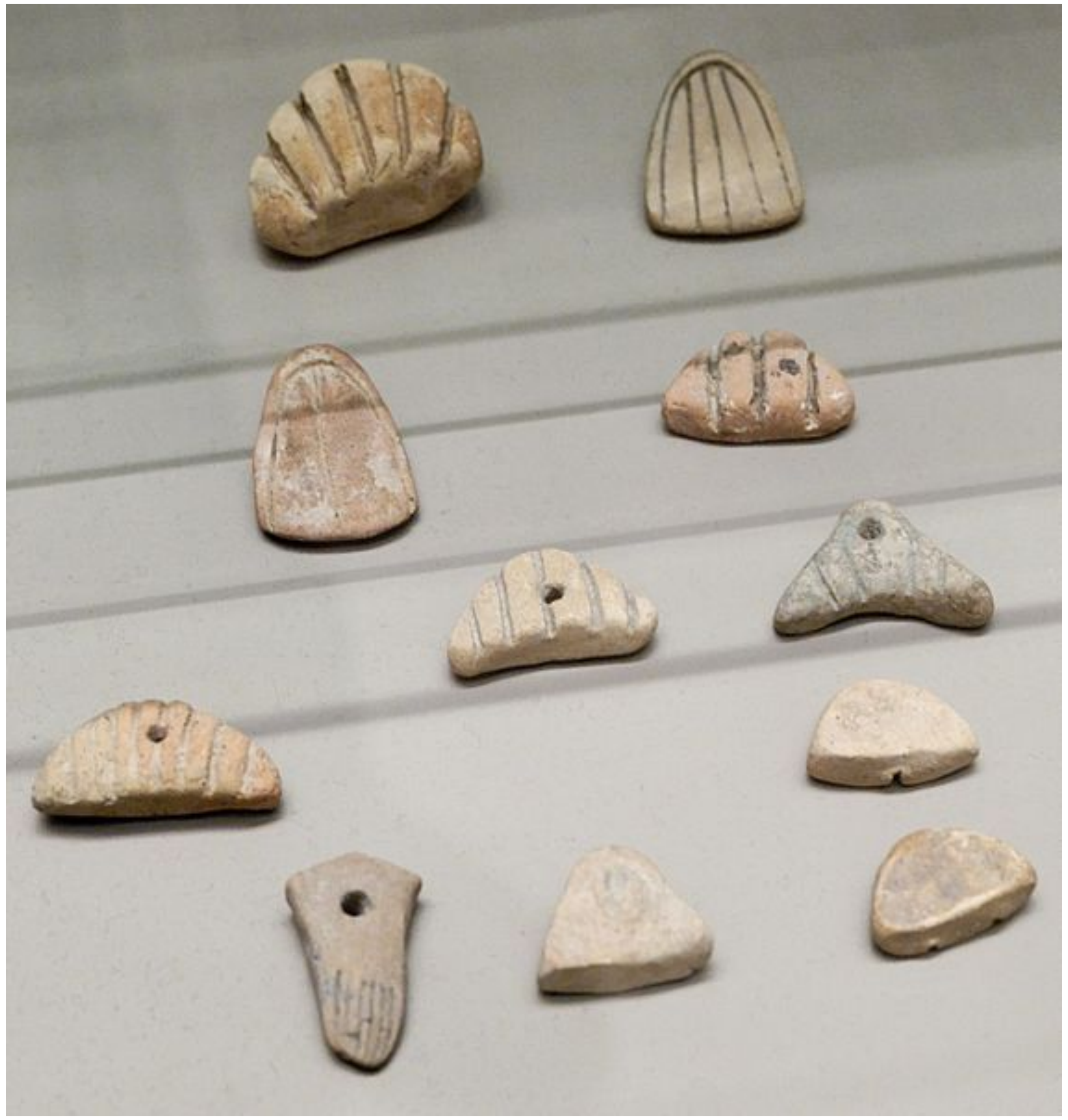
Me:

- Basic (Texas Instruments style)
- Free Trucks and Bass Boats
- Line 21
- DBA as a career path?
- Inmate Video
- Point of Sale
- General Management
- Lost In Translation
- Altas School

Record Keeping – How long?

Images: Public Domain, Library of Congress, African and Middle Eastern Division, Cuneiform Tablet Collection. Minassian, K. & Kirkor Minassian Collection. (2200) Balanced account, Cuneiform tablet no. 32. [between 2200 and 1900 B.C] Retrieved from the Library of Congress, https://www.loc.gov/item/2020741401/

Record Keeping – How long?


Clay Tokens, Uruk Period, Excavated from Susa, Iran. Louvre Museum (Department of Near Eastern Antiquities). Marie-Lan Nguyen

4,000-3,100 BCE

Pictures of goods and cuneiform symbols.

Why?

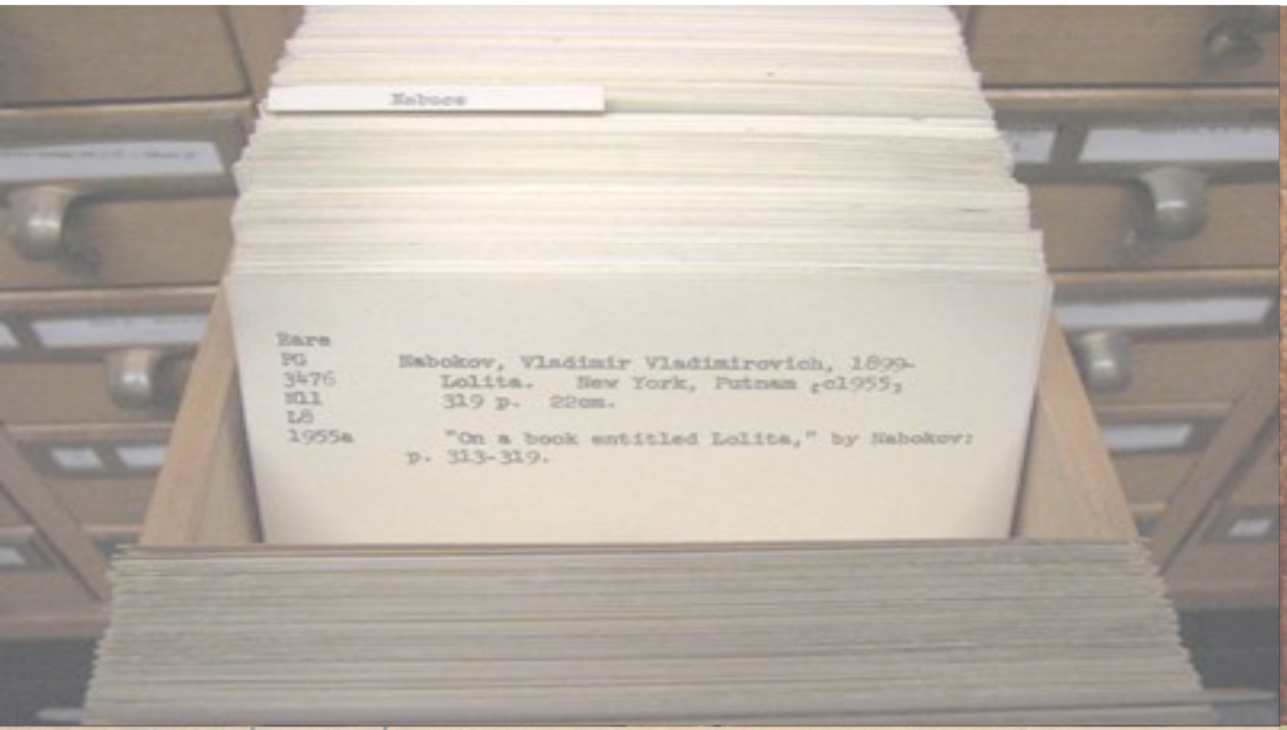
- Most of the records are not digital
- We use records to measure “stuff”

Jacquard Loom

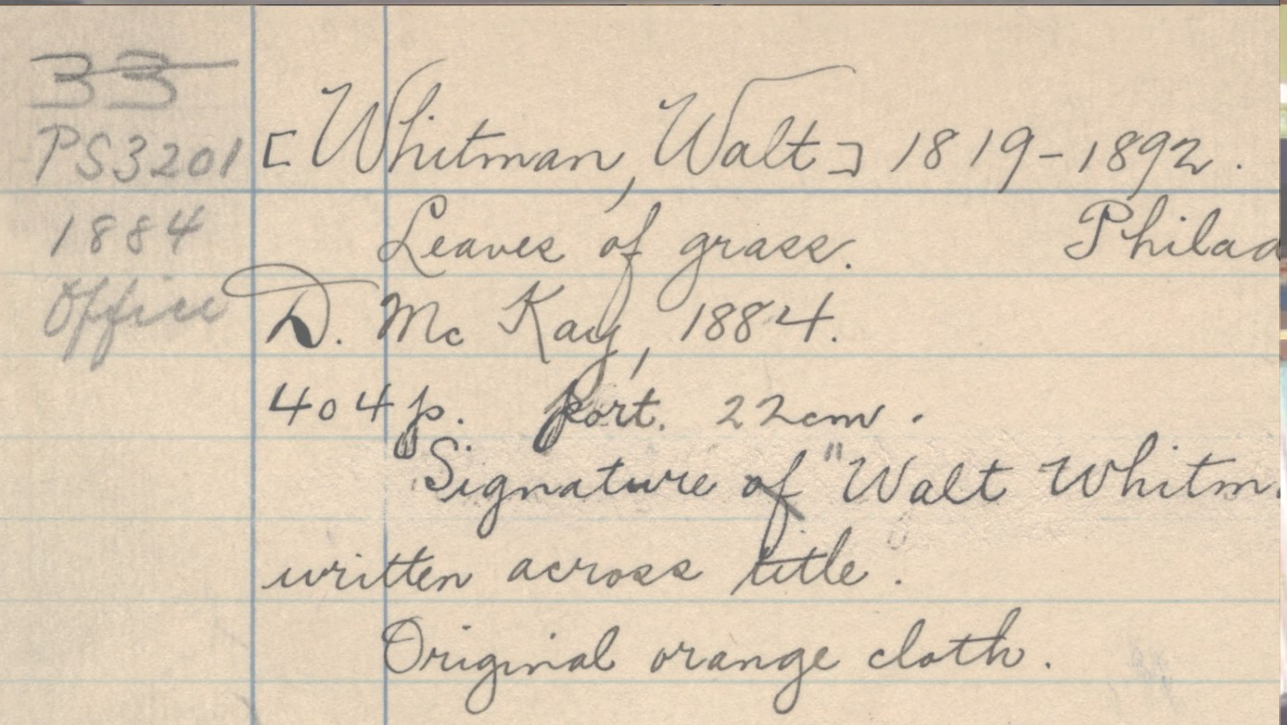
Other non-electronic records

- Motor vehicle licenses and registrations
- SS cards – 35 million hand typed between 1937-1938
- Marriage and Death records
- Financial records for companies
- Older School records

Card Catalogs – An ingeneous indexing system



```
Rare
PG        Nabokov, Vladimir Vladimirovich, 1899-
3476          Lolita.   New York, Putnam ,c1955,
N11           319 p.  22cm.
L8
1955a         "On a book entitled Lolita," by Nabokov:
          p. 313-319.
```



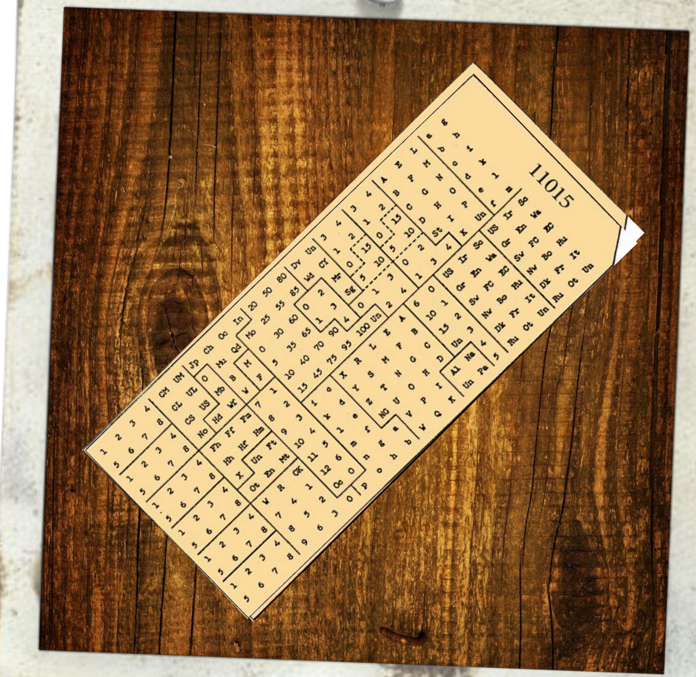
~~33~~
PS3201 [Whitman, Walt] 1819-1892.
1884 Leaves of grass. Philad
Office D. McKay, 1884.
404 p. port. 22cm.
Signature of "Walt Whitm
written across title.
Original orange cloth.

Problem – The 1890 census

Herman Hollerith

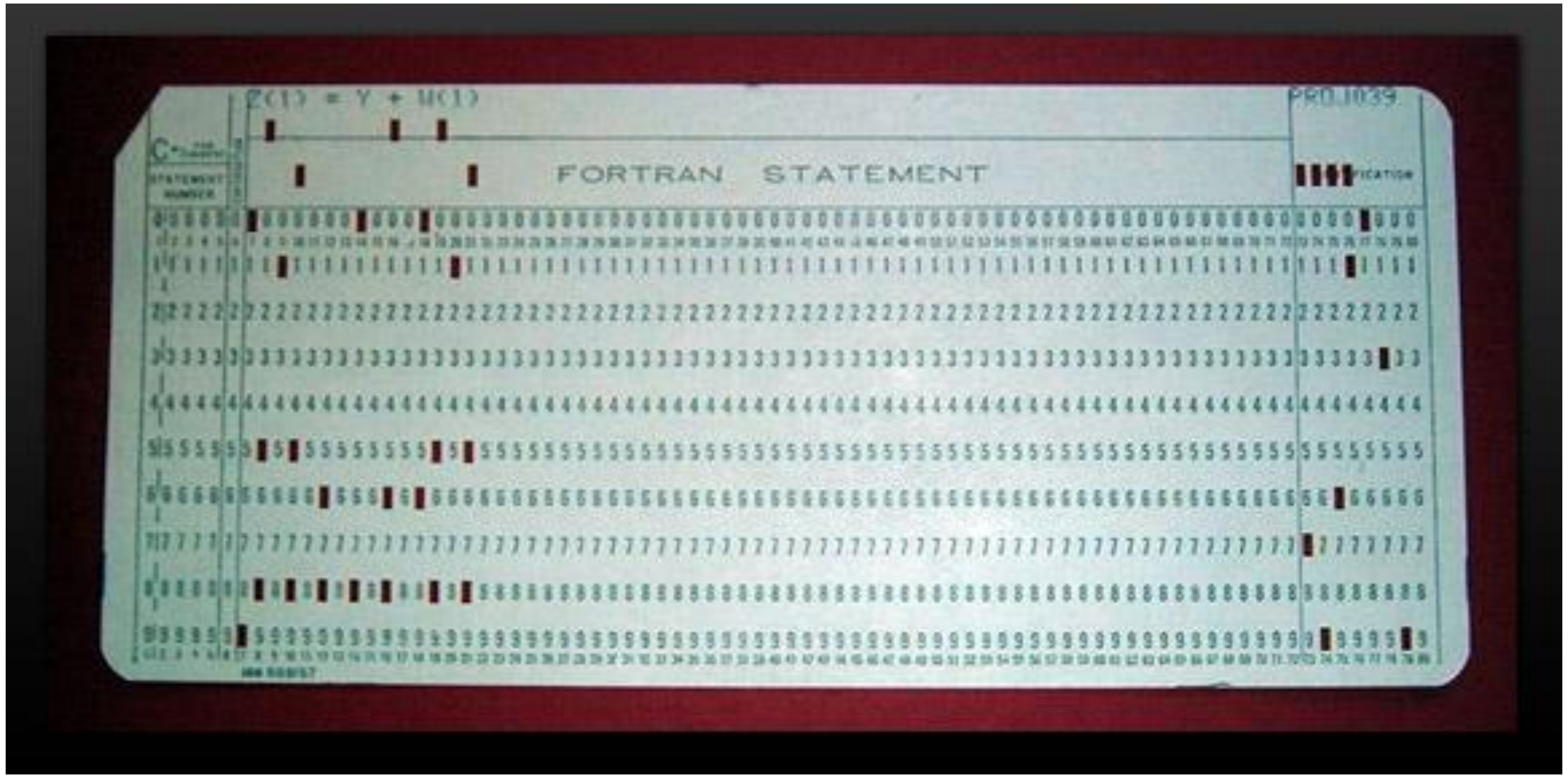
Hollerith's devices

Hollerith Pantograph


Hollerith Card

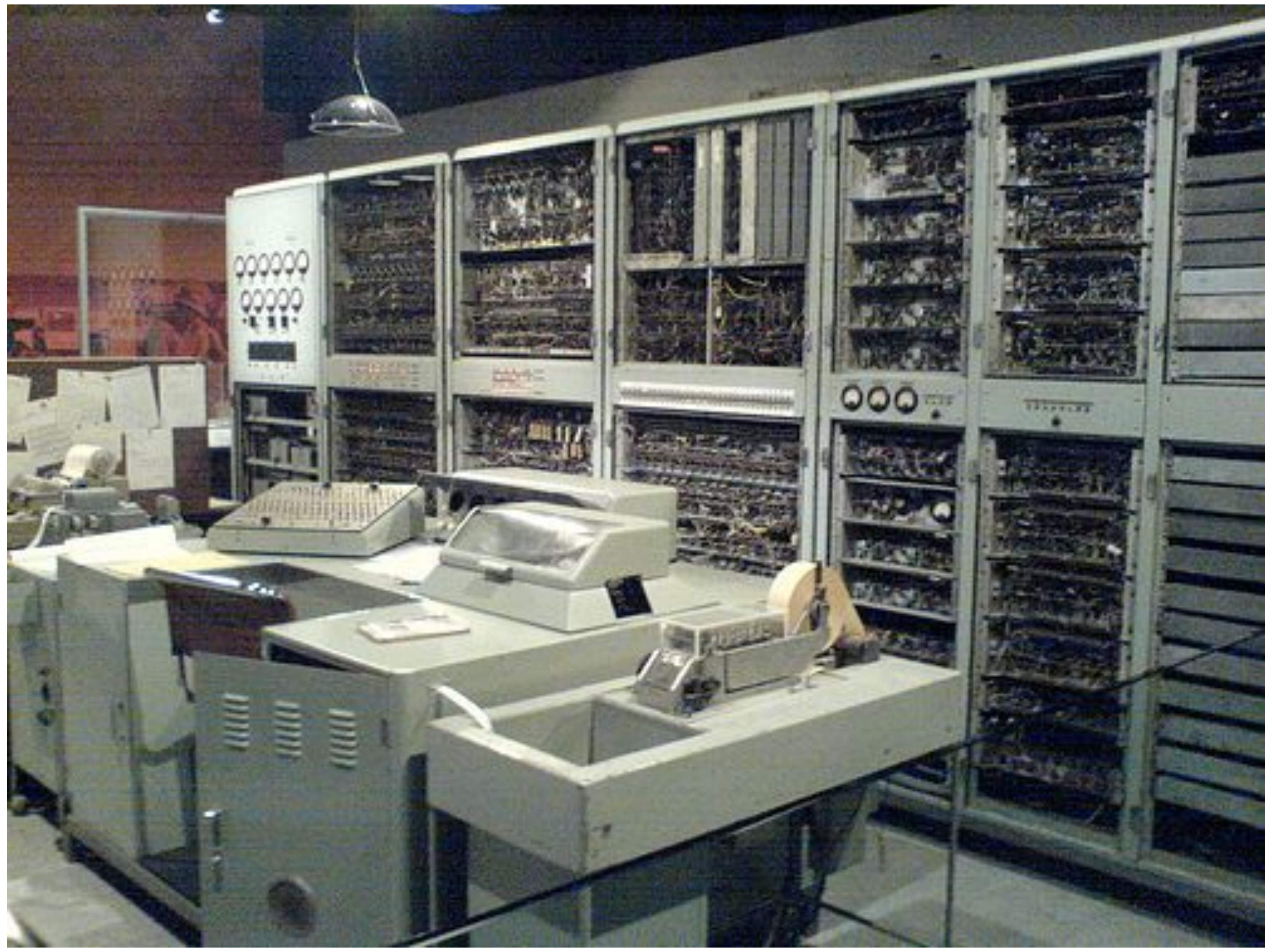
Integrating Machine

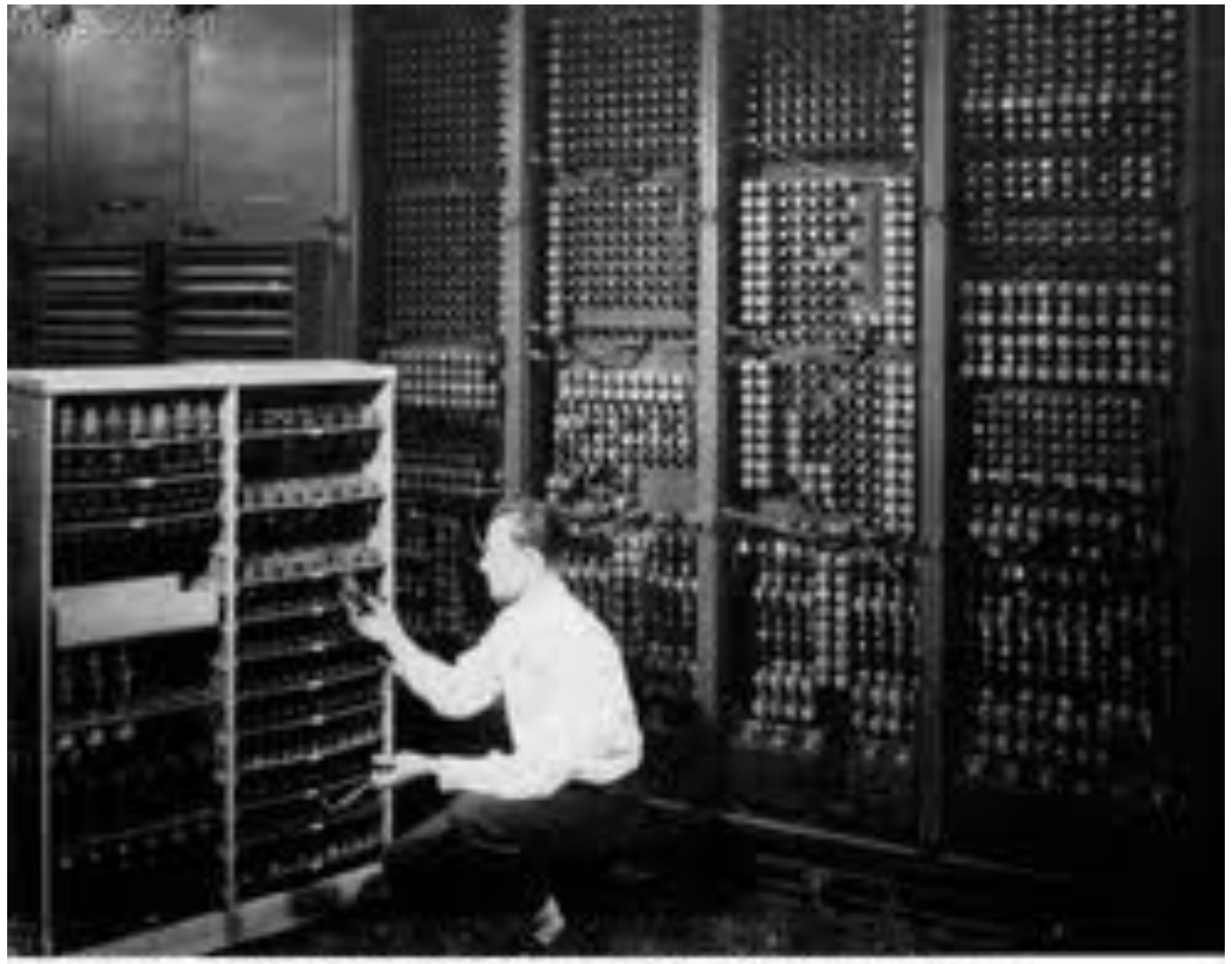
First computers



The program, the data – all done with punchcards

Electronic files – Early computing 1950s – 1970s




Replacing a bad tube meant checking among ENIAC's 19,000 possibilities.

+
Enter the database – Early 1960s

- Objects in a database can be related to one another.
- Hierarchical – One record leads to the related record. (like a tree!)
- Network – Allowed for multiple relationships (like a network!)
- The databases used pointers to relate one record to another.

Charles Bachman

Integrated Data Store – Dow Chemical

+
Some issues

- An improvement over file based systems
- These systems required knowledge of the structures to use them. There was no built in search mechanism.
- Very few users understood the structures, access was limited to an elite few.
- Queries were complex. It took time to get new information and expensive programmer time to produce.

+
Enter the relational DBMS ~1970: Edgar Codd

- Relational DBMS Work
  - Mathematician at IBM
  - Based on Relational Calculus and set theory
- IBM
  - System R
- Led to
  - Oracle
  - IBM DB2
  - Informix
  - Sybase
  - MS SQL Server (based on Sybase)

+ Relational Ideas

- Data is represented as a series of tables.
- The tables are Related to one another through a series of keys and foreign keys.
- A standard language is used to define the database (DDL) and to query the database (DML).
- Tables within the database contain the data about the database (meta data).

+ Why relational?

- It is easy for most people to “see” and “get it”.
- Makes the data accessible for a wider number of users through user friendly query tools.
- Through good database design, space usage is efficient (although this has become less of an issue of late)

Codd’s 12* Rules

Rule 0: The foundation rule:
For any system that is advertised as, or claimed to be, a relational database management system, that system must be able to manage databases entirely through its relational capabilities.

Rule 1: The information rule:
All information in a relational database is represented explicitly at the logical level and in exactly one way — by values in tables.

Rule 2: The guaranteed access rule:
Each and every datum (atomic value) in a relational database is guaranteed to be logically accessible by resorting to a combination of table name, primary key value and column name.

Rule 3: Systematic treatment of null values:
Null values (distinct from the empty character string or a string of blank characters and distinct from zero or any other number) are supported in fully relational DBMS for representing missing information and inapplicable information in a systematic way, independent of data type.

Codd’s 12* Rules

Rule 4: Dynamic online catalog based on the relational model:
The database description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data.
Rule 5: The comprehensive data sublanguage rule:
A relational system may support several languages and various modes of terminal use (for example, the fill-in-the-blanks mode). However, there must be at least one language whose statements are expressible, per some well-defined syntax, as character strings and that is comprehensive in supporting all of the following items:

- Data definition.
- View definition.
- Data manipulation (interactive and by program).
- Integrity constraints.
- Authorization.
- Transaction boundaries (begin, commit and rollback).

+
Codd’s 12* Rules

Rule 6: The view updating rule:

All views that are theoretically updatable are also updatable by the system.

Rule 7: High-level insert, update, and delete:

The capability of handling a base relation or a derived relation as a single operand applies not only to the retrieval of data but also to the insertion, update and deletion of data.

Rule 8: Physical data independence:

Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representations or access methods.

Rule 9: Logical data independence:

Application programs and terminal activities remain logically unimpaired when information-preserving changes of any kind that theoretically permit unimpairment are made to the base tables.

Rule 10: Integrity independence:

Integrity constraints specific to a particular relational database must be definable in the relational data sublanguage and storable in the catalog, not in the application programs.

Codd's 12* Rules

Rule 11: Distribution independence:
The end-user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only.

Rule 12: The non-subversion rule:
If a relational system has a low-level (single-record-at-a-time) language, that low level cannot be used to subvert or bypass the integrity rules and constraints expressed in the higher level relational language (multiple-records-at-a-time).

Structured Query Language

Designers of the SQL database language

Uses English words like

- SELECT
- FROM
- WHERE
- IN
- GROUP
- ORDER

ACID

- **Atomicity -** commits finish an entire operation successfully or the database rolls back to its prior state
- **Consistency -** any change maintains data integrity or is cancelled completely
- **Isolation -** any read or write will not be impacted by other reads or writes of separate transactions
- **Durability -** successful commits will survive permanently

NoSQL

Typically key-value storage

JSON
DynamoDB (AWS)
CosmosDB (Azure)
Memcache (warning!)
Redis

NO ACID out of the box. Be careful.

ORM:
The Object Relational Mapper

- Abstraction
- Language Consistency
- Speed of Development
- Database Portability

+ ORM: SQLAlchemy

- SQLAlchemy is a Python library that provides tools and abstractions for working with databases.
- It offers a high-level SQL expression language and an Object-Relational Mapping (ORM) framework.
- Developers can interact with databases using Python objects, making database operations more intuitive.

ORM: SQLAlchemy

- **Database-Agnostic:** Consistent API across different database backends (e.g., PostgreSQL, MySQL, SQLite, SQL Server*).
- **SQL Expression Language:** Construct queries using Python code (higher abstraction than raw SQL).
- **Performance Optimization:** Features like connection pooling, lazy loading, and query caching.
- **Integration with Frameworks:** Seamless integration with popular web frameworks like Flask and Django.

The Future?

- Object Oriented Database
  - Combine data and operations on those data
  - Allows for inheritance
  - Oracle (Object-Relational Database)
  - PostgreSQL (open source object-relational DBMS)

- XML and XML DBMS
  - XML designed to transport and store data initially envisioned as moving data across the web (w3schools.com)
  - XML Database Management System manages that data

+

Next up – Data Modeling and How To Build a Better Relationship!

- A few words about data models.