15CSE302 Database Management Systems

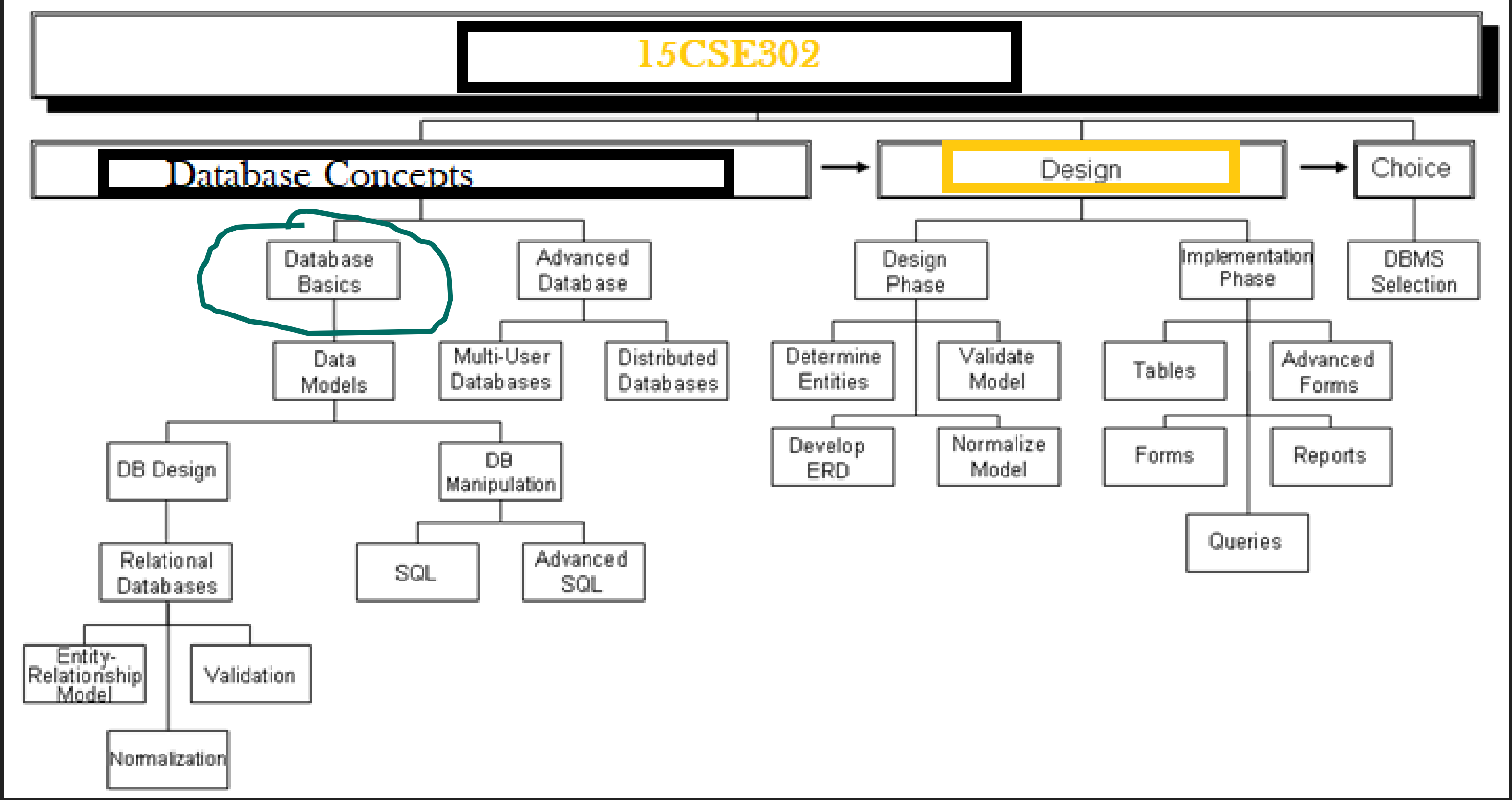
Lecture 1 Introduction

B.Tech /III Year CSE/V Semester

L T P C 2 0 2 3

DBMS Team
Dr G Jeyakumar
Bindu K R
Dr Priyanka Kumar
R. Manjusha
Department of CSE
Amrita School of Engineering

Syllabus

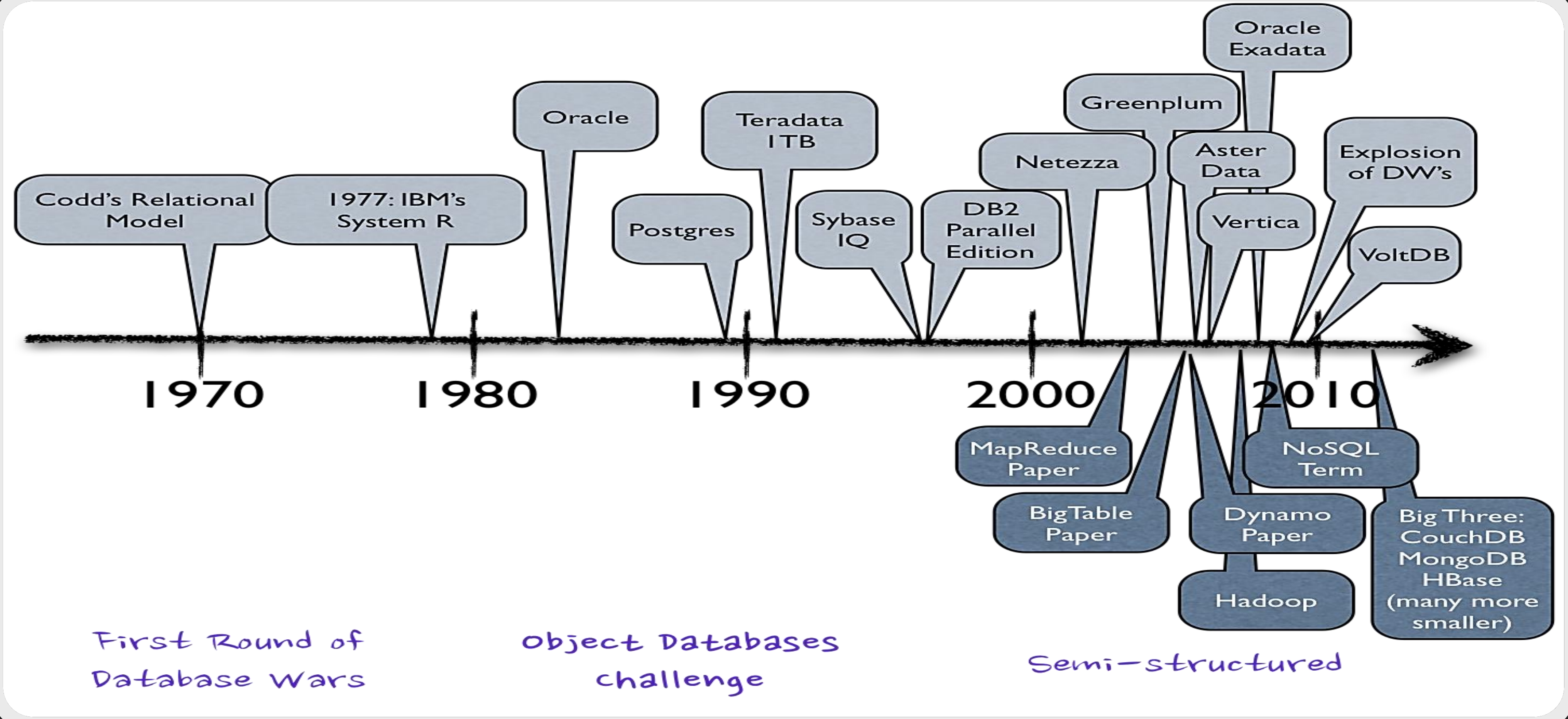


15CSE302
Database Concepts
Design
Choice
Database Basics
Advanced Database
Design Phase
Implementation Phase
DBMS Selection
Data Models
Multi-User Databases
Distributed Databases
Determine Entities
Validate Model
Tables
Advanced Forms
Develop ERD
Normalize Model
Forms
Reports
DB Design
DB Manipulation
Queries
Relational Databases
SQL
Advanced SQL
Entity-Relationship Model
Validation
Normalization

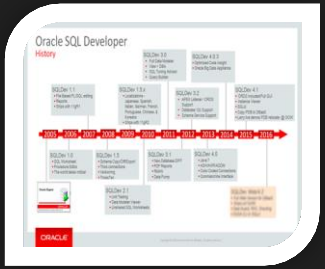
Contents

- Database Timeline
- Database Terminologies
- Purpose of Database Systems
- Database System Applications
- Components of a DBMS

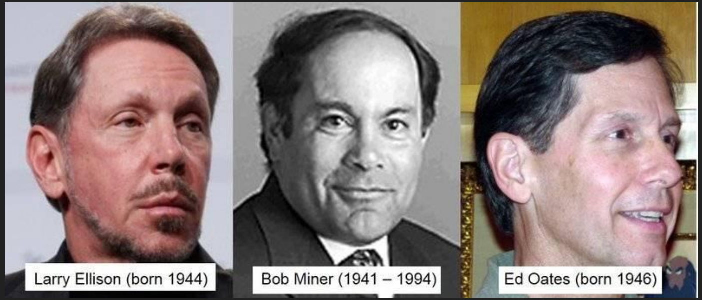
History of databases



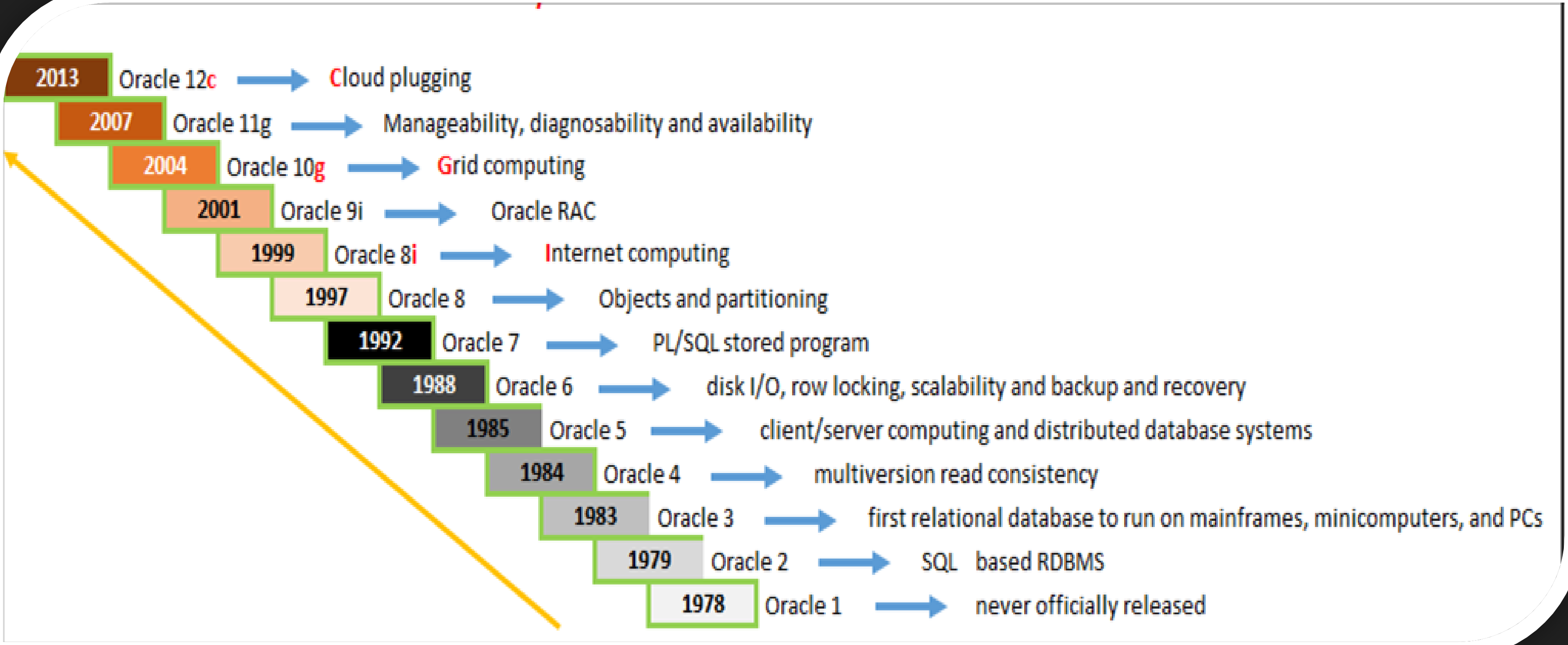
Oracle TimeLine



- Oracle was founded on June 16, 1977 by
  Larry Ellison, Bob Miner and Ed Oates
  under the name Software Development Laboratories (SDL).
- Until 1979, the company did not succeed with this name, and in 1979, three adventurous friends who changed the company name to Relational Software Inc worked in Relational Software Inc. until 1982.
- The brilliant trio, which has consistently focused on Database management systems and made its first database trial with IBM, failed.
- In one of the following experiments, the Oracle Database System developed under the leadership of Bob Miner.
- In 1982, the name of the company was identified with the name of its products and changed to Oracle Systems Corporation.
- It was changed to Oracle Corporation in 1995 and this name has continued to this day.



Oracle Timeline



Oracle TimeLine

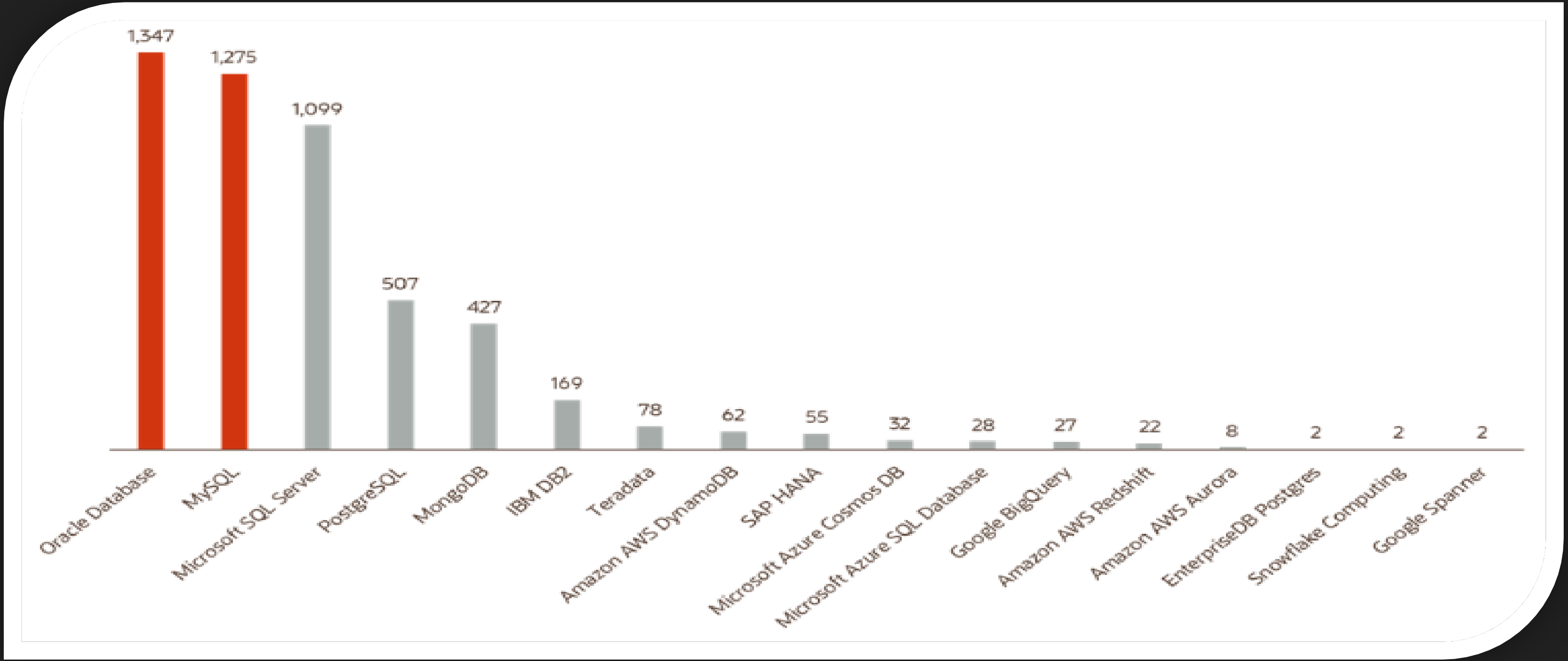
Oracle 11gR2, 2009	Data Reduction, Hybrid Columnar Compression, Cluster File System, Golden Gate Replication, Database Appliance
Oracle 12cR1, 2013	Multitenant architecture, In-Memory Column Store, Native JSON, SQL Pattern Matching, Database Cloud Service
Oracle 12cR2, 2016	Native Sharding, Zero Data Loss Recovery Appliance, Exadata Cloud Service, Cloud at Customer
Oracle 18c, 2018	Autonomous Database, Data Guard Multi-Instance Redo Apply, Polymorphic Table Functions, Active Directory Integration
Oracle 19c, 2019	Automatic Indexing, Data-guard DML Redirect, Partitioned Hybrid Tables, Real-time Stats + Stats Only Queries

Oracle Database Release 20c New Features

- Big Data and Data Warehousing Solutions
- Security Solutions
- Performance and High-Availability Options
- Oracle Sharding
- Tools and Languages
- Database Upgrade and Utilities

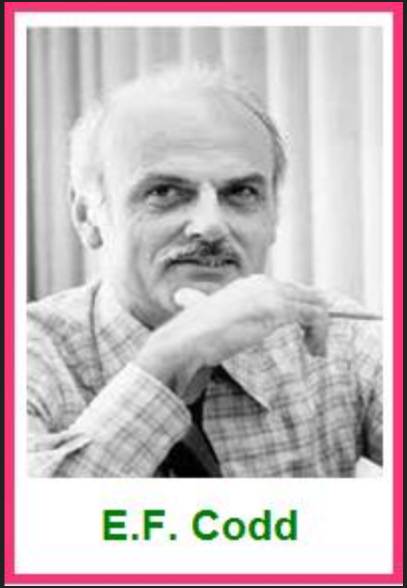
Popularity of Databases

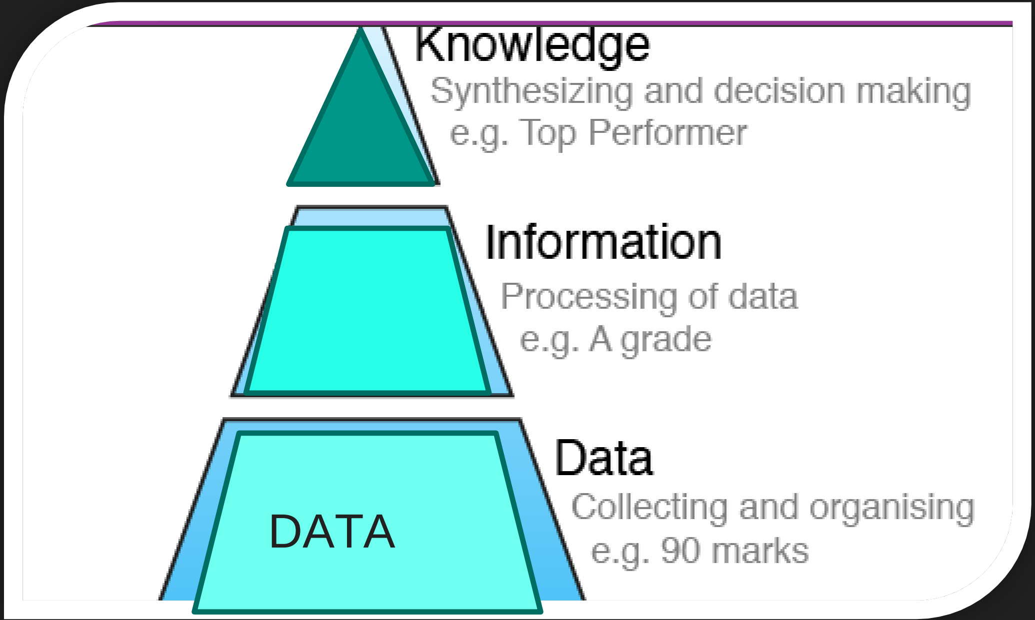
- https://blogs.oracle.com/database/oracle-databases-top-db-engines-ranking



Database Terminologies

Data, Information and Knowledge





Database Terminologies
Data, Information and Knowledge

Data
- 100

Information
- 100 miles

Knowledge
- 100 miles is quite a far distance.

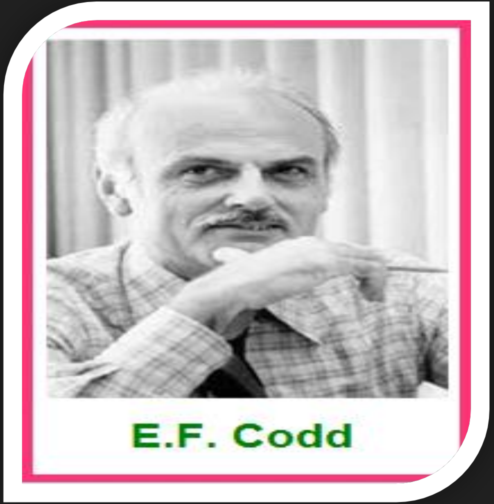
Wisdom
- It is very difficult to walk 100 miles by any person, but vehicle transport is okay

Database Terminologies
Data, Information and Knowledge

Term	Concept	Examples
Data	• Captured symbols and signal readings (recorded and stored) • Objective facts (numbers, symbols, figures) with no context or interpretation • Descriptions of events	• Raw monitoring results • Results of polls • Records of complaints from stakeholders
Information	• Message that contains relevant meaning for a decision or action • Data in context • Meaning or sense of data arising from its interpretation	• Water quality in a particular location and time or period of time • Causes of complaints from stakeholders • Trends in socio-economic indicators for the municipality
Knowledge	• Cognition (know-what)	• Effectiveness of ecological

Edgar F. Codd Creator of Databases

- **E. F. Codd** first described **relational database theory** in his landmark paper "A Relational Model of Data for Large Shared Data Banks," published in the Communications of the ACM (Association for Computing Machinery) in **June, 1970.** https://dl.acm.org/doi/pdf/10.1145/362384.362685
- E.F. Codd passed away on April 18, 2003, at the age of 79



Information Retrieval P. BAXENDALE, Editor

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 models of the data. In Section 1, inadequacies of these models are discussed. A model based on *n*-ary relations, a normal form for data base relations, and the concept of a universal data sublanguage are introduced. In Section 2, certain operations on relations (other than logical inference) are discussed and applied to the problems of redundancy and consistency in the user's model.

KEY WORDS AND PHRASES: data bank, data base, data structure, data organization, hierarchies of data, networks of data, relations, derivability, redundancy, consistency, composition, join, retrieval language, predicate

The relational view (or model) of data described in Section 1 appears to be superior in several respects to the graph or network model [3, 4] presently in vogue for non-inferential systems. It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation and organization of data on the other.

A further advantage of the relational view is that it forms a sound basis for treating derivability, redundancy, and consistency of relations—these are discussed in Section 2. The network model, on the other hand, has spawned a number of confusions, not the least of which is mistaking the derivation of connections for the derivation of relations (see remarks in Section 2 on the "connection trap").

Finally, the relational view permits a clearer evaluation of the scope and logical limitations of present formatted data systems, and also the relative merits (from a logical standpoint) of competing representations of data within a single system. Examples of this clearer perspective are cited in various parts of this paper. Implementations of systems to support the relational model are not discussed.

1.2. Data Dependencies in Present Systems

The provision of data description tables in recently developed information systems represents a major advance toward the goal of data independence [5, 6, 7]. Such tables facilitate changing certain characteristics of the data representation stored in a data bank. However, the variety

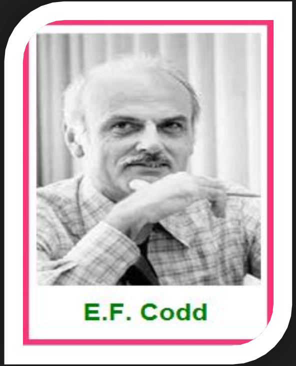
Codd's rules

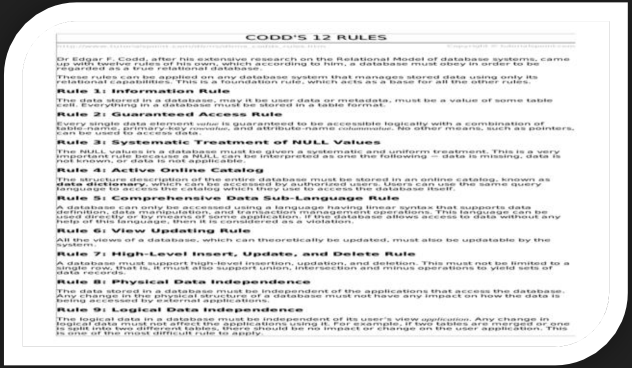
Codd's 12 rules are a set of thirteen rules (numbered zero to twelve) proposed by Edgar F. Codd

a pioneer of the relational model for databases,

designed to define what is required from a database management system in order for it to be considered relational

i.e., a relational database management system RDBMS





CODD'S 12 RULES

Dr Edgar F. Codd, after his extensive research on the Relational Model of database systems, came up with twelve rules of his own, which according to him, a database must obey in order to be regarded as a true relational database.

These rules can be applied on any database system that manages stored data using only its relational capabilities. This is a foundation rule, which acts as a base for all the other rules.

Rule 1: Information Rule

The data stored in a database, may it be user data or metadata, must be a value of some table cell. Everything in a database must be stored in a table format.

Rule 2: Guaranteed Access Rule

Every single data element *value* is guaranteed to be accessible logically with a combination of table-name, primary-key *rowvalue*, and attribute-name *columnvalue*. No other means, such as pointers, can be used to access data.

Rule 3: Systematic Treatment of NULL Values

The NULL values in a database must be given a systematic and uniform treatment. This is a very important rule because a NULL can be interpreted as one the following — data is missing, data is not known, or data is not applicable.

Rule 4: Active Online Catalog

The structure description of the entire database must be stored in an online catalog, known as **data dictionary**, which can be accessed by authorized users. Users can use the same query language to access the catalog which they use to access the database itself.

Rule 5: Comprehensive Data Sub-Language Rule

A database can only be accessed using a language having linear syntax that supports data definition, data manipulation, and transaction management operations. This language can be used directly or by means of some application. If the database allows access to data without any help of this language, then it is considered as a violation.

Rule 6: View Updating Rule

All the views of a database, which can theoretically be updated, must also be updatable by the system.

Rule 7: High-Level Insert, Update, and Delete Rule

A database must support high-level insertion, updation, and deletion. This must not be limited to a single row, that is, it must also support union, intersection and minus operations to yield sets of data records.

Rule 8: Physical Data Independence

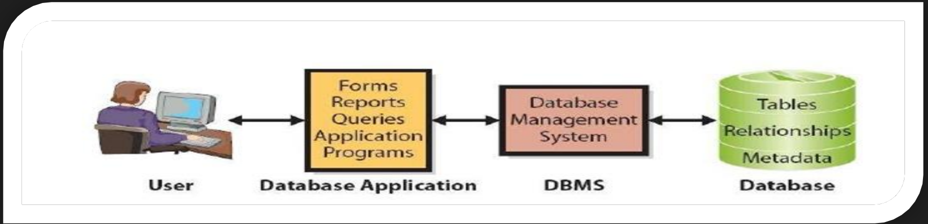
The data stored in a database must be independent of the applications that access the database. Any change in the physical structure of a database must not have any impact on how the data is being accessed by external applications.

Rule 9: Logical Data Independence

The logical data in a database must be independent of its user's view *application*. Any change in logical data must not affect the applications using it. For example, if two tables are merged or one is split into two different tables, there should be no impact or change on the user application. This is one of the most difficult rule to apply.

Terminologies

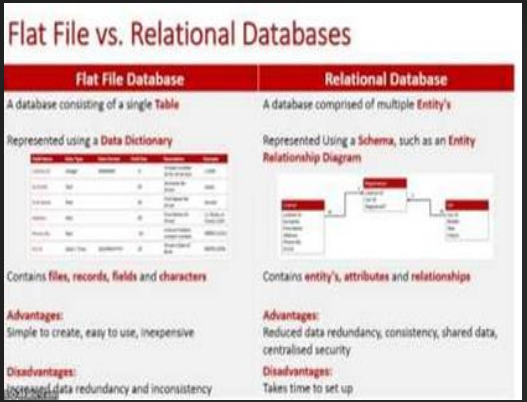
- A **Database** is a shared collection of logically related data and description of these data, designed to meet the information needs of an organization
- A **Database Management System** is a software system that enables users to define, create, maintain, and control access to the database.
- Database Systems typically have high cost and they require high end hardware configurations.



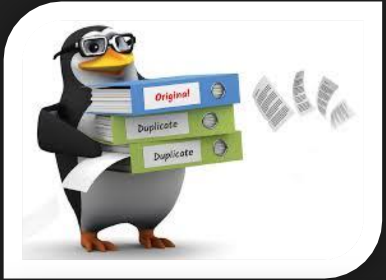
- An **Application Program** interacts with a database by issuing an appropriate request (typically a SQL statement)

Purpose of Database Systems

- Data used to be stored in flat files and can be accessed using any programming language.
- The file based approach suffers following problems:
  - Dependency of program on physical structure of data
  - Complex process to retrieve data
  - Loss of data on concurrent access
  - Inability to give access based on record (Security)
  - Data redundancy



Drawbacks of using file systems to store data

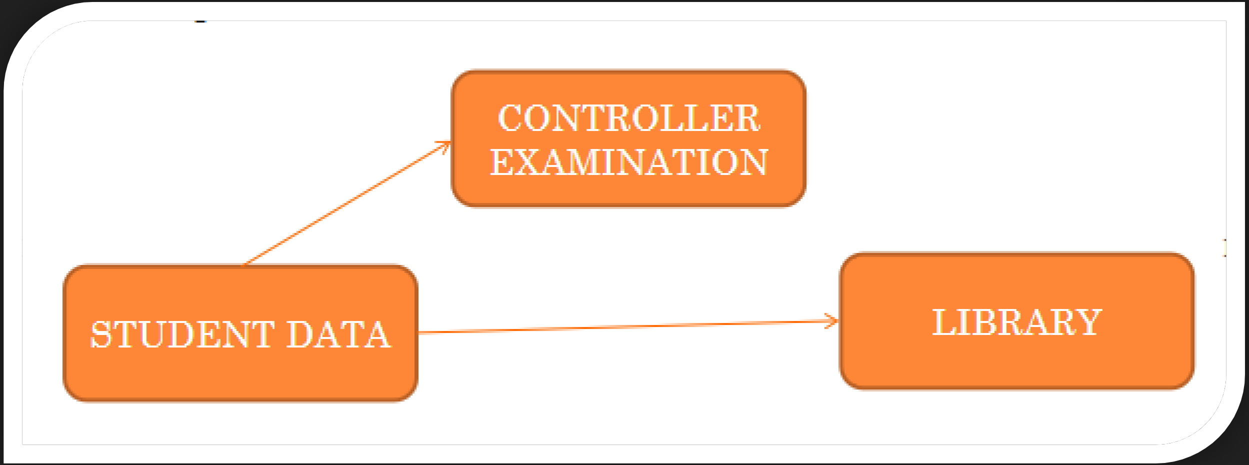


- Data redundancy and inconsistency
- Difficulty in accessing data
- Data isolation
- Integrity problems
- Atomicity of updates
- Concurrent access by multiple users
- Security problems
- Hard to provide user access to some, but not all, data

Database systems offer solutions to all the above problems

Data redundancy and inconsistency

- Data redundancy and inconsistency

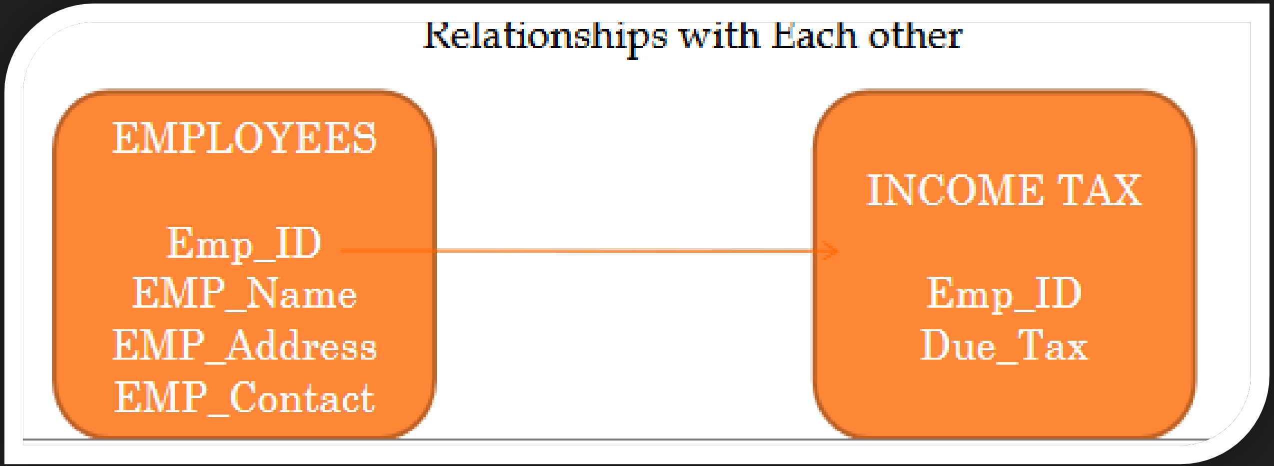


Difficulty in accessing data

- File processing is very difficult.

Data isolation

- **Data Isolation Means Scattered Data**



Data Integrity

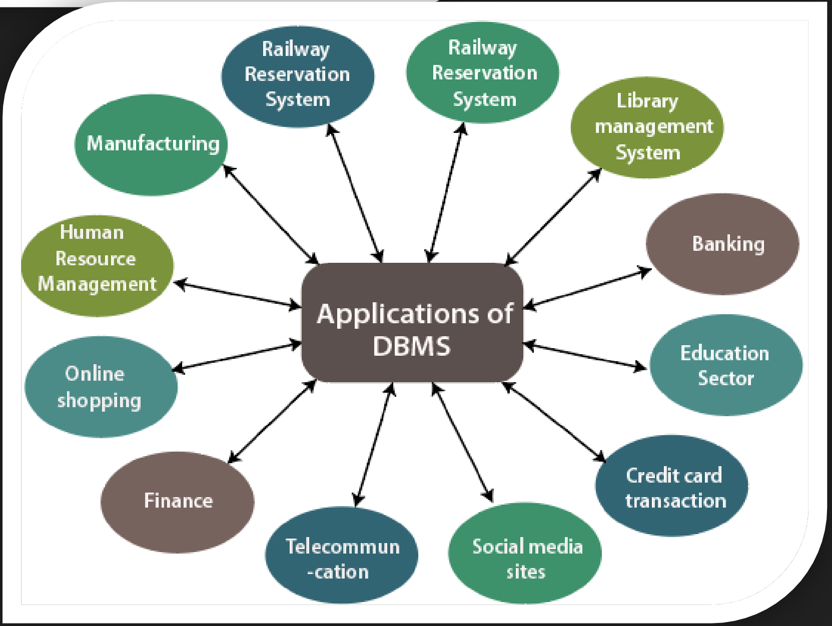
- Data Integrity Deals with the Correctness and Accuracy of Data

    E.g. Age Limit to Apply for a course

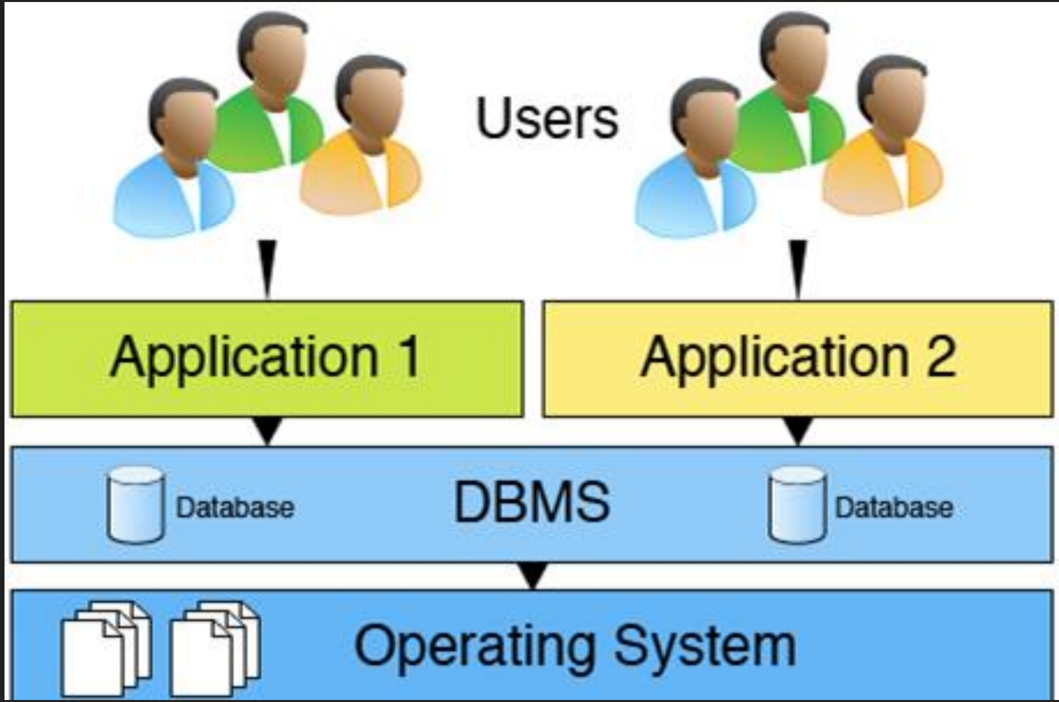
    Roll Number cannot be Negative

Database Applications

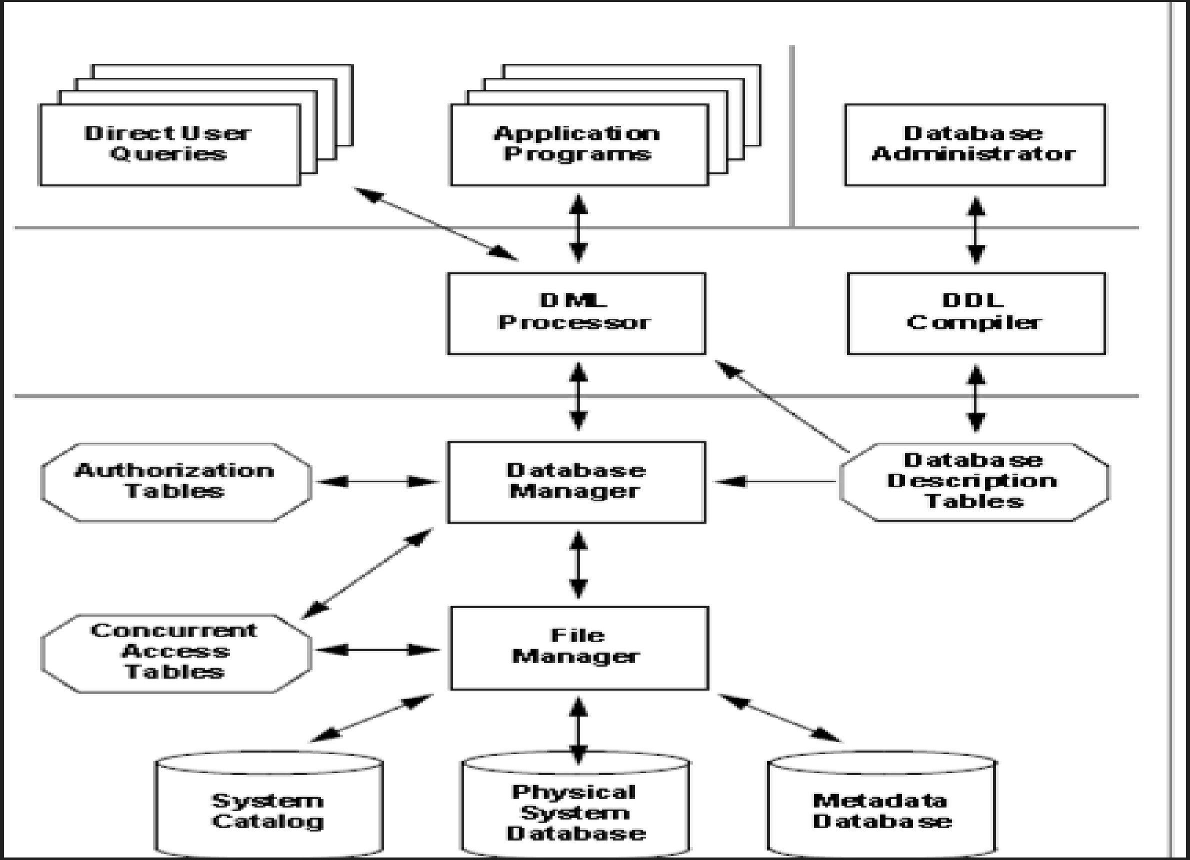
- Banking: transactions
- Airlines: reservations, schedules
- Universities: registration, grades
- Sales: customers, products, purchases
- Online retailers: order tracking, customized recommendations
- Manufacturing: production, inventory, orders, supply chain
- Human resources: employee records, salaries, tax deductions



Database Management Systems



Components of a Database System



Summary

- Database Timeline
- Database Terminologies
- Purpose of Database Systems
- Database System Applications
- Components of a DBMS

Next Session

- Database Abstraction
- Instances and Schemas
- Data Models
- Database Users

References

- https://docs.oracle.com/en/database/oracle/oracle-database/20/newft/new-features.html
- https://www.pda.org/scientific-and-regulatory-affairs/regulatory-resources/data-integrity
- https://www.digipay.guru/blog/all-you-need-to-know-about-agency-banking/
- https://md.ekstrandom.net/teaching/cs4332-f15.pdf
- https://https://bit.ly/31eE2Ar
- https://ipronline.com/oracle-the-pioneers-of-the-software-world/

About Me

Bindu K R

Assistant Professor

Areas of Interests:

1. NLP
2. Information Retrieval
3. Deep Learning

E-mail:j_bindu@cb.amrita.edu

Thank You

Happy to answer any questions ! ! !