3-4 → class | 10 mins break | 5.00 | 10 mins break | 5.50

ENIAC 40s electronic computer discovered
CIA - Computationally Intensive applications

60s DIA - Data Intensive applications → Ex - ticket reservation system

40s to (Gap → due to storage media is slow, it is sequential) 60s → eg - punch cards

Magnetic disks → large amounts of data (storage media is non sequential)

Database management system - IBM / IMS

Engine → DBMS
Fuel → Data is important and it is fuel for DBMS

DIA
    DOA
    {application logic
    {data access logic
Data

Eg - Vehicle drivers

Ticket Reservation system
Application logic
Data access logic

Hospital
Application logic
Data access logic

People decoupled as per their needs (Decoupled)

Ticket	Hospital
AL	AL

logical data representation (data) → This is called the data storage transparency (File system)

Transparency - Hides a lot of technical details

File system allows us to build Data Intensive Applications

Building DIA for simple scenario
Employee management system for his company
What question would you ask →
1. Specification details
2. What kind of system
3. What info you would like me to store

Code which will implement employment.txt

eid/ ename /salary /did /dname /dsize /dmanager
e1 /Jones /2000 /d1 /software **/1 (2) /**e1
e2 /Smith /2000 /d2 /hardware /1 /e2
e3 /Brown /1000 /d1 /software /2 /e1

Data inconsistency

To make it consistent we have to scan the whole file and make it consistent

This is an example of non-scalable design

Difference:
scalable - if a system is scalable there is a very nice relationship between performance and load
non-scalable - opposite

Load increases then performance should also be increased

to resolve this split it into 2 files

Emp.txt
eid /ename /salary /did
e1 /Jones /2000 /d1
e2 /Smith /2000 /d2
e3 /Brown /1000

Dept.txt
did /dname /dsize /dmgr
d1 /software /2 /e1
d2 /hardware /1 /e2
d1 /software /2 /e1

Appl. logic (query processed)
—---------- Build Consistency Manager —----------
—---------- Recovery Manager —----------
—---------- Concurrency Manager —---------- (For multi user)
File system

Ways to recover:
1. increment the size in dept wise
2. Remove emp record from emp.txt (This is the best option if system crashed while updating emp record)

DBMS - different techniques and optimizations for transparency
DBMS why so successful → many layers but it is query processed

Query it
1. Find all emp with salary > 1000
SQL:
Select name from emp where salary > 1000

procedural query

Creating proper data model

Relational data model - most successful data model so far
IMS
represent in form of simple tables

relational model - user friendly representation of the data so that you can focus which is important

relation is table

flat table
Atomic table

inventor of relational model - edgar codd 1970
SQL will not allow to search in nested structure (non-flat)

IBM - started project system R → Oracle 1980s

cardinality - no of rows - 3
if there is null values make sure to eliminate them

faster acess of data and resynchronize

keys
color make model year price mileage

unique identifier - key

enrollment
sid eod grade

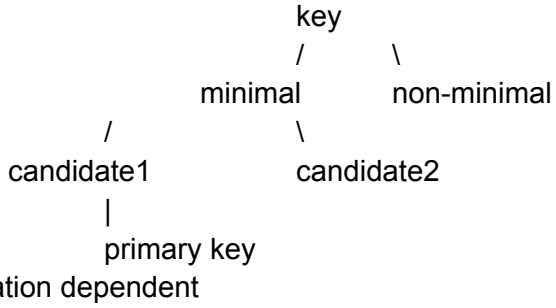
key
1. minimal
2. non-minimal

1.minimal
i. many

Extra points for hosting in cloud

Sept 16th Monday
Cardinality - no of rows
attributes - no of columns

keys can be minimal and non-minimal



application dependent

foreign keys

referential integrity - only refer if that data exists in the table
if the data is deleted then it shouldn't be referred

employee
eid, ename, esalary

eid is primary key

department
did, dname, dmgr

did is primary key
but dmgr is foriegn key for this table

procedural and non-procedular kanguages

Customer
cname, accnum

Jane , A1
Smith, A2
Brown, A3
Black, A4

account
accnum, balance

A1, 1000
A2, 2000
A3, 3000

select cname, balance
from customer, account
where customer.accnum = account.accnum and accnum = 'A1'

1. Find matching tuples
cname, accnum, balance
James, A1, 1000 (3 comparisons)
Smith, A2, 2000 (3 comparisons)
Brown, A3, 3000 (3 comparisons)
Black, A1, 1000 (3 comparisons)

alternative by minimizing the number of comparisons
1. select accnum with accnum = 'A1'
accnum, balance
A1, 1000

→ find matching tuples
1 comparison per accnum so total 4 comparisons

sql will not give you a no order of execution

to be efficient do the selection earlier if you can do it earlier.
accnum = A1(customer ⋈ account)

relational algebra

Select - σ
σ branch_name = "CSE" (college)

σ A = 3 (r)

Project operation - π

π branch_name(account)
π is very dangerous operation it can have duplicates
A, B
1, 2
3, 4
5, 2

π B (r) = 2 tuples
B
2
4

Set?
→ no duplicates
→ no order

Union operation - r ∪ s
{1,2,3} ∪ {3,4} - {1,2,3,4}
union with same schemas

cartesian product is more effective, expensive we need to make some effort to make meaningful tables

Select * from loan where amount > 1200
Select loan_number from loan where amount > 1200

control character → % - any string
(_) → underscore → any character

%_% → it will select all names because it selects names which contains atleast one chracter
%_% → backslash to escape character which slects name that contains underscore

Nested queries

IN (set membership)
we should not mix regular attribute and aggreate function

This is correct - select name from sailor s where rating = (select max(rating) from sailor s)

Imp

select s.rating avg(s.age)
from sailors s
group by s.rating
having avg(s.age) =< all (select avg(s.age) from sailors s group by s.rating)

select branch, sum(A.balance)
from account A
group by A.branch
having >= all (select sum(A.balance)
from account A
group by A.branch)

select branch, sum(A.balance)
from account A
group by A.branch
having >= sum (select sum(A.balance)
from account A
group by A.branch)

closed world assumption
null != null
but in duplicaate management null == null
null = null ⇒ unknown → so it is false

Buffer management in a DBMS
block1, block2, block3 block4
A,B
1,2
3,4
5,6
7,8

B,C
2,4
5,7
6,9
8,10

LRU -

Cost of execution - no. of disk blocks transferred from disk to main memory
Block transfer
b1, b4, b5,
b2 and b4 substituted
b4 and b1 substituted
cost = 5
Better query plan
b1, b4, b5, b2 (b1 substituted)
cost = 4

buffer replacement policies
LRU might be doesn't give better cost

Forced Output → moving all blocks to main memory due to some crash
Data Dictionary storage:
Data base system
→ Table
Name, Cardinality, attribute
account, 1, 3
Table, 2, 3

DB:
Account (acc_no, b_name, balance)

Relational Representation of System metadata
Data dictionary is very essential for query optimization

- Indexing is something that makes Data base management possible
- the engine that search web is called google spider
- this google spider collects info of pages from everywhere and puts in "index"
- index is a special structure which allows to search very fast and allows to search in that space

searching linear is time consuming
indexed search is much faster