CS 386D, Database Systems, Sp '20

Introduction: Organization and Abstractions

Lectures 1 and 2

Prof. Daniel P. Miranker

Objectives:
- Course objectives
- People level introduction (background)
- Form and Function
  - Database
  - Database Management System (DBMS)
  - Data Model(s)
  - Relational vs. Massively Parallel, Cloud Database Architecture

Welcome to CS386D, Database Systems

Database Systems Are Exciting

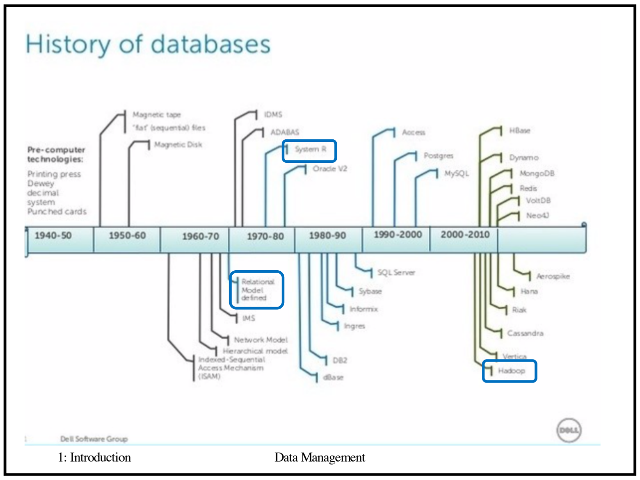
Welcome to CS386D, Database Systems

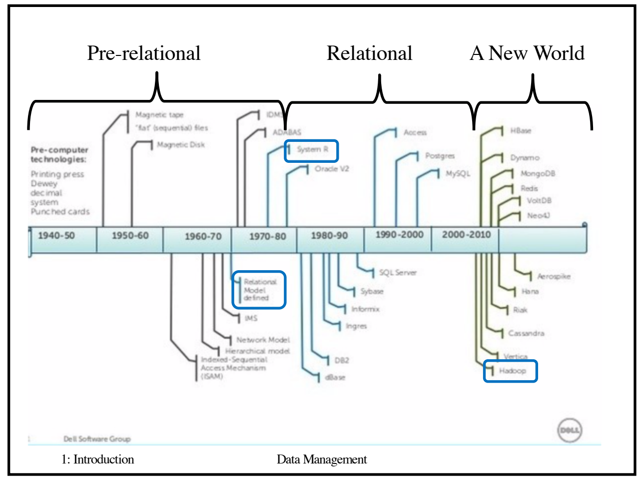
Database Systems Are Exciting
Again

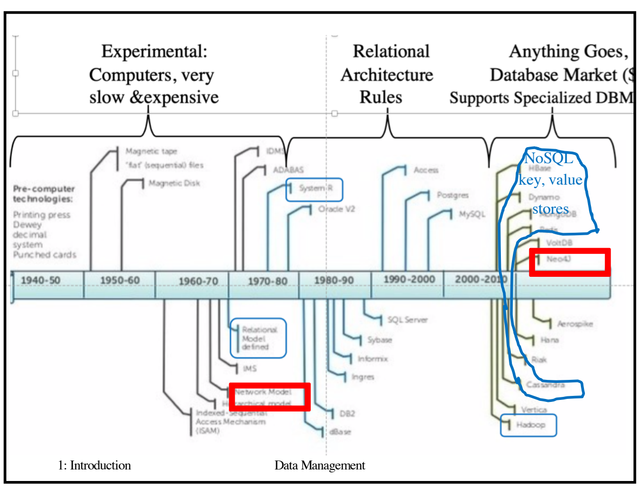
Part 1: The Big Picture

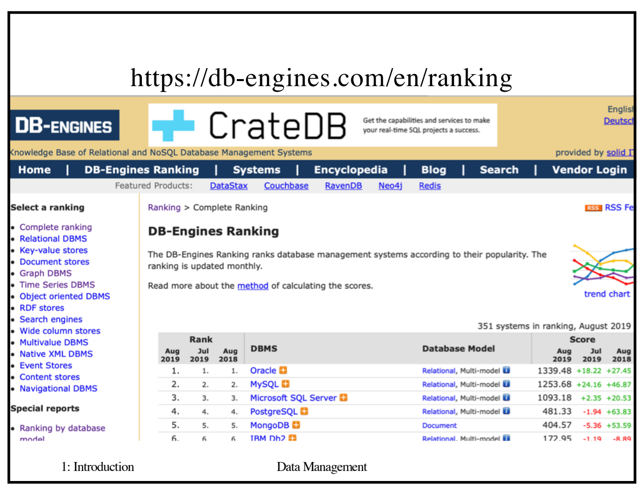
Database Management Systems (DBMSs)

- What not "database"?
- RE: Language usage and data science 😝
  - Many terms have multiple meanings
  - Many concepts are labeled with multiple terms


History of databases
Pre-computer technologies:
Printing press
Dewey decimal system
Punched cards
Magnetic tape
"flat" (sequential) files
Magnetic Disk
IDMS
ADABAS
System R
Oracle V2
Access
Postgres
MySQL
HBase
Dynamo
MongoDB
Redis
VoltDB
Neo4J
1940-50
1950-60
1960-70
1970-80
1980-90
1990-2000
2000-2010
SQL Server
Sybase
Informix
Ingres
Relational Model defined
IMS
Network Model
Hierarchical model
Indexed-Sequential Access Mechanism (ISAM)
DB2
dBase
Aerospike
Hana
Riak
Cassandra
Vertica
Hadoop
Dell Software Group
1: Introduction
Data Management

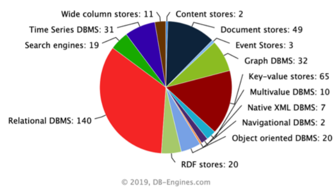

Pre-relational
Relational
A New World
Pre-computer technologies:
Printing press
Dewey decimal system
Punched cards
Magnetic tape
"flat" (sequential) files
Magnetic Disk
IDMS
ADABAS
System R
Oracle V2
Access
Postgres
MySQL
HBase
Dynamo
MongoDB
Redis
VoltDB
Neo4J
1940-50
1950-60
1960-70
1970-80
1980-90
1990-2000
2000-2010
SQL Server
Sybase
Informix
Ingres
Relational Model defined
IMS
Network Model
Hierarchical model
Indexed-Sequential Access Mechanism (ISAM)
DB2
dBase
Aerospike
Hana
Riak
Cassandra
Vertica
Hadoop
Dell Software Group
1: Introduction
Data Management


Experimental: Computers, very slow &expensive
Relational Architecture Rules
Anything Goes, Database Market ($ Supports Specialized DBM
NoSQL key, value stores
Pre-computer technologies:
Printing press
Dewey decimal system
Punched cards
Magnetic tape
"flat" (sequential) files
Magnetic Disk
IDMS
ADABAS
System R
Oracle V2
Access
Postgres
MySQL
HBase
Dynamo
VoltDB
Neo4J
1940-50
1950-60
1960-70
1970-80
1980-90
1990-2000
2000-2010
SQL Server
Sybase
Informix
Ingres
Relational Model defined
IMS
Network Model
Hierarchical model
Indexed-Sequential Access Mechanism (ISAM)
DB2
dBase
Aerospike
Hana
Riak
Cassandra
Vertica
Hadoop
1: Introduction
Data Management


https://db-engines.com/en/ranking
DB-ENGINES
CrateDB
Get the capabilities and services to make your real-time SQL projects a success.
Knowledge Base of Relational and NoSQL Database Management Systems
provided by solid IT
Home | DB-Engines Ranking | Systems | Encyclopedia | Blog | Search | Vendor Login
Featured Products: DataStax Couchbase RavenDB Neo4j Redis
Select a ranking
Complete ranking
Relational DBMS
Key-value stores
Document stores
Graph DBMS
Time Series DBMS
Object oriented DBMS
RDF stores
Search engines
Wide column stores
Multivalue DBMS
Native XML DBMS
Event Stores
Content stores
Navigational DBMS
Special reports
Ranking by database model
Ranking > Complete Ranking
RSS Feed
DB-Engines Ranking
The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.
Read more about the method of calculating the scores.
trend chart
351 systems in ranking, August 2019
Rank Aug 2019 Jul 2019 Aug 2018 DBMS Database Model Score Aug 2019 Jul 2019 Aug 2018
1. 1. 1. Oracle Relational, Multi-model 1339.48 +18.22 +27.45
2. 2. 2. MySQL Relational, Multi-model 1253.68 +24.16 +46.87
3. 3. 3. Microsoft SQL Server Relational, Multi-model 1093.18 +2.35 +20.53
4. 4. 4. PostgreSQL Relational, Multi-model 481.33 -1.94 +63.83
5. 5. 5. MongoDB Document 404.57 -5.36 +53.59
6. 6. 6. IBM Db2 Relational, Multi-model 172.95 -1.19 -8.89
1: Introduction
Data Management

https://db-engines.com/en/ranking_categories

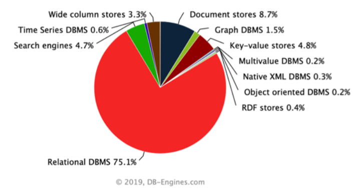
Number of systems per category, August 2019



DB-Engines lists 351 different database management systems, which are classified according to their database model (e.g. relational DBMS, key-value stores etc.).
This pie-chart shows the number of systems in each category. Some of the systems belong to more than one category.

https://db-engines.com/en/ranking_categories

Ranking scores per category in percent, August 2019



This chart shows the popularity of each category. It is calculated with the popularity (i.e. the ranking scores) of all individual systems per category. The sum of all ranking scores is 100%.

Computers at t = "the rise of RDBMSs"

1981:
- Many competing relational query languages
- The DEC Vax 11/780 – the machine to have in a CS Dept.
- Workstations (i.e. a personal computer), for the privileged
  - $80K, not adjusted for inflation
  - $19K starting salary for developer with a BS in CS
- Miranker entered graduate school

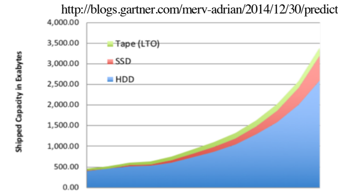
The Vax 11/780
- Adjusted for inflation around $750,000
- Speed = 1 32 bit core at 1Mhz
- Average disk seek time, 21msec.

But today we have SSD's! (or do we?)

- "By 2018, HDDs will still account for 75% to 85% of all petabytes shipped to the server and ECB storage markets, but solid-state solutions will expand to account for 15% to 25% of all mission-critical, near-line and archived data. (John Monroe and Joseph Unsworth.)"

Merv Adrian, VP Gartner

http://blogs.gartner.com/merv-adrian/2014/12/30/prediction-is-hard-especially-about-the-future/



Forbes Magazine, May 14 2018

More about SSD

- OS still introduces delays,
  i.e. context switches // how long?
  To do I/O // how many?
- Even with SSD, not back to ratios comparable to the Vax.
- Persistent Memory (non-volatile RAM) may change this? // why/how

Course Objectives:
Primarily Query Execution Focused

- Learn from fundamentals
  - Algorithms and methods,
    - Data storage & access
    - Query operators
  - Benchmarking and Performance Analysis
  - Optimization

Course Objectives:
Primarily Query Execution Focused

- Learn from fundamentals
  - Algorithms and methods,
    - Data storage & access
      // cope with disks, HDD & SDD
    - Query operators
      // algorithmic complexity, constants, parallelism, composability
  - Benchmarking and Performance Analysis
    // workloads, selectivity, data skew
  - Optimization
    // fast, efficient; these are different

Goal:

By the end of the semester, you will be able to,

- Independent of data and storage model
  - Physically optimize a database
  - Logically optimize a database
  - Build, and/or research the development of new database management query systems

Example:

- Definition of join:

Example

Select B,D
From R,S
Where R.A = "c" $\wedge$ S.E = 2 $\wedge$ R.C=S.C

R	A	B	C
	a	1	10
	b	1	20
	c	2	10
	d	2	35
	e	3	45

S	C	D	E
	10	x	2
	20	y	2
	30	z	2
	40	x	1
	50	y	3

```
Select B,D
From R,S
Where
R.A = "c"
∧  S.E = 2
∧  R.C=S.C
```

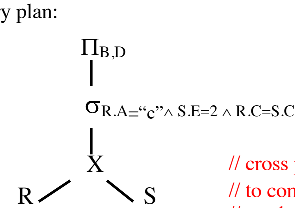
Answer

B	D
2	x

Given:

```
Select B,D
From R,S
Where R.A = "c"  ∧  S.E = 2
    ∧  R.C=S.C
```

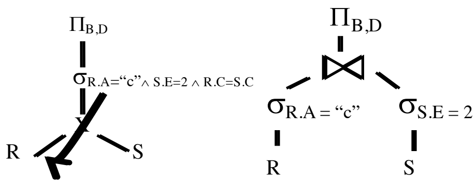
We get a query plan:



// cross product is n^2
// to compute &
// produces n^2 results
// How big in n?

OR: $\Pi_{B,D} [\sigma_{R.A=\text{“c”} \wedge S.E=2 \wedge R.C = S.C} (R X S)]$

We say *"we pushed the selects down"*



Computationally what happens?
(btw, join is still $O(n^2)$)

Join arguments are smaller

Nature of the Course
Nature of the Science

- Relational algebra(s) & calculus(s),
  - form a bedrock
- Provide formal semantics
  - A well understood foundation…
    - for defining semantics
    - providing a basis for provably correct optimizing transforms

In all those years of RDBMS domination

- Relational database extensions
  - Objects
    - Past, C++, Java, XML
    - Current: JSON
  - Graphs
    - Yes, really. Don't believe the graph DB people.
      - Lots more about this later
  - Big data, no problem
- NoSQL – that's very funny.
  - Real advances, but SQL is no less useful, (for NoSQL platforms)

So:

1. RDBMSs are a core, and still prevailing foundation.
2. This course is dominated by traditional material.
   - The major DB texts have not had a new edition in 10 years.
3. Last year: a major reorganization of the class
   - Contemporary cloud centric DB material integrated; not left to the end of the semester

Part 2: Course Administration

Personal Introduction

- State of computing and technology circa 1981
  - VAX 11/780
  - THE IBM PC, not quite yet.
  - Cray XMP – the Supercomputer, NSA bought most of them.
  - Introduction of VLSI, 32 bit processor chip
    - Data path
    - Instruction execution
    - No cache
    - No memory management

Me: Dan Miranker

Operational: August, 1985, DADO 2, 1023 cores.
Goal: Do AI on a lot of data

Research:
- search – AI inference
- database indexing
- data integration

Practice:
- Founded 2 startups
  - a dot-com
  - Capsenta (exited last year)
- Deployed (data intensive) AI applications
  - 3 big companies
  - Air Force Intelligence Agency

Prerequisite Knowledge

- An undergraduate course in databases is not necessary.

But,

- I will assume you know
  - Basics of SQL programming (Ch. 6)
    - Online SQL Tutorial: http://www.w3schools.com/sql/default.asp
  - Relational algebra (Ch. 2, 5.1, 5.2)
    - select, project, join.
    - You can (can you?) prove the equivalence of two relational expressions by applying algebraic transforms and identities.
      - e.g. RxS = SxR

Grading

- 2 midterms
- Final, comprehensive,
  - Scheduled by the registrar for finals week
- Homework, a large part of the class
  - Homework will include an integrated series of implementation assignments that will be like a small term project concerning benchmarking databases and optimizing relational DBMS

Late Homework Policy

- One late homework, 3 days maximum,
  - no questions asked
- More than one late homework, don't ask*.

* Hospitalized, death in the family? You may ask.

Learning Management Software

- Piazza: Open question answering
- UT Box Storage: Distribution of documents.
  - HW assignment, reading, sol'n sets, etc.
  - Strongly recommended: Install a driver to integrate box with your local file management
- Canvas:
  - Electronic turn-in of homeowork
  - Gradebook
    - For recording and publishing grades.
    - Do not assume anything calculated by Canvas is correct.

DBMS

- Software
  - There will be assignments where you must use
    - a SQL system and
    - involve gigabytes of data.
  - Necessarily, you must do this,
    - on a machine or cloud of your choice, (not a lab machine)
    - You may choose any DBMS that supports a SQL like language, except MySQL and SQLlite

Break

Part 3: The Basics

Definitions

- Database*: A collection of data
- Database Management System (DBMS): A software system that provides a set of services on a database. [from old Silberschatz and Korth text]

(an old slide)

- Relational Database Management System
  - A DBMS that is founded on the relational model.
- Operating Systems/File Systems
  - ? Service 1__________
  - ? Service 2__________
  - … __________
- facebook.com?

Core Services

- Storage
- Retrieval – form of queries
- Transactions
- Correctness – constraints define correctness
  - Database as an active monitor wrt correct data

Persistent Theme in Databases

- Conceptual
- Logical
- Physical

Persistent Theme in Databases

- Conceptual
  - Abstract, missing details, often seen by the outside world, (including non-technical people, i.e. business people in managment)
- Logical
  - No detail wrt implementation
- Physical
  - Actual implementation

Data Model (1)

"A *data model* is a notation for describing data or information"[text page 17]

Data Model (2)

“Generally consists of three parts:

- Structure of the data…
- Operations on the data …
- Constraints on the data …

Relational Data Model

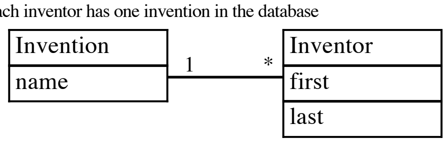
Conceptual: Data appears in tables and rows

Logical: Based on the relational algebra

Physical: Most commonly, store rows in tables.

Conceptual Model (UML)

Each invention may have multiple inventors
Each inventor has one invention in the database



Implementation details are not represented, e.g. data types,
Not even all the attributes are represented

Relational Data Model & SQL

Structure of the data

- SQL: <u>DDL</u> (data definition language)
  - CREATE TABLE Invention (iid integer, name varchar(255));
  - CREATE TABLE Inventor (iid integer, first varchar(255), last varchar(255));

Queries Languages

Three parts

- DDL data definition language
- DML data manipulation language
- DQL data query langague

Physical

Invention	
iid	name
1	dna structure
2	sequencing_machine
3	expression_chips

Inventor		
iid	first	last
1	Francis	Crick
1	James	Watson
1	Rosalyn	Franklin
1	Maurice	Wilkens
2	Lee	Hood
3	David	Botstein

Who invented the sequencing machine?

```
Select last
From Invention, Inventor
Where
    Invention.name="dna structure"
and
    Invention.iid = Inventor.iid
```


Logic

Select last
From Invention, Inventor
Where
Invention.name="dna structure"
and
Invention.iid = Inventor.iid

$\Pi_{last}(\sigma_{name = \text{"dna structure"}}(Invention \bowtie Inventor))$

on board, show pushing selects down

Correctness Constraint

CREATE CONSTRAINT FK_Inventor FOREIGN KEY (Inventor.iid)

REFERENCES Invention(iid)

Invention	
iid	name
1	dna structure
2	sequencing_machine
3	expression_chips

Inventor		
iid	first	last
1	Francis	Crick
1	James	Watson
1	Rosalyn	Franklin
1	Maurice	Wilkens
2	Lee	Hood
3	David	Botstein

Many Data and Storage Models
(now of much greater interest)

- Commercial (trade):
  - Show dbengine ranking web site
  - https://db-engines.com/en/ranking

Data Model Notions

formal commercial/trade

- Network --- graph: Neo4j, RDF
- Hierarchical – XML
- Relational
- Object-oriented -

Data Model Notions

formal commercial/trade BTW

- Network --- graph: Neo4j, RDF CODASYL
- Hierarchical – XML IBM IMS
- Relational
- Object oriented all died

Then came JSON and MongDB

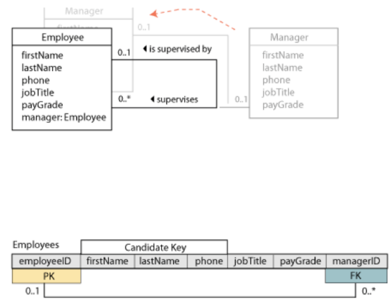
Course Motivation (old slide)

- Modern Developments
  - Commoditization of computer industry
  - Contemporary software engineering practice
- Core Database Architecture
  - How to cope with disks.
    - The only [computational] moving part.
    - Its not changing. (or at least not meaningfully quickly)

What about disks?

- Why do computers have disks? (good)
  - inexpensive, large *persistent*, storage.
    - persistent storage: data is unaltered if the power goes off.
- Why do we wish they didn't? (bad)
  - slow
    - 8-12 msec. seek time. ~ 0.1msec. rotational latency
  - they break

Relational Model for a Graph



Example on board

Part 3: Workloads and Transactions

Workloads

Workload: The contents of the database, a set of database operations, e.g. queries, updates

Workloads

- OLTP: On line transaction processing
- OLAP: On line analytic processing
- Big Data: Batch processing of analytic queries and data mining operations

OLTP

- OLTP: On line transaction processing
  - Large throughput of small transactions in real-time.
    - Small queries (measure in the arity of joins)
    - A number of updates
  - Examples:
    - Point of sales (POS) [cash registers/ etailer sales]
    - Bank withdrawal and deposit

OLAP: On line analytic processing

- Classic query – goes with specialized systems called, precisely "OLAP systems"
  - Total sales by
    - Region
    - Product Category
    - As a trend over time.
- Today: Renamed, "Dimensional Data Models"
  - Conceptual organization is great (for business people and Uis)
  - More powerful machines → less need for specialized implementations

Big Data

- Analysis
  - click streams
  - creation of recommender systems
  - network traffic analysis (e.g. intrusion detection)
  - Market segmentation
    - {feature set defining people} x {set of boxes characterizing what they buy}

Transactions

- Strong (traditional)
- Weak (arrived with Cloud Databases)

Strong Transactions

Definition: A ***Transaction*** is a unit of work that must

- be executed atomically
- in apparent isolation from other transactions
- With a guarantee of durability (that is, the work of completed transactions is never lost) [~from text page 7]

Definition: A ***Transaction**** is a unit of work that must

- be executed atomically

  // multiple side-effects on DB, all or nothing proposition
- in apparent isolation from other transactions

  // like processes in an OS
- With a guarantee of durability (that is, the work of completed transactions is never lost) [~from text page 7]

  // fault tolerance – think banks and money

* I promise the undergraduate class that I will underline the first use of key terms. Grad class – not.

Strong Transactions

- Strong Transcations has ACID properties.
  - Atomicity
  - Concurrency/Consistency
  - Isolation
  - Durability

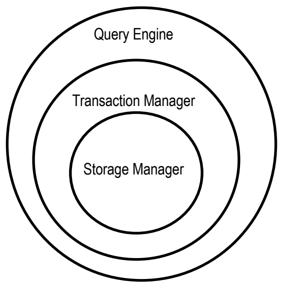
Weak Transaction

- Quorum Consistency
  - In a distributed machine,
  - Each "row" is stored in a number of the machines.
  - An update on one machine propagates to all the machines – that then update.
  - Possible to read an "old" value.

Part 5: Database Architecture

Relational Architecture

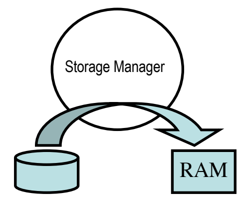
Traditional DBMS Architecture

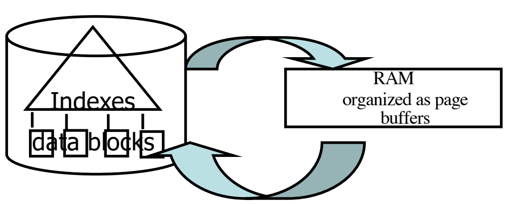


RDBMS Architecture

Storage Manager

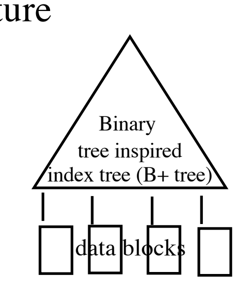
- Exploit memory hierarchy to compensate for slow disks.
  - working sets (from OS)
  - search algorithms
- Specifics
  - manage a *heap* of disk pages
  - allocation of main memory (buffer management)
  - index methods, e.g. B+ tree (access paths)





RDBMS – Basic Storage Structure

- Each table defines a primary key,
- Rows (records) stored, sorted by key, (if possible) contiguous blocks,
- Tree-based (~(log(n)) access to individual rows, by key.



How are these stored, accessed (access path)

Invention	
iid	name
1	dna structure
2	sequencing_machine
3	expression_chips

Inventor		
iid	first	last
1	Francis	Crick
1	James	Watson
1	Rosalyn	Franklin
1	Maurice	Wilkens
2	Lee	Hood
3	David	Botstein

Who invented the sequencing machine?

```
Select last
From Invention, Inventor
Where
    Invention.name="dna structure"
and
    Invention.iid = Inventor.iid
```


RDBMS: Declarative
Maximize "Associative Access"

Name

first	last
Francis	Crick
James	Watson
Rosalyn	Franklin
Maurice	Wilkins

SELECT last
FROM Name
WHERE Name.first = "James"

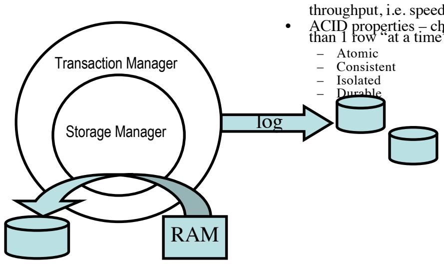
Will return "Watson"

SELECT first
FROM Name
WHERE Name.last = "Watson"

Will return "James"

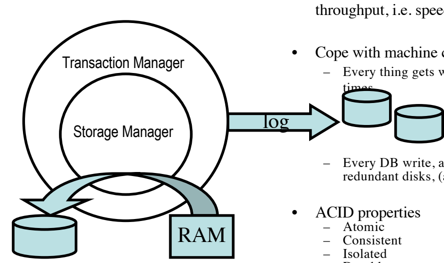
RDBMS Architecture, Transactions!

Transaction Manager
- Manage many users sharing a database, (goal: transaction throughput, i.e. speed)
- ACID properties – change more than 1 row "at a time".
  - Atomic
  - Consistent
  - Isolated
  - Durable



RDBMS Architecture, 2

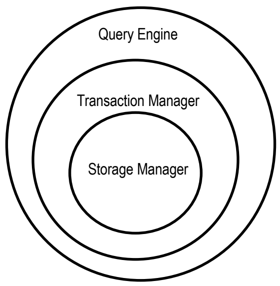
Transaction Manager
- Manage many users sharing a database, (goal: transaction throughput, i.e. speed)
- Cope with machine crashes
  - Every thing gets written at least 3 times
  - Every DB write, also logged to redundant disks, (a.k.a. stable store)
- ACID properties
  - Atomic
  - Consistent
  - Isolated
  - Durable



RDBMS Architecture 3

Query Engine

- SQL execution environment
  - parse
  - compile to logical operators
  - optimize: Choose a good set of access paths and sequence of database operators
    (a.k.a. a physical plan)

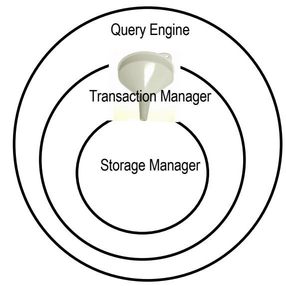


RDBMS Architecture 4

Active Management of Data Correctness

SQL execution environment:

- consistency constraints
- views
- triggers

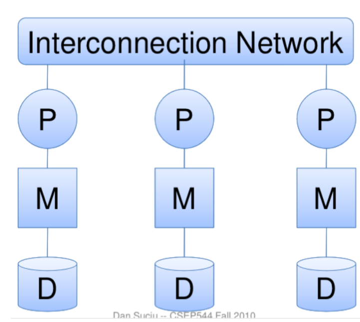


Massively Parallel Cloud Databases

- Primary features:
  - Reliable (→ redundant) storage in the network.
  - More flexible transaction models.
  - Dynamic, often on demand, scaling of resources, (both processing and storage).

- Migrating quickly to SQL ☺ ☹

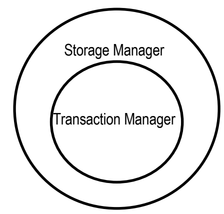
Shared Nothing (DB) Clusters



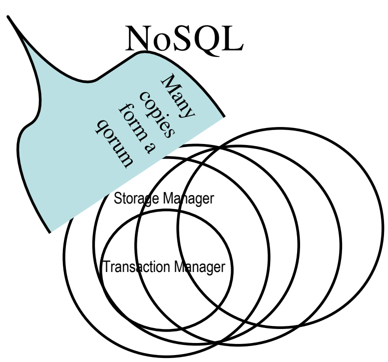
Shared Nothing – Advantages

- Most scalable
  - Minimizes interference by minimizing resource sharing
  - Memory and I/O bandwidth and capacity grow with the number of compute nodes.
- Can use commodity hardware
  - "rack and stack"

NoSQL



NoSQL



NOSQL

- Programmable interface to Cloud storage mechanisms.
- Key, Value model
  - Setname: {(key1, value1), (key2, value2) (key3, value3)…}
  - E.g.
    - name{ (k1, dan), (k2, bob), (k3, bruce)}
    - salary{ (k1, $10), (k2, $1), (k3, $1000)}
    - title {….}

NOSQL – Not as Different as "they" Would Like You to Believe

– E.g.
  - name{ (k1, dan), (k2, bob), (k3, bruce)}
  - salary{ (k1, $10), (k2, $1), (k3, $1000)}
  - title {(k1, professor), (k2, fry cook), (k3, chairman)}

– Employee

Id	Name	Salary	Title
1	dan	10	professor
2	bob	1	fry cook
3	bruce	1000	chairman
…			

Not as Different as They Would Like You to Believe

– OR. e.g.

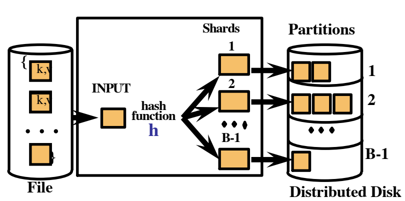
Employee{ (k1, ((name dan), (salary 10), (title professor))),
(k2, ((name bob), (salary 1), (title fry cook))),
(k3, ((name bruce), (salary 1000), (title chairman)))
}

– Employee

Id	Name	Salary	Title
1	dan	10	professor
2	bob	1	fry cook
3	bruce	1000	chairman
…			

Hashing Shards, Replicate for Durability

- Hashing



replicate

Distributed File System

- Hadoop Distributed File System (HDFS)
  – pages/blocks 16 or 64 Mbytes
  – pages are compressed for storage
  – pages are replicated for fault tolerance
  – quorum consistency is the basis of transactions

Bloom filter

- A probabilistic data structure, for testing set membership.

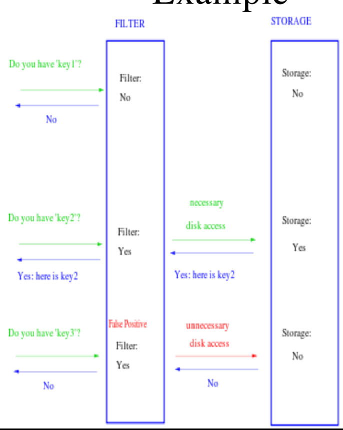
Given a set $S = \{x_1, x_2, \ldots, x_n\}$,

With high probability, is y an element of S?

Further,

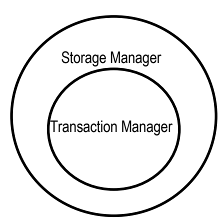
- A 'No" answer, is correct 100% of the time.
  - no *false negatives*
- A "Yes" answer, is actually "maybe".
  - *false positives* are allowed

Example



https://en.wikipedia.org/wiki/File:Bloom_filter_speed.svg

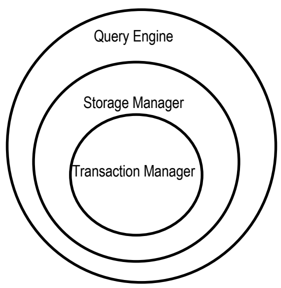
Bloom Filter: A primary indexing method for CloudDB



1: Introduction Data Management & Engineering 92

NoSQL → CloudDB

("NoSQL", hindsight: hadn't built query system yet.)



Integration of Bloom Filters with Query Processing

… my current research