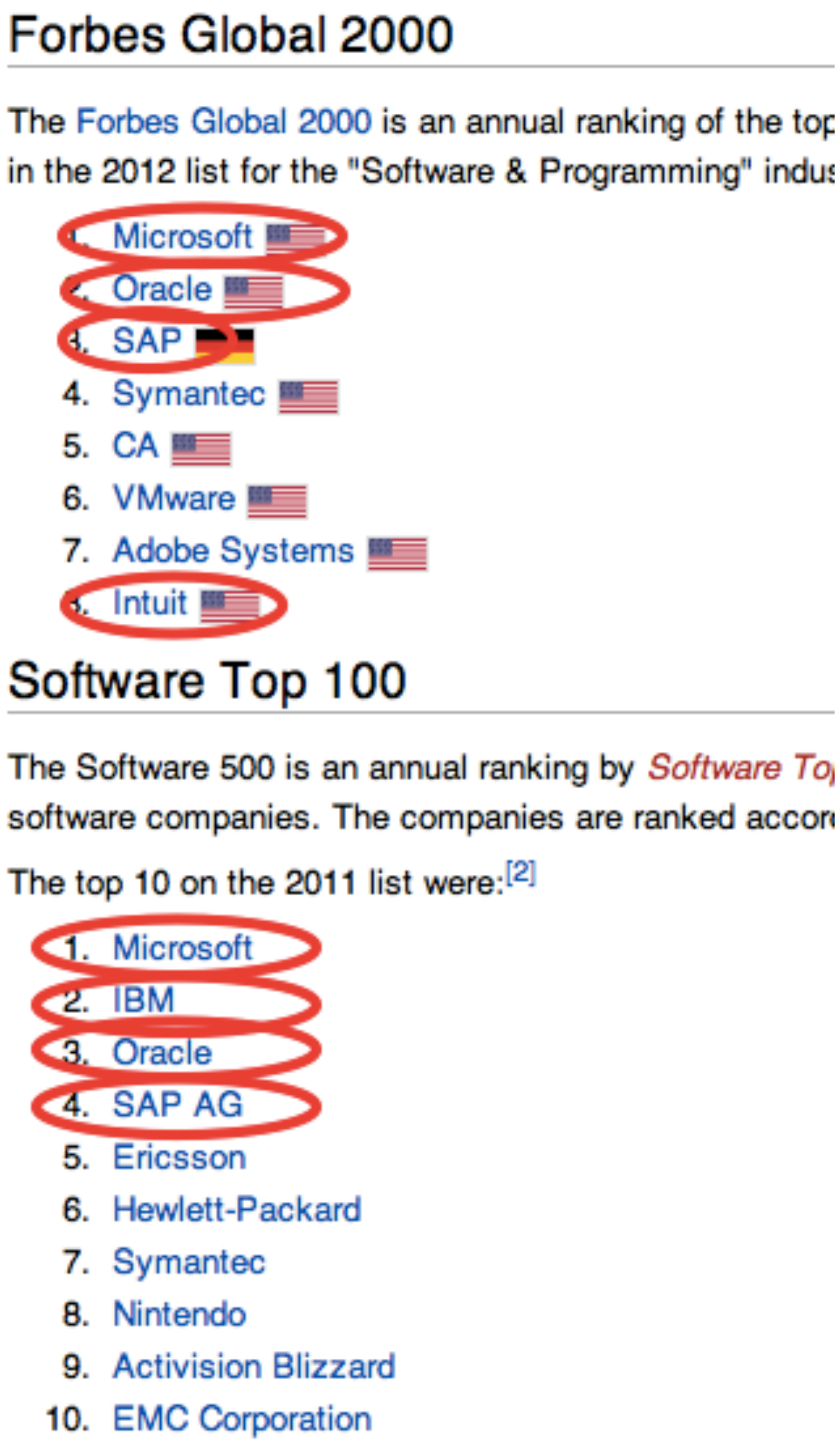
CSE 562: Database Systems

Instructor: Oliver Kennedy

okennedy@buffalo.edu



Forbes Global 2000

The Forbes Global 2000 is an annual ranking of the top
in the 2012 list for the "Software & Programming" indus

1. Microsoft
2. Oracle
3. SAP
4. Symantec
5. CA
6. VMware
7. Adobe Systems
8. Intuit

Software Top 100

The Software 500 is an annual ranking by *Software To*
software companies. The companies are ranked accor

The top 10 on the 2011 list were:[2]

1. Microsoft
2. IBM
3. Oracle
4. SAP AG
5. Ericsson
6. Hewlett-Packard
7. Symantec
8. Nintendo
9. Activision Blizzard
10. EMC Corporation

image credit: Wikipedia

Why Study Databases?

- Datasets are continually increasing in diversity and volume.
    - Digital Libraries, Skyserver, Twitter, Phone Sensors
- Information is one of the most valuable resources.
    - Database Management Systems are everywhere!
    - Search engines are ubiquitous
    - 100+ billion dollar-a-year industry (jobs!)
- Databases encompass much of CS
    - OS, Programming Languages, 'A'I, Logic, …

What is a Database?

(or a DBMS)

- A very large collection of (?curated?) data.
- A schema (or model) that indicates how the data is organized/can be used.
    - Entities (e.g., Starfleet Officers, Starships, Planets)
    - Relationships (e.g., Captain Kirk visited Vulcan)

image credit: Paramount Pictures

What is a Database?

(or a DBMS)

- A very large collection of (?curated?) data.
- A schema (or model) that indicates how the data is organized/can be used.
  - Entities (e.g., Starfleet Officers, Starships, Planets)
  - Relationships (e.g., Captain Kirk visited Vulcan)

image credit: Paramount Pictures

What is a Database?

(or a DBMS)

- A very large collection of (?curated?) data.
- A schema (or model) that indicates how the data is organized/can be used.
  - Entities (e.g., Starfleet Officers, Starships, Planets)
  - Relationships (e.g., Captain Kirk visited Vulcan)

image credit: Paramount Pictures

What is a Database?

(or a DBMS)

- A very large collection of (?curated?) data.
- A schema (or model) that indicates how the data is organized/can be used.
  - Entities (e.g., Starfleet Officers, Starships, Planets)
  - Relationships (e.g., Captain Kirk visited Vulcan)

A Database Management System (DBMS) is a software package designed to store and manage databases.

image credit: Paramount Pictures

Why use DBMSes?

- Rapid Application Development (Queries)
- Data Independence
- Concurrent Access
- Crash Recovery
- Ease of Management

Questions so far?

General Information

Instructor

Oliver Kennedy (*okennedy@buffalo.edu*)

TAs

Jie Hu (*jhu6@buffalo.edu*)
Tong Guan (*tongguan@buffalo.edu*)

General Information

Course Website

http://piazza.com/buffalo/spring2013/cse562/home

We will be monitoring the forums closely

(also has a study/project group coordination tool)

General Information

Course Website

http://piazza.com/buffalo/spring2013/cse562/home

We will be monitoring the forums closely

(also has a study/project group coordination tool)

General Information

Instructor Office Hours

Davis 338H

Monday/Thursday, 2:00-3:00

(or by appointment)

See website for more information.

Course Structure

- 2-3 Person Group Programming Assignment (40%)
  - **Part 1:** Orientation, Parser, Query Evaluation
  - **Part 2:** Index Construction
  - **Part 3:** Query Optimization
  - **Part 4:** Group's Choice Assignment
- Midterm Exam (25%)
- Comprehensive Final Exam (25%)
- ~Weekly Homework Assignments (10%)
  - Homework Grade is Avg, Dropping Lowest

Textbook

- Database Management Systems (3rd Ed)
  - By Ramakrishnan & Gehrke
- Required Textbook
- Syllabus Doesn’t Follow Textbook Exactly

Database Management Systems
Third Edition
NEW material on Database Applications
Ramakrishnan · Gehrke

Syllabus Overview

- SQL and Relational Algebra (~2 weeks)
- Basic Query Evaluation (~1 week)
- Storage and Indexing (~2.5 weeks)
- Query Optimization (~1.5 weeks)
- Data Modeling/Integrity Constraints (~1 week)
- Transactions/Concurrency (~2 weeks)
- Advanced Topics (~4 weeks)

Homework/Project Policies

- **Do discuss** the **concepts** that appear in homeworks/projects.
- **Do not submit** answers/solutions/code created by anyone other than you or your group.
- Read the Departmental Academic Integrity Policy
- All assignments should be submitted through the submission system on the departmental servers.
  - Details will be posted on the course website.

Homework/Project Policies

- No Late Submissions!
  - Late homework submissions will receive a 0% grade.
  - Your lowest homework grade will not count.
  - Late project submissions will be penalized by 10% of project grade per day late.
  - **No exceptions without medical documentation!**
- Project/Homework grades will be emailed to your UB address.
- Regrade requests must be received within 7 days of grades being distributed.

Exams

- Midterm (25%): **March 4, 1:00-1:50**
  - Closed Book Exam
- Final (25%): **On HUB**
  - Closed Book Exam
  - Cumulative with Emphasis on 2nd Half
- Please avoid scheduling conflicts
  - Poll on Piazza: Who is in Algorithms?

Workload?

- Relatively High
- So why should I take this course?
  - Big Ideas
  - Theory Meets Practice
  - Lots and lots of real-world applications
  - The Job Market!

Questions on the Class Structure?

What does a DBMS do?

(slides adapted from content by J.Gehrke, J.Shanmugasundaram, and/or C.Koch)

What does a DBMS add?
(over raw files)

- Schema Information/Data Independence
- IO/Memory Hierarchy Exploitation
- Consistency Guarantees
- Crash Recovery
- Security and Access Control

Data Independence

“officers” contains [First Name, Last Name, Starfleet ID]
Output the Starfleet ID of every officer named “Kirk”

```
File.open(“officers”).
  readlines.
  each do |line|
    line = line.split(/, */)
    if (line[1] == “Kirk”)
      then puts line[2]
    end
  end
```

Ruby

```
SELECT starfleet_id
FROM   officers
WHERE  last_name = “Kirk”
```

SQL

Data Independence

“officers” contains [First Name, Last Name, Starfleet ID]
Output the Starfleet ID of every officer named “Kirk”

Add [Middle Name]

```
File.open(“officers”).
  readlines.
  each do |line|
    line = line.split(/, */)
    if (line[1] == “Kirk”)
      then puts line[2]
    end
  end
```

```
SELECT starfleet_id
FROM   officers
WHERE  last_name = “Kirk”
```

No Change!

Logical Independence

Ruby

SQL

Data Independence

“officers” contains [First Name, Last Name, Starfleet ID]
Output the Starfleet ID of every officer named “Kirk”

```
File.open(“officers”).
  readlines.
  each do |line|
    line = line.split(/, */)
    if (line[1] == “Kirk”)
      then puts line[2]
    end
  end
```

Ruby

```
SELECT starfleet_id
FROM   officers
WHERE  last_name = “Kirk”
```

SQL

Data Independence

“officers” contains [First Name, Last Name, Starfleet ID]

Output the Starfleet ID of every officer named “Kirk”

Store sorted by Last Name

```
File.open(“officers”).
  readlines.  .skip_to(/[^,]*, *Kirk”/)
  each do |line|
    line = line.split(/, */)
    if (line[1] == “Kirk”)
      then puts line[2]
    end   else break
  end
```

```
SELECT starfleet_id
FROM   officers
WHERE  last_name = “Kirk”
```

No Change!

Physical Independence

Ruby

SQL

The Memory Hierarchy

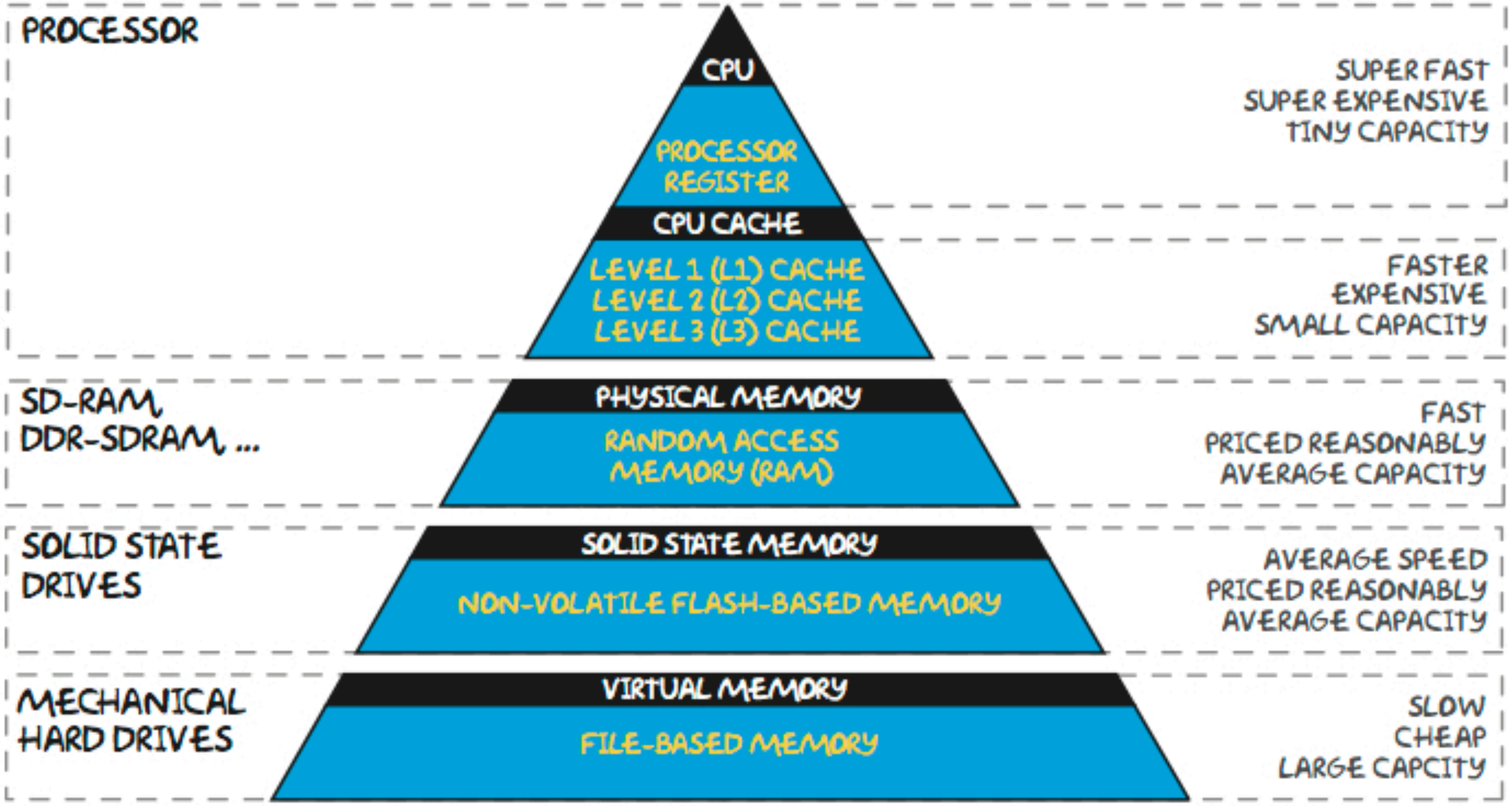


image credit: teachbook

Consistency

- Concurrent query execution is good!
  - Disk-IO has high-latencies.
  - Keep the processor busy by working on many things at once!
- Concurrent query execution is bad!
  - Operations can interact poorly.

Consistency

image credit: Paramount Pictures

Consistency

Fire a Photon Torpedo

image credit: Paramount Pictures

Consistency

Fire a Photon Torpedo

Value ‘torpedos’

```
torpedos = read(inventory);

torpedos = torpedos - 1;

write(inventory);
```

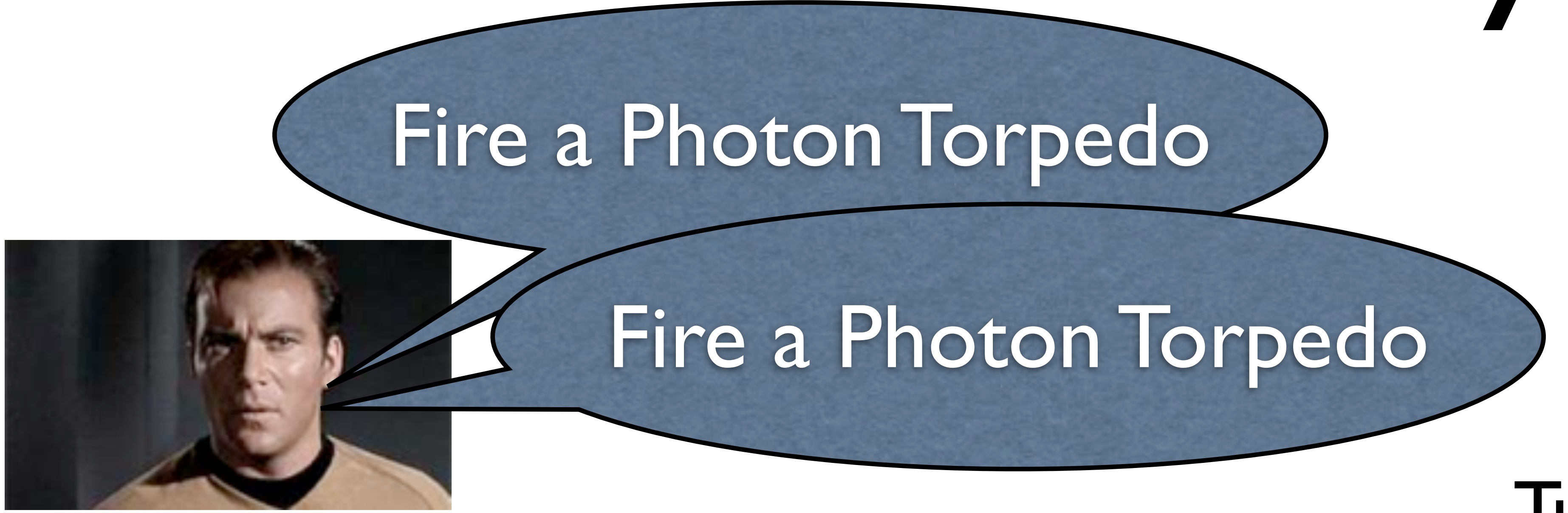
image credit: Paramount Pictures

Consistency

	Value	‘torpedos’
`torpedos = read(inventory);`	120	120
`torpedos = torpedos - 1;`	120	119
`write(inventory);`	119	119

image credit: Paramount Pictures

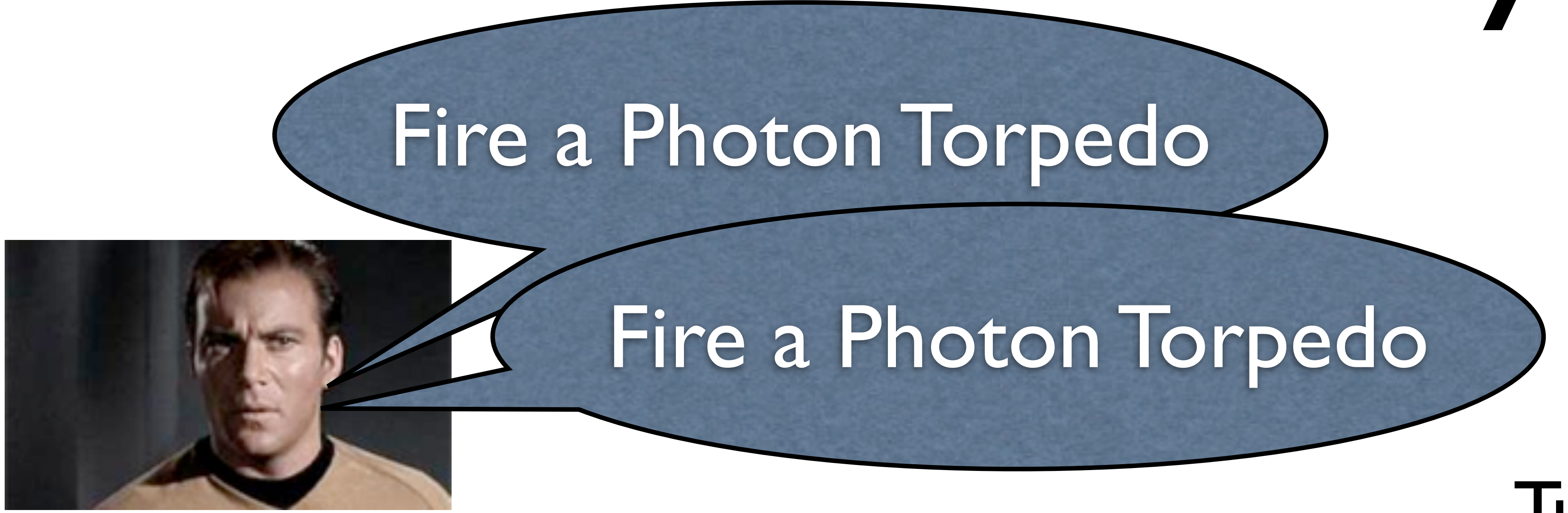
Consistency



	Value	Tube 1 'torpedos'	Tube 2 'torpedos'
`torpedos = read(inventory);`	120		
`torpedos = torpedos - 1;`			
`write(inventory);`			

image credit: Paramount Pictures

Consistency



	Value	Tube 1 ‘torpedos’	Tube 2 ‘torpedos’
`torpedos = read(inventory);`	120	120	120
`torpedos = torpedos - 1;`			
`write(inventory);`			

image credit: Paramount Pictures

Consistency

Fire a Photon Torpedo

Fire a Photon Torpedo

	Value	Tube 1 ‘torpedos’	Tube 2 ‘torpedos’
`torpedos = read(inventory);`	120	120	120
`torpedos = torpedos - 1;`	120	119	119
`write(inventory);`			

image credit: Paramount Pictures

Consistency

Fire a Photon Torpedo

Fire a Photon Torpedo

	Value	Tube 1 ‘torpedos’	Tube 2 ‘torpedos’
`torpedos = read(inventory);`	120	120	120
`torpedos = torpedos - 1;`	120	119	119
`write(inventory);`	119!!!	119	119

Consistency Assured by the DBMS!

image credit: Paramount Pictures

Transactions

- An **atomic** sequence of database actions (reads/writes).
- All operations **succeed or fail** together.
- The database is left in a **consistent state**.
    - Users can specify **integrity constraints** on the data to be enforced by the DBMS. (e.g., A ship has only one captain, each SID is unique)
    - The DBMS makes no other assumptions about the data semantics.

Atomicity/Crash Recovery

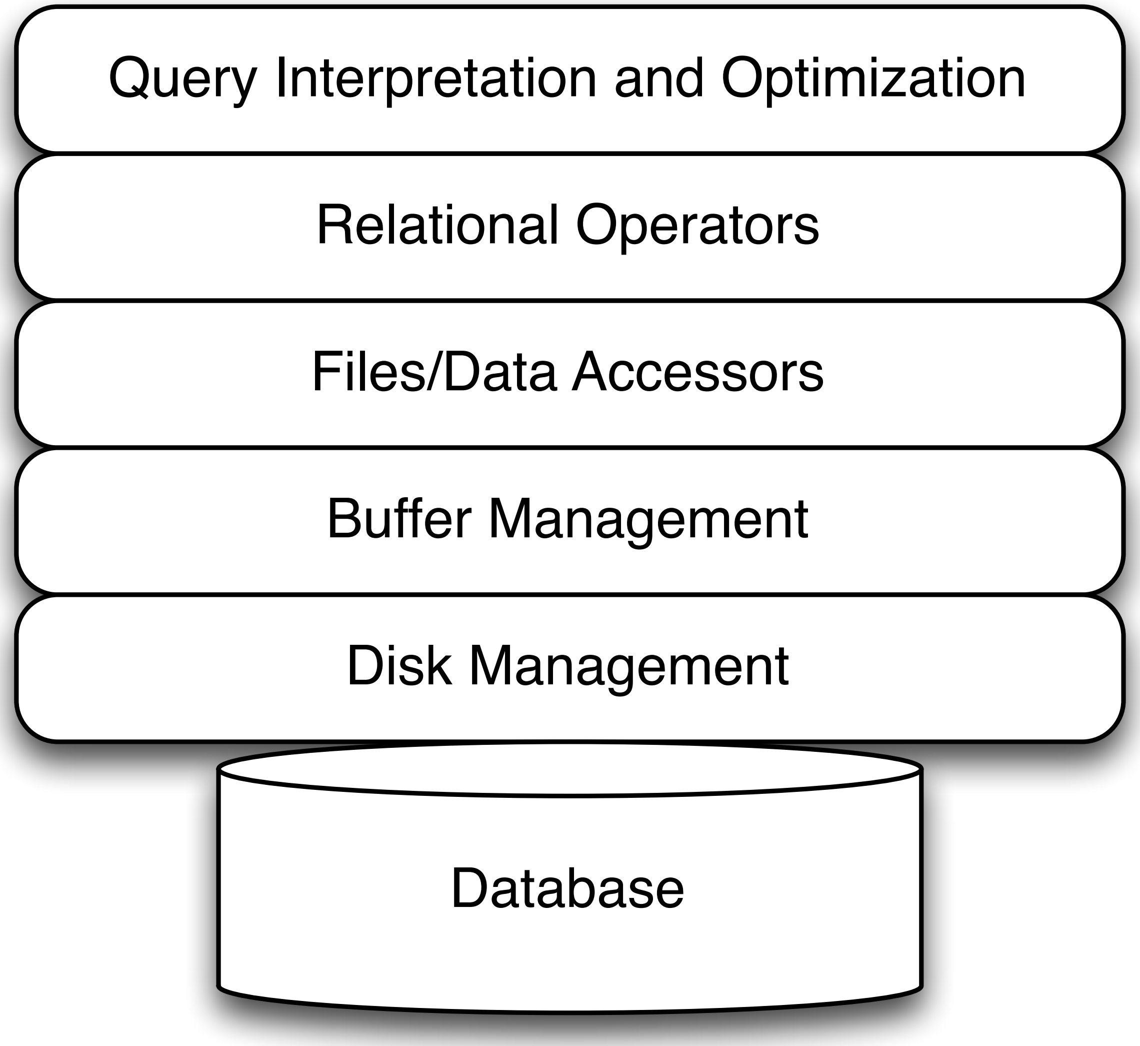
- DBMS ensures atomicity even if the system crashes!
    - Keep a record (log) of all actions performed.
    - Before actually changing the database, ensure that the log record is safely written to disk.
    - After a crash, use the log to undo incomplete transactions.
    - If the log record wasn't written, we haven't modified the database either!

Atomicity/Crash Recovery

- DBMS ensures atomicity even if the system crashes!
  - Keep a record (log) of all actions performed.
  - Before actually changing the database, ensure that the log record is safely written to disk.
  - After a crash, use the log to undo incomplete transactions.
  - If the log record wasn't written, we haven't modified the database either!

Crash Recovery Automagically Provided by DBMS!

Structure of a DBMS
(example)



Who needs to know about DBMSes?

- DBMS Vendors/Developers
- Database Users/Application Programmers
  - e.g., Webmasters
- Database Administrators (DBAs)
  - … design logical/physical schemas.
  - … handle security/authorization.
  - … tune the database as workloads change.

Overview of the Overview

- DBMSes…
  - … maintain and query large datasets.
  - … automagically provide …
    - … concurrency support.
    - … crash recovery.
    - … access control.
- Levels of abstraction give data independence.
- A DBMS typically has a layered architecture.
- Database principles are crucial.
  - …even for NoSQL systems

Overview of the Overview

- DBMSes…
  - … maintain and query large datasets.
  - … automagically provide …
    - … concurrency support.
    - … crash recovery.
    - … access control.
- Levels of abstraction give data independence.
- A DBMS typically has a layered architecture.
- Database principles are crucial.
  - …even for NoSQL systems

Questions?