CMPT 454

Introduction

Introduction

- Administration
- CMPT 454 Topics

Administration

1.1

Course Management

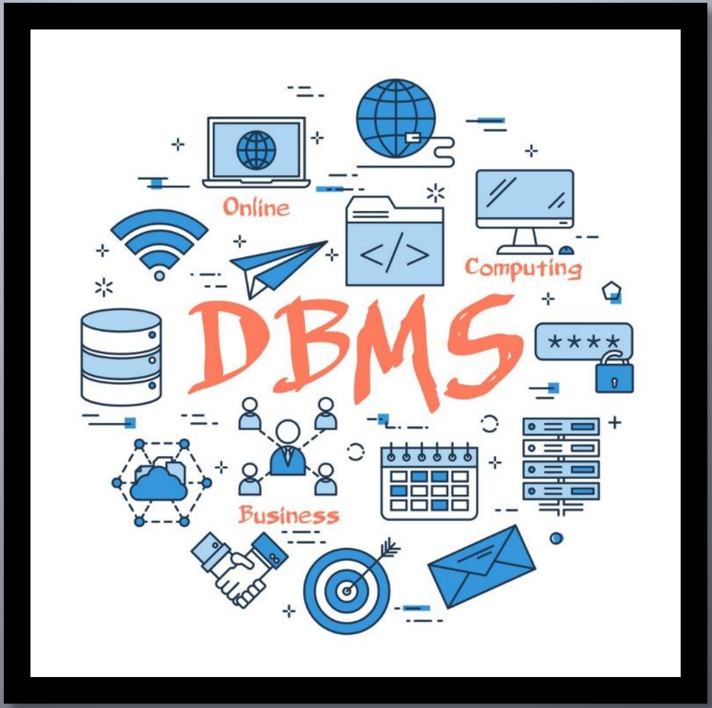
- Website
  - http://www.cs.sfu.ca/CourseCentral/454/johnwill/
- Marks are posted on coursys

Assessment

- Assignments – 25%
- Midterm exams in class – 25%
- Final exam – 50%

Database Management Systems

1.2



Why Use a DBMS?

- To store data ...
  - Data model is complex
  - Data set is large
- To handle other issues
  - Concurrency
  - Recovery
  - Security
    - Not covered in CMPT 454

A DBMS has a defined interface with the application – e.g. SQL

Alternative: application is responsible

Major Topic List

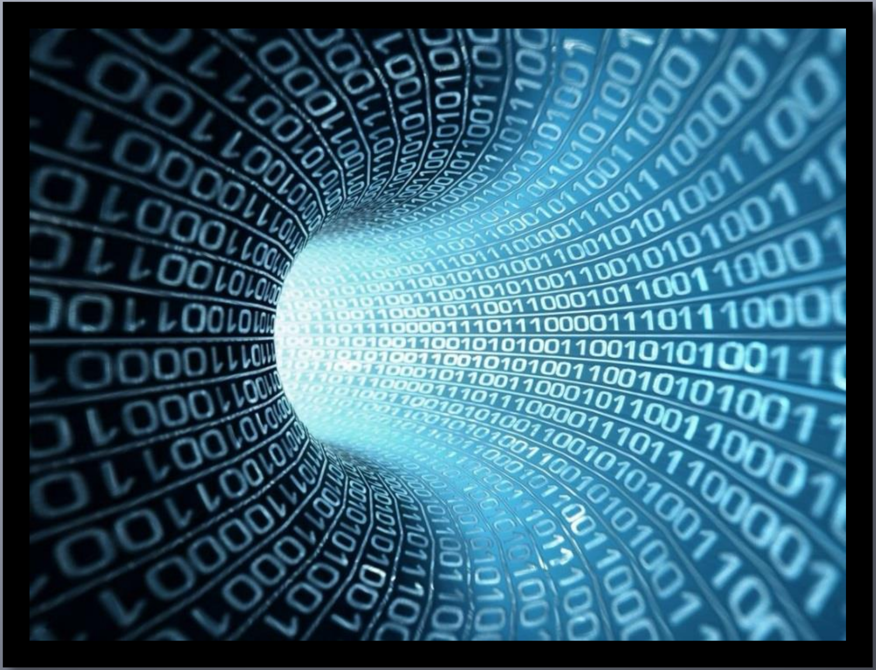
- Storage management and hardware
- Indexing
- Query optimization
  - And external sorting
- Transactions and concurrency
- Logging and recovery

Which DBMS?

- There are different types of DBMS products
  - RDBMS
  - Non-SQL
- Cost varies
  - Free
  - Expensive
- It is important to select the product that is right for the organization or application
  - Though this is not a topic of this course

Data Storage and Access

1.3



Data Storage and Access

- Data storage
  - Hardware and data organization
  - Supporting efficient access
    - Index structures
- Efficiently accessing data
  - Query optimization

SQL Queries

- SQL query operations
  - Selections
  - Projections
  - Joins
  - Set operations
  - Aggregations
  - ...
- There is often more than one algorithm

You should be familiar with SQL from CMPT 354

If not – please review!

Single Table Queries

- Queries can either be on one or more tables
- A query on a single table only requires that one table to be accessed
  - Such a query can still be time consuming
    - If the table is large
- Simplest method
  - Read all records in the table
  - Returning those that match
  - Better – use an index

```
SELECT sin, first, last
FROM Customer
WHERE job = 'journalist'
```

Data organization – how do we find a record's attributes?

Multiple Table Queries

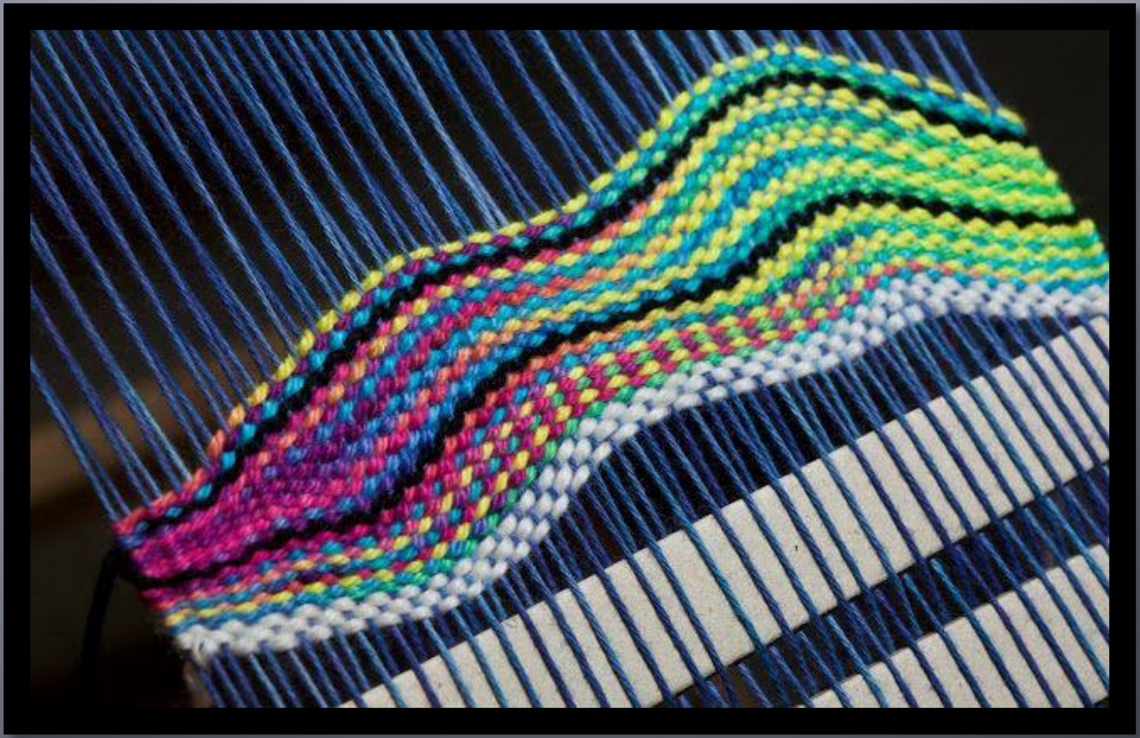
- Multiple table queries generally entail joins
  - Or some similar operation
- Joins can be expensive
  - Naïve algorithm is $O(n^2)$
  - There are a number of join algorithms
    - Each with advantages and disadvantages

```
SELECT balance
FROM Customer C, Account A
WHERE C.name = 'Jones' AND
      C.id = A.id
```

The DBMS has to determine which join algorithm to use in any given query

Transactions

1.4



Example Transactions

- A transaction is a single logical unit of work
  - Who is the owner of the largest account?
  - Which students have a GPA less than 2.0?
  - Transfer $200 from Bob to Kate
  - Add 5% interest to all accounts
  - Enroll student 123451234 in CMPT 454
  - ...
- Many transactions entail multiple actions

Transfer $200 from Bob to Kate

- Transferring $200 from one bank account to another is a single transaction
  - With multiple actions

Action	Bob	Kate
Read Bob's balance	347	
Read Kate's balance		191
Subtract $200 from Bob's balance (147)		
Add $200 to Kate's balance (391)		
Write Bob's new balance	147	
Write Kate's new balance		391

Concurrency

- A typical OLTP[1] database is expected to be accessed by multiple users concurrently
  - Consider the Student Information System

... without crying please ...

- Concurrency increases throughput[2]
  - Actions of different transactions may be *interleaved* rather than processing each transaction in series
- Interleaving transactions may leave the database in an inconsistent state
- 1 – Online Transaction Processing
- 2 – Throughput is a measure of the number of transactions processed over time

Concurrency Error Example

- T1 – Transfer $200 from Bob to Kate
- T2 – Deposit $7,231 in Bob's Account

Bob is probably not happy

time

Action	Bob	Kate
T1 – Read Bob's balance	347	
T2 – Read Bob's balance	347	
T1 – Read Kate's balance		191
T2 – Add $7,231 to Bob's balance (7,578)		
T1 – Subtract $200 from Bob's balance (147)		
T2 – Write Bob's new balance	7,578	
T1 – Add $200 to Kate's balance (391)		
T1 – Write Bob's new balance	147	
T1 – Write Kate's new balance		391

This transaction schedule should be prevented

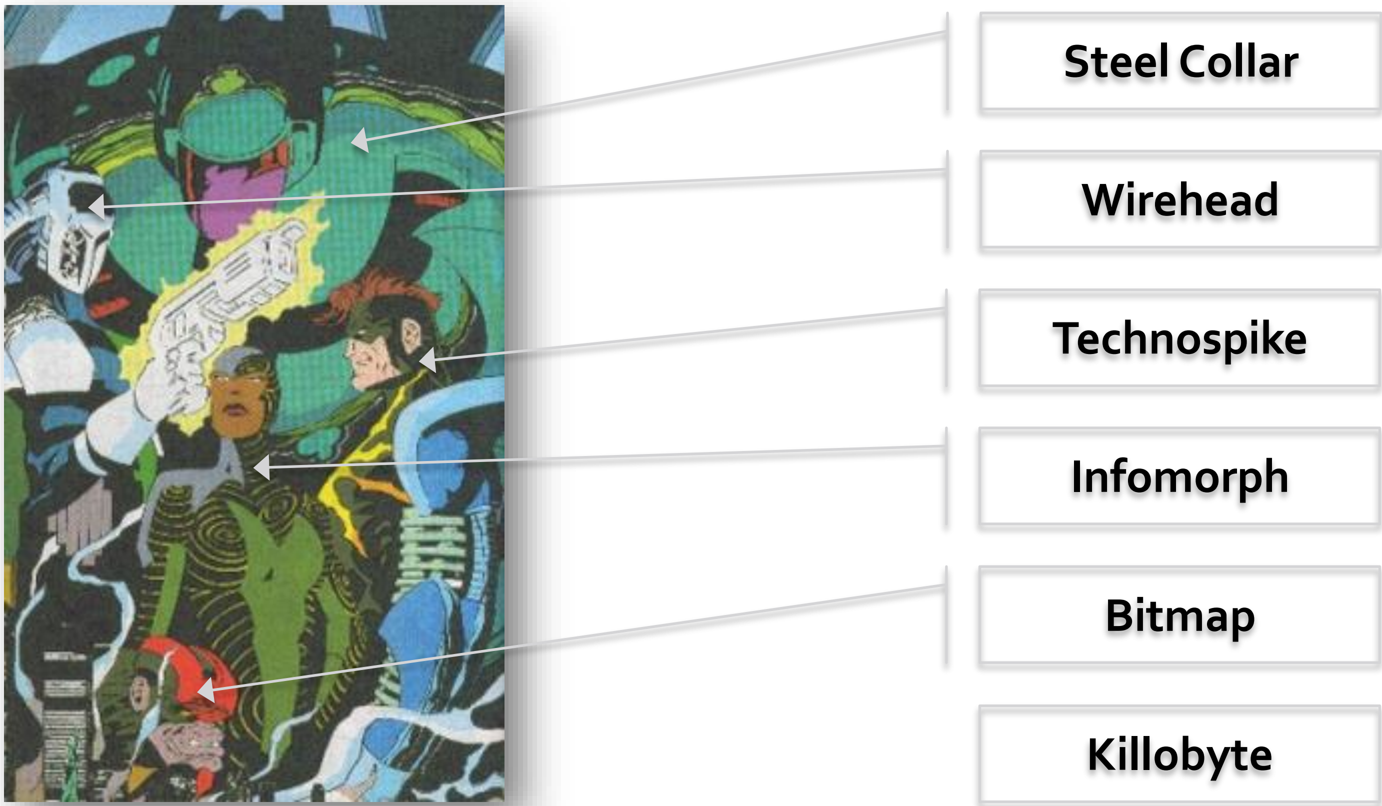
ACID Transactions

- Transactions should maintain the ACID properties
  - Atomic
  - Consistent
  - Isolated
  - Durable

Recovery

1.5

System Crash!



Daredevil #326, 1994

System Crash

- What happens in the event of a system failure?
  - The database must recover
  - And must be guaranteed to be in a consistent state after recovery
- Processing is performed in main memory
  - A transaction completed in main memory is lost if it has not been written to disc
  - But should be restored on recovery

Partial Transactions

- Transactions are often composed of multiple actions
  - Some of a transactions' actions may have been written to disc
    - Before the transaction is complete
    - Leaving the database inconsistent if the system fails during the processing of the transaction
  - Such partial transactions must be rolled back

DBMS Architecture

1.6

Lego World, San Diego 2006

DBMS Components