cs.uw-.ca/ngweddell/cs 348

4 assignments

20% ↑ 30% MT 50% Final

- factual data vs. metadata

Volatile vs. persistent data

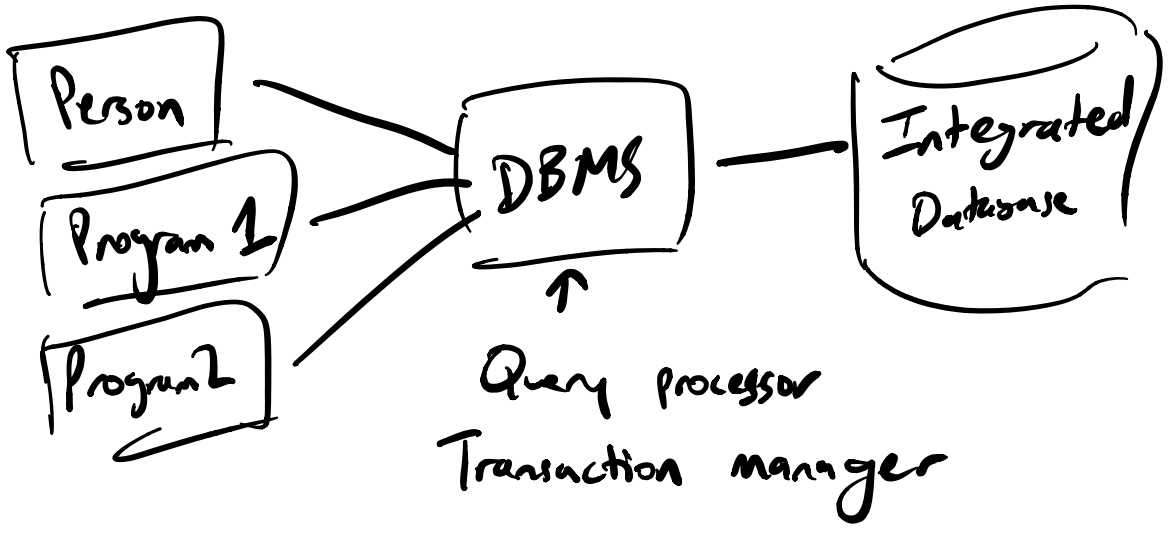
early data management

- data stored on magnetic tapes (large batch jobs)
- 1 data set / program. High data redundancy

then: file systems. File processing

- procedural interface
- supports chaining of programs that use the same initial data
- various access methods: sequential, indexed, random

Database Approach:



Definition (Database): Large persistent collection of data organized for efficient retrieval and revision.

e.g. factual data: John's age = 42

e.g. metadata: concept of employee w/ name & age

Definition (Data Model): determines nature of the metadata and how retrieval & revision is expressed

e.g. database → File cabinet
→ Library system
→ Inventory control system (Can expect perfect answers)

Definition (Database Management System — DBMS): One or more programs that implements a data model

We will focus on the Relational data model.

Which free RDBMS to choose?

→ MySQL → PostgreSQL

Idea of DBMS: Abstract common fens and give uniform well-defined intf for applications requiring a DB

1) Supports a data model
2) Access Control (Security concerns)
(authorized people)

3) Concurrency control
4) Database recovery (Reliability, nothing gets lost)
5) Database maintenance (e.g. revising metadata)

Concurrency control & Reliability are 2 of the big issues. Both strongly complicate DBMS implementation, and both make life easier for application developer.

Definition (Schema): Collection of metadata conforming to an underlying data model

Definition (Instance): Collection of factual data as defined by a given DB schema

A schema can (and does) have many possible instances

50,000 tables in SAP's releases of enterprise management software

Hierarchical data model — IBM's Ims (tree)
↓
Network data models (IDS, CODASYL)
→ collections of sets of records

navigational JSON

Cassandra (NoSQL)

RDF
→ used to integrate data Lakes

By late 80s, Relational technologies were well developed.
1976: Peter Chen proposes E-R model

3 big ideas:

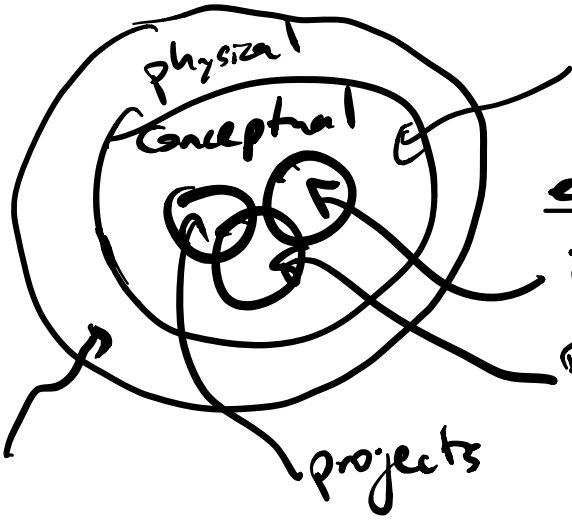
1) notion of a transaction

↳ consistency-preserving transformation of data
↳ abstracts concurrency concerns in domain-specific semantics

2) data independence

3 level schema architecture

1. external schema (view)
2. conceptual schema ← employees
3. physical schema ← tuples, Btrees



Warehouse, Paydequy, Part, Reduction

external schemas

inventory

payroll

projects

Metadata: emp, paychq, union, ...

table stored in Btree in file X on disk y

3) quantifications

(ETL, process bulk data and quantify over)

data independence: data accessed through abstraction

Physical: apps immune to changes in storage structures

Logical: Modularity
WAREHOUSE table not accessed by payroll app

One of the most important reasons to use a DBMS!

Atomicity
Consistency → Integrity Constraint
Isolated
Durable → changes are permanent

defn (transaction): An app-specified atomic and durable unit of work

Data Definition Language (DDL) for schemas

Data Manipulation Language (DML) for specifying requests

end-user: indirectly

application developers: access DB

DBA: manages schema, monitors/tunes DBMS perf, loads & reformats DB

engine devs