Gagnasafnsfræði

Kynning

Hallgrímur H. Gunnarsson

What is a DBMS?

A *Database Management System* is a software package to manage data

The term *database* usually implies that the data is managed to some extent

The most popular database model is the relational model (Codd 1969) and systems based on it are called Relational Database Management Systems (RDBMS).

Popular RDBMS include Oracle, PostgreSQL, MySQL, Microsoft SQL Server, IBM DB2, and sqlite.

Files vs. DBMS

Issues to consider:

1. Query language
2. Data independence
3. Efficient access
4. Concurrent access
5. Crash recovery
6. Transactions

Data model

A *data model* is a collection of concepts for describing data.

A *schema* is a description of a particular collection of data, using a data model.

The *relational model of data* is the most widely used model today.

1. Main concept: *relation*, basically a table with rows and columns
2. Every relation has a schema, which describes the columns, or fields

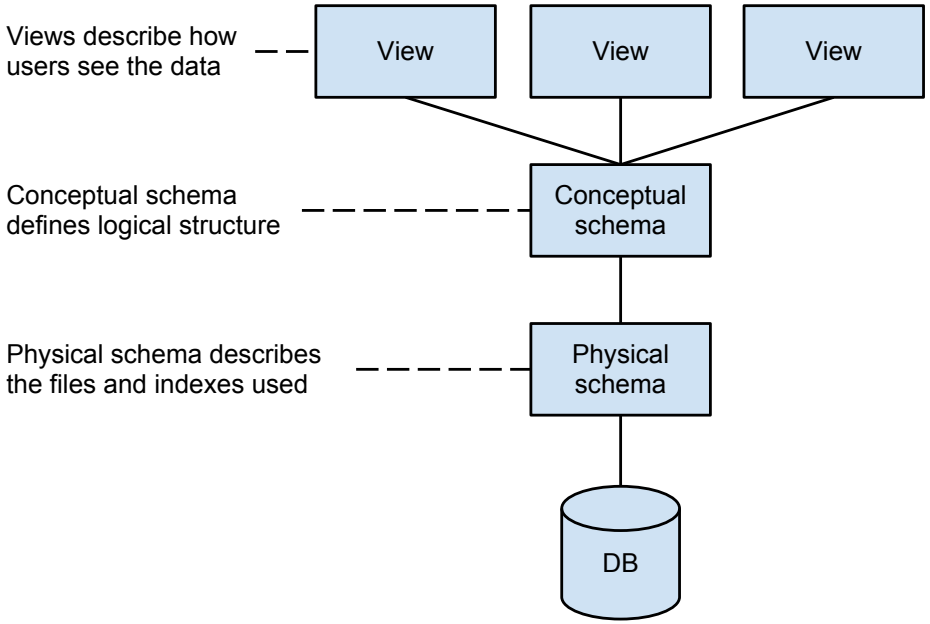
Big idea: Data Independence

> [The Relational Model] provides a basis for a high level data language which will yield *maximal independence* between programs on the one hand and machine representation on the other.
>
> (E.F. Codd, CACM 1970)

Raising the level of abstraction. We want to deal with problems at the level we are thinking about them, using the appropriate level of detail and abstraction, and in terms natural to the problems themselves.

Applications insulated from how data is structured and stored.

Levels of abstraction



Logical data independence: Protection from changes in the *logical* structure of data.

Physical data independence: Protection from changes in *physical* structure of data.

Example: University Database

Conceptual schema:

```
Students(sid: string, name: string, login: string)
Courses(cid: string, name: string, credits: integer)
Enrolled(sid: string, cid: string, grade: string)
```

Physical schema:

- Relation stored as unordered files
- Index on first column of Students

External schema:

```
Course_Info(cid: string, enrollment: integer)
```

Query language

SQL (Structured Query Language), de-facto standard for all relational database management systems

Dialects: SQL-86, SQL-89, SQL-92, SQL:1999, SQL:2003, SQL:2008

The original language, called SEQUEL (Structured English Query Language), was developed at IBM in the 70s.

SQL is a declarative language

Example SQL statement:

```
SELECT cid, COUNT(*) as enrollment FROM Enrolled GROUP BY cid;
```

Concurrent access

Concurrent execution of user programs is essential for good DBMS performance
- Because disk accesses are frequent, and extremely slow, it is important to keep the CPU busy by working on several user programs concurrently

Interleaving actions of different user programs can lead to inconsistency: e.g. race conditions

DBMS ensures such problems cannot arise: users can pretend they are using a single-user system

Transaction

Transaction is an *atomic* sequence of database actions (reads/writes)

Invariant: Each transaction, executed completely, must leave the DB in a *consistent state* if the DB is consistent when the transaction begins

Users can specify some simple *integrity constraints* on the data, and the DBMS will enforce these constraints.

Transaction properties (ACID): atomicity, consistency, isolation, durability

Atomic transactions

DBMS ensures *atomicity* (all-or-nothing property) even if the system crashes in the middle of a transaction

Basic idea: Keep a *log* (or journal) of all actions carried out by the DBMS while executing a transaction (old value, new value)

Before a change is made to the database, the corresponding log entry is forced to disk (WAL protocol). Depends on OS support.

After a crash, the effects of partially executed transactions are undone using the log. (Thanks to the WAL, if the log entry wasn't saved before the crash, then the corresponding change was not applied to the database)

Structure of a typical DBMS