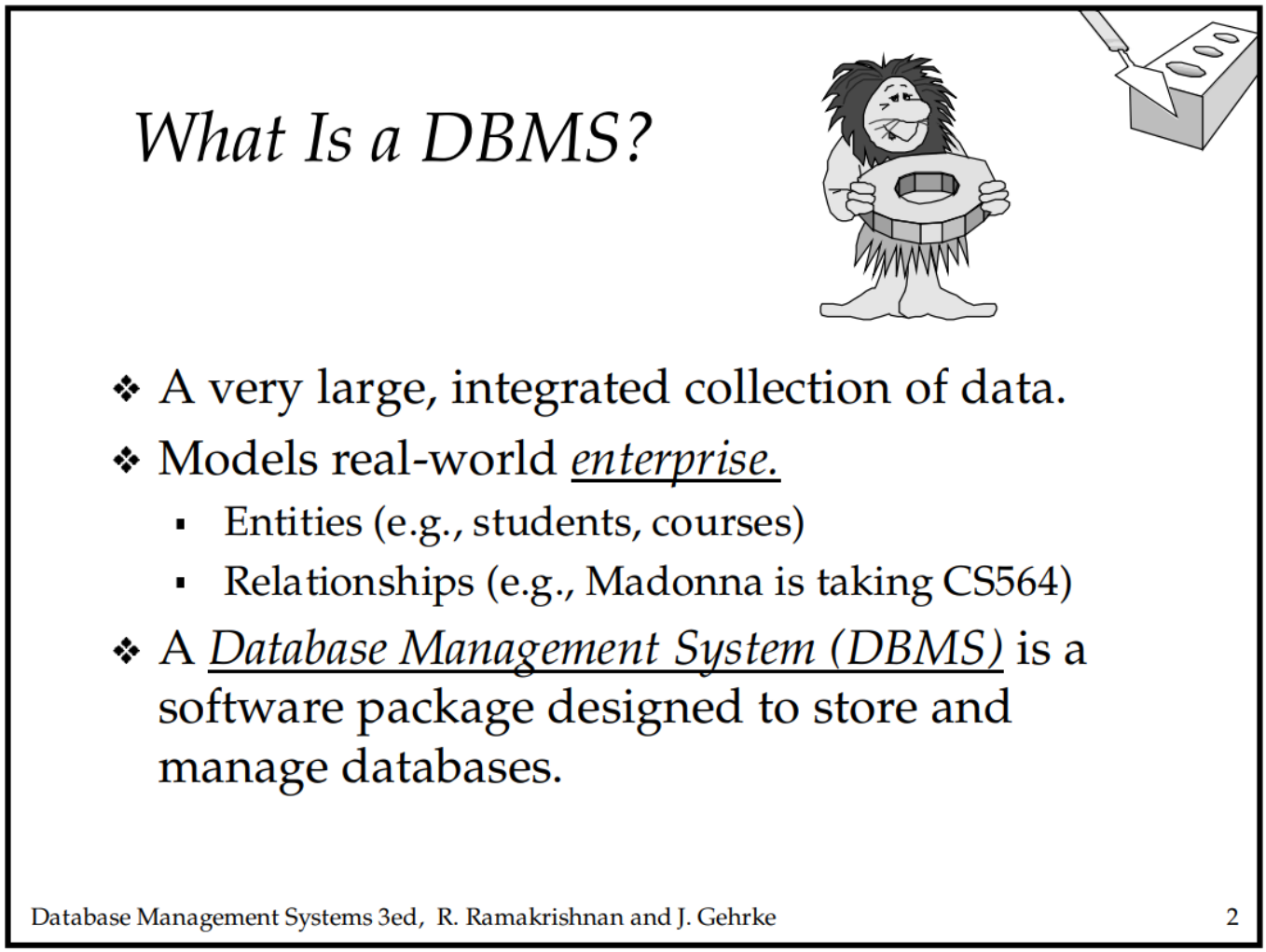
CH1

What is a DBMS, in particular, a relational DBMS?



Database
: a collection of data, typically describing the activities of one or more related organizations.

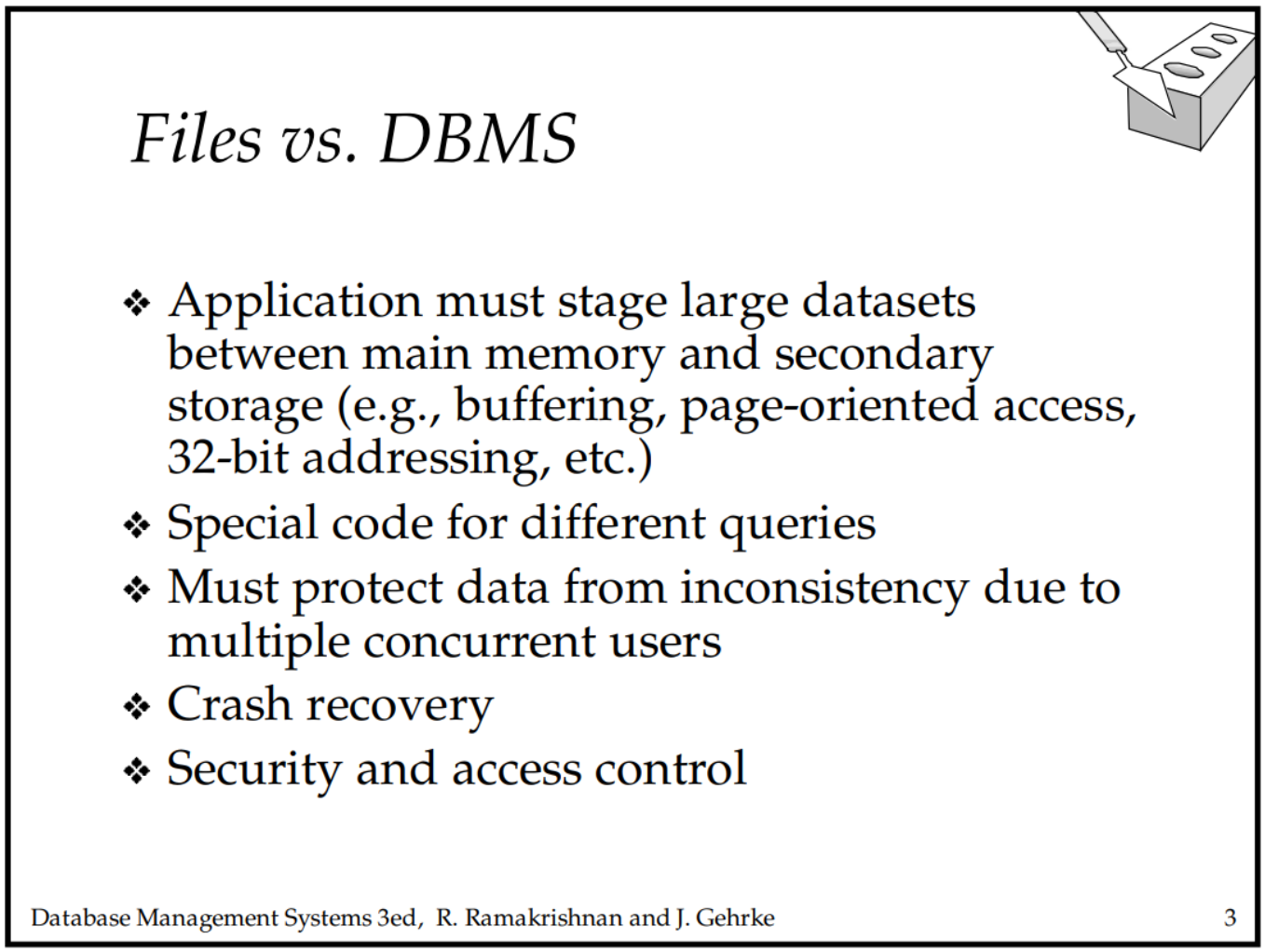
For example, a university database might contain information about the following:

- **Entities** such as students, faculty, courses, and classrooms.
- **Relationships** between entities, such as students' enrollment in courses, faculty teaching courses, and the use of rooms for courses.

Database management system, DBMS

a software designed to assist in maintaining and utilizing large collections of data

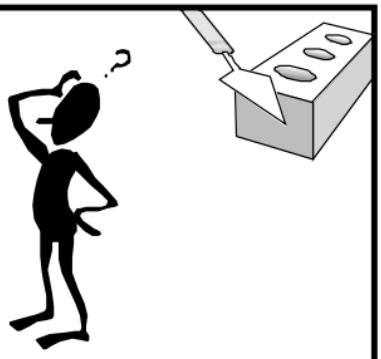
Why should we consider a DBMS to manage data?



!

How is application data represented in a DBMS?

Why Use a DBMS?

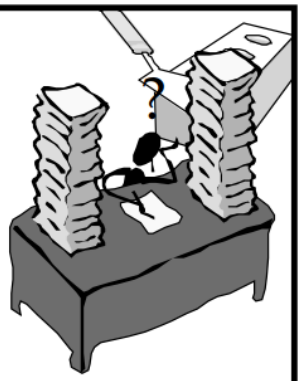


- Data independence and efficient access.
- Reduced application development time.
- Data integrity and security.
- Uniform data administration.
- Concurrent access, recovery from crashes.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 4

Why study Databases?

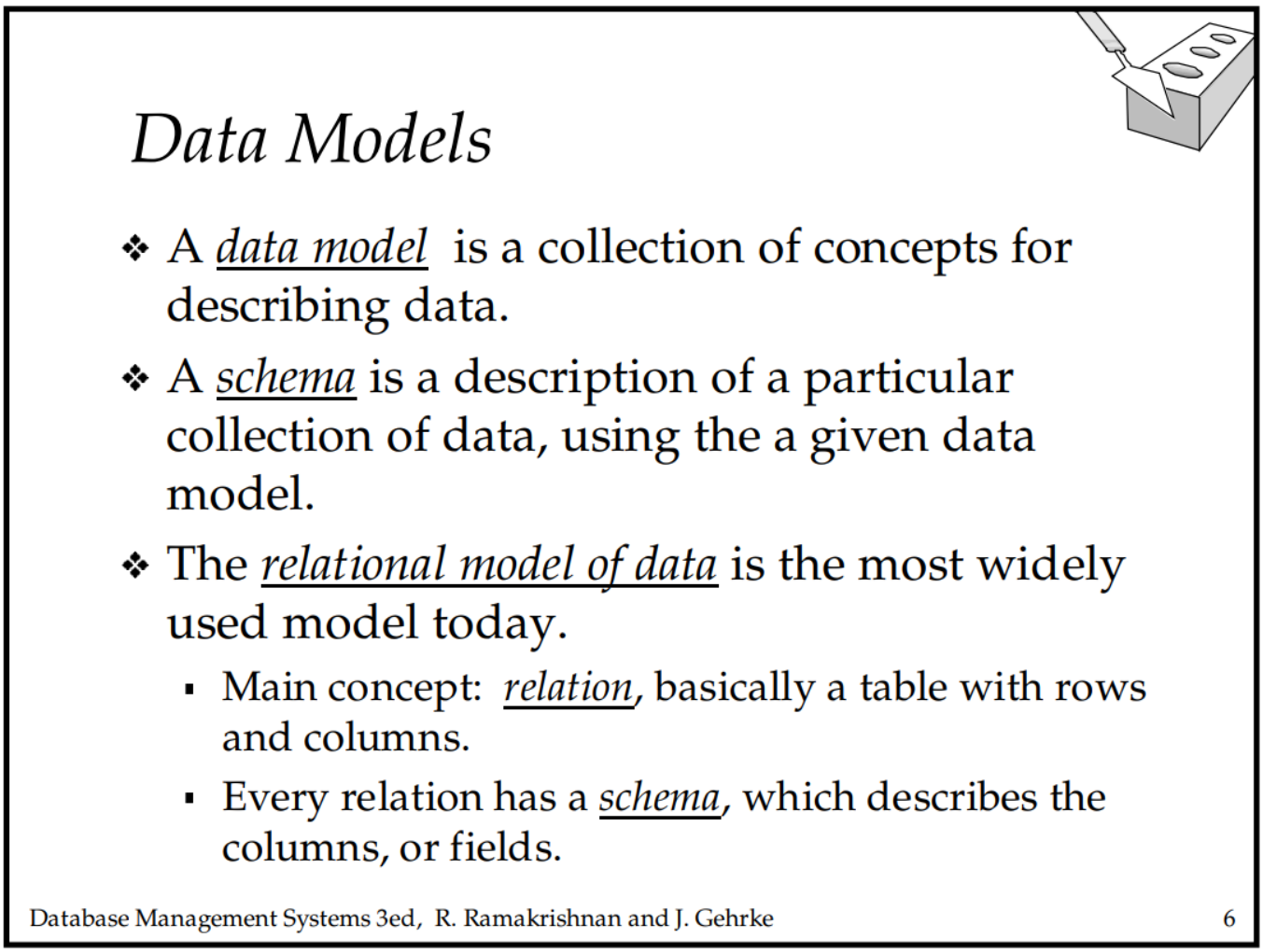
Why Study Databases??



- Shift from *computation* to *information*
  - at the "low end": scramble to webspace (a mess!)
  - at the "high end": scientific applications
- Datasets increasing in diversity and volume.
  - Digital libraries, interactive video, Human Genome project, EOS project
  - ... need for DBMS exploding
- DBMS encompasses most of CS
  - OS, languages, theory, "A"I, multimedia, logic

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 5

How is data in a DBMS retrieved and manipulated?



data model
: a collection of high-level data description constructs that hide many low-level storage details

schema
: A description of data in terms of a data model

In the relational model, the schema for a relation specifies its name, the name of each **field** (or **attribute** or **column**), and the type of each field.

As an example, student information in a university database may be stored in a relation with the following schema:

$$\text{Students}(\textit{sid: } \texttt{string}, \textit{ name: } \texttt{string}, \textit{ login: } \texttt{string}, \textit{ age: } \texttt{integer}, \textit{ gpa: } \texttt{real})$$

The preceding schema says that each record in the Students relation has five fields, with field names and types as indicated. An example instance of the Students relation appears in Figure 1.1.

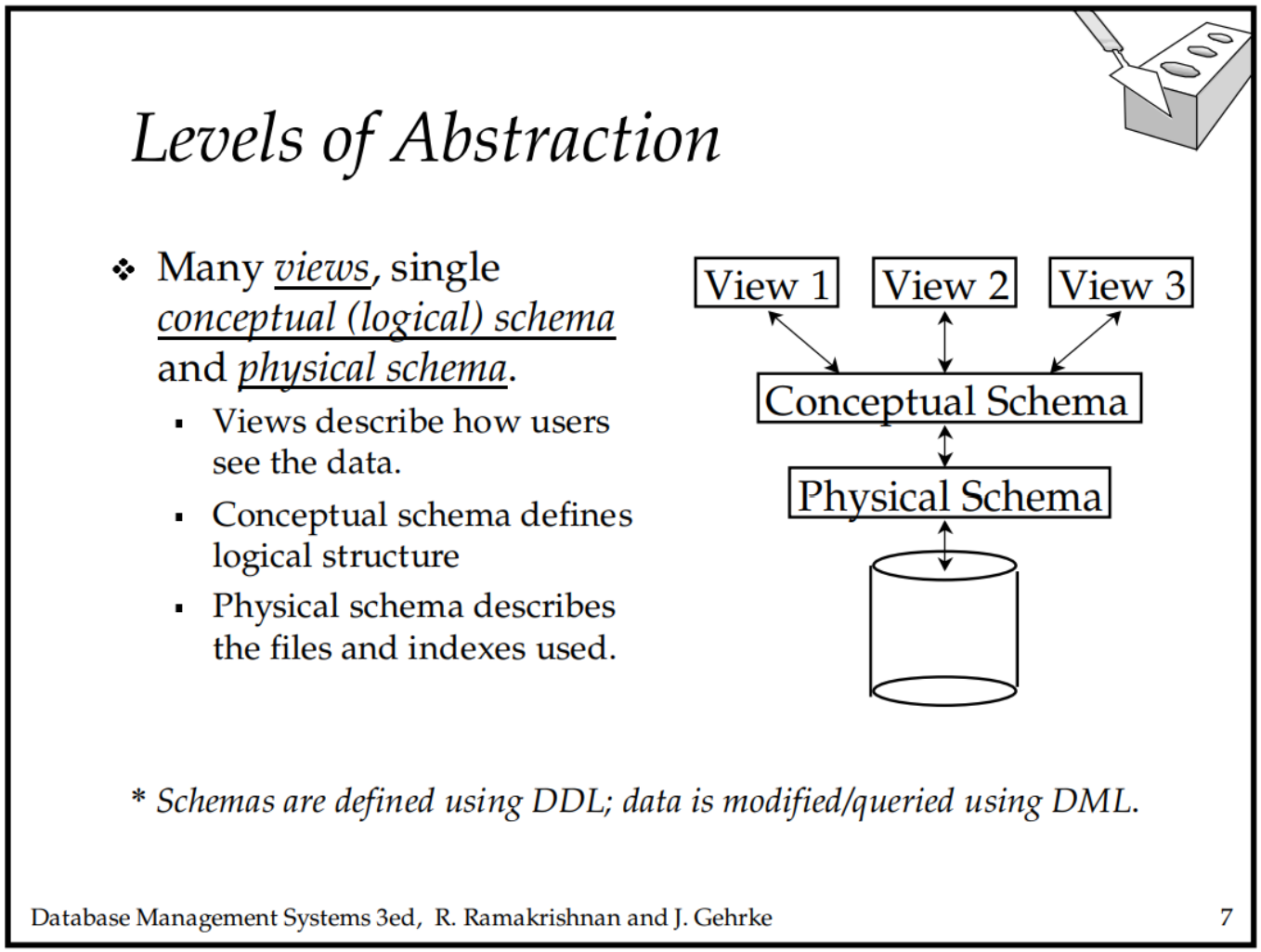
sid	*name*	*login*	*age*	*gpa*
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Figure 1.1 An Instance of the Students Relation

Every row follows the schema of the Students relation. The schema can therefore be regarded as a template for describing a student.

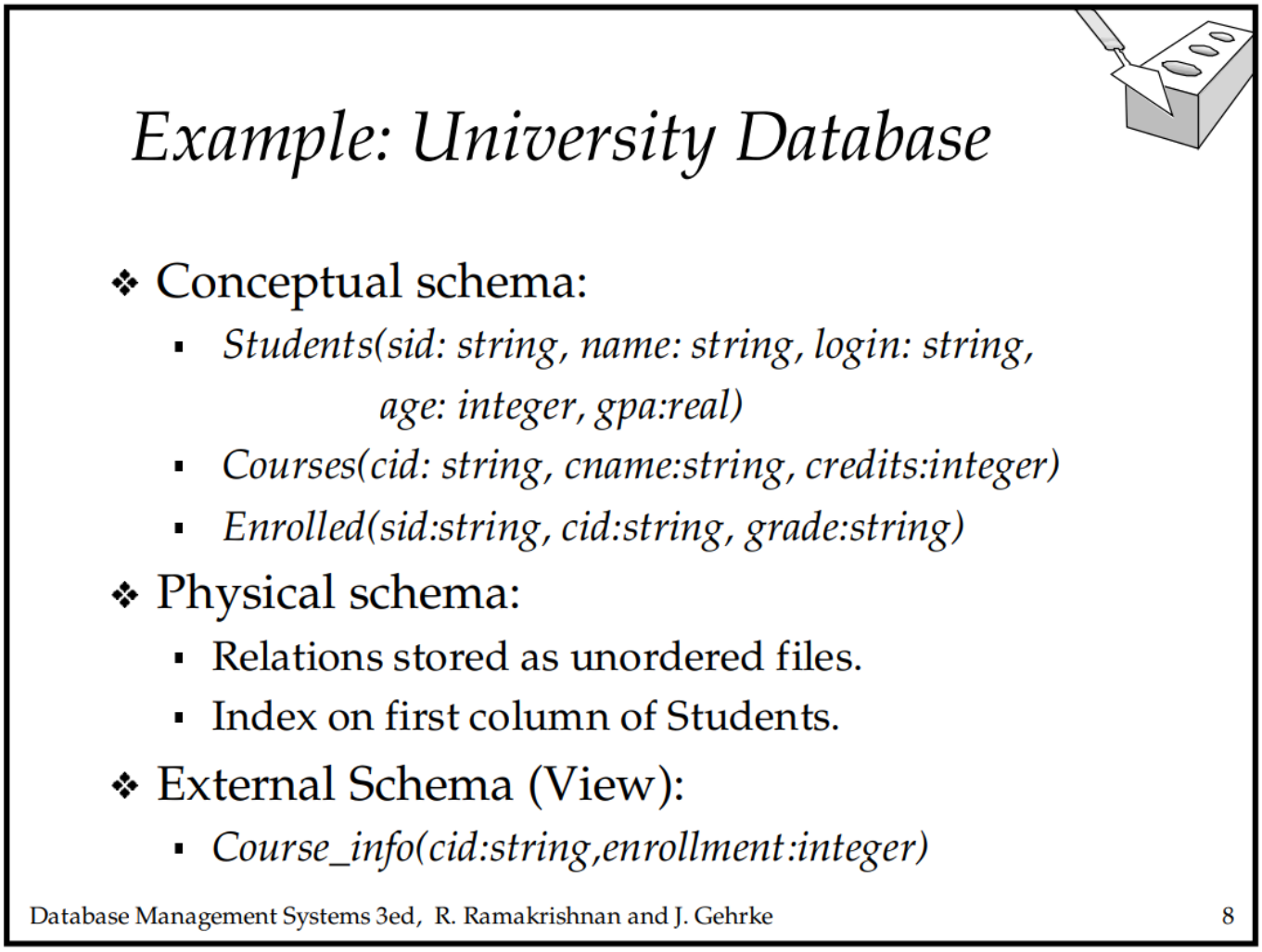
integrity constraints

conditions that the records in a relation must satisfy.



data definition language

is used to define the external(view) and conceptual schemas



physical schema

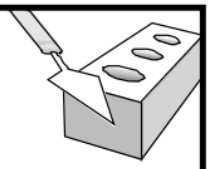
specifies additional storage details.

Essentially, the physical schema summarizes how the relations described in the conceptual schema are actually stored on secondary storage devices such as disks and tapes.

External schemas

which usually are also in terms of the data model of the DBMS, allow data access to be customized (and authorized) at the level of individual users or groups of users.

How does a DBMS support concurrent access and protect data during system failures?



*Data Independence **

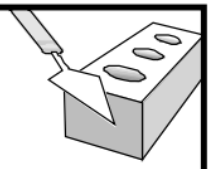
- Applications insulated from how data is structured and stored.
- *Logical data independence*: Protection from changes in *logical* structure of data.
- *Physical data independence*: Protection from changes in *physical* structure of data.

* *One of the most important benefits of using a DBMS!*

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 9

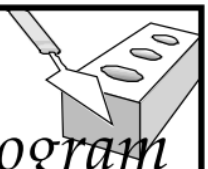
Data independence
: application programs are insulated from changes in the way the data is structured and stored

Transaction
: any one execution of a user program in a DBMS.



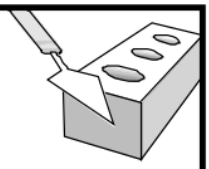
Concurrency Control

- Concurrent execution of user programs is essential for good DBMS performance.
  - Because disk accesses are frequent, and relatively slow, it is important to keep the cpu humming by working on several user programs concurrently.
- Interleaving actions of different user programs can lead to inconsistency: e.g., check is cleared while account balance is being computed.
- DBMS ensures such problems don't arise: users can pretend they are using a single-user system.



Transaction: An Execution of a DB Program

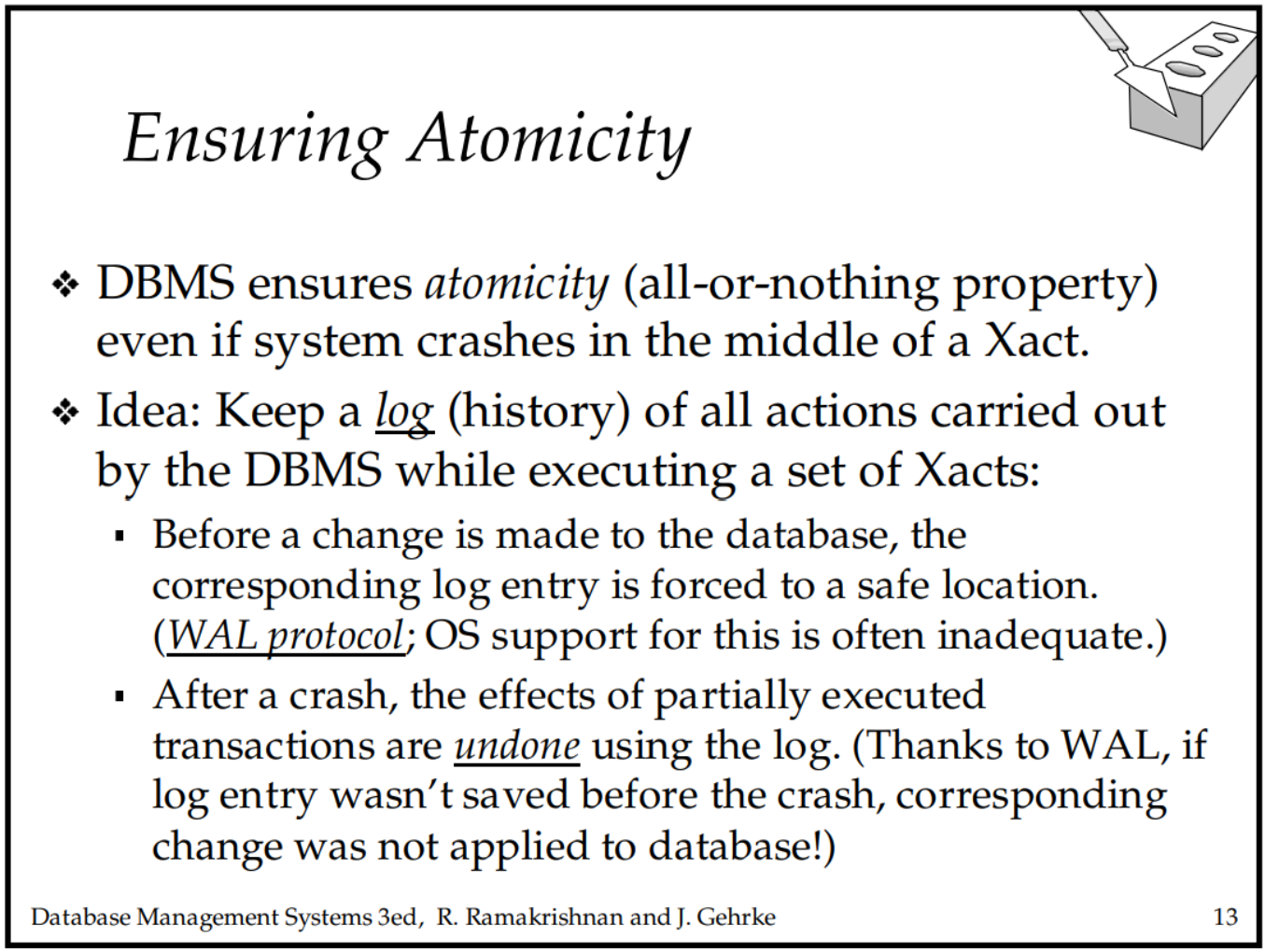
- Key concept is *transaction*, which is an *atomic* sequence of database actions (reads/writes).
- Each transaction, executed completely, must leave the DB in a *consistent state* if DB is consistent when the transaction begins.
  - Users can specify some simple *integrity constraints* on the data, and the DBMS will enforce these constraints.
  - Beyond this, the DBMS does not really understand the semantics of the data. (e.g., it does not understand how the interest on a bank account is computed).
  - Thus, ensuring that a transaction (run alone) preserves consistency is ultimately the user's responsibility!



Scheduling Concurrent Transactions

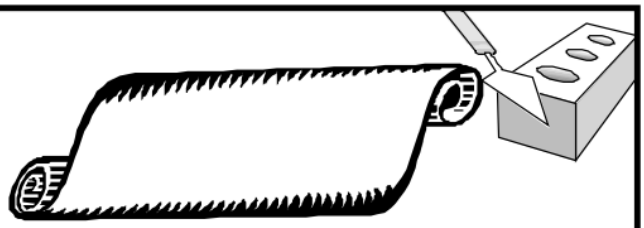
- DBMS ensures that execution of {T1, ... , Tn} is equivalent to some *serial* execution T1' ... Tn'.
  - Before reading/writing an object, a transaction requests a lock on the object, and waits till the DBMS gives it the lock. All locks are released at the end of the transaction. (Strict 2PL locking protocol.)
  - Idea: If an action of Ti (say, writing X) affects Tj (which perhaps reads X), one of them, say Ti, will obtain the lock on X first and Tj is forced to wait until Ti completes; this effectively orders the transactions.
  - What if Tj already has a lock on Y and Ti later requests a lock on Y? (Deadlock!) Ti or Tj is aborted and restarted!

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 12



A crucial property of the log is that each write action must be recorded in the log (on disk) before the corresponding change is reflected in the database itself—otherwise, if the system crashes just after making the change in the database but before the change is recorded in the log, the DBMS would be unable to detect and undo this change. This property is called **Write-Ahead Log**, or **WAL**.

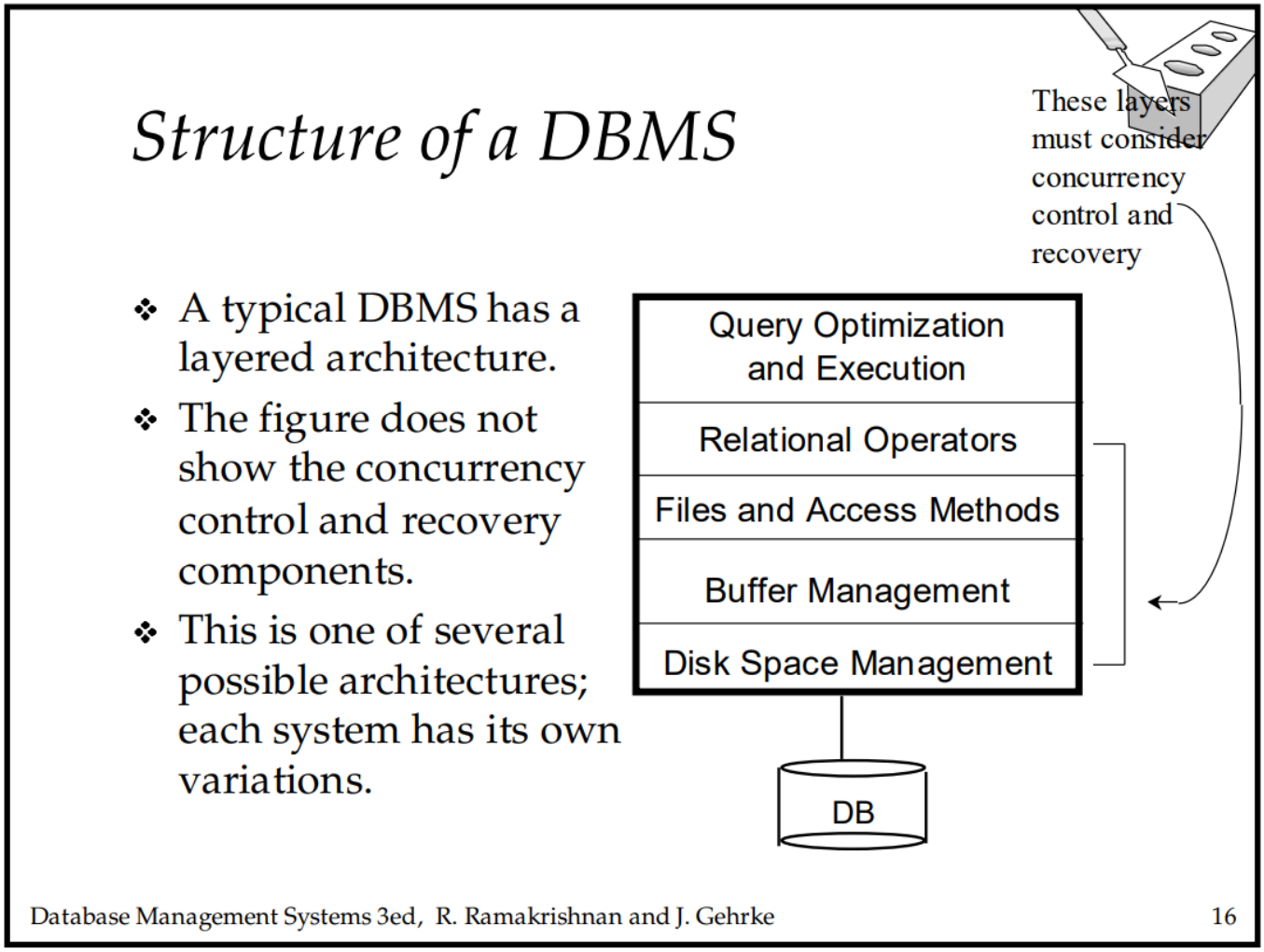
The Log



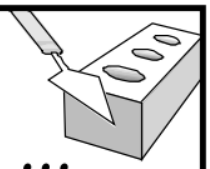
- The following actions are recorded in the log:
  - *Ti writes an object*: the old value and the new value.
    - Log record must go to disk *before* the changed page!
  - *Ti commits/aborts*: a log record indicating this action.
- Log records chained together by Xact id, so it's easy to undo a specific Xact (e.g., to resolve a deadlock).
- Log is often *duplexed* and *archived* on "stable" storage.
- All log related activities (and in fact, all CC related activities such as lock/unlock, dealing with deadlocks etc.) are handled transparently by the DBMS.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 14

What are the main components of a DBMS?



Who is involved with databases in real life?

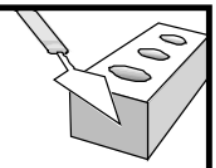


Databases make these folks happy ...

- End users and DBMS vendors
- DB application programmers
  - E.g. smart webmasters
- *Database administrator (DBA)*
  - Designs logical /physical schemas
  - Handles security and authorization
  - Data availability, crash recovery
  - Database tuning as needs evolve

Must understand how a DBMS works!

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 15



Summary

- DBMS used to maintain, query large datasets.
- Benefits include recovery from system crashes, concurrent access, quick application development, data integrity and security.
- Levels of abstraction give data independence.
- A DBMS typically has a layered architecture.
- DBAs hold responsible jobs and are well-paid!
- DBMS R&D is one of the broadest, most exciting areas in CS.

Database Management Systems 3ed, R. Ramakrishnan and J. Gehrke 17