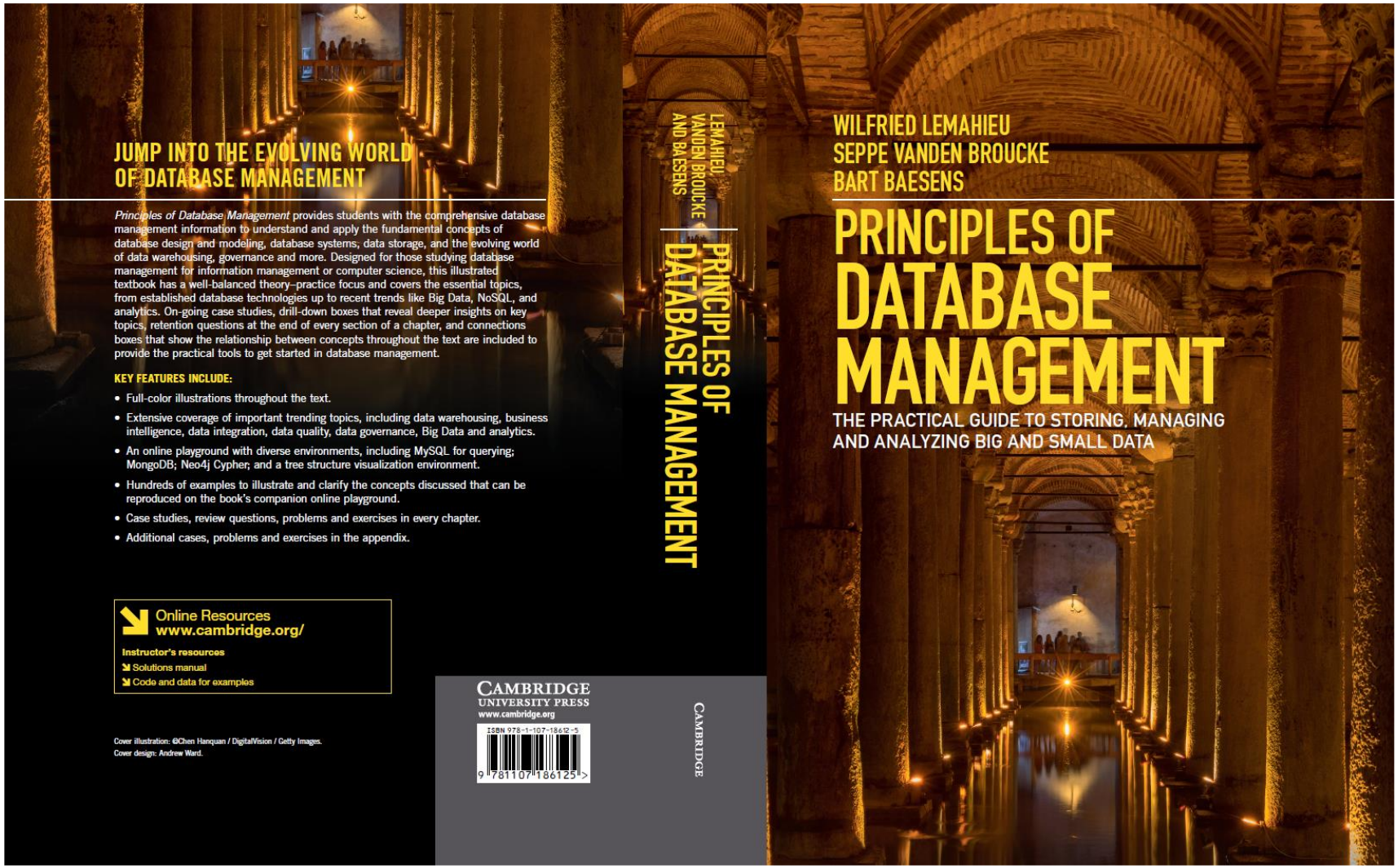
Fundamental Concepts of Database Management



www.pdbmbook.com

Introduction

- Applications of Database Technology
- Key definitions
- File versus Database Approach to Data Management
- Elements of a Database System
- Advantages of Database Systems and Database Management

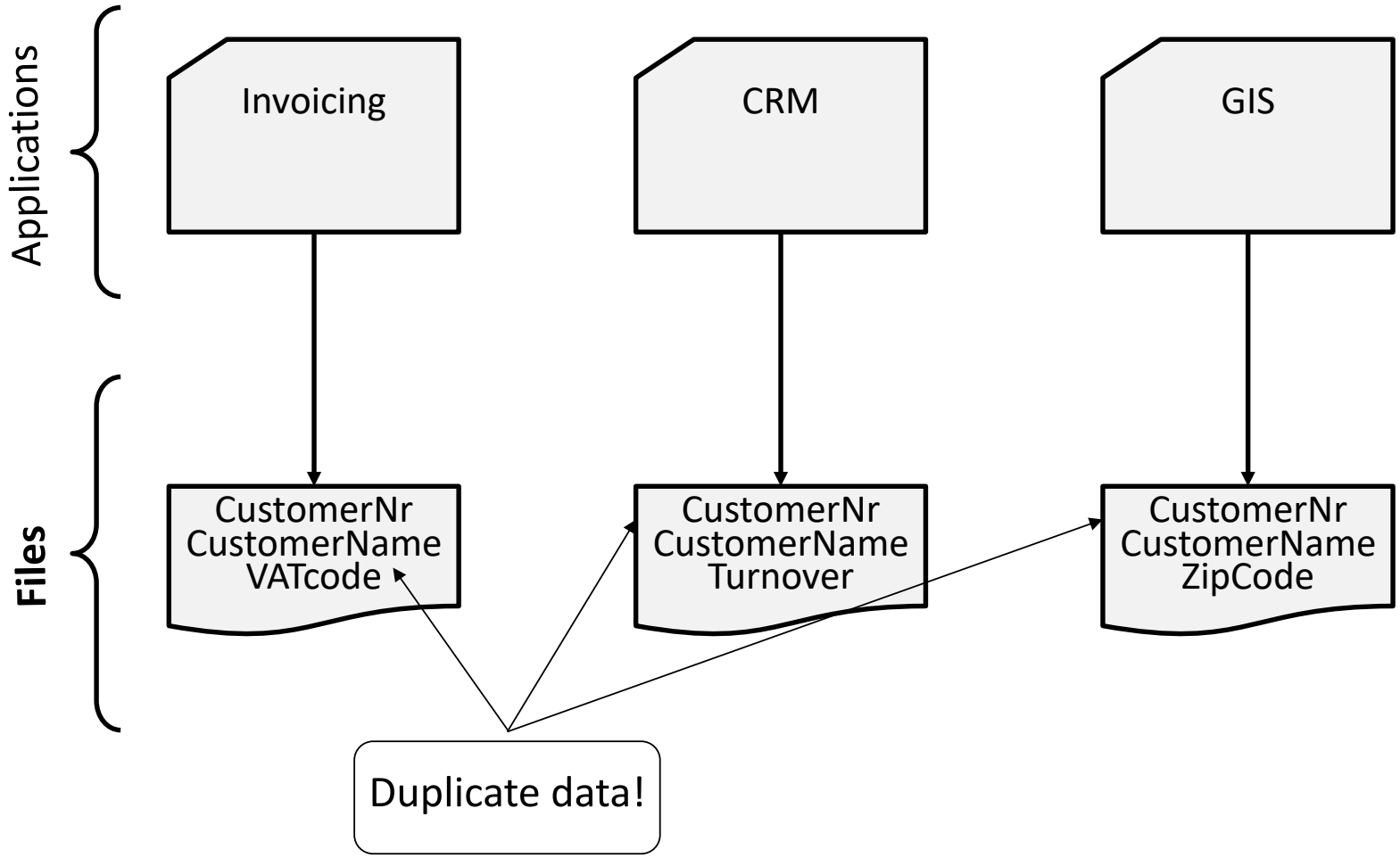
Applications of Database Technology

- Storage and retrieval of traditional numeric and alphanumeric data in an inventory application
- Multimedia applications (e.g., YouTube, Spotify)
- Biometric applications (e.g., fingerprints, retina scans)
- Wearable applications (e.g., FitBit, Apple Watch)
- Geographical Information Systems (GIS) applications (e.g., Google Maps)
- Sensor applications (e.g., nuclear reactor)
- Big Data applications (e.g., Walmart)
- Internet of Things (IoT) applications (e.g., Telematics)

Key definitions

- A database can be defined as a collection of related data items within a specific business process or problem setting
  - has a target group of users and applications
- A Database Management System (DBMS), is the software package used to define, create, use and maintain a database
  - consists of several software modules
- The combination of a DBMS and a database is then often called a database system

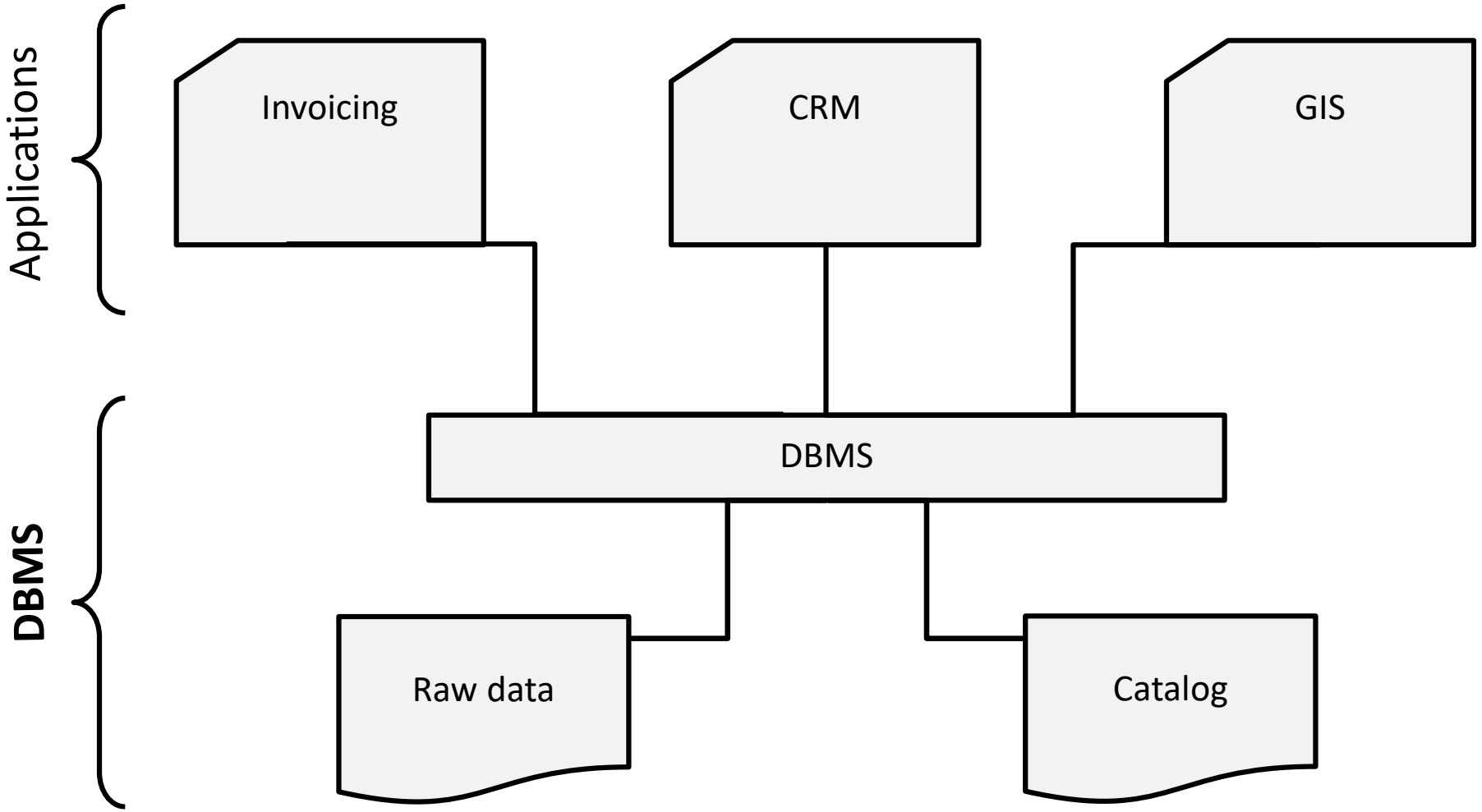
File versus Database Approach to Data Management



File versus Database Approach to Data Management

- File approach
  - duplicate or redundant information will be stored
  - danger of inconsistent data
  - strong coupling between applications and data
  - hard to manage concurrency control
  - hard to integrate applications aimed at providing cross-company services

File versus Database Approach to Data Management



File versus Database Approach to Data Management

- Database approach
  - superior to the file approach in terms of efficiency, consistency and maintenance
  - loose coupling between applications and data
  - facilities provided for data querying and retrieval

File versus Database Approach to Data Management

- File approach

```
Procedure FindCustomer;
begin
     open file Customer.txt;
Read(Customer)
While not EOF(Customer)
If Customer.name='Bart' then
display(Customer);
EndIf
Read(Customer);
EndWhile;
End;
```

- Database approach (SQL)

```
SELECT *
FROM Customer
WHERE
name = 'Bart'
```

Elements of a Database System

- Database model versus instances
- Data Model
- The Three Layer Architecture
- Catalog
- Database Users
- Database Languages

Database model versus instances

- Database model or database schema provides the description of the database data at different levels of detail and specifies the various data items, their characteristics and relationships, constraints, storage details, etc.
  - specified during database design and not expected to change too frequently
  - stored in the catalog
- Database state represents the data in the database at a particular moment
  - also called the current set of instance
  - typically changes on an ongoing basis

Database model versus instances

- Database model

  Student (number, name, address, email)

  Course (number, name)

  Building (number, address)

Database model versus instances

- Database state

STUDENT			
Number	**Name**	**Address**	**Email**
0165854	Bart Baesens	1040 Market Street, SF	Bart.Baesens@kuleuven.be
0168975	Seppe vanden Broucke	520, Fifth Avenue, NY	Seppe.vandenbroucke@kuleuven.be
0157895	Wilfried Lemahieu	644, Wacker Drive, Chicago	Wilfried.Lemahieu@kuleuven.be

COURSE	
Number	**Name**
D0I69A	Principles of Database Management
D0R04A	Basic Programming
D0T21A	Big Data & Analytics

BUILDING	
Number	**Address**
0600	Naamsestraat 69, Leuven
0365	Naamsestraat 78, Leuven
0589	Tiensestraat 115, Leuven

Data Model

- A database model is comprised of different data models, each describing the data from different perspectives
- A data model provides a clear and unambiguous description of the data items, their relationships and various data constraints from a particular perspective

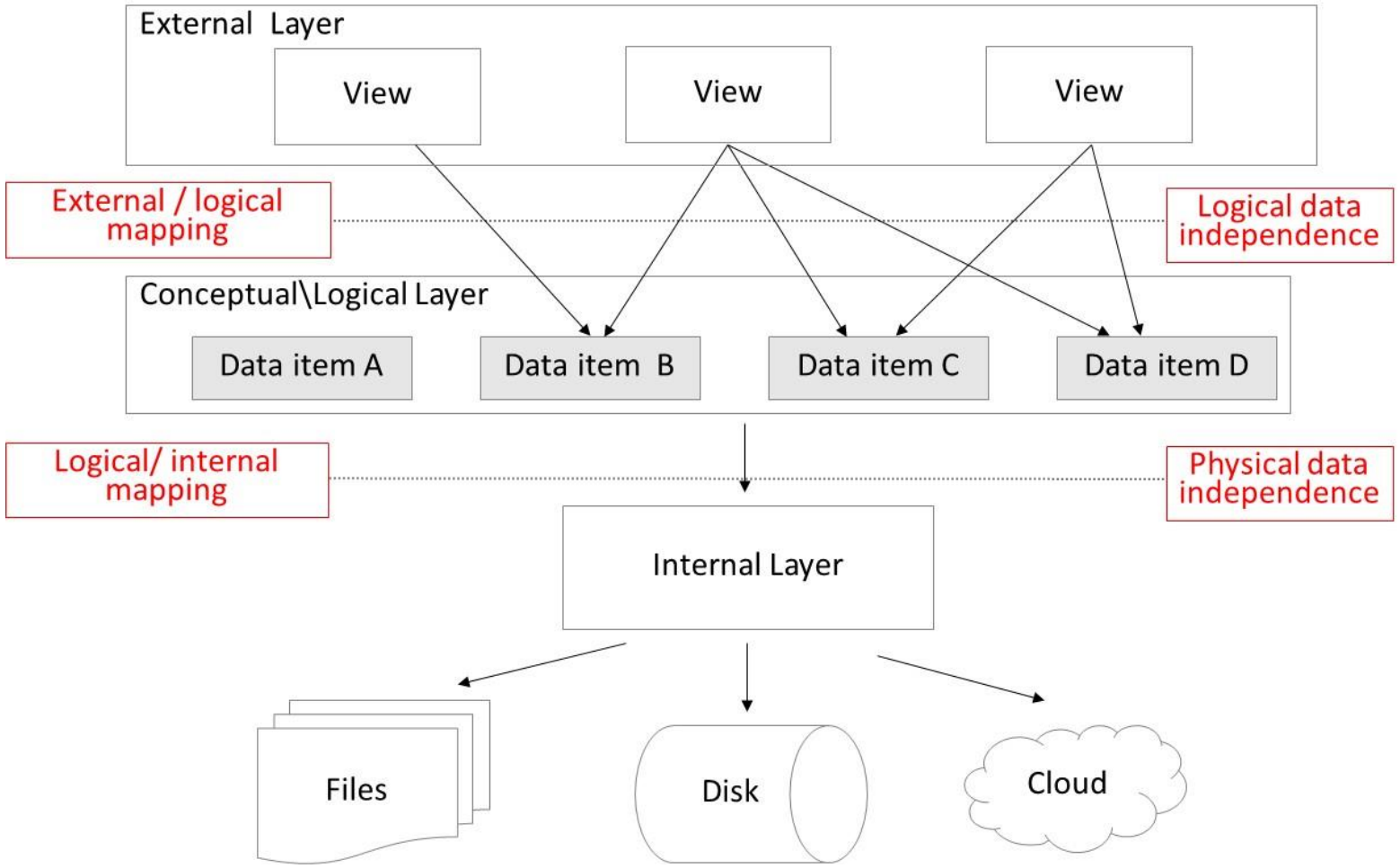
Data Model

- A conceptual data model provides a high-level description of the data items with their characteristics and relationships
  - communication instrument between information architect and business user
  - should be implementation independent, user-friendly, and close to how the business user perceives the data
  - usually represented using an Enhanced-Entity Relationship (EER) model, or an object-oriented model
- Logical data model is a translation or mapping of the conceptual data model towards a specific implementation environment
  - can be a hierarchical, CODASYL, relational, object-oriented, extended relational, XML or NoSQL model

Data Model

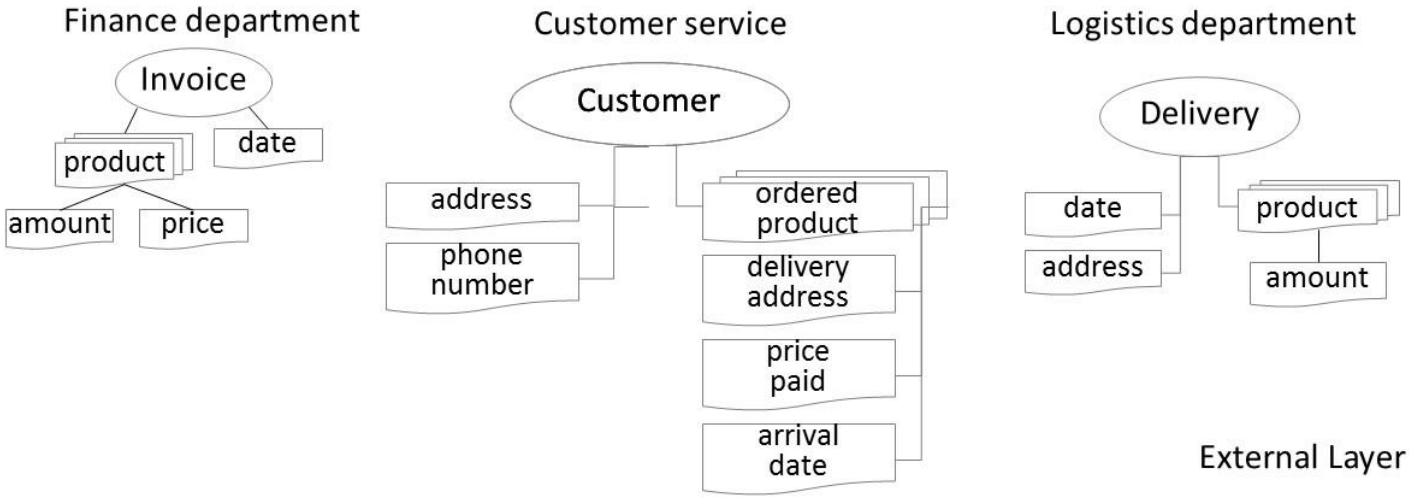
- Logical data model can be mapped to an internal data model that represents the data's physical storage details
  - clearly describes which data is stored where, in what format, which indexes are provided to speed up retrieval, etc.
  - highly DBMS specific
- External data model contains various subsets of the data items in the logical model, also called views, tailored towards the needs of specific applications or groups of users

The Three Layer Architecture

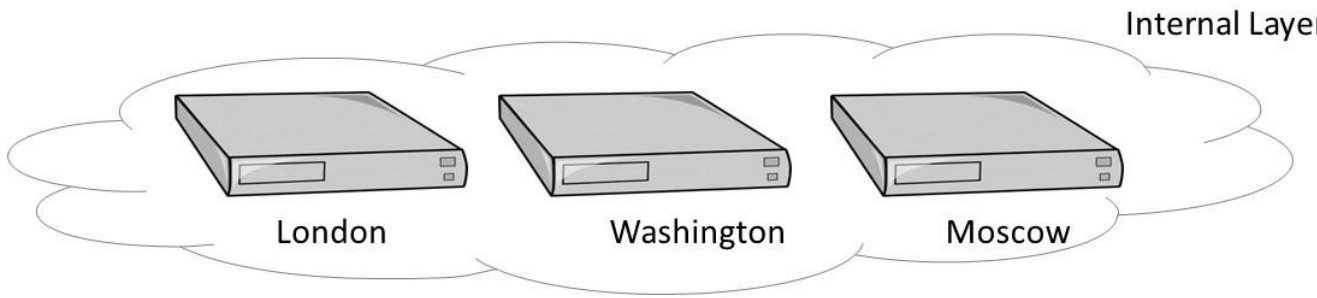


physical data + logical data independence!

The Three Layer Architecture



Product	name, description, cost, …	Conceptual\Logical Layer
Customer	name, phone, address, ...	
Invoice	customer, date, products (with price and amount), …	
Delivery	invoice, address, date, ...	



Catalog

- Heart of the DBMS
- Contains the data definitions, or metadata, of your database application
- Stores the definitions of the views, logical and internal data models, and synchronizes these three data models to make sure their consistency is guaranteed

Database Users

- Information architect designs the conceptual data model
  - closely interacts with the business user
- Database designer translates the conceptual data model into a logical and internal data model
- Database administrator (DBA) is responsible for the implementation and monitoring of the database
- Application developer develops database applications in a programming language such as Java or Python
- Business user will run these applications to perform specific database operations

Database Languages

- Data Definition Language (DDL) is used by the DBA to express the database's external, logical and internal data models
  - definitions are stored in the catalog
- Data Manipulation Language (DML) is used to retrieve, insert, delete, and modify data
  - DML statements can be embedded in a programming language, or entered interactively through a front-end querying tool
- Structured Query Language (SQL) offers both DDL and DML statements for relational database systems

Advantages of Database Systems and Database Management

- Data Independence
- Database Modelling
- Managing Structured, Semi-Structured and Unstructured Data
- Managing Data Redundancy
- Specifying Integrity Rules
- Concurrency Control
- Backup and Recovery Facilities
- Data Security
- Performance Utilities

Data Independence

- Data independence implies that changes in data definitions have minimal to no impact on the applications
- Physical data independence implies that neither the applications, nor the views or logical data model must be changed when changes are made to the data storage specifications in the internal data model
  - DBMS should provide interfaces between logical and internal data models
- Logical data independence implies that software applications are minimally affected by changes in the conceptual or logical data model
  - views in the external data model will act as a protective shield
  - DBMS must provide interfaces between conceptual/logical and external layer

Database Modelling

- A data model is an explicit representation of the data items together with their characteristics and relationships
- A conceptual data model should provide a formal and perfect mapping of the data requirements of the business process and is made in collaboration with the business user
  - translated into logical and internal data model
- Important that a data model's assumptions and shortcomings are clearly documented

Managing Structured, Semi-Structured and Unstructured Data

- Structured data
  - can be described according to a formal logical data model
  - ability to express integrity rules and enforce correctness of data
  - also facilitates searching, processing and analyzing the data
  - E.g., number, name, address and email of a student
- Unstructured data
  - no finer grained components in a file or series of characters that can be interpreted in a meaningful way by a DBMS or application
  - E.g., document with biographies of famous NY citizens
  - Note: volume of unstructured data surpasses that of structured data

Managing Structured, Semi-Structured and Unstructured Data

- Semi-structured data
  - data which does have a certain structure, but the structure may be very irregular or highly volatile
  - E.g., individual users' webpages on a social media platform, or resume documents in a human resources database

Managing Data Redundancy

- Duplication of data can be desired in distributed environments to improve data retrieval performance
- DBMS is now responsible for the management of the redundancy by providing synchronization facilities to safeguard data consistency
- Compared to the file approach, the DBMS guarantees correctness of the data without user intervention

Specifying Integrity Rules

- Syntactical rules specify how the data should be represented and stored
  - E.g., customerID is an integer; birthdate should be stored as month, day and year
- Semantical rules focus on the semantical correctness or meaning of the data
  - E.g., customerID is unique; account balance should be > 0; customer cannot be deleted if he/she has pending invoices
- Integrity rules are specified as part of the conceptual\logical data model and stored in the catalog
  - directly enforced by the DBMS instead of applications

Concurrency Control

- DBMS has built in facilities to support concurrent or parallel execution of database programs
- Key concept is a database transaction
  - sequence of read/write operations considered to be an atomic unit in the sense that either all operations are executed or none at all
- Read/write operations can be executed at the same time by the DBMS
- DBMS should avoid inconsistencies!

Concurrency Control

- Lost update problem

Time	**T1**	**T2**	**balance**
t1		Begin transaction	$100
t2	Begin transaction	read(balance)	$100
t3	read(balance)	balance=balance+120	$100
t4	balance=balance-	write(balance)	$220

Concurrency Control

- DBMS must support ACID (Atomicity, Consistency, Isolation, Durability) properties
  - Atomicity requires that a transaction should either be executed in its entirety or not all
  - Consistency assures that a transaction brings the database from one consistent state to another
  - Isolation ensures that the effect of concurrent transactions should be the same as if they would have been executed in isolation
  - Durability ensures that the database changes made by a transaction declared successful can be made permanent under all circumstances

Backup and Recovery Facilities

- Backup and recovery facilities can be used to deal with the effect of loss of data due to hardware or network errors, or bugs in system or application software
- Backup facilities can either perform a full or incremental backup
- Recovery facilities allow to restore the data to a previous state after loss or damage occurred

Data Security

- Data security can be enforced by the DBMS
- Some users have read access, whilst others have write access to the data (role-based functionality)
  - E.g., vendor managed inventory (VMI)
- Data access can be managed via logins and passwords assigned to users or user accounts
- Each account has its own authorization rules that can be stored in the catalog

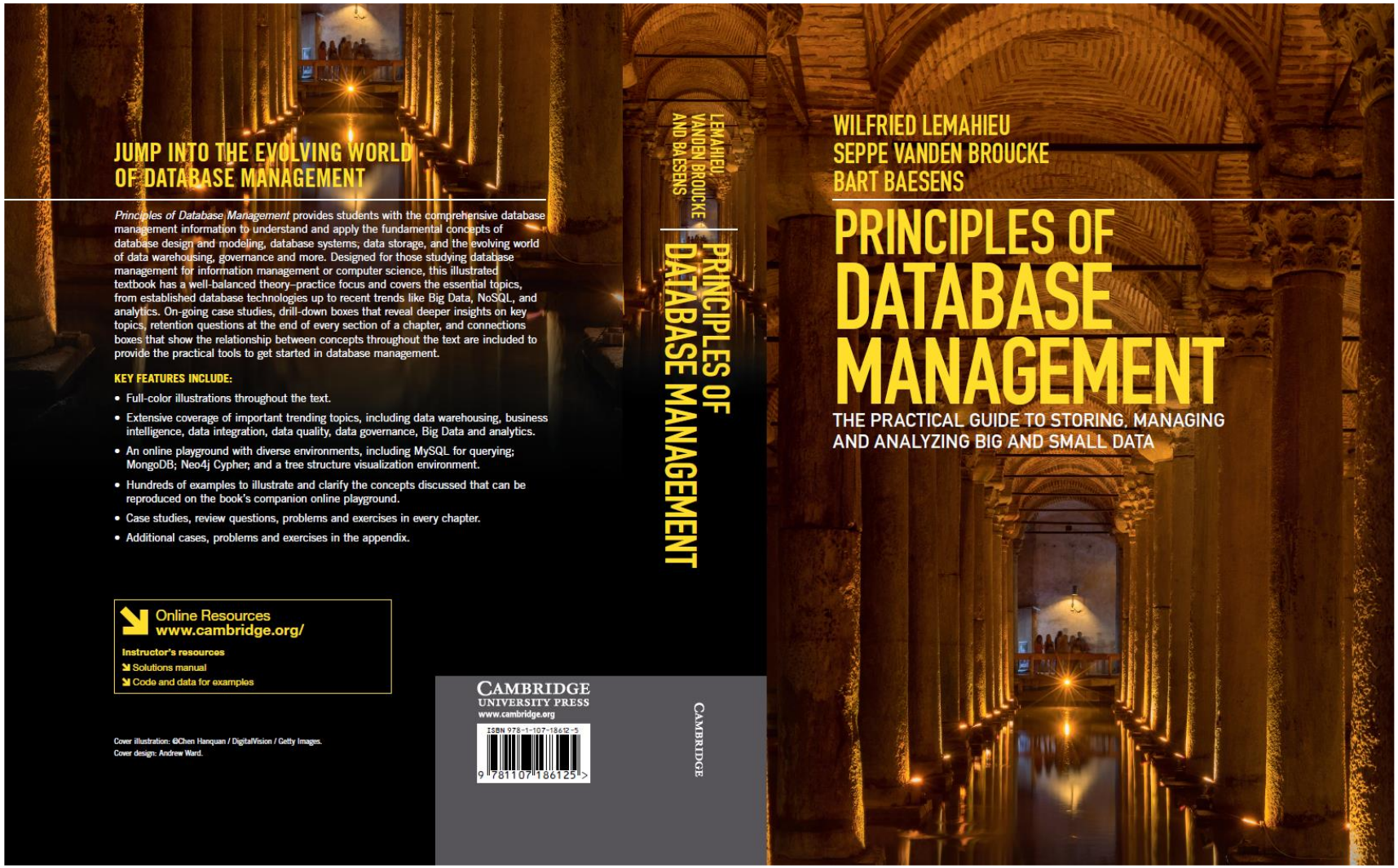
Performance Utilities

- Three KPIs of a DBMS are
  - response time denoting the time elapsed between issuing a database request and the successful termination thereof
  - throughput rate representing the transactions a DBMS can process per unit of time
  - space utilization referring to the space utilized by the DBMS to store both raw data and metadata
- DBMSs come with various types of utilities aimed at improving these KPIs
  - E.g., utilities to distribute and optimize data storage, to tune indexes for faster query execution, to tune queries to improve application performance, or to optimize buffer management

Conclusions

- Applications of Database Technology
- Key definitions
- File versus Database Approach to Data Management
- Elements of a Database System
- Advantages of Database Systems and Database Management

More information?



www.pdbmbook.com