Data Management and Databases

Welcome to the Data Management and Databases course!

About the course

- During the course, we will learn among other things:
  - The key concepts and terminology of data management and databases
  - Design and document the database's structure based on the requirements
  - Retrieve and manipulate the database's data with SQL
- Each week there are two different teaching sessions:
  - Lectures where we go through the theory of the weekly topics
  - Laboratory sessions where we apply what we've learned by working on exercises together
- The teaching session schedule can be found on the course's Moodle page
- There are mandatory weekly exercises that need to be submitted in Moodle before the next week's lecture

Introduction to databases

Database

> Data: "A representation of facts or ideas in a formalized manner capable of being communicated or manipulated by some process"

> Base: "A permanent structure for housing something"

- In a digital world we are constantly accessing and manipulating stored information:
  - When we open our favorite messaging app, we can see the previously sent messages and we can send new messages
  - When we pay a bill and transfer money from one bank account to another
- These kinds of shared collections of logically related information are *databases*

Definition of database

- Database is...
  - a *shared collection* of
  - *logically related persistent data* and
  - a *description of this data*,
  - designed to meet the *information needs*

Definition of database

- "shared collection": database is accessible to specific applications, users, and organizations
- "logically related data": the different pieces of information has logical relations, e.g. messages in a messaging app are related to the sender and the receiver users
- "persistent data": data is in permanent storage and doesn't unexpectantly vanish
- "description of this data": on top of the actual data such as the user's name, the database contains *metadata* like table and column names
- "information needs": the kind of information stored in the database is use-case specific
  - For example, a simple messaging app needs to store information about users and messages

Data management

- *Data management* is the development, maintenance and coordination of *database systems*
- A database system consists of five major components: *hardware*, *software*, *data* (the database), *procedures* and *users*
- Procedures refer to the policies, conventions, instructions, and rules that govern the design and use of the database

The objective of data management

- The objective of data management is to design, implement, coordinate, and maintain database systems in such a way that all the required data is:
  - Valid and consistent
  - Up to date
  - Available in the required format
  - Available when needed
  - Fetrievable fast enough
  - Safe from different types of technical failures and accidents
  - Protected from unauthorized access and other types of misuse

Data management example

- Let's consider the following information needs for a database:
  - A real estate company is renting properties
  - Each property has a property owner and a lease if the property is rented
  - Each lease has a client who is renting the property from the owner
  - The company has a sales team responsible for finding clients for the available properties and a contracts team responsible for managing the leases

File-based approach in data management

- Data for sales and contracts teams is in *separate files*

- Each team uses a separate application that defines and manages data in application-specific files

- Each file has a specific format

- Applications that use these files depend on knowledge about that format
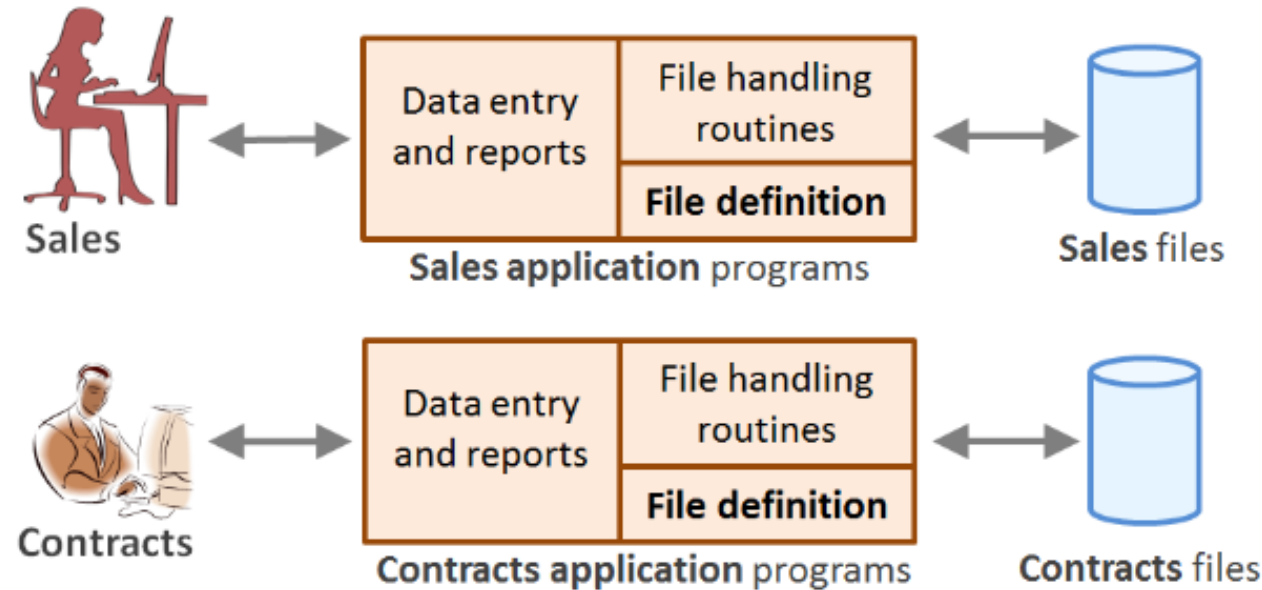
- In the filesystem, there are the following sales files:
  - PrivateOwner (ownerNo, fName, lName, address, telNo)
  - PropetyForRent (propertyNo, street, city, postcode, rooms, rent, ownerNo)
  - Client (clientNo, fName, lName, address, telNo, prefType, maxRent)
- And the following contracts files:
  - Lease (leaseNo, propertyNo, clientNo, paymentMethod, deposit, paid,
  - rentStart, rentFinish, duration)
  - PropetyForRent (propertyNo, street, city, postcode, rent)
  - Client (clientNo, fName, lName, address, telNo)

Problems of file-based approach

- Data dependence
  - Code is tightly coupled with the file structure: if it is modified, all programs that use the file have to be changed accordingly

- Duplication of data
  - Storing the same information in multiple files will lead to inconsistency

- Difficulty in accessing data
  - New requirement needs a new program or changes in an existing program

- No provision for security and shared access to the data
  - No service for providing user access to some, but not all, data

- Lack of coordination and standardisation
 - No centralised control of enterprise data

Database Management System (DBMS)

- Problems of the file-based approach can be avoided by delegating data related operations to a *separate software*
- *Database Management System* (DBMS) is the software that:
  - Controls all access to the database
  - Allows users to define the database, usually through a *Data Definition Language* (DDL)
  - Allows users to insert, update, delete, and retrieve data from the database, usually through a *Data Manipulation Language* (DML)
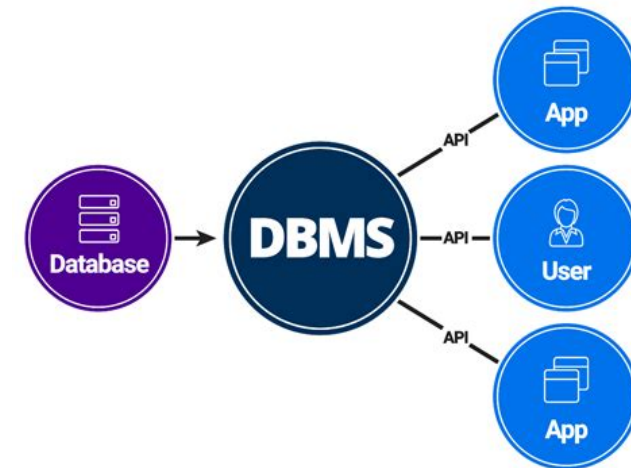
Database approach in data management

- Data for both sales and contracts team is in the *same database*

- Each team uses a separate application that communicates with the same DBMS using a data manipulation language

- The DBMS retrieves and manipulates data in the database on behalf of the application
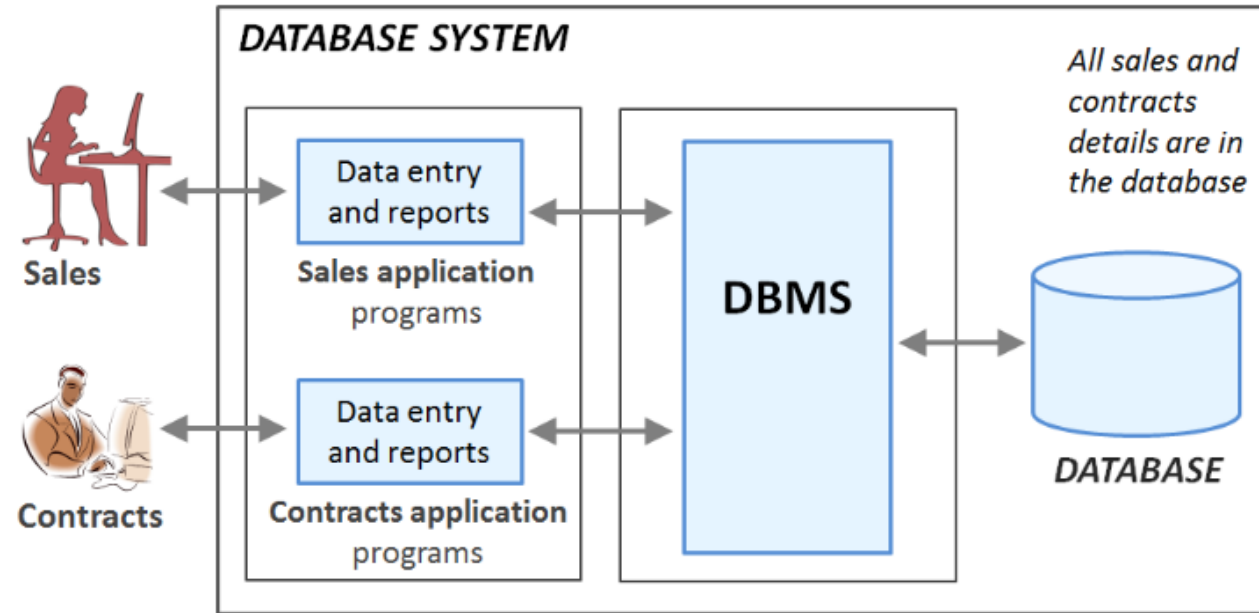
In the database, the structure of sales and contracts details is the following:

- *PrivateOwner* (ownerNo, fName, lName, address, telNo)
- *PropetyForRent* (propertyNo, street, city, postcode, rooms, rent, ownerNo)
- *Client* (clientNo, fName, lName, address, telNo, prefType, maxRent)
- *Lease* (leaseNo, propertyNo, clientNo, paymentMethod, deposit, paid, rentStart, rentFinish)

Advantages of database approach

- Program-data independence
  - Improved data accessibility
- Effective access to data
  - Standard database language with both programmatic and interactive interfaces
- Data integrity
  - Integrity can be maintained with the support of user-defined integrity constraints
- Data security
  - Security restrictions can be applied on detailed level
- Coordination and standardisation
  - Centralised data administration

Functions of a DBMS

- The most fundamental function of DBMS is *retrieving and manipulating data in the database*

- This function should be provided in such a way that the physical level storage structures are completely hidden from the user
  - This offers a great amount of flexibility: the storage structures can change without the need to touch the application's code

- Other important functions of a DBMS are:
  - Integrity Services
  - Transaction Support
  - Concurrency Control Services
  - Recovery Services
  - Authorization Services

Database Management System (DBMS)

- Nowadays, the *relational database management system* (RDBMS) is the de facto standard

- *SQL* is the formal and de facto database language standard for RDBMSs

- SQL has both DDL and DML features

- There are multiple RDBMS products, such as *MySQL* and *PostgreSQL*

Summary

- *Database* is a shared collection of logically related persistent data
- Database is designed to meet specific information needs
- *Data management* is the development, maintenance and coordination of *database systems*
- *Database management system* (DBMS) is a software that allows users to insert, update, delete, and retrieve data from the database
- The most fundamental function of DBMS is *retrieving and manipulating data in the database*