1A. Overview of Databases

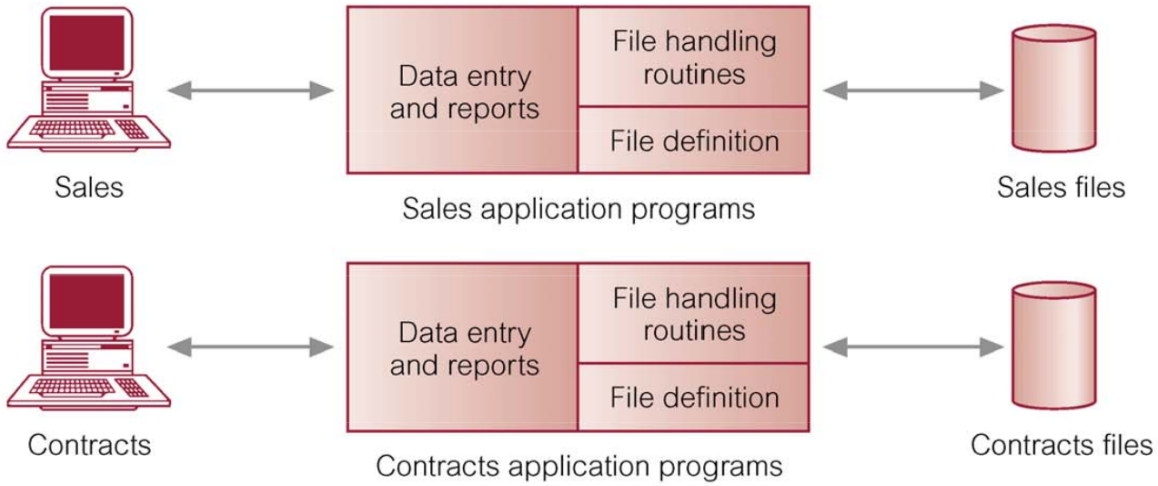
Importance of data & databases

- Organizations generate data.
  - Employees, documents, products, sales, etc.
- Behind every web application is a database.
  - Social media / web 2.0 site: users, friends/subscriptions, user-generated content (posts, photos, etc.), account settings, usage statistics.
- Important to be able to load, search, and analyze this data.

Why do we need databases?

- Contrast with older paradigm: file-based systems.
  - Individual applications defining and using their own data types,
  - Unaware of existence of other data,
  - Unintentional duplication of data (inefficient, hard to maintain consistency),
  - Heterogeneous file formats,
  - Analysis and integration of information from multiple sources may be difficult.
- A database **centralizes** data so that it can be stored and managed in one place, standardizing format and minimizing duplication.

File-based structure



Sales Files

PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo)
PrivateOwner (ownerNo, fName, lName, address, telNo)
Client (clientNo, fName, lName, address, telNo, prefType, maxRent)

Contracts Files

Lease (leaseNo, propertyNo, clientNo, rent, paymentMethod, deposit, paid, rentStart, rentFinish, duration)
PropertyForRent (propertyNo, street, city, postcode, rent)
Client (clientNo, fName, lName, address, telNo)

- Each application defines and manages its own data.

A database is…

- “a shared collection of logically related data (and a description of this data), designed to meet the information needs of an organization”
- “a *self-describing* collection of integrated records”

Components of a database

- Data definition language (DDL) – specifies format of data in the database (**data model**).
  - Includes constraints on data (integrity checks) and permissions/security.
- Data manipulation language (DML) – standardized way of specifying updates to data in the database.

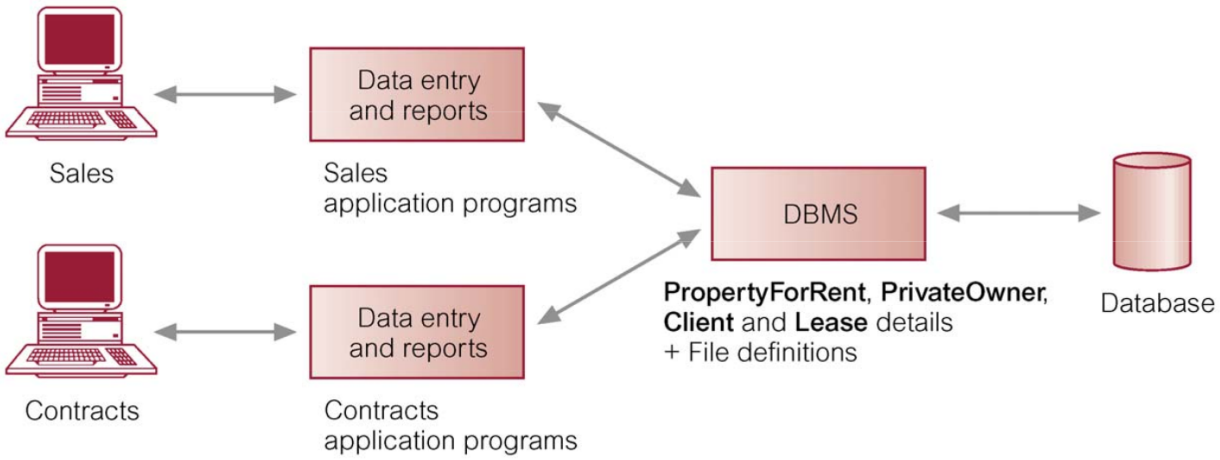
A database management system is…

- “a software system that enables users to define, create, and maintain a database and that provides controlled access to this database”

Functions of a database management system (DBMS)

- Database management systems (DBMSs) also centralize **useful functions** for managing and accessing that data across an organization/project :
  - Methods of data access,
  - Performance optimizations,
  - Security,
  - Ensuring data integrity,
  - Update failure recovery.
- Downsides:
  - Increased size / complexity / cost.

DBMS structure



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

- Multiple applications share a database managed by software.

DBMS: the current paradigm

- Relational database management systems (RDBMS).
  - **Formalized** by relational algebra/calculus.
  - **Standardized** by SQL (Structured Query Language) standard.
  - **Implemented** by various off-the-shelf products (MySQL, Oracle, MariaDB, SQLite, PostgreSQL Microsoft SQL Server, IBM Db2, etc.).
- Before RDBMS: hierarchical databases.
- Current alternatives: object-oriented (especially using JSON).

Rest of this module

- First two weeks:
  - Using RDBMSs: SQL, designing relational databases.
- Weeks 3-6:
  - Relational database implementation and optimization:
    - Relational algebra, storage, query processing & optimization, transaction management & recovery.
- Weeks 7-8:
  - Distributed databases.
- Week 9-10:
  - Beyond RDBMSs.

Early weeks

- Coursework (50%) involves designing and making use of your own SQL database.
  - Material needed to get started on this task is front-loaded into the first two weeks.
- Week 1:
  - Introduce yourself / form groups (on Moodle).
  - Understand SQL (exercises on Moodle).
- Week 2:
  - Learn about ER diagrams and normalization.
  - Finalize groups and peer-marking criteria by end of Week 2.
- Week 3:
  - Submit an ER diagram for your project by end of Week 3.

Rest of this lecture

- Introduction to databases.
- Generic architecture of database management systems.
- Introduction to relational databases.
- Introduction to SQL.

Design principles / architecture of a DBMS

Three-tier DB architecture

- External level – subschemas:
  - Views of the data accessible by users.
  - Can be set per user and hide details they don’t need to / shouldn’t know.
- Conceptual level:
  - Full specification of the logical/conceptual structure of the database, minus platform-dependent implementation details.
- Internal / physical level:
  - Data structures and algorithms defining how data is physically stored on disk.

Abstraction & interfaces

- Each layer abstracts away details from previous layers.
- Allows aspects of the database to be less coupled to each other / interfaces remain stable when other aspects are updated.
  - Updating full database schema might not affect users' views .
  - Changing implementation of database doesn't require changing the schema.
- Get the flexibility of specialized applications (decentralized) and the benefits of unified, comprehensive storage/management of data (centralized).

Common parts of a DBMS

- Query processor: translates queries into low-level instructions.
- Database manager: large and complicated piece of code with many functions such as authorization, integrity-checking, query optimization, scheduling, failure recovery, and temporary data storage.
- File manager: stores data and auxiliary data structures on physical hardware.
  - Interacts with operating system as necessary.

Multi-user databases

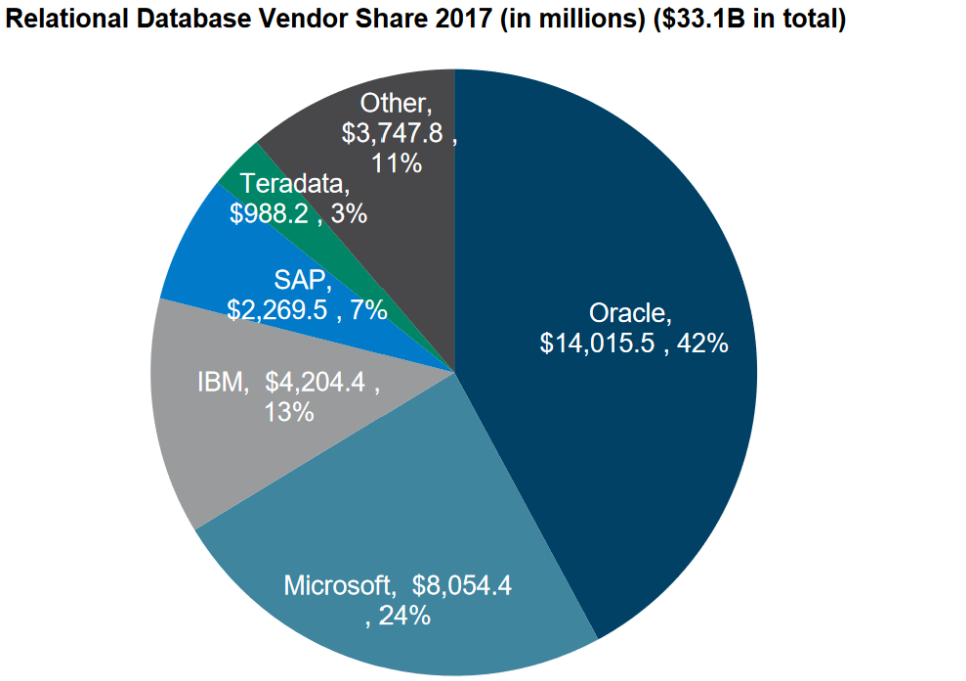
- How to set up a single database so that it can be used by multiple users / applications?
- Has changed over the years and settled into **client-server** architecture.
  - Client (frontend): [example: web browser] makes requests, handles interactivity, application logic maybe involving light data-processing.
  - Server (backend): handles DB queries + storage.

Three-tier client-server model

- Server can be further split into two parts: data server and application server.
- Three tiers:
  - Thin client: display + UI (web browser).
  - Application server: all application logic, request handling.
  - Data server: data storage + retrieval.
    - Possible to further split data server in high-traffic DBs into a transaction manager (queues/schedules updates and handles rollbacks) and resource manager (read/write from file).

Database software

- Relational database software:
  - Oracle
  - Microsoft Access / SQL Server
  - IBM Db2
  - SAP
- Open-source:
  - MySQL
  - MariaDB
  - PostgreSQL
  - MongoDB



From: "Database Software Market: The Long-Awaited Shake-up," William Blair, March 2019

Databases and cloud computing

- Cloud: “A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”
- On-demand / self-service.
- Database as a service (DBaaS) – a type of software as a service (SaaS) that is becoming popular and starting to replace DBs managed in-house.

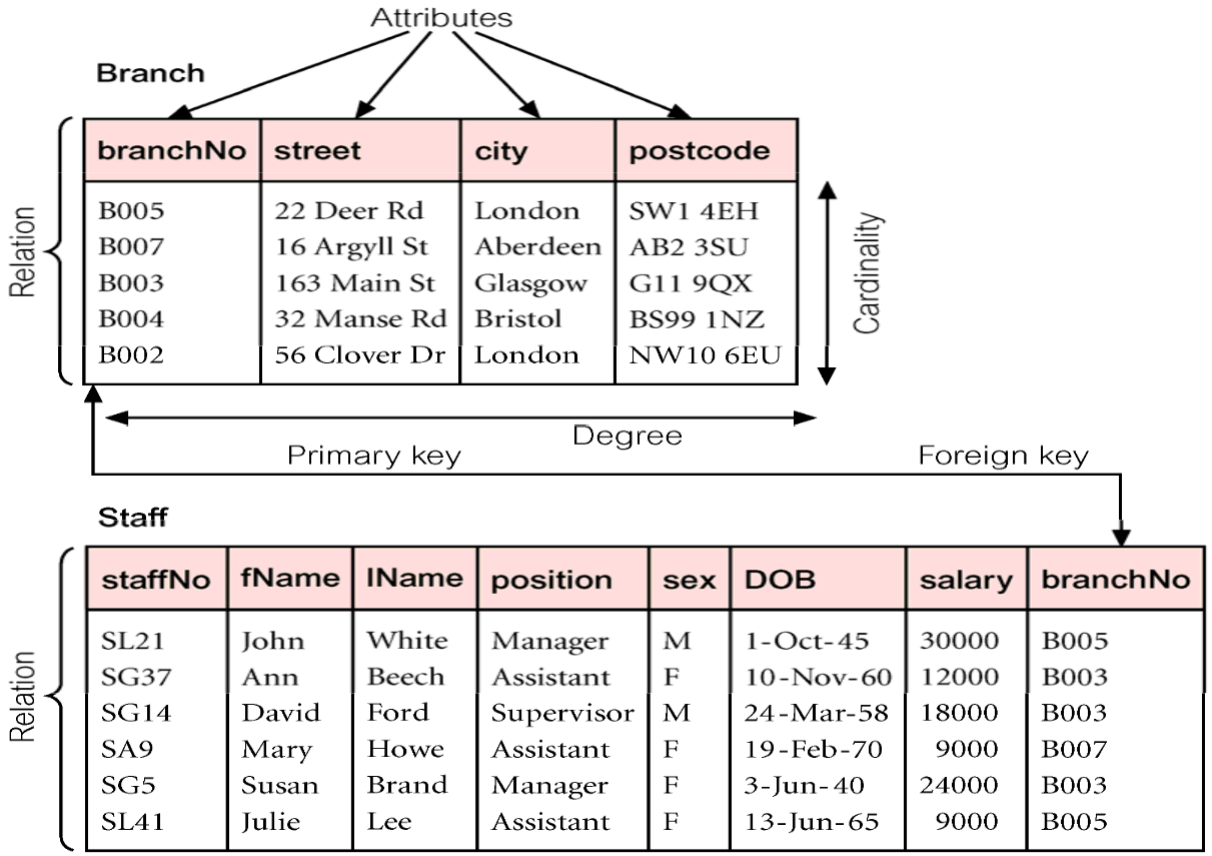
DBaaS vendors

- Examples:
  - Amazon AWS,
  - Microsoft Azure,
  - Google Cloud,
  - Cloud versions of IBM Db2, SAP, etc.
- Often provide multiple options (relational, non-relational, etc.).

Intro to relational databases

Tables

- For now, can think of the basic unit of a relational database as being a 2-D table.
  - We'll go over this in more detail later.
- One table represents a thing, entity, relationship (Capitalized)
  - Branch; Staff
- Columns = "attributes" or fields (lowerCamelCase)
  - Domain: set of allowable values
- Rows = "records"

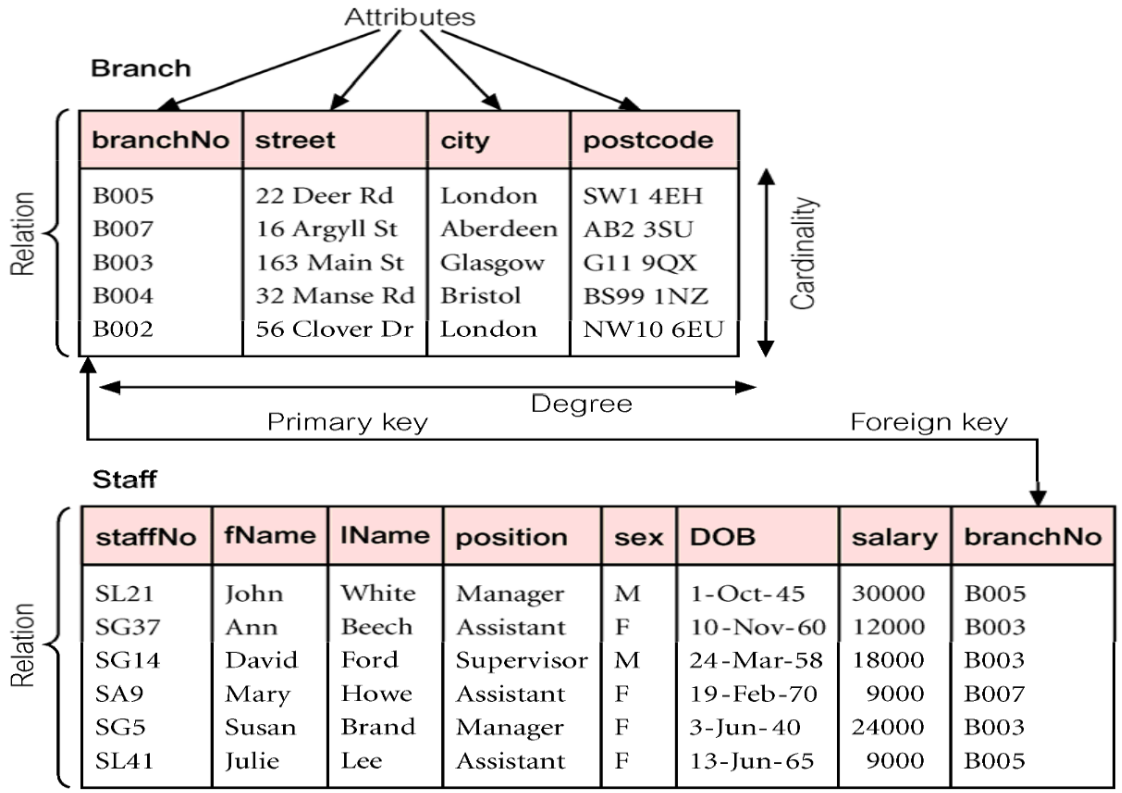


branchNo	street	city	postcode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Terminology & key concepts

- Degree = # cols
- Cardinality = # rows
- Every table has a primary key (underlined), an attribute(s) that uniquely determines a record.
  - Branch (branchNo, street, city, postcode)



Database = set of tables

- Need to define domains of attributes in all tables (data definition).
- Need to populate tables with records (data manipulation).

Simple structure!

- Similar to an Excel table.
- Regular (rectangular) structure.
- Easy to analyze and manipulate.