IN3020/4020 – Database Systems
Spring 2021, Week 1.1

Introduction

Dr. M. Naci Akkøk, Assoc. Prof., Ifi/UiO and CEET/OsloMet, CEO In-Virtualis AS

Dr. Egor Kostylev, Assoc. Prof., Ifi/UiO

Based upon slides by M. Naci Akkøk, Spring 2020 & E. Thorstensen from Spring 2019

UiO : Institutt for informatikk
Det matematisk-naturvitenskapelige fakultet

The course

- Database systems, databases and their usage

  - **Databases** are abstract objects (an entity or model with **data structures** and **a query language**) for storing and retrieving **data**
  - **Database systems** (also referred to as Database Management Systems or **DBMS**) are software systems that allow us to manage these objects, i.e., the databases: They are the “engines” that help manage/run databases.

Goals:
At the end of the course, you should be able to…

- Understand what Database Management Systems (DBMS) do, how they work and how they are implemented
- Have a good grasp of the theoretical & practical aspects of database administration
- Have a good grasp of use of SQL and how queries are optimized
- Understand ACID principles & the principles of transaction management
- Know about the TX management types (isolation levels) that modern DBMS offer, and their strengths/weaknesses

Inside the database "black box"

- What is apparent to us is that we write an SQL query, the database does something and comes up with a response
- But there is a lot that happens behind the scenes, inside the black box.
- This course is meant to show what happens behind the scenes in the black box, so that **we understand the common challenges** and what to do about them as users and database administrators and understand **how to build database engines** some day.

Some common challenges

- What can one do if the queries are slow?

- How can one read from and write to the same database at the same time without problems? How would the performance be affected?

- What happens if the database crashes while updating the data?

Some of the background we will need to build up for the course

- DBMS architecture
- DBMS file systems
- Various data structures and algorithms
- Relational algebra

The syllabus can be divided into three parts

- Main focus:
  - Relational queries, SQL and query optimization
  - ACID, DBMS characteristics and mechanisms
  - Other DBMS (slightly less than previous years)

- And an additional bit about emerging technologies and database research

Syllabus: Queries, SQL and query optimization (E. Kostylev)

- SQL (repetition and a bit more)
- Query formalization/optimization (including relational algebra), query plans
- Indexes, index usage and the underlying search

Syllabus: ACID, DBMS characteristics and Mechanisms (M. N. Akkøk)

- Managing multiple concurrent uses/users and protecting the data
  - ACID – what it stands for

  - Transaction Management
  - Locking, logging, buffer & cache management

  - Relationships between these mechanisms
  - Synchronization/replication challenges

Syllabus: Other database systems (briefly) – NoSQL DBMS and newer DBMS technologies

- Categorization of the NoSQL DBMS and the problems they address (their “reason of being”)
- A bit more about spatial & graph databases, semantic technologies and databases with built-in intelligence (AI/ML capabilities in and outside the “black box”)
- Autonomous databases
- Multimodal databases

Not really syllabus, but food for thought: Emerging DBMS trends and research areas

- Performance – how and for what purpose, exactly?
- Big or fast data – or both? How? How can newer architectures like the Data Mesh help?
- What is the relation doing in the table? What happens to indexes and keys and search (queries, joins etc.) if it is not part of the data? Can it be as cleanly formalized in an “extended” relational algebra?
- Intelligence in databases – autonomy, auto-inference (also in RDF semantics) and other built-in mechanisms of learning and intelligence.
- Extraction-transformation-loading challenges: Intelligence in data quality management and I/O
- Knowledge Bases: extend data with knowledge, usually in logical form, and efficiently manipulate (e.g., querying) both the explicit data and implicit data that logically follows from the knowledge

Dates & practical information

- Two mandatory exercises (*oblig´er*) due Friday 5. March and Friday 30. April 2021 (do follow updates on dates)
  - **Group sessions** dedicated to the mandatory exercises are **the week before the deadlines**
  - Make sure you do NOT co-operate too tightly! Check the rules for the mandatory exercises (*obligreglement*)
- The final written exam is a 4 hours **Inspera Digital Exam** without aids (*uten hjelpemidler*) on the 16th of June 2021 from 09:00 to 13:00
- **Deadline for withdrawing from the exam is 1st of May!**
- **ALWAYS CHECK DATES**, because things do change ☺

Dates & practical information

- We may have to be on Zoom throughout the semester but check the announcements on the course´s web site.
- Groups as well.

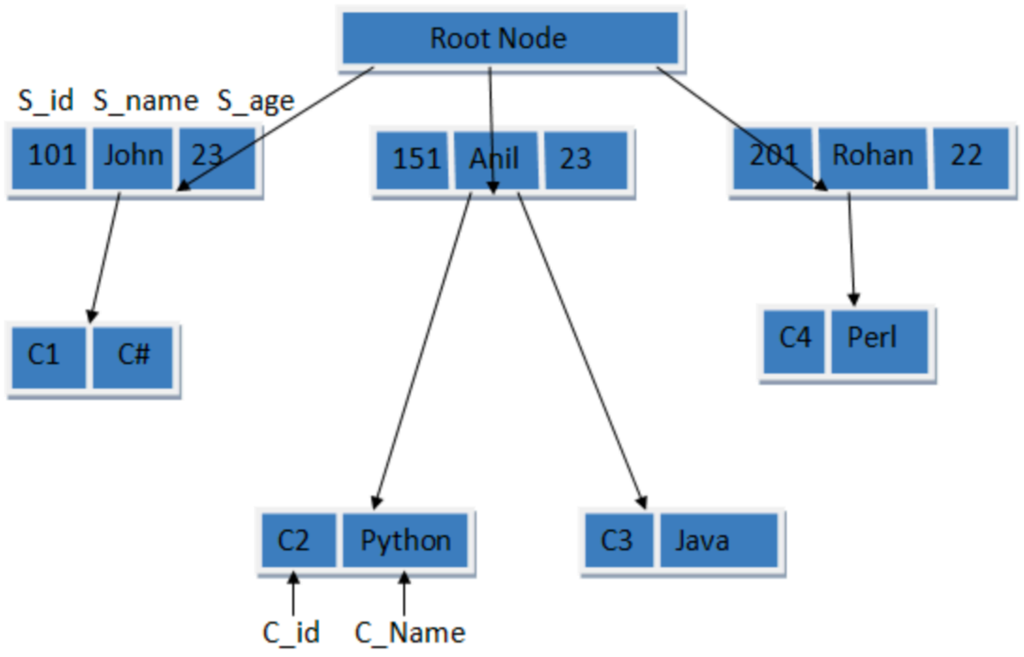
- Yes, we will record the lectures and put them out in the “Schedule”, and we will do our best to publish the slides in advance ☺
- We believe we will use Padlet (not Piazza) as the communication service. We will provide info and link on the Web-site of the course very soon

Brief Database history (look them up)

- 1960-ies:
  - IDS (network DB model) and IMS (hierarchical DB model)
  - Mainframe period. 1 «big» machine for 1 mill. 1964-US$ with 512 Kbyte RAM
  - 50 Mbyte disk
  - Other I/O-equipment magnetic & paper tape, punched-card reader and teletype (which is why unix has a funny abbreviation like tty for the «terminal»)
- Network and hierarchical databases are somewhat related.

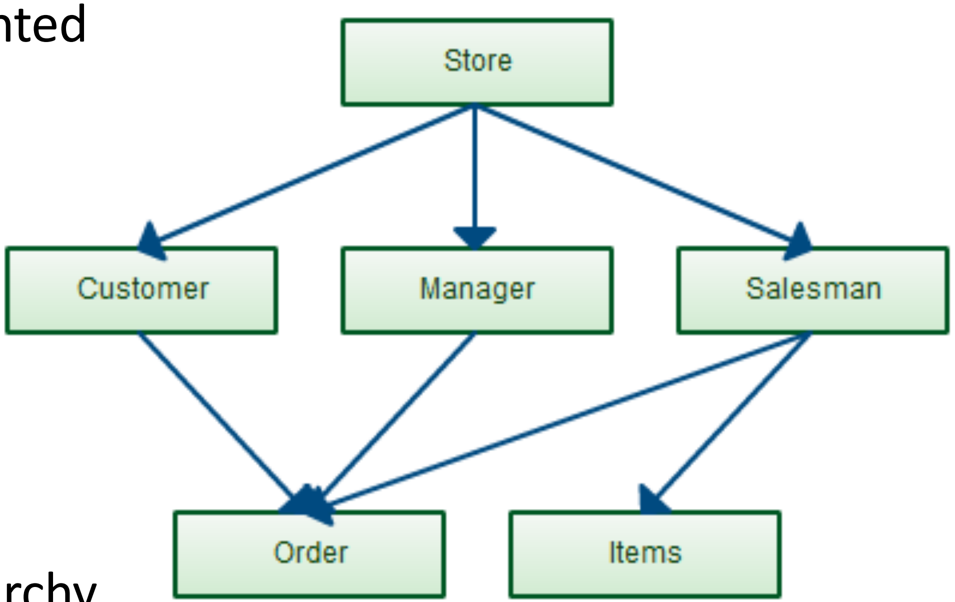
The Hierarchical Database Model

- «In the **hierarchical database model** data is represented in a tree-like structure. It represents a parent-child relationship with a single parent for each child» (but allows for more than one child per parent)
- Own query language.
- Advantages / disadvantages?



The Network Database Model

- The **Network Model** is an extension of the hierarchical model. In this data is represented in the form of graphs with more than one parent node for one child node.
- Advantages:
  - Easy to implement.
  - Datamodel follows physical model.
  - Easy to learn?
- Disadvantages:
  - Easy only when query follows the hierarchy.
  - Updates?



Relational DB/DBMS (The Relational Model)

- E. F. Codd. 1970. A relational model of data for large shared data banks. Commun. ACM 13, 6 (June 1970), 377-387.

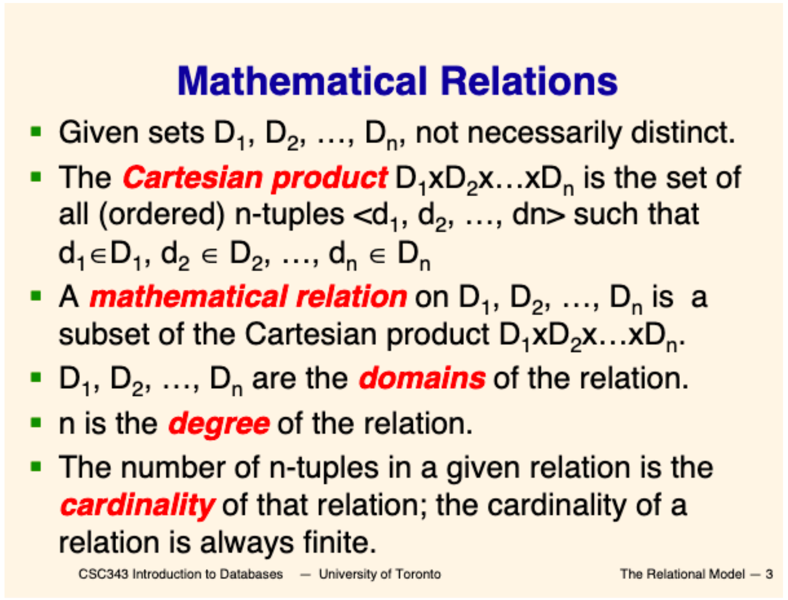
- “Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.”

What did Codd mean?

- Internal representation (files, pointers) should not be visible to the user
- Users work on an abstract data model
- Codd suggests relations as an abstract data structure

Relations (the table version)

- Tables with rows (tuples) and columns (attributes).
- A tuple has a value for each attribute
- Table is based upon a variant of the mathematical relation: Set of tuples $R \subseteq d(A1) \times \cdot \cdot \cdot \times d(Ak)$.
- Note: «Contrary to the usual definition in *mathematics*, there is no ordering to the elements of the tuples of *a relation*!»
- Can be manipulated through operations on the tables and comparison of values



Mathematical Relations

- Given sets $D_1, D_2, \ldots, D_n$, not necessarily distinct.
- The ***Cartesian product*** $D_1 x D_2 x \ldots x D_n$ is the set of all (ordered) n-tuples $<d_1, d_2, \ldots, dn>$ such that $d_1 \in D_1, d_2 \in D_2, \ldots, d_n \in D_n$
- A ***mathematical relation*** on $D_1, D_2, \ldots, D_n$ is a subset of the Cartesian product $D_1 x D_2 x \ldots x D_n$.
- $D_1, D_2, \ldots, D_n$ are the ***domains*** of the relation.
- n is the ***degree*** of the relation.
- The number of n-tuples in a given relation is the ***cardinality*** of that relation; the cardinality of a relation is always finite.

CSC343 Introduction to Databases — University of Toronto The Relational Model — 3

Simple definitions of mathematical relation, function, graph etc. with examples:

https://medium.com/brandons-computer-science-notes/mathematical-relations-5416f027dbbe

Note: In mathematics, a *tuple* is a finite ordered list (sequence) of elements.

Characteristics of Relations

- Every attribute has its own domain (i.e., set of allowed values)
- All attributes have different (distinct) names. If not, we distinguish them (for example) by giving them sequence numbers
- Relations are sets: Tuples have no order, and no two are alike
- «A table cannot have 0 columns, rows can be duplicated»
- «... and tables cannot be nested»
  (careful with newer object-relational databases and databases that allow for abstract data types or ADTs)!
- Attributtes can have the special value of nil (null)

Relational DB/DBMS (the Relational Model)

- Relation is a an abstract data model. Implementation varies
- We allow for defining redundancy and other characteristics
- We have a very elegant and compact query language (SQL) that builds upon relational algebra
- We allow for ad-hoc queries
- It is considered to be a universal data model

Fast forward …

- The relational model and relational databases (DBMS) took over eventually
- Today, everybody seems to talk about Big Data in addition.

When is Data Big? Or Fast?

In 2019, the following server could be rented for NOK900/month:

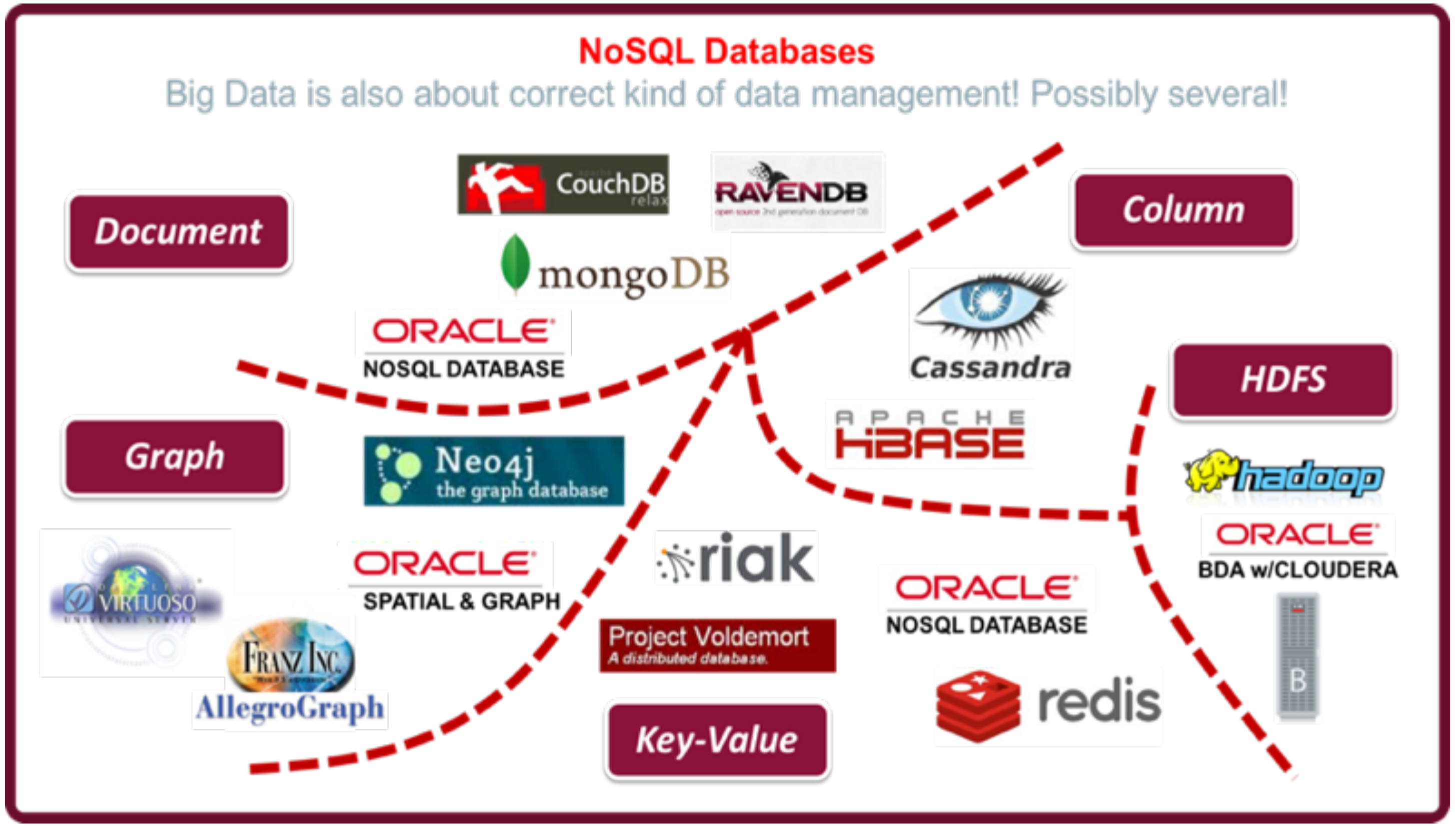
- Cores: 10 x 2.20 GHz (Single 10 Core)
- RAM: 64 GB DDR4 ECC
- HDDs: 2 x 1TB SATA 7.2k RPM

In 2020, the following cloud infrastructure can be «rented» for NOK630/month

- 64 cores
- 512 GB memory
- 2 x 25 GB Gbps network
- Up to 1 PB of block volumes (5TB/month free)
- Can be reduced or increased as needed, giving an average usage for slightly better price and with better performance

NoSQL DBMS – a collective concept

Relational took over, but both hierarchical and network models are still in use, and there are several DB technologies introduced in addition



Databases, Database Systems & Database Managament Systems

- IMPORTANT TO UNDERSTAND THE DIFFERENCE

- Database System (DS) =
  Database Management System (DBMS):
- The engines that «run» the databases
  - Run the queries, optimize, manage transactions, help manipulate the tables or other DBMS structures etc.
- What else?

The reality of database usage

- Concurrent multi-user (many users use the database at the same time)
- Usage is random and needs to be controlled, orchetsrated.
- Machines and storage can crash.
- For distributed databases: The network can be broken, messages can be lost

ACID

- Four very important characteristics
  - Atomicity
  - Consistency
  - Isolation
  - Durability

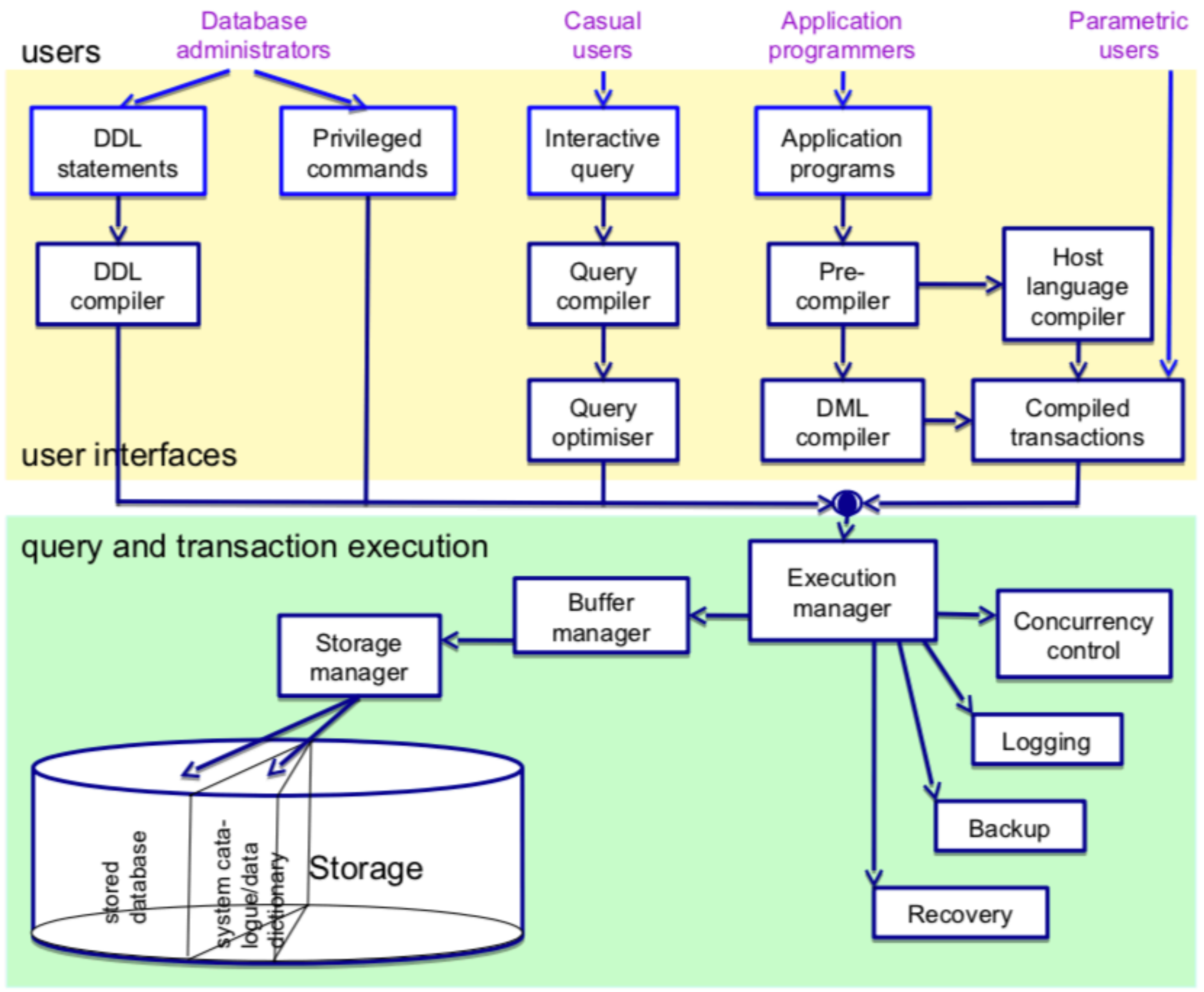
$\left.\begin{array}{l}\text{Atomicity}\\ \text{Consistency}\end{array}\right\}$ A and C are for correctness

$\left.\text{Isolation}\right\}$ I for multi-user concurrency issues

$\left.\text{Durability}\right\}$ D for avoiding data loss

Example DBMS architecture

Typical DBMS Components



users
Database administrators
Casual users
Application programmers
Parametric users
DDL statements
Privileged commands
Interactive query
Application programs
DDL compiler
Query compiler
Pre-compiler
Host language compiler
Query optimiser
DML compiler
Compiled transactions
user interfaces
query and transaction execution
Execution manager
Buffer manager
Storage manager
Concurrency control
Logging
Backup
Recovery
stored database
system cata-logue/data dictionary
Storage