IN3020/4020 – Database Systems
Spring 2021, Week 2.1 (part 1)

Summary of DBMS Architecture

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

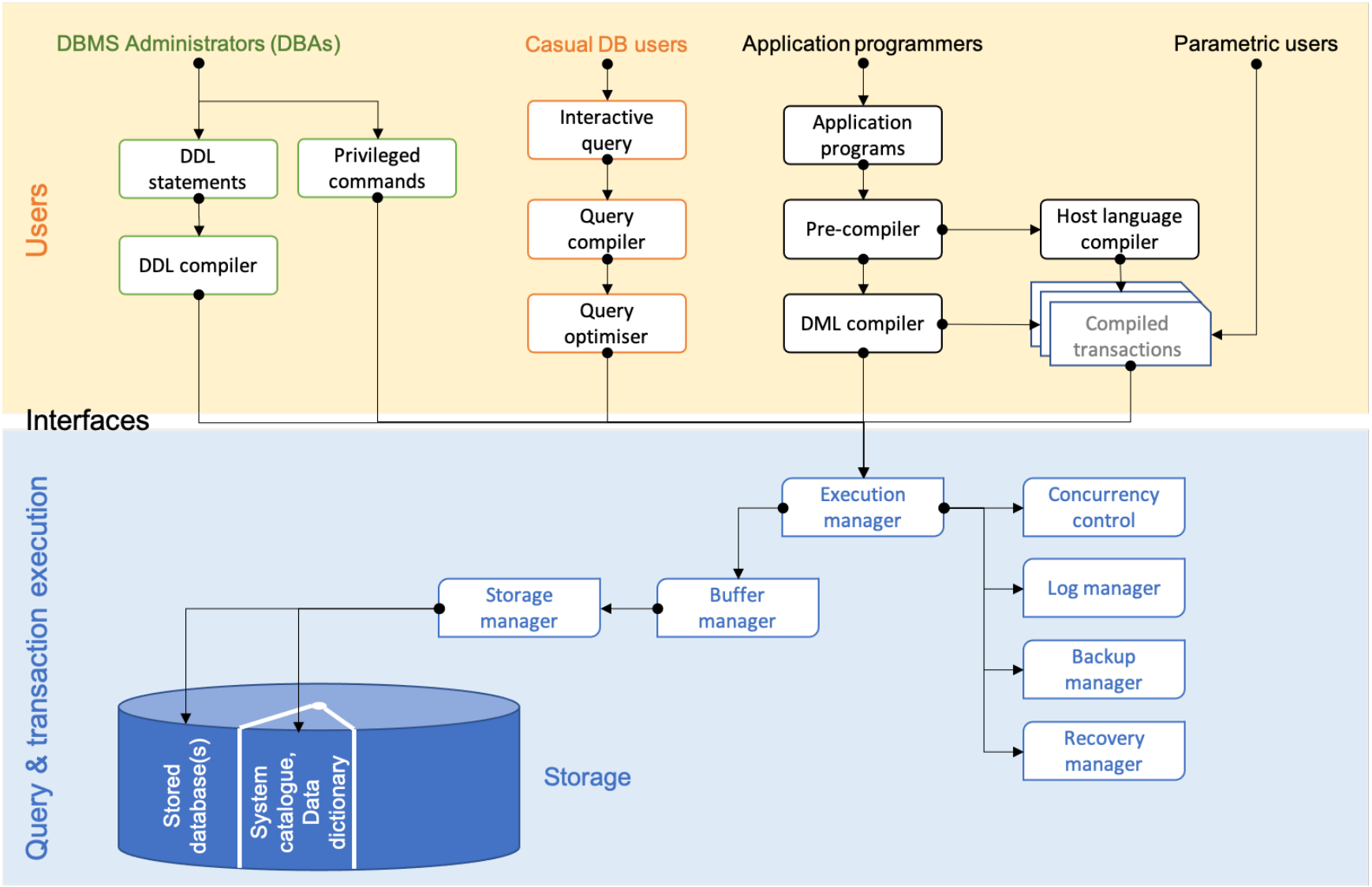
Based upon slides by M. Naci Akkøk, Spring 2020 & E. Thorstensen from Spring 2019

UiO : Institutt for informatikk
Det matematisk-naturvitenskapelige fakultet

Architecture

- Database systems are like operating systems:
  They “operate” databases for us (or they help us operate databases)
- They manage queries, access to databases, maintain/administer databases etc.
- They are made up of components like any other SW system

Example DBMS architecture and its components



Queries

- READING DATA (SELECT):
  Parsing, compilation, optimization, execution
- WRITING DATA (INSERT/UPDATE):
  Parsing, (compilation, optimization), execution
- In addition:
  - Access control
  - DDL (Data Definition Language).

Compilation

- SQL is made for being readable, especially for non-programmers
- Originally: SEQUEL (Structured English Query Language)
- SQL is compiled to relational algebra operations: Filter, Join, Union, Intersection ... (See Wikipedia summary: https://en.wikipedia.org/wiki/Relational_algebra)
- When (in which sequence) the operations are executed has a lot to say about the «cost» of the query. Remember what cost is?

Optimization

- The optimizer-module picks the order (sequence), and uses statistics on tables for choosing:
  - Index usage
  - Algorithms for partial operations

- Trade-off: Better (best) plan vs. more time required to find a better plan.
- Optimal plan will then be sent to execution

Execution

- Reading tuples from the disk or storage medium, and executing operations (in memory)
  - Since the DBMS runs on a regular disk or storage medium, the tuples are stored in regular files
  - Files are just bytes, so the DBMS has its own “file system”.
- Reading from or writing to a regular disk is relatively costly.
  - Though we have memory or memory-like storage technologies these days, I/O is still the costlier part

Files and Blocks

- Tables are stored as files
  - (Preferably) there are **blocks of fixed size** (like 8Kb/block) in the files
  - Fixed size allows us to use an array-structure. We know where the blocks are on disk or storage relative to the file start
- The tuples are in the blocks
  - Blocks are the read/write units.
  - Each block has has meta-data for the tuples and empty space

Buffer Manager

- In addition to managing the disk (or storage medium), **blocks in memory** need to be managed as well
- Buffer manager keeps track of what is in memory (and not). Rest of the system asks for the blocks from the buffer manager
- The buffer manager keeps track of:
  - Whether a block is modified (is «dirty»)
  - Whether the block is in use
  - How long blocks stay in memory (caching)
- LATER: Log & buffer management and caching are used also for replication/synchronization across databases

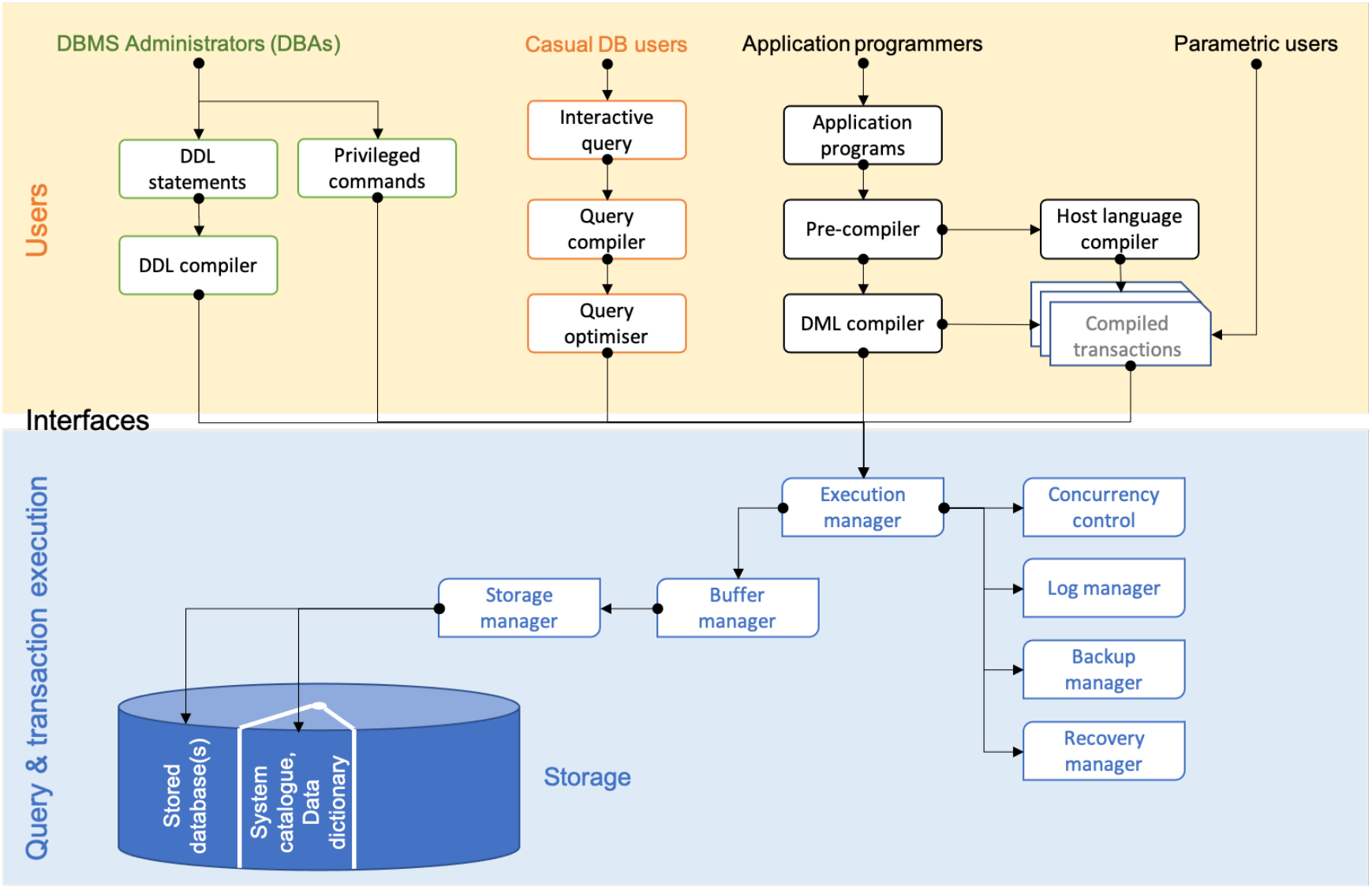
Summary so far

- Queries: Compilation, optimization, execution
- In this course, we will be looking at optimization in more detail, as well as execution (including indexing)
- We shall also look at buffer management, as well as other system level mechanisms like logging, concurrent multi-user management and transaction management
- We will also take a brief look at related backup/recovery & high availability concepts

Reminder of some abbreviations

- A query language is usually made up of three specific sub-languages:
  - DDL: Data Definition Language (CREATE)
  - DML: Data Manipulation Language (DELETE, INSERT, UPDATE)
  - DQL: Data Query Language (SELECT)

Example DBMS architecture and its components



Multi-user Control

- Users read/write concurrently
- That is why the Concurrency Control Module (CC-module) is a relatively important part of a DBMS, which is meant to ensure as little waiting as possible
- Facilitates choosing **isolation level** for transactions
- Interacts (frequently) with the buffer manager for finding out who has modified/read what
- Important and large theme later in this course

Concurrency Control, Main Challenges

- Avoid blocking:
  - Avoid keeping everybody else waiting until a query writing many blocks is finished
  - At the same time, we would prefer not to have to read incomplete updates
- It is not possible to avoid blocking for all types of operations, but it is possible for many

Logging and Recovery

- Recovery from a backup is good and necessary, but what if the server dies in the middle of an update or a critical transaction?
- Money example: Per transfers money to Kari
  - Four operations: Read Per´s account content, update it (write new content), Read Kari´s, write (update) Kari´s.
  - And the server dies after Per´s account is updated but before Kari´s can be updated!
- The system should be capable of either updating Kari´s, or changing Per´s back to pre-transaction state: **The system keeps transaction logs!**

Logging

- Module for logging updates for recovery purposes
- There are several types of logs, each with their trade-offs.

- Also used for replication/synchronization
- Theme for a later lecture in this course

Other Components

- Analytical module (statistics collection) for optimization
- Cleaning/house-keeping processes for the internal file system

- Various mechanisms for extensions, plug-ins for multi-modal DBMS, additional languages/functionality etc., which we will try to talk a bit about, also in «trends and emerging technologies»