HOGESCHOOL ROTTERDAM / CMI

Advanced Databases (Development 6B)

INFDEV026B

Number of study points:	3 ects
Course owners:	Ahmad Omar, Francesco Di Giacomo

Modulebeschrijving

Module name: (Development 6B)	Advanced Databases
Module code:	INFDEV026B
Study points and hours of effort for full-time students:	This module gives 3 ects, in correspondance with 84 hours: • 2 x 8 hours frontal lecture. • the rest is self-study for the theory and practicum.
Examination:	Written examination and practicum (with oral check)
Course structure:	Lectures, self-study, and practicum
Prerequisite knowledge:	None.
Learning tools:	• Book: Database management systems (3rd edition); authors Ramakrishnan and Gehrke • Book: NO SQL Distilled; authors Sadalage and Fowler • Presentations (in pdf): on the GitHub repository `https://github.com/hogeschool/INFDEV026B` • Assignments, to be done at home (pdf): on the GitHub repository `https://github.com/hogeschool/INFDEV026B`
Connected to competences:	• Analysis, design, and realisation of software at level 2
Learning objectives:	At the end of the course, the student can: • **realise** a normalized relational database and implement an application to execute operations on it `RDBMS,NORM` • **describe** the differences between relational and non-relational databases `NONREL` • **use** the map-reduce paradigm to run queries. `NONREL` • **describe** models of concurrency and transactions in a modern DBMS `TRANS`

Content:	• relevant concepts in relational databases normalization • fundamental properties of DBMS's: atomicity, consistency, isolation, and durability (ACID) • concurrency and transactions • Map-reduce paradigm. • BASE vs ACID. • Graph databases.
Course owners:	Ahmad Omar, Francesco Di Giacomo
Date:	15 november 2018

1 General description

Databases are ubiquitous within the field of ICT. Many business needs are centered around the gathering, elaboration, etc. of large amounts of data. This data is crucially connected to real-world operations and thus its constant availability and timely elaboration is of unmissable importance.

This course covers advanced aspects of data processing and elaboration within the different sets of tradeoffs of the precise but limiting ACID framework and the imprecise but forgiving BASE framework.

1.1 Relationship with other teaching units

This course builds upon the basic databases course.

Knowledge acquired through the databases course is also useful for some of the projects. A word of warning though: projects and development courses are largely independent, so some things that a student learns during the development courses are not used in the projects, some things that a student learns during the development courses are indeed used in the projects, but some things done in the projects are learned within the context of the project and not within the development courses.

2 Course program

The course is structured into six lecture units. The lecture units are not necessarily in a one-to-one correspondence with the course weeks.

2.1 Unit 1 - Review

The course starts with a quick review on SQL and RDBMS's:

Topics

- Entities and relationships
- SQL operators

2.2 Unit 2 - Normalization

Theoretical concepts behind normalization of relational models. Normalization algorithms:

Topics

- Redundancy problem.
- Definition of functional dependencies.
- Normal forms definition.

2.3 Unit 3 - Normalization algorithms

In this unit we cover the main normalization techniques.

Topics

- Normalization in 1NF, 2NF, 3NF, BCNF.
- Exercises on normalization.

2.4 Unit 4 - Concurrency

The fourth lecture covers handling of potentially conflicting concurrent query execution in an ACID DBMS:

Topics

- ACID property.
- Serialization
- Locks
- Deadlocks and their prevention

2.5 Unit 5 - NOSQL: Map-Reduce paradigm

The fifth lecture covers map-reduce paradigm:

Topics

- Map function.
- Reduce function.
- Map-Reduce is SELECT-FROM-WHERE
- Idea behind NoSQL
- LINQ and map-reduce in LINQ.

2.6 Unit 6 - Graph databases

The sixth lecture covers a specific example of no-SQL databases, specifically graph databases:

Topics

- Directed vs undirected graphs
- Adjacency list vs matrix
- Algorithms on graphs
- Case study: Neo4J

3 Assessment

The course is tested with one exam: the exam will contain questions about database normalization, transaction scheduling, map-reduce, and graph database. For some questions you need to run .Netcore and Neo4J (latest version). The final grade is determined as follows:

To pass the exam you must have a positive (i.e. ≥ 5.5) grade in both parts.
For additional information about the grading consult your student career on Osiris

Motivation for grade A professional software developer is required to be able to program code which is, at the very least, *correct*.
In order to produce correct code, we expect students to show: *i*) a foundation of knowledge about how a programming language actually works in connection with a simplified concrete model of a computer; *ii*) fluency when actually writing the code.
The quality of the programmer is ultimately determined by his actual code-writing skills. The quick oral check ensures that each student is able to show that his work is his own and that he has adequate understanding of its mechanisms. The theoretical exam tests that the required foundation of knowledge about databases is also present.

3.1 Theoretical examination

The general shape of a theoretical exam for the course is made up of a series of highly structured open questions. In each exam the content of the questions will change, but the structure of the questions will remain the same. For the structure of the theoretical exam, see the appendix.

3.2 Practical examination

The practical assignments are formative and will not be graded. Students are strongly advised to do the assignments to prepare for the pracitcal assessment. The theory part will contain questions about database normalization and transaction concurrency. The practical assessment will contain programming exercises about map-reduce and graph databases. The students must bring a working laptop with them with the necessary tools to run code in C# and with Neo4j installed.

Bijlage 1: Toetsmatrijs

Learning goals	Dublin descriptors
RDBMS	1, 2, 3, 4, 5
NORM	1, 2, 5
TRANS	1, 4
NONREL	1, 2, 5

Dublin-descriptors:

1. Knowledge and understanding
2. Applying knowledge and understanding
3. Making judgements
4. Communication
5. Learning skills

Exam structure

Theory

What follows is the general structure of a DEV5 exam. You can find a concrete exam sample in the course page on natschool.

Associated learning goals: `NORM`, `RDBMS`.

3.2.0.1 Question I: Normalization

Given a relational schema and defined functional dependencies, provide a normalized version in BCNF

Associated learning goals: `TRANS`, `RDBMS`.

3.2.0.2 Question II: Transaction management and concurrency

Given the following N-queries, which are run in parallel, show a strict 2PL scheduling. Detect deadlocks in a given transaction executions and identify how to break them.

Practice

Associated learning goals: `NONREL`.

3.2.0.3 Question I: Map-reduce

Implement the given queries with the map-reduce paradigm.

Associated learning goals: `NONREL`.

3.2.0.4 Question II: NoSQL databases en Graph Theory

Transform a given entity-relationship diagram into a graph model and implement the given queries in Cypher