Revision of the Database Courses Offered by the School of Informatics

The School of Informatics (SoI) currently offers the following database courses:

- **Database Systems (INFR10070, 20 Credits, Level 10), organised by Paolo Guagliardo:** this course is an introduction to the principles underlying the design and implementation of database management systems.
- **Advanced Databases (INFR11011, 10 Credits, Level 11), organised by Milos Nikolic:** this course covers advanced aspects of database systems, in particular, the data structures and algorithms underlying modern database management systems.
- **Advanced Topics in Foundations of Databases (INFR11122, 20 Credits, Level 11), organised by Andreas Pieris:** this course aims to expose postgraduate students to current research and developments in connection with the principles of data management, and prepare them for conducting research in this area.

The above courses have not been revised for quite sometime in a coordinated way that

1. takes into account their interconnections and overlaps,
2. considers possible links with other courses, especially data science courses, offered by the SoI,
3. covers the state-of-the-art of the field, and
4. reflects the real needs of our students, in particular, postgraduate students.

As a result, we have observed the following issues:

- The postgraduate courses (INFR11011 and INFR11122) are in some aspects outdated, or do not meet the expectations of the students concerning the covered material.
- The existing database courses run in isolation, without their interconnections, as well as their connections with other courses, especially data science courses, offered by the SoI, being apparent to the students.
- An applied course, dedicated to modern database technologies, that prepares our postgraduate students for the data science job market is missing.

The ultimate goal is to rectify this state of affairs. In particular, the main objective is to offer a coherent and comprehensive curriculum that covers system and foundational aspects of databases both at an introductory and advanced level. To achieve this, we are going to

1. carefully revise the syllabus of the database courses INFR10070, INFR11011 and INFR11122 along the four dimensions given above,
2. change the name of the database courses INFR10070, INFR11011 and INFR11122 to better reflect their nature,
3. extend the postgraduate course INFR11011 from a 10-credit to a 20-credit course,
4. revive and significantly revise the postgraduate course Applied Databases (INFR11015, 10 Credits, Level 11), which was running until 2016/2017.

Introduction to Databases (Revised Version of Database Systems (INFR10070))

This introductory course is quite stable and healthy in its current form, which is reflected by the large number of registered students and their positive feedback. The main issue that we have observed is that the current syllabus contains too much material to be covered in a reasonable level of detail during the semester. As a result, there are a couple of advanced topics that, despite the fact that are included in the course description, are not covered due to lack of time. In addition, those topics are too advanced for an introductory course to databases. More precisely, the two topics that can be removed from the syllabus without affecting the overall learning outcomes the course aims to deliver are:

1. Semi-structured data, which will be covered in the postgraduate courses Principles of Data Management (the revised version of INFR11122) and Applied Databases (the revised version of INFR11015).
2. Data warehousing, which will be covered in the postgraduate course Advanced Database Systems (the revised version of INFR11011).

The proposed changes in a nutshell:

- Remove semi-structured data and data warehousing from the syllabus.
- Change the name of the course to “Introduction to Databases”, which better reflects its nature as an introductory course.

Summary

Data is one of the most important assets of any enterprise and plays a central role in many aspects of everyday life, from healthcare, to education, to commerce. In order to be turned into meaningful information that enables and supports decision making, data must be stored, maintained, processed and analysed. Database management systems are complex programs that allow their users to perform these tasks in an efficient and reliable way.

Database systems encompass many areas of Computer Science, from formal logic to programming languages, from operating systems to algorithms and data structures. This course is an introduction to the principles underlying the design and implementation of relational databases and database management systems. It will cover in detail the main language for databases, SQL, which is an international standard supported by virtually all systems on the market today. It will also cover the theoretical query languages on which SQL’s core is based, namely relational algebra and relational calculus. Other important topics covered during the course include normal forms, transaction processing, concurrency control, query optimisation, and incomplete data.

Course Description

- Overview of the relational model and database management systems.
- How to create, update and query a relational database with SQL .
- Theoretical query languages: relational algebra and relational calculus.

- Analytical queries: multisets, grouping and aggregation.
- Database design: constraints and normal forms.
- Advanced SQL: nested queries, views, constraints, triggers.
- Deductive databases: Datalog and recursive queries.
- Incomplete data: null values and certain answers.
- Transaction management: concurrent schedules, conflict-serializability, locking.
- Database access from applications - using SQL in a host programming language.
- Basics of storage and indexing: file/page/record formats, B+ trees, static hashing.
- Basics of query evaluation and optimisation: join strategies and query plans.

Other Requirements

Most of the necessary background will be introduced during the course, but some familiarity with predicate logic is desirable.

There are no specific programming requirements. Familiarity with the Unix command line is a plus, though not strictly required.

Assessment

Written Exam 100%, Coursework 0%, Practical Exam 0%

Additional Information (Assessment)

Coursework is formative (assessed for learning, with feedback) and consists in writing SQL queries to a given specification.

Learning Outcomes

On completion of this course, the student will be able to:

1. Create and modify a relational database using standard software tools available on the market.
2. Compare strengths and weaknesses of different database designs.
3. Process and analyse data by means of complex SQL statements.
4. Formulate and manipulate queries in both declarative and procedural database languages.
5. Reason about the correctness and consistency of concurrent database interactions among multiple users.
6. Understand how queries are optimised and executed in relation with how data is stored and organised.

Advanced Database Systems (Revised Version of Advanced Databases (INFR11011))

The two main objectives are the following:

1. **Bring the course content up-to-date with current trends in database systems**

   The Advanced Databases course has not seen significant changes for at least 10 years. The course content is still entirely based on the classical database textbook by Ramakrishnan and Gehrke from 2003. Although this textbook remains a valuable reference nowadays, the data management landscape has changed significantly in the meantime: we have seen new database architectures such as column stores and in-memory database systems, new query processing techniques such as query compilation, new indexing structures, new concurrency control protocols, and recent explosions of cloud databases, scientific databases, and systems combining databases and machine learning. Studying these new topics would allow students to get a better understanding of how modern database technologies are used in industry and research nowadays. This is a valuable learning outcome in our opinion.

2. **Cover cutting-edge research topics**

   We believe that postgraduate students should be exposed not just to classical textbook material, but also to recent research literature to help them develop critical thinking and research skills. This revised course will include several lectures with discussions of recent research papers, which are normally not covered in any textbook yet. Examples of such topics include cloud databases, systems for in-database machine learning, scientific (array) databases, etc.

The proposed changes in a nutshell:

- Update the content of the course to better reflect the modern trends in database management systems.
- Extend the course from a 10-credit to a 20-credit course. Covering the aforementioned new topics along with the existing material would not be possible within the scope of a 10-credit course.
- Change the name of the course to “Advanced Database Systems”. This name better reflects the system-oriented nature of the course, and would hopefully mitigate confusions seen in previous years among students, who often think of “Advanced Databases” as being an advanced SQL course, which it is not.

Summary

Database management systems are at the core of computer applications that need to store, manipulate, and query data. This course takes a deep dive into how modern database systems function internally, from studying their high-level design to understanding the

underlying data structures and algorithms used for efficient data processing. The course covers a range of data management techniques from both commercial systems and cutting-edge research literature, enabling students to apply these techniques to other fields of computer science. The covered topics include database architecture, storage manager, data models (row, columnar), indexing (tree-based, hash tables), transaction processing (ACID, concurrency control), crash recovery, parallel architectures (multi-core, distributed), data warehousing and decision support (OLAP, materialised views).

Course Description

- Database systems architectures, row stores and column stores, OLTP vs. OLAP, in-memory database systems.
- Storage: secondary-storage devices.
- Indexing: tree-based and hash-based techniques, multi-dimensional indexing, learning indices from data.
- Write-optimised data structures: LSM trees, LSM hash tables, B^eps trees.
- Query evaluation: sorting and join processing, selection, projection, aggregation, query compilation.
- Query optimisation: cardinality estimation, cost-based query optimisation, dynamic programming, rule-based optimisation, learning query plans.
- Transaction management: ACID properties, concurrency control, locking and multi-version protocols, crash recovery.
- Distributed database systems: parallel query evaluation, distributed transaction processing.
- Data warehousing and decision support: OLAP, materialised views, incremental view maintenance.
- Stream processing systems, data streaming algorithms.
- Scientific (array) databases, cloud databases, database systems for machine learning.

Other Requirements

The course assumes an understanding of algorithms and data structures (e.g., quick sort, merge sort, binary trees, hash tables, big-O notation).

A good level of programming is assumed and will not be covered during lectures. The coursework will involve implementing different components (e.g., data structures, query processing algorithms) of a typical database system.

Assessment

Written Exam 70%, Coursework 30%, Practical Exam 0%

Additional Information (Assessment)

The coursework consists of two programming assignments where students will design and implement components of a database management system, experimentally evaluate their work, and write a report on their findings.

Learning Outcomes

On completion of this course, the student will be able to:

1. Understand how database management systems function internally.
2. Interpret and comparatively criticise database systems architectures.
3. Implement major components of a database management system and analyse their performance.
4. Understand, analyse, and compare the fundamental query evaluation and concurrency control algorithms.
5. Identify trade-offs among database systems techniques and contrast distributed/parallel techniques for OLTP and OLAP workloads.

Principles of Data Management (Revised Version of Advanced Topics in Foundations of Databases (INFR11122))

The two main objectives are the following:

1. **Make the course content accessible to postgraduate students**

   The current version of the Advanced Topics in Foundations of Databases course aims to prepare students for conducting research on the foundations of data management. To this end, several cutting-edge topics are covered such as approximation of queries, semantic optimisation of queries under constraints, querying tree-structured data using Monadic Second-Order (MSO) logic and alternating tree automata, expressive graph query languages, and advanced aspects of knowledge-enriched data. However, the majority of our postgraduate students do not have the background on computational logic, complexity and computability theory for following the covered topics. Thus, the course in its current form does not serve its purpose, which is confirmed by the low number of registered students.

2. **Demonstrate the importance of studying real-life problems in a mathematically rigorous way**

   This second objective is not completely unrelated with the first one. Since, as discussed above, the course covers cutting-edge research topics, which are technically very challenging, it is difficult for the students to relate the content of the course with real-life database concepts. Moreover, it is extremely difficult for the students to understand the implications of the foundational studies covered in the course for real-life problems and applications.

The proposed changes in a nutshell:

- Significantly simplify the technical content of the course to help the students to understand the foundations of data management, and realise the relevance of studying real-life database problems from a mathematical point of view.
- Make the course self-contained by introducing during the lectures all the necessary concepts and technical tools from logic and complexity theory.
- Change the name of the course to “Principles of Data Management”, which better reflects the nature of the revised course.

Summary

Data is everywhere, coming in different shapes and volumes, and needs to be stored and managed using appropriate data management technologies. The basic software package that supports the management of data is called a database management system (DBMS). The main goal of this course is to explain some of the underlying principles and characteristics of DBMSs. More precisely, this course will explain how real-life concepts (such as a database and a query) and phenomena (such as incompleteness and

inconsistency of data), can be abstracted from their physical implementation and formalised using tools coming from other areas such as computational logic and graph theory. This will pave the way towards the study of query evaluation, that is, the central task of extracting meaningful information from (possibly incomplete and inconsistent) data by means of queries, following a mathematically rigorous approach. This analysis will expose the source of complexity in evaluating a query over a database, which in turn provides ideas and tools on how to devise more efficient query evaluation algorithms.

Course Description

- Relational model - the classics: data model, relational algebra, relational calculus (first-order queries), first-order query evaluation, static analysis of first-order queries (satisfiability and containment).
- Conjunctive queries (CQs): syntax and semantics (via the notion of homomorphism), CQ evaluation, static analysis of CQs (satisfiability, containment and the Homomorphism Theorem), minimization of CQs.
- Fast conjunctive query evaluation: acyclic CQs, evaluating acyclic CQs (Yannakaki's algorithm), semantically acyclic CQs and their evaluation, size bounds for joins (AGM bound), worst-case optimal join algorithms.
- Adding recursion - Datalog: inexpressibility of recursive queries, syntax and semantics of Datalog, Datalog query evaluation, static analysis of Datalog queries, Datalog vs. first-order queries.
- Uncertainty - reasoning over possible worlds: incomplete databases, inconsistent databases, knowledge-enriched databases.
- Tree-structured data: data model, tree pattern queries (syntax and semantics), tree pattern query evaluation, minimization of tree pattern queries.
- Graph-structured data: data model, basic query languages (regular path queries and extensions thereof), query evaluation, static analysis.

Other Requirements

While there are no formal prerequisites, it is recommended that students taking this course have passed an introductory course in Databases such as the undergraduate course Introduction to Databases (the revised version of Database Systems (INFR10070)). It is also assumed that students have a basic familiarity with complexity theory (standard complexity classes such as PTIME and NP, and the notion of completeness). In any case, this course is self-contained, and all the necessary tools will be properly introduced during the lectures.

Assessment

Written Exam 0%, Coursework 100%, Practical Exam 0%

Additional Information (Assessment)

The coursework consists of three essays (each worth 15%), a final project (worth 40%), and an in-class presentation of the final project (worth 15%).

Essay guidelines. The course is split into seven topics (see the course description above where each bullet corresponds to a topic). For each topic, a list of conference/journal papers will be given at the beginning of the semester. For each essay, the students should pick *one* paper (in particular, for Essay 1 a paper from topics 1-3, for Essay 2 a paper from topics 4-5, and for Essay 3 a paper from topics 6-7), and present:

- A summary of the paper.
- Analysis and critical thoughts; e.g., a criticism of the paper, ideas for extending its results, or analysis of follow-up papers that show how the ideas of the paper under review have influenced the field.

Crucially, the essay must be understood by someone who has *not* read the paper. Of course, in reality, it will be marked by someone who has read the paper, but still this is an important criterion that will be used in marking.

Project guidelines. Projects follow the same path as essays by choosing a paper, not previously studied, from topics 1-7, but a *new contribution* done by the student is also required. This new contribution could be, for example:

- An implementation of a theoretical algorithm with performance analysis.
- An extension of some of the results to cover new cases.
- An improvement for an existing solution, perhaps under some restrictions.

The above list is by no means exhaustive. It is up to the student to decide what will be the new contribution, which is an important criterion that will be used in marking.

Learning Outcomes

On completion of this course, the student will be able to:

1. Abstract relational, tree-structured, and graph-structured data, and queries over this data, from their physical implementation, and formalise them in a rigorous way.
2. Analyse the complexity of querying relational, tree-structured, and graph-structured data, isolate the source of complexity of query evaluation, and understand the key ideas that lead to more efficient query evaluation algorithms.
3. Understand the semantics of recursive queries, in particular, Datalog queries, analyse the complexity of evaluating Datalog queries, and model real-life queries and problems in a declarative way using Datalog.
4. Understand how real-life phenomena concerning data, in particular, incompleteness, inconsistency, and the existence of implicit knowledge of the underlying domain, can be formalised in a rigorous way.
5. Analyse the complexity of querying incomplete, inconsistent, and knowledge-enriched data, and understand the reasons that lead to intractability.
6. Read, analyse, and summarize scientific papers.

Revised Version of Applied Databases (INFR11015)

The Applied Databases course was running until 2016/2017 by Sebastian Maneth, who is now a Professor at the University of Bremen. The goal is to revive and significantly revise this course, which will be offered in 2020/2021, and organised by Yang Cao who is currently a Chancellor's Fellow. The two main objectives are the following:

1. **Bring the course content up-to-date**

   Many topics that are covered in the 2016/2017 version of the course have to be either updated in order to reflect the latest advances in the area, or completely removed from the curriculum. Here are some examples: (a) ER diagrams are not the go-to method for designing a database schema anymore, given the diverse range of application requirements and database architectures; (b) index design needs also to include methods based on machine learning (ML) to reflect the latest developments in database auto-tuning using ML; and (c) graph datasets are now preferably processed using native graph computing systems and graph databases, instead of mapping them to relational databases and using traditional database systems.

2. **Cover emerging topics and prepare students for the data science job market**

   Databases is a fast expanding area that covers a wide range of emerging topics that go beyond SQL and traditional database systems. However, several of those topics, which are now becoming mainstream in database industry and research, e.g., graph databases, spatial and temporal databases, and database systems for ML, are not covered in the 2016/2017 version of the course. Including these new topics would allow postgraduate students to become familiar with recent database developments, and prepare themselves for the data science job market.

The proposed changes in a nutshell:

- Update the existing content of the 2016/2017 version of the course to better reflect the latest trends in the area of databases.
- Include emerging topics such as distributed and cloud databases, parallel computing frameworks, graph databases and analytics, spatial and temporal databases, access controls and privacy, and databases for ML, that would help the students to prepare themselves for the data science job market.

Summary

Databases, as one of the core building blocks of data science, is a fast expanding area that encompass a wide range of emerging topics that go beyond SQL and traditional database systems. The goal of the course is to familiarise students with such emerging topics, and prepare them for the data science job market. The course covers a range of recent developments in data management, providing a general toolkit and methodology of how modern database technologies can be effectively used in diverse data-driven applications.

The covered topics include cloud and distributed databases, graph databases and analytics, parallel computing frameworks, spatial and temporal databases, and database systems for machine learning. The course will also provide additional references for those students who want to pursue further studies in areas related to data management and data science.

Course Description

- Data models: relational data, key-value stores, trees, graphs, triple stores, document stores, arrays.
- Relational databases: SQL, PL/SQL, database connectors, database (auto-)tuning, database administration (access controls, permissions, privacy and security).
- Parallel, distributed and cloud databases: architectures, parallel/distributed query processing, data partition, consistency, transactions, and example systems including Greenplum (parallel database), Snowflake and Redshift (cloud database).
- NoSQL databases: why NoSQL, key-value stores, column-family stores, document stores.
- Large scale parallel computing frameworks: MapReduce, dataflow, HPC, BSP/AP/GAS.
- Graph databases: XML (DTD, XPath, XSLT), RDF databases (RDF model, SPARQL, RDFS), graph databases (property graph model, languages: Cypher/G-core/GSQL), graph analytics (algorithms and systems).
- Spatial and temporal databases: data models and languages, query processing, spatial and temporal index, systems.
- Databases for machine learning (ML): data wrangling, in-database learning, declarative ML, database systems for managing ML models.

Other Requirements

While there are no formal prerequisites, it is assumed that students taking this course are familiar with basic Linux system scripting and programming, have a basic understanding of relational databases (in particular, have passed an introductory course in Databases such as the undergraduate course Introduction to Databases, the revised version of Database Systems (INFR10070)), and have experience in algorithm design, analysis and implementation. In any case, this course will be self-contained.

Assessment

Written Exam 0%, Coursework 100%, Practical Exam 0%

Additional Information (Assessment)

Summative assessment. The coursework consists of one experimental assignment (40%) and one report (60%):

- For the experimental assignment, the students will be asked to carry out an experimental study and write a report based on it. It may require some lightweight

scripting for data preparation, configuration, basic algorithm implementation, and analysis of results. This is to allow students to get hands-on experience in deploying, using, and tuning modern database systems.

- For the report, the students will be asked to write a survey on one of the topics covered during the course. This survey will be based on (but not limited to) a list of references that will be provided at the beginning of the semester. Below is what the students are expected to do:

    1. Pick a set of systems/algorithms/frameworks on a relevant topic, and discuss their main objective.
    2. Develop a set of criteria for evaluating the systems/algorithms/frameworks, i.e., a set of key factors for developing a concrete technique to achieve the main objective. Justify why these criteria are indeed important.
    3. Assess the systems/algorithms/frameworks based on the selected criteria, comment on their limitations, and suggest possible improvements.

Formative assessment. The students will be asked to construct a "complete" list of open-source systems (or algorithm implementations) related to the course, and build a personal knowledge base for the available database systems for data science. The students are expected to try out the systems and understand what they offer and how they perform.

Learning Outcomes

On completion of this course, the student will be able to:

1. Understand how different types of datasets are modelled, stored, and queried.
2. Tune database systems to meet the performance, scalability, and privacy and security requirements imposed by database applications.
3. Understand and use large scale parallel frameworks to process datasets, and compare them with conventional database systems.
4. Design, analyse, and implement search algorithms over graph data based on various search criteria.
5. Know when to use which system/method/algorithm for a given application and justify the choice.
6. Write a survey of research papers and systems.