Syllabus
Course Program

NO SQL databases

Specialty
122 – Computer science

Institute
Institute of Computer Modeling, Applied Physics and Mathematics

Educational program
Computer science. Modeling, design, and computer graphics

Department
Computer modeling of processes and systems

Level of education
Master's level

Course type
Professional training, Mandatory

Semester
1

Language of instruction
English

Lecturers and course developers

Tatarinova Oksana

oksana.tatarinova@khpi.edu.ua
Associate professor, PhD in Engineering Science
Associate Professor at the Department of Computer Modeling of Processes and Systems

Specialist in mathematical and computer modeling of nonlinear processes. Author of over 60 scientific articles and conference papers, co-author of certificates of authorship, monographs, and textbooks.
More about the lecturer on the department's website

Hroshevyi Mykhailo

mykhailo.hroshevyi@kpi.edu.ua
Master of Computer Science, postgraduate at NTU "KhPI"

Middle Java Software Development Engineer at "Akvelon Ukraine".
Has 4 years of experience working on commercial software development projects.

Palamarchuk Pavlo

pavlo.palamarchuk@infiz.khpi.edu.ua
PhD Student, Department of Computer Modeling of Processes and Systems

Currently working on a dissertation on the topic 'Development of a methodology for stress-strain calculations and wear of structural elements'. The author has 3 scientific publications.

General information

Summary

Familiarization with contemporary views on database design methodology; identification and resolution of key issues arising in the development of database management systems (DBMS); study of the theoretical principles underlying modern DBMS.

.

Course objectives and goals

Attainment of fundamental knowledge and the acquisition of skills and competencies in the domain of database design, familiarization with contemporary perspectives on select chapters of database design theory, assimilation of the central principles of database design, and the cultivation of database design skills and competencies.

Format of classes

Lectures, laboratory classes, consultations, self-study. Final control in the form of an exam.

Competencies

1. The ability to apply knowledge in practical situations.
2. The ability to develop, describe, analyze, and optimize architectural solutions for information and computer systems of various purposes.
3. The capability to develop software in accordance with formulated requirements, taking into account available resources and constraints.
4. The ability to develop and administer databases and knowledge bases.
5. The ability to initiate, plan, and implement processes for the development of information and computer systems and software, including its development, analysis, testing, system integration, implementation, and support.

Learning outcomes

1. Possess specialized conceptual knowledge that includes modern scientific achievements in the field of computer science and forms the basis for original thinking and research, critical understanding of problems in the field of computer science, and at the intersection of knowledge domains.
2. Develop a conceptual model of an information or computer system.
3. Design and maintain databases and knowledge bases.

Student workload

The total volume of the course is 150 hours (5 ECTS credits): lectures - 32 hours, laboratory classes - 32 hours, self-study - 86 hours.

Course prerequisites

Knowledge of the fundamentals of working with traditional relational databases (SQL). Understanding of database theory foundations, including data models and normalization. Programming skills in one or more languages commonly used for database work.

Features of the course, teaching and learning methods, and technologies

The student must have a computer or laptop with the Linux, macOS or Windows operating system installed.

Program of the course

Topics of the lectures

Topic 1. Introduction. Differences between relational and NoSQL databases. Types of NoSQL databases.

Changed requirements for storage systems due to transition to the cloud infrastructure. Relational database management systems scaling. ACID principle and CAP theorem. Pros and cons of using NoSQL databases as a distributed storage. Types of NoSQL databases.

Topic 2. Key-value databases on the example of Redis.

Key-value and in-memory storage systems. Persistence in Redis: Snapshots vs. Append Only File. Data replication and sharding. Modular infrastructure of Redis: Redis Stack. What cache is and why modern applications need it? Why Redis is good for storing cache? Can we use Redis as application's database?

Topic 3. Data types and operations with them in Redis.

Basic data types and operations with them: Strings, Lists, Sets, Hashes, Sorted Sets, Bitmaps, Bitfields. What messaging is and why applications may need it? What compact prefix tree is? Stream data type: how we can implement an event broker using Redis?

Topic 4. Additional features of Redis and Redis Stack.

Redis is modular: Redis Stack. What JSON is and why we use it? How to work with JSON using JSONPATH? Storing JSON in Redis. Solving filtering with indexes, reverse indexes. Redis Search: indexes, queries, text searches, aggregations. Redis Time Series: streaming data from sensors. What probabilistic algorithms are and why are they used? Redis Bloom: Bloom Filter, Top-K, HyperLogLog.

Topic 5. Document-oriented databases on the example of MongoDB.

Document-oriented storage systems. How applications benefit from storing data in the form of JSON. Data replication and sharding in MongoDB. MongoDB cluster: Router, Config Server, Replica Set. How to choose sharding key? BSON and data types. Database, collection, and document. How applications connect to MongoDB cluster?

Topic 6. Manipulating data in MongoDB. Data consistency and schema validation. Indexes.

CRUD operations with MongoDB. Searches: expressions and operators. Search by nested objects/collections. Read Preference, Read Concerns, Write Concerns. JSON schema and validators: how to enforce schema in MongoDB? Indexes: sorting or hashing. Compound indexes. Text searches using text indexes.

Topic 7. MongoDB aggregation engine.

Why applications aggregate data? Why aggregations on application side may be slow. How to aggregate data with MongoDB: aggregation pipeline. Expressions, operators, and system variables. List of available operators to work with strings, lists, dates, etc. Nesting expressions. List of available aggregation pipeline stages: match, unwind, group, project, etc. Lookup as ana alternative to joins in relational databases.

Topic 8. Additional MongoDB functionality.

Time series: configuration, writing and aggregation. Geospatial: storing different geometries in MongoDB. 2dsphere index: querying geometry. Bulk writes, order of operations. Transactions in MongoDB. Change streams: implementing CDC or/and messaging.

Topic 9. Wide-column databases on the example of Apache Cassandra.

What wide-column database is? Partition key: how data is partitioned. Data redundancy and replication in Cassandra. Deploying Cassandra across different datacenters/regions. How cluster is working: node roles, gossip protocol. Consistency levels: what are they and how to configure?

Topic 10. Data modeling with Apache Cassandra.

Cassandra: good old tables working in a different way. Working with keyspaces. Building primary key: how to choose partition key, how to choose clustering key? Denormalizing data: there are no joins in Cassandra. Supported data types.

Topic 11. Manipulating data in Cassandra: CQL (DDL and DML)

CQL: Cassandra Query Language. How CQL is different from SQL? DDL: working with keyspaces and tables. DML: working with data. SELECT, INSERT, UPDATE, and DELETE: syntax and limitations. Working with indexes in Cassandra. SASI index and custom indexes.

Topic 12. Additional Cassandra functionality.

List of available functions. Using Java to extend Cassandra. Creating custom functions and aggregations. Creating and using custom triggers. Replications strategies: different types and their configurations.

Topic 13. Graph-oriented databases on the example of Neo4j.

What graph is and why is it useful to store data as a graph sometimes? Clustering Neo4J: data replication, federation, and sharding. Neo4J Fabric. What is supported in Community Edition? What AuraDB is?

Topic 14. Data modelling with Neo4j.

Understanding graphs: vertices and edges. Graphs instead of relations: converting relational model to graph model. Writing queries first.

Topic 15. Manipulating data in Neo4j. Cypher query language.

Cypher: Neo4J query language. Querying data with Cypher. Filtering queries. Create, update, and delete operations. Aggregations in Cypher. Subqueries.

Topic 16. Conclusion. How to choose a database for an application.

Other types and implementations of NoSQL databases. Choosing between relational and NoSQL databases. Choosing NoSQL database. Database as a service. Choosing between cloud and self-managed storages.

Topics of the workshops

Not included in the curriculum

Topics of the laboratory classes

Laboratory class 1. Installation of Docker Desktop.

Enabling virtualization and installing Docker Desktop.

Laboratory class 2. Basic data manipulation in Redis.

Running Redis container in Docker, executing basic CRUD operations using redis-cli.

Laboratory class 3. Working with indexes and searches in Redis.

Saving JSON data, creating index, and querying data using redis-cli.

Laboratory class 4. Performing actions with various data types in Redis.

Working with Time Series, Top-K, and HyperLogLog using redis-cli.

Laboratory class 5. Basic data manipulation in MongoDB.

Running MongoDB container in Docker, executing basic CRUD operations using mongosh.

Laboratory class 6. Using schema validators and indexes in MongoDB.

Setting up schema validator, inserting data, creating index, and querying data using mongosh.

Laboratory class 7. Using aggregations in MongoDB.

Saving data to collection and writing aggregation pipeline using mongosh.

Laboratory class 8. Working with geospatial and 2dsphereindex in MongoDB.

Saving geometries to collection, creating 2dsphere index, and querying data using mongosh.

Laboratory class 9. Basic data definition in Cassandra.

Running Cassandra container in Docker, creating keyspace and table, altering table using cqlsh.

Laboratory class 10. Data modelling for Cassandra.

Creating database schema for Cassandra having business data model.

Laboratory class 11. Basic data manipulation in Cassandra.

Executing basic CRUD operations on a table using cqlsh.

Laboratory class 12. Working with functions in Cassandra.

Designing and writing basic custom aggregation using cqlsh.

Laboratory class 13. Creating AuraDB instance.

Creating AuraDB instance, connecting to it and executing basic commands using cypher-shell.

Laboratory class 14. Data modelling for Neo4j.

Creating database schema for Neo4j having business data model.

Laboratory class 15. Basic data manipulation in Neo4j.

Executing basic CRUD operations on a graph using cypher-shell.

Laboratory class 16. Database Selection for the Project.

Choosing best suited NoSQL database for different example projects.

Self-study

1. Study of documentation and technical literature on a specific NoSQL database that is not included in the main course program.
2. Independent analysis and experiments with different types of indexes in NoSQL databases to optimize queries.
3. Investigation of encryption methods and data security measures in NoSQL databases.
4. Setting up database clusters locally, configuring replication and sharding in distributed NoSQL systems.
5. Execution of a personal project, which involves designing, developing, and optimizing a database based on NoSQL technologies according to defined business requirements and usage scenarios.

.

Course materials and recommended reading

1. Guy Harrison. Next Generation Databases: NoSQL, NewSQL, and Big Data. 2015. – 235 c.
2. M. Tamer Özsu, Patrick Valduriez. Principles of Distributed Database Systems. Fourth Edition, 2021. – 673
3. Perkins L., Redmond E., Wilson J. Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement, 2nd edition. 2018. – 354 c.

Assessment and grading

Criteria for assessment of student performance, and the final score structure

The processing of questions assigned for self-study is evaluated by the lecturer during the exam at the end of the academic semester.
Points are awarded according to the following ratio:
• exam: 40% of the semester grade
• laboratory works: 60% of the semester grade).

Grading scale

Total points	National	ECTS
90–100	Excellent	A
82–89	Good	B
75–81	Good	C
64–74	Satisfactory	D
60–63	Satisfactory	E
35–59	Unsatisfactory (requires additional learning)	FX
1–34	Unsatisfactory (requires repetition of the course)	F

Norms of academic integrity and course policy

The student must adhere to the Code of Ethics of Academic Relations and Integrity of NTU "KhPI": to demonstrate discipline, good manners, kindness, honesty, and responsibility. Conflict situations should be openly discussed in academic groups with a lecturer, and if it is impossible to resolve the conflict, they should be brought to the attention of the Institute's management.
Regulatory and legal documents related to the implementation of the principles of academic integrity at NTU "KhPI" are available on the website: http://blogs.kpi.kharkov.ua/v2/nv/akademichna-dobrochesnist/

Approval

Approved by

Head of the department
Dmytro BRESLAVSKY

Guarantor of the educational program
Oleksii VODKA