

Embedding parallel computing readiness in database education: quasi-experimental evidence from Kazakhstan

Saule Zhumagulova¹, Ainur Tokzhigitova², Gulnur Abildaeva³, Zhana-Gul Yessendauletova¹, Leilya Fazylova¹

¹Department of Computer Science, Faculty of Mathematics and Information Technology, Karaganda National Research University named after E.A. Buketov, Karaganda, Kazakhstan

²Department of Information Technology, Faculty of Computer Science, Toraighyrov University, Pavlodar, Kazakhstan

³Department of Information and Computing Systems, Faculty of Information Technology, Saginov Technical University, Karaganda, Kazakhstan

Article Info

Article history:

Received Jan 16, 2026

Revised Jul 19, 2026

Accepted Aug 27, 2026

Keywords:

Cloud databases

Competency-based learning

Database systems education

Parallel computing education

Performance-aware structured query language

Quasi-experimental design

ABSTRACT

Although modern database systems increasingly rely on parallel and distributed execution, parallel computing is often taught separately from database courses. This study addresses this gap by embedding parallel computing readiness directly into database education through a competency-based instructional module. The study designed and evaluated an integrated module that introduced students to parallel execution reasoning through authentic database tasks. A quasi-experimental pre-test/post-test design with an experimental group (EG) (N=81) and a control group (CG) (N=71) was utilized to assess learning outcomes over four competency dimensions: parallel computing readiness, content/theoretical competence, organizational/practical competence, and motivational competence. Clear evidence of a shift from low competency levels to average and high competency levels within the EG was evident across all competency dimensions, with the strongest changes observed in content/theoretical competence, where low-level ratings decreased from 87.7% to 25.9%, and organizational/practical competence, where low-level ratings decreased from 88.9% to 37.0%. In contrast, the CG showed only limited improvement and remained predominantly at the low level across the main competency dimensions. The study provides support for the integration of parallelism with authentic database projects and offers a replicable model for assessing and developing parallel computing readiness in modern database curricula.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Ainur Tokzhigitova

Department of Information Technology, Faculty of Computer Science, Toraighyrov University

Pavlodar, Kazakhstan

Email: tokzhigitova.an@teachers.tou.edu.kz

1. INTRODUCTION

The proliferation of data-intensive applications has fundamentally altered the way that we think about executing the many tasks of database management systems in modern computing. The increasing demands of such applications are changing the nature of how we interact with them, to include new paradigms for large-scale analytics, processing of contested cloud systems, and the like. As a result, database management systems are evolving from being just a large collection of records, to being the primary execution engines for running multiple tasks concurrently and simultaneously interpreting and optimizing query results across the available resources in the distributed environment. Large-scale data processing is commonly represented through the MapReduce model and modern unified engines such as Apache Spark,

both of which rely on distributed and parallel execution principles [1], [2]. In this context, Orynbekova *et al.* [3] show that learners need structured support to understand how conventional tasks can be reformulated for distributed MapReduce-style execution. As such, students who can write structured query language (SQL) may still struggle to interpret how database systems actually process queries, optimize workloads, and distribute computation if they lack parallel-processing concepts such as task decomposition, overhead, synchronization, and performance measurement. These competencies are directly relevant to computer engineering because they support the analysis and optimization of database workloads across multicore, distributed, and cloud computing environments.

Database education research increasingly emphasizes that a key learning goal is not only writing SQL queries but also understanding how SQL queries are processed in an relational database management system (RDBMS) in practice [4]. This is important because query processing connects SQL formulation with validation, algebraic transformation, execution plans, physical access methods, database statistics, and execution-cost estimation [5]. Query execution plans can also reveal optimizer decisions, semantic issues, and performance-related behavior that are often hidden from novice users [6]. At the same time, SQL learning itself remains challenging. Taipalus [7] showed that increasing database schema complexity can reduce students' query formulation success, while Miedema *et al.* [8] and Miedema *et al.* [9] showed that SQL errors often reflect misconceptions, incomplete mental models, and difficulties with declarative reasoning. These findings suggest that database instruction should develop conceptual and diagnostic reasoning, not only syntactic query-writing ability.

Parallel computing education research also shows sustained challenges in teaching both theoretical concepts and practical skills [10]. Most curriculum proposals in the area of parallel and distributed computing emphasize the significance of basic subject knowledge gained by computer science students through compulsory courses taken by them [11]. Results from earlier studies have emphasized the need to implement parallel and distributed computing into more courses rather than confining it to elective courses only [12]. Empirical studies further suggest that parallel concepts can be taught effectively when instruction is carefully scaffolded and connected to meaningful tasks. For example, Brodanac *et al.* [13] found that students can understand and apply basic parallel programming concepts, although they may perceive the topic as more difficult than other computing topics. More specifically, the work of Orynbekova *et al.* [3] demonstrates the impact of structured, task-based instruction on the learning effectiveness of parallel and distributed paradigms, particularly when learners compare solution strategies across different platforms. Recent studies further show that parallel and distributed computing education increasingly emphasizes early exposure, hands-on programming, performance analysis, data-intensive workloads, and GPU-based parallelism [14]-[16].

In database education, this need for scaffolding is especially visible in the teaching of query execution plans. Liu *et al.* [17] proposed natural-language support for helping learners understand query execution strategies, while Wang *et al.* [18] emphasized that understanding alternative query plans is important for explaining why one execution plan is selected over another. At the same time, practical work with real systems provides an opportunity for students to link theoretical concepts with such aspects of the implementation of database execution plans as profiling information, storage system performance, limitations of CPU and memory, and performance issues. This engineering context now also includes multicore processors, distributed database systems, cloud platforms, GPU accelerators, and parallel data-processing frameworks, where workload partitioning, resource allocation, scalability, and execution cost become important design considerations [14]-[16], [19], [20]. This is consistent with performance-oriented cloud and HPC education, where theoretical concepts are combined with real platforms, workload characterization, and performance evaluation tools [19].

Similarly, Striki and Haghani [21] argue that cloud computing courses require careful selection of concepts, tools, and pedagogical pathways because the field combines distributed and parallel computing, programming models, systems design, data science, and machine learning. This supports the need to integrate parallelism into database education without turning the course into an unbounded systems survey. In Kazakhstan, the need for practice-oriented and digitally supported computing education is also visible in recent higher education research. Kurakbayeva and Xembayeva [22] showed that digital educational interventions can improve cognitive outcomes, motivation, and professional orientation among university students, while Abildinova *et al.* [23] reported positive student perceptions of active learning and digital integration in a computer science course.

For the Kazakhstani context, this evidence is important because higher education institutions are expected to prepare graduates for increasingly digital and performance-oriented professional environments. Broader evidence from active learning and STEM interventions also indicates that practice-centered approaches can improve student outcomes and reduce disengagement compared with passive instruction [24]-[26]. At the same time, cloud and distributed learning environments require attention to privacy, security, and governance of learner data, particularly when logs, performance traces, or telemetry are collected for assessment [27].

Motivation is also an important consideration. Since parallel computing is often perceived as difficult [13], integrating it into database education should not add unnecessary cognitive burden or reduce student engagement. Active and practice-centered learning designs can support engagement while helping

students persist in demanding technical tasks [24], [25]. For the topic of parallelism-in-databases, it would be preferable for students to be trained by providing them with opportunities to apply parallel reasoning, such as interpreting plans, predicting bottlenecks, and benchmarking configurations, on a continuous basis rather than simply reading about them.

AI technologies have also begun to play a larger role in modern technical learning environments. Recent reviews show that large language models are increasingly relevant to engineering and computing tasks, including code generation, debugging, verification, and analysis [28], [29]. Although AI is not required to teach students about parallel databases, it is part of a complex ecosystem of tools that affects how students search for help, debug code, and verify work. Therefore, the curriculum should explicitly outline acceptable tool use and verification expectations, especially in technical modules where students may be tempted to use AI assistance rather than develop their own reasoning capabilities.

Taken together, this literature shows a specific gap: although prior studies address SQL and query-plan education, parallel and distributed computing courses, and broader hands-on computing instruction, limited evidence examines parallel computing readiness as an integrated database-engineering competency within a standard database course. This gap establishes an opportunity to embed parallel computing readiness directly into database education through a competency-based and tool-mediated instructional module. In this scope of research, learning outcomes are framed through four parallel competency dimensions: parallel computing readiness, content/theoretical competence, organizational/practical competence, and motivational competence. The framework captures conceptual understanding of parallel operators and execution behavior, operational ability to execute benchmarking and interpret performance evidence, and learner motivation, including perceived usefulness and willingness to engage with challenging material. This framework is also consistent with previous findings indicating that parallel instruction is feasible but demanding [13], that hands-on approaches can improve student learning outcomes [19], [26], and that structured digital interventions can produce measurable improvement in Kazakhstan's higher education context [22], [23].

The novelty of this study therefore lies not in arguing that parallel computing is important, but in operationalizing parallel computing readiness as a database-focused instructional and assessment framework that combines authentic database tasks with multidimensional competency evaluation. By providing a database-specific sequential/parallel query comparison, execution-plan interpretation, benchmarking, workload distribution, and cloud-based database tasks together with multidimensional competency assessment, the study facilitates parallelism instruction without requiring a separate course dedicated solely to parallel programming and provides interpretable evidence of learning impact. The goal of the present study is to create and assess a competency-based instructional module that embeds parallel computing concepts into database instruction and develops students' ability to analyze and interpret parallel database execution and performance.

2. PEDAGOGICAL FRAMEWORK, COMPETENCY MODELS, AND LEARNING ENVIRONMENT

This project applied a competence-based model framework to evaluate students' preparedness to use parallel computing for database tasks. It proposes that preparedness can be viewed through four related dimensions, with parallel computing readiness treated as the overarching outcome supported by motivational competence, content/theoretical competence, and organizational/practical competence. Competencies for working with parallel computing include: i) reasoning about and utilizing ideas of parallel execution while performing tasks within a database, ii) engagement, iii) perceiving usefulness, iv) self-confidence, v) understanding the concepts of parallelism and execution models and their impact on performance, and vi) ability to manage workflows, run tools, execute experiments, and interpret the results.

All instruction was linked together by having students work on realistic database projects or tasks while gradually increasing their complexity. This was accomplished by moving from processing database tasks sequentially to processing them in parallel and comparing their execution performance over time. Representative activities included comparing sequential and parallel SQL execution, interpreting query execution plans, decomposing database-processing tasks with Python, executing workloads in local and networked environments, and benchmarking execution performance. These activities introduced workload partitioning, synchronization, resource allocation, and performance optimization as applied database-engineering problems. Formative feedback on laboratory reports was discussed during collaborative practical work and was used to help students interpret performance results and refine their explanations of sequential and parallel execution.

Figure 1 provides a visual representation of the pedagogical architecture linking curriculum content, laboratory activities, parallel-computing concepts, assessment evidence, and expected engineering competencies. The instructional sequence moves from database content through sequential/parallel query execution, Python-supported task decomposition, local/networked execution, and cloud-based tasks. These activities develop concepts of concurrency, workload partitioning, synchronization, resource allocation, and performance evaluation, while execution plans, benchmarking results, and practical task performance provide

assessment evidence for the competency framework. This framework also shows how conceptual understanding, practical workflow competence, and motivation support students' ability to interpret database execution behavior and performance differences.

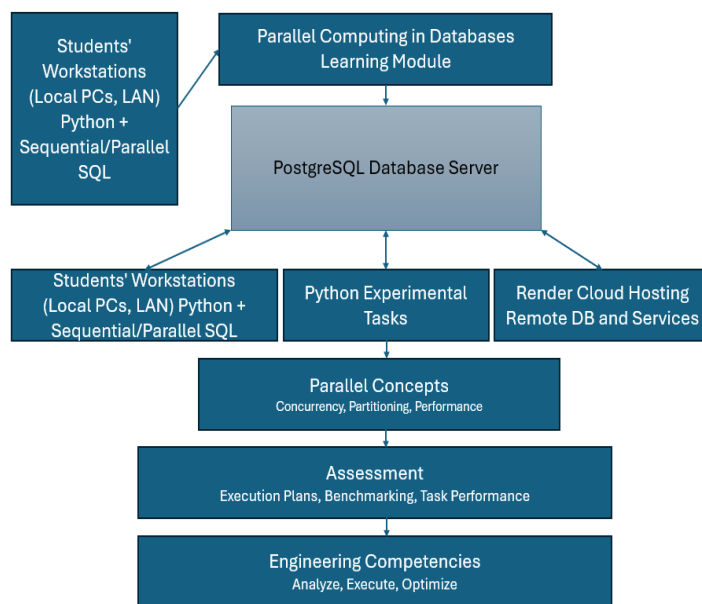


Figure 1. Pedagogical framework for parallel computing readiness and its four competency dimensions

All learning objectives associated with the four competencies were mapped through performance metrics to observable indicators. For example, content/theoretical competence was mapped to the interpretation of execution plans and explanation of sequential–parallel performance differences; organizational/practical competence was mapped to the ability to configure, execute, validate, and interpret database benchmarks and workload-distribution tasks; motivational competence was mapped to sustaining engagement with performance tasks. Parallel computing readiness represented students' combined ability to apply these competencies when completing authentic parallel-database tasks.

The module was set up in a learning environment using a client/server architecture, as shown in Figure 2, which displayed the use of local client workstations (PCs/desktop clients), cloud services (remote access and standardized conditions), and a centralized PostgreSQL database server in the cloud. The software environment included python for task decomposition and processing, PostgreSQL for database execution and query-plan analysis, sequential and parallel SQL activities, local/networked execution, and render for cloud-based deployment.

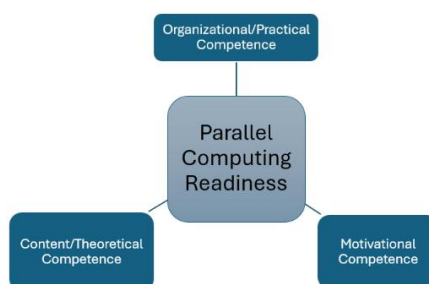


Figure 2. Learning environment architecture for the parallel computing in databases instructional module

The workstation and server configurations were standardized as far as possible to support comparable execution conditions. The setup allowed for the same execution path between where students were testing (client-side) and where teachers were running or monitoring the tasks (server-side). This helped maintain reproducibility of tasks and ease of comparison of students' work.

3. METHODS

Prior to implementation and after completion, a pre-test/post-test quasi-experimental design was used to compare the experimental group (EG), N=81) and control group (CG), N=71). The participants were 152 third-year students enrolled in the 6B06103 information systems and 6B01505 computer science programmes at Karagandy Buketov University and Abylka Saginov Karaganda Technical University. The students had basic knowledge of information technology and had previously studied the fundamentals of Python programming.

The EG consisted of 81 students, including 77 information systems students and four computer science students from Karagandy Buketov University. The CG consisted of 71 students, including 15 computer science students from Karagandy Buketov University and 56 Information Systems students from Abylka Saginov Karaganda Technical University. Group allocation was non-random and followed the existing programme-based student groups within the participating universities. Because the intervention was integrated into scheduled database courses, the established groups were retained rather than reorganizing individual students.

In order to establish a baseline assessment of both EG and CG, both groups completed a baseline assessment prior to receiving the intervention. Students in the EG received instruction using the parallel-computing-integrated database module, while the CG received standard instruction on databases. The experimental instruction included structured activities involving sequential and parallel queries, Python-supported parallel processing, local and networked execution, PostgreSQL performance analysis, and cloud-based database deployment. In contrast, the CG continued with the existing database curriculum without the integrated parallel-computing module. The main differences between the experimental and control instructional conditions are summarized in Table 1.

Table 1. Comparison of the experimental and control instructional conditions

Element	EG	CG
Participants	81 students from the information systems and computer science programmes at Karagandy Buketov University.	71 students from the information systems and computer science programmes at Karagandy Buketov University and Abylka Saginov Karaganda Technical University.
Database instruction	Standard database content with the integrated parallel-computing module.	Standard database instruction without the integrated module.
Additional activities	Sequential–parallel query comparison, Python-supported processing, local and networked execution, cloud database deployment, and performance interpretation.	No structured parallel-computing intervention.
Assessment	Pre-test and post-test using the competency-level framework.	The same pre-test and post-test framework.

After the instructional period, both groups were assessed through their respective post-tests that measured the impact of each type of instruction on changes in their performance as compared to their performance prior to the instruction. Students were assessed using a competency rubric designed to measure the competencies required by the students to perform successfully on parallel-computing database tasks. There were four competencies included in the competency rubric outlined in section 2 of the paper. Using the ordinal three-level rating scale of low, average, and high, students were assessed in regard to conceptual knowledge as well as practical execution performance and readiness to perform the parallel-computing activities; low, average, and high represented limited, partial, and independent performance, respectively. Practical performance was evaluated using observable evidence from the assigned database tasks, including execution-plan interpretation, execution of sequential and parallel workloads, benchmark implementation, and interpretation of execution-performance results.

The competency indicators were adapted from the criteria used in the preceding pedagogical work on parallel computing in databases. The motivational criterion assessed students' interest in and perceived professional value of parallel computing; the content criterion assessed their theoretical knowledge; and the organizational criterion assessed their ability to apply relevant methods and use the required hardware and software. In the present study, these indicators were organized into motivational competence, content/theoretical competence, and organizational/practical competence, while parallel computing readiness was treated as the overarching outcome reflecting students' combined preparedness to perform parallel-computing database tasks. The rubric was developed specifically for this instructional context and was not presented as a standardized psychometrically validated instrument.

The study followed three stages: a pre-test for both groups, the instructional intervention for the EG only, and a post-test for both groups, allowing changes in competency levels from pre-test to post-test to be assessed. The instructional tasks included sequential and parallel SQL activities, PostgreSQL-based database work, Python-supported tasks, benchmarking, and interpretation of execution performance. Representative activities included comparing the execution times of sequential and parallel SQL queries, using Python-supported tools to distribute database-processing tasks, testing parallel execution on local and

networked computers, and deploying PostgreSQL-based applications in a cloud environment. Students also interpreted execution plans and performance results to explain why particular queries or configurations produced different execution times. Execution time and benchmark outputs were therefore used as technical evidence within the practical activities, although they were not treated as separate standardized outcome variables in the statistical analysis. The main instructional stages, representative tasks, and intended learning outcomes of the integrated module are summarized in Table 2.

Table 2. Main instructional stages of the integrated module

Instructional stage	Topic	Representative task	Intended learning outcome
1	Sequential and parallel query execution	Write equivalent sequential and parallel SQL queries and compare their execution times.	Distinguish sequential and parallel execution and interpret performance differences.
2	Python-supported database processing	Use Python and data-processing libraries to divide database-processing tasks into independently executable parts.	Apply task decomposition and parallel-processing principles.
3	Local and networked execution	Execute database-processing tasks on a local computer and across computers connected through a local network.	Understand resource allocation, synchronization, and distributed execution.
4	Cloud-based database processing	Deploy a PostgreSQL-based application using Render and test query processing under changing workloads.	Apply parallel and scalable database-processing principles in a cloud environment.

Data for both groups were summarized as frequencies and percentages across the three competency levels at pre-test and post-test under the same assessment conditions. Descriptive statistics, chi-square tests, and effect-size estimates were used to compare competency-level distributions.

The chi-square test was used to compare the distribution of students across the three rubric levels (low, average, and high) between EG and CG at post-test for each competency dimension. This test was selected because both group membership and competency level were categorical variables and the primary analysis concerned differences in their frequency distributions. For each comparison, a 2×3 contingency table was constructed, where rows represented the groups and columns represented the rubric levels. Pearson's chi-square statistic was calculated from the observed and expected frequencies, and the p-value was interpreted using two degrees of freedom. The assumptions of independent observations, mutually exclusive categories, and adequate expected cell frequencies were checked before interpretation of the tests. Cramer's V was used as the effect-size estimate to indicate the strength of association between group membership and competency-level distribution, with 95% confidence intervals reported alongside the corresponding significance tests. For supplementary descriptive reporting, the ordinal categories low, average, and high were coded as 1, 2, and 3, respectively. Means and standard deviations were calculated, and pre-test to post-test change was summarized using Cohen's d_{av} , based on the average of the pre-test and post-test variances. These estimates were interpreted cautiously because the competency levels constituted an ordinal scale. Bootstrap 95% confidence intervals for Cramer's V were estimated using 20,000 resamples conducted separately within the experimental and CGs.

All standards of ethical conduct were followed throughout the study to ensure confidentiality of participants and minimize the risk of disclosure of participants' identities by analyzing participants' responses anonymously and only reporting aggregate statistical summaries. As well, there were no sources of sensitive student information collected, and there were no foreseeable sources of harm resulting from the activities associated with normal educational practices.

4. RESULTS

Results are reported in the form of frequency distributions across three levels of the rubric (low, average, and high) by individual competency dimension for the EG and CG at pre- and post-test (EG=81; CG=71). Figure 3 displays how these distributions change over time through the use of stacked columns, which allows viewers to compare distributions by; i) visualizing baseline profiles, ii) tracking changes to within-group profile distribution pre- and post-testing, and iii) contrasting distributions between EG and CG within a given competency dimension.

At pre-test, both groups showed predominantly low profile distributions for all four competency dimensions, thus indicating a low level of initial readiness to undertake database tasks utilizing parallel computing. When comparing profile distributions, we observe that for parallel computing readiness, low profile constitutes 66.7% (54/81) of EG and 77.5% (55/71) of CG; for motivational competence, low profile encompasses 72.8% (59/81) of EG and 80.3% (57/71) of CG; and for content/theoretical competence, low profile constitutes 87.7% (71/81) of EG and 85.9% (61/71) of CG; for organizational/practical competence,

low profile comprises 88.9% (72/81) of EG and 94.4% (67/71) of CG. While the two groups varied to some extent with respect to their makeup, their pre-test distributions were similar in that both groups started out with a significant majority of low profiles in all four competency dimensions. Baseline chi-square tests showed no statistically significant differences between EG and CG in parallel computing readiness, $\chi^2(2) = 2.18$, $p = 0.336$; motivational competence, $\chi^2(2) = 1.70$, $p = 0.428$; content/theoretical competence, $\chi^2(2) = 0.37$, $p = 0.832$; or organizational/practical competence, $\chi^2(2) = 1.53$, $p = 0.466$. Descriptive summaries and post-test group comparisons are presented in Table 3.

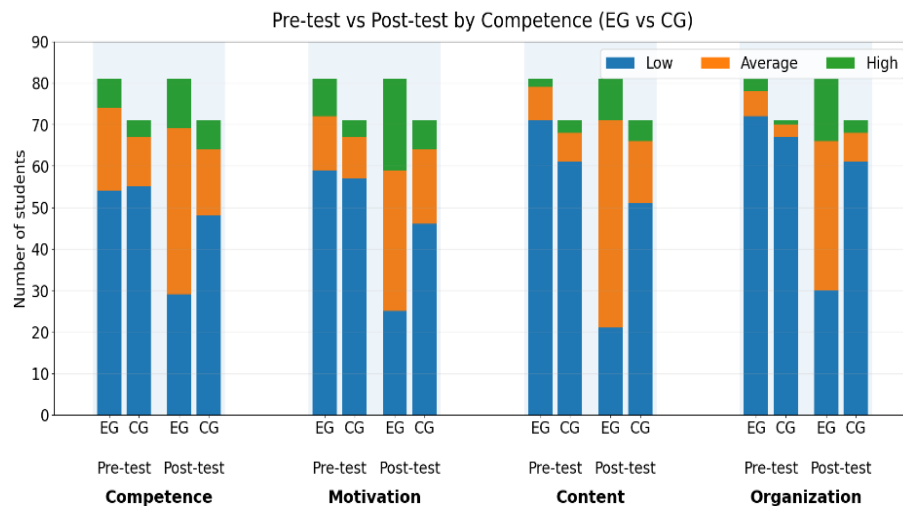


Figure 3. Pre-test vs post-test distributions by competency dimension

Table 3. Descriptive and inferential statistics by competency dimension

Competency dimension	EG pre-test M (SD)	EG post-test M (SD)	EG d_{av}	CG pre-test M (SD)	CG post-test M (SD)	CG d_{av}	Post-test $\chi^2(2)$	p	Cramer's V [95% CI]
Parallel computing readiness	1.42 (0.65)	1.79 (0.68)	0.56	1.28 (0.57)	1.42 (0.67)	0.23	15.70	<0.001	0.321 [0.183, 0.474]
Motivational competence	1.38 (0.68)	1.96 (0.77)	0.80	1.25 (0.55)	1.45 (0.67)	0.32	18.31	<0.001	0.347 [0.209, 0.498]
Content/theoretical competence	1.15 (0.42)	1.86 (0.61)	1.37	1.18 (0.49)	1.35 (0.61)	0.31	32.50	<0.001	0.462 [0.327, 0.606]
Organizational/practical competence	1.15 (0.45)	1.81 (0.73)	1.10	1.07 (0.31)	1.18 (0.49)	0.28	37.62	<0.001	0.498 [0.370, 0.627]

On the ordinal-coded scale, the largest standardized pre-test to post-test changes in the EG occurred in content/theoretical competence ($d_{av} = 1.37$) and organizational/practical competence ($d_{av} = 1.10$). The corresponding effects in the CG were substantially smaller, at 0.31 and 0.28, respectively. At post-test, Cramer's V values ranged from 0.321 to 0.498 across the four dimensions, indicating that the statistically significant group differences were accompanied by meaningful effect sizes. After the final test had concluded, clear differences in the distribution of scores among the members of both groups were noted. Chi-square comparisons of post-test distributions confirmed significant EG–CG differences across all four competency dimensions. The members of the EG showed a noticeable shift from low to average/high in every category. The CG showed minor shifts in all categories.

For example, in parallel computing readiness, the members of the EG group moved away from the low range (54 to 29 members; 66.7% to 35.8%) and toward the average (20 to 40 members; 24.7% to 49.4%) and high ranges (7 to 12 members; 8.6% to 14.8%). On the other hand, the CG group was still heavily dominant in the low range (55 to 48 members; 77.5% to 67.6%) after being tested. They did show slight improvement in both average and high numbers (12 to 16 members and 4 to 7 members, respectively). The post-test EG–CG difference for this dimension was significant, $\chi^2(2)=15.70$, $p<0.001$, Cramer's V=0.321, 95% CI [0.183, 0.474].

Regarding motivational competence, there was a noticeable change in the EG group when looking at scores. Again, the EG group showed a significant decrease in members rated low (59 to 25 members; 72.8% to

30.9%) while increasing those rated average (13 to 34 members; 16.0% to 42.0%) and high (9 to 22 members; 11.1% to 27.2%). On the other hand, CG group members also improved their scores but not nearly to the same extent as the EG group, with low ratings decreasing from 57 to 46 members, average ratings increasing from 10 to 18 members, and high ratings increasing from 4 to 7 members. The post-test EG–CG difference for this dimension was significant, $\chi^2(2)=18.31$, $p<0.001$, Cramer's $V=0.347$, 95% CI [0.209, 0.498].

The most significant changes in the EG group occurred in the area of content/theoretical competence. In this area, members moved from a baseline of 71 members rated low to only 21 members rated low at the end of the study (87.7% to 25.9%). Members in the CG group decreased their low rating in this area only slightly from 61 members to 51 members. The post-test EG–CG difference for this dimension was significant, $\chi^2(2)=32.50$, $p<0.001$, Cramer's $V=0.462$, 95% CI [0.327, 0.606].

In the area of organizational/practical competence, there was a large movement from low to average and high scores by the EG group members. Their low rating dropped from 72 members to 30 members, while average ratings increased from 6 to 36 members and high ratings increased from 3 to 15 members. There was only slight improvement in the CG group from a total of 67 members rated low to 61 members rated low; average ratings increased from 3 to 7 members, and high ratings increased from 1 to 3 members. The post-test EG–CG difference for this dimension was significant, $\chi^2(2)=37.62$, $p<0.001$, Cramer's $V=0.498$, 95% CI [0.370, 0.627]. This was the strongest post-test association among the four dimensions and is particularly relevant to students' practical ability to execute and interpret parallel-database workflows.

The data indicate that the EG group members were able to increase their average or high scores across all competency dimensions; by contrast, the CG group remained near the bottom of the distribution of their scores. Descriptive statistics and chi-square comparisons show these differences in results visually and statistically in Figure 3, which depicts the considerable increases in the average and high scores for the members of the EG group compared to the reasonably small increases in both the average and high scores for the CG group.

5. DISCUSSION

The EG exhibited strong increases in the content/theoretical and organizational/practical competence dimensions of learning, demonstrating that students acquired not simply declarative knowledge of parallelism but also developed practical skills needed to execute workflows, make sense of system outputs, and achieve performance tasks. Additionally, students learned about “what parallelism is” more consistently by repeatedly witnessing parallelism demonstrated in concrete database artifacts—execution behaviour, query execution plans, benchmarking outputs, resource contention, and performance differences across query variants [19]—than by receiving conventional instruction. Thus, the EGs strong learning gains in these two dimensions suggest that the learning mechanism associated with developing parallel competence occurs at the intersection of concepts and mediated practice with tools.

Likewise, the very limited gains in motivational competence experienced by the CG are explained logically by the same reasoning. The majority of students perceive parallel computing as difficult to understand, and the absence of well-structured activities that include visible progress signals leads to disengagement from learning about parallel computing [13]. Therefore, by providing students with experiences that deliver concrete progress signals, students can better relate the task of learning about parallel computing to their skill level. The active-learning literature in STEM supports the hypothesis that practice-centred designs positively influence student outcomes compared to instructional designs based on passive learning experiences, and decrease the frequency of disengagement that commonly occurs due to the challenges of learning abstract theoretical content [24]. The gains in motivational competence achieved in this study should not be seen as “secondary” outcomes; rather, they were likely enabling conditions that contributed to the likelihood that students would persist in cognitively demanding parallel reasoning tasks.

As a conclusion, the results of the analysis have shown that parallel computing readiness acts as a composite dependent on the other three facets. The EG demonstrated substantial gains in readiness; however, the relative distribution of these gains was much less dramatic compared to content/organizational gains. One would expect this to be the case for the reason that readiness is not a one-dimensional skill but is instead an amalgamation of; i) conceptual frameworks of parallel execution skills; ii) the ability to create and debug workflows and outcomes; and iii) the willingness to undertake performance assessments with respect to uncertainty. Therefore, readiness will likely be the last capability to be established. Accordingly, the basis for undertaking multidimensional evaluation as opposed to solely relying on a single “knowledge test” outcome is supported.

The results support previous findings that demonstrate that students' understanding of parallel concepts can be enhanced when they are introduced through scaffolding in familiar environments. For example, [13] showed that secondary school students can begin to learn how to program in parallel as early as the first year of their program but still view it as a demanding subject. The findings of the present research extend this rationale into a database context where parallel execution was treated as a naturally parallel

environment. The students were constantly exposed to concurrency, partitioning, and performance trade-offs as part of authentic tasks, rather than treating parallel execution as an isolated topic.

The results also support evidence from a Kazakhstan-based study that has demonstrated that using a structured task-based approach to teach parallel/distributed paradigms provides greater learning gains than other instructional approaches. Orynbekova *et al.* [3] reported an increase in student performance outcomes when students engaged in comparing and implementing various types of distributed solutions (e.g., MapReduce patterns) to a variety of problem types. The findings of the present research have a pedagogical philosophy that is similar to previous studies in terms of repeated exposure to parallel abstractions through implementation and comparison. However, a significant difference exists with regard to the curricular placement; the integration occurs within the context of a database education program where parallel execution cannot be avoided operationally and is therefore easier to view as a core competency.

In terms of instructional design, the findings are consistent with the hands-on orientation as presented by Pons *et al.* [19]. Their contention is that for many systems-related topics, the theoretical component will be more evident when students are provided with an opportunity to work in authentic settings while utilising realistic tools and performance evaluation methodologies. The relatively minor shift to the CG further substantiates this assertion. Without providing a framework that includes explicit tool-mediated performance reasoning and a defined progressive path for students, researchers indicate that many students will complete database-related assignments with minimal development of parallel reasoning and workflow competencies. In addition to this research, the curated scope of the module parallels the rationale for the selection of hardware, software, and methodologies as discussed by Striki and Haghani [21], especially in terms of preserving clarity when adding “parallelism” to a systems course, as expanding these topics could easily result in unwieldy survey courses. Compared with broader project- or problem-based approaches, the present framework uses progressively structured database tasks and controlled performance comparisons within an existing database course rather than organizing the entire learning process around an open-ended project or problem. These approaches are complementary, but the present framework places greater emphasis on the explicit progression and assessment of parallel-database competencies.

In terms of methodological approaches, the quasi-experimental pre- and post- comparison method using multi-dimensional outcomes fall in line with previous regional studies that evaluated the impacts educational interventions in the digital space. Kurakbayeva and Xembayeva [22] demonstrated the ability of pre- and post-testing design models to capture improvements in both cognitive understanding of participants and also motivation-oriented constructs in the Kazakh higher educational context. As with the broader intervention work in STEM fields, there is an emerging body of evidence that indicates that hands-on and tool-based learning improves higher-order cognitive capabilities [26]. Consequently, the present study expands this larger manifestation of educational research into a more specific domain (parallelism in database education). Overall, the pattern of substantial improvement in the EG and limited change in the CG is consistent with the task-based and hands-on interventions discussed above, although direct comparison of effect magnitudes is limited by differences in measures and educational contexts.

Curricular implications arise from this study. Database courses should teach parallel execution as an explicit learning goal, practice it, and assess it alongside other goals. Because many databases are designed to process workloads in parallel at scale [1], [2], “SQL proficiency” by itself does not reflect a complete graduate profile for new databases. Incorporating parallel reasoning into database instruction aligns with the way systems are behaving in practice. This is also relevant for student readiness, because graduates increasingly need to interpret SQL performance, cloud database behaviour, and parallel execution rather than only write syntactically correct queries. These competencies are directly relevant to cloud and high-performance computing environments, where specialists must interpret execution plans, benchmark workloads, and evaluate resource-related performance trade-offs. In practical engineering terms, the module develops students’ ability to analyse execution behaviour, compare workload performance, and reason about resource allocation when database tasks move from local to distributed or cloud environments. As an industry-aligned extension, the module could incorporate technologies such as Spark SQL, distributed PostgreSQL systems, cloud data warehouses, and other distributed query engines. This approach is also consistent with established parallel and distributed computing curriculum initiatives that recommend baseline PDC competence for undergraduate computing students [11] and the integration of parallel and distributed concepts across existing computer science courses rather than restricting them to separate electives [14]. The present framework applies this principle specifically within database education.

An implication of this research involves how assessment is done. Significant differences were observed among the students in their theoretical knowledge, operational competence, and practical experience; therefore, an assessment based only on conceptual knowledge (e.g., SQL syntax and usage) would fail to demonstrate the range of impacts from instructional methods and would not provide educators with informed recommendations for course improvement. A competency rubric that assesses readiness

(motivation), conceptual learning, practical ability, and workflow assists educators to identify issues with their students. For example, a student may demonstrate strong conceptual knowledge of SQL, but not be able to execute it because issues regarding tools and support were not covered in the course. Additionally, students may be motivated to learn about SQL; however, while in their learning environment and encountering a difficult task, the absence of timely feedback will likely result in decreased motivation.

The study findings have implications for instructor preparation. Teaching parallelism in databases requires instructors to not only understand databases but also to understand how database performance occurs, the importance of having well-defined experimental controls (e.g., comparable benchmarking), and the ability to interpret system outputs, including query execution plans and performance trade-offs [19].

Implementation of recommended instructional approaches and practices requires special consideration for the operational constraints involved. For institutions with limited computing resources, the framework can be implemented using local PostgreSQL installations, Python, and LAN-based execution, while shared institutional servers or cloud services can support activities requiring additional resources. For larger classes, standardized datasets, benchmark scripts, and group-based laboratory activities can reduce infrastructure and monitoring demands. When using cloud or other remote services for accessing databases, it is essential to ensure that all aspects of privacy, security, and governance are defined in a responsible manner as part of the instructional design process, especially if logs or telemetry are to be collected [27]. Moreover, because students increasingly rely on AI-augmented tools in their technical learning environments, the course's policies on tool use and verification need to clearly articulate acceptable use as well as expectations for verification to help ensure that learning goals and academic integrity are addressed and accomplished [28], [29].

Limitations of this study stem from its quasi-experimental design (i.e., a randomized assignment to groups was not done) and thus the impact of potential unobserved differences between the groups may have affected the outcomes of the study. More specifically, the experimental and CGs represented different programmed cohorts and were partly drawn from different universities. Consequently, institutional conditions, instructor-related effects, and differences between the existing student groups could not be fully separated from the effect of the instructional intervention. Differences in prior programming experience may also have contributed to individual variation, although all participants had basic information-technology knowledge and previous exposure to Python fundamentals. The use of the same pre-test/post-test assessment framework and baseline group comparisons reduced, but did not eliminate, these threats to validity. The evaluations conducted were based on a three-level ordinal rubric (low/average/high) which lends itself to producing reporting that is interpretable, but as a trade-off, it reduces granularity of measurement and may mask true or smaller improvements within category levels (e.g., for example, the students who received the treatment and were rated as Low, may have been ranked within a different category had their measurement been conducted at a higher level of granularity). In addition to the above limitations, the intervention was short term and was not assessed through a delayed post-test; therefore, the findings do not establish whether the observed competencies will persist over time or transfer to other database environments or independent industry-scale projects. Although contextual issues may limit the ability to generalize the findings of the study, the sample size and two-university, single-country context also limit the generalizability of the findings. In particular, motivational responses may vary according to institutional culture, students' prior educational experiences, and local perceptions of parallel computing. Therefore, it is important to realize that findings from the study are unlikely to be universally applicable and will be influenced by factors such as varying levels of student preparedness, computer capability, and instructor expertise. As such, future research should replicate the framework across multiple cohorts and academic semesters and supplement competency-rubric outcomes with standardized programming assessments, execution-time and scalability measures, software-quality metrics, and task-performance trace data.

6. CONCLUSION

This study presents the design and evaluation of a competency-based instructional module that integrates parallel computing into database education, measuring learning using four distinct dimensions. These include; i) parallel computing readiness, ii) content/theoretical competence, iii) organizational/practical competence, and iv) motivational competence. Through pre-test results from the control and EGs participating, the majority of both groups developed very low profiles across all four dimensions prior to instruction, indicating that both groups had significant readiness gaps in completing database tasks involving parallel computing. Following instruction, the EG exhibited significant redistribution from low to average (and High). The clearest improvements were observed in content/theoretical competence and organizational/practical competence, where low-level profiles in the EG decreased from 87.7% to 25.9% and from 88.9% to 37.0%, respectively. On the other hand, the CG showed only moderate redistribution toward average from low, and remained predominantly Low across the dimensions. This redistribution pattern lends support to the conclusion that the embedding of authentic database tasks with parallel content, combined with

incremental complexity, tool mediated practice, and feedback, can support substantial multilevel gains in this study context that are not easily captured through conceptual evaluations alone.

This research makes the contribution in two basic aspects: first of all, it envisions an instructional scheme for integrating parallel concepts into database education that implies the use of parallel concepts as a framework for the introduction of content, second, it provides an evaluation framework to measure readiness to learn parallel computing that identifies conceptual understanding, organizational workflow competency, and motivational readiness of students. The research presents the instructive model for creating easy-to-implement instructional schemes that allow for achieving the learning goals without the establishment of separate parallel programming courses, while still providing necessary assessment data about the ability of students to interpret execution behaviour. For curriculum developers and engineers, the advice that can be given is the application of sequential-parallel queries comparison, execution plan interpretation, benchmarking, workload allocation, and distributed execution as measurable activities within the standard database courses.

The study demonstrates that the integration of parallel computing into database education can have a positive impact on student learning/competency through four distinct dimensions. It also provides guidance to instructors on assessing students' readiness to perform parallel computing in a database environment. The results of the study indicate that students who received a parallel instructional paradigm developed a greater understanding of the impact of parallel computing in database education compared to those who received traditional database instruction without the benefit of embedded parallel content. Future studies may replicate and adapt this module in other institutional and international contexts, especially in database curricula that include cloud-based, distributed, or performance-oriented database systems. Future work should also use delayed post-tests to assess the persistence of the observed competencies and examine LLM-assisted query-plan explanation, adaptive learning systems, and intelligent tutoring technologies under controlled conditions to determine whether such support improves understanding without replacing students' independent reasoning.

FUNDING INFORMATION

This research received no external funding.

AUTHOR CONTRIBUTIONS

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Saule Zhumagulova	✓	✓								✓		✓		
Ainur Tokzhigitova	✓	✓				✓		✓	✓	✓			✓	
Gulnur Abildaeva				✓	✓	✓				✓				
Zhana-Gul						✓	✓	✓		✓				
Yessendauletova														
Leilya Fazylova				✓	✓					✓	✓			

C : **C**onceptualization

M : **M**ethodology

So : **S**oftware

Va : **V**alidation

Fo : **F**ormal analysis

I : **I**nvestigation

R : **R**esources

D : **D**ata Curation

O : Writing - **O**riginal Draft

E : Writing - Review & **E**ding

Vi : **V**isualization

Su : **S**upervision

P : **P**roject administration

Fu : **F**unding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request. The data are not publicly available to protect participant confidentiality.

REFERENCES

- [1] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008, doi: 10.1145/1327452.1327492.
- [2] M. Zaharia *et al.*, "Apache spark: A unified engine for big data processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, Oct. 2016, doi: 10.1145/2934664.
- [3] K. Orynbekova, A. Bogdanchikov, S. Cankurt, A. Adamov, and S. Kadyrov, "MapReduce Solutions Classification by Their Implementation," *International Journal of Engineering Pedagogy*, vol. 13, no. 5, pp. 58–71, Jul. 2023, doi: 10.3991/ijep.v13i5.38867.
- [4] S. S. Bhowmick and H. Li, "Towards Technology-Enabled Learning of Relational Query Processing," *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, vol. 45, no. 3, pp. 29–46, 2022.
- [5] K. C. Davis, "Instructional Design for Teaching Relational Query Optimization to Undergraduates," in *Proceedings of the 1st ACM SIGMOD International Workshop on Data Systems Education: Bridging Education Practice with Education Research, DataEd 2022*, Jun. 2022, pp. 44–50, doi: 10.1145/3531072.3535325.
- [6] T. Taipalus, "Query Execution Plans and Semantic Errors: Usability and Educational Opportunities," in *Conference on Human Factors in Computing Systems-Proceedings*, Apr. 2023, pp. 1–6, doi: 10.1145/3544549.3585794.
- [7] T. Taipalus, "The effects of database complexity on SQL query formulation," *Journal of Systems and Software*, vol. 165, p. 110576, Jul. 2020, doi: 10.1016/j.jss.2020.110576.
- [8] D. Miedema, E. Aivaloglou, and G. Fletcher, "Identifying SQL Misconceptions of Novices: Findings from a Think-Aloud Study," in *ICER 2021 - Proceedings of the 17th ACM Conference on International Computing Education Research*, Aug. 2021, pp. 355–367, doi: 10.1145/3446871.3469759.
- [9] D. Miedema, G. Fletcher, and E. Aivaloglou, "Expert Perspectives on Student Errors in SQL," *ACM Transactions on Computing Education*, vol. 23, no. 1, pp. 1–28, Mar. 2022, doi: 10.1145/3551392.
- [10] T. D. J. O. Duraes, P. S. L. D. Souza, G. Martins, D. J. Conte, N. G. Bachiega, and S. M. Bruschi, "Research on Parallel Computing Teaching: State of the art and future directions," in *Proceedings - Frontiers in Education Conference, FIE*, Oct. 2020, vol. 2020, pp. 1–9, doi: 10.1109/FIE44824.2020.9273914.
- [11] S. K. Prasad *et al.*, "NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing-Core topics for undergraduates," in *SIGCSE'11-Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, 2011, pp. 617–618, doi: 10.1145/1953163.1953336.
- [12] A. Danner and T. Newhall, "Integrating parallel and distributed computing topics into an undergraduate CS curriculum," in *Proceedings - IEEE 27th International Parallel and Distributed Processing Symposium Workshops and PhD Forum, IPDPSW 2013*, May 2013, pp. 1237–1243, doi: 10.1109/IPDPSW.2013.67.
- [13] P. Brodanac, J. Novak, and I. Boljat, "Has the time come to teach parallel programming to secondary school students?," *Heliyon*, vol. 8, no. 1, p. e08662, Jan. 2022, doi: 10.1016/j.heliyon.2021.e08662.
- [14] T. Newhall, K. C. Webb, V. Chaganti, and A. Danner, "An introductory-level undergraduate CS course that introduces parallel computing," *Journal of Parallel and Distributed Computing*, vol. 199, p. 105044, May 2025, doi: 10.1016/j.jpdc.2025.105044.
- [15] M. Gowanlock, "Teaching parallel and distributed computing using data-intensive computing modules," *Journal of Parallel and Distributed Computing*, vol. 202, p. 105093, Aug. 2025, doi: 10.1016/j.jpdc.2025.105093.
- [16] B. Plancher, "Parallel Optimization for Robotics: An Undergraduate Introduction to GPU Parallel Programming and Numerical Optimization Research," in *2024 IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2024*, May 2024, pp. 342–345, doi: 10.1109/IPDPSW63119.2024.00079.
- [17] S. Liu, S. S. Bhowmick, W. Zhang, S. Wang, W. Huang, and S. Joty, "Neuron: Query execution plan meets natural language processing for augmenting DB education," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Jun. 2019, pp. 1953–1956, doi: 10.1145/3299869.3320213.
- [18] H. Wang, H. Li, S. S. Bhowmick, and B. Xu, "ARENA: Alternative Relational Query Plan Exploration for Database Education," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Jun. 2023, pp. 107–110, doi: 10.1145/3555041.3589713.
- [19] L. Pons, S. Petit, and J. Sahuquillo, "Advanced resource management: A hands-on master course in HPC and cloud computing," *Journal of Parallel and Distributed Computing*, vol. 202, p. 105091, Aug. 2025, doi: 10.1016/j.jpdc.2025.105091.
- [20] B. Guo, J. Wu, Y. Pu, J. Zhang, and J. Yu, "Energy consumption estimation and profiling for queries in distributed database systems based on a bottom-up comprehensive energy model," *Future Generation Computer Systems*, vol. 159, pp. 379–394, Oct. 2024, doi: 10.1016/j.future.2024.04.059.
- [21] M. Striki and S. Haghani, "Finding the Needles in the Haystack: Identifying Essential Concepts and Tools for Inclusion in a Cross-Listed Introductory Semester Course in Cloud Computing," in *IEEE Global Engineering Education Conference, EDUCON*, 2024, doi: 10.1109/EDUCON60312.2024.10578822.
- [22] A. Kurakbayeva and S. Xembayeva, "Enhancing professional abilities of university students through digital educational interventions: a study in Kazakhstani universities," *Frontiers in Education*, vol. 9, Jan. 2024, doi: 10.3389/feduc.2024.1478622.
- [23] G. Abildinova, T. Sembayev, and K. Mukhtarkyzy, "Enhancing computer science education: student insights on active learning and digital integration," *World Transactions on Engineering and Technology Education*, vol. 22, no. 4, pp. 299–305, 2024.
- [24] S. Freeman *et al.*, "Active learning increases student performance in science, engineering, and mathematics," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 111, no. 23, pp. 8410–8415, Jun. 2014, doi: 10.1073/pnas.1319030111.
- [25] E. J. Theobald *et al.*, "Active learning narrows achievement gaps for underrepresented students in undergraduate science, technology, engineering, and math," in *Proceedings of the National Academy of Sciences of the United States of America*, vol. 117, no. 12, pp. 6476–6483, Mar. 2020, doi: 10.1073/pnas.1916903117.
- [26] Y. Dosymov, E. Ergobek, S. Ramankulov, A. Ualikhan, I. Usembayeva, and B. Kurbanbekov, "A New Approach to Development of Students' Research Abilities in STEM Education," *Emerging Science Journal*, vol. 9, no. 2, pp. 741–763, Apr. 2025, doi: 10.28991/ESJ-2025-09-02-013.
- [27] M. U. Afzal, A. A. Abdellatif, M. Zubair, M. Q. Mehmood, and Y. Massoud, "Privacy and Security in Distributed Learning: A Review of Challenges, Solutions, and Open Research Issues," *IEEE Access*, vol. 11, pp. 114562–114581, 2023, doi: 10.1109/ACCESS.2023.3323932.
- [28] M. Abdollahi, S. F. Yegani, M. Baharloo, and A. Baniyasi, "Hardware Design and Verification with Large Language Models: A Scoping Review, Challenges, and Open Issues," *Electronics (Switzerland)*, vol. 14, no. 1, p. 120, Dec. 2025, doi: 10.3390/electronics14010120.

- [29] M. Shao, A. Basit, R. Karri, and M. Shafique, "Survey of Different Large Language Model Architectures: Trends, Benchmarks, and Challenges," *IEEE Access*, vol. 12, pp. 188664–188706, 2024, doi: 10.1109/ACCESS.2024.3482107.

BIOGRAPHIES OF AUTHORS



Saule Zhumagulova    received her Ph.D. degree from L.N. Gumilyov Eurasian National University, Astana, Kazakhstan. She is currently an Associate Professor at Karaganda National Research University named after E.A. Buketov, Karaganda, Kazakhstan. Her academic interests include education, digital learning, computer science education, and competency-based teaching approaches. She can be contacted at email: zhumagulova.S@karnu-buketov.edu.kz.



Ainur Tokzhigitova    received her Ph.D. degree from L.N. Gumilyov Eurasian National University, Astana, Kazakhstan. She is currently an Associate Professor at the Faculty of Computer Science, Toraighyrov University, Pavlodar, Kazakhstan. Her research interests include computer science education, digital technologies in education, competency-based learning, and instructional design. She is the corresponding author and can be contacted at email: tokzhigitova.an@teachers.tou.edu.kz.



Gulnur Abildaeva    holds a Master's degree and is currently a Senior Lecturer at the Department of Information and Computing Systems, Faculty of Information Technology, Saginov Technical University, Karaganda, Kazakhstan. Her research interests include programming technologies, information systems, computer technologies, three-dimensional graphics, and animation. She can be contacted at email: g.abildaeva@ktu.edu.kz.



Zhana-Gul Yessendauletova    is a Senior Lecturer at Karaganda National Research University named after E.A. Buketov, Karaganda, Kazakhstan. Her academic interests include higher education, digital pedagogy, and competency-oriented instructional practices. She can be contacted at email: yesendauletova.zh@karnu-buketov.edu.kz.



Leilya Fazylova    is a Senior Lecturer at Karaganda National Research University named after E.A. Buketov, Karaganda, Kazakhstan. Her academic interests include educational technologies, computer science education, and student competency development in higher education. She can be contacted at email: fazylova.l@karnu-buketov.edu.kz.