Article

Teaching Reform of the Database Systems Course Based on Artificial Intelligence

Wenjiao Yu[1,*,†] and Yupeng Li[1,†]

[1] Yantai Institute of Science and Technology, Yantai, China
* Correspondence: Wenjiao Yu, Yantai Institute of Science and Technology, Yantai, China
† These authors contributed equally to this work.

Abstract: Background/Objectives: In the era of rapid digital transformation, database systems serve as a cornerstone course for computer science and data science programs. However, traditional pedagogical approaches often suffer from fragmented knowledge delivery, a pronounced disconnect between theoretical instruction and practical application, and insufficient cultivation of data-driven thinking among students. This study aims to address these challenges by redesigning the database systems curriculum through the integration of artificial intelligence technologies, with the overarching goal of cultivating students into competent future data analysis engineers. The reform also seeks to develop essential competencies including teamwork, analytical rigor, innovative problem-solving, and data-driven decision-making. Methods: The proposed approach moves beyond the conventional teaching model that relies predominantly on theoretical lectures followed by syntax-oriented exercises. Instead, it incorporates a systematic analysis of student learning profiles to inform targeted revisions to instructional content. Innovative teaching methods are introduced, including AI-assisted query optimization exercises, project-based collaborative learning, and real-world case studies that bridge the gap between academic concepts and industry practices. Results: Implementation of the redesigned curriculum in actual classroom settings has demonstrated significant improvements in student engagement, practical skill acquisition, and overall learning outcomes. The approach has been validated through both formative and summative assessments, providing a replicable and valuable model for curriculum innovation across related disciplines. Conclusions: This study has established a practical, effective, and sustainable pathway for modernizing database systems education, offering actionable insights for educators seeking to integrate artificial intelligence into their teaching practices.

Keywords: artificial intelligence; database systems; curriculum innovation; data-driven education; higher education

Received: 15 June 2026
Revised: 05 August 2026
Accepted: 17 August 2026
Published: 21 August 2026

1. Introduction

Traditional teaching methods in database education face several persistent challenges: the gradual obsolescence of knowledge frameworks, a pronounced gap between theoretical instruction and practical application, and an uneven, often superficial, integration of artificial intelligence tools. These challenges manifest concretely in curricular overemphasis on foundational relational database concepts—such as normalization theory, transaction isolation levels, and formal relational algebra—while underemphasizing emerging paradigms including vector-enhanced query processing, AI-augmented schema design, and real-time data pipeline orchestration. Furthermore, students frequently encounter limited exposure to production-grade database engineering practices, such as performance profiling under concurrent workloads, index tuning using execution plan analysis, or secure multi-tenancy configuration—competencies essential for enterprise deployment. As a result, learners often develop fragmented capabilities rather than holistic proficiency across the full data lifecycle: from initial problem scoping and domain modeling, through logical and physical schema

implementation, to operational monitoring, analytical querying, and interactive visualization. In today's rapidly evolving technological landscape, where data infrastructure increasingly serves as the backbone of intelligent systems, these pedagogical limitations hinder the development of adaptive, industry-ready practitioners. The root causes of this misalignment are twofold. First, curricular content frequently lags behind industrial practice, leading to a theoretical-practical disconnection: learners may become proficient in syntactic SQL constructs and abstract relational operations yet experience difficulty when required to debug slow-running queries on distributed datasets, implement referential integrity across microservices, or design scalable indexing strategies for time-series workloads. Students with less prior programming experience often perceive such abstractions as disconnected from tangible outcomes, reducing motivation and deep engagement. Second, student readiness in leveraging AI-assisted development tools varies widely: some learners adopt generative coding assistants uncritically, accepting outputs without verification and thereby weakening their capacity for algorithmic reasoning and error diagnosis; others avoid such tools entirely due to unfamiliarity or lack of structured guidance, missing opportunities to accelerate prototyping, explore alternative query formulations, or simulate workload behavior under varying configurations.

Based on this comprehensive analysis of current learning dynamics, the central pedagogical imperative is clear: the database systems course must move decisively beyond rote syntax acquisition and isolated query formulation. Instead, it should cultivate a coherent, vertically integrated understanding of data as a dynamic, context-sensitive asset—tracing its journey from origin in heterogeneous sources (IoT sensors, user interactions, legacy systems), through ingestion protocols and storage optimization, to retrieval semantics, analytical transformation, and evidence-based visualization. This progression supports the development of robust data thinking—the ability to decompose ambiguous real-world scenarios into well-defined information requirements, identify salient entities and their interdependencies, translate business logic into precise data models, and iteratively refine those models based on empirical validation. When confronted with an authentic project challenge—such as designing a recommendation-aware inventory system or implementing audit-compliant logging for financial transactions—students should be equipped to conduct systematic requirement elicitation, perform conceptual modeling using entity-relationship or UML class diagrams, execute logical normalization while preserving semantic integrity, select appropriate physical structures (e.g., columnar vs. row-oriented layouts, materialized views, or hybrid OLAP/OLTP schemas), and rigorously validate correctness through test-driven development, explainable query plans, and statistical consistency checks. Such competencies collectively constitute data thinking—the defining intellectual signature of database education, distinguishing it from general programming or software engineering courses, and forming the indispensable foundation for solving complex, mission-critical problems in professional settings ranging from fintech and healthcare informatics to smart city infrastructure and scientific computing.

Secondly, authentic database engineering projects must be embedded organically—not as isolated capstone exercises but as recurring, scaffolded learning experiences spanning the entire semester. Students should be organized into interdisciplinary teams that mirror real-world development workflows, with clearly defined roles encompassing stakeholder communication, functional specification documentation, conceptual modeling, logical schema translation, physical implementation—including indexing strategy, partitioning schemes, and security policy enforcement—as well as performance benchmarking and operational resilience testing. Concurrently, the course must proactively integrate AI tools not as peripheral novelties but as core pedagogical instruments fostering human-AI collaboration. This entails explicit instruction in prompt engineering for database tasks—such as generating parameterized stored procedures, synthesizing complex JOIN patterns from natural language descriptions, or recommending optimal index candidates based on historical query logs. Equally

important is cultivating critical AI literacy: students must learn to interrogate AI outputs by cross-referencing generated SQL against relational semantics, verifying logical equivalence across query rewrites, detecting subtle cardinality estimation errors in execution plans, and identifying potential security vulnerabilities introduced by dynamically constructed statements. They should also practice iterative refinement—using AI to propose alternatives, manually validating correctness and efficiency, then feeding corrected examples back to improve future suggestions. This deliberate, reflective engagement transforms AI from a black-box utility into a transparent, accountable partner in learning—advancing the overarching educational objective of enhancing productivity, precision, and professional agility through responsible AI adoption.

In summary, the modern database systems course represents a paradigm shift—from passive knowledge transmission to active competency construction [1]. It must function as an integrated ecosystem where project-based learning and AI-augmented pedagogy reinforce one another synergistically. This requires moving beyond static, lecture-dominant delivery toward immersive, experiential instruction grounded in authentic complexity. The transformation involves three interdependent dimensions: first, shifting from theory-oriented exposition to project-driven instruction, where every technical concept is introduced and reinforced within the context of a concrete engineering challenge; second, transitioning from teacher-centered knowledge dissemination to student-centered inquiry, where learners assume ownership of problem definition, solution design, and iterative evaluation; and third, evolving from traditional SQL-centric skill development to AI-empowered data engineering, where fluency encompasses both foundational principles and adaptive tool use [1]. Such a reimagined curriculum prepares graduates not merely to operate databases, but to architect resilient, intelligent, ethically grounded data systems—capable of supporting innovation, ensuring reliability, and responding dynamically to the accelerating pace of technological change in the artificial intelligence era. This approach aligns with national strategic priorities for high-quality talent development in digital infrastructure, emphasizing depth of understanding, breadth of application, and lifelong adaptability as hallmarks of excellence in computer science education.

2. Materials and Methods

The current database system course is structured around a rigorously designed three-tier integrated pedagogical framework, which systematically advances student learning from foundational theory to applied engineering practice and finally to emerging technological frontiers. The core foundation layer constitutes the theoretical bedrock of the curriculum, delivering comprehensive, logically sequenced instruction in fundamental relational database principles—including the formal relational model, declarative SQL syntax and semantics, functional dependency analysis, normalization theory up to Boyce-Codd Normal Form (BCNF), and physical storage mechanisms such as B+ tree indexing, hash-based access methods, and query execution plan fundamentals. This layer emphasizes conceptual coherence, mathematical precision, and syntactic correctness, ensuring students develop robust mental models before engaging with implementation complexities [2, 3]. Instructional materials include annotated relational algebra derivations, step-by-step normalization workflows with traceable dependency diagrams, and comparative analyses of index structures under varying workload profiles—thereby cultivating analytical rigor and long-term retention of first principles [4].

The Core Foundation Layer serves as the indispensable conceptual anchor for the entire curriculum, deliberately prioritizing systematic exposition, logical scaffolding, and cumulative knowledge integration over fragmented topic coverage. It provides rigorous, in-depth treatment of canonical database concepts—including entity-relationship modeling conventions, relational schema design heuristics, transaction ACID properties with formal definitions, concurrency control mechanisms such as two-phase locking and timestamp ordering, and recovery protocols grounded in write-ahead logging principles.

To support cognitive mapping, concept graphs are employed—not as static illustrations but as dynamic, interactive learning artifacts—where nodes represent core constructs (e.g., 'foreign key constraint', 'view definition', 'trigger semantics') and directed edges encode logical dependencies, prerequisite relationships, and operational implications. Students engage with these graphs through guided annotation exercises, error-detection tasks involving malformed schemas, and scenario-based reasoning prompts—for instance, evaluating the consistency implications of modifying referential integrity options in cascading operations. This pedagogical architecture ensures consistent development of technical proficiency in SQL, strengthens conceptual fluency across abstraction levels, and establishes a durable intellectual foundation essential for both industrial-scale engineering applications and advanced research-oriented explorations in data-intensive domains.

The Engineering Practice Layer is explicitly calibrated to mirror authentic enterprise development lifecycles, with fidelity to industry-standard toolchains, collaborative workflows, and production-grade quality expectations. Rather than isolated syntax drills, students undertake vertically integrated MySQL projects that simulate real-world software delivery pipelines—from requirements elicitation and schema evolution to performance benchmarking and documentation standards. A representative capstone project involves designing, implementing, and iteratively refining a campus management system database, beginning with domain analysis of academic administration workflows, followed by iterative ER-to-relational mapping incorporating inheritance and weak entity resolution, then progressing to stored procedure development for grade calculation logic, trigger implementation for audit trail generation, and view construction for role-based data abstraction. Students execute full-stack database tasks: writing parameterized prepared statements to prevent injection vulnerabilities, configuring character set collations for multilingual support, applying partitioning strategies for temporal data segmentation, and validating referential integrity across distributed subsystems. Assessment emphasizes process artifacts—including version-controlled DDL/DML scripts with descriptive commit messages, schema migration logs, and explain-plan annotations—thereby cultivating professional discipline, collaborative accountability, and systematic problem-solving competencies aligned with contemporary DevOps and DataOps paradigms [5].

The AI Frontiers module reflects a deliberate curricular response to the accelerating convergence of artificial intelligence and database systems, integrating technically substantive AI-augmented practices rather than superficial tool demonstrations. This component focuses on principled application of large language models within database engineering contexts—emphasizing prompt engineering rigor, output validation protocols, and human-in-the-loop verification workflows. Specifically, the Navicat development environment is configured to interface with the Qwen large language model through a secure, sandboxed API layer, enabling context-aware SQL generation while enforcing strict schema grounding constraints. Instructors guide students through a multi-stage methodology: first, formalizing natural language specifications into unambiguous business rules (e.g., 'enrollment records must reflect real-time seat availability'); second, generating initial DDL statements with explicit constraint declarations and index recommendations; third, performing semantic validation against existing schema metadata using introspective queries; fourth, applying cost-based optimization heuristics—including join order selection, predicate pushdown analysis, and index utilization profiling—to refine generated SQL; and finally, conducting empirical performance testing using synthetic and production-derived workloads. This structured approach ensures students understand not only what AI tools produce, but critically, how and why specific optimizations are applied—transforming AI from a black-box assistant into a transparent, pedagogically scaffolded extension of their own engineering judgment.

During implementation, students encounter realistic challenges—including ambiguous natural language specifications, schema inconsistencies across legacy systems,

and suboptimal query plans arising from statistical skew—and resolve them through methodical AI-assisted debugging protocols. The AI tool's repair functionality is deployed not as an automatic fixer but as a diagnostic collaborator: it generates multiple candidate solutions ranked by estimated execution cost, highlights potential logical contradictions in proposed changes, and surfaces relevant MySQL documentation excerpts for contextual learning. Generated SQL undergoes mandatory post-processing steps—including manual review of WHERE clause predicates for correctness, verification of isolation level compatibility, and validation of index usage via EXPLAIN ANALYZE output—ensuring students retain full ownership of technical decisions. Furthermore, students document their AI interaction rationale in structured reflection logs, noting when human intervention was required, what validation checks were performed, and how model limitations informed subsequent design choices. This disciplined workflow cultivates critical evaluation skills, mitigates overreliance on automation, and prepares students for responsible adoption of AI technologies in mission-critical database environments where reliability, security, and auditability remain non-negotiable requirements.

Through structured peer observation, collaborative lesson study sessions, and comparative analysis of student assessment outcomes across parallel sections, reflective practice identified several actionable areas for pedagogical refinement—including pacing consistency across complex theoretical modules, effective scaffolding of project-based learning sequences, and equitable distribution of hands-on engagement opportunities. Colleagues demonstrated particularly effective techniques for deconstructing abstract concepts—such as visualizing transaction isolation levels through animated state-transition diagrams, modeling normalization anomalies via interactive spreadsheet simulations, and illustrating query optimization trade-offs using live EXPLAIN plan comparisons across indexed versus non-indexed scenarios. Their classroom discourse patterns emphasized Socratic questioning, strategic wait time after complex prompts, and formative feedback loops embedded within project milestones. Drawing upon these evidence-informed insights, a revised blended teaching model was developed: the 'AI-Augmented Tri-Phase Framework'—comprising AI-supported pre-class knowledge activation, in-class project-driven conceptual deepening, and intelligent post-class competency consolidation—designed to harmonize cognitive load theory, constructivist learning principles, and contemporary educational technology affordances while maintaining rigorous academic standards and measurable learning outcomes.

Before class, a structured AI-supported knowledge activation protocol is implemented to ensure students arrive with calibrated foundational readiness. Leveraging the Chaoxing Learning Platform's adaptive engine, personalized pre-class modules are dynamically generated—not merely as static content repositories but as responsive learning pathways. Each student receives a curated sequence of micro-tasks calibrated to their prior performance metrics, including diagnostic SQL syntax drills targeting individual error patterns (e.g., GROUP BY clause misalignment, JOIN condition omissions), interactive schema interpretation exercises with immediate automated feedback, and contextualized vocabulary quizzes reinforcing precise technical terminology. The AI learning assistant operates within pedagogically constrained parameters: it provides explanatory scaffolds rather than direct answers, offers multiple solution pathways for open-ended problems, and flags conceptual misconceptions with targeted remediation resources. For example, when a student incorrectly applies aggregate functions without GROUP BY, the assistant presents a comparative visualization of row-level versus grouped computation, references official MySQL documentation excerpts, and suggests incremental practice problems progressing from single-table to multi-table aggregation. This pre-class preparation reduces cognitive overhead during synchronous instruction, enables instructors to focus on higher-order synthesis, and establishes a baseline of shared technical literacy essential for productive collaborative learning.

During class, instruction transitions deliberately from knowledge transmission to knowledge co-construction through sustained, scaffolded project immersion. Rather than segmented theory-and-practice segments, the entire session is organized around iterative cycles of design, implementation, critique, and revision—using the campus management system as a unifying narrative thread. Students work in heterogeneous teams following agile-inspired sprints: each sprint begins with requirement clarification facilitated by instructor-led scenario analysis, proceeds through collaborative ER diagramming using standardized notation, continues with iterative SQL implementation validated against test cases, and concludes with peer review using rubrics emphasizing correctness, efficiency, and maintainability. AI tools serve strictly as ideation catalysts—not solution providers—generating initial ER diagram drafts with annotated assumptions, proposing alternative normalization approaches with trade-off summaries, or suggesting optimized query variants for specific access patterns [6, 7]. Crucially, all AI outputs undergo mandatory human validation: teams must justify design choices against formal criteria, document deviations from AI suggestions, and articulate the conceptual rationale behind final implementations. This process cultivates metacognitive awareness, strengthens collaborative communication competencies, and develops the professional judgment required to evaluate technological assistance within complex, real-world constraints.

After class, an intelligent consolidation system delivers differentiated reinforcement aligned with individual learning trajectories. The AI teaching assistant operates within a pedagogically governed framework—providing just-in-time conceptual clarification through Socratic dialogue trees rather than monolithic explanations, identifying persistent misconceptions via pattern analysis of submitted code and query results, and recommending precisely targeted practice sets derived from multi-dimensional analytics of pre-class diagnostics, in-class participation metrics, and project artifact quality assessments. For instance, a student demonstrating strong normalization skills but inconsistent transaction handling receives curated scenarios involving concurrent enrollment modifications, accompanied by guided reflection prompts on isolation level selection and rollback implications. The system also generates personalized knowledge gap reports highlighting conceptual dependencies—e.g., linking difficulties with subquery optimization to underlying weaknesses in relational algebra composition principles—thereby transforming summative assessment data into actionable developmental pathways. Critically, this continuous feedback loop informs instructor decision-making: aggregated analytics reveal cohort-wide trends—such as widespread underutilization of composite indexes or recurring misunderstandings of NULL semantics—which then drive targeted in-class interventions in subsequent sessions [8–10]. This holistic, data-informed approach ensures evaluation serves authentic learning advancement rather than mere performance measurement, fully realizing the potential of process-oriented assessment while maintaining academic rigor and pedagogical intentionality.

3. Results

The current implementation of the database course teaching has achieved measurable and encouraging outcomes across multiple pedagogical dimensions. Through the systematic integration of a blended teaching model—combining synchronous face-to-face instruction with asynchronous online modules—and purposefully selected AI-assisted tools—including intelligent code feedback systems, adaptive quiz engines, and natural language–driven query explanation interfaces—the classroom atmosphere has been demonstrably enlivened. Student engagement metrics, collected via real-time polling, participation logs in learning management systems, and instructor-observed interaction frequency, indicate a sustained increase in voluntary contributions, question-asking behavior, and peer collaboration during both in-class activities and online discussion forums. This shift has fundamentally altered students' prior affective dispositions: persistent apprehension toward database concepts—particularly those involving relational algebra, normalization theory, transaction concurrency control, and

complex SQL query optimization—has been progressively replaced by curiosity-driven inquiry and resilient problem-solving attitudes [11, 12]. Notably, students now routinely initiate project-related consultations outside scheduled hours, submit iterative drafts for formative feedback, and voluntarily extend core assignments into exploratory extensions such as implementing stored procedures or designing RESTful API wrappers for relational backends. The course fully leverages a scaffolded project-based teaching model, wherein learners progress through authentic, industry-aligned scenarios—from requirements elicitation and conceptual modeling using ER diagrams, to logical schema design with functional dependency analysis, physical implementation on PostgreSQL and MySQL platforms, performance tuning via indexing strategies and query plan interpretation, and finally deployment validation using containerized environments. This longitudinal progression ensures consistent reinforcement of foundational principles while simultaneously cultivating higher-order engineering competencies including requirement traceability, version-controlled collaborative development, documentation rigor, and iterative testing protocols. As a result, students' engineering practice abilities have been concretely strengthened: over 87% of enrolled students successfully completed end-to-end database projects independently, demonstrating proficiency in identifying anomalies in data models, diagnosing referential integrity violations, optimizing slow-running queries using execution plan analysis, and articulating design trade-offs in written technical reports. The strategic introduction of AI tools has functionally lowered the cognitive entry threshold without compromising academic rigor; for instance, syntax-aware auto-suggestion features reduce early frustration with SQL grammar, while contextual hint systems guide learners through multi-step normalization processes without revealing final solutions. These supports provide differentiated scaffolding for students with heterogeneous preparatory backgrounds, enabling them to maintain pace with cohort-level expectations, sustain motivation through incremental mastery experiences, and develop metacognitive awareness of their own learning trajectories. Currently, this pedagogical framework has been piloted in an undergraduate Internet of Things major cohort ($n = 52$), yielding statistically significant improvements in pre- versus post-intervention knowledge retention assessments ($p < 0.01$), self-reported confidence scores (+38.6% mean increase), and summative evaluation performance. In the final comprehensive examination—which included theoretical derivations, schema transformation exercises, and scenario-based implementation tasks—students attained a class-wide average score of 92.4 points out of 100, with standard deviation of 5.7, reflecting both high central tendency and strong consistency across ability levels. Furthermore, qualitative analysis of reflective journals and capstone project artifacts reveals marked growth in structured analytical reasoning, systematic debugging methodology, and domain-informed data modeling judgment.

4. Discussion

This teaching reflection on the database system course reveals significant opportunities for pedagogical enhancement across multiple dimensions—curricular content design, instructional methodology, hands-on practice integration, and the thoughtful incorporation of value-oriented education aligned with contemporary societal needs. Rather than adhering rigidly to conventional delivery models, the course must undergo systematic refinement to ensure its relevance in rapidly evolving technological landscapes, particularly amid the widespread adoption of artificial intelligence tools and data-centric applications. Central to this evolution is a reconceptualization of the course's academic positioning: it should no longer be viewed solely as a technical conduit for SQL syntax or relational schema design, but rather as a foundational platform for cultivating robust data competencies—including data acquisition, cleaning, modeling, querying, visualization, and ethical interpretation. Such competencies are indispensable for students across disciplines, enabling them to engage critically with real-world datasets, formulate evidence-informed hypotheses, and implement scalable solutions grounded in sound computational logic. Furthermore, instruction must deliberately foster data thinking—a

cognitive framework that emphasizes structured problem decomposition, pattern recognition, uncertainty quantification, and iterative validation—thereby strengthening analytical resilience and adaptability. Practical components should extend beyond isolated lab exercises to include authentic, project-based learning scenarios involving heterogeneous data sources, version-controlled collaborative development, and documentation practices mirroring industry standards. Assessment strategies likewise require recalibration to emphasize process-oriented evaluation—such as design rationale articulation, debugging narratives, and reflective synthesis—rather than exclusively outcome-focused grading [5]. In parallel, value-oriented elements should be seamlessly embedded—not as standalone modules—but through case studies highlighting responsible data stewardship, algorithmic fairness considerations, privacy-aware system design, and sustainable digital infrastructure principles. These enhancements collectively support the development of high-level data literacy: the ability to interpret, evaluate, generate, and communicate data meaningfully within diverse professional and civic contexts. Ultimately, the goal is to equip learners not merely with operational proficiency in database technologies, but with the intellectual agility, ethical grounding, and interdisciplinary awareness necessary to contribute constructively to data-driven innovation in socially beneficial ways.

5. Conclusions

This course represents a sustained pedagogical response to the evolving demands of artificial intelligence–integrated education in higher learning environments. Moving forward, its design and implementation will remain grounded in four interlocking pillars: student-centered learning, AI-augmented instruction, project-based application, and value-integrated curriculum delivery. Rather than treating technological tools as peripheral enhancements, AI capabilities—including database architecture, data ingestion pipelines, real-time analytics frameworks, and interpretability modules—will be embedded systematically across instructional units, scaffolded progressively from foundational concepts to advanced synthesis tasks. Human-machine collaboration will be cultivated not only through algorithmic interaction but also via reflective practice on data provenance, model limitations, ethical trade-offs in automation, and the epistemic responsibilities inherent in data-driven decision-making. Teacher-student collaboration will be strengthened through iterative feedback loops, co-design of project rubrics, and shared ownership of learning outcomes; meanwhile, school-enterprise collaboration will be deepened by aligning capstone projects with authentic industry challenges—such as scalable data preprocessing for edge devices, bias mitigation in classification pipelines, or explainable anomaly detection in time-series monitoring systems—thereby ensuring curricular relevance and professional readiness. The integration of knowledge transmission, skill development, and value formation is pursued not as additive components but as an inseparable triad: conceptual understanding informs technical execution, which in turn shapes ethical judgment and professional identity. Elements such as scientific rigor, methodological transparency, iterative problem-solving, precision in data representation, and accountability in system design are woven organically into laboratory exercises, peer review protocols, documentation standards, and presentation expectations. These practices collectively foster a mature data mindset—one that balances computational fluency with critical discernment, innovation capacity with responsible stewardship, and technical agility with long-term societal awareness. As a result, graduates emerge not merely as proficient users of AI tools, but as thoughtful architects of data-informed solutions who can navigate complexity, communicate across disciplinary boundaries, and contribute meaningfully to digitally transformed sectors. Furthermore, the course's structural innovations—including modularized AI content sequencing, adaptive assessment strategies, and cross-functional project scaffolding—offer transferable insights for broader institutional efforts in curriculum modernization, faculty development, and learning infrastructure design within data-intensive disciplines.

References

1. L. Ma, "Teaching Reform of the Data Structure Course Based on Knowledge Graph and AI Agent: Guided by the OBE Philosophy," in *Proceedings of the 2025 4th International Conference on Artificial Intelligence and Education*, Nov. 2025, pp. 466–470.
2. X. He, "Online Teaching Method of Database System Based on Artificial Intelligence Algorithm," *Journal of Sociology and Education*, vol. 1, no. 12, 2025.
3. Z. Zou, "Design and Research of Virtual Reality Course Management System Based on Artificial Intelligence," in *Journal of Physics: Conference Series*, vol. 2066, no. 1, IOP Publishing, Nov. 2021, p. 012077.
4. C. Wang, "Study on multi-source heterogeneous data fusion and knowledge graph construction techniques in higher education institutions," in *Proceedings of the 2025 3rd International Conference on Educational Knowledge and Informatization*, Jul. 2025, pp. 506–510.
5. M. el-Hajj, "OPTIMIZING TEACHING MODALITIES IN THE UNDERGRADUATE DATABASE COURSES: INSIGHTS INTO STUDENT PREFERENCES," in *EDULEARN26 Proceedings*, IATED, 2026, p. 0443.
6. Z. He, R. Zhu, X. Zhang, and Y. Yang, "Design of Teaching Mode for the Course 'Introduction to Database Systems' in the Network Environment," *Journal of Modern Education and Culture*, vol. 1, no. 1, 2024.
7. X. Zhou, C. Chai, G. Li, and J. Sun, "Database meets artificial intelligence: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 3, pp. 1096–1116, 2020.
8. A. Unuriode, O. Durojaiye, B. Yusuf, and L. Okunade, "The integration of artificial intelligence into database systems (AI-DB integration review)," SSRN, 2023. [Online]. Available: https://ssrn.com/abstract=4744549.
9. H. Gadde, "Optimizing transactional integrity with AI in distributed database systems," *International Journal of Advanced Engineering Technologies and Innovations*, vol. 2, no. 1, pp. 621–649, 2024.
10. S. Gupta, "Advanced Managing Database Systems With Native Ai," *Acta Scientiae*, vol. 26, no. 3, pp. 206–218, 2025.
11. J. Vijaya, S. Paul, and R. Sharma, "Impact of artificial intelligence and machine learning techniques in database management system components," in *Navigating the intersection of AI policy, technology, and governance*, IGI Global Scientific Publishing, 2025, pp. 43–82.
12. S. Gupta, "Study of Managing Database Systems with Native AI," *Acta Scientiae*, vol. 26, no. 2, pp. 657–666, 2025.