

ExPeerience: Towards AI-Assisted Learnersourcing to Bridge Conceptual Understanding and Problem Solving in Database Programming Education

Yuzhe Zhou
School of Applied and Creative
Computing
Purdue University
West Lafayette, Indiana, USA
zhou922@purdue.edu

Prithvi Manjunatha Babu
School of Applied and Creative
Computing
Purdue University
West Lafayette, Indiana, USA
manjunp@purdue.edu

Udayan Pandey
Purdue University
West Lafayette, Indiana, USA
pandey64@purdue.edu

Alejandra J. Magana
School of Applied and Creative
Computing
Purdue University
Lafayette, Indiana, USA
admagana@purdue.edu

Tianyi Li
School of Applied and Creative
Computing
Purdue University
West Lafayette, Indiana, USA
li4251@purdue.edu

Abstract

Learnersourcing, an educational approach that positions students as active contributors rather than passive consumers, offers a scalable approach to co-creating instructional resources while engaging students in authentic problem-solving. However, it faces a fundamental tension: effective “learning” requires scaffolding that minimizes extraneous cognitive load and focuses attention on reasoning, while effective “sourcing” requires structure, completeness, and standardization to ensure student-generated content can be reused. These competing goals create a tradeoff: students either learn but produce content that is difficult to reuse, or generate usable resources but receive limited learning benefit. We propose a new AI-assisted learnersourcing paradigm to address this tension. By assigning collaborative roles to both learners and AI, the approach enables students to focus on cognitively meaningful sub-tasks that foster “learning”, while large language models (LLMs) handle mechanical and procedural sub-tasks for “sourcing”. Guided by user-centered design principles, we implement this workflow in ExPeerience, a system that scaffolds students in co-creating contextualized worked-out examples for database programming. Within ExPeerience, the AI serves as a collaborator for ideation, a co-creator of artifacts, and an evaluator of students’ inputs. Our evaluation with 24 participants showed that structuring AI into distinct collaborative roles improves learning engagement while producing high-quality student-generated content. Compared to a baseline using the Gemini chatbot, ExPeerience users created SQL problems in more diverse and personally meaningful contexts. They actively evaluated, edited, and refined AI-generated components, and most authored

their own SQL solutions, whereas baseline participants largely accepted AI outputs without modification and did not attempt to solve the problem. Overall, ExPeerience produced more contextualized, varied, and thoughtfully constructed worked-out examples. These findings demonstrate the potential of AI-assisted learnersourcing as a paradigm to balance learning and sourcing goals. We also draw design implications for future AI-assisted learnersourcing systems that aim to produce reusable, high-quality learner-generated content while promoting educational value.

CCS Concepts

• **Applied computing** → **Interactive learning environments; Computer-assisted instruction; Collaborative learning**; • **Human-centered computing** → **Human computer interaction (HCI)**.

Keywords

Human-AI Collaboration, Database Programming, Learnersourcing, Student-Generated Content, Worked-Out Example

ACM Reference Format:

Yuzhe Zhou, Prithvi Manjunatha Babu, Udayan Pandey, Alejandra J. Magana, and Tianyi Li. 2026. ExPeerience: Towards AI-Assisted Learnersourcing to Bridge Conceptual Understanding and Problem Solving in Database Programming Education. In *31st International Conference on Intelligent User Interfaces (IUI '26)*, March 23–26, 2026, Paphos, Cyprus. ACM, New York, NY, USA, 24 pages. <https://doi.org/10.1145/3742413.3789139>

1 Introduction

Problem-solving in database programming is challenging because students often struggle to translate their knowledge of SQL syntax and query structures into solutions for authentic, domain-specific problems [46, 47, 69]. While many students can correctly construct queries, they frequently face difficulty interpreting real-world scenarios and designing solutions that accurately reflect underlying data relationships. Worked-out examples (WEs), which consist of a problem statement, step-by-step solutions, and supporting representations, provide an effective mechanism for scaffolding this



This work is licensed under a Creative Commons Attribution 4.0 International License.
IUI '26, Paphos, Cyprus
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1984-4/26/03
<https://doi.org/10.1145/3742413.3789139>

learning [70]. Extending WEs with contextualization further helps students connect abstract computing concepts to realistic scenarios, and foster deeper problem-solving skills [58]. However, ideating authentic problem contexts relatable to students and translating diverse contexts into learning materials designed to fulfill course-specific learning objectives requires extensive instructor efforts [32].

Learnersourcing has shown promise as an educational approach by positioning students as co-creators rather than passive recipients [1, 31, 32, 34]. By engaging students in the generation of worked-out examples, explanations, or problem contexts, learnersourcing can scaffold the translation of abstract SQL syntax into meaningful solutions for realistic, domain-specific scenarios [80]. This approach encourages students to reflect on computational thinking, evaluate alternative solutions, and articulate their knowledge for others, thereby deepening conceptual understanding and improving metacognitive skills [22, 24, 31, 32, 34].

Despite its potential, learnersourcing faces challenges when applied to complex, domain-specific tasks such as engaging students to co-create contextualized WEs for database programming subjects. Students may lack the scaffolding to effectively generate meaningful WEs or to contextualize abstract concepts in realistic problem settings. Building on prior work, we investigate how learnersourcing can be structured to support problem-solving learning objectives in database programming, using contextualized WEs as the central artifact for both learning and co-creation.

Through a formative student-centered design process, we identified challenges and leveraging points for scaffolding the learnersourcing process. Students often struggle to translate SQL knowledge into solutions for authentic, domain-specific problems, despite being able to write syntactically correct queries. Although offering a promising way to bridge this gap by encouraging reflection, evaluation, and articulation of knowledge, learnersourcing faces challenges: open-ended tasks can overwhelm students and produce inconsistent or incomplete materials.

To overcome these challenges, we developed **ExPeerience**, an AI-assisted learnersourcing system to balance the dual goals of learning and sourcing. The system design integrated just-in-time scaffolding to support students in identifying relatable problem contexts, formulating clear problem descriptions, generating representative data automatically, and receiving feedback on their solutions within a cohesive and integrated interface. ExPeerience distributes cognitive responsibilities between students and large language models (LLMs): students focus on higher-order reasoning tasks such as composing solutions and explanations, while the AI handles procedural subtasks, including data generation and evaluative feedback. From a pedagogical perspective, this division aligns with Bloom’s Taxonomy [37], shifting students’ cognitive focus away from routine procedural tasks toward complex problem-solving and knowledge articulation.

We evaluated the latest design of ExPeerience with 24 participants from a large land grant university. We found that the system outperforms the baseline condition across the context quality of learnersourced materials, learner experience, and learning outcome. Students using ExPeerience produced more contextually grounded and diverse WEs with richer narratives, coherent data structures, and step-by-step explanations that reflected their own reasoning

and understanding, whereas the baseline group commonly use content generated by the AI without modification or review. ExPeerience group also reported higher usability and a stronger sense of ownership of the materials co-created with AI support. Students tend to iteratively refine AI-generated outputs instead of relying on them wholesale. ExPeerience participants also demonstrated positive trends in confidence gains in concept understanding, application, and problem-solving.

This paper makes the following contributions:

- (1) **Structured student-AI co-creation workflow:** We present a workflow that engages students in producing learning materials contextualized in relatable scenarios, illustrating how different AI roles provide scaffolding across ideation, creation, and evaluation to support learnersourcing.
- (2) **AI-augmented learnersourcing system:** We introduce ExPeerience, a learnersourcing system enhanced with LLM-based support to help students co-create WEs in database programming.
- (3) **Empirically grounded design principles:** Through iterative, learner-centered design, we identify trade-offs in learnersourcing workflows, highlighting tasks that maximize “learner” benefit versus tasks that primarily fulfill “sourcing” requirements, and show how these insights inform system features.
- (4) **Design implications for AI-supported learnersourcing systems:** Based on pilot and final evaluations, we provide evidence-based recommendations for incorporating AI into learner-centered learnersourcing workflows, showing how AI can support meaningful student engagement, reduce routine effort, and enhance learning outcomes.

2 Related Work

2.1 Database Programming Education

Database programming is a foundational skill in computer science and information systems curricula, where SQL and related languages remain central to students’ learning [46, 47, 69]. Beyond syntax, students must interpret data schemas, reason about relationships, and translate information needs into query logic. However, SQL is often taught in what Taipalus and Seppänen [69] term “unnatural learning environments,” using simplified schemas and predefined results. While effective for introducing syntax, such exercises limit exposure to authentic, context-rich problems-leading to persistent misconceptions and incomplete mental models of relational logic [3, 48]. Conversely, authentic database problems are conceptually valuable but can overwhelm novices with complexity [60, 78].

Prior research has applied diverse pedagogical strategies, including project-based learning [62, 65] and contextualized instruction [15, 63]. These methods immerse students in realistic problem-solving, prompting them to apply SQL concepts within meaningful contexts. More recent studies integrate visualization tools and AI-based feedback to help students understand query execution and relational dependencies [16]. Despite these advances, designing authentic, contextualized exercises remains labor-intensive for instructors and challenging to scale, as each problem must maintain

both relevance and data consistency. Our work extends this literature by introducing an AI-supported learnersourcing approach that engages students to co-create contextualized learning materials.

2.2 Learnersourcing and Worked-Out Examples (WEs) in Computer Science Education

Guided by Singh et al.'s framework for studying, designing, and deploying learnersourcing systems [67], two central questions arise: “what is being produced?” and “what are the contributors learning through the task?” Prior work has explored the creation of various types of educational content, including multiple choice questions [13], coding exercises [55], programming hints [22], and peer explanations [24]. In this work, we explore using worked-out examples (WEs) as the main learning artifact. A well-designed WE typically integrates a clearly articulated problem, a structured step-by-step solution procedure, and supplementary representations that support learners’ understanding of both the task and its resolution [71]. Empirical studies in computer science education have demonstrated that WEs can improve students’ problem-solving performance and learning transfer [64, 80]. Based on Muldner et al.’s review [50], there are two main forms of WEs in computing contexts: code-tracing examples, which illustrate program execution processes [8, 17], and code-generation examples, which present complete solutions to programming problems [43, 44]. In our study, we primarily focus on code-generation worked-out examples, while also encouraging students to include intermediate reasoning steps that make execution-level thinking explicit.

From the learning aspect, prior studies have shown that learnersourcing activities can deepen understanding by prompting students to reflect on their reasoning [22, 34], evaluate alternative solutions [24, 73], and articulate knowledge for others—thereby strengthening metacognitive awareness and conceptual retention [31, 32]. To better structure and scale these processes, various learnersourcing systems have been developed, such as PeerWise [11], RiPPLE [33], and AXIS (Adaptive eXplanation Improvement System) [75]. Building on this foundation, our work employs AI to assist students in the learnersourcing process by providing just-in-time scaffolding throughout content creation. While recent versions of RiPPLE have incorporated generative AI to offer feedback on student-created learning resources [56], our approach differs in the scope and timing of AI involvement. Rather than limiting AI feedback to post hoc evaluation, our system assigns distinct roles to AI across the learnersourcing workflow during the creation process.

2.3 Human-AI Co-Creation and Student-AI Interaction

Recent advances in AI have driven growing research interest in human-AI co-creation within the Intelligent User Interfaces (IUI) community. This interest is reflected in a sustained series of workshops on Human-AI Co-Creation with Generative Models, held in conjunction with IUI over multiple years [19–21, 42, 49, 74]. Prior IUI research has explored human-AI co-creation across a range of domains, including music [4, 7, 45, 77], writing [4, 77], programming [51, 52], and UX design [26]. Across these domains, studies have examined how AI systems can act as collaborators rather than

passive tools, supporting users in tasks such as ideation, sensemaking, and content production while preserving human agency and decision-making [23, 35, 59, 66]. In the education field, student-AI collaboration [9, 10, 14, 57] have drawn growing attention in three areas: *AI for Ideation*, where models scaffold creativity and brainstorming; *AI for Content Creation*, where they generate learning materials; and *AI for Evaluation*, where they provide adaptive feedback.

AI for Ideation. LLMs can act as generative partners that propose initial ideas or analogies based on student input [39, 72]. Unlike fully automated content generation, which risks homogenizing ideas and reducing learner agency [53], AI-supported ideation tools can preserve ownership while lowering barriers to creativity. Studies in design and writing education have shown that AI-augmented ideation can reduce creative anxiety, promote divergent thinking, and stimulate reflection on problem framing [10, 57].

AI for Content Creation. Becker et al. [5] highlighted AI code generation tools for producing exemplar solutions and scalable learning resources. Farinetti and Cagliero [16] used LLMs to generate SQL problems and explanations, encouraging critical evaluation while exposing risks of hallucination. LLMs have also been applied to create step-by-step worked examples [28], generate Bloom-aligned assessment questions [27], and support lecture design, as in VIVID [6]. While these studies demonstrate AI’s potential to scale content creation, concerns remain regarding accuracy and pedagogical validity. Gilardi et al. [12] cautioned that LLM outputs, though coherent, often include biases and errors, reinforcing the need for validation.

AI for Evaluation. In parallel, AI has also been used to support evaluation by generating feedback and guiding student learning. Both open- and closed-source models can produce programming feedback with comparable quality [36]. Real-time LLM feedback helps learners identify errors, though it may also reinforce misconceptions [30]. Classroom deployments such as the CS50 Duck [41] and UC Berkeley’s 61A Bot [79] show reduced task time and targeted support. Beyond code, AI has been applied in database education, as in Riazi and Rooshenas [61], who generated feedback on ER diagrams through structured analysis and iterative guidance.

Building on this body of research, our work extends student-AI collaboration into a learnersourcing context, where the AI takes on complementary roles across multiple stages of content creation. An earlier IUI work on human-AI co-creation of worked examples [25] primarily focused on co-authoring code and corresponding explanations with AI assistance. In that work, instructors or teaching assistants served as the primary human collaborators. In contrast, our work shifts the locus of co-creation to students themselves and centers on contextualization and problem-solving.

3 Student-Centered Design Process

Following user-centered design principles [2], we adopted a student-centered approach consisting of three phases: needs finding, initial system design, and pilot evaluation. Insights from these phases informed the final AI-assisted learnersourcing workflow (see Figure 5) and the ExPeerience system (see section 4). The study procedure

was reviewed and approved by the Human Research Protection Program (IRB) in the author’s institution.

3.1 Phase 1: Needsfinding through Classroom Experience

In the first phase, we iteratively designed a learnersourcing workflow guided by both theoretical frameworks and empirical student needs elicited from classroom deployment throughout four semesters.

3.1.1 Procedure. Guided by Gance’s constructivist principles [18] and Sweller’s cognitive load theory [68], we decompose the learnersourcing process for co-creating contextualized WEs for database programming into three components: (1) identifying lived experiences relevant to their learning, (2) applying computational knowledge within these contexts, and (3) developing computational solutions to the problem.

To scaffold students’ co-creation process and enhance learning, we iteratively refined the workflow over four semesters in an upper-level database programming course. Detailed results are reported in our prior work [40, 80]. Each semester followed a structured *define-design-deploy-evaluate* cycle (Table 1):

- **Define:** Specify the learnersourcing procedure and contexts based on literature and prior semester insights.
- **Design:** Develop scaffolding artifacts, instructional materials, and integrate platform features to support co-creation.
- **Deploy:** Implement the learning activities in the classroom.
- **Evaluate:** Collect student feedback and learning performance data to identify areas for improvement, including clarity, scaffolding effectiveness, platform usability, and reflection support. Insights are then incorporated into the next cycle’s *define* stage.

3.1.2 Results. We synthesized the results from across the four semesters into five primary *Challenges (C)* and translated each challenge into a corresponding **design goal (DG)**.

C1: Inconsistencies and Gaps Across Example Components. Although students could identify contexts and define problems with the improved workflow, they often struggled to design solutions that were consistent with the problem description, data schema, and intended learning objective. This difficulty likely stems from

the high working memory demands of constructing WEs during learning.

DG1: Reduce cognitive load The system should reduce students’ cognitive load during the creation process, allowing students to focus on meaningful problem-solving rather than juggling multiple complex tasks simultaneously.

C2: Lack of clear structure in problem formulation. Students’ problem statements were often brief, abstract, and lacking the necessary background, particularly when domain-specific knowledge was needed.

DG2: Introduce explicit structure in assignment instructions. The system should visually guide students through each component to scaffold the creation process step by step to produce richer, clearer problem contexts and descriptions.

C3: Manual creation of data tables distracts students from learning objectives. Manually generating tables and dummy data was tedious and often yielded incomplete or low-quality outputs, diverting attention from deeper computational reasoning.

DG3: Automate structured data generation to reduce workload. The system should handle tedious table creation and dummy data generation, freeing students to focus on core learning tasks.

C4: Inconsistent and low-quality coding solutions. Students frequently submitted plain-text solutions or pseudocode instead of executable code, which led to inconsistencies and reduced the evaluability of their WEs.

DG4: Require step-by-step executable solutions. The system should structure solution development in ways that allow students to organize reasoning and executable code into distinct, clearly defined steps.

C5: Students struggled to iteratively improve their WEs across multiple sessions. Due to task complexity, many students could not complete examples in a single session and resorted to external documents, risking loss of progress and context.

DG5: Save progress and improve the work across multiple sessions. The system should allow students to save work and iteratively refine examples across multiple sessions.

Table 1: Summary of four iterative define-design-deploy-evaluate design cycles.

Define/Improvements	Design	Deployment	Evaluation
Two contextualized problems with data tables and solutions, one easy and one hard	Brightspace LMS Quiz with an instructor-provided example	Spring 2022 (N = 23)	Frequent omissions of key components (e.g., data tables, coding solutions), limited formatting, minimal input validation
Improved structure; clearer submission expectations	Structured homework assignment with submission template and peer-review platform	Fall 2022 (N = 45)	Persistent variability in submission quality; key components still omitted
Significantly improved completeness and consistency; fewer errors	Qualtrics survey with mandatory input fields and verification pages	Spring 2023 (N = 23)	Usability concerns (limited code formatting support, cumbersome multi-step submission, risk of losing work)
Consistent high-quality submissions; improved accuracy and organization	Refined Qualtrics implementation with additional verification and clearer instructions	Fall 2023 (N = 54)	Continued usability challenges; students rely on external editors to prepare submissions before entry into Qualtrics

3.2 Phase 2: Initial ExPeerience Interface Design

Guided by the design goals developed from Phase 1, we designed and developed a web-based application to facilitate the learnersourcing process.

3.2.1 Design Workshop. We conducted a design workshop with a team of one senior HCI researcher, one HCI PhD student, one UX design Master’s student, and one undergraduate researcher to ideate interface designs for each design goal. The workshop comprised three sessions.

In the first session, the team reviewed and clarified the challenges and design goals, and brainstormed information architectures and features to address each design goal. During the second session, each member independently sketched their design concepts on a whiteboard (see Figure 1) separately. In the third session, team members presented their designs, and the group discussed the rationales, advantages, and limitations of all proposed designs.

After the design workshop, the UX Master’s student synthesized these ideas into a low-fidelity prototype (see Figure 2) and presented it for group feedback on workflow, interface clarity, and feature feasibility. Insights from this discussion guided refinements that informed the development of the initial version of ExPeerience.

3.2.2 Initial Design of ExPeerience. ExPeerience is implemented using the Django framework¹, integrated with a PostgreSQL database, and HTML, CSS, JavaScript, and Bootstrap for the front end. The initial design of the system scaffolds the learnersourcing process with four primary views.

The **Assignment List** (Figure 3a) lists all open assignments, with each assignment tied to a specific topic. Additionally, any incomplete or saved WEs appear under the section *The Worked-Out Examples In Progress* [DG5] (currently empty, as depicted in Figure 3a).

Selecting an assignment leads students to the **Worked-Out Example Workspace** [DG1, DG2]. This workspace provides a floating navigation menu ([DG2], Figure 3b) that guides students through problem background, data (Figure 3c), and solution ([DG4], Figure 3d). Each development step includes a *preview panel* [DG1] that dynamically shows the current state and appearance of the overall WE.

Data Generation Page [DG3]. The data generation page employs a structured form for users to create data tables. Each data table required a table name. Within each table row, attribute names allow for free-text entry, while attribute types and constraints are chosen from drop-down menus with predefined options. We implement the automatic data generation with the Gemini 1.5 Flash model². Sample prompts used are provided in Appendix A.

Completeness Validation and Progress Saving [DG1, DG5]. By clicking the **Save and Submit** button, the system validates required fields and provides clear warnings for any fields that do not meet the specified requirements [DG1]. Clicking the **Save and Exit** button saves users’ progress [DG5] and lists incomplete WEs in the Assignment View (Figure 3a).

3.3 Phase 3: Evaluation of Initial Design

We evaluated the initial design of ExPeerience from November to December 2024 by comparing it with a baseline condition in which participants completed the same workflow steps using a Qualtrics-based survey interface. Participants in each condition were tasked to create WEs. Instructions were consistent between the Qualtrics survey and the ExPeerience interface. We focused exclusively on comparing the worked-out example creation interfaces in the two conditions, without evaluating other system features like *Assignment List*.

3.3.1 Participants. We recruited 17 undergraduate students from a database programming course at a large land-grant university. Among the 17 participants, 16 majored in Computer and Information Technology, and one majored in Construction Management Technology. All participants were juniors ($N = 6$) or seniors ($N = 11$). The participant pool consisted of nine male and eight female students. All participants had taken database-related courses. The participants are randomly assigned to the ExPeerience and baseline (Qualtrics) conditions, with nine using ExPeerience and eight using Qualtrics to create a WE.

3.3.2 Procedure. Each study session lasted approximately 60 minutes and was conducted remotely via Zoom or Microsoft Teams. Sessions began with a pre-test survey assessing participants’ confidence and proficiency in the given database programming concepts (e.g., PL/SQL FOR loops) as well as their prior experience with learnersourcing and creating learning materials.

Next, participants were shown a sample WE created by the research team before being asked to create their own. Participants using ExPeerience completed their examples through the **Worked-Out Example Workspace**, while those in the baseline condition used guiding questions provided in a Qualtrics survey. During the creation process, participants shared their screens so researchers could observe their workflow.

After completing the task, participants took a post-test survey that mirrored the pre-test questions on confidence and proficiency, and also included items on usability and demographics. Each session concluded with a brief five-minute interview, in which researchers asked follow-up questions about the creation process. All sessions were audio- and screen-recorded for analysis.

3.3.3 Results. One participant provided incomplete responses and identical ratings in both the pre- and post-test surveys. Thus, their data were excluded, leaving seven participants in the baseline condition.

To evaluate the effectiveness of ExPeerience, we adapted Singh et al.’s framework for studying, designing, and deploying learnersourcing systems [67] and assessed ExPeerience across three dimensions: **content quality**, **learner experience**, and **learning outcomes**.

Content Quality is Varied Within Both Conditions. The content quality of the participant-created WEs was evaluated based on the four components of the WEs.

For the *problem context* component, most participants chose contexts distinct from course assignments, ranging from hobbies (e.g., Pokémon card collections) to student life (e.g., class registration, grade tracking). Three participants, two from Qualtrics and one

¹<https://www.djangoproject.com/>

²<https://cloud.google.com/vertex-ai/generative-ai/docs/learn/models>

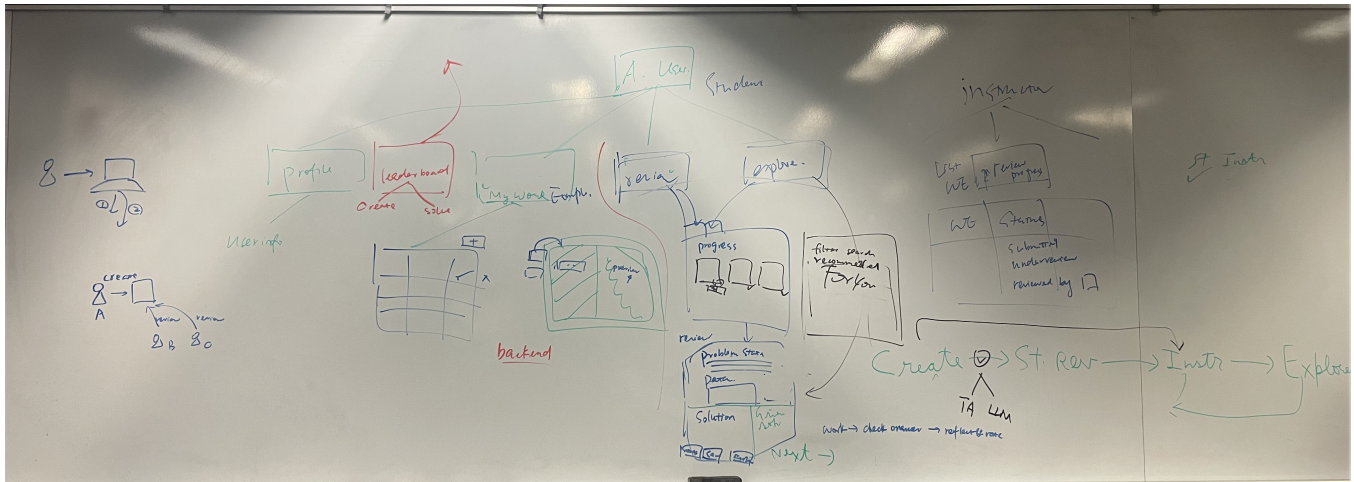


Figure 1: Whiteboard sketch from the design workshop illustrating the user flow and information architecture of the ExPeerience system.

Profile

Hello UserName, show your expertise to the world!

Home | Sign In | [Create a Worked-Out Example](#)

Good morning! afternoon! evening! Username

Worked-out Example ID: XXXN001N, Topic: Basic PL/SQL

Problem Statement

Data

Solution

Self-Review

Topics Practiced in this Worked-Out Example

Basic PL/SQL

Functions

Procedure

Triggers and Constraints

Others

Problem Context

Among the many attractions, Hong Kong is famous for its diverse range of dessert places that offer everything from traditional sweet soups to modern and innovative desserts. There are a many dessert places to choose from, each with its own unique offerings and specialties. There are few iconic desserts that you would like to try and you want to find a restaurant that accels across multiple items on your list of favorites.

Edit

Save

Submit

Need Help? Contact Us

Preview

Qui labore ut velit non. Necessitatibus et qua quam laboris. In nobis magni iusto tenetur ornisi.
Consequatur aliqua non dignissimos. Sed veritatis ea dolorem voluptate impedit deserunt. Dolorebus ornisi impedit non nemo positum ipsum voluptatis. Detectus voluptatibus deserunt laborum necessitatibus necessitatibus recundantur. Aut ipsa minus similis similis, tunc et ratione qua enim neque suscipit voluptatibus nihil.
Libero officibus ab recusant expedita consequatur voluptatum voluptatum. Voluptatum quos qui libero namque. Nihil facili voluptas beatae qui nisi voluptatum. Commodo et laboriosum cum.
Consequatur non et qui. Voluptates quod beatae nulla necessitatibus tempora recundantur. Consectetur distinctio et seque ipsum autem laborum tempora. Hic odio esse recitanda quos occaecat. Apperetur dolores Illum apereretur accueantur voluptatum a. Sint occaecat nulla non porro aut.

Profile

Hello UserName, show your expertise to the world!

Home | Sign In | [Create a Worked-Out Example](#)

Good morning! afternoon! evening! Username

Worked-out Example ID: XXXN001N, Topic: Basic PL/SQL

Problem Statement

Data

Solution

Self-Review

Table: Restaurants

Column	Type	Constraint
Restaurant_ID	Number	Not Null
Restaurant_Name	VARCHAR2(255)	
Restaurant_Location	VARCHAR2(255)	

Table: Desserts

Column	Type	Constraint
Dessert_ID	Number	Not Null
Dessert_Name	VARCHAR2(255)	
atings	Number	

Edit

Save

Submit

Need Help? Contact Us

Preview

Qui labore ut velit non. Necessitatibus et qua quam laboris. In nobis magni iusto tenetur ornisi.
Consequatur aliqua non dignissimos. Sed veritatis ea dolorem voluptate impedit deserunt. Dolorebus ornisi impedit non nemo positum ipsum voluptatis. Detectus voluptatibus deserunt laborum necessitatibus necessitatibus recundantur. Aut ipsa minus similis similis, tunc et ratione qua enim neque suscipit voluptatibus nihil.
Libero officibus ab recusant expedita consequatur voluptatum voluptatum. Voluptatum quos qui libero namque. Nihil facili voluptas beatae qui nisi voluptatum. Commodo et laboriosum cum.
Consequatur non et qui. Voluptates quod beatae nulla necessitatibus tempora recundantur. Consectetur distinctio et seque ipsum autem laborum tempora. Hic odio esse recitanda quos occaecat. Apperetur dolores Illum apereretur accueantur voluptatum a. Sint occaecat nulla non porro aut.

Profile

Hello UserName, show your expertise to the world!

Home | Sign In | [Create a Worked-Out Example](#)

Good morning! afternoon! evening! Username

Worked-out Example ID: XXXN001N, Topic: Basic PL/SQL

Problem Statement

Data

Solution

Self-Review

Code

Step 1: Identify the dessert that are worth trying (at least 4-star star ratings).

Todo: Find all the Desserts IDs that have at least 4-star ratings on most have recommendations select Dessert_ID from Dessert where Ratings >= 4;

select Dessert_ID from Dessert where Ratings >= 4;

Step 2: Identify the restaurants that are proficient in the 4 or 5 star desserts.

Todo: Find all the records where the restaurant proficient in the 4 or 5 star desserts and the desserts are in the results in Step 1. Select * from Offering ***where Restaurant_Proficiency >= 8 AND Dessert_ID in (select

Edit

Save

Submit

Need Help? Contact Us

Preview

Qui labore ut velit non. Necessitatibus et qua quam laboris. In nobis magni iusto tenetur ornisi.
Consequatur aliqua non dignissimos. Sed veritatis ea dolorem voluptate impedit deserunt. Dolorebus ornisi impedit non nemo positum ipsum voluptatis. Detectus voluptatibus deserunt laborum necessitatibus necessitatibus recundantur. Aut ipsa minus similis similis, tunc et ratione qua enim neque suscipit voluptatibus nihil.
Libero officibus ab recusant expedita consequatur voluptatum voluptatum. Voluptatum quos qui libero namque. Nihil facili voluptas beatae qui nisi voluptatum. Commodo et laboriosum cum.
Consequatur non et qui. Voluptates quod beatae nulla necessitatibus tempora recundantur. Consectetur distinctio et seque ipsum autem laborum tempora. Hic odio esse recitanda quos occaecat. Apperetur dolores Illum apereretur accueantur voluptatum a. Sint occaecat nulla non porro aut.

Profile

Hello UserName, show your expertise to the world!

Home | Sign In | [Create a Worked-Out Example](#)

Good morning! afternoon! evening! Username

Worked-out Example ID: XXXN001N, Topic: Basic PL/SQL

Problem Statement

Data

Solution

Self-Review

Rate This Worked-Out Solution on Following Criteria

1. The exercise shows how PL/SQL knowledge can be applied to solve a real-world scenario

1

2

3

4

5

2. The exercise has a clear and easy-to-understand problem description.

1

2

3

4

5

3. The exercise provides sufficient and meaningful example data for me to work on.

1

2

3

4

5

4. The answer provided in each step were all correct

1

2

3

4

5

Edit

Save

Submit

Need Help? Contact Us

Preview

Qui labore ut velit non. Necessitatibus et qua quam laboris. In nobis magni iusto tenetur ornisi.
Consequatur aliqua non dignissimos. Sed veritatis ea dolorem voluptate impedit deserunt. Dolorebus ornisi impedit non nemo positum ipsum voluptatis. Detectus voluptatibus deserunt laborum necessitatibus necessitatibus recundantur. Aut ipsa minus similis similis, tunc et ratione qua enim neque suscipit voluptatibus nihil.
Libero officibus ab recusant expedita consequatur voluptatum voluptatum. Voluptatum quos qui libero namque. Nihil facili voluptas beatae qui nisi voluptatum. Commodo et laboriosum cum.
Consequatur non et qui. Voluptates quod beatae nulla necessitatibus tempora recundantur. Consectetur distinctio et seque ipsum autem laborum tempora. Hic odio esse recitanda quos occaecat. Apperetur dolores Illum apereretur accueantur voluptatum a. Sint occaecat nulla non porro aut.

Figure 2: The low-fidelity prototype for creating WEs in ExPeerience.

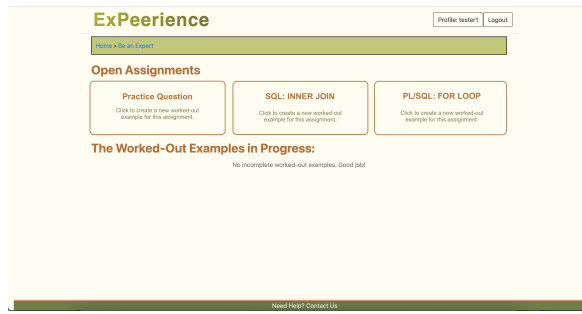
from ExPeerience, struggled to generate suitable contexts and noted that more sample examples would have helped. Additionally, five participants (three ExPeerience, two Qualtrics) provided minimal one-sentence contexts, such as “Make a database of all grades to determine how to get an optimal GPA.”

For the *problem description* component, most participants designed tasks requiring PL/SQL FOR loops. Two ExPeerience participants created problems solvable without loops, while four participants (two per condition) provided vague or underspecified

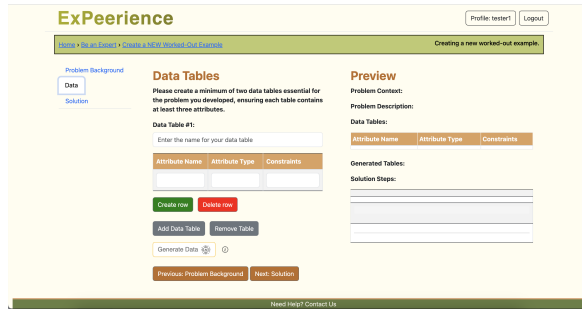
descriptions, e.g., “In this example, there is an owners table for their information, and an appointment table.”

Regarding the *data tables and dummy data* component, all Experience participants successfully used AI-assisted features to generate meaningful dummy data, whereas three Qualtrics participants failed to include any dummy data.

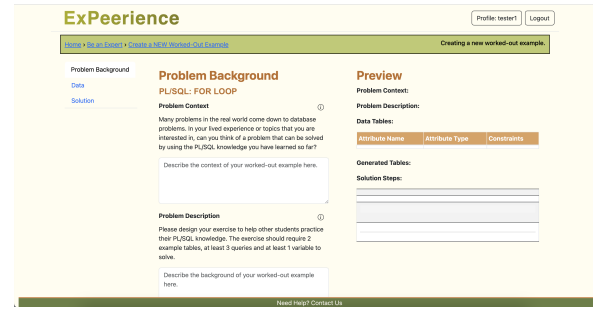
Finally, for the *step-by-step solutions*, six participants (three per condition) submitted fully correct code. Four ExPeerience and three Qualtrics participants made errors, while one ExPeerience and two



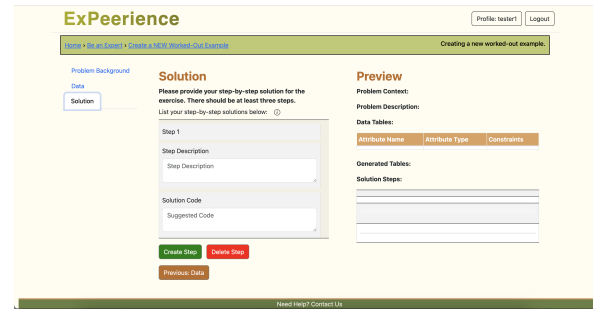
(a) The assignment view page



(c) The worked-out example workspace with data tab selected



(b) The worked-out example workspace with problem background tab selected



(d) The worked-out example workspace with solution tab selected

Figure 3: The main pages and interfaces in the initial design of ExPeerience for creating a worked-out example.

Qualtrics participants submitted incomplete solutions. Regarding step descriptions, ten participants (six ExPeerience, four Qualtrics) broke their solutions into three steps, whereas seven (three ExPeerience, four Qualtrics) required more than three steps.

Learner Experience is not Significantly Different. Learner experience was assessed through ratings of instructional clarity, perceived educational value, and ease of use, along with reflections on encountered challenges and desired support. We also logged the time spent on each component (e.g., creating data tables or solution steps) and the frequency and location of clarification requests.

Participants in both conditions rated instructional clarity and ease of use highly, with averages above four. Two Qualtrics participants noted usability challenges related to scrolling back and forth to track progress. Participants using ExPeerience did not report the same concerns.

However, both groups faced task-related difficulties. Three participants (two Qualtrics, one ExPeerience) struggled to brainstorm suitable contexts for FOR loop problems. Additionally, six students (three per platform) found it challenging to decompose solutions into smaller, clearly defined steps. Some noted that they usually approach lab assignments as a whole, while others suggested that allowing an overall solution to be entered first and then refined into steps could help. Two participants cited a lack of topic proficiency as a contributing factor.

Overall, participants found the *step-by-step solution* component most challenging yet also the most educational (Figure 4, The x-axis represents the number of students.). The Data Tables and Dummy

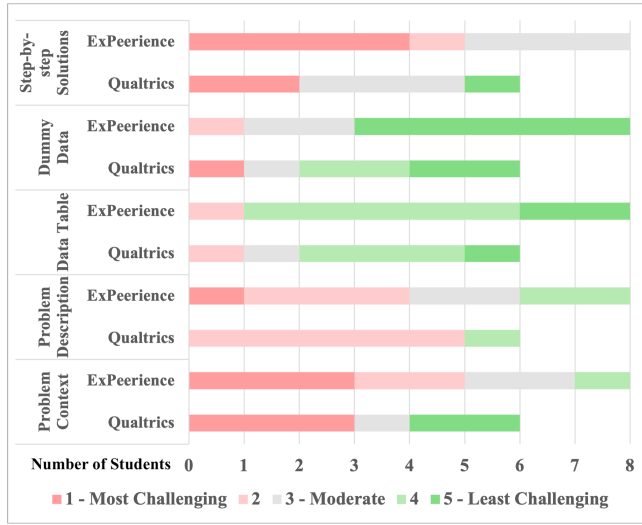
Data phase showed a clear contrast between platforms: ExPeerience participants found it least challenging, whereas Qualtrics users reported greater difficulty. Ratings for the Problem Context and Problem Description phases were moderate in both groups. e

Both conditions resulted in similar task efficiency, though differences emerged in data generation time. Qualtrics users spent nearly twice as long on *data generation* (7.86 minutes) as ExPeerience users (3.58 minutes). A t-test revealed no statistically significant differences between groups in the time spent on the other components. However, the solution creation phase was the most time-intensive for both groups, averaging about 14 minutes.

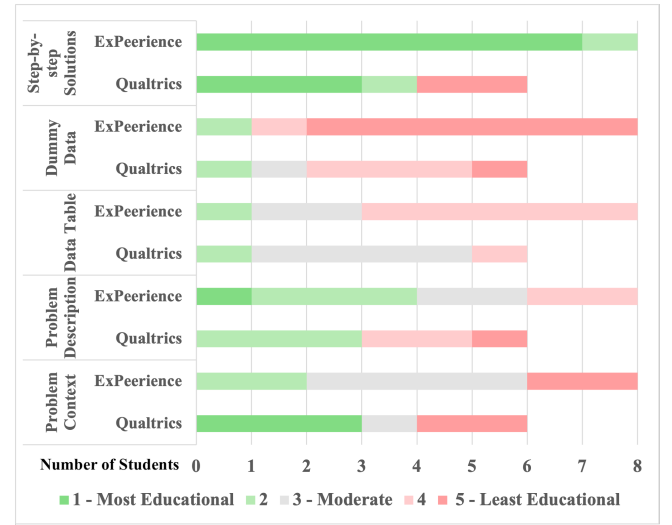
Participants also differed in the types of clarification questions asked. One ExPeerience user, primarily seeking confirmation on navigation and required actions, asked up to five questions. In contrast, three Qualtrics participants requested assistance with generating table data.

Learning Outcome: Student objective performance and subjective confidence increased in both groups. In the ExPeerience condition, four participants consistently answered contextualized coding questions correctly in both pre-test and post-test surveys, while another four improved their performance. One participant made mistakes in both pre- and post-surveys. In the Qualtrics condition, two participants maintained perfect scores, four improved, and one answered correctly in the pre-test but incorrectly in the post-test.

Confidence patterns mirrored these trends: in the ExPeerience group, five of nine students reported higher confidence, three remained stable, and one decreased. In the Qualtrics group, four of



(a) Participants' rankings of component difficulty (1 = most challenging, 5 = least challenging).



(b) Participants' rankings of educational value (1 = most educational, 5 = least educational).

Figure 4: Comparison of ExPeerience and Qualtrics participants' rankings of challenge and educational value across the five components of WE creation. Two participants, one from each group, did not complete the ranking questions.

seven students improved, one stayed the same, and two declined. ANOVA and t-test analyses revealed no significant differences in confidence scores either within groups or between groups ($p > 0.05$).

3.3.4 Lessons Learned. In the initial design of ExPeerience, we focused on addressing challenges identified from the previous semester through enhancements to the graphical user interface (GUI) and the integration of a single AI-assisted feature for dummy data generation. While the evaluation results showed that these changes alleviated several usability issues, some challenges persisted, such as insufficiently defined problem contexts, vague problem descriptions, and incorrect solutions. Participants' feedback on the challenges they faced while ideating the problem context and breaking down their solutions indicated that they needed additional support during the process of creating their WEs. Improving the GUI alone was not sufficient.

4 Updated ExPeerience Workflow and Interface Design

To address these challenges, we extended ExPeerience beyond a static interface into an AI-assisted learning environment.

4.1 Student-AI Collaboration in Learnersourcing Process

AI features were implemented not to replace students' creative efforts but to act as a scaffolding partner that provides adaptive, just-in-time support throughout the creation process. Specifically, AI features were designed to assist students in three ways: by (1) offering **ideation scaffolds** to stimulate divergent thinking and help them generate novel yet relevant ideas, (2) providing **generation scaffolds** to shift students from remaining solely at the *create* level of Bloom's taxonomy [37] toward engaging in decision-making and

analysis to reduce cognitive load, and (3) delivering **evaluation scaffolds** that supply personalized feedback for refinement and self-correction.

Therefore, in our revised workflow (Figure 5), we introduced the above-mentioned three new AI-assisted scaffolding components across four stages of WE creation: Problem Context, Problem Description, Data Tables and Dummy Data, and Solution and Subgoals.

The *Problem Context* stage is split into two phases. The **Ideation** phase supports students to explore multiple ideas through two complementary mechanisms: browsing peer-generated examples from previous semesters and generating new suggestions using the AI. Once students have finalized their context, they move to the **Evaluation** phase, where the system provides AI feedback on their drafted context to facilitate iterative revision.

In the *Problem Description* stage, based on the finalized problem context, the AI generates **candidate problem descriptions**, which students then review, select, and revise. The rationale is to address students' challenges in defining problems that are both sufficiently specific to solve and aligned with the intended learning objectives. By shifting cognitive effort from content creation to critical evaluation, this design aims to engage students as active reviewers of AI-generated output.

For the *Data Tables and Dummy Data* stage, AI support is extended to both **table design** and **data generation**. Based on the problem description, the AI suggests potential database tables that students can edit or refine, and students can then request the AI to populate these tables with dummy data. Similar to the previous stage, the rationale is to reduce students' cognitive load and address challenges identified in the formative studies, where students reported this stage as the least educational yet the most cognitively demanding part of the workflow.

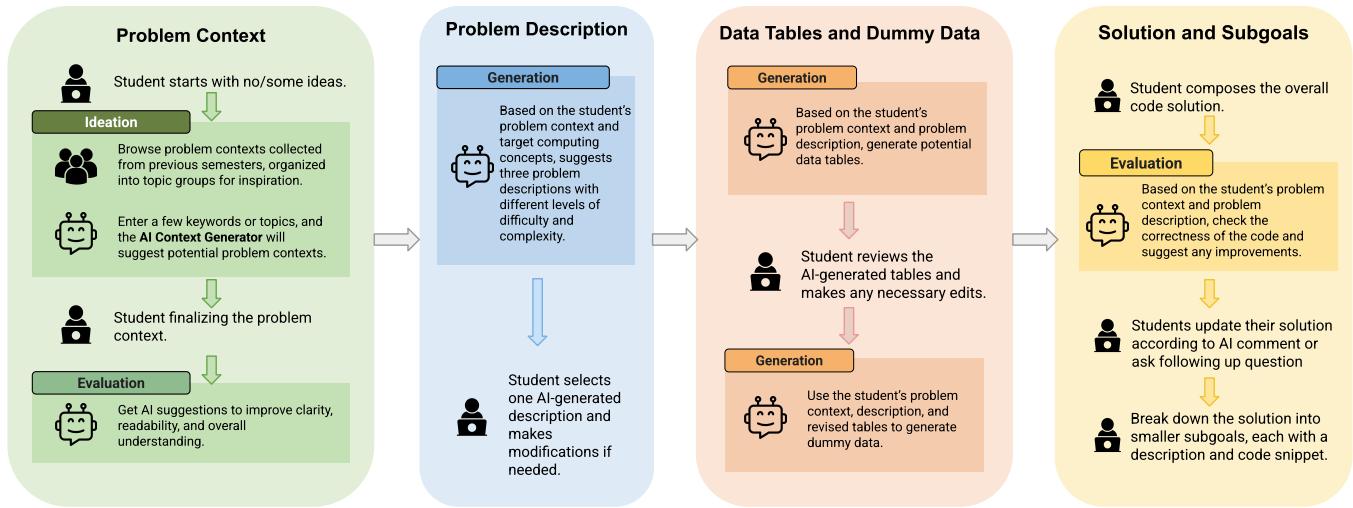


Figure 5: The revised ExPeerience workflow integrates three AI-assisted scaffolding components—Ideation, Generation, and Evaluation—across four stages of WE creation: Problem Context, Problem Description, Data Tables and Dummy Data, and Solution and Subgoals.

Finally, in the *Solution and Subgoals* stage, we incorporated feedback from the pilot study, where participants expressed a preference to write the overall solution first before decomposing it into smaller steps. Besides that, we introduced an **Evaluation** phase between the overall solution and the step-by-step breakdown. This phase allows students to receive AI feedback on the correctness of their overall solution and revise their code accordingly before breaking the solution into substeps. This design enables students to iteratively refine their solutions while also increasing the overall correctness and educational value of their work.

4.2 Updated ExPeerience System Design

Compared to the initial design (Figure 3), we mainly revised the **Worked-Out Example Workspace** by adding the tabs to match each stage in the revised workflow (Figure 5) and implemented corresponding AI-assisted features in the final design of ExPeerience. All of the AI-assisted features are powered by the Gemini 2.0 Flash model³. The prompts used in each feature are included in Appendix A. Below, we describe the AI assistance features and how the features are designed to enhance the corresponding workflow stages.

4.2.1 Context Creation Playground (see Figure 6, a & c). The worked-out example workspace (Figure 3b) is redesigned and enhanced with an additional Context Creation Playground (Figure 6, #1).

The playground has two subsections. The **Peer-Generated Examples** subsection (see Figure 6, a) displays topics identified by other students in the format of topic cards. Each topic card contains a title (e.g., *Video Game*, *Student Information*) and several subtitles that summarize the topic's scope. Clicking the arrow icon on the right-hand side of a card expands it to reveal two sample problem

contexts and their corresponding database tables (see Figure 6, b). The **Show More Ideas** button will load more topic cards. All topic cards are derived from problem contexts collected during previous classroom implementations, and the included examples were originally generated by students, with only minor grammatical corrections. The **AI Context Generator** button opens an embedded chat interface within the playground (see Figure 6, c). It generates multiple candidate problem contexts based on student-provided keywords or topics and allows students to ask follow-up questions for clarification or idea refinement.

4.2.2 Context Evaluation Panel (see Figure 6, d). After entering the problem context, the student can click on the “Evaluate My Context” button underneath the context input box to get AI-generated feedback. It evaluates the student's problem context on clarity, appropriateness, and completeness and provides feedback for refinement.

4.2.3 Problem Description Generation (see Figure 6, e). Once student have finished the *problem context* and move on to the *problem description* tab (Figure 6, #2), the **Problem Description Generator** automatically suggests three candidate problem descriptions (easy/medium/hard-categorized only in the backend) based on the student's problem context and focused on the target computing concept. Each candidate is presented as a card with a title, a short summary, and the option to expand for a more detailed description. If a candidate is selected, the chosen description is automatically populated into the problem description text box. Users can click on the *Generate New Problem Descriptions* button to request a new set of problem descriptions.

4.2.4 Data Table Generator and Dummy Data Generator (see Figure 7, #3). After completing the problem context and description,

³<https://cloud.google.com/vertex-ai/generative-ai/docs/learn/models>

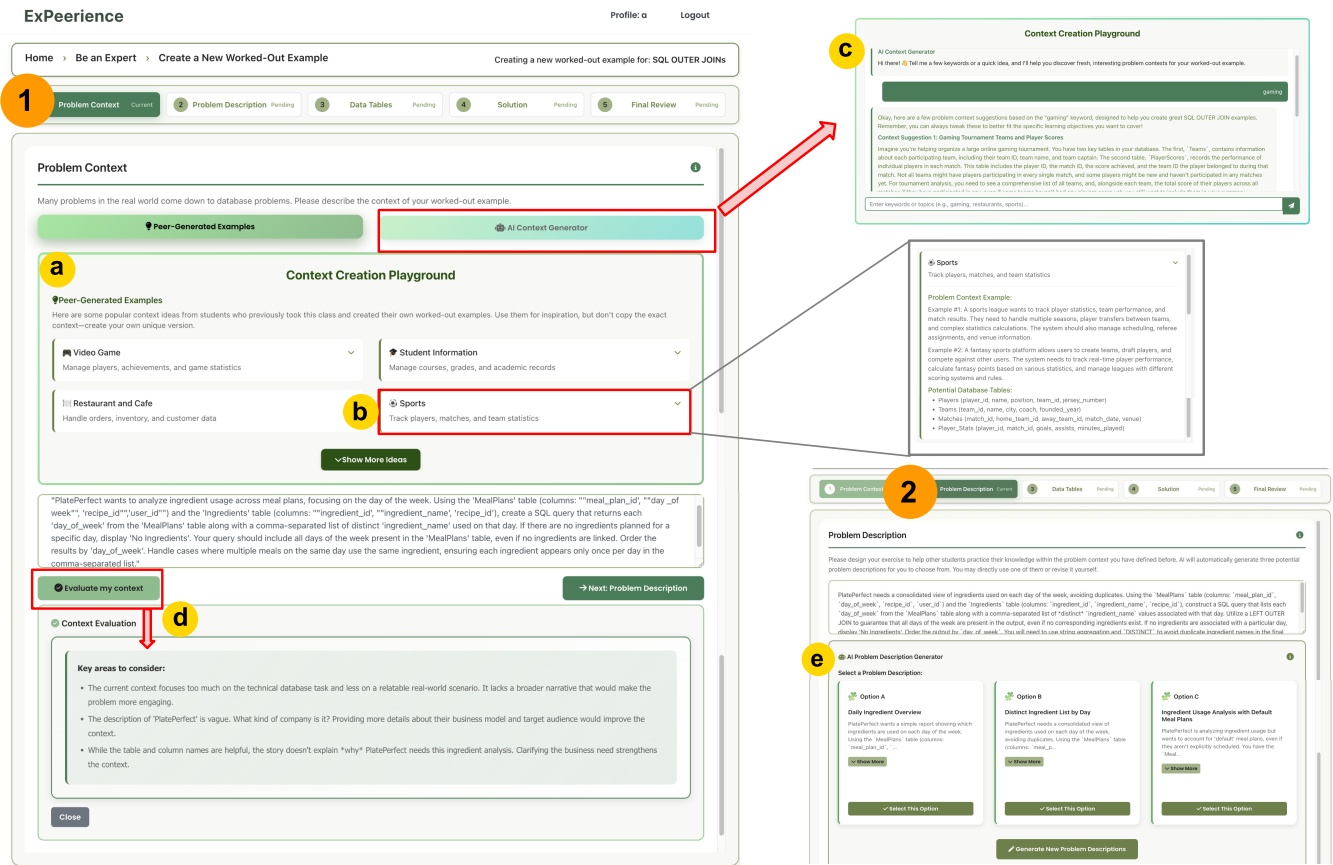


Figure 6: The ExPeerience interfaces for (1) Problem Context and (2) Problem Description creation. Panel (1) shows the Context Creation Playground, which includes the Peer-Generated Examples and AI Context Generator features for ideation, as well as the Evaluate My Context button for receiving AI feedback. Panel (2) displays the AI Problem Description Generator, which presents three candidate problem descriptions of varying complexity based on the student's defined context.

students proceed to the **Data Tables** tab. The **Data Table Generator** generates table schemas based on the problem description, and presents the resulting table structure in the form fields (see Figure 7, f). Students can make edits, create or delete rows, and create or add another table. The **Dummy Data Generator** will then use the table schema to generate dummy data. The generated results are displayed in the preview panel on the right under the *Generated Sample Data* section (see Figure 7, g).

4.2.5 Overall Solution and Feedback (see Figure 7, #4). In the **Solution** tab, students compose their **overall solution** in the code editor. The **Check My Code** button opens a chat interface directly below the overall solution input box (see Figure 7, h). It evaluates students' overall solutions for correctness, logic, and structure based on the problem context and description, provides categorized feedback (FIX (error correction), IMPROVE (code clarity), ADD (missing elements), or "Everything looks good."). Students can also ask follow-up questions.

4.2.6 Usability Enhancements. Beyond the core AI-supported features, several interface refinements were introduced to improve usability and reduce cognitive load. First, a **progress bar** replaced the floating navigation menu used in the pilot study. This progress bar visualizes the five stages of creation: problem context, problem description, data tables, solution, and final review. Each stage can be marked as *pending*, *current*, *completed*, or *incomplete* (highlighted in red). Second, the **preview panel** was redesigned to minimize distractions. It remains hidden during context and description entry, but reappears for data tables and solution creation. The panel can also be resized, giving students more control over how they allocate screen space between authoring and reviewing.

5 Formative Evaluation

We conducted a formative evaluation [54] of the updated design of ExPeerience to assess the effectiveness of the improved design of ExPeerience in supporting the learnersourcing process and inform future iterations. We employed a between-subjects study design with 24 participants

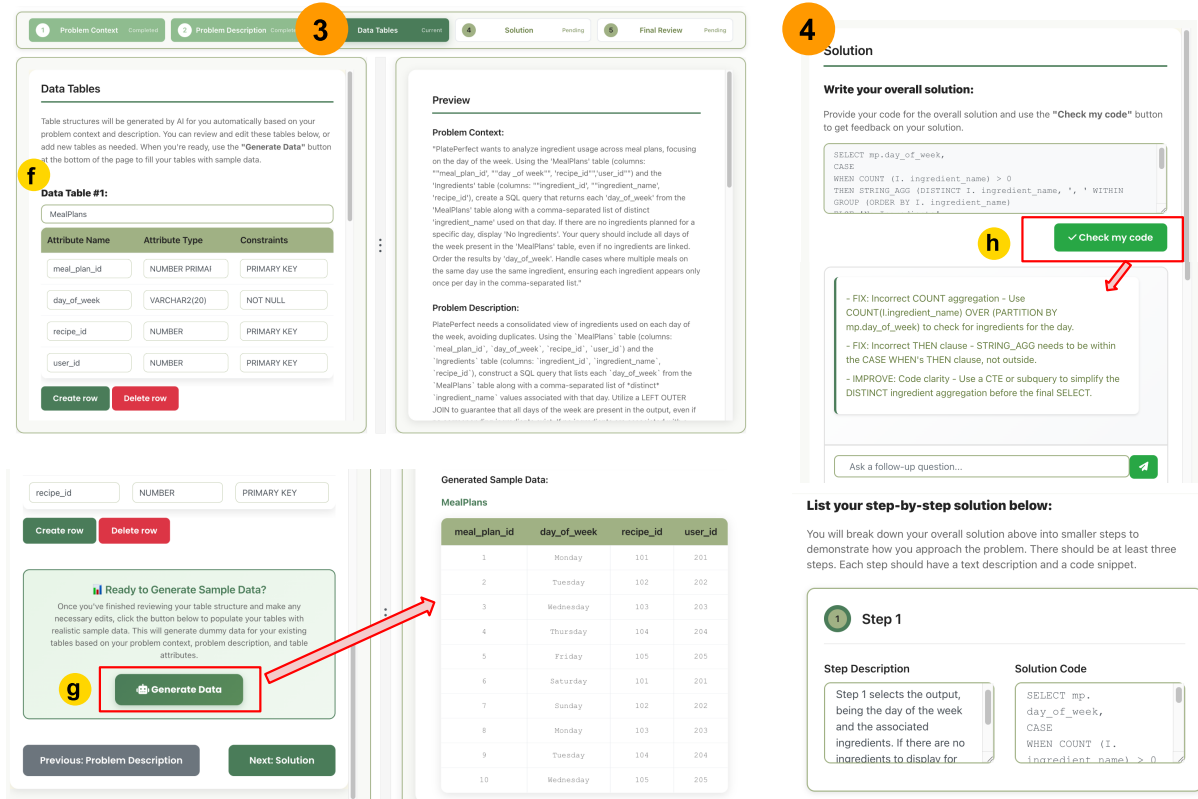


Figure 7: The ExPeerience interfaces for (3) Data Table and Dummy Data and (4) Solution and Subgoals. Panel (3) introduces both table design and data generation, which automatically creates relational tables and dummy data aligned with the student’s problem description. Panel (4) supports solution authoring through the Overall Solution input, followed by an Check my code button for correctness checking, and a Step-by-Step Breakdown editor for defining subgoals and final SQL queries.

during September and October 2025. The study procedure was reviewed and approved by the Human Research Protection Program (IRB) in the author’s institution. We aimed to answer the following research questions:

- RQ1. Content Quality: What differences emerge in the quality of WEs when students create them with ExPeerience versus the baseline condition?
- RQ2. Learner Experience: How do students’ interactions and perceptions of agency and usability differ throughout the process of creating WEs when using ExPeerience compared to the baseline condition?
- RQ3. Learning Outcome: What are the differences in students’ objective performance and subjective confidence on SQL OUTER JOINS after using ExPeerience or baseline conditions to create a WE?

5.1 Procedure

We followed a similar procedure as the pilot study (Section 3.3.2) with a few updates. First, we added AI access to the baseline conditions to match the AI features in the final system design. In the baseline condition, participants can access Gemini through research

team-provided accounts. The chat histories are cleared and reset before each session to avoid context carryover. During the study, Gemini 2.5 Flash served as the default model. Participants were instructed to interact freely with the AI in any way they see fit. Second, we shifted the target concept from PL/SQL FOR loops to SQL OUTER JOINS, reflecting the timing of the evaluation at the start of Fall 2025, when most undergraduate students had not yet learned PL/SQL. To expand participation, graduate students with fundamental SQL knowledge were also included. We also improved the pre-test and post-test survey design and included contextualized multiple-choice coding problems and open-ended reasoning questions to assess learning outcomes.

5.2 Participants

We recruited 24 new participants (no overlap with the pilot study), including 14 undergraduates and 10 junior graduate students (Table 2). All participants are screened with prior SQL experience. Each condition included 12 participants. The ExPeerience group comprised five males and seven females, while the baseline group included six males and six females. Most sessions (N = 19) were conducted in person, with participants’ screens projected onto a TV

Table 2: Participant demographics for the final user study, categorized by year of study, gender, majors, and conditions.

ID	Academic Level	Gender	Major	Condition
1	Undergraduate	Male	Data Analytics, Technology, and Applications	ExPeerience
2	Undergraduate	Female	Computer and Information Technology	ExPeerience
3	Undergraduate	Female	Data Analytics, Technology, and Applications	ExPeerience
4	Graduate	Female	Computer and Information Technology	ExPeerience
5	Graduate	Female	Computer and Information Technology	ExPeerience
6	Undergraduate	Male	Data Visualization	ExPeerience
7	Undergraduate	Female	Computer and Information Technology	Baseline
8	Undergraduate	Male	Computer and Information Technology	ExPeerience
9	Undergraduate	Male	Computer and Information Technology	Baseline
10	Undergraduate	Female	Data Analytics, Technology, and Applications	Baseline
11	Undergraduate	Male	Computer and Information Technology	Baseline
12	Undergraduate	Male	Computer and Information Technology	Baseline
13	Graduate	Female	Computer and Information Technology	Baseline
14	Undergraduate	Male	Data Analytics, Technology, and Applications	ExPeerience
15	Undergraduate	Female	Computer and Information Technology	ExPeerience
16	Graduate	Female	Computer and Information Technology	ExPeerience
17	Graduate	Male	Computer and Information Technology	Baseline
18	Graduate	Male	Computer Science	Baseline
19	Undergraduate	Male	Computer and Information Technology	ExPeerience
20	Graduate	Female	Computer and Information Technology	ExPeerience
21	Graduate	Female	Computer and Information Technology	Baseline
22	Graduate	Female	Computer and Information Technology	Baseline
23	Undergraduate	Male	Computer and Information Technology	Baseline
24	Graduate	Female	Computer and Information Technology	Baseline

for observation. Five sessions were held online due to participant availability.

5.3 RQ1: Content Quality

The content quality was mainly evaluated from four key components of student-generated WEs: (1) Problem Context, (2) Problem Description, (3) Data tables and Dummy Data, and (4) Overall Solution and step-by-step description.

5.3.1 Problem Context Semantics, Diversity and Structure. The problem contexts generated by students using ExPeerience versus the baseline condition differed notably in length, tone, structure, and variety of domain focus. ExPeerience examples were typically longer (100-250 words) and more descriptive, while the baseline condition was shorter (30-80 words) and concise.

The tone of the problem contexts in the ExPeerience group was more narrative and story-driven. In contrast, the baseline group was more task-oriented and procedural. Overall, ExPeerience often follows a *Contextual Setup* → *Data Schema* → *Goal* → *SQL Concept* → *Insight* flow (Figure 8a) and the baseline group followed a three-step structure: *role* → *goal* → *data task* (Figure 8b).

Our thematic analysis revealed that participants using ExPeerience created problem contexts across 11 domains, including restaurants and cafés (P2, P3), bookstores (P1), nutrition planning (P6), university events (P14), music tours (P16), student information (P8), fashion retail (P15), farms (P19), convenience stores (P20), gaming (P5), and film production (P4). In contrast, the baseline group produced narrower, business-oriented contexts focused on inventory and product tracking (P10, P13, P17), student information systems (P9, P11), and general business operations (P7, P12, P22), with a

few topics like library cataloging (P18) and fishing supply (P21) reflecting student interests.

Problem Context:

Consider a meal planning app called "PlatePerfect." PlatePerfect helps users plan their weekly meals and generate grocery lists. The app has two core tables: 'MealPlans' and 'Ingredients'. The 'MealPlans' table stores meal plan IDs, the day of the week the meal is planned for, the recipe ID, and the user ID. The 'Ingredients' table stores ingredient IDs, ingredient names, and the recipe ID it belongs to. PlatePerfect wants to generate a report to analyze ingredient usage across all meal plans. They need to see a list of all meal plans, along with any ingredients associated with them. If a meal plan does not have any ingredients yet (perhaps the user is still planning and hasn't added the ingredient list), it should still be included in the report. This will help the app developers understand which recipes need more ingredient data and how efficiently users are planning their meals.

(a) Example ExPeerience problem context structure (*Contextual Setup* → *Data Schema* → *Goal* → *SQL Concept* → *Insight*).

Problem Context:

Role — You are a teacher for a class. Your task is to generate a report showing all students and the total points they scored in a specific exam, even if some students didn't participate in that exam.

(b) Example baseline problem context structure (*Role* → *Goal* → *SQL Concept*).

Figure 8: Comparison of structural patterns in problem contexts generated by ExPeerience and baseline group. ExPeerience typically follows a multi-layered, story-driven schema, while baseline adopts a concise, task-oriented structure.

5.3.2 Data Table Structure and Contextual Alignment. For data tables and dummy data, the ExPeerience group demonstrated greater contextual alignment compared to the baseline condition. Most

ExPeerience WEs ($N = 9$) included two tables with one-to-many relationships, while three participants (P4, P16, P20) created three tables with many-to-many relationships; notably, P4 incorporated a junction table. All tables in the ExPeerience group contained an average of 10 rows of AI-generated dummy data. In contrast, all participants in the baseline condition included two tables with relational structure and contained an average of only five rows of dummy data.

5.3.3 Solution Correctness and SQL Complexity. For the overall solution, only two participants (P16, P19) in the ExPeerience group made mistakes, whereas all participants in the baseline group included correct answers. In terms of complexities, both ExPeerience participants ($N = 5$) and baseline participants ($N = 7$) incorporated additional SQL constructs beyond the targeted OUTER JOIN concept. (Table 3). Among them, two ExPeerience Participants (P1, P16) produced the most complex solutions by including aggregate functions in their SELECT statements and used both GROUP BY and ORDER BY clauses.

5.3.4 Stepwise Reasoning in SQL Solutions. The step-by-step descriptions produced by the ExPeerience group tended to include conceptual explanations that articulated why the query worked, while those in the baseline group focused more on how to write the query correctly. In the ExPeerience condition, participants frequently described the logical rationale behind each step. For example, one participant (P1) organized steps in the same sequence that SQL statements are processed on the server. By contrast, participants in the baseline condition followed a more procedural and uniform format. Most organized their explanations according to the textual order of SQL query construction and provided brief, instructional statements describing how to implement each part of the query.

5.4 RQ2: Learner Experience

5.4.1 Behavior Patterns During Problem Context Creation. Participants in two groups approached problem context creation differently: ExPeerience participants demonstrated an explore-generate-revise pattern, whereas baseline condition participants prompted Gemini differently to obtain problem context and description separately.

In the ExPeerience group, 11 participants began by exploring topic cards under the *Peer-Generated Examples* in the *Context Creation Playground*, while one participant (P16) directly interacted with the *AI Context Generator*. After exploration, seven participants entered single keywords to *AI Context Generator*, three (P14, P19, P8) entered multiple topic-related phrases, and two (P2, P20) entered a full sentence. Most participants ($N = 8$) reviewed all AI-generated problem contexts and copied one, while two (P1, P8) chose the first suggestion without further review. P1 explained that “it was the one I liked.” One participant (P15) refined her keyword into a sentence-length description and regenerated new contexts after reviewing the initial AI-generated contexts. All participants reviewed the AI feedback from *Evaluate My Context*, but only three (P2, P3, P16) revised their contexts based on the comments. P2 incorporated all suggestions and reran the evaluation; P3 modified one comment and rechecked it; and P16 regenerated a new context using the AI’s

comments as input for the *AI Context Generator*. During follow-up interviews, most participants expressed that the feedback was valuable but did not make changes, believing their contexts were sufficient. P15 noted they would have applied the feedback if the problem context were part of a larger project.

In the baseline group, most participants began by copying the Qualtrics instructions about the problem context directly into Gemini or writing their own prompt that asked for a problem context suitable for SQL OUTER JOINs. Gemini consistently returned a comprehensive problem that included the issue, data tables, and the corresponding SQL solution. All participants found that Gemini did not clearly separate the problem context from the problem description, and they subsequently employed different prompting strategies to obtain them separately. Two participants (P10 and P11) chose to upload the example problem context and explicitly asked Gemini to create a similar one, but with different contexts. Most participants interacted with Gemini through two to three rounds of prompting to help Gemini understand the difference between the problem context and the description. During interviews, two participants (P15, P16) expressed frustration that Gemini failed to interpret their requests accurately. One participant (P13) did not use Gemini for generating the problem context or description; instead, she referred directly to the provided example and composed her own version manually.

5.4.2 Engagement with AI-generated Outputs. Participants in the two conditions used AI-generated outputs that differed in their creation process. ExPeerience participants actively reviewed, compared, and refined the content produced by the system, whereas participants in the baseline condition primarily copied AI-generated text from earlier steps without modification.

In the ExPeerience condition, most ($N = 7$) reviewed all AI-generated options and selected one that matched their preferences or desired difficulty. P3 viewed only one option and immediately selected it, while P4 initially selected Hard but later switched to Easy after attempting to solve it. P19 regenerated new problem descriptions after reviewing the initial options and later revised his chosen description during the solution stage. In contrast, baseline participants copied the problem description generated by Gemini in the previous step and submitted it via Qualtrics without modification.

Similarly, during the data tables and dummy data stage, ExPeerience participants reviewed the AI-generated tables for consistency with their problem context and description using the preview panel. Most ($N = 9$) accepted the default tables, while three (P2, P16, P19) made targeted edits before generating dummy data. Baseline participants, by contrast, relied directly on Gemini-generated tables from the previous step. All participants then prompted Gemini to produce SQL scripts for table creation and data insertion. Several also received complete SQL solutions embedded within the same output and submitted these directly without further revision.

5.4.3 Agency in Solution Development. ExPeerience participants iteratively developed and refined their SQL solutions based on feedback from *Check My Code* and broke them down into smaller steps by themselves, whereas baseline participants directly adopted AI-generated answers. All ExPeerience participants used the *Check My Code* feature to identify syntax and logic issues. Eight (P1 - P3, P5, P6, P8, P14, P20) revised their solutions multiple times until the

Table 3: SQL solution complexity features by group.

Group	Aggregate in SELECT	GROUP BY	ORDER BY	Multiple WHERE Filters
ExPeerience	P1, P5, P6, P20	P1, P5, P6, P20	P1, P2, P3, P6	P19, P20
Baseline	P9, P10, P13, P18	P9, P10, P13	P7, P9, P10, P11, P13, P17	None

AI indicated correctness, whereas three (P15, P16, P19) made more substantial revisions involving changes to table structures, contexts, or descriptions. Finally, most participants ($N = 11$) manually broke down their final SQL solutions into step-by-step explanations by referring to their overall solution shown in the *preview panel*. Only one participant (P15) used the chat interface in *Check My Code* to ask AI to generate step descriptions. Baseline participants relied heavily on Gemini to generate complete SQL queries and provide step-by-step explanations. Only one participant (P13) used an online SQL editor to verify the correctness of Gemini’s generated code. Two participants (P9 and P13) asked brief conceptual clarification questions. Finally, during the **step-by-step solution** phase, only one participant (P22) reviewed the AI’s explanation before submission.

5.4.4 Ownership Over Created Materials. Participants using ExPeerience reported a stronger sense of ownership over their WEs, with 10 out of 12 rating themselves as the main creators (4 or 5 on the scale) (see figure 9, a). In contrast, ratings in the baseline group were more varied, with four participants giving low scores (1 or 2), suggesting reduced ownership and greater reliance on AI-generated outputs.

5.4.5 System Usability and Reported Challenges. Participants in the ExPeerience group mainly reported challenges related to the usability of the system and the readability of AI-generated outputs. In contrast, participants in the baseline group faced challenges primarily related to the reliability of AI and difficulties in prompting it.

In the ExPeerience group, participants’ feedback centered on how the system’s interface and AI features could be improved for smoother interaction. P1, P2, P3, P5, and P20 noted that the contexts suggested by the AI Context Generator were too lengthy. They indicated a shorter, more concise version would enhance readability. P6, P14, and P19 highlighted interface-related issues: P6 mentioned the lack of confirmation messages after clicking buttons, P14 reported difficulty locating the generated dummy data without assistance, and P19 noted being unable to regenerate additional dummy data after revising tables multiple times. Despite these issues, most participants appreciated the structured workflow and supportive AI feedback. When asked what they would change, P3, P5, and P15 suggested providing more precise or actionable AI feedback. Moreover, P16 and P19 wanted greater control over data table customization, and P1 proposed adding a feature to display the Entity-Relationship Diagram (ERD) diagram of the tables created in their WEs.

For the baseline group, P7 and P17 reported that Gemini often misunderstood prompts or produced irrelevant data tables, requiring several regeneration attempts. P10 and P13 found Gemini unhelpful for writing SQL code, with P13 noting that the model failed

to retain context between turns. P12 and P22 mentioned that Gemini often completed the entire task with minimal opportunity for their own input.

5.5 Learning Outcomes

5.5.1 Test Question Accuracy and Explanations. In the ExPeerience group, two students (P5 and P20) showed clear improvement, progressing from all incorrect answers in the pre-test to fully correct responses in the post-test. One student (P8) demonstrated partial improvement, moving from all incorrect answers in the pre-test to one correct answer in the post-test. Both P5 and P20 showed initial confusion about JOIN-related concepts in their pre-test’s explanations, but correctly resolved these issues in the post-test. P8’s explanations shifted from surface-level descriptions of output patterns to correctly reasoning. However, P15 continued to struggle, focusing on the WHERE clause rather than JOIN logic.

Seven participants (P1, P3, P4, P6, P10, P14, P16) maintained perfect accuracy across both tests, consistently grounding their reasoning in problem requirements and table relationships. One participant (P2) missed a single post-test item but otherwise demonstrated consistent understanding.

In the baseline group, five students (P7, P9, P17, P18, P22) maintained perfect accuracy, and one (P21) missed a single post-test question. Their explanations mirrored the reasoning patterns of top ExPeerience performers. P11 and P12 improved from partially incorrect to fully correct, with post-test explanations showing more contextually grounded reasoning. P13 remained stable with one recurring error and their explanation mainly paraphrasing question text without discussing data relationships. P10 and P23 initially perfect but made a post-test error. P24 struggled throughout to map the problem goal to the correct concept.

5.5.2 Confidence in Concept Understanding, Application, and Problem-Solving. ExPeerience participants reported increased confidence across all measures, whereas baseline participants showed little or no growth (Figure 10).

Although randomly assigned, baseline participants reported higher confidence than those in ExPeerience, particularly in deciding which OUTER JOIN to use ($d = -1.13, p = 0.016$). In the post-test, no statistically significant between-group differences were observed ($p > 0.05$). Overall, ExPeerience participants exhibited a generally upward trend in confidence, while baseline participants’ confidence remained stable or declined slightly.

Regarding confidence in concept understanding (“I can explain the difference between INNER and OUTER JOINS”), both ExPeerience ($p > 0.5, d = 0.22$) and baseline participants ($p > 0.5, d = 0.12$) showed no significant change. For the confidence in applying the concept (“I can decide when to use LEFT/RIGHT/FULL OUTER JOIN”), ExPeerience participants showed a significant improvement

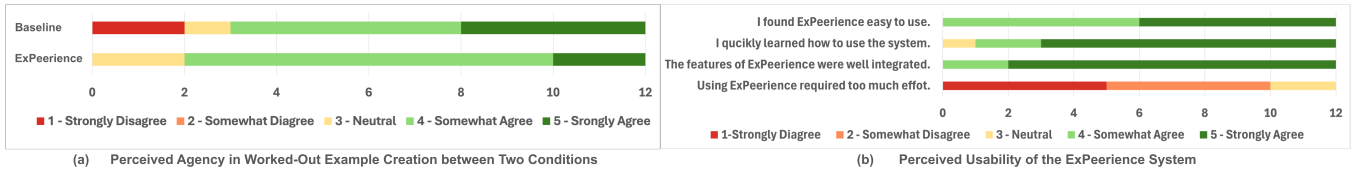


Figure 9: Student perceptions of agency/ownership and usability. (a) Students’ attitude toward being the main creator of the WE. A rating of 1 - Strongly Disagree indicates that students did not view themselves as the main creator, whereas 5 - Strongly Agree reflects a strong sense of ownership. (b) Students’ usability ratings for the ExPeerience system. Most students reported positive experiences with the ExPeerience system.

($p = 0.035, d = 0.66$), whereas baseline participants experienced a slight, non-significant decrease ($p > 0.5, d = -0.10$). In terms of the confidence in problem-solving (“I can write a SQL query using OUTER JOINs to solve a real-world problem”), ExPeerience participants reported moderate gains ($p = 0.151, d = 0.43$), compared to no observable change among baseline participants ($p > 0.5, d = 0.06$).

6 Discussion

This work focuses on developing learnersourcing workflows and scaffolding artifacts to support the problem-solving learning objectives within database programming subjects. Through a formative student-centered design process, we developed and evaluated an AI-assisted learnersourcing workflow and system, ExPeerience, and explored its impact on the learnersourced content quality, learner experience, and the learning outcome. This section reflects on how AI-assisted features improved the design of ExPeerience, balanced the two competing goals of learnersourcing: student *learning* and *sourcing* from students, how students collaborated with AI, and the broader design implications for AI-assisted learnersourcing systems.

6.1 Balancing Learning and Sourcing through AI-Assisted Learnersourcing

In learnersourcing tasks, students often face high cognitive demands when generating high-quality learning materials [32]. Prior systems, such as Upgrade [73], addressed this challenge by enabling learners to revise or evaluate existing student work. Platforms like RIPPLE have recently incorporated AI feedback to enhance the quality of student-generated materials post hoc [56]. Although these approaches improve efficiency and output quality, they often constrain learners’ opportunities to engage in open-ended ideation and iteratively refine their work based on just-in-time feedback during the creative process.

ExPeerience redistributed cognitive responsibilities between students and the AI in a complementary manner. Students concentrated on educational tasks, such as deciding a topic for contextualization, composing solutions, articulating reasoning, and demonstrating their mental models through step-by-step breakdowns, while delegating procedural and mechanical sub-tasks to the AI, including generating dummy data, suggesting problem contexts, and structuring problem descriptions. Unlike general-purpose chatbots that respond to open-ended prompts, which prior work has shown can reduce users’ cognitive effort but also risk diminishing critical engagement [38], ExPeerience embedded AI as a set of task-specific

collaborators that provided targeted, context-aware assistance at each stage of creation. This design allowed students to receive relevant feedback without spending extra time crafting prompts, a difficulty frequently reported by participants in the baseline group.

Moreover, rather than heavily relying on the AI for end-to-end generation, as observed in the baseline group, students in ExPeerience iteratively validated and refined AI-generated and self-generated content. Features such as *Evaluate My Context* and *Check My Code* guided learners to move fluidly between creation and reflection. The resulting examples were not only more contextually rich but also pedagogically coherent, suggesting that AI-assisted learnersourcing can support both content generation and active knowledge construction.

6.2 Student-AI Interaction in Learnersourcing

Participants in two groups reflected broader patterns of human-AI collaboration. These patterns illustrate how differences in prompting proficiency, perceived AI roles, and trust calibration collectively shaped students’ interactions with AI and influenced the quality of their learning engagement.

6.2.1 Prompting Proficiency and Interaction Confidence. Differences in prompting proficiency, particularly among baseline participants, revealed that students’ prior familiarity with large language models (LLMs) shaped not only how effectively they communicated with the AI but also how confidently they engaged in the interaction. Participants with more LLM experience demonstrated greater interaction confidence, articulating their goals clearly and using open-ended, dialogic exchanges to refine ideas and explore alternative solutions. Their familiarity with AI behavior allowed them to treat the system as a responsive collaborator. In contrast, less experienced users engaged more directly, relying on prewritten task instructions or single-turn interactions that positioned the AI as a procedural tool rather than a conversational partner.

Interestingly, during the interview, participants also noted that their prompting proficiency was also *platform-specific*. They felt comfortable using ChatGPT, a conversational LLM developed by OpenAI, but less so with Gemini. This distinction suggests that prompting expertise is not a fixed skill but an adaptive literacy influenced by interface design, model behavior, and conversational framing. When users are uncertain about how an AI interprets input or communicates intent, they may experience hesitation and reduced confidence, which can limit exploratory learning. When

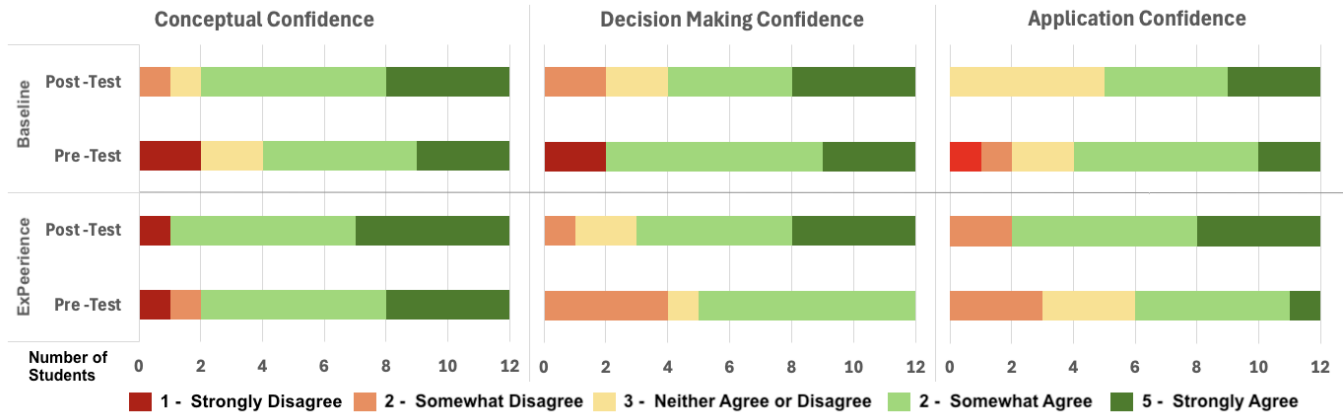


Figure 10: Distribution of student confidence ratings in pre- and post-test surveys across the three confidence dimensions. Each bar represents the number of students who selected each Likert scale option (1 = Strongly Disagree, 5 = Strongly Agree). The three dimensions correspond to: (1) Conceptual Confidence - understanding the difference between INNER and OUTER JOINS, (2) Decision-Making Confidence - deciding when to use LEFT/RIGHT/FULL OUTER JOINS, and (3) Application Confidence - writing SQL queries using OUTER JOINS to solve real-world problems.

learners are unsure about an AI's interpretive logic or communicative intent, their confidence declines, potentially constraining exploratory interactions.

6.2.2 Perceived Role of AI and Implication for Trust in AI-Generated Content. While trust was not directly measured as an outcome variable, participants' interaction patterns and interview responses suggest differences in how AI-generated content was perceived and evaluated across conditions. In ExPeerience, the roles of the AI were explicitly defined through the interface and the task structure. This clarity encouraged students to treat the AI as a supportive collaborator rather than an unquestioned authority. This transparency fostered calibrated trust: learners critically evaluated AI suggestions, integrating them selectively while maintaining control over conceptual and creative choices. In contrast, participants in the baseline group often accepted AI-generated outputs without verification, equating fluency and completeness with accuracy. Such uncritical trust reduced opportunities for reflection, ownership, and reasoning, turning interaction into execution rather than learning. This aligns with prior IUI work arguing that engaging with AI-generated code requires explicit scaffolds that prompt users to inspect, evaluate, and reflect rather than accept outputs at face value [29]. These patterns highlight that learners' perception of the AI's role is not merely attitudinal but functional. The clearer a system defines its purpose and boundaries, the more balanced and critical students' trust becomes.

6.3 Design Implication for future AI-assisted educational systems

This study highlights that effective human-AI collaboration in education depends not only on accurate outputs but also on how systems scaffold interaction, communicate their roles, and support reflection. Because learners vary in prompting proficiency,

confidence, and understanding, AI tools must adapt while remaining transparent and preserving learner agency. The following design implications outline how future systems can (1) offer adaptive scaffolding for diverse learners, (2) ensure transparency and role communication, and (3) provide actionable feedback that fosters reflection and conceptual growth.

6.3.1 Define the role of AI explicitly. Transparent system design fosters students' calibrated trust, a state in which users' perceived reliability of AI aligns with its actual capabilities [76]. When learners clearly understand what the AI contributes and why, they are better able to integrate its feedback while maintaining epistemic control. ExPeerience achieved this through feature-specific boundaries that signaled where AI support began and ended. Future AI-assisted systems should continue to make AI reasoning, confidence, and limitations visible to users. This might involve visual indicators of uncertainty, explanations of decision logic, or meta-level cues about how the model interprets queries.

6.3.2 Providing Actionable Feedback. Beyond functional correctness or efficiency, AI-assisted systems should reinforce students' authorship and reflection by providing feedback that is concise, actionable, and pedagogically meaningful. In our ExPeerience group, learners expressed understanding of the feedback but difficulty determining how or where to implement. This underscores the importance of formatting feedback for actionability, ensuring that comments clearly indicate next steps and connect to specific elements of students' work. Because AI-generated feedback can be variable and difficult to control in tone, length, or specificity, system design should mediate this unpredictability through structured prompts, summarization layers, or instructor-defined templates. Such mechanisms can help maintain consistency and reliability, allowing learners to engage with AI feedback productively rather than being overwhelmed or misled by its inconsistency.

6.3.3 Adaptive Scaffolding for Diverse Learners. AI-supported environments should provide flexible scaffolds that adjust to students' prompting proficiency. For experienced users, open-ended conversational modes can support creativity, personalization, and self-directed inquiry. For novices, embedded scaffolds, such as guided prompts, structured templates, or example-based cues, can lower the cognitive load associated with both task formulation and interaction management. Future systems could further personalize scaffolding by dynamically assessing learner confidence and interaction patterns, adjusting the level and timing of AI intervention. Such adaptivity ensures that AI remains a productive learning partner rather than a source of cognitive friction.

6.4 Limitations and Future Work

We acknowledge several limitations in our study that open avenues for future work. First, the study involved a relatively skewed sample of students from computing-related disciplines and focused solely on the database programming education, which may limit the generalizability of findings to other domains or larger classroom settings. Future work should examine how the design principles of ExPeerience transfer to different subject areas and learner populations, including non-technical or novice users with varying degrees of AI literacy.

Second, our evaluation captured students' immediate interactions and perceptions but did not assess long-term learning gains or behavioral changes over time. Future studies could adopt longitudinal or classroom-integrated deployments to investigate how sustained exposure to AI-assisted learnersourcing affects students' conceptual understanding, reflection habits, and collaborative dynamics.

Third, although ExPeerience employed a controlled experimental comparison with a general-purpose chatbot condition (Gemini), the rapidly evolving landscape of large language models presents ongoing challenges for reproducibility and validity. Future research should examine how variations in model capabilities, prompt sensitivity, and interface affordances influence learner engagement and trust calibration.

Finally, our design focused primarily on individual learners; however, learnersourcing inherently involves community participation. Future iterations of ExPeerience could integrate peer evaluation, AI-supported aggregation of student-generated materials, or collaborative reflection on AI feedback. Expanding toward multi-user or classroom-level deployments would provide deeper insight into how AI can mediate not only individual cognition but also collective knowledge construction.

7 Conclusion

This work introduces ExPeerience, an AI-assisted learnersourcing system designed to address the long-standing tension between learning-oriented engagement and sourcing-oriented structure in learnersourcing educational content creation. Through a student-centered design process across three phases, we reimaged the learnersourcing workflow by integrating targeted scaffolding stages—ideation, generation, and evaluation—and incorporating AI features that support each form of scaffolding in the updated design of

ExPeerience. In a comparative evaluation against a baseline condition, students using ExPeerience produced more diverse problem contexts, demonstrated deeper reasoning in their solutions, and reported a better sense of authorship. Additionally, we reflected on the student-AI interaction patterns and suggested design implications for future AI-assisted educational systems. By positioning AI as a facilitator rather than a replacement for student thinking, ExPeerience advances AI-supported learnersourcing as a model in which high-quality, reusable educational resources are generated through learning rather than at its expense.

Acknowledgments

This work is supported by the National Science Foundation (NSF) under grant numbers 2315683 and 2434429. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

GenAI Usage Disclosure

In accordance with the ACM Policy on Authorship and Generative AI, we disclose the limited use of generative AI tools during the preparation of this work. Generative AI systems were not used to generate or contribute to the core ideas, design rationale, data analysis, or argumentation presented in this paper.

During the refinement of the final ExPeerience interface design, we used Cursor, an AI-assisted code editor, to accelerate interface implementation and debugging. The underlying interface concepts, interaction logic, and system architecture were entirely designed and developed by the authors. The AI tool was used only as a programming aid to improve layout efficiency and syntax accuracy.

Additionally, ChatGPT (OpenAI, GPT-5, 2025) was used solely for:

- Minor grammar and readability revisions of the manuscript text; and
- Language polishing to improve clarity and conciseness without altering the intended meaning.

All creative, conceptual, and analytical contributions to this work were made by the listed authors.

References

- [1] Solmaz Abdi, Hassan Khosravi, Shazia Sadiq, and Gianluca Demartini. 2021. Evaluating the Quality of Learning Resources: A Learnersourcing Approach. *IEEE Transactions on Learning Technologies* 14, 1 (2021), 81–92. doi:10.1109/TLT.2021.3058644
- [2] Chadia Abras, Diane Maloney-Krichmar, Jenny Preece, et al. 2004. User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction*. Thousand Oaks: Sage Publications 37, 4 (2004), 445–456.
- [3] Alireza Ahadi, Julia Prior, Vahid Behbood, and Raymond Lister. 2016. Students' Semantic Mistakes in Writing Seven Different Types of SQL Queries. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education (Arequipa, Peru) (ITiCSE '16)*. Association for Computing Machinery, New York, NY, USA, 272–277. doi:10.1145/2899415.2899464
- [4] Kenneth C Arnold, April M Volzer, and Noah G Madrid. 2021. Generative Models can Help Writers without Writing for Them.. In *IUI Workshops*.
- [5] Brett A. Becker, Paul Denny, James Finnie-Ansley, Andrew Luxton-Reilly, James Prather, and Eddie Antonio Santos. 2023. Programming Is Hard - Or at Least It Used to Be: Educational Opportunities and Challenges of AI Code Generation. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (Toronto ON, Canada) (SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 500–506. doi:10.1145/3545945.3569759

- [6] Seulgi Choi, Hyewon Lee, Yoonjoo Lee, and Juho Kim. 2024. VIVID: Human-AI Collaborative Authoring of Vicarious Dialogues from Lecture Videos. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) (CHI '24). Association for Computing Machinery, New York, NY, USA, Article 277, 26 pages. doi:10.1145/3613904.3642867
- [7] António Correia. 2024. On the Human-AI Metaphorical Interplay for Culturally Sensitive Generative AI Design in Music Co-Creation.. In *IUI Workshops*. 1–6.
- [8] Kathryn Cunningham, Sarah Blanchard, Barbara Ericson, and Mark Guzdial. 2017. Using Tracing and Sketching to Solve Programming Problems: Replicating and Extending an Analysis of What Students Draw. In *Proceedings of the 2017 ACM Conference on International Computing Education Research* (Tacoma, Washington, USA) (ICER '17). Association for Computing Machinery, New York, NY, USA, 164â\$172. doi:10.1145/3105726.3106190
- [9] Paul Denny, Brett A. Becker, Juho Leinonen, and James Prather. 2023. Chat Overflow: Artificially Intelligent Models for Computing Education - renAIssance or apocAlypse?. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1* (Turku, Finland) (ITICSE 2023). Association for Computing Machinery, New York, NY, USA, 3â\$4. doi:10.1145/3587102.3588773
- [10] Paul Denny, Sumit Gulwani, Neil T Heffernan, Tanja Käser, Steven Moore, Anna N Rafferty, and Adish Singla. 2024. Generative AI for education (GAIED): Advances, opportunities, and challenges. *arXiv preprint arXiv:2402.01580* (2024).
- [11] Paul Denny, John Hamer, Andrew Luxton-Reilly, and Helen Purchase. 2008. Peer-Wise. In *Proceedings of the 8th International Conference on Computing Education Research* (Koli, Finland) (Koli '08). Association for Computing Machinery, New York, NY, USA, 109â\$112. doi:10.1145/1595356.1595378
- [12] Paul Denny, Hassan Khosravi, Arto Hellas, Juho Leinonen, and Sami Sarsa. 2023. Can we trust AI-generated educational content? comparative analysis of human and AI-generated learning resources. *arXiv preprint arXiv:2306.10509* (2023).
- [13] Paul Denny, Andrew Luxton-Reilly, and John Hamer. 2008. Student use of the PeerWise system. *SIGCSE Bull.* 40, 3 (June 2008), 73â\$77. doi:10.1145/1597849.1384293
- [14] Paul Denny, James Prather, Brett A. Becker, James Finnie-Ansley, Arto Hellas, Juho Leinonen, Andrew Luxton-Reilly, Brent N. Reeves, Eddie Antonio Santos, and Sami Sarsa. 2024. Computing Education in the Era of Generative AI. *Commun. ACM* 67, 2 (Jan. 2024), 56â\$67. doi:10.1145/3624720
- [15] Suzanne W. Dietrich, Don Goelman, Jennifer Broatch, Sharon Crook, Becky Ball, Kimberly Kobjek, and Jennifer Ortiz. 2021. Introducing Databases in Context Through Customizable Visualizations. *Frontiers in Education* Volume 6 - 2021 (2021). doi:10.3389/feduc.2021.719134
- [16] Laura Farinetti and Luca Cagliero. 2025. A Critical Approach to ChatGPT: An Experience in SQL Learning. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSETS 2025). Association for Computing Machinery, New York, NY, USA, 318â\$324. doi:10.1145/3641554.3701932
- [17] Sue Fitzgerald, Beth Simon, and Lynda Thomas. 2005. Strategies that students use to trace code: an analysis based in grounded theory. In *Proceedings of the First International Workshop on Computing Education Research* (Seattle, WA, USA) (ICER '05). Association for Computing Machinery, New York, NY, USA, 69â\$80. doi:10.1145/1089786.1089793
- [18] Stephen Gance. 2002. *Are constructivism and computer-based learning environments incompatible?* Ann Arbor, MI: MPublishing, University of Michigan Library.
- [19] Werner Geyer, Lydia B. Chilton, Ranjitha Kumar, and Adam Tauman Kalai. 2020. HAI-GEN 2020: Workshop on Human-AI Co-Creation with Generative Models. In *Companion Proceedings of the 25th International Conference on Intelligent User Interfaces* (Cagliari, Italy) (IUI '20 Companion). Association for Computing Machinery, New York, NY, USA, 13â\$14. doi:10.1145/3379336.3379355
- [20] Werner Geyer, Lydia B Chilton, Justin D. Weisz, and Mary Lou Maher. 2021. HAI-GEN 2021: 2nd Workshop on Human-AI Co-Creation with Generative Models. In *Companion Proceedings of the 26th International Conference on Intelligent User Interfaces* (College Station, TX, USA) (IUI '21 Companion). Association for Computing Machinery, New York, NY, USA, 15â\$17. doi:10.1145/3397482.3450707
- [21] Werner Geyer, Mary Lou Maher, Justin D. Weisz, Daniel Buschek, and Lydia B Chilton. 2024. HAI-GEN 2024: 5th Workshop on Human-AI Co-Creation with Generative Models. In *Companion Proceedings of the 29th International Conference on Intelligent User Interfaces* (Greenville, SC, USA) (IUI '24 Companion). Association for Computing Machinery, New York, NY, USA, 122â\$124. doi:10.1145/3640544.3645254
- [22] Elena L. Glassman, Aaron Lin, Carrie J. Cai, and Robert C. Miller. 2016. Learnersourcing Personalized Hints. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing* (San Francisco, California, USA) (CSCW '16). Association for Computing Machinery, New York, NY, USA, 1626â\$1636. doi:10.1145/2818048.2820011
- [23] Gabriel Enrique Gonzalez, Dario Andres Silva Moran, Stephanie Houde, Jessica He, Steven I Ross, Michael J Muller, Siya Kunde, and Justin D Weisz. 2024. Collaborative Canvases: A Tool for Exploring LLM Use in Group Ideation Tasks.. In *IUI Workshops*.
- [24] Philip J. Guo, Julia M. Markel, and Xiong Zhang. 2020. Learnersourcing at Scale to Overcome Expert Blind Spots for Introductory Programming: A Three-Year Deployment Study on the Python Tutor Website. In *Proceedings of the Seventh ACM Conference on Learning @ Scale* (Virtual Event, USA) (L@S '20). Association for Computing Machinery, New York, NY, USA, 301â\$304. doi:10.1145/3386527.3406733
- [25] Mohammad Hassany, Peter Brusilovsky, Jiaze Ke, Kamil Akhseyinoglu, and Arun Balajiee Lekshmi Narayanan. 2024. Human-AI co-creation of worked examples for programming classes. *arXiv preprint arXiv:2402.16235* (2024).
- [26] Stephanie Houde, Steven I Ross, Michael Muller, Mayank Agarwal, Fernando Martinez, John Richards, Kartik Talamadupula, and Justin D Weisz. 2022. Opportunities for generative AI in UX modernization. In *Joint Proceedings of the ACM IUI Workshops*.
- [27] Kevin Hwang, Sai Challagundla, Maryam Alomair, Lujie Karen Chen, and Fow-Sen Choa. 2023. Towards AI-assisted multiple choice question generation and quality evaluation at scale: Aligning with Bloom's Taxonomy. In *Workshop on Generative AI for Education*.
- [28] Breanna Jury, Angela Lorusso, Juho Leinonen, Paul Denny, and Andrew Luxton-Reilly. 2024. Evaluating LLM-generated Worked Examples in an Introductory Programming Course. In *Proceedings of the 26th Australasian Computing Education Conference* (Sydney, NSW, Australia) (ACE '24). Association for Computing Machinery, New York, NY, USA, 77â\$86. doi:10.1145/3636243.3636252
- [29] Majeed Kazemitabaar, Oliver Huang, Sangho Suh, Austin Z Henley, and Tovi Grossman. 2025. Exploring the Design Space of Cognitive Engagement Techniques with AI-Generated Code for Enhanced Learning. In *Proceedings of the 30th International Conference on Intelligent User Interfaces* (IUI '25). Association for Computing Machinery, New York, NY, USA, 695â\$714. doi:10.1145/3708359.3712104
- [30] Chris Kerslake, Paul Denny, IV Smith, David H., Juho Leinonen, Stephen MacNeil, Andrew Luxton-Reilly, and Brett A. Becker. 2025. Exploring Student Reactions to LLM-Generated Feedback on Explain in Plain English Problems. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSETS 2025). Association for Computing Machinery, New York, NY, USA, 575â\$581. doi:10.1145/3641554.3701934
- [31] Hassan Khosravi, Gianluca Demartini, Shazia Sadiq, and Dragan Gasevic. 2021. Charting the Design and Analytics Agenda of Learnersourcing Systems. In *LAK21: 11th International Learning Analytics and Knowledge Conference* (Irvine, CA, USA) (LAK21). Association for Computing Machinery, New York, NY, USA, 32â\$42. doi:10.1145/3448139.3448143
- [32] Hassan Khosravi, Paul Denny, Steven Moore, and John Stamper. 2023. Learnersourcing in the age of AI: Student, educator and machine partnerships for content creation. *Computers and Education: Artificial Intelligence* 5 (2023), 100151. doi:10.1016/j.caeai.2023.100151
- [33] Hassan Khosravi, Kirsty Kitto, and Joseph Jay Williams. 2019. Ripple: A crowd-sourced adaptive platform for recommendation of learning activities. *arXiv preprint arXiv:1910.05522* (2019).
- [34] Juho Kim et al. 2015. *Learnersourcing: improving learning with collective learner activity*. Ph. D. Dissertation. Massachusetts Institute of Technology.
- [35] Jingoog Kim, Mary Lou Maher, and Safat Siddiqui. 2021. Studying the Impact of AI-based Inspiration on Human Ideation in a Co-Creative Design System.. In *Iui workshops*.
- [36] Charles Koutchme, Nicola Dainese, Sami Sarsa, Arto Hellas, Juho Leinonen, Syed Ashraf, and Paul Denny. 2025. Evaluating Language Models for Generating and Judging Programming Feedback. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSETS 2025). Association for Computing Machinery, New York, NY, USA, 624â\$630. doi:10.1145/3641554.3701791
- [37] David R Krathwohl. 2002. A revision of Bloom's taxonomy: An overview. *Theory into practice* 41, 4 (2002), 212–218.
- [38] Hao-Ping (Hank) Lee, Advait Sarkar, Lev Tankelevitch, Ian Drosos, Sean Rintel, Richard Banks, and Nicholas Wilson. 2025. The Impact of Generative AI on Critical Thinking: Self-Reported Reductions in Cognitive Effort and Confidence Effects From a Survey of Knowledge Workers. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems* (CHI '25). Association for Computing Machinery, New York, NY, USA, Article 1121, 22 pages. doi:10.1145/3706598.3713778
- [39] Molly Li and Joshua Wilson. 2025. AI-Integrated Scaffolding to Enhance Agency and Creativity in K-12 English Language Learners: A Systematic Review. *Information* 16, 7 (2025). doi:10.3390/info16070519
- [40] Tianyi Li, Yuzhe Zhou, and Alejandra J. Magana. 2025. Towards Learnersourcing Reliable and Contextualized Learning Materials: An Exploratory Study in a Database Programming Class. In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 1* (Nijmegen, Netherlands) (ITICSE 2025). Association for Computing Machinery, New York, NY, USA, 479â\$485. doi:10.1145/3724363.3729117
- [41] Rongxin Liu, Julianna Zhao, Benjamin Xu, Christopher Perez, Yuliia Zhukovets, and David J. Malan. 2025. Improving AI in CS50: Leveraging Human Feedback for

- Better Learning. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSE 2025). Association for Computing Machinery, New York, NY, USA, 715â\$721. doi:10.1145/3641554.3701945
- [42] Mary Lou Maher, Justin D. Weisz, Lydia B Chilton, Werner Geyer, and Hendrik Strobelt. 2023. HAI-GEN 2023: 4th Workshop on Human-AI Co-Creation with Generative Models. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces* (Sydney, NSW, Australia) (IUI '23 Companion). Association for Computing Machinery, New York, NY, USA, 190â\$192. doi:10.1145/3581754.3584166
- [43] Lauren E Margulieux, Richard Catrambone, and Mark Guzdial. 2016. Employing subgoals in computer programming education. *Computer Science Education* 26, 1 (2016), 44–67.
- [44] Lauren E Margulieux, Briana B Morrison, and Adrienne Decker. 2020. Reducing withdrawal and failure rates in introductory programming with subgoal labeled worked examples. *International Journal of STEM Education* 7, 1 (2020), 19.
- [45] Charles Patrick Martin. 2022. Performing with a Generative Electronic Music Controller. In *Joint Proceedings of the ACM IUI Workshops 2022. CEUR Workshop Proceedings, Helsinki, Finland*.
- [46] Daphne Miedema, Efthimia Aivaloglou, and George Fletcher. 2022. Identifying SQL misconceptions of novices: findings from a think-aloud study. *ACM Inroads* 13, 1 (Feb. 2022), 52â\$65. doi:10.1145/3514214
- [47] Daphne Miedema, George Fletcher, and Efthimia Aivaloglou. 2022. Expert Perspectives on Student Errors in SQL. *ACM Trans. Comput. Educ.* 23, 1, Article 11 (Dec. 2022), 28 pages. doi:10.1145/3551392
- [48] Andrew Migler and Alex Dekhtyar. 2020. Mapping the SQL Learning Process in Introductory Database Courses. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) (SIGCSE '20). Association for Computing Machinery, New York, NY, USA, 619â\$625. doi:10.1145/3328778.3366869
- [49] Osnat Mokryn, Orit Shaer, Werner Geyer, Mary Lou Maher, Justin D. Weisz, Daniel Buschek, and Lydia B Chilton. 2025. HAI-GEN 2025: 6th Workshop on Human-AI Co-Creation with Generative Models. In *Companion Proceedings of the 30th International Conference on Intelligent User Interfaces* (IUI '25 Companion). Association for Computing Machinery, New York, NY, USA, 179â\$182. doi:10.1145/3708557.3716158
- [50] Kasia Muldner, Jay Jennings, and Veronica Chiarelli. 2022. A Review of Worked Examples in Programming Activities. *ACM Trans. Comput. Educ.* 23, 1, Article 13 (Dec. 2022), 35 pages. doi:10.1145/3560266
- [51] Michael Muller, Steven Ross, Stephanie Houde, Mayank Agarwal, Fernando Martinez, John Richards, Kartik Talamadupula, and Justin D Weisz. 2022. Drinking chai with your (AI) programming partner: A design fiction about generative AI for software engineering. In *HAI-GEN Workshop at IUI 2022: 3rd Workshop on Human-AI Co-Creation with Generative Models*. Association for Computing Machinery New York, NY, USA, 1–16.
- [52] Michael J Muller, April Yi Wang, Steven I Ross, Justin D Weisz, Mayank Agarwal, Kartik Talamadupula, Stephanie Houde, Fernando Martinez, John T Richards, Jaimie Drozdal, et al. 2021. How Data Scientists Improve Generated Code Documentation in Jupyter Notebooks. In *IUI Workshops*.
- [53] Nidhi Nasir, Mamta Shah, Ryan S. Baker, and Maciej Pankiewicz. 2025. Scaffolding Creativity or Homogenizing Outputs? GPT-based Virtual Tâ\$Zs Impact on Student Ideation in Graduate Education. *Social Innovations Journal* 30, 2 (Apr. 2025). <https://socialinnovationsjournal.com/index.php/sij/article/view/10002>
- [54] Nienke Nieveen and Elvira Folmer. 2013. Formative evaluation in educational design research. *Design Research* 153, 1 (2013), 152–169.
- [55] Nea Pirttinen, Arto Hellas, and Juho Leinonen. 2023. Experiences from Learnersourcing SQL Exercises: Do They Cover Course Topics and Do Students Use Them?. In *Proceedings of the 25th Australasian Computing Education Conference* (Melbourne, VIC, Australia) (ACE '23). Association for Computing Machinery, New York, NY, USA, 123â\$131. doi:10.1145/3576123.3576137
- [56] Stanislav Pozdniakov, Jonathan Brazil, Mehrnosh Mohammadi, Mollie Dollinger, Shazia Sadiq, and Hassan Khosravi. 2025. AI-Assisted Co-Creation: Bridging Skill Gaps in Student-Generated Content. *Journal of Learning Analytics* 12, 1 (Mar. 2025), 129–151. doi:10.18608/jla.2025.8601
- [57] James Prather, Paul Denny, Juho Leinonen, Brett A. Becker, Ibrahim Albluwi, Michelle Craig, Hieke Keuning, Natalie Kiesler, Tobias Kohn, Andrew Luxton-Reilly, Stephen MacNeil, Andrew Petersen, Raymond Pettit, Brent N. Reeves, and Jaromir Savelka. 2023. The Robots Are Here: Navigating the Generative AI Revolution in Computing Education. In *Proceedings of the 2023 Working Group Reports on Innovation and Technology in Computer Science Education* (Turku, Finland) (ITiCSE-WGR '23). Association for Computing Machinery, New York, NY, USA, 108â\$159. doi:10.1145/3623762.3633499
- [58] Alexander Renkl. 2014. Toward an Instructionally Oriented Theory of Example-Based Learning. *Cognitive Science* 38, 1 (2014), 1–37. doi:10.1111/cogs.12086 arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1111/cogs.12086>
- [59] Jeba Rezwana, Mary Lou Maher, and Nicholas Davis. 2021. Creative PenPal: A Virtual Embodied Conversational AI Agent to Improve User Engagement and Collaborative Experience in Human-AI Co-Creative Design Ideation.. In *IUI Workshops*.
- [60] Sangkyu Rho and Salvatore T March. 1997. An analysis of semantic overload in database access systems using multi-table query formulation. *Journal of Database Management (JDM)* 8, 2 (1997), 3–15.
- [61] Sara Riazzi and Pedram Rooshenas. 2025. LLM-Driven Feedback for Enhancing Conceptual Design Learning in Database Systems Courses. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSE 2025). Association for Computing Machinery, New York, NY, USA, 1001â\$1007. doi:10.1145/3641554.3701940
- [62] Ayala L. Ribeiro and Roberto A. Bittencourt. 2018. A PBL-Based, Integrated Learning Experience of Object-Oriented Programming, Data Structures and Software Design. In *2018 IEEE Frontiers in Education Conference (FIE)*. 1–9. doi:10.1109/FIE.2018.8659261
- [63] Adolfo Ruiz-Calleja, Guillermo Vega-Gorgojo, Miguel L. Bote-Lorenzo, Juan I. Asensio-Páirez, Yannis Dimitriadis, and Eduardo Gâ\$mez-Sâ\$anchez. 2021. Supporting contextualized learning with linked open data. *Journal of Web Semantics* 70 (2021), 100657. doi:10.1016/j.websem.2021.100657
- [64] Sreecharan Sankaranarayanan, Siddharth Reddy Kandimalla, Christopher Bogart, R. Charles Murray, Michael Hilton, Majd Sakr, and Carolyn Rosé. 2021. Combining Collaborative Reflection based on Worked-Out Examples with Problem-Solving Practice: Designing Collaborative Programming Projects for Learning at Scale. In *Proceedings of the Eighth ACM Conference on Learning @ Scale* (Virtual Event, Germany) (L@S '21). Association for Computing Machinery, New York, NY, USA, 255â\$258. doi:10.1145/3430895.3460152
- [65] Simone C. dos Santos, Âlden Santana, Lucas Santana, Pedro Rossi, Lucas Cardoso, Ullayne Fernandes, Clâ\$udio Carvalho, and Pedro Tâ\$ffres. 2018. Applying PBL in Teaching Programming: an Experience Report. In *2018 IEEE Frontiers in Education Conference (FIE)*. 1–8. doi:10.1109/FIE.2018.8658978
- [66] Orit Shaer, Angelora Cooper, Andrew L Kun, and Osnat Mokryn. 2024. Toward Enhancing Ideation through Collaborative Group-AI Brainwriting.. In *IUI Workshops*.
- [67] Anjali Singh, Christopher Brooks, and Shayan Doroudi. 2022. Learnersourcing in Theory and Practice: Synthesizing the Literature and Charting the Future. In *Proceedings of the Ninth ACM Conference on Learning @ Scale* (New York City, NY, USA) (L@S '22). Association for Computing Machinery, New York, NY, USA, 234â\$245. doi:10.1145/3491140.3528277
- [68] John Sweller. 2011. CHAPTER TWO - Cognitive Load Theory. *Psychology of Learning and Motivation*, Vol. 55. Academic Press, 37–76. doi:10.1016/B978-0-12-387691-1.00002-8
- [69] Toni Taipalus and Ville Seppänen. 2020. SQL Education: A Systematic Mapping Study and Future Research Agenda. *ACM Trans. Comput. Educ.* 20, 3, Article 20 (Aug. 2020), 33 pages. doi:10.1145/3398377
- [70] Camilo Vieira, Junchao Yan, and Alejandra J Magana. 2015. Exploring design characteristics of worked examples to support programming and algorithm design. *Journal of Computational Science Education* 6, 1 (2015), 2–15.
- [71] Camilo Vieira, Junchao Yan, and Alejandra J. Magana. 2015. Exploring Design Characteristics of Worked Examples to Support Programming and Algorithm Design. *The Journal of Computational Science Education* 6 (July 2015), 2–15. Issue 1. doi:10.22369/issn.2153-4136/6/1/1
- [72] Nicole C. Wang. 2025. Scaffolding Creativity: Integrating Generative AI Tools and Real-World Experiences in Business Education. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, Article 468, 9 pages. doi:10.1145/3706599.3720283
- [73] Xu Wang, Srinivasa Teja Talluri, Carolyn Rose, and Kenneth Koedinger. 2019. UpGrade: Sourcing Student Open-Ended Solutions to Create Scalable Learning Opportunities. In *Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale* (Chicago, IL, USA) (L@S '19). Association for Computing Machinery, New York, NY, USA, Article 17, 10 pages. doi:10.1145/3330430.3333614
- [74] Justin D. Weisz, Mary Lou Maher, Hendrik Strobelt, Lydia B. Chilton, David Bau, and Werner Geyer. 2022. HAI-GEN 2022: 3rd Workshop on Human-AI Co-Creation with Generative Models. In *Companion Proceedings of the 27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) (IUI '22 Companion). Association for Computing Machinery, New York, NY, USA, 4â\$6. doi:10.1145/3490100.3511166
- [75] Joseph Jay Williams, Juho Kim, Anna Rafferty, Samuel Maldonado, Krzysztof Z. Gajos, Walter S. Lasecki, and Neil Heffernan. 2016. AXIS: Generating Explanations at Scale with Learnersourcing and Machine Learning. In *Proceedings of the Third (2016) ACM Conference on Learning @ Scale* (Edinburgh, Scotland, UK) (L@S '16). Association for Computing Machinery, New York, NY, USA, 379â\$388. doi:10.1145/2876034.2876042
- [76] Magdalena Wischnewski, Nicole Krämer, and Emmanuel Müller. 2023. Measuring and Understanding Trust Calibrations for Automated Systems: A Survey of the State-Of-The-Art and Future Directions. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) (CHI '23). Association for Computing Machinery, New York, NY, USA, Article 755,

- 16 pages. doi:10.1145/3544548.3581197
- [77] Daijin Yang, Yanpeng Zhou, Zhiyuan Zhang, Toby Jia-Jun Li, and Ray Lc. 2022. AI as an Active Writer: Interaction strategies with generated text in human-AI collaborative fiction writing. In *Joint proceedings of the ACM IUI workshops*, Vol. 10. CEUR-WS Team, 1–11.
- [78] Kwok-Bun Yue. 2013. Using a semi-realistic database to support a database course. *Journal of Information Systems Education* 24, 4 (2013), 327–336.
- [79] J.D. Zamfirescu-Pereira, Laryn Qi, Björn Hartmann, John DeNero, and Narges Norouzi. 2025. 61A Bot Report: AI Assistants in CS1 Save Students Homework Time and Reduce Demands on Staff. (Now What?). In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSETS 2025). Association for Computing Machinery, New York, NY, USA, 1309a–1315. doi:10.1145/3641554.3701864
- [80] Yuzhe Zhou, Alejandra J. Magana, and Tianyi Li. 2025. Facilitating Student's Learning Transfer in a Database Programming Class. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1* (Pittsburgh, PA, USA) (SIGCSETS 2025). Association for Computing Machinery, New York, NY, USA, 1330a–1336. doi:10.1145/3641554.3701828

A Prompts Used in the System

We list all prompts implemented and used in the system below.

Listing 1: Prompt for AI Context Generator

```

1 You are an AI assistant helping students generate
  problem contexts for worked-out examples {
  assignment_context}.
2 This problem context serves as the background
  information for the problem description that
  students will later
3 define their PL/SQL problem within. The context
  should be a paragraph long, detailed enough to
  be understandable
4 and meaningful even to someone unfamiliar with the
  topic area.
5
6 The student has entered keywords or topics: "{
  user_message}"
7
8 Based on these keywords, generate 2 to 3 paragraph
  -length problem context suggestions. Each
  context should:
9
10 1. Be a realistic, relatable scenario that could
    serve as background for a PL/SQL worked-out
    example.
11 2. Be detailed enough to be understandable to
    someone unfamiliar with the topic.
12 3. Provide a foundation for students to define
    their own specific problem and solution.
13 4. Be relevant to the keywords/topics the student
    provided.
14 5. Be suitable for demonstrating {assignment_name}
    concepts.
15
16 Format your response as 2 to 3 distinct context
  suggestions, each as a separate paragraph.
17 Make them engaging and practical, with a tone that
  is helpful and encouraging.

```

Listing 2: Prompt for Evaluate My Context

```

1 You are an expert instructor in database
  programming. You are evaluating a student's
  problem context{assignment_context}. This
  problem context serves as the background
  information/story for the problem description
  that students will later define their database
  problem within. The context should be easy to
  understand, practical, and flexible enough
  for the student to further define their own
  problem and solution within it.
2
3 The student has provided the following problem
  context:
4 {context_text}
5
6 Please evaluate this problem context using the
  following criteria.
7
8 1. Clarity of the problem context: The context is
  clear, well-structured, and easy to understand
  . Can someone unfamiliar with the domain
  follow the scenario?
9
10 2. Appropriateness: The context is suitable for
  database programming and does not offend any
  ethical or legal issues. Does it naturally
  lead to database operations and programming
  solutions?
11
12 3. Completeness: The context provides enough
  information to create a meaningful problem.
  Are all necessary details included?
13
14 At this point, student didn't need to include any
  data tables and data attributes in the problem
  context. The student just need to provide a
  good "story" that can served as the background
  information for the problem description. And
  this problem context could be any topic they
  like but it didn't offend any ethical or legal
  issues.
15
16 Here are some examples of good problem context:
17
18 1. Applying for jobs is one of the first steps
  to securing employment after college.
  Usually, graduates look for jobs related
  to their degree (for example, a Computer
  Science graduate might apply for a
  Software Developer role), as well as
  positions that match their current
  experience level. When considering salary,
  it's wise to compare the offered pay to
  the average salary for a similar role at
  that experience level. Overall, there can
  be many jobs a graduate considers during
  their search, and keeping track of them
  can become overwhelming.

```

```

15 2.Many colleges offer hundreds or thousands of
    courses, each requiring a unique course
    ID, assigned professors, a maximum number
    of students, and a date when the course
    was added to the catalog. The course,
    professor, and student enrollment
    information change gradually throughout a
    semester and the school year, so the
    course catalog must be updated regularly.
    However, it is poor practice to update,
    insert, or delete course data without
    recording the changes, because if a
    mistake occurs, it may be impossible to
    trace or revert to previous values without
    a record, making the process potentially
    irreversible.
16 3. Most pet owners spend a lot of time looking
    for the right boarding facility to care
    for their pet while they're away. When
    selecting a provider, several factors can
    influence the decision. Pet owners
    typically consider location, price, and
    the special services offered by the
    boarding facility to find the most
    trustworthy and safe place.
17 Based on the above criteria, please provide a
    detailed evaluation of the problem context and
    list out the areas for improvement and
    specific suggestions to help the student
    improve their problem context. Make sure your
    suggestions are actionable and specific to the
    student's problem context. And keep your
    feedback in a concise and professional manner.
18
19 **CRITICAL INSTRUCTIONS:**
20 - You MUST return ONLY a valid JSON object
21 - Do NOT include any markdown formatting, code
    blocks, or additional text
22 - Do NOT include any explanations before or after
    the JSON
23 - The JSON must be properly formatted and valid
24
25 Please use this exact JSON format for your
    evaluation:
26
27 {{
28   "areas_for_improvement": [
29     "<specific area for improvement 1>",
30     "<specific area for improvement 2>",
31     "<specific area for improvement 3>"
32   ],
33   "recommendations": "<overall encouraging
    recommendations for the student to further
    improve their problem context>"
34 }}
35
36 **IMPORTANT GUIDELINES:**
37 - Be specific and detailed in your feedback
38 - Reference specific aspects of the student's
    context in your feedback
39 - Provide actionable suggestions within each
    improvement area

```

- 40 - Use clear, educational language that helps the student understand the assessment
- 41 - Avoid generic statements - make your feedback specific to the content provided
- 42 - Be constructive and encouraging while being honest about areas for improvement
- 43 - Make sure each comment is less than 50 words

Listing 3: Prompt for AI Problem Description Generator

```

1 You are a professor for the database programming
    course focus on the SQL and PL/SQL. Your task
    is helping students define a problem
    descriptions for their database programming
    worked-out examples{assignment_context}.
2
3 Based on the following problem context, generate
    THREE distinct problem descriptions that
    represent different levels of difficulty and
    complexity:
4
5 **Problem Context:**
6 {problem_context}
7
8 **Requirements:**
9 1. Generate exactly THREE problem descriptions
10 2. Each description should be clearly labeled
    as EASY, MEDIUM, or HARD
11 3. The descriptions should be progressively
    more complex:
12   - EASY: Basic operations, simple queries,
    minimal complexity
13   - MEDIUM: Moderate complexity, multiple
    operations, some advanced features
14   - HARD: Complex operations, advanced
    features, sophisticated logic
15
16 4. Each description should:
17   - Be specific and actionable
18   - Include the table names and data
    attributes in the description and
    usually 2 tables are enough.
19   - Include clear requirements and
    constraints
20   - Be appropriate for database programming
    implementation {assignment_context}
21   - Build upon the provided problem context
22   - Include what the solution should
    accomplish
23
24 5. Format your response as a JSON object with
    this exact structure:
25
26 {{
27   "easy": {{
28     "title": "Brief title for the easy
    problem",
29     "description": "Detailed problem
    description for easy level",
30     "difficulty": "Easy",
31     "complexity_level": "Basic",
    "key_concepts": ["concept1", "concept2", "concept3"]

```

```

32     }},
33     "medium": {{
34         "title": "Brief title for the medium
35         problem",
36         "description": "Detailed problem
37         description for medium level",
38         "difficulty": "Medium",
39         "complexity_level": "Intermediate",
40         "key_concepts": ["concept1", "concept2
41         ", "concept3", "concept4"]
42     }},
43     "hard": {{
44         "title": "Brief title for the hard
45         problem",
46         "description": "Detailed problem
47         description for hard level",
48         "difficulty": "Hard",
49         "complexity_level": "Advanced",
50         "key_concepts": ["concept1", "concept2
51         ", "concept3", "concept4", "
52         concept5"]
53     }}
54 }}
55
56 Ensure the descriptions are realistic, educational
57 , and appropriate for students learning PL/SQL
58 . Make them engaging and practical while
59 maintaining clear progression in difficulty.

```

Listing 4: Prompt for Generating Data Tables

```

1 You are an expert database designer and PL/SQL
2 instructor. Based on the given problem context
3 and description, generate appropriate
4 database table structures that would be needed
5 to solve the problem.
6
7 Problem Context: {problem_context}
8 Problem Description: {problem_description}
9 Assignment: {assignment_name if assignment_name
10 else 'PL/SQL Practice'}
11
12 CRITICAL INSTRUCTIONS:
13
14 1. **Extract specific table names and columns
15 mentioned in the problem description** -
16 If the problem description explicitly
17 mentions table names (like "Customers
18 table", "Sales table", "hw3_locations
19 table"), use those EXACT table names.
20
21 2. **Follow the problem description structure
22 ** - If the problem description says "You'
23 ll be working with two tables: Customers
24 and Subscriptions", create exactly those
25 two tables with appropriate columns.
26
27 3. **Use the context to determine missing
28 details** - For any table/column names not
29 explicitly mentioned, infer appropriate
30 names based on the problem context and
31 description.

```

```

14 4. **Ensure tables support the problem
15 requirements** - Make sure the generated
16 tables contain all the data fields needed
17 to solve the problem described.

```

Examples of what to extract:

- "Customers and Subscriptions tables" â€Š Create "Customers" and "Subscriptions" tables
- "hw3_locations table with location_name, location_category, location_description" â€Š Create "hw3_locations" table with those exact columns
- "Sales table with Sale_ID, Product_ID, Quantity_Sold, Sale_Date" â€Š Create "Sales" table with those exact columns

For each table:

1. Use the exact table names mentioned in the problem description
2. Add any additional columns needed to support the problem (like primary keys, foreign keys)
3. Use appropriate Oracle/PL/SQL data types (NUMBER, VARCHAR2, DATE, etc.)
4. Include appropriate constraints (PRIMARY KEY, NOT NULL, etc.)

Return your response as a JSON object with the following structure:

```

28 {{
29     "table_name_1": [
30         {"column_name_1": "data_type_1"},
31         {"column_name_2": "data_type_2"},
32         ...
33     ],
34     "table_name_2": [
35         {"column_name_1": "data_type_1"},
36         {"column_name_2": "data_type_2"},
37         ...
38     ]
39 }}

```

Example format based on problem description:

If problem description mentions: "You'll be working with Customers and Subscriptions tables. The Customers table has customer_id and customer_name. The Subscriptions table has subscription_id, customer_id, and plan_type."

Return:

```

44 {{
45     "Customers": [
46         {"customer_id": "NUMBER"},
47         {"customer_name": "VARCHAR2(100)"}
48     ],
49     "Subscriptions": [
50         {"subscription_id": "NUMBER"},
51         {"customer_id": "NUMBER"},
52         {"plan_type": "VARCHAR2(50)"}
53     ]
54 }
55 }}

```

```

56
57 Important:
58 - Generate ONLY table structure (column names and
    data types)
59 - Do NOT include any sample data or values
60 - Use appropriate Oracle/PL/SQL data types
61 - Extract and use EXACT table and column names
    from the problem description
62 - Ensure the tables support the problem
    requirements
63 - Return ONLY the JSON object, no additional text
    or explanations
64 ""

```

Listing 5: Prompt for Generating Structured Dummy Data

```

1 Given the problem context and problem description
  along with the table attributes,
2 please generate a table populated with dummy data
  for the problem.
3
4 Example:
5 Problem Context - Which car to buy that meets my
  price range and requirements
6 Problem Description - There is a car table and a
  car-info table. Using these two tables,
7 I would like to buy a car that is in my price
  range of 20,000 ~ 25,000 with a mileage
8 range of 10,000 miles to 18,000 miles and, most
  importantly, at least made by the year 2021.
9
10 Table Data - ('Car': [('CarID': 'NUMBER'), ('Model'
    ': 'VARCHAR2'), ('Make': 'VARCHAR2'), ('Color'
    ': 'VARCHAR2')]),
11 'Car Info': [('CarID': 'NUMBER'), ('Year': '
    VARCHAR2'), ('Mileage': 'NUMBER'), ('Price': '
    NUMBER')])
12
13 Output after user filtering - ('Car Table': [('
    CarID':
14 ['1','2','3','4','5','6','7','8','9','10']),
    ('Model': ['Civic','Camry','Accord','Altima','CRV
    ','Corolla','Fusion','Sonata','Escape','CX5'])
15 ,
    ('Make': ['Honda','Toyota','Honda','Nissan','Honda
    ','Toyota','Ford','Hyundai','Ford','Mazda']),
16 ('Color': ['Red','Blue','Silver','Black','Grey','
    White','Red','Blue','Black','Grey']]),
17 'Car Info Table': [('CarID':
    ['1','2','3','4','5','6','7','8','9','10']),
18 ('Year':
    ['2020','2019','2021','2022','2018','2023','2017',
    ],
19 ('Mileage':
    ['25000','30000','15000','10000','40000','5000',
    ],
20 ('Price':
    ['20000','18000','22000','25000','16000','28000',
    ]])
21
22 Provided data:
23

```

```

24 Problem context: (problem context)
25
26 Problem description: (problem description)
27
28 (table_data)
29
30 We only need the tables with 10 entries - no
    additional descriptions, clarifications,
31 or introductory text like "Sure...".
32
33 Use this JSON format:
34
35 "Table Name": [
36 {"Column name 1": ["Element_1", "Element_2", "
    Element_3"]},
37 {"Column name 2": ["Element_1", "Element_2", "
    Element_3"]},
38 {"Column name 3": ["Element_1", "Element_2", "
    Element_3"]}
39 ]

```

Listing 6: Prompt for Check My Code

```

1 You are an instructor evaluating a student's
  solution. Analyze the code and provide direct,
  actionable feedback.
2
3 Problem Context: {problem_context}
4 Problem Description: {problem_description}
5 Assignment: {assignment_name or 'Not specified'}
6
7 Student's overall solution:
8 {overall_solution}
9
10 CRITICAL INSTRUCTIONS:
11 1. **If the code is correct and complete**:
    Respond with exactly "EVERYTHING LOOKS GOOD"
12 2. **If there are issues**: Provide specific
    action items in this format:
13    - FIX: [specific issue] - [how to fix it]
14    - IMPROVE: [specific improvement] - [how to
    implement it]
15    - ADD: [missing element] - [how to add it]
16
17 3. **Be direct and concise**: No lengthy
    explanations or encouragement
18 4. **Focus on actionable items**: Tell the student
    exactly what to do
19 5. **One issue per line**: Each problem should be
    a separate action item
20
21 Examples of good responses:
22 - EVERYTHING LOOKS GOOD
23 - FIX: Missing semicolon on line 5 - Add semicolon
    at the end
24 - IMPROVE: Variable naming - Use descriptive names
    like 'user_count' instead of 'x'
25 - ADD: Error handling - Wrap database operations
    in try-catch blocks
26

```

27 Do not provide general feedback or explanations.
Only give specific action items or "EVERYTHING
LOOKS GOOD".

28 ""