600 Jahre Universität Rostock

Traditio et Innovatio

Master Thesis on the Topic

# Can AI Solve It? Systematic Investigation of the Ability of Generative AI to Answer Database Exams


| | |
|---|---|
| Degree Course: | Computer Science |
| Author: | Nour Aouadi |
| Enrollment Number: | 220202475 |
| Work Period: | 20. January 2026 – 23. June 2026 |
| Supervisor: | Dr. Hannes Grunert |
| Secondary Supervisor: | Prof. Peter Daniels |

Database Research Group
Institute of Computer Science
Faculty of Computer Science and Electrical Engineering
University Rostock


44.5mm27.5mm

# Contents

# List of Figures

# List of Tables

# Abstract

Large Language Models (LLMs) such as Claude, Gemini, Kimi, and ChatGPT are increasingly used in higher education to support learning and solve academic tasks. This thesis investigates how reliably LLMs answer database examination questions and how their performance changes under different evaluation conditions. The study combines three empirical parts. First, Claude, Gemini, and Kimi were benchmarked on real Database and Data Warehouses exam questions using the professor's grading criteria. Claude achieved the highest overall correctness with 85.6 %, followed by Kimi with 63.4 % and Gemini with 59.4 %. Second, selected questions were reformulated to test model robustness. The reformulated benchmark showed that stricter wording, output constraints, and multi-step requirements reduced answer quality, especially for Kimi and Gemini. Third, a real-life experiment showed that LLM assistance improved average correctness from 34.7 % to 63.8 %, but the benefit depended on prompting strategy, context, input format, and the student's ability to verify the generated answer. Overall, the results show that LLMs can solve many database-related exam tasks, but they are not consistently reliable. Their performance depends strongly on task design, prompt formulation, input modality, language, available context, and user interaction. This thesis therefore provides empirical evidence for the need to design AI-aware assessment methods in higher education.

**Keywords:** Artificial Intelligence, Generative AI, Large Language Models, Benchmarking, Question Reformulation.

# Abstrakt

Large Language Models (LLMs) wie Claude, Gemini, Kimi und ChatGPT werden zunehmend in der Hochschulbildung eingesetzt, um Studierende beim Lernen und bei der Bearbeitung akademischer Aufgaben zu unterstützen. Diese Masterarbeit untersucht, wie zuverlässig LLMs Prüfungsfragen aus dem Bereich Datenbanken beantworten und wie sich ihre Leistung unter verschiedenen Bewertungsbedingungen verändert. Die Studie besteht aus drei empirischen Teilen. Im ersten Teil wurden Claude, Gemini und Kimi anhand realer Prüfungsfragen aus den Kursen Datenbanken und Data Warehouses getestet. Die Bewertung erfolgte auf Grundlage der Bewertungskriterien des Professors. Claude erreichte mit 85,6 % die höchste Gesamtgenauigkeit, gefolgt von Kimi mit 63,4 % und Gemini mit 59,4 %. Im zweiten Teil wurden ausgewählte Fragen umformuliert, um die Robustheit der Modelle zu prüfen. Die Ergebnisse zeigten, dass strengere Formulierungen, klare Ausgabevorgaben und mehrstufige Anforderungen die Qualität der Antworten verringerten, besonders bei Kimi und Gemini. Im dritten Teil wurde in einem realitätsnahen Experiment untersucht, wie sich die Nutzung von LLMs auf die Leistung von Studierenden auswirkt. Dabei stieg die durchschnittliche Korrektheit der Antworten von 34,7 % auf 63,8 %. Der Nutzen der LLM-Unterstützung hing jedoch stark von der Prompting-Strategie, dem vorhandenen Kontext, dem Eingabeformat und der Fähigkeit der Studierenden ab, die generierten Antworten kritisch zu überprüfen. Insgesamt zeigen die Ergebnisse, dass LLMs viele datenbankbezogene Prüfungsaufgaben lösen können, jedoch nicht durchgehend zuverlässig sind. Ihre Leistung hängt stark vom Aufgabendesign, der Prompt-Formulierung, der Eingabemodalität, der Sprache, dem verfügbaren Kontext und der Interaktion mit den Nutzenden ab. Diese Masterarbeit liefert daher empirische Hinweise darauf, dass Hochschulen KI-bewusste Prüfungs- und Bewertungsmethoden entwickeln sollten.

**Keywords:** Künstliche Intelligenz, Generative KI, Large Language Models, Benchmarking, Frageumformulierung.

Abstract

# Chapter 1

## Introduction

The rapid adoption of Generative Artificial Intelligence tools has changed the way students learn, prepare for examinations, and approach academic problem-solving. Large Language Models (LLMs), such as ChatGPT [1] and similar systems, are now often used to receive quick answers to complex academic questions. Recent empirical studies show that many university students have already included generative AI tools in their study routines, for example when preparing for exams or writing assignments [1].

This increasing use of LLMs can be explained by their ability to produce fluent, well-structured, and contextually appropriate answers in many subject areas. However, a well-written answer is not always a reliable answer. Research has shown that strong linguistic quality does not necessarily mean that a model has real understanding or that its reasoning is correct. Bender et al. argue that LLMs generate statistically plausible text without true semantic grounding [2]. Therefore, LLM-generated answers may appear convincing while still containing small errors, missing details, or logical inconsistencies.

The growing role of generative AI in education also raises important concerns about academic integrity, assessment validity, and fairness. Previous studies have shown that AI systems can perform well on standardized tests [3] and even on professional examinations [4]. At the same time, other research points to clear weaknesses, especially in tasks that require multi-step reasoning, compositional thinking, and planning [5, 6].

This creates an important tension for higher education. On the one hand, LLMs can support learning by providing explanations, summaries, examples, and alternative perspectives. On the other hand, their use in examination settings can challenge traditional assessment formats and may reduce the validity of exams if students rely on AI-generated solutions instead of their own understanding. This thesis does not argue against the use of generative AI in education. Instead, it takes the position that LLMs should be used as complementary tools that support reflection and understanding, rather than as complete substitutes for independent reasoning and knowledge acquisition.

[1] https://chatgpt.com/

The educational impact of generative AI is therefore broader than assessment alone. It also raises the question of how students develop independent reasoning, problem-solving skills, and the ability to critically evaluate information when AI-generated answers are easily available. However, this thesis focuses specifically on university-level database examinations. It investigates how reliably LLMs answer database-related exam questions and how robust traditional assessment formats remain when such systems are available to students.

## 1.1 Motivation

Despite the rapidly growing research on generative AI, there is still limited systematic evidence on how LLMs perform in university database examinations. Many existing studies evaluate LLMs on general reasoning tasks, standardized tests, or professional exams such as medical examinations [7]. However, fewer studies focus on domain-specific technical exams and on the way exam questions are structured. This is especially important because the design of a question may strongly influence how well an LLM can answer it. For example, small changes in wording, required output format, or task structure may affect the quality and correctness of the generated answer. Therefore, there is a clear research gap regarding how question design influences LLM performance and how reformulation strategies can reduce AI effectiveness while still keeping exams fair and pedagogically meaningful.

## 1.2 Problem Statement

If LLMs are capable of answering traditional exam questions with high accuracy, the reliability of those assessment formats must be reconsidered. At the same time, banning AI tools is neither realistic nor pedagogically constructive. A more sustainable approach is to systematically investigate:

- under which conditions LLMs provide correct and reliable answers,
- where their systematic weaknesses lie,
- and how exam questions can be redesigned to remain robust in the presence of generative AI.

Based on these considerations, the main research questions addressed in this thesis are:

- **RQ1:** How do LLMs generate answers to exam questions, and which LLMs provide the most accurate and reliable responses?
- **RQ2:** How can exam questions be reformulated in order to cause LLMs to generate incorrect or partially incorrect answers? What are the main weaknesses and limitations of generative AI when answering exam-style questions?

## 1.3 Structure of the Work

This thesis is organized into eight chapters:

- Chapter 2 provides the theoretical background necessary to understand the work. It introduces generative artificial intelligence, explains how Large Language Models (LLMs) function, and presents the concept of prompt engineering.
- Chapter 3 reviews related scientific work. It focuses on previous studies about LLM performance in exams, evaluation methodologies, and prompt sensitivity.
- Chapter 4 presents the first part of the experimental work, which benchmarks different LLMs on database examination questions. The methodology, evaluation metrics, and results are described and analyzed.
- Chapter 5 reports on an experimental study conducted with university students. It compares their performance on database examination tasks before and after using LLMs, providing insights into the educational impact of AI-assisted learning.
- Chapter 6 focuses on the second part of the thesis, where exam questions are systematically reformulated in order to analyze how such changes affect LLM performance.
- Chapter 7 discusses the results of the empirical chapters and highlights their implications for exam design and AI usage in education.
- Chapter 8 summarizes the main findings and outlines possible directions for future work.

# Chapter 2

# *Background and Definitions*

The ability of LLMs to generate fluent and structured responses explains why they often perform well on definition-based or descriptive exam questions [8]. When an exam question is similar to examples or explanations that may have appeared in the model's training data, the model can often produce a strong answer. In such cases, it may recognize familiar patterns and apply them in a way that appears accurate and well structured.

However, this does not mean that the model truly understands the task in the same way as a human learner. LLMs generate answers by predicting likely word sequences, not by reasoning from grounded knowledge. For this reason, their performance can become weaker when a task requires deeper reasoning, careful attention to specific constraints, or adaptation to a particular domain context [9]. This point is especially relevant for database examinations, where questions often include schemas, algorithms, formal procedures, or decisions that depend on the exact context of the task.

For this thesis, it is therefore important to understand how generative AI models work at a conceptual and algorithmic level. This background helps to interpret the benchmarking results and to explain why the models perform well on some database exam questions but struggle with others. If a model performs well on certain types of questions but fails on reformulated or context-specific variants, such behavior can be explained by the underlying statistical nature of its architecture.

That is why, in this chapter, we briefly define what generative AI is, introduce Large Language Models (LLMs) and explain how they work, and finally provide an overview of prompts and prompt engineering.

## 2.1 Generative Artificial Intelligence

Generative Artificial Intelligence (Generative AI) refers to a class of AI systems that are capable of producing new content such as text, images, code, or audio based on patterns learned from large amounts of training data [8]. Unlike traditional rule-based systems, generative AI models do not follow explicitly programmed instructions for each possible task. Instead, they learn statistical relationships within data and use these learned patterns to generate new outputs.

### 2.1.1 Large Language Models (LLMs)

In recent years, LLMs have become one of the most visible forms of generative Artificial Intelligence. These models are trained on very large collections of text and can generate answers that are fluent, coherent, and often relevant to the given context [9]. At a basic level, they work by predicting the most likely next word, or token, based on the text that came before it. Through repeated training over billions of text examples, they develop the ability to produce responses that resemble human-written text.

The rapid improvement of generative AI systems has significantly influenced various domains, including education. In academic contexts, LLMs are increasingly used to answer theoretical questions, explain technical concepts, and even solve structured exam tasks [8]. This development is directly relevant to the focus of this thesis, which investigates whether such systems are capable of reliably answering database examination questions.

### 2.1.2 How Large Language Models Work

Modern LLMs are mostly built on the Transformer architecture. This architecture uses a mechanism called self-attention, which helps the model identify relationships between words in a text [9]. In simple terms, self-attention allows the model to decide which parts of the input are most important for understanding the current context. Because of this, LLMs can connect information across longer passages and produce responses that often follow the structure and meaning of the input.

From an algorithmic perspective, LLMs are usually trained with large amounts of text through unsupervised or self-supervised learning. During this process, the model learns to predict the next token in a sequence and is adjusted whenever its prediction differs from the expected output [8]. Technically, this is done by calculating probabilities for possible next tokens and updating the model parameters with optimization methods such as gradient descent.

Importantly, LLMs do not explicitly encode logical rules or symbolic reasoning processes. Instead, they approximate reasoning patterns through statistical learning [9]. This distinction is crucial for this thesis. While LLMs can generate answers that appear logically structured, their reasoning is derived from learned correlations rather than explicit rule-based computation.

Recent versions of these models, like OpenAI's GPT-4o , are able to follow complex instructions, remember conversation context, generate structured outputs, and assist in multi-domain tasks. However, they remain highly sensitive to how the user formulates their prompt. Poorly worded prompts often lead to irrelevant, generic, or hallucinated outputs. Well-structured prompts, on the other hand, can guide LLMs toward precise and meaningful answers even in technical domains like databases and Data Science [10].

## 2.2 Prompt Engineering

Large Language Models do not generate answers independently; instead, their behavior is directly conditioned by the textual input they receive, commonly referred to as a prompt [10]. A prompt can therefore be defined as a structured instruction or query provided to the model in order to guide its response generation process [11].

Since LLMs operate by predicting the most probable next token based on contextual information, the formulation of the prompt directly influences the probability distribution over possible outputs [12]. As a result, even minor changes in wording, structure, or explicit instructions can significantly alter both the reasoning behavior and the final answer generated by the model [13].

**Example 1: Basic vs. Contextual Prompt**

**Prompt A:** "What is a primary key in a database?"
**Prompt B:** "Explain the role of a primary key in a database schema used in a real-world application."
Prompt A typically leads to a short and correct definition, while Prompt B requires a more detailed and context-aware explanation. This shows how adding context increases the complexity and depth of the response.

The systematic design and optimization of such inputs is known as prompt engineering [10]. It includes techniques such as zero-shot prompting, few-shot prompting, chain-of-thought prompting, and role-based instructions, each of which influences how the model interprets the task and structures its output [10, 11].

**Example 2: Prompting Techniques**

**Zero-shot:** "What is a foreign key in a relational database?"
**Few-shot:** Providing one or more small examples before asking a similar question.
**Chain-of-thought:** "Explain step by step whether this relation is in Second Normal Form."
**Role-based:** "Act as a database lecturer and explain indexing to a beginner."
These variations show that different prompting techniques lead to different response styles, levels of detail, and reasoning quality.

In particular, explicitly instructing the model to reason step by step has been shown to improve performance on complex multi-step reasoning tasks [13]. However, since LLMs rely on learned statistical correlations rather than explicit logical reasoning mechanisms, poorly designed prompts can lead to hallucinations, overconfident statements, or logically inconsistent answers [14].

**Example 3: Structured vs. Vague Prompt**

**Structured:** "Determine step by step whether this relation is in Third Normal Form and justify each step."
**Vague:** "Is this relation correct?"
The structured prompt encourages reasoning, while the vague prompt often leads to unclear or unreliable answers.

Recent studies further emphasize that the clarity and structure of prompts determine whether the model performs deeper reasoning or only produces superficial pattern-based answers [**ye2024promptengineer**]. This is particularly important in educational contexts, where exam questions themselves act as prompts that guide the model's interpretation of the task.

**Example 4: Simple vs. Analytical Question**

**Simple:** "Explain the difference between a star schema and a snowflake schema."

**Analytical:** "Explain the difference between a star schema and a snowflake schema and discuss which one is more suitable for analytical queries in a data warehouse."

The second version encourages deeper reasoning and application, while the first often results in a basic textbook answer.

Therefore, reformulating an exam question can be understood as a controlled modification of the prompt in order to evaluate the robustness, reasoning depth, and reliability of LLM-generated answers [10]. This understanding forms the conceptual foundation for the second part of this thesis, where different reformulation strategies are systematically applied to database exam questions.

# Chapter 3

# Related Work

The rapid progress of Large Language Models (LLMs) has led to a growing body of research investigating their performance on standardized tests, professional examinations, and structured reasoning tasks. The literature search for this chapter was conducted using Google Scholar [1], IEEE Xplore [2], and arXiv[3]. These sources were selected because they provide access to peer-reviewed publications, conference papers, preprints, and recent technical reports in the fields of artificial intelligence. In addition, the search started from two initial research papers provided in the thesis topic description by the supervisors [7][3]. The references cited in these papers were also reviewed in order to identify further relevant studies. The search terms included combinations of keywords such as “Large Language Models”, “LLMs”, “Generative AI”, “prompt engineering”, “LLM evaluation”, “LLMs in education”, “AI in examinations”, “benchmarking large language models”, “prompt sensitivity”, “robustness of LLMs”, and “question reformulation”. Additional searches were performed with more specific terms related to the focus of this thesis, such as “LLMs exam questions”, “LLMs reasoning tasks”, and “LLM performance in academic assessment”. The results were filtered based on their relevance to the research questions of this thesis. Priority was given to studies that evaluate LLMs in examination settings, propose systematic evaluation frameworks, analyze prompt sensitivity, or investigate robustness under reformulated inputs. Publications were also selected based on recency, scientific relevance, citation context, and their connection to higher education or structured reasoning tasks.
Since this thesis focuses on benchmarking LLMs on database examination questions and analyzing how question reformulation influences model performance, related work can be structured into three main areas: (1) Section 3.1, (2) Section 3.2, and (3) Section 3.3.

## 3.1 LLMs in Exam and Test Environments

One of the most directly relevant studies to this thesis is the work by Sadeq et al. [7], which evaluates multiple LLMs on official UK medical licensing examination questions. The authors report that modern models achieve relatively high overall accuracy; however, performance varies significantly depending on question type and structural complexity. Importantly, the study highlights that LLM

[1] https://scholar.google.com/
[2] https://ieeexplore.ieee.org/Xplore/home.jsp
[3] https://arxiv.org/

errors are often not purely factual mistakes but reasoning inconsistencies or failures in applying domain-specific constraints. In addition to academic studies, practical benchmarking platforms such as ArtificialAnalysis.ai [15] provide continuous and large-scale comparisons of LLM performance across diverse tasks, including reasoning, coding, and structured problem solving. These platforms show that model performance can vary significantly depending on task formulation, evaluation setup, and prompt design. This observation supports the core assumption of this thesis: that LLM performance in exam contexts is not fixed but highly dependent on how questions are structured and presented.

Similarly, the GPT-4 Technical Report [4] documents strong performance across standardized tests, while explicitly acknowledging limitations such as hallucinations, overconfidence, and reasoning failures. These findings reinforce the need for domain-specific benchmarking, particularly in structured domains such as database systems.

Earlier foundational work by Clark and Etzioni [3] already questioned whether success on standardized tests reflects true intelligence. They argue that strong performance may result from pattern recognition rather than deep understanding, a distinction that is central to interpreting the results of this thesis.

## 3.2 Systematic Evaluation Frameworks for LLMs

Beyond individual benchmarking studies, several works propose broader evaluation frameworks. The HELM framework [16] argues that evaluating LLMs requires a multidimensional perspective, including accuracy, robustness, efficiency, and reliability. Rather than reporting a single performance score, HELM emphasizes structured evaluation across different task categories. This approach directly informs the methodology of this thesis, particularly the decision to analyze LLM performance across different database question types and to complement accuracy metrics with robustness analysis.

In addition, recent survey work on LLM evaluation [17] categorizes evaluation approaches into capability testing, reasoning assessment, robustness analysis, and domain-specific benchmarking. The authors stress that evaluation must reflect realistic task conditions. This principle strongly supports the experimental design of this thesis, which uses authentic database examination questions instead of artificial benchmark tasks.

The BIG-bench initiative [18] further demonstrates that large-scale benchmarking across diverse tasks is essential to reveal capability gaps. The authors show that model performance can vary substantially depending on task formulation and reasoning demands. This finding directly motivates the structured comparison of multiple LLMs in this thesis.

## 3.3 Prompt Sensitivity and Robustness to Reformulation

Since the second part of this thesis focuses on reformulating exam questions, research on prompt sensitivity is particularly relevant. Wei et al. [13] demonstrate that chain-of-thought prompting can significantly improve performance on multi-step reasoning tasks. Their results indicate that LLM reasoning behavior is highly dependent on prompt structure. However, improvements are not universal, and some reasoning failures persist despite advanced prompting strategies.

Further work in the BIG-bench context [19] shows that even minor changes in task wording can cause significant performance variations. This sensitivity suggests that LLM outputs are strongly

influenced by surface-level linguistic cues. For this thesis, this insight provides theoretical justification for systematically reformulating database exam questions to test the robustness of model performance. The CheckList framework [20] introduces behavioral testing for NLP models by systematically perturbing inputs and observing output consistency. The authors argue that accuracy alone is insufficient, as models may fail under slight modifications that should not alter the correct answer. This perspective directly aligns with the reformulation experiments conducted in this thesis, where modified database questions are used to identify structural weaknesses in LLM reasoning.

## 3.4 Synthesis and Research Gap

Overall, prior research demonstrates that LLMs can achieve strong performance on standardized examinations, but also exhibit structural weaknesses in reasoning, robustness, and prompt sensitivity. While medical and general reasoning benchmarks have been extensively studied, there remains limited systematic investigation of LLM performance in domain-specific technical examinations such as university-level database courses.

Moreover, although prompt engineering and robustness testing are well established in NLP research, their application to structured academic exam design remains underexplored.

Therefore, this thesis addresses a clear research gap by combining systematic benchmarking of multiple LLMs on database examination questions with structured reformulation experiments that evaluate robustness and reasoning sensitivity. By integrating evaluation methodology with exam design considerations, this work contributes both empirical findings and practical recommendations for AI-aware assessment formats. Building on this research gap, the following Chapter 4 presents the first empirical part of this thesis: the systematic benchmarking of selected LLMs on real database examination questions.

Table 3.1: Summary of related work and its relevance to this thesis.

| **Reference** | **Area** | **Main insight** | **Relevance** |
|---|---|---|---|
| Sadeq et al. [7] | Exam benchmarking | Good overall accuracy, but weaknesses in complex reasoning | Motivates testing database exam questions |
| OpenAI [4] | Standardized tests | Strong benchmark performance, but hallucinations and reasoning errors remain | Supports evaluating reliability, not only correctness |
| Clark and Etzioni [3] | Evaluation theory | High scores do not necessarily imply genuine understanding | Helps interpret model answers beyond raw accuracy |
| Liang et al. [16] | Evaluation framework | Language models should be assessed across multiple dimensions | Guides the multi-dimensional evaluation in this thesis |
| Chang et al. [17] | Evaluation survey | Realistic and domain-specific evaluation is essential | Justifies using real database exam questions |
| BIG-bench authors [18] | Large-scale benchmarking | Performance varies across tasks and formulations | Supports comparing different question types and formats |
| Wei et al. [13] | Prompt engineering | Step-by-step prompting can improve reasoning | Supports reformulating questions to probe reasoning ability |

| | | | |
|---|---|---|---|
| Suzgun et al. [19] | Reasoning limits | Even advanced prompting fails on difficult reasoning tasks | Highlights limitations in structured domains |
| Ribeiro et al. [20] | Robustness testing | Small input changes can strongly affect model behavior | Motivates systematic reformulation of exam questions |
| ArtificialAnalysis.ai [15] | Practical benchmarking | Results depend strongly on task design and evaluation setup | Confirms the importance of question formulation |

# Chapter 4

# *Benchmarking of the LLMs' Responses*

This chapter presents the first experimental part of this thesis, which focuses on benchmarking the performance of selected Large Language Models on real database examination questions. The main goal is to evaluate how accurately and reliably these models answer questions from different database-related topics and formats when compared with the reference solutions provided by the professor. In addition to answer correctness, this chapter also investigates whether model performance changes depending on the language of the question and the way the question is presented to the model, for example as typed text or as an image. The results of these experiments provide the empirical basis for the later analysis of LLM strengths, weaknesses, and robustness in exam settings.

## 4.1 Methodology and description of the Experiments

The first part of the empirical study was designed as a controlled benchmarking experiment using authentic examination material from database-related courses. For this purpose, nine different exam questions, including their corresponding sub-questions, were selected from several database exams provided by the supervisor. These questions cover different thematic areas of databases, including, among others, Business Intelligence, SQL queries, and ontologies. For each question, a reference solution provided by the professor was available and served as the basis for the evaluation.
A key objective of this experiment was to ensure that the benchmark reflects the diversity of real examination settings. Therefore, the selected questions were intentionally taken from different types of tasks. Some questions required short conceptual explanations, others required more structured reasoning, while some involved diagram drawing, table completion, or graphical representations. Purely theoretical definition questions, such as "Define a star schema" or "What is a star schema?", were excluded from the main benchmarking question types. Such questions are usually direct and standardized, and their answers are widely available in textbooks, lecture material, and online resources. Since LLMs are trained on large amounts of text, they are often able to answer these common definition-based questions relatively easily by reproducing familiar explanation patterns. In contrast, this benchmark focused more on tasks that require applying knowledge to a concrete database-related problem, interpreting specific task constraints, or producing a structured output. This diversity was important because LLM performance can vary significantly depending on the format and cognitive

demands of the question. As a result, the benchmark was not limited to one single question type, but instead aimed to reflect a realistic range of database exam tasks. Three LLMs were selected for the experiments: Gemini 3, Claude Sonnet 4.0.6, and Kimi K2.5 Instant. These models were chosen because they represent widely used contemporary generative AI systems with strong text-generation capabilities and are suitable for direct comparison in an educational question-answering scenario. Each model was tested separately under the same general conditions in order to make the comparison as fair and consistent as possible.

To investigate whether the language of the prompt influences answer quality, selected exam questions were prepared in German, English, and French. Since the database exams were originally available mainly in German and English, these two languages formed the primary basis for comparison. French was added as an additional language in order to observe whether the same academic content produces different model behavior when expressed in another linguistic context. The purpose of this part of the experiment was not to build a full multilingual benchmark, but rather to observe whether the linguistic formulation of the same question affects correctness, completeness, or answer structure.

Another important methodological aspect was the way in which the questions were presented to the models. In real examination misuse scenarios, students do not always type questions manually into an AI tool, because this may take too much time. Instead, they may rely on quicker methods such as taking a photograph of the exam sheet. For this reason, the experiment included two different input modalities: manual text input and image-based input. In the manual setting, the exact exam question was typed into the LLM interface. In the image-based setting, a photo of the question was uploaded or provided to the model when the system supported image input. This comparison was included to examine whether LLM responses differ when the same task is provided in textual or visual form.

In order to avoid contamination between runs and to keep each answer independent, every question was tested in a new chat session. This setup increased the internal consistency of the experiment and made the comparison between responses more reliable.

In the general benchmarking setup, each exam question and sub-question was tested once per model. This decision was based on the results of the initial benchmarking in Subsection 6.2, where Question 1a was submitted multiple times to each model in separate chat sessions. The repeated runs did not lead to fundamentally different answers. Instead, the models mostly reproduced the same conceptual structure, with only minor variations in terminology, wording, and presentation. Therefore, repeating every question several times was not expected to provide significant additional insights and the remaining questions were tested once per model.

Question 1a, was treated as a more detailed pilot and comparison case. This question was asked three times to each model, each time in a new chat. One run used an image of the question, while two runs used manual text input. The purpose of this repeated testing was to observe whether repeated presentation of the same prompt leads to a more accurate answer, whether the model changes its response strategy, or whether it mainly reproduces the same answer structure with only small variations. Because Question 1a was used as a detailed demonstration case, its full responses and analysis are presented in greater depth in the benchmarking section, whereas the later questions are discussed more compactly to avoid unnecessary repetition. After each model generated its answer, the response was compared manually with the professor's reference solution. The evaluation was not based on a vague overall impression, but on a predefined scoring rubric developed for each question. This rubric assigned

points to the most important required answer components. If a student or an LLM omitted a required part, only answered it partially, or made a clear mistake, the corresponding points were reduced. The same grading logic was applied consistently to all models, just as it would be applied to a student answer. In this way, the correctness of LLM responses could be translated into a quantitative score and then into a percentage of correctness. To illustrate this grading procedure, Question 1a may be used as an example. The prompt given to the models was the following:

**Example: Exam Question as a Prompt**

**Question 1a:**

*Erstellen Sie eine Ontologie, welches den Besuch von Vorlesungen durch Studierende darstellt. Wählen Sie geeignete Klassen (mind. 4), charakterisieren Sie diese durch Eigenschaften und stellen Sie Beziehungen zwischen den Klassen her! Stellen Sie die Ontologie grafisch dar!*

This example shows that an exam question itself can function as a prompt. It does not only ask for factual knowledge, but also for structure, relationships, and a graphical representation. Therefore, the quality of the LLM response depends not only on its knowledge of ontologies, but also on how well it interprets the different requirements contained in the prompt.

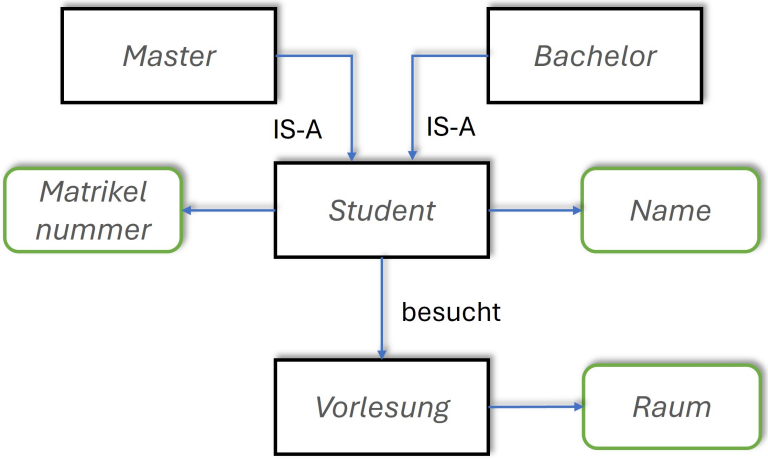



Figure 4.1: Professor's Answer for Question 1(a)

The corresponding professor solution required, in summarized form, a correct ontology representation including entities such as *Student*, *Vorlesung*, and *Raum*, the relation *besucht*, a hierarchy such as *Master IS-A Bachelor*, and a graphical representation of the ontology as shown in Figure 4.1.

Table 4.1: Scoring rubric for Question 1a

| **Criterion** | **Points** |
|---|---|
| Classes (at least 4) | 1 |
| Attributes / datatype properties | 1 |
| Relations | 1 |
| Graphical representation | 3 |
| **Total** | **6** |

To make the evaluation procedure transparent, Table 4.1 presents the scoring rubric used for Question 1a. The same general principle was applied to the other questions, although the specific criteria varied depending on the task type and expected solution.

This results in a total of 6 points. If a required component was missing, incomplete, or incorrect, the corresponding points were reduced. The same principle was applied to the remaining questions, with task-specific adjustments depending on the structure and requirements of each question.
In addition to answer correctness, a small time-based comparison was also conducted. For Question 1a, the time required to enter the question manually or to take a photo of it was measured, as well as the time each LLM needed to generate a full response. This time analysis was intended as an exploratory comparison rather than a full benchmark across all questions. Its purpose was to provide an initial indication of the practical speed advantage of AI systems in examination-like situations and to compare the effort of different input methods.
Overall, the methodology of this chapter combines authentic exam material, controlled model interaction, structured scoring, multilingual variation, and input-modality comparison. This design makes it possible to evaluate not only whether selected LLMs can answer database exam questions correctly, but also under which conditions their answers improve, remain stable, or become less reliable.

## 4.2 Time Comparison

In addition to evaluating the correctness of LLM-generated answers, this study also investigates the time efficiency of different input methods and model responses. The objective of this experiment is to simulate a realistic exam scenario and to analyze how quickly a student can interact with an LLM, as well as how fast the model generates a complete answer. The analysis focuses on two main aspects. First, the time required by the user to provide the question to the model, either by manually typing the prompt or by taking a photo of the exam question. Second, the time required by the LLM to process the input and generate a full response. The experiment was conducted using Question 1a as a representative example, since it includes both conceptual and structural elements. The same question was tested using both input modalities across three LLMs: Gemini 3, Claude Sonnet 4.0.6, and Kimi K2.5 Instant. The time measurement was performed once for each model and input modality. Therefore, the results should be understood as an exploratory comparison rather than as a statistically repeated experiment.

Table 4.2: Detailed time comparison of text-based and photo-based workflows (in seconds)

| **Model** | **Text Input Workflow** | | | **Photo Input Workflow** | | |
|---|---|---|---|---|---|---|
| | Input Time (s) | Response Time (s) | **Total (s)** | Input Time (s) | Response Time (s) | **Total (s)** |
| Gemini | 126 | 17.09 | **143.09** | 16.3 | 24.47 | **40.77** |
| Claude | 126 | 90 | **216** | 13.77 | 76.8 | **90.57** |
| Kimi | 186 | 63 | **249** | 36.66 | 191.4 | **228.06** |

To provide a descriptive overview of the measured times, mean values and standard deviations were calculated across the three models for each workflow. In the text-based workflow, the mean user input time was 146.00 seconds with a standard deviation of 34.64 seconds, while the mean LLM response time was 56.70 seconds with a standard deviation of 36.88 seconds. This resulted in a mean total

interaction time of 202.70 seconds with a standard deviation of 54.18 seconds. In the photo-based workflow, the mean user input time was 22.24 seconds with a standard deviation of 12.54 seconds, while the mean LLM response time was 97.56 seconds with a standard deviation of 85.38 seconds. The mean total interaction time for this workflow was 119.80 seconds with a standard deviation of 97.05 seconds. These descriptive values indicate that photo-based input substantially reduced the time required from the user. However, the total interaction time did not depend only on the input method, but also on the response behavior of the individual model. This is especially visible in the high variation of response times in the photo-based workflow. To provide a clearer overview of the time experiment, the results of both input methods and response times are combined into a single stacked diagram.

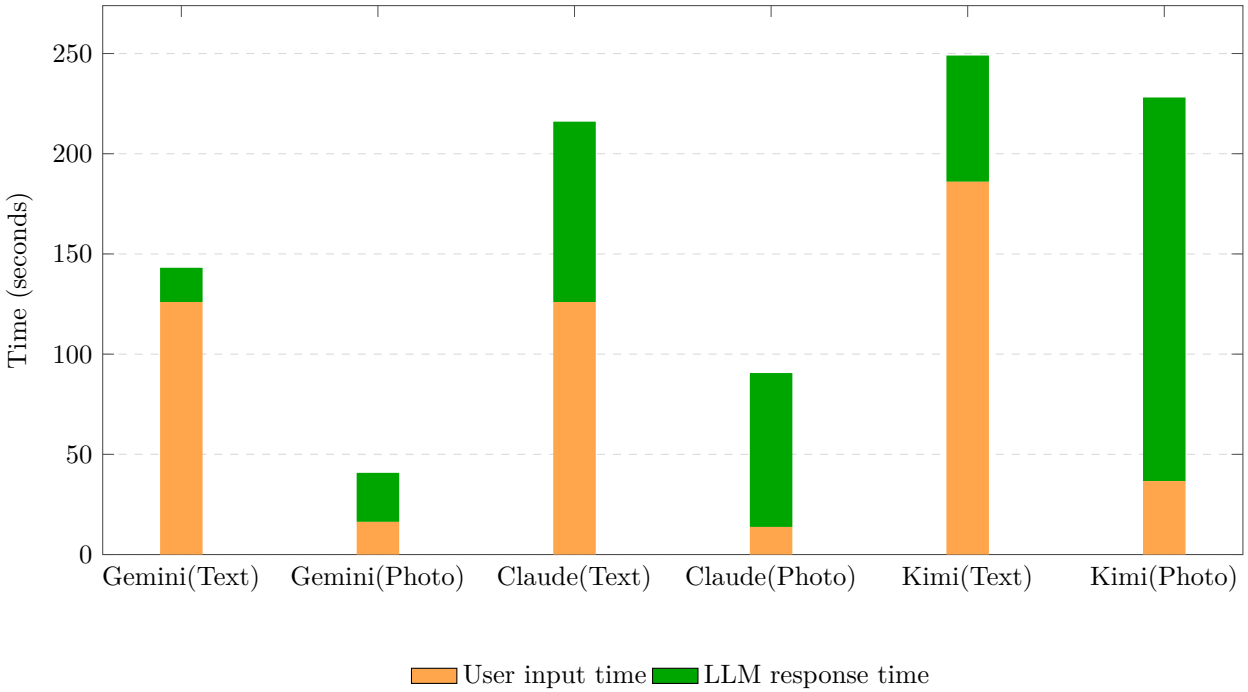



Figure 4.2: Total interaction time for Question 1a across text-based and photo-based workflows

Figure 4.2 shows the total interaction time for Question 1a across text-based and photo-based workflows for Gemini, Claude, and Kimi. A clear difference can be observed between the two input methods. When the prompt was entered manually, the user needed approximately 126 seconds for Gemini and Claude, while the input time for Kimi increased to 186 seconds. This means that the manual input process for Kimi took about 60 seconds longer than for the other two models. Since this time was measured on the user side, this difference should not be interpreted as a pure model characteristic. This difference should not be interpreted as a pure model-performance characteristic. During manual input, the Kimi interface showed noticeable latency: the typed text appeared with a delay, which increased the measured input time. Therefore, the longer manual-input duration for Kimi reflects both the user interaction process and interface responsiveness, not only the complexity of the prompt.
In contrast, photo-based input was much faster for all models. The photo workflow required only 16.3 seconds for Gemini and 13.77 seconds for Claude, while Kimi required 36.66 seconds. Even in this case, photo input remained clearly faster than manual typing. However, the advantage was smaller for Kimi because image-based prompting on Kimi could not be performed without an additional written instruction.

On the model side, Gemini was the fastest system in both workflows. After manual input, it generated a full answer in only 17.09 seconds, and after photo input in 24.47 seconds. Claude was much slower, with 90 seconds for text input and 76.8 seconds for photo input, although its performance remained relatively stable across both modalities. Kimi showed a mixed pattern. Its response time for text input was 63 seconds, which is lower than Claude's but clearly higher than Gemini's. However, for photo-based input, Kimi required 191.4 seconds, which is by far the highest value in the experiment. The experiment reveals a clear trade-off between input speed and response efficiency. While image-based input significantly reduces the time required from the user, it does not always lead to faster overall interaction time. The performance strongly depends on the capabilities of the specific LLM.
Gemini appears to provide the best balance between fast input processing and rapid response generation. Claude, although slower, may prioritize more detailed reasoning. Kimi shows limitations in multimodal interaction, which affects both usability and performance.
These findings are particularly relevant in the context of exam environments. They suggest that although LLMs can provide fast answers, the effectiveness of their use depends on both the interaction method and the specific model. Therefore, time efficiency alone cannot be considered independently from model behavior and input modality.

## 4.3 Initial Benchmarking

In order to evaluate the performance of the selected Large Language Models (Kimi, Gemini, and Claude), the analysis first focuses on Question 1a, which was introduced in Section 4.1 and used as a representative example for the initial benchmarking.
The evaluation focuses on the correctness and consistency of the generated answers, based on the official grading scheme provided by the professor. According to this scheme Figure 4.1, a total of 6 points can be achieved: 1 point for classes, 1 point for attributes, and 4 points for relations and graphical representation as explained in Section 4.1 .The results show that all models are capable of generating valid ontology structures, but their performance differs in terms of completeness and adherence to the formal requirements of the task.

Table 4.3: Correctness Results for Question 1a

| **LLM** | **Points** | **Max Points** | **Correctness (%)** |
|---|---|---|---|
| Kimi | 6 | 6 | 100 % |
| Claude | 4 | 6 | 66.66 % |
| Gemini | 3 | 6 | 50 % |

A closer analysis of the results reveals important differences between the models. Kimi achieves the highest score (100 %) and provides a complete solution that satisfies all requirements of the task. It includes more than four classes, detailed attributes, well-defined relationships, and a correct graphical representation. Although the structure of the ontology is not identical to the professor's solution, it fully meets the expected criteria and demonstrates a strong understanding of the task.
Claude also produces a detailed and logically consistent answer, achieving 66.66%. The model correctly identifies the required components and provides a complete ontology structure. However, the graphical

representation is implemented as a UML diagram rather than a standard ontology diagram. While this still reflects a correct interpretation of the relationships, it does not fully comply with the expected format, which leads to a partial loss of points.

Gemini obtains the lowest score (50 %). The model successfully identifies the main classes, attributes, and relationships, which shows that it understands the conceptual structure of the task. However, it fails to provide a graphical representation of the ontology. Since this component accounts for the largest portion of the total score, its absence has a significant negative impact on the final result.

Another important observation is that all LLMs generate very long and detailed answers. In many cases, the responses include additional explanations, extended structures, or multiple types of diagrams that are not explicitly required by the question. While this reflects strong generative capabilities, it may reduce clarity and make the answers more difficult to interpret in an exam context, where concise and precise responses are expected . To better illustrate the differences in performance, Figure 4.3 presents a stacked bar chart showing the distribution of points across the evaluation criteria.

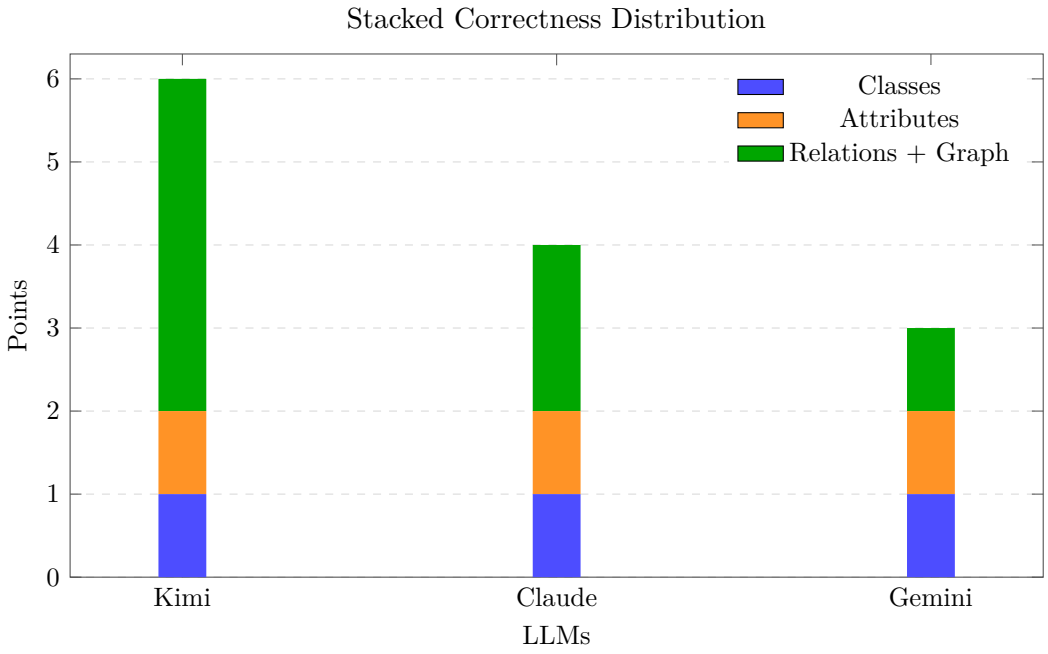


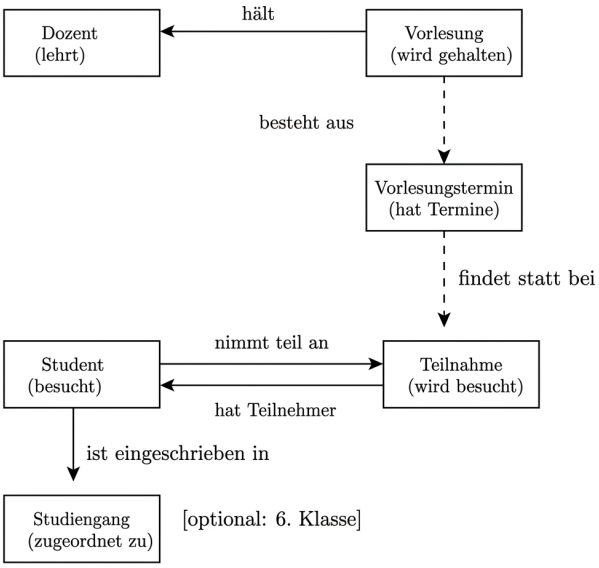

Figure 4.3: Stacked correctness distribution for Question 1a across the evaluated LLMs

The stacked diagram provides a clearer understanding of how each model performs across the different components of the evaluation. It shows that all models achieve similar results in identifying basic elements such as classes and attributes. These components are relatively straightforward and are handled consistently by all LLMs. However, the main differences appear in the relations and graphical representation component, which has the highest weight in the grading scheme. Kimi fully satisfies this requirement, resulting in a complete score. Claude partially fulfills it due to the use of an alternative diagram format, while Gemini does not provide a graphical representation at all, which explains its significantly lower score. This analysis highlights that the key difference between the models is not their ability to understand the task, but their ability to fully comply with all formal requirements. In particular, the correct representation and visualization of knowledge structures remains a critical challenge for some LLMs.

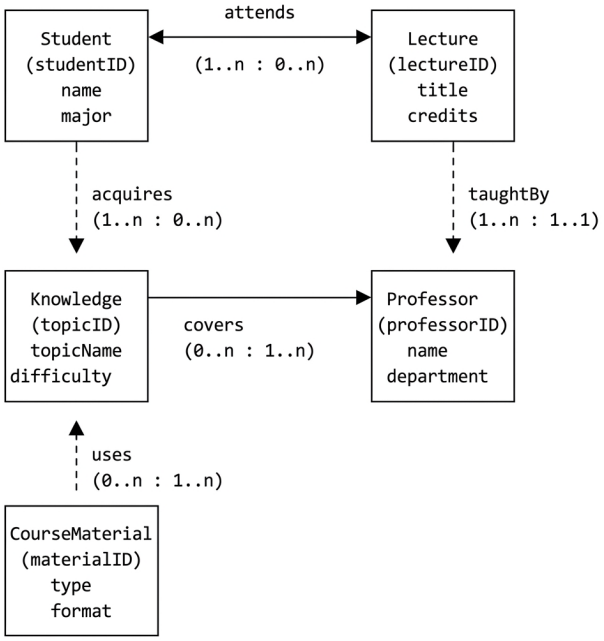
### 4.3.1 Consistency Analysis of LLMs answers Across Multiple Interactions

To further investigate the robustness and stability of LLM responses, Question 1a was submitted three times for each model in separate chat sessions using different input methods: copying the prompt, manually writing the prompt, and providing the prompt as an image. The corresponding responses are illustrated in the following figures.

**(a)** Kimi‘s response to the copied question




**(b)** Kimi‘s response to the manually written question



**(c)** Kimi's response to the photographed question

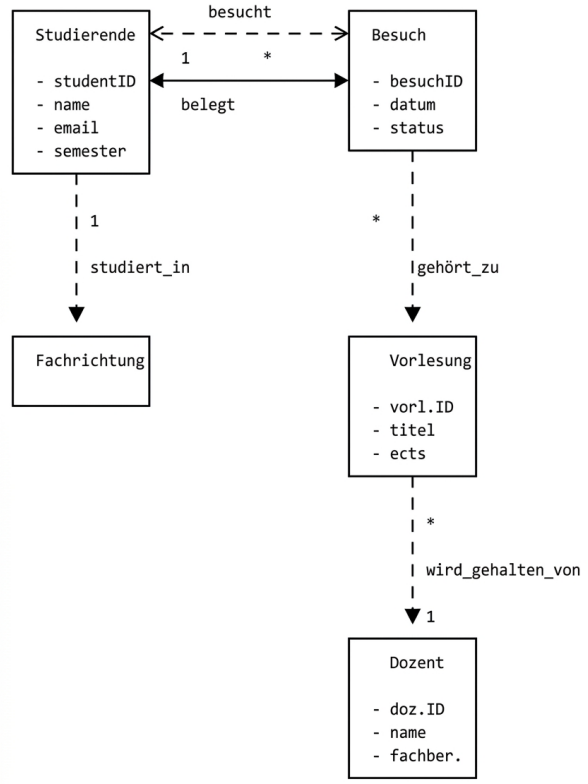
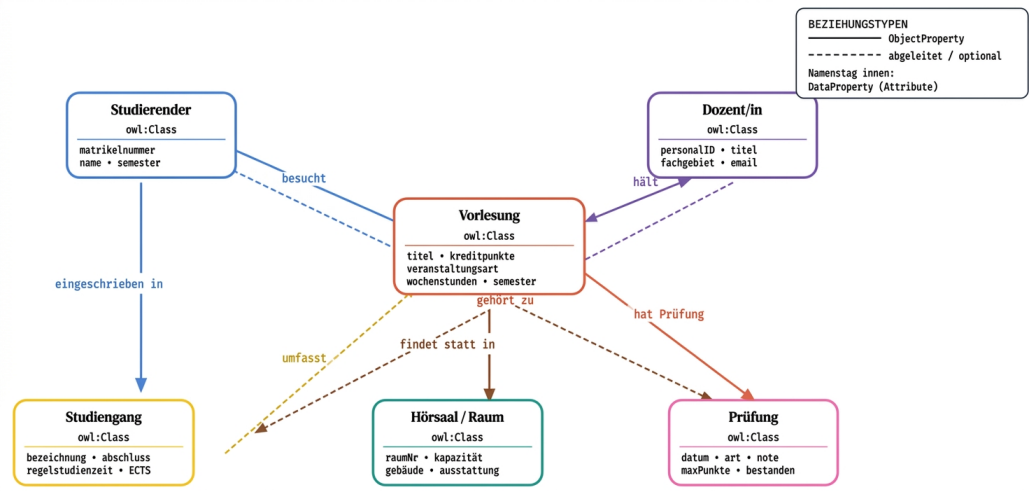



Figure 4.4: Kimi's responses to Question 1a across the three input modes

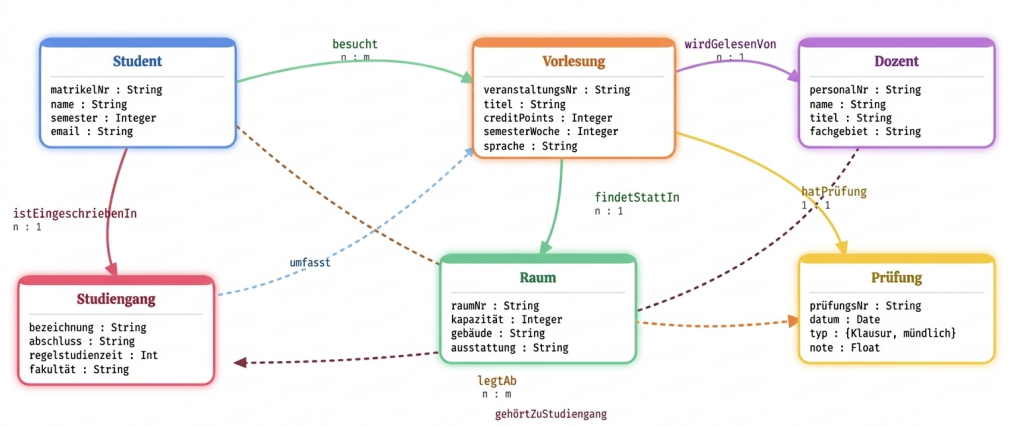
The three responses generated by Kimi show a very high level of consistency. In all cases, the model produces a complete ontology that includes more than four classes, detailed attributes, and well-defined relationships. The graphical representation is also consistently provided.

The main differences between the responses are limited to minor variations in naming conventions and formatting. For example, the same concepts, such as Student, Lecture, and Room, are always present, but may appear under slightly different labels or with additional attributes. In some cases, the model includes extra elements or alternative graphical representations, but these additions do not significantly change the overall structure. One noticeable variation occurs in the second Kimi response: although the question was provided in German, the generated diagram uses English labels. This does not change the correctness of the ontology, but it shows that the model does not always preserve the language of the original prompt consistently across all output components.

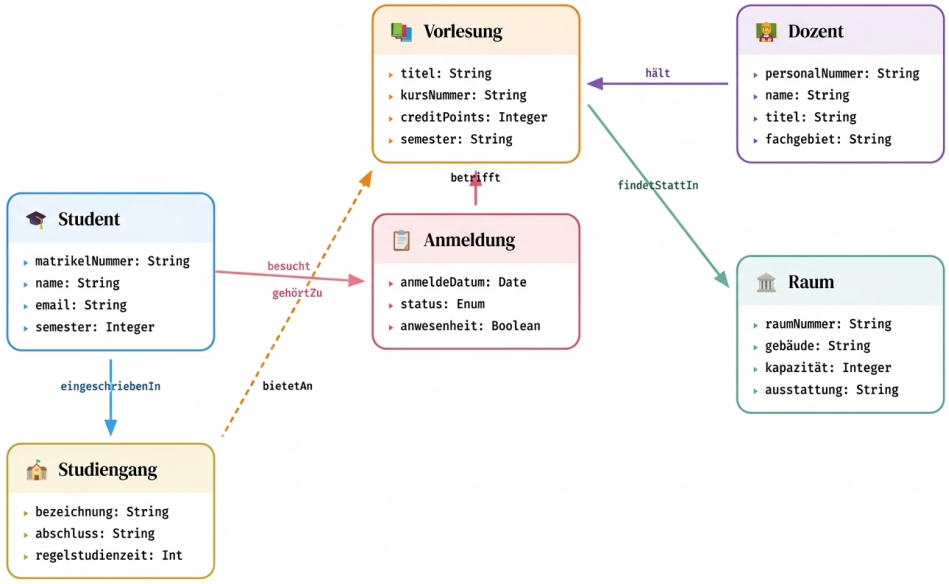
This indicates that Kimi relies on stable internal representations and produces reproducible outputs, even when the input method changes. The correctness of the answer remains unchanged across all interactions.




**(a)** Claude's response to the copied question




**(b)** Claude's response to the manually written question




**(c)** Claude's response to the photographed question

Figure 4.5: Claude's responses to Question 1a across the three input modes

Claude also demonstrates a high degree of consistency across the three interactions. In all responses, the model generates a structured ontology with clearly defined classes, attributes, and relationships. The answers are detailed and logically coherent.
However, a consistent pattern can be observed in the type of graphical representation used. Claude repeatedly produces a UML-style diagram instead of a standard ontology representation. While this still reflects a correct understanding of the relationships, it deviates from the expected format defined in the task.
The differences between the responses are again minor and mainly concern the level of detail and formatting. The overall structure and correctness remain stable, confirming that Claude produces reliable and reproducible outputs.

**Gemini's response to Question 1a**

```
graph TD
    %% Klassen
    S[Studierende]
    D[Dozierende]
    V[Vorlesung]
    H[Hörsaal]

    %% Beziehungen
    S -- "besuchen (n:m)" --> V
    D -- "halten (1:n)" --> V
    V -- "findet statt in (n:1)" --> H

    %% Eigenschaften als Notiz
    style S fill:#f9f,stroke:#333
    style D fill:#f9f,stroke:#333
    style V fill:#bbf,stroke:#333
    style H fill:#dfd,stroke:#333
```

Figure 4.6: Gemini's response to the copied question

**Gemini's response to Question 1a**

```
graph TD
    %% Klassen
    S[Studierender]
    V[Vorlesung]
    D[Dozent]
    R[Raum]

    %% Beziehungen
    S -- "schreibt sich ein in / besucht" --> V
```

```
    D -- "hält / verantwortet" --> V
    V -- "findet statt in" --> R

    %% Attribute (Beispielhaft)
    S -.-> s1(Matrikelnummer)
    V -.-> v1(Modultitel)
    D -.-> d1(Lehrstuhl)
    R -.-> r1(Kapazität)

    style S fill:#f9f,stroke:#333,stroke-width:2px
    style V fill:#bbf,stroke:#333,stroke-width:2px
    style D fill:#dfd,stroke:#333,stroke-width:2px
```

Figure 4.7: Gemini's response to the manually written question

**Gemini's response to Question 1a**

```
graph TD
    %% Klassen
    S[Studierender]
    V[Vorlesung]
    D[Dozent]
    H[Hörsaal]

    %% Beziehungen
    S -- besucht --> V
    D -- hält --> V
    V -- findet statt in --> H

    %% Beispielhafte Attribute (optional zur Veranschaulichung)
    subgraph Attribute
        S -.- attr1(Matrikelnr)
        V -.- attr2(Titel)
        D -.- attr3(Name)
        H -.- attr4(Raumnr)
    end

    style S fill:#f9f,stroke:#333,stroke-width:2px
    style V fill:#bbf,stroke:#333,stroke-width:2px
    style D fill:#dfd,stroke:#333,stroke-width:2px
    style H fill:#fdd,stroke:#333,stroke-width:2px
```

Figure 4.8: Gemini's response to the photographed question

The responses generated by Gemini also show a consistent structure across all interactions. The model correctly identifies the main classes, attributes, and relationships required for the ontology.
However, a key limitation remains unchanged across all runs: Gemini does not provide a graphical representation of the ontology. This missing component significantly affects the completeness of the answer and leads to a lower evaluation score.
Similar to the other models, the differences between the responses are limited to small variations in wording and explanation. The overall structure and reasoning remain stable, indicating that Gemini also produces consistent outputs.
Across all three models, the results clearly show that repeating the same question in different chats does not lead to fundamentally different answers. The models consistently reproduce the same conceptual structure, with only minor variations in terminology and presentation.
This confirms that LLM outputs are highly stable and rely on learned patterns rather than generating diverse alternative solutions. Therefore, repeating the same experiment for additional questions would not provide significant new insights. Based on the detailed analysis of Question 1a, a global comparison of the LLMs can be established by considering correctness, completeness, consistency, and response characteristics.

Table 4.4: Global Comparison of LLM Performance

| **Criteria** | **Kimi** | **Claude** | **Gemini** |
|---|---|---|---|
| Correctness | High (100%) | Medium (66.66%) | Moderate (50%) |
| Completeness | Very High | High | Medium |
| Graphical Representation | Correct | Alternative (UML) | Missing |
| Consistency | Very High | Very High | Very High |
| Level of Detail | Very High | High | Medium |

The comparison highlights that all LLMs demonstrate strong capabilities in understanding and solving ontology-related tasks. However, clear differences emerge when evaluating how well they satisfy all requirements of the question.
Kimi achieves the best overall performance. It consistently provides complete and correct answers, including all required components and a proper graphical representation. The model also shows very high stability across multiple interactions, making it the most reliable system in this experiment.
Claude performs well in terms of reasoning and structure. Its responses are detailed and logically consistent, but the use of UML diagrams instead of ontology-specific representations leads to a partial mismatch with the expected format. This results in a moderate reduction in correctness.
Gemini demonstrates a good understanding of the conceptual aspects of the task, such as identifying classes and relationships.However, the absence of a graphical representation in all responses significantly reduces its overall performance. This limitation remains consistent across all interactions.
Another important observation is that all models tend to generate very detailed answers, often including additional explanations and elements that are not required. While this reflects strong generative capabilities, it may reduce clarity in an exam setting, where concise and focused answers are preferred.

In summary, all evaluated LLMs show high consistency and strong conceptual understanding. However, differences in completeness, formatting, and adherence to formal requirements highlight the limitations of relying solely on LLMs in academic contexts. These results emphasize the importance of human evaluation, especially when precision and structure are critical.

### 4.3.2 Overall evaluation of the LLMs' performance

After the detailed analysis of Question 1a, this section presents a broader evaluation of the performance of the selected LLMs across the remaining exam questions (Question 1b to Question 9). The goal is to assess how consistently each model is able to generate correct and complete answers across different types of database-related tasks, including conceptual questions, SQL queries, ontology modeling, and reasoning-based problems.

For each question, the answers generated by the LLMs were compared to the reference solutions provided by the professor. Based on the predefined grading scheme introduced in the methodology, points were assigned to each response depending on correctness and completeness. The results were then converted into correctness percentages in order to allow a fair comparison between different questions and models.

Table 4.5: Overall benchmarking results for Questions 1a–9.

| **Question** | **Prompt in German** | **Claude** | **Gemini** | **Kimi** | **Max** |
|---|---|---|---|---|---|
| Q1a | Erstellen Sie eine Ontologie, welches den Besuch von Vorlesungen durch Studierende darstellt. Wählen Sie geeignete Klassen (mind. 4), charakterisieren Sie diese durch Eigenschaften und stellen Sie Beziehungen zwischen den Klassen her! Stellen Sie die Ontologie grafisch dar! | 4 | 3 | 6 | 6 |
| Q1b | Überführen Sie folgende Ontologie in eine passende Darstellung in RDF, OWL oder Turtle! | 5 | 5 | 4.5 | 5 |
| Q1c | Stellen Sie obigen Klassenbeziehungen mittels RDF-S dar! | 3 | 3 | 3 | 3 |
| Q2 | Kennzeichnen Sie die erfüllten und nicht erfüllten Eigenschaften der Modellierungsstile. | 4 | 4 | 0 | 4 |
| Q3a | Setzen Sie die Beschreibung in ein Standard-ER-Modell um. | 2 | 2.5 | 1 | 4 |
| Q3b | Überführen Sie das ER-Modell in das relationale Modell. | 2.5 | 1.5 | 2 | 4 |
| Q3c | Geben Sie die Ergebnisrelationen der Anfragen an. | 5 | 1.5 | 0 | 6 |

| Question | Prompt in German | Claude | Gemini | Kimi | Max |
|---|---|---|---|---|---|
| Q4a | Überprüfen Sie die Gültigkeit des XML-Dokuments bezüglich der DTD. | 5 | 4 | 2 | 5 |
| Q4b | Bestimmen Sie die Termmenge nach Stoppwortliste und Stammwortreduktion. | 4 | 5.5 | 4 | 6 |
| Q4c | Umranden Sie zwei disjunkte, unvollständige und konkave Cluster. | 3 | 0 | 1 | 3 |
| Q5a | Bestimmen Sie die Wahrheitswerte der gegebenen Formel. | 0 | 0 | 3 | 4 |
| Q5b | Zeigen Sie, ob die Formel eine Tautologie ist. | 3 | 0 | 3 | 3 |
| Q5c | Beweisen Sie die Äquivalenz mittels Äquivalenzumformung. | 3 | 0 | 3 | 3 |
| Q5d | Stellen Sie die Aussagen in der Prädikatenlogik dar. | 4 | 0 | 4 | 4 |
| Q6a | Verwenden Sie den Satz von Bayes zur Berechnung. | 7 | 7 | 7 | 7 |
| Q6b | Bestimmen Sie alle häufigen Itemsets mit support-Wert. | 4 | 3 | 3 | 4 |
| Q6c | Welche allgemeinen Assoziationsregeln lassen sich ableiten? | 3 | 0 | 3 | 3 |
| Q7a | Schreiben Sie eine SQL-Anfrage für den Gesamtwert. | 2 | 2 | 2 | 2 |
| Q7b | Schreiben Sie eine SQL-Anfrage mit ROLLUP. | 6 | 6 | 4 | 6 |
| Q7c | Schreiben Sie eine SQL-Anfrage mit CUBE. | 6 | 7 | 4 | 7 |
| Q8 | Wählen Sie Kompressionsmethoden und zeigen Sie die komprimierte Darstellung. | 9 | 3 | 2.5 | 10 |
| Q9 | Transformieren Sie Snapshot $i$ in Snapshot $i + 1$. | 3 | 3 | 3 | 3 |
| **Total** | | **86.5** | **60** | **64** | **101** |
| **Correctness** | | **85.6 %** | **59.4 %** | **63.4 %** | |

Table 4.5 summarizes the performance of Claude, Gemini, and Kimi on Questions 1a to 9. Across all questions, Claude achieved the highest overall result with 86.5 out of 101 points, corresponding to 85.6 % correctness. Gemini obtained 60 out of 101 points (59.4 %), while Kimi achieved 64 out of 101 points (63.4 %). The gap between Claude and the other two models is therefore substantial and suggests that Claude was the most reliable model in this benchmark, especially when the questions required precise formal outputs rather than only general conceptual explanations

A first important pattern is that the three models do not fail in the same way. Claude usually attempts the required task and often produces a complete answer, even when small formal mistakes remain. This can be seen, for example, in the RDF/Turtle and RDF-S tasks in Questions 1b and 1c, where Claude achieved full points, and in the SQL and compression tasks later in the exam, where it also remained strong . Overall Gemini shows a different pattern. It performs very well when the task matches a familiar and highly structured problem type, such as Bayes calculation, Data Warehouse SQL syntax, or vocabulary reduction, but it loses many points when the answer must include several explicit sub-parts or a constructed formal output. Kimi, in contrast, often shows good reasoning ability in logic and formula-based questions, but it is more likely to drift away from the exact requirement of the question and answer something related rather than exactly what was asked

The RDF/Turtle and RDF-S tasks in Questions 1b and 1c A were handled very well by all three models. Claude and Gemini received full points, and Kimi lost only 0.5 points because of a small simplification in the modeling of *betreutLehrveranstaltung*. This suggests that all three models are strong when the task is based on a familiar symbolic representation with clear syntax rules and a limited semantic space. At the same time, the answers were often too long, which is important in an exam context: even when the content is correct, verbosity may reduce efficiency and increase the risk of unnecessary mistakes.

Question 2 A shows one of the clearest differences between the models. Claude and Gemini received full points, whereas Kimi received 0 points because it answered a different topic, namely MOLAP/ROLAP/HOLAP, instead of filling the required table about multidimensional modeling styles. A reasonable explanation is that Kimi recognized the broader topic of data warehousing but did not remain anchored to the exact task instruction. In other words, it appears to have matched the question to a related concept from the same domain rather than solving the concrete table-filling task. This is a meaningful finding for the thesis, because it shows that some LLMs are not only sensitive to wording, but may also substitute the requested task with a semantically related one when the domain triggers a familiar pattern.

Question 3 A is especially informative because it combines three different formal tasks: ER modeling, transformation into the relational model, and exact SQL result relations. Claude performed best overall with 9.5 out of 14 points, Gemini achieved 5.5 out of 14, and Kimi only 3 out of 14. The results suggest that all three models struggle when the task requires strict formal notation and exact output rather than explanatory text. In Question 3a, Claude and Kimi mainly described the ER model verbally instead of producing a proper ER diagram, which shows that LLMs may prefer linguistic explanation over formal graphical modeling. In Question 3b, Gemini lost many points because of key and relationship errors, especially around primary keys, foreign keys, and merge conditions. This indicates difficulty with schema-level precision. In Question 3c, Gemini and Kimi performed particularly poorly because the task required exact result relations for SQL queries, and both models failed either by computing wrong outputs or by not recognizing the table structure correctly. A reasonable explanation is that these questions depend on step-by-step symbolic correctness. If one intermediate interpretation is wrong, the final output is also wrong. Claude appears more robust here because it continues to compute and structure the answer even when uncertainty exists, while Gemini and Kimi are more brittle.

Question 4 A again reveals differentiated strengths. Claude achieved 12 out of 14 points, Gemini 9.5 out of 14, and Kimi 7 out of 14. Claude performed very well in XML/DTD validation and clustering, showing balanced performance across formal checking and semi-visual reasoning. Gemini was especially

strong in vocabulary reduction after stopword filtering and stemming, which is plausible because this kind of text processing aligns closely with the language-oriented strengths of LLMs. However, Gemini received 0 points in Question 4c because no clustering answer was given. A reasonable interpretation is that Gemini handled the text-based part of the task well, but failed when the question required an explicit visual construction. This matches a broader pattern in the benchmark: Gemini often succeeds on language-heavy or syntax-pattern tasks, but loses points when the answer must include a directly constructed visual or schematic output. Kimi, on the other hand, seems weaker in XML/DTD logic because this task requires careful attention to explicit constraints, ordering, and invalid attributes; its answers suggest more guessing than exact verification.

Question 5 A is one of the most revealing blocks in the entire benchmark. Claude scored 10 out of 14, Gemini 0 out of 14, and Kimi 13 out of 14. This is strong evidence that the models have very different strengths in formal logic. Kimi clearly performs best on logic proofs, equivalence transformations, and predicate-logic formalization. Claude is also strong, but it failed in Question 5a because it appears to have misread or misparsed the logical formula and therefore did not provide the required truth table. Gemini's complete collapse in this question block is especially important. It did not provide the truth table, did not complete the proofs, and gave incomplete predicate logic formalization. A reasonable explanation is that Gemini had difficulty handling the symbolic formatting of the task and the multi-part formal reasoning structure. Another plausible explanation is that once the first sub-task was misinterpreted, the later sub-questions were not followed through completely. This supports a key thesis claim: some LLMs may appear strong in general academic tasks but still fail badly when the question requires strict symbolic reasoning and complete multi-step formal output.

Question 6 A shows a more balanced picture. Claude achieved 13 out of 13, Gemini 10 out of 13, and Kimi 12 out of 13. All three models solved the Bayes sub-question correctly, which is not surprising because the task follows a well-known computational template. This suggests that formula-driven problems with a familiar structure are handled very well by all evaluated models. The differences arise in the enumeration tasks. Claude was best at listing frequent itemsets and association rules completely. Gemini lost points in Question 6c because it did not provide the association rules at all, while Kimi made small counting or omission errors. This again highlights Gemini's weakness in answer completeness and Kimi's tendency toward partial or approximate enumeration.

Question 7 A is the strongest area for Gemini. It achieved the full 15 out of 15 points, compared with 14 out of 15 for Claude and 10 out of 15 for Kimi. This suggests that Gemini is especially good at data-warehouse SQL patterns such as *ROLLUP* and *CUBE*. These tasks appear to match training patterns that the model can reproduce accurately. Claude also performed very well, losing only one point because it used *ROLLUP* instead of *CUBE* in one sub-question. Kimi, however, lost several points because its SQL statements did not match the expected output structure closely enough. A reasonable explanation is that Gemini and Claude both follow known SQL templates effectively, while Kimi is more likely to generate a plausible query that is semantically related but not fully aligned with the exact result table shown in the exam.

Question 8 A, the column-store compression task, produces another sharp distinction. Claude scored 9 out of 10, Gemini 3 out of 10, and Kimi 2.5 out of 10. This task is particularly useful because it does not only ask for naming a method; it also asks for the actual compressed representation. Claude is the only model that consistently provides both the chosen method and a concrete encoded output,

which explains its much better score. Gemini and Kimi remain too often at the explanation level or provide incomplete representations. A reasonable explanation is that this question requires "show your work" behavior. The model must not only recognize a suitable compression strategy, but also execute it concretely on the data. Claude appears much better at this transition from explanation to formal construction.

Question 9 A is the most uniform result in the benchmark: all three models achieved full points. This suggests that differential snapshot transformation is a highly structured task with clear add, delete, and update operations, and that all models can follow this pattern successfully when the data and expected action types are explicit.

From a broader perspective, the benchmark reveals that each model has a characteristic competence profile. Claude is the strongest overall model because it combines good task understanding with comparatively high completeness and formal precision across very different question types. Gemini performs particularly well on pattern-based tasks, especially SQL and some language-processing tasks, but its main weakness is incompleteness: when a question contains several sub-parts or requires explicit constructed outputs, entire components may be omitted. Kimi is strongest in formal logic and does reasonably well on some symbolic tasks, but it is more vulnerable to task drift: in several cases, it answers something related and reasonable rather than the exact required task.

These differences are highly relevant for the broader goals of the thesis. They show that "LLM performance" should not be treated as a single global capability. Instead, performance depends strongly on the type of task, the degree of formal precision required, the need for completeness across sub-questions, and the extent to which the model must construct an explicit output rather than explain a concept. This also explains why a model may answer one question perfectly and fail badly on another one from the same exam: the underlying cognitive demand is different, even if the domain remains the same.

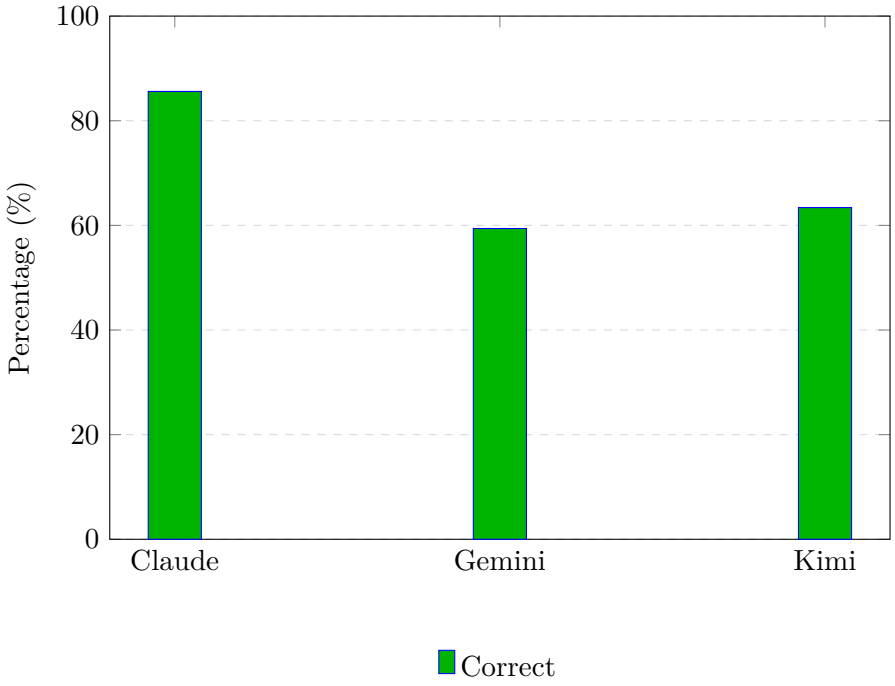



Figure 4.9: Overall correctness comparison of LLMs across all exam questions

Figure 4.9 illustrates the overall correctness distribution of the evaluated LLMs across all exam questions. Claude clearly outperforms the other models, achieving nearly 86 % correctness, while Gemini remains around 59 % and Kimi around 63 %. The gap between Claude and the other two models remains substantial and highlights a significant difference in reliability.

While Gemini and Kimi show closer overall performance, their error patterns differ: Gemini mainly loses points due to missing answers, whereas Kimi loses points due to incorrect interpretations of the tasks. Claude, on the other hand, shows a balanced and robust performance with relatively few incorrect responses.

Figure 4.9 confirms the overall conclusion of the benchmarking phase: Although all models are capable of solving certain types of exam questions, only Claude provides consistently reliable results across different domains and task structures.

The best-performing LLM is not necessarily the one that sounds most confident or provides the longest answer. In this benchmark, high-quality performance is associated with four factors: Following the exact task instruction, completing all sub-parts, producing the requested formal representation, and avoiding unnecessary elaboration. These results support the argument that the evaluation of LLMs in exam settings must go beyond surface fluency and focus on structured correctness, completeness, and alignment with the required answer format.

## 4.4 Performance of LLMs in different languages

After evaluating the LLMs on the original German exam questions, an additional benchmark was conducted to investigate whether the language of the prompt influences the quality of the generated answers. In this multilingual experiment, three prompt languages were used: German, English, and French. The benchmark was deliberately restricted to two selected questions, namely Question 3 and Question 9.

These two questions were chosen for complementary reasons. Question 3 was selected because it had already shown substantial differences between the models in the German benchmark. It combines three formal sub-tasks, namely ER modeling, relational transformation, and exact SQL result relations, and therefore makes it possible to observe whether language affects the handling of complex and highly structured tasks. Question 9 was selected for the opposite reason. In the German benchmark, all three LLMs answered this question correctly and received full points. It therefore serves as a stable reference task and makes it possible to test whether correct performance remains stable when the same task is presented in another language.

The multilingual experiment was carried out by asking the same two questions in German, English, and French. The German scores were taken from the previous benchmark, since they had already been evaluated using the professor's correction style. The English and French answers were graded using the same grading logic, criteria, and point distribution as in the German version. This ensures that the comparison remains methodologically consistent and that any differences in performance can be interpreted mainly as language-related effects rather than as changes in evaluation standards.

**Question 3 – German**

**Prompt**

*(a) Setzen Sie die folgenden Beschreibungen in jeweils einem Standard-ER-Modell um. Nutzen Sie die Notation der Übung und wenden Sie die Intervallnotation für die Kardinalitäten an. Kritiker können Rezensionen schreiben. Kritiker besitzen einen Namen und eine eindeutige ID, während Rezensionen teilweise durch das betroffene Produkt charakterisiert werden. Rezensionen erhalten eine Menge von Fotos, eine Bewertung und eine Beschreibung.*
*(b) Überführen Sie die nachfolgenden ER-Modelle in das relationale Modell. Kennzeichnen Sie Primär- und Fremdschlüssel. Verschmelzen Sie, falls möglich.*
*(c) Geben Sie die Ergebnisrelationen der gegebenen Anfragen als Relationen an.*

**Claude's Results:**

- Partial ER model
- Mostly correct relational transformation
- Correct SQL result relations

**Gemini's Results:**

- Incomplete ER model
- Weak relational mapping
- Several incorrect SQL result relations

**Kimi's Results:**

- Weak formal modeling
- Incomplete relational transformation
- Failed SQL result output

**Question 3 – English**

**Prompt**

*(a) Convert the following description into a standard ER model. Use the notation from the exercise and use interval notation for the cardinalities. Critics can write reviews. Critics have a name and a unique ID, while reviews are partly characterized by the product being reviewed. Reviews have a set of photos, a rating, and a description.*
*(b) Convert the following ER models into the relational model. Mark primary and foreign keys. Merge relations if possible.*
*(c) Write the result relations of the given queries as relations.*

**Claude's Results:**

- Strong answer across all three sub-parts
- The English answer remains very close to the German version in structure and correctness

**Gemini's Results:**

- The model provides a partial ER answer and a relational transformation
- It does not fully deliver the SQL result section

**Kimi's Results:**

- The model gives a much stronger answer than in German
- It provides a usable ER description
- More structured relational conversion
- A substantially better SQL answer

**Question 3 – French**

**Prompt**

*(a) Transformez la description suivante en un modèle ER standard. Utilisez la notation du cours et la notation par intervalles pour les cardinalités. Les critiques peuvent écrire des avis. Les critiques ont un nom et un ID unique, et les avis sont en partie caractérisés par le produit concerné. Les avis contiennent un ensemble de photos, une note et une description.*
*(b) Transformez les modèles ER suivants en modèle relationnel. Indiquez les clés primaires et les clés étrangères. Fusionnez si possible.*
*(c) Donnez les relations résultat des requêtes données, sous forme de relations.*

**Claude's Results:**

- Again, a stable and strong answer
- Very close to the German and English versions

**Gemini's Results:**

- The answer is still incomplete
- It is slightly better than in English because at least a partial SQL result section is present

**Kimi's Results:**

- The response drifts between the sub-questions
- Some useful content is given
- Several parts do not match the exact task required

**Question 9 – German**

**Prompt**

*Gegeben sind zwei Datenbanksnapshots $i$ und $i+1$. Bestimmen Sie die minimale Menge an Operationen (ADD, DELETE, UPDATE), die notwendig ist, um Snapshot $i$ in Snapshot $i+1$ zu überführen.*

**Claude's Results:**

- Correctly identifies all required operations
- Detects the deletion, update, and insertion events
- Produces a complete and accurate differential snapshot

**Gemini's Results:**

- Correct identification of all changes between the snapshots
- Provides the required sequence of operations
- Concise but accurate solution

**Kimi's Results:**

- Correctly detects deleted, updated, and inserted tuples
- Provides additional explanations and comparison tables
- More verbose than the other models but still correct

**Question 9 – English**

**Prompt**

*Given two database snapshots $i$ and $i+1$, determine the minimum set of operations (ADD, DELETE, UPDATE) required to transform snapshot $i$ into snapshot $i+1$.*

**Claude's Results:**

- Nearly identical to the German answer
- Correctly identifies all modifications
- Produces a complete and accurate operation list

**Gemini's Results:**

- Stable performance across languages
- Correctly identifies all snapshot differences
- Generates the expected sequence of operations

**Kimi's Results:**

- Provides a detailed comparison table
- Correctly classifies all changes as DELETE, UPDATE, or ADD
- The answer is more structured than in the German version

**Question 9 – French**

**Prompt**

*Étant donnés deux snapshots de base de données $i$ et $i+1$, déterminez l'ensemble minimal d'opérations (ADD, DELETE, UPDATE) nécessaires pour transformer le snapshot $i$ en snapshot $i+1$.*

**Claude's Results:**

- Maintains the same high level of correctness
- Correctly identifies every required operation
- Very similar to the German and English versions

**Gemini's Results:**

- Produces a correct and concise answer
- Correctly identifies insertion, deletion, and update operations
- No significant difference compared with the English version

**Kimi's Results:**

- Gives the most detailed explanation among the three languages
- Correctly derives the differential snapshot operations
- Shows good robustness for this procedural database task

Table 4.6: Multilingual benchmarking results for Questions 3 and 9

| Question | Language | Claude | Gemini | Kimi | Max | Main observation |
|---|---|---|---|---|---|---|
| Q3 | German | 9.5 | 5.5 | 3 | 14 | Strong differences between the models Kimi is weakest in the German version. |
| Q3 | English | 9.5 | 4 | 10 | 14 | Kimi improves strongly Gemini remains incomplete. |
| Q3 | French | 9.5 | 6 | 2.5 | 14 | Claude remains stable Gemini improves slightly Kimi drifts away from the task. |
| Q9 | German | 3 | 3 | 3 | 3 | All three models are fully correct. |
| Q9 | English | 3 | 3 | 2 | 3 | Kimi loses one point because of the wrong update tuple. |
| Q9 | French | 3 | 3 | 3 | 3 | All three models are fully correct again. |
| **Total** | **German** | **12.5** | **8.5** | **6** | **17** | German remains difficult, especially for Kimi on Q3. |
| **Total** | **English** | **12.5** | **7** | **12** | **17** | English clearly helps Kimi, but not Gemini. |
| **Total** | **French** | **12.5** | **9** | **5.5** | **17** | French is stable for Claude, moderate for Gemini, and weak for Kimi. |
| **Correctness** | **German** | **73.5 %** | **50.0 %** | **35.3 %** | – | – |
| **Correctness** | **English** | **73.5 %** | **41.2 %** | **70.6 %** | – | – |
| **Correctness** | **French** | **73.5 %** | **52.9 %** | **32.4 %** | – | – |

Table 4.6 shows that the effect of prompt language is not uniform across the evaluated LLMs. Claude is the most stable model in this multilingual experiment. Its score remains constant across German, English, and French, with 12.5 out of 17 points in each case. This indicates that Claude's handling of both the difficult formal task (Question 3) and the more structured operational task (Question 9) is largely independent of prompt language. In other words, once Claude understands the task, translating the prompt does not significantly change its performance. Gemini shows a more variable pattern. Its German total is 8.5 out of 17, its English total decreases to 7 out of 17, and its French total rises slightly to 9 out of 17. The differences are mainly caused by Question 3. In English, Gemini provides the ER and relational parts, but the SQL result section is not fully delivered. In French, the answer is still incomplete, but the model gives slightly more useful content than in English. This confirms an important observation from the previous benchmark: Gemini's main limitation is often not basic task understanding, but incomplete execution. It can start correctly, especially in familiar structured tasks, but may omit required sub-parts when the task becomes longer and more formal. Kimi is the most language-sensitive model in this experiment. Its German total is only 6 out of 17, its English total rises sharply to 12 out of 17, and its French total drops again to 5.5 out of 17. The strongest improvement appears in English Question 3, where Kimi gives a much better answer than in German and French. The English response includes a clearer ER interpretation, a more structured relational conversion, and an explicit SQL result section. In contrast, the French version drifts between the sub-questions, mixes content from different parts, and does not remain sufficiently anchored to the

exact formal task. This pattern suggests that Kimi is particularly sensitive to the linguistic framing of the prompt. When the formulation matches a representation style that the model handles well, the performance improves strongly. When the wording becomes less stable for the model, task drift becomes more likely. The comparison between Questions 3 A and 9 A is also important. Question 3 is clearly the more language-sensitive task. It requires several formal steps, precise modeling decisions, and complete output structures. Small differences in language therefore have a stronger impact on the final result. Question 9, in contrast, remains largely stable across all languages. Claude and Gemini solve it correctly in German, English, and French. Kimi also solves it correctly in German and French, but loses one point in English because it changes the update tuple from $(F, F)$ to $(F, E)$ in Questions 9 A. Even so, the overall pattern remains clear: Question 9 is much less affected by language than Question 3 because it is operational, explicit, and highly constrained. Overall, the multilingual benchmark shows two main findings. First, prompt language can influence LLM performance, especially for multi-part formal tasks such as Question 3. Second, this effect is model-dependent. Claude is the most robust model across languages, Gemini is moderately sensitive mainly because of missing sub-parts, and Kimi is the most sensitive to prompt language. These results support the broader argument of this thesis that LLM performance should not be treated as a fixed property. Instead, it is shaped by the interaction between task structure, prompt formulation, and the internal strengths and weaknesses of the specific model.

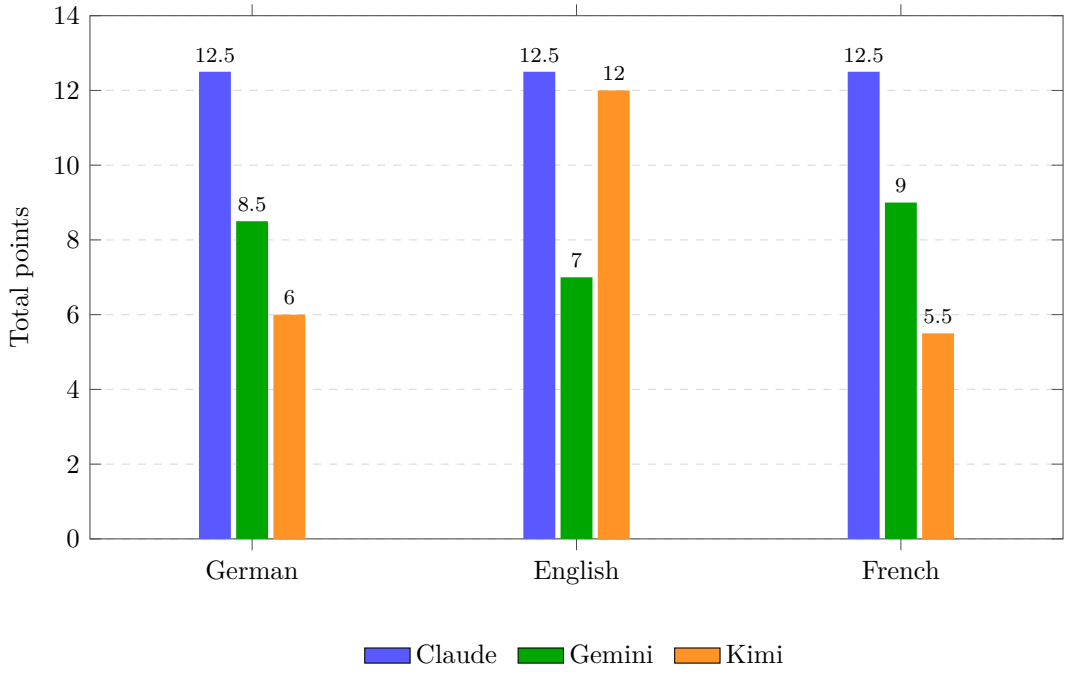



Figure 4.10: Comparison of multilingual benchmarking scores for Questions 3 and 9

Figure 4.10 summarizes the multilingual benchmark from a language-centered perspective. Each bar represents one language, and the stacked segments show how many points were obtained by Claude, Gemini, and Kimi within that language. This makes it possible to compare not only the overall amount of correct performance per language, but also the contribution of each model to that language-specific total. The figure highlights three important tendencies. First, Claude contributes almost the same number of points in all three languages. This again confirms that Claude is the most stable model in the multilingual benchmark. Second, the English bar is noticeably higher than the German and

French bars mainly because of Kimi. This shows that English clearly improves Kimi's performance in the selected questions, especially in Question 3. Third, Gemini remains in the middle range across all languages. Its performance does not collapse completely, but it never reaches the stability of Claude and it does not benefit from English in the same way as Kimi. In general, the diagram shows that the overall multilingual performance is not determined by language alone. Rather, the effect of language depends on how a specific model interacts with a specific task. In this experiment, English is the most favorable language overall because it leads to the highest total combined score, while French and German remain closer to each other. The multilingual benchmark confirms that prompt language can affect LLM performance, but the size and direction of this effect depend on both the model and the question type. For a highly structured operational task such as Question 9, the answers remain mostly stable across languages. For a complex multi-part formal task such as Question 3, the language effect is much more visible. Claude remains robust across German, English, and French; Gemini continues to struggle mainly with completeness, and Kimi shows the strongest language sensitivity. These results strengthen the central thesis argument that LLM performance in exam settings is not only a matter of general model capability, but also of prompt formulation, task structure, and language.

## 4.5 Discussion of the Results

The results presented in this chapter provide a comprehensive evaluation of the performance of the selected Large Language Models (Claude, Gemini, and Kimi) in solving database examination questions under different conditions, including variations in task type, prompt formulation, input modality, and language. Overall, the benchmarking clearly shows that while all three models are capable of generating correct answers in certain contexts, their performance is neither uniform nor consistently reliable. Claude achieves the highest overall correctness (85.6 %) and demonstrates the most stable and robust behavior across nearly all question types. This strong performance is not only reflected in the numerical score, but also in its ability to follow instructions precisely, complete multi-part tasks, and produce outputs that closely match the expected formal representations. In contrast, Gemini (59.4 %) and Kimi (63.4 %) exhibit more heterogeneous behavior. Gemini often generates partially correct answers but fails to include all required components, which leads to significant point losses in questions with multiple sub-parts. Kimi, on the other hand, frequently produces logically plausible but misaligned answers, indicating a tendency toward semantic approximation rather than strict task adherence.

A central finding of this chapter is that LLM performance is strongly dependent on the structure and cognitive demands of the task. Tasks that follow well-known patterns and require standard procedural knowledge—such as SQL queries, Bayes' theorem, or RDF transformations—are handled relatively well by all models. This suggests that LLMs are particularly effective when the solution space is constrained and closely aligned with patterns learned during training. However, performance drops significantly in tasks that require strict formal correctness, step-by-step reasoning, or the complete construction of structured outputs. This is especially visible in Question 3, where all models struggle to correctly produce ER models, relational schemas, and exact query results. These tasks require not only understanding the problem, but also maintaining consistency across multiple transformation

steps. The results indicate that even small intermediate errors can propagate and lead to incorrect final outputs, revealing a limitation in the models' ability to handle chained symbolic reasoning.

Another important observation concerns the difference between linguistic fluency and actual correctness. All evaluated models produce answers that are well-formulated, detailed, and often accompanied by additional explanations or diagrams. However, these responses frequently include information that is not required by the question. In an exam context, this verbosity can be problematic, as it increases cognitive load for the reader and may hide errors within otherwise convincing explanations. This finding highlights that high-quality language generation should not be confused with accurate problem-solving performance. In fact, the tendency of LLMs to produce confident and elaborate answers may increase the risk of overestimating their correctness.

The consistency analysis based on repeated executions of Question 1a further reveals that LLM outputs are highly stable across different interactions. The models reproduce similar structures, class selections, and relationship patterns, regardless of whether the prompt is provided as text or image. While this consistency can be interpreted as robustness in output generation, it also indicates limited variability and adaptability. Instead of exploring alternative valid solutions, the models tend to rely on fixed internal templates.

This behavior suggests that LLMs operate primarily through pattern completion rather than flexible reasoning, which may limit their usefulness in tasks that require creative or non-standard solutions.

The time comparison experiment adds a practical dimension to the evaluation. The results show that manual input of prompts requires significantly more time (between 126 and 186 seconds) compared to photo-based input (between 13.77 and 36.66 seconds), making image-based interaction more efficient from a user perspective. However, response times vary considerably between models. Gemini provides very fast responses (around 17 seconds for text input), whereas Claude requires more time (up to 90 seconds), likely due to more extensive internal reasoning and output generation. Kimi shows the highest response time in the image-based setting (over 190 seconds), which can be explained by its limitation of requiring an additional textual instruction alongside the image. These results demonstrate that efficiency is not only determined by the input method, but also by the internal processing characteristics of each model.

The multilingual experiment provides further insight into the robustness of LLMs. The results show that Claude maintains stable performance across German, English, and French, indicating strong generalization capabilities across languages. Gemini shows moderate variation, with performance differences mainly caused by incomplete answers rather than misunderstanding. In contrast, Kimi is highly sensitive to language changes, showing significant improvement in English and weaker performance in French and German. This suggests that the internal representations of some models are more strongly aligned with specific linguistic patterns. Furthermore, the impact of language is more pronounced in complex tasks such as Question 3, while simpler and highly structured tasks like Question 9 remain stable across all languages. This confirms that prompt sensitivity increases with task complexity and that linguistic variation can amplify existing weaknesses in reasoning and task execution.

Taken together, the results of this chapter demonstrate that LLM performance in exam scenarios is the result of a complex interaction between multiple factors, including the model architecture, the structure of the task, the requirement for formal precision, and the formulation of the prompt. The findings

show that no model is universally optimal across all types of questions. Instead, each model exhibits a specific competence profile: Claude is characterized by high reliability and completeness, Gemini by strong performance on pattern-based and syntax-driven tasks but weaknesses in completeness, and Kimi by good reasoning in logic-related tasks but a higher tendency toward task misinterpretation.
From a broader perspective, these findings have important implications for the use of LLMs in educational and assessment contexts. While LLMs can provide valuable support and achieve high performance in certain scenarios, their limitations in terms of completeness, formal correctness, and robustness must be carefully considered. In particular, the results emphasize that evaluation should not rely solely on surface-level correctness or fluency, but must also take into account whether the model fully satisfies the requirements of the task. These insights directly support the central objective of this thesis and provide a strong motivation for the following chapter, which investigates how systematic reformulation of exam questions can further reveal the strengths and limitations of LLMs.

# Chapter 5

# *Real-Life Student Experiment*

This chapter complements the controlled benchmarking experiment presented in Chapter 4 examining how students actually use Large Language Models in an exam-like situation. While Chapter 4 evaluated selected LLMs under controlled and identical conditions, the present chapter focuses on a more realistic educational setting. The main objective is to compare student performance without external assistance to student performance with LLM support, and to analyze how students interact with LLMs when they are free to choose the model and prompting strategy themselves.
The experiment is important because real-world LLM usage is different from controlled benchmarking. In practice, students may upload photos, write the question manually, reformulate the task, or provide additional context such as lecture material. Therefore, this chapter does not only evaluate correctness, but also investigates prompting behavior, time pressure, model choice, and the practical role of LLMs in exam preparation.

## 5.1 Methodology

The real-life experiment was conducted on 13 May 2026 with ten volunteer students from the Data Warehouse course. The experiment was anonymous and was not graded. The goal was to simulate an exam-preparation situation and observe how students solve database-related exam questions both independently and with LLM assistance.
The experiment consisted of two phases. In the first phase, students had 30 minutes to answer the questions by themselves without using any external help. They were not allowed to use LLMs, lecture slides, notes, books, websites, friends, or any other resources. This phase was used to measure the students' own knowledge and their ability to solve the selected questions under time pressure.
In the second phase, students had another 30 minutes to answer the same questions with the help of an LLM of their choice. Students were free to use any model, such as ChatGPT, Gemini, Claude, Copilot, or another tool. They were also allowed to choose their own interaction method. Some students uploaded photos of the questions, while others typed the questions manually or wrote their own prompts.
The experiment included five main questions taken from real past Data Warehouse examinations. The selected questions covered different types of knowledge and skills, including multidimensional modeling,

XML validation, SQL for data warehouse queries, column-store compression, and differential snapshot transformation. The maximum achievable score was therefore 37 points. The questions and their maximum points were as follows:

- Question 1: Properties of multidimensional modeling styles (4 points)
- Question 2: XML and DTD validation (5 points)
- Question 3a: SQL query for the total number of participants (2 points)
- Question 3b: SQL query using data warehouse extensions for city and country aggregation (6 points)
- Question 3c: SQL query using data warehouse extensions for distance and country aggregation (7 points)
- Question 4: Column-store compression strategies and compressed representation (10 points)
- Question 5: Differential snapshot transformation using add, delete, and update operations (3 points)

The selected questions were intentionally diverse. Question 1 tests conceptual knowledge about multidimensional modeling styles. Question 2 requires formal validation of an XML document against a DTD. Question 3 focuses on SQL and data warehouse extensions, including aggregation queries and result-table reconstruction. Question 4 tests whether students can select and apply suitable compression methods for different column types. Question 5 evaluates whether students can identify state changes between two database snapshots.

In addition to the answers, students were asked to document how they used the LLM. They were asked to write down the name of the LLM they used, whether they used the original exam question as a prompt, whether they wrote their own prompt, and whether the prompt was typed manually or provided through a photograph. At the end of the experiment, students also had the possibility to write comments and feedback about the experience.

This methodology makes it possible to analyze three aspects at the same time. First, it measures the difference between human-only performance and LLM-assisted performance. Second, it shows how students naturally interact with LLMs in a realistic setting. Third, it allows the results to be compared with the controlled benchmark from Chapter 4.

## 5.2 Results and Analysis

Table 5.1: Student performance without LLM assistance

| Student | Q1 | Q2 | Q3a | Q3b | Q3c | Q4 | Q5 | Total | Correctness |
|---|---|---|---|---|---|---|---|---|---|
| **Max** | 4 | 5 | 2 | 6 | 7 | 10 | 3 | 37 | 100 % |
| S1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.0 % |
| S2 | 1.5 | 0 | 0.5 | 0 | 0 | 4 | 3 | 9 | 24.3 % |
| S3 | 3.5 | 0 | 2 | 6 | 6 | 3 | 3 | 23.5 | 63.5 % |
| S4 | 2 | 1 | 2 | 0 | 0 | 0 | 0 | 5 | 13.5 % |
| S5 | 1.5 | 1 | 0 | 0 | 0 | 2.5 | 0 | 5 | 13.5 % |
| S6 | 3 | 1 | 2 | 5 | 6 | 0.5 | 0 | 17.5 | 47.3 % |
| S7 | 2 | 1 | 2 | 2 | 3 | 5 | 2.5 | 17.5 | 47.3 % |
| S8 | 3.5 | 1.5 | 1 | 4 | 5 | 7.5 | 3 | 25.5 | 68.9 % |
| S9 | 1.5 | 0 | 0 | 0 | 0 | 7 | 3 | 11.5 | 31.1 % |
| S10 | 3.5 | 2 | 2 | 2 | 3 | 0 | 1.5 | 14 | 37.8 % |
| **Average** | **2.0** | **0.75** | **1.15** | **1.9** | **2.3** | **2.95** | **1.6** | **12.85** | **34.7 %** |

Table 5.1 summarizes the performance of the ten students during the first phase of the experiment, where no external help was allowed. The human-only results show that the selected questions were challenging for the students under the given time limit. The average score was 12.85 out of 37 points, which corresponds to 34.7 % correctness. This relatively low score does not necessarily mean that the students did not understand the course content. Instead, the answers and feedback indicate that time pressure and the need for exact formal notation strongly influenced the results.

Question 1 was answered better than most other tasks. This is expected because it is a conceptual table-filling question about modeling styles for multidimensional data. Students could often answer this question based on general understanding, even when they were unsure about exact notation. In contrast, Question 2 was difficult for many students because XML/DTD validation requires careful checking of element order, missing mandatory elements, and invalid attributes.

The SQL block in Question 3 shows mixed results. Some students achieved high scores, especially in Question 3a, while others left the SQL questions unanswered. This confirms that SQL tasks are strongly dependent on syntax recall and familiarity with the expected query structure. Several students seemed to understand the goal of the query but were unable to write the exact SQL statement within the limited time.

Question 4, which required column-store compression methods and their application to the given data, also produced large differences between students. Some students achieved good scores, while others left the task empty or only provided the name of a compression method without applying it. This suggests that students may know the names of compression techniques but still struggle to apply them correctly to a concrete relation.

Question 5 was one of the more accessible tasks for several students. The task required identifying delete, update, and add operations between two snapshots. Since the expected answer is short and highly structured, students who understood the idea of snapshot comparison could solve it relatively quickly.

To better interpret these mock-exam results, they can also be compared with the available results from the real exam for the same type of questions. The comparison shows that the mock-exam performance was generally lower than the performance observed in the real exam. This difference is reasonable because the mock experiment was conducted under stricter time pressure: students had only 30 minutes to solve all selected questions without assistance, while a real exam usually provides more time and students are better prepared for the exact exam situation. Nevertheless, the same general tendency can be observed in both settings. Questions that required precise syntax, formal notation, or exact application of a method remained more difficult than short and highly structured tasks. This supports the interpretation that the low mock-exam scores are not only caused by missing knowledge, but also by time pressure, stress, and the difficulty of recalling exact formal structures during the exam situation. Therefore, the mock experiment can still be considered meaningful, because it reproduces several patterns also visible in real exam results, even if the absolute scores are lower.

Table 5.3 summarizes the LLM-assisted phase. The table includes the model used, the interaction type, whether the original question was used as prompt, and the final LLM-assisted score. Some entries are based on the written notes collected during the experiment and may be slightly corrected in the final version if needed.

Table 5.2: LLM and prompts used

| Student | LLM used | Input type | Prompt strategy | Extra context |
|---|---|---|---|---|
| S1 | ChatGPT | Photo | Original question | No |
| S2 | ChatGPT | Photo | Original question | No |
| S3 | ChatGPT | Photo | Original question | No |
| S4 | ChatGPT | Photo | Original question | No |
| S5 | ChatGPT | Photo | Own prompt / partial reformulation | No |
| S6 | ChatGPT | Photo | Original question | No |
| S7 | Gemini | Photo | Own prompt / selected information | No |
| S8 | ChatGPT | Photo | Original question | Lecture script uploaded |
| S9 | ChatGPT | Photo | Original question | No |
| S10 | ChatGPT | Photo / text | Own prompt | No |

Table 5.3: Student performance with LLM assistance

| Student | Q1 | Q2 | Q3a | Q3b | Q3c | Q4 | Q5 | Total | Correctness |
|---|---|---|---|---|---|---|---|---|---|
| **Max** | 4 | 5 | 2 | 6 | 7 | 10 | 3 | 37 | 100 % |
| S1 | 4 | 2 | 2 | 2 | 7 | 10 | 3 | 30 | 81.1 % |
| S2 | 3 | 4 | 2 | 6 | 1 | 0 | 0 | 16 | 43.2 % |
| S3 | 1.5 | 2 | 0 | 0 | 0 | 1.5 | 0 | 5 | 13.5 % |
| S4 | 1.5 | 2 | 2 | 6 | 7 | 7.5 | 3 | 29 | 78.4 % |
| S5 | 2.5 | 5 | 2 | 7 | 7 | 3 | 3 | 29.5 | 79.7 % |
| S6 | 1.5 | 2 | 2 | 6 | 4 | 7.5 | 3 | 26 | 70.3 % |
| S7 | 3 | 5 | 2 | 6 | 7 | 5 | 0 | 28 | 75.7 % |
| S8 | 3 | 1 | 2 | 6 | 6 | 7 | 3 | 28 | 75.7 % |
| S9 | 2 | 0 | 2 | 4 | 4 | 0.5 | 0 | 12.5 | 33.8 % |
| S10 | 2 | 2 | 2 | 6 | 7 | 10 | 3 | 32 | 86.5 % |
| **Average** | **2.40** | **2.50** | **1.80** | **4.90** | **5.00** | **5.20** | **1.80** | **23.60** | **63.8 %** |

The LLM-assisted results show a clear improvement compared with the human-only phase. The average score increased from 12.85 to 24.00 out of 37 points. In percentage terms, the average correctness increased from 34.7 % to 63.8 %. This means that the use of LLMs almost doubled the average performance in this experiment.

The improvement was not equally distributed among all students. Some students achieved very high LLM-assisted scores, while others improved only moderately. This difference can be explained by several factors. First, the quality of the generated answer depended on the model used. Second, the quality of the prompt played an important role. Third, students still had to transfer, understand, and sometimes correct the LLM output within the limited time.

A closer analysis of Table 5.3 shows that the strongest LLM-assisted results were obtained by S10 with 32 points (86.5 %), S1 with 30 points (81.1 %), and S5 with 29.5 points (79.7 %). These results show that LLM support can lead to very high performance when the prompt is understood correctly and when the generated answer is transferred successfully to the answer sheet. However, the table also shows that LLM assistance did not automatically guarantee a good result for every student. For example, S3 achieved only 5 points (13.5 %), and S9 achieved 12.5 points (33.8 %). In the case of S3, the LLM-assisted score was even lower than the score achieved in the human-only phase. This result should not be interpreted only as a failure of the LLM answer quality, because S3 did not answer several questions with LLM support due to the limited time available during the experiment. This case shows that LLM assistance can only improve performance if students have enough time to use the tool, transfer the answer, and complete the required tasks. This indicates that LLM use remains dependent on the quality of interaction, the ability to guide the model, the available time, and the ability to recognize whether the generated answer actually fits the question.

The question-level averages reveal that LLM assistance was particularly useful for SQL-related subtasks and structured database operations. However, these averages must be interpreted together with the maximum score of each sub-question. After checking the point distribution, the results show that short SQL subtasks benefited strongly from LLM support, while longer tasks such as column-store compression still produced large variation between students. This indicates that LLMs can support syntax generation and structured query formulation, but they do not automatically guarantee correct application of a method to concrete data. Question 3b and Question 3c also show strong results, with average scores of 4.90 out of 6 and 5.00 out of 7. This confirms that LLMs were particularly useful for SQL and data warehouse query tasks, where students often struggle with syntax and exact formulation under time pressure.

Question 4 reached an average of 5.20 out of 10 points. This shows that LLMs helped with column-store compression, but the answers were still often incomplete. Some students received high scores, such as S1 and S10 with 10 points, while others received very low scores, such as S2 with 0 points and S9 with 0.5 points. This large variation suggests that compression tasks require not only naming a suitable method, but also applying it correctly to the given data. LLMs can explain compression methods well, but the generated answer still needs to be checked carefully because the concrete compressed representation may be incomplete or wrong.

Question 1 and Question 2 show more moderate average results, with 2.40 out of 4 and 2.50 out of 5 points. These tasks required conceptual knowledge and formal validation. The results suggest that LLMs can support these tasks, but they do not always identify all required details. This is especially relevant for XML and DTD validation, where small errors such as wrong element order, missing mandatory elements, or invalid attributes can easily be missed.

Question 5 had the lowest average score with 1.80 out of 3 points. Although the snapshot transformation task is short and structured, several students still received 0 points. This may be due to incomplete transfer of the LLM answer, misunderstanding of the expected add/delete/update format, or lack of time. The result shows that even simple structured tasks can fail if the prompt is unclear or if the student does not verify the generated output.

The most common LLM used in the experiment was ChatGPT. This is important because Chapter 4 showed that Claude achieved the highest score in the controlled benchmark. However, in the real-life experiment, most students selected ChatGPT. This shows that the model most frequently used by students is not necessarily the model that performs best in a scientific benchmark. Familiarity, accessibility, and user habits strongly influence real-world model choice.

Most students used photos as input. This confirms that image-based interaction is highly relevant in realistic exam-like situations, especially when questions contain tables, diagrams, or result relations. Several student comments also support this observation. One student noted that without a camera it would be difficult to describe the relations. This is especially relevant for SQL and database modeling questions because typing all tables manually would be time-consuming and error-prone.

Only a few students used their own prompts. Most participants used the original question directly or uploaded the question as a photo. This means that advanced prompt engineering was not common in the experiment. Students generally preferred the fastest and simplest interaction method rather than carefully designing prompts.

A particularly important case was observed for one student who uploaded the lecture script before using the LLM. This student instructed the model to answer based on the course material. The generated answers were very close to the professor's model solution, especially in the SQL section. This result is important because it shows that LLM performance can improve substantially when the model is provided with relevant domain-specific context. It also suggests that the use of lecture material can reduce the gap between generic LLM answers and the specific expectations of a course. The prompting behavior observed in the experiment is summarized in Table 5.4.

Table 5.4: Observed prompting behavior during the student experiment

| **Observed behavior** | **Count** | **Interpretation** |
|---|---|---|
| Used ChatGPT | 9 | ChatGPT was the dominant model because of familiarity and accessibility. |
| Used Gemini | 1 | Only one student used Gemini in the LLM-assisted phase. |
| Used photos as input | 9 | Most students preferred uploading the exam sheet instead of typing the prompt manually. |
| Used original exam question as prompt | 7 | Most students did not strongly reformulate the question. |
| Used own prompt or partial reformulation | 3 | A minority of students adapted the task or added their own wording. |
| Uploaded lecture material | 1 | One student used course-specific context, which strongly improved answer quality. |
| Reported time pressure | Majority | Several students did not finish all tasks within the available time. |

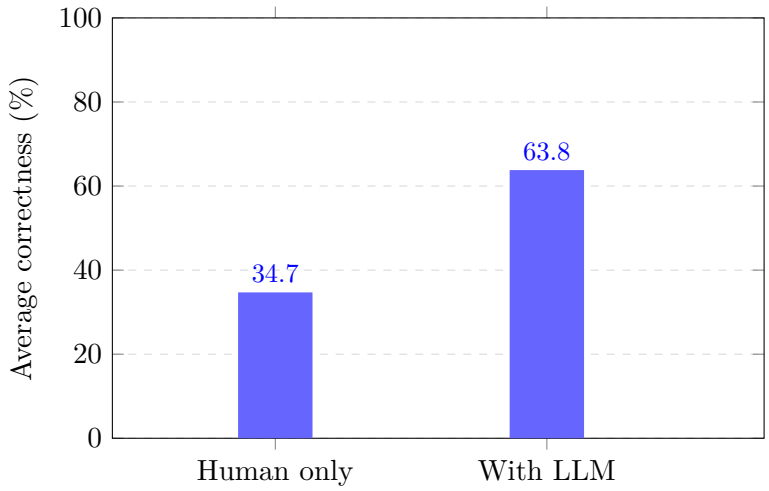



Figure 5.1: Average student performance without and with LLM assistance

Figure 5.1 visualizes the average performance difference between the human-only phase and the LLM-assisted phase. It shows that LLM assistance led to a strong improvement in average performance. The correctness increased by 29,1 percentage points. This confirms that LLMs can provide substantial support for solving database exam questions, especially under time pressure. However, the results should not be interpreted as showing that LLMs replace student knowledge. In several cases, students copied incomplete or partially incorrect answers from the LLM. Moreover, some students did not have enough time to verify the generated responses. Therefore, the benefit of LLMs depends not only on the model output, but also on the student's ability to interpret, adapt, and critically evaluate the answer.

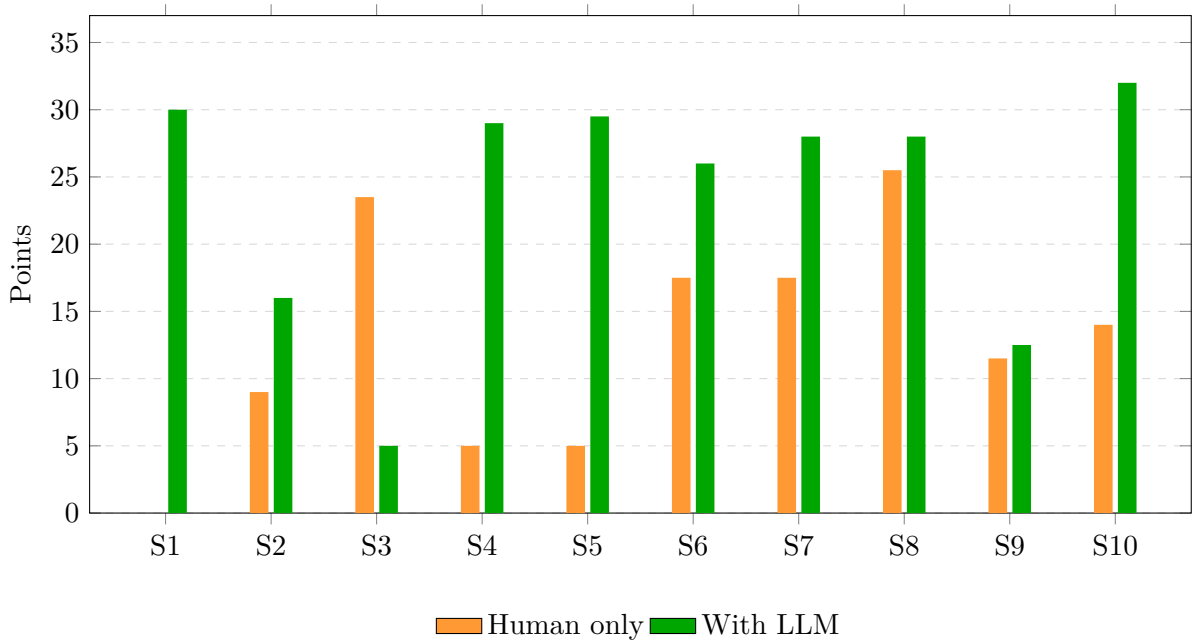



Figure 5.2: Individual comparison of student scores before and after LLM assistance

Figure 5.2 compares the individual scores of each student before and after LLM assistance. The individual comparison shows that every student improved after using an LLM. The improvement was especially large for students who initially left many questions unanswered. For example, Student 1 achieved no points in the human-only phase but obtained a much higher score after using an LLM. This suggests that LLMs can help students overcome missing syntax knowledge and provide a structured

starting point for solving complex tasks. At the same time, the figure also shows that students who already performed well without assistance still benefited from LLM use. This indicates that LLMs are not only helpful for weaker students, but can also improve the efficiency and completeness of stronger students.

When comparing these results with Chapter 4, one important difference becomes visible. In Chapter 4, the LLMs were evaluated directly and under controlled conditions. In this chapter, the LLM outputs are mediated by the students. The final answer depends on the prompt, the model response, and the student's ability to transfer the answer to the answer sheet. Therefore, the LLM-assisted score is not only a model score, but a combined human–AI performance score.

This distinction is important. In the controlled benchmark, Claude was the strongest model. In the real-life experiment, however, most students used ChatGPT. This shows that practical adoption does not necessarily follow benchmark performance. Students tend to use the tools they already know and trust, even if another model performs better in a controlled evaluation.

The experiment also provides an important contrast to the question reformulation chapter that follows. In the reformulation study, prompts are intentionally changed to test robustness and expose weaknesses in LLM reasoning. In the real-life experiment, most students did not use adversarial or complex prompts. Instead, they used simple and direct prompts, often by uploading the original question as a photo. This means that the student experiment reflects natural usage, while the reformulation chapter represents a controlled stress test.

## 5.3 Discussion

The real-life student experiment provides an important bridge between the controlled benchmark in Chapter 4 and the reformulation analysis in the following chapter. It shows that LLMs can significantly improve student performance in database exam questions, but it also reveals that the effectiveness of LLM assistance depends strongly on how the tool is used.

The first major finding is that LLM assistance nearly doubled the average score. The average correctness increased from 34.7 % in the human-only phase to 63.8 % in the LLM-assisted phase. This confirms that LLMs are powerful support tools for database-related tasks, especially when questions require formal syntax, structured output, or procedural reasoning.

The second major finding is that students mostly used LLMs in a simple and pragmatic way. Most participants uploaded photographs of the questions and used the original exam prompt without major changes. Very few students applied advanced prompt engineering strategies. This is important because it shows that real student behavior differs from controlled experimental setups. In practice, students often choose the fastest interaction method rather than the most carefully designed one.

The third finding concerns the role of multimodal input. Since many database exam questions contain tables, diagrams, and structured relations, image-based input is especially useful. Several students relied on photographs because typing all relations manually would have taken too much time. This confirms that LLM usage in technical subjects is not only a language interaction, but often a multimodal interaction involving images, tables, and diagrams.

The fourth important finding is the role of context. The student who uploaded the lecture script obtained answers that were much closer to the professor's expected solution. This suggests that LLMs

perform better when they are grounded in course-specific material. Generic LLM answers may be correct in principle but still differ from the notation, structure, or expectations used in a specific lecture. Providing the lecture script helps reduce this mismatch.

The fifth finding is that LLMs support students most strongly in syntax-heavy tasks. Several students struggled to remember SQL syntax, XML validation rules, or compression representations in the human-only phase. With LLM support, these tasks improved substantially. This suggests that LLMs are particularly useful as syntax and structure assistants. However, conceptual understanding remains necessary because students still need to judge whether the generated answer fits the task.

Compared with Chapter 2, the real-life experiment shows that benchmark performance and real usage are not identical. Chapter 4. In contrast, most students used ChatGPT in the real experiment. This shows that accessibility and familiarity are decisive factors in real educational settings.

Compared with the reformulation chapter, this experiment shows a different side of LLM usage. Reformulated prompts can reduce LLM reliability, especially when they introduce stricter constraints or require exact task interpretation. However, in the real-life experiment, students mostly used direct prompts and sometimes added helpful context. Therefore, prompt modification can have two opposite effects: poor or adversarial reformulation can reduce performance, while useful contextual enrichment can improve it.

Overall, this chapter confirms the central argument of the thesis: LLM performance in educational settings is not a fixed property of the model alone. It depends on the task, the prompt, the input format, the available context, and the user. A strong model can fail if the prompt is unclear, while a weaker or more common model can perform well if the prompt is direct and supported by relevant course material.

The results also have important implications for education. Simply banning or allowing LLMs is not enough. Students need guidance on how to use these systems critically and responsibly. They should understand that LLMs can provide useful support, but generated answers must still be checked carefully. At the same time, teachers should be aware that students may use multimodal input and course material to obtain highly accurate answers.

In conclusion, the real-life experiment demonstrates that LLMs can substantially support students in solving database exam questions, but their effectiveness depends strongly on practical usage behavior. This chapter therefore adds an important real-world perspective to the controlled benchmarking results and prepares the foundation for the next chapter, where the robustness of LLMs is tested through systematic question reformulation.

# Chapter 6

# *Reformulation Strategies for Evaluating LLM Limitations*

The benchmarking results presented in Chapter 4 demonstrated that Large Language Models (LLMs) are capable of solving a substantial number of database examination questions with relatively high accuracy. However, the previous experiments also revealed that model performance is not equally reliable across all task types. While some questions were solved correctly and consistently, other tasks exposed important weaknesses related to reasoning, completeness, formal precision, and task interpretation.

Recent research has shown that LLM performance is highly sensitive to prompt formulation, wording, structure, and contextual framing [21, 22]. Even semantically equivalent prompts may lead to significantly different outputs. This phenomenon is commonly referred to as prompt sensitivity. In educational contexts, this means that a model may answer a standard exam question correctly, while failing when the same question is reformulated in a slightly different but logically equivalent way.

For this reason, the second part of this thesis investigates the robustness of LLMs through systematic reformulation of database examination questions. The objective of this experiment is not to create unfair or misleading tasks, but rather to evaluate whether the models truly understand the underlying reasoning process or mainly rely on learned surface-level patterns. This methodological perspective is strongly supported by recent research on AI-resistant assessment design and adversarial robustness testing [23, 24].

## 6.1 Prompt Sensitivity and Robustness-Oriented Reformulation Strategies

The second experimental part of this thesis investigates whether Large Language Models remain reliable when database exam questions are reformulated in a controlled way. While Chapter 4 evaluated the performance of Claude, Gemini, and Kimi on the original exam questions, this chapter focuses on the robustness of their answers under modified prompt formulations. The aim is not to create unfair or misleading questions, but to test whether the models can still follow the exact task requirements when the wording, structure, or expected output format becomes more restrictive. This approach is motivated by research on prompt sensitivity, which shows that even small changes in wording, structure, or contextual framing can substantially influence LLM behavior [21, 22]. The reformulation strategies used in this chapter are based on recent scientific work on LLM-resistant assessments, prompt

robustness, and adversarial evaluation. Larsen argues that exam questions should be redesigned in a way that requires contextual understanding, reasoning, and precise application instead of relying only on standard textbook knowledge [23]. Stoyanov and Nedelcheva similarly show that LLMs perform better on direct factual questions than on questions requiring application, interpretation, or domain-specific reasoning [25]. McDanel and Novak further demonstrate that LLMs struggle more with long, multi-step, visually complex, and formally structured computer science tasks [26]. This is directly relevant to database exams, where students must often combine modeling, transformation, SQL reasoning, and formal notation in one task.

The reformulated questions were therefore designed around six main methods. First, some questions were made more restrictive by requiring exact output formats, because LLMs often generate fluent but unnecessarily broad answers [23]. Second, several questions were reformulated to require complete enumeration, since benchmarks such as BIG-bench show that model performance varies strongly on tasks requiring exhaustive reasoning over all valid cases [18]. Third, some questions were redesigned as multi-step reasoning tasks, because previous work shows that LLMs may produce plausible final answers while failing to maintain correct reasoning chains [26, 4]. Fourth, tasks were reformulated to reduce implicit assumptions and force the model to explicitly handle missing or ambiguous information, following robustness-testing ideas similar to MedFuzz, where semantically modified questions are used to expose unreliable model behavior [24]. Fifth, some questions required strict formal precision, because LLMs often struggle when natural language explanations must be transformed into exact symbolic, logical, or schema-based outputs [19]. Finally, the questions were designed to test the gap between explanation and execution, since LLMs may correctly describe a method but fail to apply it precisely to the given data [4, 16].

Each reformulated question was submitted to Claude, Gemini, and Kimi in a new chat session. This was done to avoid context contamination from previous interactions. The generated answers were then evaluated using the same grading logic as in Chapter 4: correctness, completeness, adherence to the task, and formal precision.

**Reformulated Question 3: ER Modeling**

(a) Convert the description into a standard ER model using interval notation. You have only to include attributes explicitly mentioned.

(b) Transform the ER model into the relational model. You have to include only necessary relations and clearly indicate primary and foreign keys.

(c) Provide the result relations of the queries. You have to write down all intermediate steps. If the result cannot be uniquely determined, explicitly justify why.

The reformulation of Question 3 A focuses on multi-step reasoning, explicit constraint following, and the reduction of implicit assumptions. In the original benchmark, this question already exposed several weaknesses because it combines ER modeling, relational transformation, and SQL result computation. The reformulated version makes these requirements more explicit by stating that only attributes explicitly mentioned may be used, only necessary relations should be included, and intermediate steps must be shown. This method is supported by McDanel and Novak, who show that LLMs are less reliable on long, multi-step, and structurally complex computer science assignments [26]. It is also supported by

research on prompt sensitivity, because restructuring a task into stricter sub-requirements can change the way a model interprets and solves the problem [21, 22]. The main goal of this reformulation is to prevent the model from filling gaps with plausible but unsupported assumptions. This is important because LLMs often generate confident answers even when the available information is incomplete or ambiguous [4]. By requiring the model to explicitly justify cases where a result cannot be uniquely determined, the question tests whether the model recognizes uncertainty instead of hallucinating missing information. This is closely related to adversarial robustness testing, where modified but semantically valid inputs are used to examine whether a model remains reliable under changed conditions [24].

**Reformulated Question 4: XML and DTD Validation**

(a) Examine whether the given XML document is valid with respect to the provided DTD. Identify all elements that violate the DTD and, for each error, specify the exact rule that is violated, the position in the document, and a short justification. Only the incorrect elements should be listed.

The reformulation of Question 4 A transforms the task from a general validation question into an error-localization task. Instead of allowing the model to give a broad explanation or rewrite the XML document, the reformulated question requires a precise list of incorrect elements, the violated DTD rule, the position of the error, and a short justification. This method is based on the idea that evaluation should not only measure whether a model can generate plausible text, but also whether it can verify and localize specific errors in a structured input [16]. The CheckList framework also supports this type of behavioral testing, because it argues that models should be evaluated on specific capabilities rather than only on global accuracy [20].

This reformulation targets a common LLM weakness: models often produce correct-looking explanations but miss small formal violations. XML validation requires exact rule application, correct ordering, and attention to declared attributes. By asking only for incorrect elements, the question reduces the space for unnecessary explanation and forces the model to focus on formal verification. This follows the broader principle of AI-resistant assessment design, where tasks should require precise application rather than generic textbook-style answers [23, 25].

**Reformulated Question 5: Formal Logic**

(a) Construct the complete truth table for the given formula, ensuring that all possible combinations of truth values are included.

(b) Based solely on the constructed truth table, determine whether the formula is a tautology.

(c) Prove the equivalence of the given expressions using formal transformation rules. Each step of the transformation must be explicitly stated and must follow a valid logical rule.

(d) Translate the given statements into predicate logic using a direct and exact translation. No simplification or interpretation beyond the given expressions is allowed.

The reformulation of Question 5 A focuses on strict symbolic reasoning and formal proof construction. The question no longer accepts intuitive explanations or short final answers; instead, it requires a complete truth table, a tautology decision based only on that table, formally valid equivalence transformations, and direct predicate-logic translation. This method is grounded in research showing that LLMs can struggle with symbolic reasoning tasks, especially when every intermediate step

must be logically valid [19]. Even when chain-of-thought prompting improves performance on some reasoning tasks, it does not fully eliminate failures in difficult formal reasoning problems [13, 19]. This reformulation is particularly relevant because formal logic tasks are easy to make superficially convincing. A model may state that a formula is a tautology or that two expressions are equivalent without providing a complete and valid proof. By requiring every truth-table row and every transformation rule, the question tests whether the model can execute the reasoning process rather than only produce a plausible conclusion. This also addresses the distinction between surface fluency and actual correctness, which is a central concern in LLM evaluation [3, 16].

**Reformulated Question 6: Frequent Itemsets and Association Rules**

Determine all frequent itemsets of size one and two for the given dataset. The solution must include all valid itemsets; incomplete or additional itemsets are considered incorrect. Based on these itemsets, derive all association rules with a confidence of at least 0.5 and show the calculation of the confidence for each rule.

The reformulation of Question 6 A focuses on completeness and exhaustive enumeration. Frequent itemset mining is not only about finding some correct itemsets, but about identifying all valid itemsets under the given condition. Therefore, the reformulated question explicitly states that incomplete or additional itemsets are considered incorrect. This method is supported by large-scale benchmarking research showing that LLM performance varies strongly when tasks require exhaustive coverage rather than a single final answer [18]. It is also consistent with the multidimensional evaluation perspective of HELM, where completeness and robustness are treated as important evaluation dimensions in addition to accuracy [16].

This reformulation targets a weakness already observed in the benchmarking chapter: LLMs often provide correct subsets while omitting required elements. By forcing the model to list all size-one and size-two itemsets and then derive all rules with confidence at least 0.5, the task becomes sensitive to omissions. A single missing itemset can affect the following association rules, which makes this reformulation useful for testing whether the model can maintain completeness across dependent steps. This also reflects the idea of multi-step robustness testing, where errors in early steps can propagate into later parts of the answer [26, 24].

**Reformulated Question 7: SQL Query Construction**

Write a single SQL query that produces exactly the result table shown. The structure of the query must correspond to the expected solution, and only the required SQL constructs may be used. Queries that produce the correct result but use a different structure or alternative operators are considered incorrect.

The reformulation of Question 7 A focuses on strict output structure and exact alignment with the expected solution. SQL questions are often solved by LLMs using syntactically valid but structurally different queries. However, in an exam context, especially when a specific construct such as `ROLLUP` or `CUBE` is expected, an alternative query may not demonstrate the intended learning objective. This reformulation therefore removes flexibility by requiring one query that uses only the required constructs. This method follows the idea that assessment tasks should evaluate not only whether the final result

is plausible, but also whether the required procedure and representation are followed [16, 25]. This reformulation is also related to prompt sensitivity research, because the explicit restriction to one acceptable structure changes the output space of the model [21]. Instead of allowing the model to generate any semantically similar SQL solution, the task tests whether it can follow formal constraints. This is important because LLMs often optimize for plausible completion rather than exact conformity to an expected solution format [4]. In this thesis, this strategy is used to test whether the models can distinguish between a generally correct SQL idea and the exact SQL structure required in the exam.

**Reformulated Question 8: Column Compression**

For each column of the given dataset, select an appropriate compression method and apply it step by step to the data. The final compressed representation must be explicitly shown. Answers that only describe the method without applying it to the data are not considered sufficient.

The reformulation of Question 8 A targets the difference between explaining a method and actually applying it. LLMs are often able to describe database compression methods such as dictionary encoding, run-length encoding, or delta encoding, but this does not necessarily mean that they can apply the method correctly to a concrete dataset. This reformulation therefore requires the model to select a compression method for each column and then explicitly produce the compressed representation. The method is supported by evaluation research emphasizing that LLM assessment should go beyond general explanation and test concrete task execution [16, 4].

This reformulation also follows the broader principle of AI-resistant assessment design: instead of asking for a definition or a general description, the question requires applied reasoning on specific data [23, 25]. It tests whether the model can transfer conceptual knowledge into a correct procedural output. This is particularly important for database exams because many tasks require applying an algorithm or transformation to given tables, not only explaining the algorithm in natural language. By requiring the final compressed representation, the task exposes incomplete answers that remain at the explanation level.

**Reformulated Question 9: Differential Snapshot Transformation**

Determine the sequence of operations required to transform snapshot $i$ into snapshot $i+1$ using the operations ADD, DELETE, and UPDATE. Each operation must include the complete tuple, and the order of operations must be correct. Missing or unnecessary operations will result in an incorrect solution.

The reformulation of Question 9 A focuses on strict sequential transformation and exact output control. In the original benchmark, this question was solved correctly by all models, which indicates that the basic task is highly structured and relatively easy for LLMs when the required operations are clear. The reformulated version therefore increases the precision requirements by demanding the complete tuple, correct operation type, correct order, and no missing or unnecessary operations. This method is consistent with robustness-oriented evaluation, where a task that was previously solved correctly is reformulated to test whether the model remains stable under stricter constraints [24, 21].

This reformulation is also connected to reasoning about state changes. A snapshot transformation requires the model to compare two states and derive the exact sequence of changes. Research on LLM

reasoning and planning has shown that models may struggle when they need to maintain consistency across changing states or sequential operations [27]. By requiring exact ADD, DELETE, and UPDATE operations, the question tests whether the model can perform precise state-transition reasoning rather than only describing the difference in general terms. Since the expected output is short but strict, this question also helps identify whether reformulation is more effective for complex tasks than for simple, highly constrained ones.

## 6.2 Evaluation of the LLMs' Results

To evaluate the impact of the reformulation strategies presented in Section 6.1, the same benchmarking methodology from Chapter 4 was applied to the reformulated exam questions. The objective was to measure how the reformulated prompts influenced the performance of the selected Large Language Models (LLMs), especially regarding completeness, precision, reasoning consistency, and the ability to follow strict instructions.

The benchmarking process followed the same evaluation criteria used in the overall benchmark of Subsection 4.3.2. Each answer was manually corrected and compared with the expected solutions. Points were assigned according to correctness, completeness, and respect of the requested formal structure.

The reformulated questions were intentionally designed to increase ambiguity control, require stricter formal reasoning, and force the models to follow precise constraints. Unlike the original questions, several reformulated tasks explicitly requested complete intermediate steps, exact rule localization, or the use of one specific method only. This makes the reformulated benchmark particularly useful for evaluating whether LLMs truly understand the task requirements or simply generate plausible-looking answers.

Table 6.1: Benchmark Results for Reformulated Questions

| **Model** | **Points** | **Correctness before Reformulation** | **Correctness after Reformulation** |
|---|---|---|---|
| Claude | 56 / 68 | 86 % | 82.4 % |
| Kimi | 28.5 / 68 | 63 % | 41.9 % |
| Gemini | 22 / 68 | 59 % | 32.4 % |

Table 6.1 summarizes the final results obtained by each model. The results show that Claude remained the strongest model after reformulation, while Kimi and Gemini experienced a strong performance decrease. This confirms that reformulation strategies can significantly influence the reliability and stability of LLM outputs.

#### Claude Benchmarking Results

Claude achieved 56 out of 68 points, corresponding to 82.4% correctness. Compared to the original benchmark from Subsection 4.3.2, Claude remained relatively stable and demonstrated strong robustness against the reformulation strategies.

The model performed particularly well in structured reasoning tasks such as logic proofs, association rules, SQL aggregation queries, and snapshot transformation tasks. Claude also provided clear intermediate reasoning steps in most questions, which improved readability and traceability of the generated solutions.

However, the reformulated questions still exposed several weaknesses. In Question 4 A, Claude correctly detected the missing `<planer>` element and the ordering violation in the XML document, but failed to identify the invalid `type` and `typ` attributes in the `<veranstaltung>` elements. This indicates that even strong models can miss localized rule violations when multiple constraints must be checked simultaneously.

A similar issue appeared in Question 8 A. Claude correctly applied compression techniques such as dictionary encoding, delta encoding, and run-length encoding, but initially proposed multiple possible methods for the Ozon column before finally selecting one. The reformulated question explicitly required exactly one compression method per column, meaning that the model still showed a tendency to over-explain instead of strictly following the requested constraint.

Another important observation concerns Question 7 A, where Claude used `ROLLUP` instead of the expected `CUBE` operation in one SQL aggregation task. This behavior is similar to the problems already observed in the original benchmark and suggests that the model sometimes prefers statistically common solutions instead of strictly matching the expected formal answer.

Overall, Claude proved to be the most resistant model against the reformulation strategy. Although some precision errors remained, the model generally preserved answer completeness and logical consistency.

Table 6.2: Claude Results on Reformulated Questions

| **Q** | **Points before Reformulation** | **Points after Reformulation** | **Max** | **What Claude did correctly** | **What Claude did wrong** |
|---|---|---|---|---|---|
| Q3 | 9.5 | 9 | 14 | Correct ER interpretation, relational mapping, and mostly correct SQL results with intermediate steps. | Added unnecessary formal details and did not fully respect the strict reformulated instructions. |
| Q4a | 5 | 2 | 5 | Correctly detected missing `<planer>` and ordering violation. | Missed invalid attributes `type` and `typ`. |
| Q5 | 10 | 12 | 14 | Correct tautology, equivalence proof, and predicate logic translation. | Truth table based on interpreted formula instead of exact question formula. |
| Q6 | 7 | 7 | 7 | Complete frequent itemsets and association rules. | No major error. |
| Q7 | 14 | 14 | 15 | Correct SQL aggregation queries and structured answers. | Used `ROLLUP` instead of expected `CUBE` in one query. |

| | | | | | |
|---|---|---|---|---|---|
| Q8 | 9 | 9 | 10 | Correct compression methods with explicit compressed representations. | Initially proposed multiple methods before selecting one. |
| Q9 | 3 | 3 | 3 | Correct DELETE, UPDATE, and ADD operations. | No error. |
| **Total** | **57.5** | **56** | **68** | | |

**Kimi Benchmarking Results**

Kimi achieved 28.5 out of 68 points, corresponding to 41.9 % correctness. The reformulated questions had a strong negative impact on Kimi's performance, especially regarding task completeness and document interpretation.

The most significant weakness of Kimi was its inability to consistently execute all required subtasks. In Question 3, the model focused mainly on the SQL section while completely ignoring parts of the ER transformation task. Similarly, in Questions 7 and 8, Kimi treated the questions as incomplete or missing information even though the required data was visible in the reformulated exam document.

This behavior is particularly important for the thesis because it demonstrates that reformulation strategies can expose weaknesses related to document parsing and contextual understanding. Kimi often interpreted formatting ambiguities or visually complex layouts as missing information, which caused it to stop solving the task entirely.

At the same time, Kimi performed relatively well in bounded formal reasoning tasks. In Question 4, it eventually identified all XML validation errors correctly, even though its reasoning process was initially confused. The model also produced a correct logical equivalence proof and handled the snapshot transformation task correctly.

The results suggest that Kimi is capable of solving formal reasoning problems when the task boundaries are explicit and localized. However, the model becomes unreliable when the task requires combining several subtasks, interpreting complex layouts, or maintaining completeness across multiple sections.

Table 6.3: Kimi Results on Reformulated Questions

| **Q** | **Points before Reformulation** | **Points after Reformulation** | **Max** | **What Kimi did correctly** | **What Kimi did wrong** |
|---|---|---|---|---|---|
| Q3 | 3 | 4 | 14 | Solved large parts of the SQL queries with intermediate steps. | Ignored Q3(a) and Q3(b); changed projection structure in SQL output. |
| Q4a | 2 | 5 | 5 | Correctly identified all XML validation errors. | Reasoning process was initially inconsistent and self-corrected later. |

| | | | | | |
|---|---|---|---|---|---|
| Q5 | 13 | 11.5 | 14 | Correct tautology, proof, and predicate logic translation. | Truth table based on interpreted formula rather than exact formula. |
| Q6 | 7 | 5 | 7 | Listed many frequent itemsets and association rules. | Incorrect support count for Bathrobe caused several wrong confidence values. |
| Q7 | 10 | 0 | 15 | No valid answer. | Treated question as incomplete and did not solve it. |
| Q8 | 2.5 | 0 | 10 | No valid compression answer. | Incorrectly assumed the dataset was missing. |
| Q9 | 3 | 3 | 3 | Correct DELETE, UPDATE, and ADD operations. | No error. |
| **Total** | **40.5** | **28.5** | **68** | | |

**Gemini Benchmarking Results**

Gemini achieved 22 out of 68 points, corresponding to 32.4 % correctness, making it the weakest model in the reformulated benchmark. The main weakness observed in Gemini was incompleteness. In several questions, the model answered only selected parts of the task while omitting other required sections entirely. This behavior was particularly visible in Question 5, where Gemini correctly solved the tautology and predicate logic translation but omitted the complete truth table and formal equivalence proof. A similar pattern appeared in Question 6. Gemini identified some frequent itemsets and association rules, but the answer remained highly incomplete and missed several valid combinations explicitly required by the reformulated instructions. Another important issue appeared in Question 3, where Gemini hallucinated or misread the relation data used in the SQL queries. As a result, the generated SQL outputs were based on incorrect tables and therefore produced invalid results. This confirms the same tendency already observed in Subsection 4.3.2: Gemini can produce partially useful answers, but it struggles with precise data extraction and strict formal completeness. Despite these weaknesses, Gemini still demonstrated partial understanding in localized tasks such as XML validation and snapshot transformation. However, the overall results indicate that reformulation strategies strongly destabilized the model, especially in multi-part formal reasoning tasks.

Table 6.4: Gemini Results on Reformulated Questions

| **Q** | **Points before Reformulation** | **Points after Reformulation** | **Max** | **What Gemini did correctly** | **What Gemini did wrong** |
|---|---|---|---|---|---|
| Q3 | 5.5 | 2 | 14 | Partial relational-model transformation. | Did not answer Q3(a); used wrong relation data in SQL queries. |

| | | | | | |
|---|---|---|---|---|---|
| Q4a | 4 | 4 | 5 | Correctly identified most XML validation errors. | Did not clearly separate invalid `type` and `typ` attributes. |
| Q5 | 0 | 7 | 14 | Correct tautology and predicate logic translation. | Missing truth table and equivalence proof. |
| Q6 | 7 | 2 | 7 | Identified some itemsets and association rules. | Highly incomplete answer with many missing valid rules. |
| Q7 | 15 | 0 | 15 | No valid answer. | SQL task omitted completely. |
| Q8 | 3 | 4 | 10 | Partial dictionary encoding and RLE answer. | Compression representation incomplete. |
| Q9 | 3 | 3 | 3 | Correct DELETE, UPDATE, and ADD operations. | No error. |
| **Total** | **37.5** | **22** | **68** | | |

To summarize the errors observed in the reformulated benchmark, the main failure patterns can be organized into a taxonomy of LLM failure types. This taxonomy makes it possible to distinguish between different kinds of wrong answers, instead of treating all errors as the same problem.

Table 6.5: Taxonomy of LLM failure types in the reformulated benchmark

| **Failure type** | **Description** | **Example observed in this thesis** | **Main affected model** |
|---|---|---|---|
| Omission of subtasks | The model answers only part of a multi-part question and leaves out required sections. | Gemini omitted parts of the formal logic task and did not provide all required intermediate outputs. | Gemini |
| Task drift | The model answers a related topic instead of the exact task that was asked. | Kimi sometimes produced answers that were semantically related, but did not match the requested database task. | Kimi |
| Formal mismatch | The model understands the general idea, but uses the wrong required representation. | Claude produced a UML-style diagram instead of the expected ontology-style representation. | Claude |
| Hallucinated or misread data | The model changes, reconstructs, or invents input data instead of using the provided data exactly. | Gemini generated SQL results based on incorrect relation data. | Gemini |
| Overgeneration | The model gives unnecessary alternatives or explanations even though the task asks for a strict output format. | Claude proposed several compression methods before selecting one. | Claude |
| Layout or document-interpretation failure | The model treats visible information as missing because of the document structure or visual complexity. | Kimi assumed that required data was missing in some reformulated tasks. | Kimi |
| Incomplete execution | The model describes the correct method, but does not fully apply it to the given data. | Gemini and Kimi often explained compression methods without producing a complete compressed representation. | Gemini and Kimi |

This taxonomy shows that LLM errors in database examination tasks are not only factual mistakes. Many errors are caused by omitted subtasks, weak alignment with the required formal representation, incomplete execution, or sensitivity to document structure. This supports the central argument of this thesis: LLM performance should not be evaluated only by final correctness, but also by completeness, formal precision, and robustness when task conditions change.

The reformulated benchmark clearly demonstrates that the different LLMs react differently to prompt reformulation strategies. Claude remained relatively robust and preserved most of its reasoning quality even under stricter instructions. In contrast, Kimi and Gemini showed major reductions in completeness and reliability.

The most common failure patterns observed after reformulation were:

- Omission of subtasks
- Incorrect interpretation of document structure
- Hallucination of missing data
- Inability to maintain strict completeness
- Failure to follow exact output constraints

These findings confirm the hypothesis of this thesis: reformulation strategies can significantly affect LLM performance, especially in formal academic tasks that require precision, structured reasoning, and complete execution of all requested subtasks.

## 6.3 Discussion of the Question Reformulation

The results of the reformulated benchmark demonstrate that question reformulation has a significant impact on the behavior and reliability of Large Language Models (LLMs). While all models were able to solve at least parts of the reformulated tasks, the experiments clearly showed that changing the structure, wording, constraints, and level of precision in the prompts can strongly influence answer quality, completeness, and reasoning stability.

One of the most important observations is that reformulation does not necessarily make the questions more difficult in terms of domain knowledge. Instead, the reformulations mainly increase the cognitive and structural complexity of the task. Several reformulated questions explicitly required additional constraints, which exposed weaknesses that were less visible in the original benchmark from Chapter 4:

- Complete intermediate reasoning steps
- Exact localization of errors
- Strict output formatting
- Explicit method selection
- Complete coverage of all subtasks

A major pattern observed across the experiments is the problem of incompleteness. Both Gemini and Kimi frequently answered only parts of the reformulated tasks while ignoring or omitting other required sections. This behavior appeared especially in multi-part formal questions such as SQL exercises, logical proofs, and data compression tasks. Even when the models demonstrated partial understanding, they often failed to fully execute all instructions contained in the reformulated prompt.Another important finding concerns document interpretation and contextual understanding. Kimi, in particular, showed high sensitivity to document layout and visual structure. In several cases, the model incorrectly

assumed that information was missing, although the required data was present in the reformulated exam document. This suggests that some LLMs struggle not only with reasoning itself, but also with extracting structured information from complex or visually dense inputs.

The experiments also revealed that reformulation can increase hallucination risks. Gemini demonstrated this behavior in Question 3 by generating SQL results based on incorrect relation data. Instead of strictly following the provided tables, the model partially reconstructed or altered the input information. This confirms that reformulated prompts can destabilize the internal assumptions used by LLMs during reasoning and data extraction.

Claude showed the highest resistance against the reformulation strategies. In most cases, the model preserved logical consistency, answer completeness, and formal reasoning quality. However, even Claude exhibited several weaknesses when the reformulated questions required strict rule localization or exact constraint following. For example, Claude missed XML attribute violations in Question 4 and initially proposed multiple compression methods in Question 8 despite explicit instructions to choose only one. This demonstrates that even advanced models remain vulnerable to instruction precision and constraint sensitivity. The benchmark also highlights an important distinction between localized reasoning tasks and globally connected tasks. All models generally performed better in isolated formal tasks with clearly bounded rules, such as tautology checks, XML validation, or snapshot transformation. In contrast, performance decreased significantly when the questions required:

- Combining multiple subtasks
- Maintaining long reasoning chains
- Interpreting complex structures
- Strictly respecting several simultaneous constraints

This finding is highly relevant for educational and academic contexts. Many university exam questions require not only theoretical knowledge, but also the ability to carefully follow instructions, maintain completeness, and connect several reasoning steps together. The experiments demonstrate that LLMs may appear highly capable when solving isolated subtasks, while still failing to fully satisfy formal academic requirements.

Another important observation is that reformulation strategies can function as a robustness test for LLMs. Small changes in wording, structure, or formatting were often sufficient to reduce performance significantly, especially for weaker models. This suggests that benchmark results based only on standard prompts may overestimate the true reliability of LLMs in real-world educational environments, where questions are naturally phrased in many different ways.

Overall, the reformulated benchmark confirms the central hypothesis of this thesis: the performance of LLMs is highly sensitive to prompt formulation. Reformulation strategies successfully exposed weaknesses related to completeness, reasoning consistency, instruction following, and structured data interpretation. At the same time, the experiments also showed that stronger models such as Claude remain more robust under reformulation, although they are not completely immune to precision-related failures.These findings are important for both educational practice and future research. In academic settings, they demonstrate that LLM-generated answers should not automatically be considered reliable simply because they appear fluent or convincing. From a research perspective, the results indicate that future evaluations of LLMs should include adversarial or reformulated prompts in order to better measure robustness, reasoning stability, and true understanding capabilities.

# Chapter 7

# *Discussion*

The results of this thesis show that LLMs can solve many database examination questions with high fluency and, in several cases, high correctness. However, the findings also show that this performance is not stable enough to be interpreted as reliable exam competence in every situation. Across the empirical chapters, one central pattern becomes clear: LLMs perform well when a task follows a familiar structure, contains explicit instructions, and can be solved through standard patterns. Their reliability decreases when a task requires strict formal precision, complete execution of several subtasks, interpretation of structured input, or robustness against reformulation.

The discussion therefore focuses on four central insights. First, LLM performance in Database examinations is strongly task-dependent and cannot be reduced to one general model score. Second, the practical effect of LLMs in education depends not only on model capability, but also on user behavior, input modality, and available context. Third, reformulation strategies can expose weaknesses that remain less visible in standard benchmark settings. Fourth, the results have direct implications for the design of AI-aware database examinations.

The controlled benchmark in Chapter 4 provides the first important result. Claude achieved the highest overall score with 86.5 out of 101 points, corresponding to 85.6 % correctness. Kimi reached 64 out of 101 points, corresponding to 63.4 %, and Gemini achieved 60 out of 101 points, corresponding to 59.4 %. These results show that all three models were able to solve a considerable part of the selected database examination questions. At the same time, the gap between Claude and the other two models was substantial.

Claude was the most reliable model in the controlled benchmark because it usually followed the task instructions more precisely, completed more sub-parts, and produced more formally correct answers. Gemini and Kimi showed more unstable behavior. Gemini often produced useful partial answers but omitted required parts. Kimi sometimes answered a related topic instead of the exact question. This confirms that LLM performance should not be understood as one general ability. It is a task-dependent and model-dependent phenomenon.

This result is consistent with previous studies on LLM performance in examination environments. Sadeq et al. evaluated several publicly available AI chatbots on simulated UK medical board-style examination questions and found that the models showed promising performance, but also important limitations depending on question type and reasoning complexity [7]. Their study tested 423 board-style

questions from several UK medical examinations and showed that even strong models did not perform equally well across all categories [7]. The findings of this thesis show a similar pattern in the domain of database examinations. Claude achieved strong overall results, but still made errors in XML validation, SQL structure, and exact formal representation. Gemini and Kimi also solved some tasks correctly, but failed strongly in others. Therefore, both studies support the same broader conclusion: LLMs can be useful in exam-like tasks, but their reliability depends strongly on task structure, domain constraints, and answer format.

A second important finding from Chapter 4 is that LLMs are especially strong on tasks that resemble known patterns. For example, all three models performed well on several RDF, RDF-S, Bayes, SQL template, and snapshot-transformation tasks. These questions have relatively clear structures and limited solution spaces. When the model recognizes the expected pattern, it can often generate a correct or nearly correct answer. This agrees with the view that LLMs are powerful pattern-based systems that can reproduce learned solution forms effectively [2, 4].

However, this strength also becomes a weakness when the task requires exact interpretation rather than pattern completion. In Question 2 of the benchmark, Kimi answered a related data warehouse topic instead of filling the required table about modeling styles. The answer was not meaningless, but it did not solve the actual task. This kind of error is important because it shows that an LLM can produce a plausible answer from the same domain while still missing the specific requirement of the question.

The results also confirm that fluency is not the same as correctness. All evaluated models generated answers that were well written, structured, and often very detailed. In several cases, however, the models added unnecessary explanations, alternative solutions, or information that was not requested. This is problematic in an examination context because excessive detail can hide mistakes and make the answer harder to evaluate. Clark and Etzioni argue that success on standardized tests does not necessarily prove deeper intelligence or understanding, because systems can succeed through surface-level patterns and test-specific strategies [3]. The results of this thesis support this argument. A model may sound confident and produce a polished answer while still missing a required diagram, using the wrong SQL operator, or omitting part of a formal proof.

The comparison between different task types further shows that the most difficult questions were those requiring exact constructed outputs. ER modeling, relational transformation, exact SQL result relations, column-store compression, and formal logic were more challenging than direct conceptual questions. These tasks require several connected steps. A wrong interpretation at the beginning can propagate through the rest of the answer. This was visible in Question 3, where the models had to combine ER modeling, relational transformation, and SQL result reasoning. Claude performed best, but all models showed at least some difficulty.

This finding is also aligned with research on mathematical and scientific question answering. Bhattacharya explains that math and science question answering requires more than retrieving text: systems must analyze the question, understand structured information, and apply reasoning steps to reach the answer [28]. Database examination questions share this property. They often require structured reasoning over tables, schemas, relations, constraints, and transformations, not only natural-language explanation. A central contribution of this thesis is the observation that LLM performance depends strongly on the type of database examination task. Some task types are easier for

LLMs because they follow familiar patterns, while others are more difficult because they require exact construction, complete enumeration, or careful interpretation of structured data. Table 7.1 summarizes this characterization.

Table 7.1: Characterization of database exam task types according to LLM performance

| **Task type** | **LLM difficulty** | **Reason** | **Implication for exam design** |
|---|---|---|---|
| RDF, RDF-S, and Turtle transformations | Low to medium | These tasks follow known symbolic patterns and clear syntax rules. | Useful for testing basic knowledge, but not very resistant to LLM support when asked in a standard form. |
| Bayes calculation and formula-based tasks | Low | The solution follows a standard mathematical template. | Should be combined with interpretation, justification, or changed data to avoid purely template-based answers. |
| Snapshot transformation | Low | The task is explicit and operational, with clear add, delete, and update operations. | Can be solved well by LLMs; stronger assessment requires explanation of the state changes. |
| SQL query construction | Medium | LLMs often know SQL patterns, but may use a different structure than the expected solution. | Require exact constructs, result verification, and explanation of why the query produces the required table. |
| XML and DTD validation | Medium | The task requires attention to small formal errors, element order, and attribute constraints. | Error-localization tasks are stronger than general validation questions. |
| ER modeling and relational transformation | Medium to high | These tasks require design decisions, correct cardinalities, primary keys, foreign keys, and consistency across transformations. | Good for AI-aware exams, especially when students must justify their modeling decisions. |
| Formal logic and proof tasks | High | These tasks require complete truth tables, valid transformation rules, and exact symbolic reasoning. | Strong assessment tasks when every intermediate step must be shown and justified. |
| Frequent itemsets and association rules | High | The task requires complete enumeration; one missing itemset can affect later rules. | Useful for testing completeness and step-by-step reasoning. |
| Column-store compression | High | LLMs can describe compression methods, but often fail to apply them completely to concrete data. | Very useful for AI-aware exams because it tests the gap between explanation and execution. |
| Multi-part reformulated questions | High | These tasks combine constraints, strict output formats, and several dependent steps. | Strongest type for testing robustness, completeness, and real understanding. |

This characterization shows that standard textbook-style questions are often easier for LLMs than tasks requiring exact application to concrete data. LLMs are especially strong when the expected solution follows a known pattern. They become less reliable when they must connect several steps, respect strict output constraints, or produce a complete formal representation. This distinction is important for future database examinations. It suggests that AI-aware assessment should not only ask whether students know a concept, but also whether they can apply it, justify it, and verify the correctness of the result.

The time comparison in Chapter 4 adds a practical dimension to the discussion. Photo-based input was much faster than typing the prompt manually. Typing the prompt required around 126 seconds for Gemini and Claude and 186 seconds for Kimi, while photo input required only 16.3 seconds for Gemini, 13.77 seconds for Claude, and 36.66 seconds for Kimi. This indicates that, in realistic exam-like situations, multimodal interaction can strongly reduce the effort required to use an LLM.

However, faster input does not always mean better overall performance. Kimi required 191.4 seconds to generate an answer from photo input, which made the total interaction much longer. This shows that usability depends on both the input method and the model behavior. In practice, students may prefer photos because they are faster and easier, especially for questions containing tables, diagrams, or schemas. Nevertheless, the quality and speed of the final answer still depend on the model.

The multilingual benchmark also supports the broader conclusion that LLM performance is not fixed. Claude remained stable across German, English, and French with 12.5 out of 17 points in all three languages. Gemini varied moderately, and Kimi showed strong language sensitivity, improving in English but performing much worse in German and French. This result is important because database exams at universities may exist in different languages, and students may also translate prompts before using an LLM. The findings show that language choice can change answer quality, especially for complex multi-part tasks.

The effect of language was much stronger for Question 3 than for Question 9. Question 9 remained stable because it was short, explicit, and operational. Question 3 required several formal steps and was therefore more sensitive to language changes. This supports the idea that prompt sensitivity increases when the task becomes more complex and requires more structured reasoning [13, 20].

Chapter 5 adds an important real-world perspective. In the student experiment, ten students solved selected database exam questions first without assistance and then with an LLM of their choice. The average human-only score was 12.85 out of 37 points, corresponding to 34.7 %. With LLM assistance, the average score increased to 24 out of 37 points, corresponding to 63.8 %. This is a strong improvement of 29.1 percentage points. The result shows that LLMs can significantly support students in solving database examination questions, especially under time pressure.

However, the experiment also shows that the improvement is not only caused by the model itself. It depends on the interaction between the student, the prompt, the input method, and the ability to transfer the generated answer correctly. The final score in the LLM-assisted phase should therefore be understood as a human–AI performance score, not as a pure model score.

One of the most important findings of the real-life experiment is that students mostly used LLMs in a simple and pragmatic way. Most students used ChatGPT, uploaded photos of the exam sheets, and used the original question as the prompt. Only a few students wrote their own prompts or reformulated the task. This differs from controlled benchmarking, where every prompt is standardized,

and from the reformulation experiment, where prompts are intentionally designed to test robustness. In real usage, students usually choose the fastest and easiest method. This finding is important for educational practice because real student behavior is not necessarily based on advanced prompt engineering. Students may rely on direct photo input and accept the answer with limited verification, especially when time is limited.

The real-life experiment also shows that LLMs are particularly useful for syntax-heavy and structure-heavy tasks. Several students struggled in the human-only phase because they did not remember SQL syntax, XML validation details, or compression representations. With LLM support, many of these answers improved. This suggests that LLMs often function as syntax and structure assistants. They help students produce formal answers that they may understand conceptually but cannot write correctly under exam pressure. This does not mean that LLMs simply replace knowledge. In some cases, they compensate for missing syntax recall or provide a structured starting point. However, this also creates a risk: students may submit answers that they cannot fully explain or verify.

A particularly important observation from Chapter 5 is the student who uploaded lecture material before asking the LLM to answer the questions. The generated answers were much closer to the professor's model solutions, especially in the SQL part. This finding shows that context can strongly improve LLM performance. Generic LLM answers may be correct in principle, but still differ from the exact notation and expectations of a specific course. When lecture material is added, the model becomes more aligned with the course-specific solution style. This connects to the broader question-answering literature, where retrieval and context are important for producing accurate answers [28]. It also shows that future assessments must consider not only whether students can access an LLM, but also whether they can provide it with lecture slides, scripts, examples, or previous solutions from the course.

Chapter 6 tested the robustness of LLMs through reformulated questions. This chapter showed that reformulation can strongly reduce LLM performance, especially for models that are already sensitive to task structure. Claude remained relatively robust with 56 out of 68 points, corresponding to 82.4 %. Kimi dropped to 28.5 out of 68 points, corresponding to 41.9 %, and Gemini reached only 22 out of 68 points, corresponding to 32.4 %. This means that reformulation affected the models differently. Claude remained strong, while Kimi and Gemini became much less reliable. This confirms that question formulation is not only a superficial detail; it can fundamentally change answer quality.

The reformulated questions were not designed to be unfair for human students. They were designed to demand stricter output, complete enumeration, exact rule localization, explicit intermediate steps, or applied execution. These are legitimate academic requirements. The fact that they reduced LLM performance shows that many model answers depend on the surface structure of the prompt. This aligns with prompt sensitivity research, which shows that controlled changes in wording, structure, and framing can influence LLM behavior [21, 22]. It also aligns with the CheckList approach, which argues that model evaluation should include behavioral tests and input variations rather than relying only on average accuracy [20].

The most effective reformulation strategies were those that forced the model to execute a method instead of only explaining it. This was visible in the compression question, where models had to select a method and apply it to concrete data. Claude performed well, while Gemini and Kimi often remained incomplete or stayed at the explanation level. This distinction between explaining and executing is one of the most important findings of the thesis. LLMs are often very good at describing a method in

natural language, but applying the method precisely to a given dataset is more difficult. This also appears in formal logic, itemset mining, SQL construction, and XML validation. The model may know what the task is about, but still fail to produce the complete formal output.

The reformulation results also show that incompleteness is one of the most serious weaknesses of LLMs in exam settings. Gemini frequently answered only selected parts of the task and omitted other required sections. Kimi sometimes treated visible information as missing, especially when the layout or document structure was complex. Claude was more complete overall, but still missed small formal details such as invalid XML attributes or exact method restrictions. This supports the argument made by HELM that LLMs should not be evaluated only by accuracy, but also by robustness, reliability, and task-specific behavior [16]. In an exam context, a partially correct answer may still lose many points if it omits required components.

To synthesize the observed errors across the benchmark and reformulation experiments, the main failure patterns can be grouped into a taxonomy of LLM failure types. This taxonomy helps distinguish between different forms of incorrect behavior instead of treating all wrong answers as the same type of error.

Table 7.2: Taxonomy of LLM failure types observed in database examination tasks

| **Failure type** | **Description** | **Example observed in this thesis** | **Main affected model** |
|---|---|---|---|
| Omission of subtasks | The model answers only part of a multi-part question and ignores required sections. | Gemini omitted parts of the formal logic task and did not provide all required intermediate outputs. | Gemini |
| Task drift | The model answers a related topic instead of the exact task. | Kimi produced semantically related answers that did not match the requested database task. | Kimi |
| Formal mismatch | The model understands the general idea but uses the wrong required representation. | Claude produced a UML-style diagram instead of the expected ontology-style representation. | Claude |
| Hallucinated or misread data | The model reconstructs, changes, or invents input data instead of using the provided data exactly. | Gemini generated SQL results based on incorrect relation data. | Gemini |
| Overgeneration | The model provides unnecessary alternatives or explanations despite strict output requirements. | Claude proposed multiple compression methods before selecting one. | Claude |
| Layout or document-interpretation failure | The model treats visible information as missing because of document structure or visual complexity. | Kimi assumed that required data was missing in some reformulated tasks. | Kimi |
| Incomplete execution | The model describes the correct method but does not fully apply it to the given data. | Gemini and Kimi often explained compression methods without producing a complete compressed representation. | Gemini and Kimi |

This taxonomy shows that LLM errors in database examination tasks are not limited to factual mistakes. Many failures are caused by missing subtasks, weak formal alignment, incomplete execution, or sensitivity to document structure. This supports the central claim of the thesis that LLM performance must be evaluated not only by final correctness, but also by completeness, formal precision, and robustness under changed task conditions.

## 7.1 Implications for AI-Aware Database Examination Design

The comparison between Chapter 4, Chapter 5, and Chapter 6 leads to a more nuanced understanding of LLMs in education. Chapter 4 shows what the models can do under controlled conditions. Chapter 5 shows how students actually use them. Chapter 6 shows how fragile the models can become when questions are reformulated.

Together, these chapters demonstrate that LLM performance is not a fixed property of the model. It emerges from the interaction between model capability, prompt formulation, input modality, user behavior, available context, and task type.

The results also show that benchmarking alone is not enough. In Chapter 4, Claude was the strongest model, but in Chapter 5, most students used ChatGPT. This means that the best model in a controlled benchmark is not necessarily the model most used in practice. Real adoption depends on accessibility, habit, interface design, and user trust. Therefore, educational institutions should not only ask which model performs best, but also which tools students actually use and how they use them.

At the same time, the student experiment shows that LLMs can improve performance even when students use simple prompts. This is important because it means that even without advanced prompt engineering knowledge, students can gain substantial support. From an assessment perspective, this makes traditional take-home or unsupervised exam formats vulnerable. If a student can photograph an exam question and receive a structured answer within a short time, then exams that rely mainly on standard textbook-style questions may no longer measure independent student performance reliably.

However, the reformulation experiment shows that exam design can still influence LLM effectiveness. Questions that require complete reasoning, exact formal output, justification of uncertainty, or application to novel data are more resistant than simple factual or standard pattern questions. This supports the recommendations of Larsen, who argues that AI-resistant exams should focus on contextual reasoning, application, and precise task design rather than memorized textbook knowledge [23]. It also agrees with Stoyanov and Nedelcheva, who found that LLM-resistant questions are often those requiring interpretation, application, and domain-specific reasoning rather than direct factual recall [25].

The results of this thesis support these findings in the specific domain of database examinations.

Based on the empirical findings, several practical recommendations can be derived for AI-aware database examination design.

Table 7.3: Recommendations for AI-aware database examination design

| Recommendation | Reason based on the thesis results |
|---|---|
| Require intermediate reasoning steps. | LLMs may generate plausible final answers without showing valid reasoning. Requiring steps makes omissions and mistakes easier to detect. |
| Use course-specific notation and examples. | Generic LLM answers may be correct in principle but may not match the professor's expected solution style. |
| Ask students to justify modeling decisions. | ER modeling and schema design require decisions that go beyond pattern reproduction. |
| Require exact output formats. | Reformulated questions showed that strict format requirements expose weaknesses in instruction following. |
| Use novel datasets or modified examples. | LLMs perform better on familiar patterns than on new structured data that must be interpreted carefully. |
| Include error-detection tasks. | XML validation and formal checking tasks showed that small errors are often missed by LLMs. |
| Ask students to critique an LLM-generated answer. | This turns LLM use into a critical reasoning task instead of simple copying. |
| Combine written answers with short oral explanations. | Oral follow-up questions can help verify whether students understand the submitted solution. |
| Assess both final answers and reasoning quality. | Several LLM answers were partially correct but incomplete or based on weak reasoning. |

These recommendations show that AI-aware assessment does not require banning LLMs completely. Instead, examination design should shift from reproducing standard answers toward explaining, verifying, applying, and defending solutions. In database education, this means that students should not only write SQL queries or draw models, but also justify why their solution is correct, identify possible errors, and explain the consequences of design decisions.

## 7.2 Broader Educational Implications

The results show that AI-assisted learning and AI-assisted cheating are technically similar but pedagogically different. The same tool can help a student understand SQL syntax during preparation, but it can also generate answers during an exam. Therefore, the main question is not whether LLMs are good or bad, but how they are integrated into learning and assessment. A complete ban may be unrealistic, but unrestricted use can undermine assessment validity. A more balanced approach is needed. Students should be taught how to use LLMs critically, verify outputs, and understand limitations. At the same time, assessments should be redesigned to measure reasoning, explanation, oral defense, practical application, and individual understanding.

Another important conclusion is that multimodal input changes the practical risk of LLM use in exams. In Chapter 4, photo input was much faster than manual typing. In Chapter 5, most students used

photos. This means that image input is not a marginal feature; it is central to how students actually interact with LLMs in technical subjects. Database questions often contain tables and diagrams, which are difficult to type manually. If a model can process images, the barrier to using it during an exam-like situation becomes much lower. This has consequences for exam security and assessment design. Teachers should assume that students may not only copy text into an LLM, but also upload full pages, tables, diagrams, or lecture material.

The findings also suggest that evaluation rubrics should distinguish between correct final answers and correct reasoning processes. In several cases, LLMs generated a plausible final result but used incomplete or incorrect intermediate reasoning. In other cases, the final answer was partially correct but missed required elements. Therefore, grading should reward not only the final output, but also the explanation, intermediate steps, and alignment with the required method. This is particularly important in technical domains such as databases, where the process is often as important as the final result.

## 7.3 Threats to Validity

Like any empirical study, this thesis is subject to several threats to validity. These limitations do not invalidate the findings, but they define the scope within which the results should be interpreted.

One threat to internal validity concerns the manual evaluation of LLM-generated answers. Although the grading was based on the professor's reference solutions and predefined point schemes, partial answers sometimes required judgment. To reduce this risk, the same grading logic was applied consistently across models and questions. A further internal validity concern is that commercial LLMs can produce slightly different answers across repeated interactions. This was partially addressed by using separate chat sessions and evaluating the first generated answer for each prompt.

The results are based on selected database and data warehouse examination questions from one university context. Therefore, the findings should not be generalized to all database courses, all examination formats, or all academic disciplines without caution. The student experiment also involved only ten participants, which limits the generalizability of the observed user behavior. Nevertheless, the experiment provides useful exploratory evidence because it reflects realistic student interaction with LLMs under time pressure.

The thesis uses correctness scores as the main quantitative measure of model performance. However, correctness alone does not fully capture reasoning quality, robustness, or educational usefulness. For this reason, the analysis also considered completeness, formal precision, instruction following, response consistency, and failure patterns. In the student experiment, the LLM-assisted score should not be interpreted as a pure model score, but as a combined human–AI performance score influenced by prompting behavior, input modality, and the student's ability to verify the generated answer.

The conclusions are based on a limited number of models, questions, reformulations, and student participants. In addition, LLM technology evolves rapidly, meaning that future model versions may perform differently. Therefore, the numerical results should be interpreted as a snapshot of the tested models under the tested conditions. The broader conclusions, however, remain relevant because they concern structural properties of LLM-based assessment: task dependence, prompt sensitivity, context dependence, and the need for AI-aware examination design.

Overall, the thesis demonstrates that LLMs are already capable of solving many database examination questions, but they are not consistently reliable across all formats and conditions. Claude showed the strongest controlled performance, ChatGPT dominated real student usage, and reformulation exposed major weaknesses, especially in Gemini and Kimi. The most important conclusion is that LLM performance depends on the task, the prompt, the input format, the language, the available context, and the user. For education, this means that LLMs should neither be ignored nor blindly trusted. They should be treated as powerful but imperfect tools that require critical use, careful evaluation, and adapted assessment design.

The discussion of all empirical chapters therefore supports the main research questions of this thesis. LLMs can answer many database examination questions correctly, but their reliability varies strongly by model and task type. They are strongest on familiar and structured patterns, but weaker on formal precision, completeness, and robust reasoning. Reformulation can expose these weaknesses, especially when it demands exact output, complete enumeration, or applied execution. In real student usage, LLMs can strongly improve performance, but the benefit depends on prompting behavior, input modality, and available context. These findings show that the future of assessment in database education should not be based on simple prohibition, but on a deeper understanding of how LLMs work, where they fail, and how exams can be designed to remain meaningful in the presence of generative AI.

# Chapter 8

## *Conclusion and Future Work*

This thesis investigated the capabilities and limitations of Large Language Models in the context of Database and Data Warehouses examination questions. The objective was not only to measure how well modern LLMs can answer exam questions, but also to understand how reliable these systems are when question formulations change and how they are used by students in realistic educational settings.
The first part of the thesis evaluated several state-of-the-art LLMs on a set of database examination questions covering different topics, including multidimensional modeling, XML validation, SQL, data compression, ontology modeling, and differential snapshots. The results showed that LLMs are capable of solving many database-related tasks with a high degree of correctness.
However, important differences between models were observed. Some models demonstrated stronger reasoning abilities and higher completeness, while others were more prone to omissions, misunderstandings, or formal errors. The findings showed that overall correctness alone is not sufficient to evaluate an LLM. Completeness, consistency, and the ability to satisfy all requirements of a task are equally important.
The second part of the thesis focused on question reformulation. The experiments demonstrated that even logically equivalent questions can lead to significantly different model outputs when the wording, structure, or constraints are modified. Reformulated questions exposed weaknesses that were often hidden in the original benchmark. The most common failure patterns included omission of subtasks, hallucination of information, incorrect interpretation of document structures, and failure to follow strict instructions. These findings confirm that LLM performance is highly sensitive to prompt formulation and that benchmark results alone may overestimate the true robustness of a model.
The third part of the thesis examined how students use LLMs in practice. The real-life experiment showed that LLM assistance substantially improved student performance. Students who initially achieved relatively low scores were often able to solve a much larger proportion of the tasks when supported by an LLM. At the same time, the experiment revealed that the effectiveness of LLM support depends not only on the model itself but also on the way students interact with it. Providing additional context, lecture material, screenshots, or carefully formulated prompts often led to better results. This highlights the growing importance of prompt literacy and critical evaluation skills in modern education.
The results of this thesis allow the research questions formulated in Chapter 1 to be answered directly.

**RQ1: How do LLMs generate answers to exam questions, and which LLMs provide the most accurate and reliable responses?**

The benchmarking experiments showed that LLMs are capable of generating correct and well-structured answers for many database examination tasks. However, their performance differs considerably depending on the model and the type of question. Among the evaluated models, Claude achieved the highest overall correctness and demonstrated the most consistent performance across different topics and question formats. Gemini and Kimi also produced correct answers in several cases but showed greater variability and were more likely to omit important information or provide incomplete solutions. The results therefore indicate that LLMs can successfully answer many exam questions, but their reliability depends on both the selected model and the characteristics of the task.

**RQ2: How can exam questions be reformulated in order to cause LLMs to generate incorrect or partially incorrect answers? What are the main weaknesses and limitations of generative AI when answering exam-style questions?**

The reformulation experiments demonstrated that LLM performance can be significantly affected even when the semantic meaning of a question remains unchanged. Reformulations that introduced additional constraints, required exhaustive completeness, changed the order of information, demanded precise output formats, or increased reasoning complexity frequently reduced performance. The most common weaknesses observed were omitted subtasks, hallucinated information, incomplete reasoning, difficulties following strict instructions, and sensitivity to changes in wording and context. These findings show that strong performance on standard examination questions does not necessarily imply robustness or deep understanding of the underlying concepts.

Overall, the answers to both research questions demonstrate that LLMs are powerful educational tools that can support learning and problem solving in database-related domains. Nevertheless, their performance remains highly dependent on prompt formulation, task design, contextual information, and user interaction. Consequently, the results highlight the importance of developing AI-aware assessment methods that evaluate not only final answers but also reasoning quality, robustness, and the ability to apply knowledge in unfamiliar situations.

For higher education, these findings suggest that the discussion should move beyond the simple question of whether LLMs should be allowed or prohibited. Instead, educators should focus on understanding where these systems provide genuine support, where they fail, and how assessments can be adapted accordingly. Examination tasks that require deeper reasoning, application of knowledge, justification of solutions, and interpretation of results are likely to remain more resistant to straightforward LLM usage than tasks based primarily on pattern recognition or memorization.

## Future Work

Although this thesis provides a comprehensive evaluation of LLM performance on Database examination questions, several directions remain open for future research. First, future studies could investigate newer generations of LLMs as the technology continues to evolve rapidly. Second, a larger-scale student experiment involving more participants and multiple universities could provide more generalizable insights into real-world educational usage. Third, future research could examine multimodal scenarios in greater detail, including the use of screenshots, lecture slides, diagrams, and uploaded documents.

Finally, additional work is needed to develop systematic methodologies for designing robust and AI-aware examinations that remain meaningful in an educational environment increasingly influenced by generative artificial intelligence.

In conclusion, this thesis demonstrates that LLMs have already become highly capable assistants for solving database examination questions. Nevertheless, their limitations remain significant and must not be overlooked. Understanding both the strengths and weaknesses of these systems is essential for students, educators, and researchers alike. As LLMs continue to advance, the challenge will not be to compete with them, but to learn how to use them responsibly, critically, and effectively within the educational process.

# Bibliography


[1] Abdulhadi Shoufan. **Exploring Students' Perceptions of ChatGPT: Thematic Analysis and Follow-Up Survey**. **in** *IEEE Access* 11 (2023), **pages** 38805–38818. DOI: 10.1109/ACCESS.2023.3268224.

[2] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major **and** Shmargaret Shmitchell. **On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? in** *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '21. Virtual Event, Canada: Association for Computing Machinery, 2021, **pages** 610–623. ISBN: 9781450383097. DOI: 10.1145/3442188.3445922. URL: https://doi.org/10.1145/3442188.3445922.

[3] Peter Clark **and** Oren Etzioni. **My Computer Is an Honor Student - but How Intelligent Is It? Standardized Tests as a Measure of AI**. **in** *AI Mag.* 37 (2016), **pages** 5–12. URL: https://api.semanticscholar.org/CorpusID:16968840.

[4] OpenAI **andothers**. **GPT-4 Technical Report**. **in** (2024). arXiv: 2303.08774 [cs.CL]. URL: https://arxiv.org/abs/2303.08774.

[5] Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui **and** Yejin Choi. **Faith and Fate: Limits of Transformers on Compositionality**. **in** (2023). arXiv: 2305.18654 [cs.CL]. URL: https://arxiv.org/abs/2305.18654.

[6] Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan **and** Subbarao Kambhampati. **PlanBench: An Extensible Benchmark for Evaluating Large Language Models on Planning and Reasoning about Change**. **in** (2023). arXiv: 2206.10498 [cs.CL]. URL: https://arxiv.org/abs/2206.10498.

[7] Mohammed Ahmed Sadeq, Reem Mohamed Farouk Ghorab, Mohamed Hady Ashry, Ahmed Mohamed Abozaid, Haneen A Banihani, Moustafa Salem, Mohammed Tawfiq Abu Aisheh, Saad Abuzahra, Marina Ramzy Mourid, Mohamad Monif Assker **andothers**. **AI chatbots show promise but limitations on UK medical exam questions: a comparative performance study**. **in** *Scientific reports* 14.1 (2024), **page** 18859.

[8] Yogesh K. Dwivedi, Nir Kshetri, Laurie Hughes, Emma Louise Slade, Anand Jeyaraj, Arpan Kumar Kar, Abdullah M. Baabdullah, Alex Koohang, Vishnupriya Raghavan, Manju Ahuja, Hanaa Albanna, Mousa Ahmad Albashrawi, Adil S. Al-Busaidi, Janarthanan Balakrishnan,

Yves Barlette, Sriparna Basu, Indranil Bose, Laurence Brooks, Dimitrios Buhalis, Lemuria Carter, Soumyadeb Chowdhury, Tom Crick, Scott W. Cunningham, Gareth H. Davies, Robert M. Davison, Rahul Dé, Denis Dennehy, Yanqing Duan, Rameshwar Dubey, Rohita Dwivedi, John S. Edwards, Carlos Flavián, Robin Gauld, Varun Grover, Mei-Chih Hu, Marijn Janssen, Paul Jones, Iris Junglas, Sangeeta Khorana, Sascha Kraus, Kai R. Larsen, Paul Latreille, Sven Laumer, F. Tegwen Malik, Abbas Mardani, Marcello Mariani, Sunil Mithas, Emmanuel Mogaji, Jeretta Horn Nord, Siobhan O'Connor, Fevzi Okumus, Margherita Pagani, Neeraj Pandey, Savvas Papagiannidis, Ilias O. Pappas, Nishith Pathak, Jan Pries-Heje, Ramakrishnan Raman, Nripendra P. Rana, Sven-Volker Rehm, Samuel Ribeiro-Navarrete, Alexander Richter, Frantz Rowe, Suprateek Sarker, Bernd Carsten Stahl, Manoj Kumar Tiwari, Wil van der Aalst, Viswanath Venkatesh, Giampaolo Viglia, Michael Wade, Paul Walton, Jochen Wirtz **and** Ryan Wright. **Opinion Paper: "So what if ChatGPT wrote it?" Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy**. **in** *International Journal of Information Management* 71 (2023), **page** 102642. ISSN: 0268-4012. DOI: https://doi.org/10.1016/j.ijinfomgt.2023.102642. URL: https://www.sciencedirect.com/science/article/pii/S0268401223000233.

[9] Yash Raj Shrestha, Vamsi Krishna **and** Georg von Krogh. **Generative AI Systems: Capabilities, Limitations, and Implications**. **in** *IEEE Engineering Management Review* 52.1 (2024), **pages** 14–26. DOI: 10.1109/EMR.2024.3369093.

[10] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Sohel Mondal **and** Aman Chadha. **A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications**. **in** *ArXiv* abs/2402.07927 (2024). URL: https://api.semanticscholar.org/CorpusID:267636769.

[11] Lee Boonstra. **Prompt Engineering**. Whitepaper (PDF). URL: https://cdn.jsdelivr.net/gh/abncharts/abncharts.public.1/abnasia.org/1744390948632_www.abnasia.org.pdf.

[12] Tongshuang Wu, Michael Terry **and** Carrie Jun Cai. **AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts**. **in**. CHI '22. New Orleans, LA, USA: Association for Computing Machinery, 2022. ISBN: 9781450391573. DOI: 10.1145/3491102.3517582. URL: https://doi.org/10.1145/3491102.3517582.

[13] Jason Wei andXuezhi Wang andDale Schuurmans andMaarten Bosma andEd H. Chi andQuoc Le andDenny Zhou. **Chain of Thought Prompting Elicits Reasoning in Large Language Models**. **in**. **volume** abs/2201.11903. 2022. arXiv: 2201.11903. URL: https://arxiv.org/abs/2201.11903.

[14] A. Gao. **Prompt Engineering for Large Language Models**. **in** *SSRN Electronic Journal* (2023). DOI: 10.2139/ssrn.4504303. URL: https://ssrn.com/abstract=4504303.

[15] Artificial Analysis. **LLM Benchmarking Platform**. 2025. URL: https://artificialanalysis.ai.

[16] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng,

Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang **and** Yuta Koreeda. **Holistic Evaluation of Language Models**. **in** (2023). arXiv: 2211.09110 [cs.CL]. URL: https://arxiv.org/abs/2211.09110.

[17] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang **and** Xing Xie. **A Survey on Evaluation of Large Language Models**. **in** (2023). arXiv: 2307.03109 [cs.CL]. URL: https://arxiv.org/abs/2307.03109.

[18] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew Dai, Andrew La, Andrew Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, César Ferri Ramírez, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Chris Waites, Christian Voigt, Christopher D. Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, Daniel Freeman **and** Daniel Khashabi. **Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models**. **in** (2023). arXiv: 2206.04615 [cs.CL]. URL: https://arxiv.org/abs/2206.04615.

[19] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou **and** Jason Wei. **Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them**. **in** (2022). arXiv: 2210.09261 [cs.CL]. URL: https://arxiv.org/abs/2210.09261.

[20] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin **and** Sameer Singh. **Beyond Accuracy: Behavioral Testing of NLP models with CheckList**. **in**. 2020. arXiv: 2005.04118 [cs.CL]. URL: https://arxiv.org/abs/2005.04118.

[21] Amirhossein Razavi, Mina Soltangheis, Negar Arabzadeh, Sara Salamat, Morteza Zihayat **and** Ebrahim Bagheri. **Benchmarking Prompt Sensitivity in Large Language Models**. **in** (2025). arXiv: 2502.06065 [cs.CL]. URL: https://arxiv.org/abs/2502.06065.

[22] Ruiyang Qin, Qingzhuo Wang, Tian Wang, Zhihua Wei **and** Wen Shen. **Evaluating and Explaining Prompt Sensitivity of LLMs Using Interactions**. **in** (2026). URL: https://openreview.net/forum?id=6fHZR6uxNa.

[23] Simon kaare Larsen. **Creating Large Language Model Resistant Exams: Guidelines and Strategies**. **in** (2023). arXiv: 2304.12203 [cs.CL]. URL: https://arxiv.org/abs/2304.12203.

[24] Robert Osazuwa Ness, Katie Matton, Hayden Helm, Sheng Zhang, Junaid Bajwa, Carey E. Priebe **and** Eric Horvitz. **MedFuzz: Exploring the Robustness of Large Language Models in Medical Question Answering**. **in** (2024). arXiv: 2406.06573 [cs.CL]. URL: https://arxiv.org/abs/2406.06573.

[25] Asen Stoyanov **and** Anely Nedelcheva. **Identifying Features of LLM-Resistant Exam Questions: Insights from Artificial Intelligence (AI)–Student Performance Comparisons**. **in** *Sci* 7.4 (2025). ISSN: 2413-4155. DOI: 10.3390/sci7040183. URL: https://www.mdpi.com/2413-4155/7/4/183.

[26] Bradley McDanel **and** Ed Novak. **Designing LLM-Resistant Programming Assignments: Insights and Strategies for CS Educators**. **in** *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*. SIGCSETS 2025. Pittsburgh, PA, USA: Association for Computing Machinery, 2025, **pages** 756–762. ISBN: 9798400705311. DOI: 10.1145/3641554.3701872. URL: https://doi.org/10.1145/3641554.3701872.

[27] Karthik Valmeekam, Sarath Sreedharan, Matthew Marquez, Alberto Olmo **and** Subbarao Kambhampati. **On the Planning Abilities of Large Language Models (A Critical Investigation with a Proposed Benchmark)**. **in**. 2023. arXiv: 2302.06706 [cs.AI]. URL: https://arxiv.org/abs/2302.06706.

[28] Arindam Bhattacharya. **A Survey of Question Answering for Math and Science Problem**. **in** (2017). arXiv: 1705.04530 [cs.AI]. URL: https://arxiv.org/abs/1705.04530.

# Appendix A

## Appendix

### Database Examination Material and Reference Solutions

### Question 1:

(a) Erstellen Sie eine Ontologie, welches den Besuch von Vorlesungen durch Studierende darstellt. Wählen Sie geeignete Klassen (mind. 4), charakterisieren Sie diese durch Eigenschaften und stellen Sie Beziehungen zwischen den Klassen her! Stellen Sie die Ontologie grafisch dar!

**Musterlösung:**

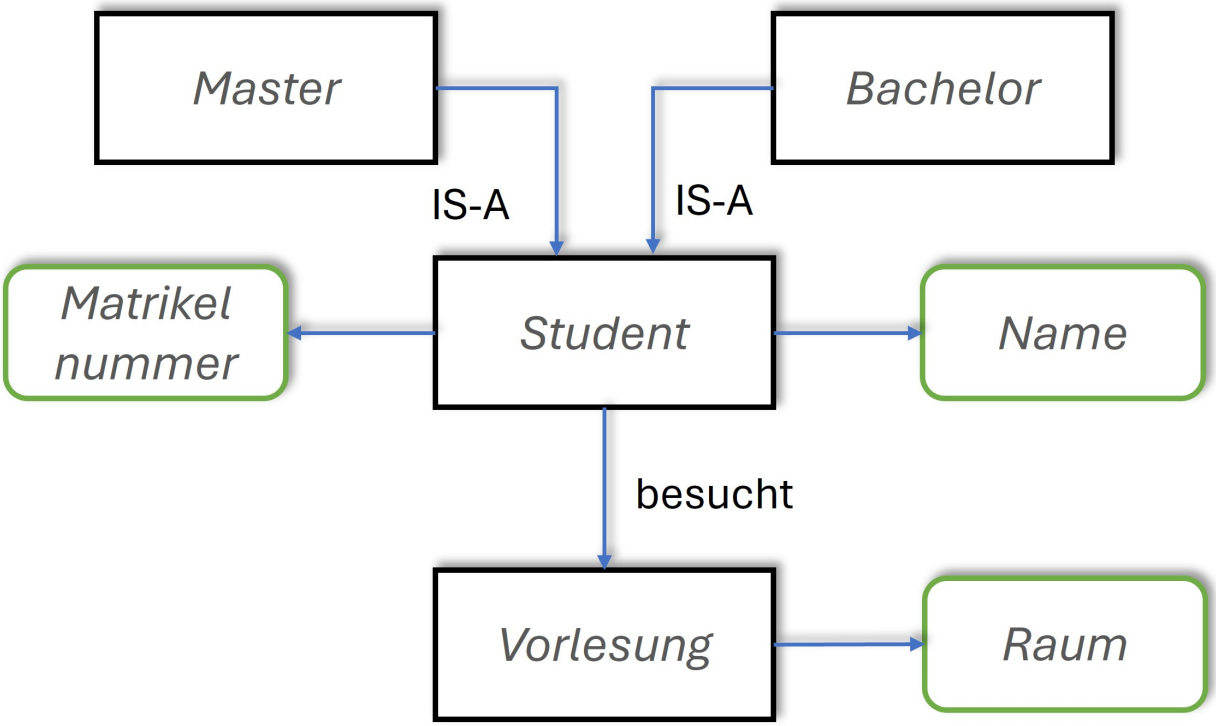



Figure A.1: Answer of Question 1(a)

(b) Überführen Sie folgende Ontologie in eine passende Darstellung in RDF, OWL oder Turtle! Namespaces können weggelassen werden.

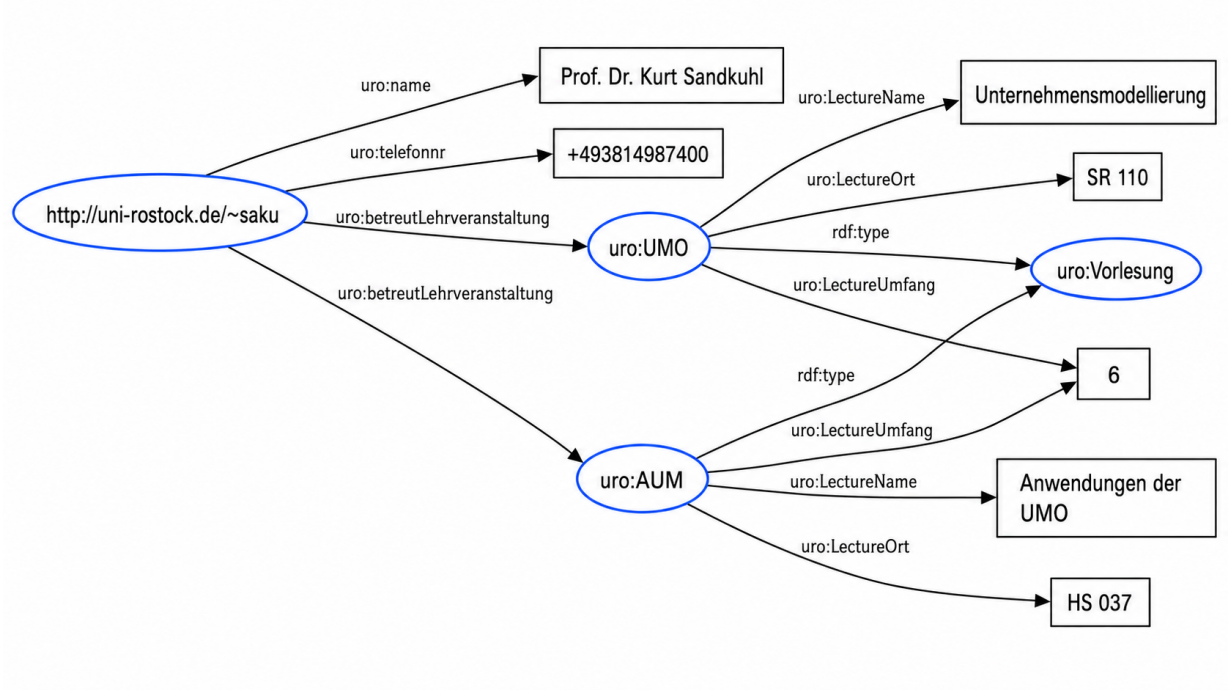



Figure A.2: Question 1b

**Musterlösung:**

```
<rdf:Description rdf:about="http://uni-rostock.de/ saku"> <uro:name>Prof.
Dr.  Kurt Sandkuhl</uro:name> <uro:telefonnr>+493814987400</uro:telefonnr>
<uro:betreutLehrveranstaltung rdf:resource="AUM"/> <uro:betreutLehrveranstaltung
rdf:resource="UMO"/> </rdf:Description>
<rdf:Description rdf:about="AUM"> <rdf:type rdf:resource="Vorlesung"/>
<uro:LectureName>Anwendungen der UMO</uro:LectureName> <uro:LectureOrt>HS
037</uro:LectureOrt> <uro:LectureUmfang>6</uro:LectureUmfang> </rdf:Description>
<rdf:Description rdf:about="UMO"> <rdf:type rdf:resource="Vorlesung"/>
<uro:LectureName>Unternehmensmodellierung</uro:LectureName> <uro:LectureOrt>SR
110</uro:LectureOrt> <uro:LectureUmfang>6</uro:LectureUmfang> </rdf:Description>
```

(c) Stellen Sie obigen Klassenbeziehungen mittels RDF-S dar!

```
<rdfs:Class rdf:ID="Vorlesung"> <rdfs:SubClassOf rdf:resource=
"http://www.w3.org/2000/01/rdf-schemaResource"/> </rdfs:Class>
<rdfs:Class rdf:ID="Person"> <rdfs:SubClassOf rdf:resource=
"http://www.w3.org/2000/01/rdf-schemaResource"/> </rdfs:Class>
<rdf:Property rdf:about="uro:BetreutLehrveranstaltung"> <rdfs:SubPropertyOf
rdf:resource="http://www.w3.org/2000/01/rdf-schemaProperty"/> <rdfs:domain
rdf:resource="Vorlesung"/> <rdfs:range rdf:resource="Person"/> </rdf:Property>
<rdf:Description>
```

## Question 2:

Kennzeichnen Sie die erfüllten (-) und nicht erfüllten (X) Eigenschaften der Modellierungsstile für multidimensionale Daten.

**Musterlösung:**

Table A.1: Characteristics of schema types

| **Schema** | **Normalisierte Fakten** | **Normalisierte Dimensionen** | **geringe Anfragezeit** |
|---|---|---|---|
| **Sternschema** | - | X | - |
| **Schneeflockenschema** | - | - | X |
| **Fullfact-Schema** | X | - | - |

## Question 3:

(a) Setzen Sie die folgenden Beschreibungen in jeweils einem Standard-ER-Modell um. Nutzen Sie die Notation der Übung und wenden Sie die Intervallnotation für die Kardinalitäten an.

**Kritiker** können **Rezensionen** schreiben. Kritiker besitzen einen Namen und eine eindeutige ID, während Rezensionen teilweise durch das betroffene Produkt charakterisiert werden. Rezensionen erhalten eine Menge von Fotos, eine Bewertung und eine Beschreibung.

**Musterlösung:**

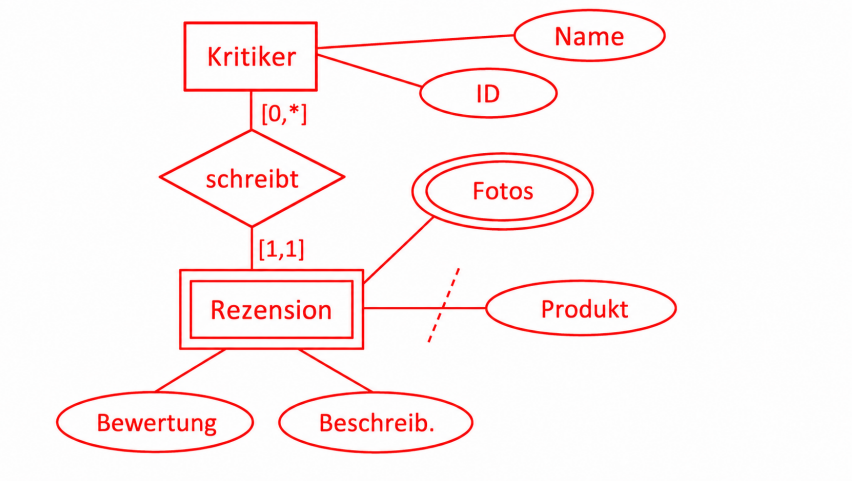



Figure A.3: Answer of Question 3a

(b) Überführen Sie die nachfolgenden ER-Modelle in das relationale Modell. Kennzeichnen Sie Primär- und Fremdschlüssel. Verschmelzen Sie, falls möglich.

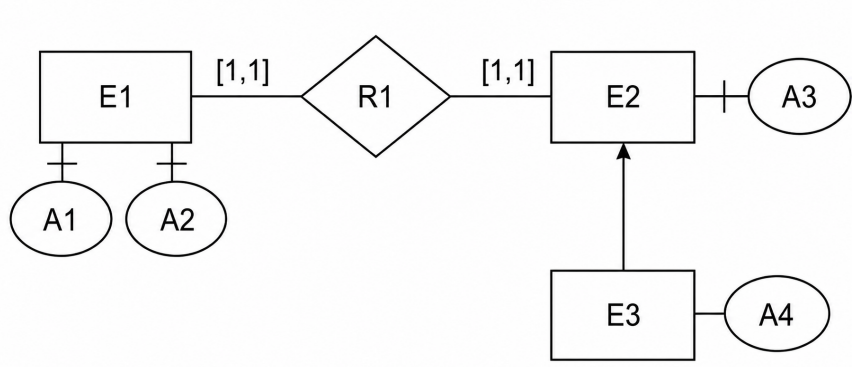



Figure A.4: Question 3b

**Musterlösung:**

E1E2({<u>A1</u>, <u>A2</u>, A3}, {{A1}, {A2}, {A3}})

E3({<u>A3</u>, A4}, {{A3}}), E3(A3) → E1E2(A3)

statt A3 auch A1 oder A2 möglich.

(c) Geben Sie die Ergebnisrelationen der gegebenen Anfragen als Relationen an.

| **A** | **B** |
|---|---|
| 1 | 4 |
| 2 | 3 |

(a) Rel1

| **B** | **C** |
|---|---|
| 4 | 5 |
| 7 | 8 |

(b) Rel2

**SELECT** *
**FROM** Rel1, Rel2
**WHERE** Rel1.B != Rel2.B

**Müsterlösung:**

| A | Rel1.B | Rel2.B | C |
|---|---|---|---|
| 1 | 4 | 7 | 8 |
| 2 | 3 | 4 | 5 |
| 2 | 3 | 7 | 8 |

**SELECT DISTINCT** A, C
**FROM** Rel1 **NATURAL JOIN** Rel2
**WHERE** A = 1

**Musterlösung:**

| A | C |
|---|---|
| 1 | 5 |

**SELECT** count(B) **AS** cnt
**FROM** Rel2
**GROUP BY** C

**Musterlösung:**

| cnt |
|---|
| 1 |
| 1 |

## Question 4:

(a) Überprüfen Sie die Gültigkeit des XML-Dokumentes bezüglich der gegebenen DTD! Markieren Sie die fehlerhaften Stellen im Dokument!

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE planung [
<!ELEMENT planung (planer, stundenplan+)>
<!ELEMENT planer EMPTY>
<!ATTLIST planer name CDATA #REQUIRED>
<!ELEMENT stundenplan (tag*)>
<!ELEMENT tag (name?, termin*)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT termin (#PCDATA | veranstaltung)*>
<!ELEMENT veranstaltung (#PCDATA)>
]>

<planung>
<stundenplan>
<tag>
<name>Montag</name>
<termin>Jour Fix</termin>
<termin>
<veranstaltung type='Uebung'>Knowledge Repr.</veranstaltung>
</termin>
</tag>
<tag>
<termin>
<veranstaltung typ='Konsultation'>Reasoning</veranstaltung>
</termin>
<name>Montag</name>
</tag>
</stundenplan>
</planung>
```

**Musterlösung:**

```
<planung>                                                    Planer fehlt
<stundenplan>
<tag>
<name>Montag</name>
<termin>Jour Fix</termin>
<termin>
  <veranstaltung type='Uebung'>Knowledge                     Veranstaltung hat kein
Repr.</veranstaltung>                                        Attribut type
</termin>
</tag>
<tag>
<termin>                                                     Termin und name vertauscht
  <veranstaltung typ='Konsultation'>Reasoning</veranstaltung>
</termin>                                                    Veranstaltung hat kein typ
<name>Montag</name>
</tag>
</stundenplan>
</planung>
```

(b) Gegeben ist eine Dokumentenkollektion bestehend aus folgenden Dokumenten:

- $D_1$ = Es ist Winter, daher ist das kalte Wetter soweit in Ordnung.
- $D_2$ = Die winterliche Landschaft ist durch das Wetter begründet. Verschneite Landschaften sehen einfach gut aus.
- $D_3$ = Das kalte Wetter ist unangenehm, auf eine andere Wetterlage müssen wir leider warten.

Bezüglich dieser Dokumente wird die folgende Anfrage gestellt:

- $Q$ = Wie sieht die Landschaft bei kaltem Wetter aus?

Es existiert eine Stoppwortliste mit folgenden Termen:

ab, andere, auf, aus, begründet, bei, daher, das, der, die, durch, ein, eine, eines, einfach, er, es, gut, in, ist, leider, müssen, schlecht, sehen, sie, sieht, soweit, unangenehm, verschneite, warten, wie, wir

Für die Stammwortreduktion gelten folgende Regeln:

1. kalte → kalt
2. kaltem → kalt
3. Wetterlage → Wetter
4. winterliche → Winter
5. Landschaften → Landschaft

Bestimmen Sie die *Termmenge* (Vokabular) nach Anwendung der *Stoppwortliste* und *Stammwortreduktion* für alle Dokumente und die Anfrage Q!

**Musterlösung:**

$D_1$: Winter, kalt, Wetter, Ordnung

$D_2$: Winter, Landschaft, Wetter, Landschaft

$D_3$: Kalt, Wetter, Wetter

$Q$: Landschaft, kalt, Wetter

**(c)** Umranden Sie in der Abbildungen die Datenpunkte so, dass zwei disjunkte, unvollständige und konkave Cluster entstehen!

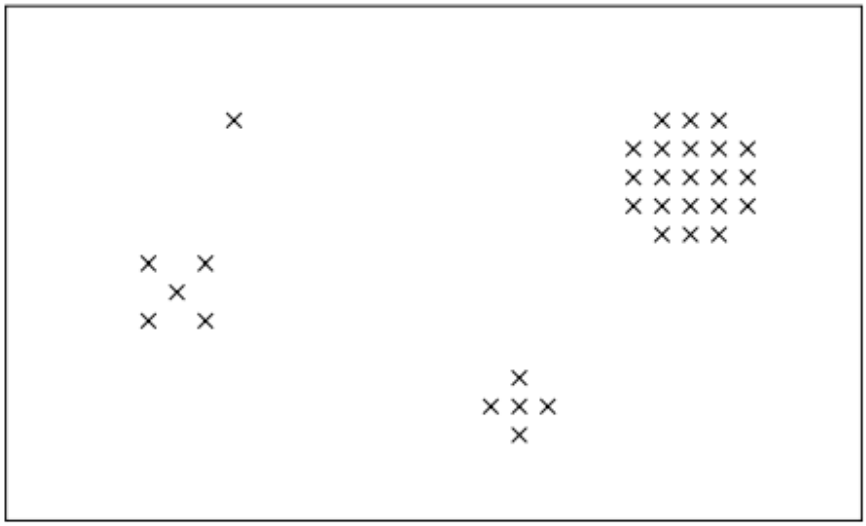


Figure A.5: Question 4c

**Musterlösung:**

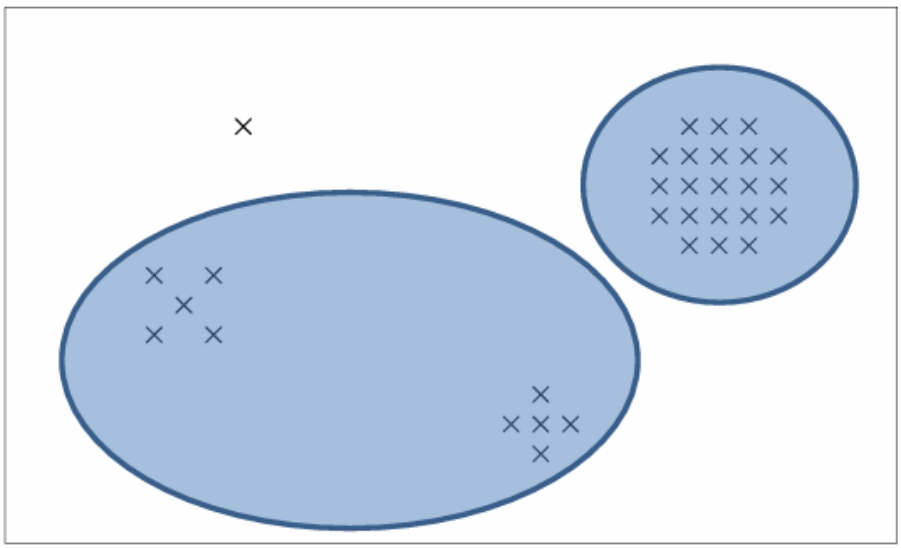


Figure A.6: Answer: Question 4c

## Question 5:

(a) Bestimmen Sie die *Wahrheitswerte* für folgende aussagenlogische Formel und tragen Sie die Ergebnisse in der *Tabelle* ein!

$$\neg X \vee (X \rightarrow (Y \leftrightarrow Z))$$

**Musterlösung:**

| **X** | **Y** | **Z** | $\neg X \vee (X \rightarrow (Y \leftrightarrow Z))$ |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

**Platz für Zwischenergebnisse**

| $Y \leftrightarrow Z$ | $X \rightarrow (Y \leftrightarrow Z)$ | |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| 1 | 1 | |
| 0 | 0 | |
| 0 | 0 | |
| 1 | 1 | |

(b) Bestimmen Sie unter Ausnutzung logischer Äquivalenzen, ob die Formel $(p \wedge q) \rightarrow p$ eine Tautologie ist.

**Musterlösung:**

$\neg(P \wedge Q) \vee P$ (Def. $\rightarrow$)

$\neg P \vee \neg Q \vee P$ (De Morgan)

$TRUE \vee \neg Q \Rightarrow TRUE$

(c) Beweisen Sie mittels Äquivalenzumformung, dass die Formel $p \rightarrow (q \rightarrow r)$ äquivalent zu der Formel $(p \wedge q) \rightarrow r$ ist.

**Musterlösung:**

$\neg P \vee (\neg Q \vee R)$ (Def. $\rightarrow$)

$\neg(P \wedge Q) \vee R$ (De Morgan)

$(P \wedge Q) \rightarrow R$ (Def. $\rightarrow$)

(d) Gegeben seien folgende Aussagen: „Alle Menschen sind sterblich." und „Sokrates ist ein Mensch." Stellen Sie die Aussagen in der Prädikatenlogik dar.

**Musterlösung:**

$\forall x(M(x) \rightarrow S(x))$

$M(sokrates)$

## Question 6:

(a) Ein schneller Bluttest soll bei Patient:innen feststellen, ob sie an einer bestimmten Krankheit (K) leiden. Die folgenden Informationen liegen vor:

- Die Grundwahrscheinlichkeit dafür, dass eine beliebig ausgewählte Person in der betreffenden Population diese Krankheit hat, beträgt $P(K) = 0{,}02$.
- Die Trefferquote (Sensitivität) des Tests beträgt $P(positivesTestergebnis \mid K) = 0{,}99$.
- Die falsche Alarmquote beträgt $P(positivesTestergebnis \mid \neg K) = 0{,}05$.

Verwenden Sie den Satz von Bayes, um die Wahrscheinlichkeit dafür zu berechnen, dass eine Person tatsächlich krank ist, wenn ihr Testergebnis positiv ausgefallen ist.

**Musterlösung:**

**Schritt 1:** $P(positivesTestergebnis)$
$= (0{,}99) \cdot (0{,}02) + (0{,}05) \cdot (0{,}98) = 0{,}0198 + 0{,}049 = 0{,}0688$
also 6,88 %

**Schritt 2:** $P(K \mid positivesTestergebnis)$
$= (0{,}99) \cdot (0{,}02)0{,}0688 = 0{,}01980{,}0688 \approx 0{,}288$
also ungefähr 28,8 %

**Interpretation:** Obwohl das Testergebnis positiv ausfällt, liegt die Wahrscheinlichkeit, tatsächlich krank zu sein, nur bei ca. 28,8 %.

(b) Die folgenden Transaktionen eines Geschäfts sind gegeben. Die Namen in der Übersicht sind die angebotenen Produkte. Jeder Transaktion gibt an, welche Produkte ein Kunde gekauft hat.

Table A.2: Transaction data

| ID | Transactions |
|---|---|
| 1 | Towel, Guide, Bathrobe, Toothbrush |
| 2 | Bathrobe, Toothbrush |
| 3 | Towel, Guide |
| 4 | Bathrobe, Lyrics |
| 5 | Towel, Guide, Bathrobe, Lyrics, Toothbrush |

Bestimmen Sie **alle häufigen Itemsets** mit einem oder zwei Items und deren support-Wert größergleich dem **min-support von 2** ist.

**Musterlösung:**

Towel 3x  Guide 3x  Bathrobe 4x  Toothbrush 3x  Lyrics 2x

{Towel, Guide} 3x  {Towel, Bathrobe} 2x  {Towel, Toothbrush} 2x

{Guide, Bathrobe} 2x  {Guide, Toothbrush} 2x  {Bathrobe, Toothbrush} 3x

{Lyrics, Bathrobe} 2x

(c) Welche allgemeinen Assoziationsregeln lassen sich aus dem Itemset {Towel, Guide, Bathrobe} ableiten?

**Musterlösung:**

Towel → Guide, Bathrobe  Guide → Towel, Bathrobe

Bathrobe → Towel, Guide  Towel, Guide → Bathrobe

Towel, Bathrobe → Guide  Bathrobe, Guide → Towel

## Question 7:

a) Geben Sie bitte die SQL-Anfrage an, die die Ergebnistabelle erzeugt, die auf der rechten Seite angegeben ist. Mit der ersten Anfrage soll die Anzahl aller Teilnehmer in allen Läufen ermittelt werden.

**Location**

| CityID | City | Region | Country |
|---|---|---|---|
| C1 | Rostock | MV | Germany |
| C2 | Schwerin | MV | Germany |
| C3 | Leipzig | Sachsen | Germany |
| C4 | Dresden | Sachsen | Germany |
| C5 | Prague | Czechy | Czech Republic |

**SportsDiscipline**

| DisciplineID | Distance | km |
|---|---|---|
| D1 | Marathon | 42 |
| D2 | Half marathon | 21 |

**Participants**

| DisciplineID | CityID | Date | NumberOfPart |
|---|---|---|---|
| D1 | C1 | 2020-03-10 | 280 |
| D1 | C2 | 2020-03-20 | 100 |
| D1 | C3 | 2020-07-20 | 120 |
| D1 | C4 | 2020-07-21 | 200 |
| D1 | C5 | 2020-08-10 | 500 |
| D2 | C1 | 2020-03-10 | 420 |
| D2 | C2 | 2020-03-10 | 130 |
| D2 | C3 | 2020-07-20 | 180 |
| D2 | C4 | 2020-07-21 | 300 |
| D2 | C5 | 2020-08-10 | 500 |

Table A.3: Question 7a

| Query: | Result: |
|---|---|
| select<br>**Musterlösung:**<br>**sum(NumberOfPart) as Overall**<br>**from Participants** | **Overall**<br>2730 |

b) Die nächste SQL-Anfrage soll die Ergebnistabelle generieren, die rechts angezeigt wird. Dazu sollen wieder die drei Relationen verwendet werden. Sie können SQL Erweiterungen für Data Warehouses verwenden, die in der Vorlesung eingeführt wurden.

Table A.4: Question 7b

| Query: | Result: |
|---|---|
| select<br>**Musterlösung:**<br>**City, Country,**<br>**sum(NumberOfPart) as Persons**<br>**from Location natural join Participants**<br>**group by rollup(Country, City)** | **City** \| **Country** \| **Persons**<br>Rostock \| Germany \| 700<br>Schwerin \| Germany \| 230<br>Leipzig \| Germany \| 300<br>Dresden \| Germany \| 500<br>Prague \| Czech Republic \| 1000<br>- \| Germany \| 1730<br>- \| Czech Republic \| 1000<br>- \| - \| 2730 |

c) Die nächste SQL-Anfrage soll das Ergebnis generieren, die rechts gezeigt wird. Sie können wieder die SQL-Erweiterungen für Data Warehouses verwenden, die in der Vorlesung eingeführt wurden.

Table A.5: Question 7c

| Query: | Result: |
|---|---|
| select<br>**Musterlösung:**<br>**Distance, Country,**<br>**sum(NumberOfPart) as Persons**<br>**from Participants natural join SportsDiscipline**<br>**natural join Location**<br>**group by cube(Country, Distance)** | **Distance** \| **Country** \| **Participants**<br>Marathon \| Germany \| 700<br>Half marathon \| Germany \| 1030<br>Marathon \| Czech Republic \| 500<br>Half marathon \| Czech Republic \| 500<br>Marathon \| - \| 1200<br>Half marathon \| - \| 1530<br>- \| Germany \| 1730<br>- \| Czech Republic \| 1000<br>- \| - \| 2730 |

## Question 8:

Die folgende Relation ist gegeben.

Table A.6: Question 8

| **Location** | **Time** | **Day** | **Ozon** |
|---|---|---|---|
| Kritzmow | 1598726121 | 29 | 75 |
| Rostock, City center | 1598726124 | 29 | 76 |
| Nienhagen | 1598726127 | 29 | 75 |
| Rostock, Südstadt | 1598726130 | 29 | 77 |
| Kritzmow | 1598726134 | 29 | 78 |
| Rostock, City center | 1598726139 | 29 | 75 |
| Nienhagen | 1598726140 | 30 | 75 |
| Nienhagen | 1598726145 | 30 | 74 |

**(d)** Wie sieht die Speicherung einer solchen Relation in einem Column Store aus? Geben Sie bitte zu jeder Column eine gut geeignete Kompressionsstrategie **(Part a)** an und zeigen Sie diese für das oben angegebene Beispiel **(Part b)**.

**Column 1: Location**

**Part a)** Which compression method can you apply?

Compression method: Dictionary Encoding

**Part b)** How does the compressed column 1 look like?

```
0 Dictionary
1 0:  Kritzmow
2 1:  Rostock, City Center
3 2:  Nienhagen
0 3:  Rostock, Südstadt
1
2
2
```

**Column 2: Time**

**Part a)** Which compression method can you apply?

Compression method: Differential Encoding

**Part b)** How does the compressed column 2 look like?

```
1598726121
         3
         3
         3
         4
         5
         1
         5
```

**Column 3: Day**

**Part a)** Which compression method can you apply?

Compression method: Reference Encoding

**Part b)** How does the compressed column 3 look like?

```
Frame:  29
0
0
0
0
0
0
1
1
```

**Column 4: Ozon**

**Part a)** Which compression method can you apply?

Compression method: Reference Encoding

**Part b)** How does the compressed column 4 look like?

```
Frame:  76
        -1
        +0
        -1
        +1
        +2
        -1
        -1
        -2
```

## Question 9:

Der Differential Snapshot Algorithmus kann eingesetzt werden, um die Unterschiede zwischen 2 Snapshots einer Datenbank zu bestimmen, die zu verschiedenen Zeiten gemacht wurden. Diese Unterschiede können durch eine Folge von add, delete und update Operationen beschrieben werden. Geben Sie bitte für das folgende Beispiel an, mit welcher Folge von add, delete und update Operationen Snapshot $i+1$ aus Snapshot $i$ gebildet werden kann.

**Snapshot** $i$

| **ID** | **Attr1** | **Attr2** |
|---|---|---|
| 1 | A | A |
| 2 | B | B |
| 3 | C | C |
| 4 | D | D |

**Snapshot** $i+1$

| **ID** | **Attr1** | **Attr2** |
|---|---|---|
| 2 | B | B |
| 3 | C | C |
| 4 | F | F |
| 5 | E | E |

Table A.7: Question 9

| **List of operations:** | |
|---|---|
| | `DEL 1` |
| | `UPD 4:  (F, F)` |
| | `INS 5:  (E, E)` |
| | |
| | |
| | |

# Declaration

I hereby declare that I have completed this Master's thesis independently and have not used any sources or aids other than those indicated.
All passages taken either literally or in substance from publications have been clearly identified as such.
The thesis has not yet been published and has not been submitted in a similar or identical form for recognition, assessment, or evaluation as an examination achievement.
Large Language Models (ChatGPT 5.5) were used in this work. The thesis was originally written in French and subsequently translated into English with the assistance of LLMs.
Rostock, the June 23, 2026