viSQLizer:
An interactive visualizer for learning SQL

Kristin Annabel Torjussen Folland

Supervisor: Prof. Rune Hjelsvold

01-06-2016

NTNU

Master's Thesis
Master of Science in Applied Computer Science
30 ECTS
Department of Computer Science and Media Technology
Norwegian University of Science and Technology

Acknowledgments

I would like to express a deep appreciation to Rune Hjelsvold, for supervising me this past year, and giving me advice during my master thesis project work.

I would like to thank Tom Røise and Simon McCallum, for helping me recruit students for experiments. I would also like to thank all students and the three professors that have participated in the interviews/experiments in my master thesis.

To all relatives and friends who have supported me physically, morally and financially – thank you.

Without you, none of this would have been possible.

Abstract

Structed Query Language (SQL) is used for interaction between database technology and its users. In higher education, students often struggle with understanding the underlying logic of SQL, thus have trouble with understanding how and why a result table is created from a query. A prototype of a visual learning tool for SQL, viSQLizer, has been developed to determine if visualizations could help students create a mental model and thus enhance their understanding of the underlying logic of SQL. Trough the use of animations and decomposing, our results indicate that visualizations might give students a better understanding of the underlying logic, and that students gain the same learning outcome through visualizations as when using a online tutorial with explanatory text and exercises. Feedback from both professors and students from conducted interviews and experiments indicate that the tool could be used by professors as a visualization tool in lectures, and by students as an practical tool; not as a replacement of but as an addition to traditional teaching methods.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Problem description

Database Systems is an important subject in Computer Science degrees. Data manipulation and organization has been a pillar in Computer Science degrees for decades [1, p. 154]. The learning methodologies used today by course coordinators varies, but often consist of lectures, lab sessions and programming projects, and advanced DBMS systems or blackboards are often used to visualize the database structure and logic [2, 3].

Different database professors organize database courses differently, and assess the students on different topics, skills and knowledge. DBMS concepts, database design and transaction processing are all topics covered in most database courses [2]. In the course "*Database Systems*" at Colorado State University, "*Query languages*" and "*Relational databases*" are the topics that are allocated most time within a duration of fifteen weeks [2, p. 299]. Query languages are perceived as a difficult subject for the students to learn. Students are expected to know how to create queries when they graduate with a Computer Science degree [4].

Structured Query Language (SQL) is supported by relational database systems and is a seemingly simple way to access and change data in relational databases [5], but students often struggle with learning the fundamental SQL concepts and the syntax. According to Sadiq et al. [6], the SQL SELECT command is often perceived by students as simple but is in fact capable of formulating complicated queries. Kearns et al. states that "*students do not understand the fundamental concepts that drive the language*" [4], and tend to miss "*the linkage clause from a multi-table query*" [4] or make other syntax errors, resulting in an result table with incorrect data. A number of researchers state that students have difficulties learning SQL [6, 7, 8].

To solve this problem, this study will investigate the use of visualizations in an interactive learning tool for teaching novice students SQL. A learning tool could give students a better understanding of SQL, by visualizing the underlying logic and letting students learn through a practical approach.

1.2 Motivation

Students using an interactive learning tool could get a better understanding of how SQL works. An interactive learning tool could also be motivating for the students, which might result in them spending more time using the tool. Database professors could benefit from an interactive learning tool, using the learning tool as a means to visualize the database structure and logic in lectures. It could also be easier for database professors to teach SQL when students use more time in the course; the students could get a better understanding and develop a mental model. The university could benefit by graduating students with a higher level of SQL knowledge and thus getting a better reputation. Developers of learning tools could benefit, because research in this area might result in better understanding of how an efficient SQL learning tool should be developed and visualized. Educational material authors could also benefit from this research when developing interactive platforms for students learning SQL.

1.3 Keywords

Relational databases – SQL – Visualization – Animation – Decomposition – Learning tools – Student learning

1.4 Research questions

We ask the following research questions:

Research question 1:
How can visualizations be integrated in a learning tool to help students create a mental model, and thus understand the underlying logic of SQL?

We want to look into different visualization ideas to be used in a SQL learning tool, to find some visualizations that could help students create a mental model on how the result table is created from an SQL query, and determine if this makes the students more aware of the underlying logic.

Research question 2:
What are the effects on motivation and learning outcome when using a learning tool with visualizations compared to using an ordinary online tutorial?

We want to compare a SQL learning tool using visual features with an online SQL tutorial, to determine the effects on students' learning outcome per time unit, and determine the effects on students' motivation when using a learning tool compared to traditional teaching methods.

Research question 3:
How can interactive learning tools be applied in relation to traditional methods for teaching SQL?

We want to figure out how our learning tool could be applied to the traditional teaching methods present in database courses today.

1.5 Thesis structure

SQL, animations and decomposition are described in in Chapter 2. In Section 2.1, related work in the field is discussed and various learning tools are grouped in relation to visualization and learning outcome. The overall methodology and the methodology structure for our study is described in Chapter 3. In Chapter 4, 5, 6 and 7, four different phases of the project are described; expert interviews, development, experiment and expert feedback – with methodology, results and discussion. In Chapter 8, the results from all the project phases are discussed and analysed, before a conclusion is stated in Chapter 9. Future work is discussed in Chapter 10, both in terms of scientific work and further implementation of a visual learning tool.

2 Background

According to Prior [8], SQL is a practical skill. Animations and visualizations have often been used in various disciplines to support complex topics [9], and research has shown that students "*respond well to a practice-in-theory approach*" [10]. Allenstein et al. proposes that visualization can be used as a tool to provide "*another way to think about a given concept*" [10]. According to Patwardhan & Murthy [11], interactive visualization offers a considerable amount of benefits for students. "*The learning benefits derived from their use have covered a range of abilities and skills, such as explanation ability, comprehension, ability to correlate scientific concepts, learning of conceptual and procedural knowledge, process of skill acquisition, building of mental models, as well as increased learners' engagement*" [11].

SQL is a declarative language, which means that the query specifies what is to be done in the database, whereas a procedural language explains how to do it. Prior [8] states that students have problems with mentally visualizing the intermediary result from the execution of various SQL operations. SQL could be difficult for students to understand because of its declarative nature, "*whereas procedural languages allow students to approach complex problems in steps, SQL requires learners to think sets rather than steps*" [6]. As such, SQL queries cannot be separated into various steps to be solved separately, but can instead be separated into temporary sets of data, resulting from various parts of the same query. This can be done through decomposition, where the query is broken down into smaller, simpler queries/sets. This is done by separating the various operations in the query through a parser, and representing them in their natural order [7].

Brovold, Johnsen & Folland [12] has developed a learning tool with decomposition as the main research area of the project. The main goal of the Decomposer project was to create a learning tool for students to better understand how querying in SQL works, by decomposing the query into multiple steps. The project work with the viSQLizer prototype builds on the results for the Decomposer [12]. The Decomposer uses a PHP SQL Parser [13] to parse the query into multiple temporal sets. The temporal sets are then visualized one at a time in the following order; *join – selection – order – projection*. To explain the decomposition in detail, an example query is provided; "*SELECT name, boss_name FROM person JOIN work ON person.id = work.employee_id WHERE work.occupation = 'lawyer' AND person.age > 35 ORDER BY name*". This query is decomposed into four steps [12], which are explained below.

1. The tables "*person*" and "*work*" are **joined** with the ON conditions. The query for this step is; "*SELECT * FROM person INNER JOIN work ON person.id = work.employee*".
2. The **WHERE** clause is added to the existing query. The rows that fulfills the requirements in

the **selection** are displayed in the result table. The query for this step is; "*SELECT * FROM person INNER JOIN work ON person.id = work.employee* ***WHERE work.occupation = 'lawyer' AND person.age > 35***".

3. The rows are **ordered by** the provided column name. By default, this ordering is done ascending. The rows are displayed in the result table in the correct order, specified in the query. The query for this step is; "*SELECT * FROM person JOIN work ON person.id = work.employee_id WHERE work.occupation = 'lawyer' AND person.age > 35* ***ORDER BY name***".
4. The **asterisk (*)** is replaced with the column names provided in the original query. This step **projects** the selected columns in the result table. The query for this final step is the same as the provided query; "*SELECT* ***name, boss_name*** *FROM person JOIN work ON person.id = work.employee_id WHERE work.occupation = 'lawyer' AND person.age > 35 ORDER BY name*".

2.1 Related work

Several various SQL learning tools have been developed in the past years. Sadiq et al. [6] propose that the best way to learn SQL is trough practice, "*and learning from one's own mistakes*" [6]. Cembalo et al. [7] proposes that these solutions can be divided into two groups:

- Systems that explain which physical components of the DBMS are involved and interact with each other when executing a SQL query.
- Systems that explain the SQL commands and how they can be used.

We further propose to divide the second group into several sub-groups based on type of visualization:

Game Systems that use visual metaphors of SQL with game-like mechanics.

Interaction Systems that let the user interact with the system; write various queries and see the resulting table.

Animation Systems that animate data in the database tables for a better understanding of the SQL logic.

Decomposing Systems that use decomposition to visualize the hierarchical break-down of SQL queries.

These categories are represented in Table 1. We want to determine if decomposition and animation are elements that are widely used in SQL learning tools, or if other types of learning tools are more common today. The various learning tools are described in the following sections.

Explaining DBMS	***Explaining SQL commands***			
-	***Game***	***Interaction***	***Animation***	***Decomposing***
Allenstein et al. [10]	GalaXQL [14]	The Exploratorium [15]	Khan Academy [16]	SAVI [7]
ADbC [9]	Schemaverse [17]	ADbC [9]	SAVI [7]	QueryViz [18]
	Belov [19]	W3Schools [20]		esql [4]
		eledSQL [21]		
		SQLFiddle [22]		
		SQLator [6]		
		SQL-Tutor [3]		
		SQL course [23]		
		CodeCademy [24]		
		Vertabelo Academy [25]		
		SQLZoo [26]		

Table 1: SQL learning systems divided into groups by system explanation and visualization

2.1.1 Explaining DBMS

Allenstein et al. [10] developed a query simulation system for teaching students "*about the steps involved in processing data manipulation language (DML) queries*" [10]. This was done because they felt it was "*fundamental to understanding database management systems*" [10]. In this desktop application, students get to write their own query, or select predefined select-, insert-, update- or delete-statements, to see the animation of the various steps related to the memory structure and the disk structure of the query execution process. The system is designed to be as modular as possible, where various DBMS components are visualized to show the user what is happening in the DBMS for each step. The student can click through the various steps of the process or go back to a previous step. At some points in the visualization process the student has to predict what is going to happen in the next step before it is shown. Allenstein et al. [10] evaluated the system by offering it to students in a Database Systems course. The students reported "*the visual aspect of the system to be better than text explanations*" [10], and that "*the system helped them to understand the timing of various operations*" [10]. Initial informal evaluation showed that the system seemed to improve the students understanding of the database management systems, but Allenstein et al. states that they want to do a formal evaluation to confirm this statement. In the future, Allenstein et al. wants the system to provide the user with additional information about "*the internal state of the DBMS throughout the query execution process*" [10].

Murray & Guimaraes [9] developed a system for animating various tutorials for "*database design, interactive SQL, stored procedures and triggers, transactions and security*" [9], the Animated Database Courseware (ADbC). Murray & Guimaraes state that the traditional teaching methods are not adequate, and thus need some additional teaching methods to teach students complex and dynamic topics. They state that they use visualizations because abstract concepts can best be learned by adding "*instructional material such as software animations*" [9]. The system does not contain actual animations to our knowledge, but visualizes the various topics e.g. through diagrams or tables. The system is intended for use in a classroom setting, either by in-class use or by students using it outside of school for assistance and information. Their preliminary evaluations of the system with both students and faculty were positive, and during a pilot control group evaluation the students using the ADbC system "*outperformed those who did not*" [9]. Murray & Guimaraes [9] state that their system "*facilitate student learning and also provides an opportunity to include more depth and breadth to the concepts covered in a database course*" [9].

2.1.2 SQL games

GalaXQL [14] is a game-like online tool for learning SQL, "*disguised as a galaxy manipulation application*" [14]. The user is introduced to a storyteller teaching SQL commands through multiple subjects. The information is textual, and a galaxy is displayed to the user for visualizing the results from various queries. The user may go through the various subjects of the tutorial, solving tasks given by the professor to alter the appearance of the galaxy. The tasks are solved by writing queries, and feedback is given to the user indicating if the answer is correct. GalaXQL covers all SELECT statements, as well as table operations such as CREATE table, UPDATE table etc. This game appears to be targeting SQL novices, and provides good information to the users in a structured setting.

Schemaverse [17] is an online SQL strategy game that takes place in space. The users are competing with each other, and the objective of the game is to buy fleets and ships and travel through space to conquer planets. To execute all these actions, the user has to write SQL queries on predefined tables. To buy a new ship for instance, the correct query is "*INSERT INTO my_ships(name) VALUES('My Awesome Ship')*" [17]. The game covers both SELECT statements and table operations. Schemaverse provides a tutorial on how to play the game, but we find this hard to follow. This game does not appear to be targeting SQL beginners, but users already familiar with SQL and its commands. This is because the queries do not create any result table, but instead create some action in the game that is not visualized, which makes it hard for a novice to understand how SQL is used in the real world.

Belov [19] developed a serious game prototype for teaching relational databases. The goal was to create an effective educational SQL game in regards of time spent playing the game and average grade score on the final exam. A game board is displayed with various animals at different ages located at different locations in the grid. The game has eleven tasks, and tasks involve selecting various animals on the board through submitting an SQL query. The goal of the game is to execute all eleven tasks successfully. Information about the SQL syntax is given to the user, as well as a graphical interface showing the database table structure. Belov's findings implies that learning through educational games are more effective than self-learning using other teaching methods. The game only covers simple SELECT statements. To our knowledge, the user does not need to use JOIN or aggregation to complete a specific task. This may be considered a serious weakness in the design of the game, as JOIN is an important part of SQL and should be covered in all tools meant to teach SQL to novices.

2.1.3 Interactive SQL

Combination of various learning tools

Brusilovsky et al. [15] "*combines different SQL learning tools in an integrated Exploratorium for database courses*" [15]. The main purpose of the project was to determine the "*importance of integration of multiple interactive educational activities in a single system*" [15], for supporting the needs of a database course. The SQL Exploratorium offers three various systems to the students. One system offers annotated SQL examples, where each line of the query is being described in detail. The Exploratorium also offers a Knowledge Tester (SQL-KnoT) [15], which contains questions that that has to be solved by writing a query. A SQL-Lab is also part of the Exploratorium, where the students get to submit an example query provided through the Explotatorium, or write their own queries. After a query is submitted, the result table is displayed to the student both for the SQL-KnoT and SQL-Lab system. Brusilovsky et al. introduced the Exploratorium in two database courses with undergraduate and graduate students. Results show that more than half of the student groups chose to use the Exploratorium, and that students used all three systems in every fifth session. Feedback suggested that the students overall were satisfied with the various systems, and that the systems helped them during the course. As future work, Brusilovsky et al. want to conduct a second round of evaluations in an undergraduate database course.

Predefined queries

The tutorial by Murray & Guimaraes [9], ADbC, does, as previously mentioned, explain DBMS, but does also include the topics of table manipulation, query building and query execution. The tables, data sets and queries in the database are predefined. Students can choose amongst various SELECT statements and get information about the definitions of the various operations. Both the original table and the result table are visualized in this tutorial.

Freedom to write queries

W3Schools [20] is an online tool for writing SQL queries, that offers the user the possibility to write his/her own queries and to see the result table. The tables that are available to the user are displayed for easy access to table information as column names. The user also has the possibility to execute table operations to insert, update or delete rows, or to create or delete tables in the database. The online tool SQLFiddle [22] has a similar design, but the user has to create his/her own database and tables and insert data into the tables using SQL queries before trying out various SELECT-statements. This might not be a good solution for novices unfamiliar to SQL because table operations are unknown to them and thus they are unable to create their own databases and tables using SQL.

Grillenberger & Brinda [21] created an interactive tool, eledSQL, meant to be used by novice students learning databases in a 9th grade class. They wanted the learning tool only to cover the functionality needed for this specific group of students, and not all functionality that are offered by professional database tools. eledSQL is divided into three levels; "*At the lowest level the formulation of queries is partly based on forms and partly based on natural language. Prior SQL knowledge is not required there. On the next higher level queries can be entered as on the lowest level, in addition, however, the underlying SQL syntax appears automatically. On the third level the interaction with the database should be only possible with SQL*" [21]. Information about the available database tables is provided to the user and a result table is displayed when a query is executed. The database teacher has the possibility to create tables and data sets, and decide the knowledge level for an entire class. eledSQL was introduced to students in a 9th grade class. Their experience was positive, and Grillenberger & Brinda discovered that the students managed to use the tool without instructions about how to use it.

Cigas & Kushan [27] have researched various SQL tools to be used in a database course. According to this work, Teradata SQL Assistant/Web Edition [28] is an SQL environment where students can write queries and see the result table. This is done on a number of predefined databases. Both SELECT statements and table operations can be done on students' private sets of tables. Students also have the ability to save the queries on their own computer. Database course coordinators can to look at the students' databases and saved queries. Cigas & Kushan state that they would like the Teradata SQL Assistant/Web Edition to be available online in the future, so that students does not have to "*install any additional software*" [27] to use the tool.

Query exercises

A wide range of SQL tools offer exercises to their users; exercises that need to be solved by typing in the correct query before the user may continue through the learning tool.

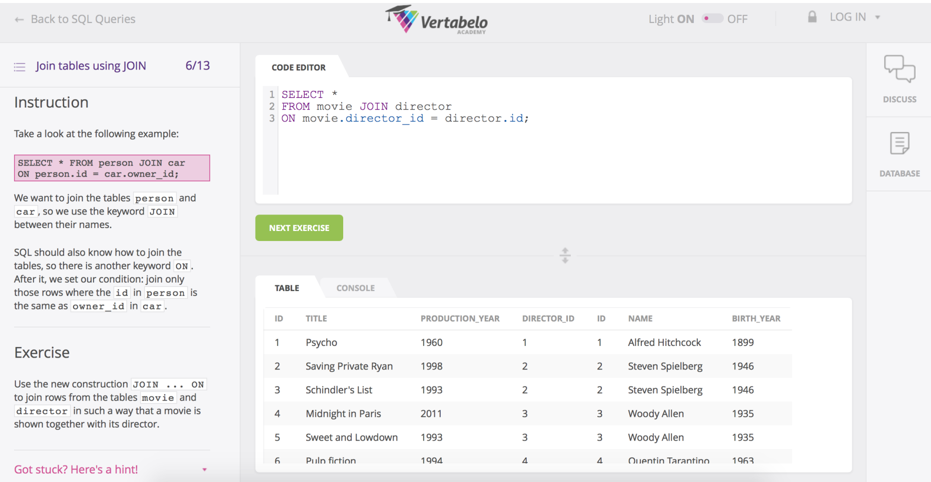
Sadiq et al. [6] developed a web-based interactive tool for learning SQL, called SQLator. Their goal was to achieve a deeper understanding of SQL amongst students in higher education by developing technology handling the full power of SQL. The tool is meant to be used by students as an addition to the traditional teaching methods used in database courses. It offers multimedia tutorials for SQL concepts, and provide students with the ability to execute both some preset practice queries and to write their own SQL queries using SELECT-statements. Students are presented with a task, and the correct query has to be submitted to continue with further exercises and topics. The tool presents the result table of the submitted query and provides feedback to the students about the correctness of the query. The SQLator was introduced in a first year database course where the students had six weeks to use the tool and complete an assignment. The results show that there were a "*substantial use of SQLator outside lab times*" [6], and the exam results were overall better than in the previous year. The number of students getting a grade below the average was lower than the previous year, and the number of students getting a grade above the average increased. Approximately 40% of the students were placed in the second highest grade group, while approximately 30% of the students from the previous year were placed in the same group. Sadiq et al. [6] state that the feedback provided to the students in the learning tool, as well as the possibility to practice various queries with varying difficulty, are the reasons for the successful deployment of SQLator. In the future, Sadiq et al. want their system to contain hints, used as guidelines for each query.

Mitrovic [3] developed an intelligent desktop system for teaching students SQL, called SQL-Tutor. SQL-Tutor was developed for students to practice SQL and was meant as an addition to a traditional database course. SQL-Tutor only covers SELECT statement. The learning system is tailored for each individual student in regards of instructions, tasks and feedback based on their level of skill. The SQL-Tutor is designed with the student's cognitive load in mind; displaying the available keywords for use in an SQL query, and the relative order of the clauses. The database table structure is also displayed, to "*remove some of the cognitive load*" [3] for the students. If there are any errors in the submitted query, a user-friendly error message appears, explaining what is wrong with the query. SQL-Tutor has been shown to database professors and the feedback was supportive and enthusiastic; the professors wanted to use this system in their own courses. Mitrovic implies that the SQL-Tutor system strengthens the students' knowledge in practice and that it helps students understand in a deeper way. "*It supports three kinds of learning: conceptual, problem solving and meta-learning*" [3]. In the future, Mitrovic wants to conduct an evaluation study. In addition, Mitrovic wants to connect the learning tool to a database, to display the result table when submitting a query.

SQL Course [23] is an online learning tool that provides information about specific SQL subjects and exercises where the user can submit queries to answer. The tool also provides the user with the possibility to look at the answers to the exercises. Both SELECT statements and table operations are covered by SQL Course.

CodeCademy [24] is another online learning tool. CodeCademy has multiple courses available, and amongst these, three SQL courses. It provides a short introduction to the current SQL subject and provides the user with exercises. To solve the exercises, a query can be copy-pasted from the introduction and modified before it is submitted. This seems as a shortcut for the users, as the users do not have to know how to write a query to submit the answer, and thus this might not help the users learn SQL. When the correct query is entered, the result table is displayed to the user and the user can proceed to the next subject in the learning tool. CodeCademy covers SELECT statements, table operations, and business metrics analysis.

The online learning tool Verabelo Academy [25] provides the user with three different SQL courses. Their "*SQL queries*" course is divided into several subjects. For each subject, both information and an exercise is provided to the user. A query is submitted as an answer to the exercise and the system checks if the answer is correct. In this case, the resulting table is displayed and the user is able to proceed to the next subject. The exercise has to be solved before the user can proceed in the learning tool. The tab "database" is available to the user, containing information about the available tables, columns and data. Hints are available to the users in the tutorial, and badges are given through the learning tool for finishing various subjects in the tutorial. Vertabelo Academy cover both SELECT statements and table operations. The design of the Vertabelo Academy tutorial is shown in Figure 1.



Join tables using JOIN 6/13

Instruction

Take a look at the following example:

```
SELECT * FROM person JOIN car
ON person.id = car.owner_id;
```

We want to join the tables `person` and `car`, so we use the keyword `JOIN` between their names.

SQL should also know how to join the tables, so there is another keyword `ON`. After it, we set our condition: join only those rows where the `id` in `person` is the same as `owner_id` in `car`.

Exercise

Use the new construction `JOIN ... ON` to join rows from the tables `movie` and `director` in such a way that a movie is shown together with its director.

CODE EDITOR

```
SELECT *
FROM movie JOIN director
ON movie.director_id = director.id;
```

ID	TITLE	PRODUCTION_YEAR	DIRECTOR_ID	ID	NAME	BIRTH_YEAR
1	Psycho	1960	1	1	Alfred Hitchcock	1899
2	Saving Private Ryan	1998	2	2	Steven Spielberg	1946
3	Schindler's List	1993	2	2	Steven Spielberg	1946
4	Midnight in Paris	2011	3	3	Woody Allen	1935
5	Sweet and Lowdown	1993	3	3	Woody Allen	1935
6	Pulp fiction	1994	4	4	Quentin Tarantino	1963

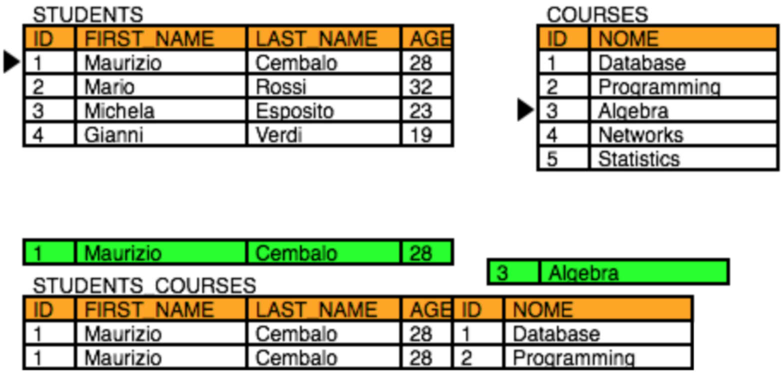
Figure 1: The design of Vertabelo Academy [29]

SQLZoo [26] is an online SQL tool where the user is provided with information about the current SQL subject and an exercise. The answer, represented as a query, just has to be modified by the user, because a similar query is represented in the provided information text. This seems like an disadvantage for the users, because the provided query examples might lead to limited learning outcome. SQLZoo also has an interactive quiz for each subject. A question is presented to the user as well as a handful of options. The correct option, containing a table or a query, has to be clicked to answer the question. The user has to choose one right answer for each question, and when all questions are answered the user get a score to see how many answers were answered correctly. SQLZOO covers both SELECT statements and table operations with explanatory text and exercises, but only the SELECT statements are subject in the available quizzes.

2.1.4 SQL with animations

Khan Academy [16] a SQL course online. This course is divided into multiple topics where some of the topics contain information, some contain exercises, and some contain interactive video with sound. The video displays various queries, and a voice explains the database and the various queries. When a new query is written in the video, the result table is displayed. The user has the ability pause the video and do changes to the queries or write new ones. The various resulting tables are updated/created on the fly. When the video is restarted, the text that was edited is returned back to its former state and the movie continues where it left off. Khan Academy covers both SELECT statements and table operations in their SQL course.

Cembalo et al. [7] developed a System For Advanced SQL Visualization (SAVI), visualizing the SQL operators selecting data from a database. Their visualization was created using the HTML5 Canvas element. Cembalo et al. created their system using standard web technologies, making it available for all students in a standard web browser. Cembalo et al. wanted to do this project for "*supporting the teaching and the understanding of the semantics of the SQL language*" [7], and to "*improve the visualization approach provided by existing systems*" [7]. As the SAVI system is restricted to selection queries only, a variety of predefined select-operations can be run and visualized in the system. When a query is run, an animation visualizes the data moving from the original table(s) to the result table. An arrow goes through each row in the original table and animates the row moving to the resulting table if it fulfills the specified criterion in the query. Figure 2 shows the SAVI visualization in progress. In the future, Cembalo et al. want to make their system work with bigger data sets, which according to Cembalo et al. require some sort of alternative visualization of the tables, using metaphors or similar solutions. Cembalo et al. also want to perform an experiment "*to evaluate the effectiveness of our system when used, during a class, to support the explanation of some of the topics related to SQL queries*" [7].



STUDENTS

ID	FIRST NAME	LAST NAME	AGE
1	Maurizio	Cembalo	28
2	Mario	Rossi	32
3	Michela	Esposito	23
4	Gianni	Verdi	19

COURSES

ID	NOME
1	Database
2	Programming
3	Algebra
4	Networks
5	Statistics

1	Maurizio	Cembalo	28

3	Algebra

STUDENTS_COURSES

ID	FIRST NAME	LAST NAME	AGE	ID	NOME
1	Maurizio	Cembalo	28	1	Database
1	Maurizio	Cembalo	28	2	Programming

Figure 2: The SAVI visualization [30]

2.1.5 SQL with decomposition

The solution by Cembalo et al. [7], SAVI, also uses decomposition when visualizing the SQL query. This is done by using a Java SQL parser. When a student writes a query, the query is divided into various steps by the SQL Parser, and the user can click through various steps to see the various decomposed queries being animated. Cembalo et al. implies that by using decomposition, their students are assisted to "*overcome the mental visualization problem, concerning the difficulty in understanding the way the various operators of an SQL complex query interact with a database setup to the final result*" [7].

Danaparamita & Gatterbauer [18] developed a visualization tool for SQL queries, called QueryViz. The tool targets various audiences, including "*novices that try to familiarize themselves with the logic behind alternative patters of SQL queries*" [18]. Danaparamita & Gatterbauser states that the understanding of queries are "*often as hard as creating a new query*" [18] and thus have developed QueryViz to allow students to read and understand existing queries faster. They also state that "*students often have difficulties understanding the logic behind correlated and uncorrelated nested queries*" [18]. QueryViz parses the table schema and queries, and outputs a visualization of a graph. The learning tool only focuses on set semantics and has not implemented support for grouping, outer join, aggregation or disjunctions [18]. In QueryViz, the user selects a schema and an operation. This operation is visualized as text, and the corresponding query is visualized to the user. Upon execution, a graph is represented to the user, visualizing the logic behind the query. The QueryViz graph is shown in Figure 3, for the query "*SELECT S.bar FROM Serves S, Likes L WHERE S.drink = L.drink AND L.person = 'Joe' AND not exists (SELECT S2.bar FROM Serves S2, Likes L2 WHERE S2.drink = L2.drink AND L2.person = 'Michael' AND S.bar = S2.bar)*" [18]. Danaparamita & Gatterbauer state that they want to implement more visual constructs for QueryViz in the future.

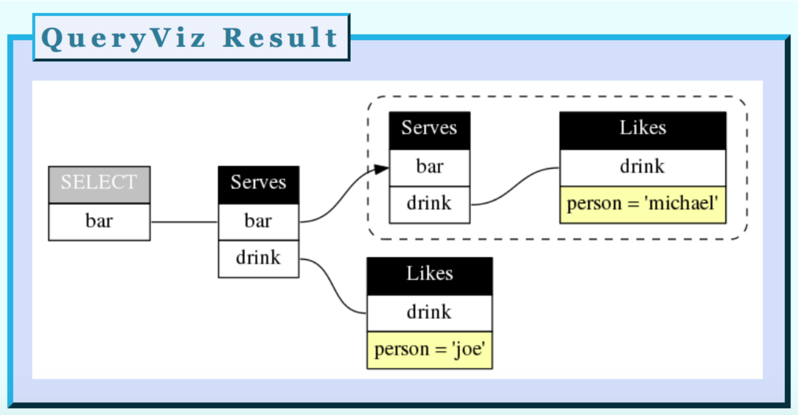


Figure 3: The QueryViz graph visualization [31]

Kearns et al. [4] developed a system, called esql, to teach students how to express queries in SQL. "*The aim of the project was to design a system which would be used in conjunction with more conventional learning methods, so that students could learn more effectively*" [4]. The system was meant as a learning system for SQL novices. Their system has an emphasis on showing how a result table is created. Students can write any SELECT statement and the queries are decomposed into multiple steps. Students can click through each step to see how the content of the result table is produced for each step, visualized as images of database tables. As such, students might understand how the final result table is created, by viewing all the decomposed steps leading up to the final result table. As the various steps display a table, an explanatory text is provided to the user, explaining what happens to the table in the current step. Kearns et al. use a small data set because a big data set could be "*slow and cumbersome to display*" [4]. Kearns et al. state that their system meet their goals, and that "*the dynamic stepping-through mechanism which allowed students to see a query in action as well as its overall result is both novel and far superior to the usual pen-and-paper explanations given in lectures and textbooks*" [4]. In the future, they want to add a tutorial in the system to make it easy for the lecturers to create and mark exercises that he/she wants the students to work with.

2.2 Discussion

The majority of the encountered projects and learning tools for SQL are located in the "*interaction*" group, as seen in Table 1. Only two of the tools contain animations, which might indicate that the power of animations in a learning tool has not yet been fully researched. Cembalo et al. [7] are the only encountered researchers who have discussed animations, but they have not done any experiments to measure the effectiveness of an learning activity using animation. The efficiency of an learning activity using animations remains unsolved, and we want to conduct an experiment to get some results on this matter.

Only three of the encountered learning tools use decomposition when visualizing SQL [7, 18, 4] which might indicate that the power of decomposition in a learning tool has not been fully researched. As Kearns et al. [4] stated, "*this is much more informative than merely displaying the result of the query, as in commercial SQL systems or other teaching systems*" [4]. Because SQL is a declarative language, the decomposition has to be done in temporal sets rather than procedural steps. Temporary tables needs to be created for the various query sets to create a step-based visualization of SQL. None of the researchers did perform any experiments with decomposition in a learning tool, and thus there are no reported disadvantages or benefits of using decomposition in a learning tool. We want to determine if temporal sets could help students understand the underlying logic of SQL.

Relational algebra is the mathematical foundation of relational databases. Join, selection and projection are basic operations in relational algebra, which makes these a good choice for teaching students SQL logic. Sorting, however, is not a part of relational algebra, but is still an important step in SQL. We want to visualize sorting of rows to students, which relational algebra is not suitable for; this is possible by using decomposition. In addition, relational algebra creates new notations that we do not need in the decomposition of SQL. We want to develop a tool that does not require previous knowledge of relational algebra; a tool that is useful both with and without knowledge of relational algebra. This is why we choose to use SQL decomposition when teaching students the underlying logic of SQL.

The learning tool developed by Cembalo et al. [7] is the only encountered learning tool that uses both animation and decomposition. The absence of research on the effects of learning activities using both decomposition and animation, in relation to learning activities without these visualizations, makes us believe that it is worth the effort trying to answer research question 1: *How can visualizations be integrated in a learning tool to help students create a mental model, and thus understand the underlying logic of SQL?* and research question 2: *What are the effects on motivation and learning outcome when using a learning tool with visualizations compared to using an ordinary online tutorial?*.

Patwardhan & Murthy [11] states that interactive visualization cover a range of abilities and skills, and we hypothesise that interactive visualizations could be beneficial in a lecture setting as well as a practical tool for students in addition to traditional methods of learning. We want to research this further to answer research question 3: *How can interactive learning tools be applied in relation to traditional methods for teaching SQL?*.

In most learning tools only the result table is visualized to the user, and only two of the encountered learning tools visualize both the original table and an result table [7, 9]. We argue that by visualizing both tables, students get a greater understanding of how the content of the result table is produced during a query execution.

Almost all of the encountered learning tools ([7, 20, 25, 26] amongst others) are accessible on the Internet, through a web browser. This makes it easy for any student/professor to access the learning tool from any location and any computer with an Internet connection. Many of these learning tools ([6, 7, 25, 26] amongst others) contain a sample database and sample data sets so that the user can jump right into learning SQL. This seems as a good solution, especially for learning tools with a focus on SELECT statements.

The majority of the encountered learning tools only support SELECT statements. As Cembalo et al. [7] state, these are the most common types of queries. SELECT statements are needed in all client/server applications with a connection to a database. We believe that SELECT is the core subject in SQL, and want to focus on these type of statements when developing the prototype.

The SAVI application [7] uses both animations and decomposition. The highlighting on the selected rows makes it clear for the user what row is being animated. The visualization of an original table and a result table makes it possible to animate cells from the original table to the result table. The animations are slow because all decisions are being animated. Our objective is to animate the cells quicker in our solution to make the visualizations efficient.

Vertabelo Academy [25] provides good and informative explanations of the various SELECT statements and suitable exercises with appropriate difficulty. The tutorial visualizes the result table and provides the user with information about the available tables and columns. We aim at an online tutorial that has various learning activities and uses a different design than our prototype, to answer research question 2: *What are the effects on motivation and learning outcome when using a learning tool with visualizations compared to using an ordinary online tutorial?*. We choose to use Vertabelo Academy in our experiment; comparing our prototype to this tutorial and its various learning activities.

3 Overall methodology

This project work is structured to follow a path of different methods, as shown in Figure 4. In the expert interviews, we want to get information from different database professors about their course organization and students' threshold concepts. This information could help us decide what to focus on during the development of our viSQLizer prototype. The development process is done in two iterations, with a preliminary user test in the middle. Through this user test we want to get information about how students perceive the various implemented visualization ideas. This information could help us decide what visualization ideas to keep in the next development iteration. After the development phase, an experiment is conducted to measure the students' learning outcome and to get some insight into the students' understanding of the underlying logic of SQL. Feedback is given to us from database professors on the final version of the prototype, in terms of how they would like to use our developed learning tool in their database courses and which features they would want to be implemented in the future.

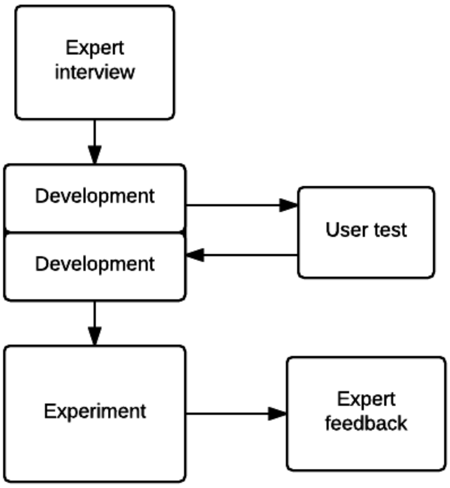


Figure 4: The project structure

Random sampling is used when choosing students to participate in the interviews. This is done by picking students from random locations in the room. All interviews are done individually, and recorded on an audio recorder. The interviews are later transcribed using the recordings, and the transcribed documents are analysed. It is important to understand that the colloquial and written language is different [32, p. 36], and the transcription is therefore not a "pretty" document with text. It consists of pauses, unfinished sentences and words that are not in the dictionary. The transcriptions of the interview data are written with a colloquial language.

The Norwegian Centre for Research Data [33] (NSD), is notified about the conducted user test and experiment, and forms related to this can be found in Appendix C and F. All students and professors participating in interviews or experiments are asked to sign a agreement form. These forms contain information about the experiment/interview setup and the master thesis itself. These agreement forms can be found in Appendix B, D, G and H. All students participating in the experiment are observed, and notes of observations are taken by us for further analysis later. Multiple observers are present at large experiments conducted at multiple locations, observing the students participating in the experiment without having any conversation with them.

The version control system Git is used in the development process. In chapter 5, the technical specifications for the server and client side technology are provided. The code snippets provided here are "beautified" for the purpose of being easy to understand, and is not identical to the original implementation. The implementation for viSQLizer can be found at our GitHub repository[1].

[1] https://github.com/kristinannabel/viSQLizer

4 Expert interviews

4.1 Methodology

Expert interviews are conducted to determine what database professors does to motivate their students, and what parts of SQL are perceived as troublesome for the students.

The area of interest is concretized into some main topics and problem statements that are to be answered by various database professors. The interviews could give us an overview of how the database courses are conducted today and information about students' threshold concepts [34] in database courses. This information could help us decide what topics are more important to implement in our prototype, and how a database course could be improved by using a learning tool. The problem statements for the interview are:

1. How are traditional database courses organized today?
2. What are the preferred learning methods of the database professors?
3. Do database professors see the need for a SQL learning tool as an addition to traditional learning methods?
4. What are the students' threshold concepts?
5. How do the professors think the students get motivated through learning?

The professors asked to participate in an interview are selected on the grounds of their profession and their knowledge of teaching SQL. Rune Hjelsvold, the supervisor of this thesis, is a database professor at NTNU Gjøvik and has contacts in the community of database professors. He, per our wish, sent out a mail to some of these database professors who could be of interest to us. This mail contained information about the project work and the topics to be discussed at the interviews. We wanted as many professors for the interview as possible, for getting a wide range of interpretations of the questions and answers. The mail was sent out to seven database professors at NTNU spread out on various campuses.

The interviews are organized as semi-structured. The questions are created as open questions, where the professor get to answer what ever he or she feel is relevant to the topic. An interview guide was created, including the central topics and questions needed for better understanding of the overall area of interest. A pilot interview was conducted with the first professor that replied on the request sent out by mail, and some changes was done to the interview guide after the pilot. The final version of the interview guide is included in Appendix A. The results from the pilot interview are used in later analysis and work with the thesis in addition to the results from the other interviews. For coding and analysing the collected material and transcribed results, the transcriptions are read through and the interesting parts answering the interview problem statements are organized together in a document, under headings for each problem statement.

4.2 Results

In total three database professors were interviewed. One of the professors have not been teaching databases in the last few years, while the other two professors are actively teaching databases. The duration of the interviews were from half an hour to one hour depending on the amount of information each professor found relevant to the questions.

We wanted to address the problem statements introduced in the method section above. In Table 2, these five problem statements are organized with the corresponding answers from each professor. The results are described more in detail below.

Problem statements	Professors		
	A	B	C
Organization: step by step	Basic DB structure	Single-table selection/projection	Algebra
	Planning structures	Multiple tables with cartesian product	Relational models
	Normalizing	Join	Single-table selection/projection
	Scripting	Aggregation	Join
		Schema operations	Aggregation
			Nested queries
Preffered learning methods	Lab	Lab	
	Competitive tasks	In-class quiz	In-class tasks
	Blackboard	Blackboard	Blackboard
		Problem-based learning	
Would like to use a learning tool	Yes	Yes	Yes
Threshold concepts	Concepts	Join	Join
		Grouping	Grouping
		Nested queries	
Motivation through learning	Database field	Own experience / industry	Various learning styles
	Competition	Exercises	Exercises
	Games/Fun	Quiz in lectures	Tasks in lectures
			Discussion

Table 2: Coded results from the expert interviews

How are traditional database courses organized today?

As seen in Table 2, the three database professors organize their course differently. One starts of the database course with basic structures while another professor starts the course with a repetition of algebra and relational models. But as all professors state, basic SQL knowledge is the starting point. "*First we look at one table; simple queries with selection and projection. Then we look at multiple tables with cartesian product and JOIN. Then we explain sub queries, and maybe aggregation. And at the end, we explain topics like insert, delete, and schema operations*". "*We teach algebra and relational models first, then relational algebra and selection, projection, JOIN and aggregation. We start with algebra, because with algebra it gets operationalized in the direction of implementation. I think that is a good way to start, because SQL just smears everything into one sentence*".

What are the preferred learning methods of the database professors, for teaching SQL?

The three database professors state that they use the blackboard for visualizing SQL during lectures. "*The reason is that it takes some time to visualize on the blackboard, and thus the students have more time to understand it. If I use PowerPoint for instance, the pace is too quick for the student to follow*". All three also prefer to use some kind of exercises or quiz during their lectures. Two of the three also mention lab as a preferred learning method when teaching SQL. "*Learning is facilitated when the students get to try for themselves. In lab sessions you spend time with the students. They get to try while you are present to watch and help them, but you are dependent on assistants and help*".

Do database professors see the need for a SQL learning tool, as an addition to traditional learning methods?

As shown in Table 2, the three database professors state that they would like to use a SQL learning tool as part of their lectures, or as an offer to the students as an addition to the traditional lectures. "*It would be really exciting*". "*I think it is good to have an offer in addition to supervised lectures. You got someone who can tell you what is important, what is difficult, what you should work with in the lectures, and then in addition, you could try it out yourself. Maybe some questions will arise when you use the tool, and with the help of the lectures and people who know more than you, you might get an answer to your questions*". "*When a good system appears where I can drill the students on SQL syntax, then I would adapt it and use it in my own classroom. But I haven't found any such system that I want to use yet*". "*When I look at the exam results I quickly get an understanding of which students have tried SQL in practice and who have not. I think that SQL is easy if you practice, and that there is a limit to how little you can manage to get right on an exam if you have been through the course as intended*".

What are the students' threshold concepts, when learning SQL?

Two of the three professors state that join and grouping are the most difficult concepts to grasp for the students. "*JOIN is difficult in relation to projection and selection. It is like going from addition to division in math, or when we get to a loop in programming*". Nested queries were also mentioned by one professor as a difficult concept.

How do the professors think the students get motivated through learning?

Two of the professors state that information about the field and the professors own experience from the industry motivate their students. "*I am trying to explain the meaning of databases as a field, in relation to its necessity in the base of most applications*". "*I think it helps to tell stories from my own experience in the industry; draw from my experiences to show that the material is relevant*".

All professors state that exercises, competitions or games/quizzes might motivate students. "*It is really important to have some breaks in a lecture. It is not too many minutes you can lecture on the blackboard before the students begin to sleep, so it is important to have some exercises, quizzed and such they can do for themselves to break up a bit*". "*I often organize competitions. ... I do some stuff to motivate them to have a good time during the lecture. I have experienced this as positive; the students find this as amusing*".

One of the professors also mention discussions and tasks in the lectures as being motivating for the students. "*Sometimes the students are nodding and seems to understand the content before lunch, and cannot understand anything after lunch. So I changed the course organization to 'Stop. Do a exercise, try it for yourselves'. I try, if I have the time, to let the students show their solution on the blackboard. And someone else might have another solution. Then we can discuss each solution, and why they are good or not. These types of discussions are important*".

4.3 Discussion

The interviewed professors seem to organize the query part of their database course in similar ways. The professors start off by teaching single table operations, with a focus on selection and projection, and proceeds with more advanced, multiple table operations with JOIN and aggregation. We believe that knowledge can best be acquired when the teaching material is based on previously acquired knowledge. Thus, we choose to implement both single table operations as well as multi table operations in the prototype, to facilitate learning for multiple levels of difficulty as the database professors do. Still, we want to develop a learning tool for novice students, which is why we choose not to focus on sub-queries, aggregation etc. One of the professors state that scripting is the last step in his/her database course, but does not specify how this scripting part is organized. Thus, we want to look at the answers from the other two professors in regards of the organization of the query part in the database courses.

All three professors use blackboard for visualization during database lectures. Visualizing tables on a blackboard could be cumbersome, and some students might not understand the logic of SQL just by looking at a blackboard. Using a learning tool with visualizations in lectures could help students create a mental model for SQL.

Two of the professors state that they would like the opportunity to use a learning tool in lectures. This could assist the professors when showing examples in lectures; explaining how a result table would look like when a specified query is executed, or explaining the logic of producing a result table. Two of the professors also state that a learning tool could be used by students at home, as an addition to lectures. This could help students learn the SQL syntax, as well as understand the logic by using more time with the learning tool.

The database professors state that they consider JOIN and grouping to be threshold concepts for students learning SQL. These are both part of SELECT statement, which suggests SELECT statements are an important part of SQL that should be covered by a learning tool. Table operations seem to be one of the last subject the professors teach in their database courses. We believe that knowledge of table operations is something that is not essential for the students understanding of SQL, and thus does not need to be implemented in a learning tool for novice students. We focus our prototype towards novice students learning SQL and thus want to focus on simple SELECT statements such as WHERE and JOIN, but propose that further development should be done to facilitate learning on a higher level of knowledge. When asked about students' threshold concepts, one professor answered "concepts". We are not sure what this answer relates to, and thus we choose to look at the answers from the two other professors to find students' threshold concepts.

5 Development of the prototype

5.1 Methodology

5.1.1 Preliminary development

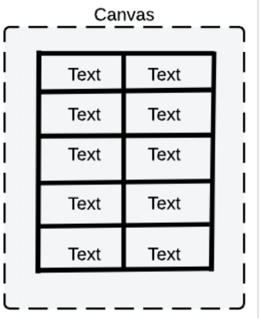
The goal when developing the prototype was to find a suitable method for visualizing SQL, to give the students the best possible learning outcome and for them to understand the underlying logic. To accomplish this, a subgoal was to create a number of different visualizations for later testing amongst a handful of students in a preliminary user test. Some of the ideas for the different visualizations were taken from other similar SQL learning tools/tutorials. As mentioned earlier, the SAVI application [7] gave a good overall idea of the visualization with animated rows from a original table to a resulting table. Brainstorming with the supervisor, Rune Hjelsvold, and fellow master students, was done in addition to come up with different visualization ideas.

An earlier study was done to identify what technology to use for displaying the animations [35]. Two HTML5 elements were researched, the Canvas element and the SVG element. The conclusion of the study is open, but suggests that it is slightly better to choose Canvas because of "*code simplicity, browser support and technical solutions*" [35]. This is why we chose to use the HTML5 Canvas element to display animations in the viSQLizer prototype.

A variety of JavaScript libraries were assessed to find a library compatible with HTML5 Canvas, to make it possible to animate HTML DOM elements. Efforts to create tables containing data in Canvas was done, and multiple JavaScript libraries for animation in Canvas were researched and tested, but drawing tables in Canvas containing the right number of rows and columns as in the database proved to be difficult. To create a table in Canvas, horizontal and vertical lines have to be drawn and text between these lines have to be added. A lot of work has to be done to create text objects for the table data in the database, and to place these objects at the right position relative to the database table and the Canvas. It is also difficult to add style properties to a table drawn on a Canvas. Figure 5 shows what a table could look like when drawn on a Canvas.

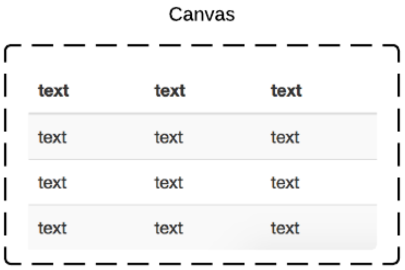
Because of the issues mentioned above, a decision was made to animate HTML DOM elements instead of drawing directly on the Canvas. A suite of modular libraries was found, CreateJS[1] with the modular libraries EaselJS and TweenJS. These libraries make it easy to animate HTML DOM elements using Canvas, which means that a table can be made using HTML and style sheets and placed on top of the Canvas. Figure 6 shows how a HTML table can look like.

[1] http://www.createjs.com



Text	Text
Text	Text
Text	Text
Text	Text
Text	Text

Figure 5: A table drawn on Canvas



text	text	text
text	text	text
text	text	text
text	text	text

Figure 6: A HTML table on Canvas

5.1.2 Preliminary design

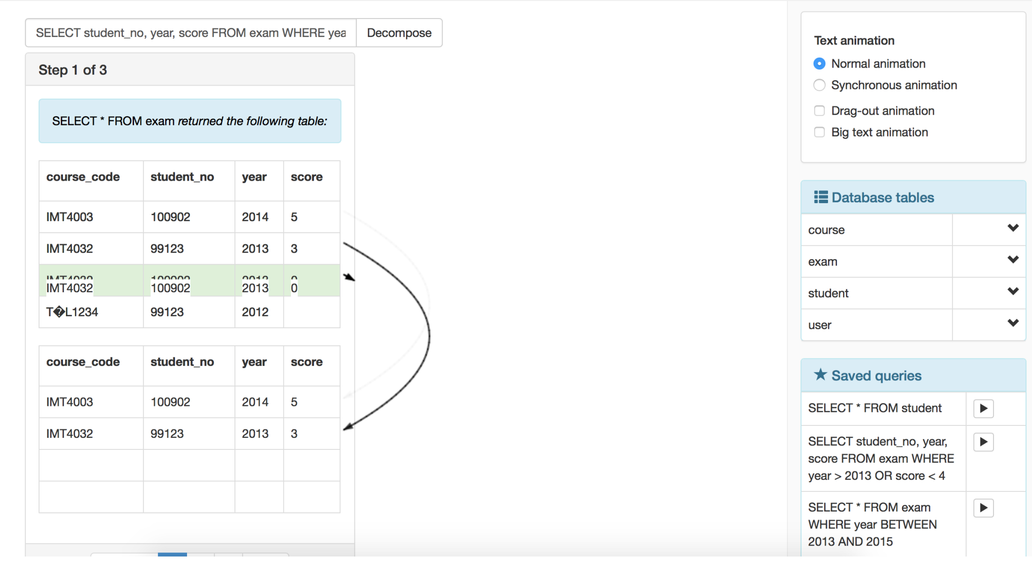
A prototype of a learning tool was developed, named viSQLizer, using animations and decomposition. The preliminary design of the prototype contained various GUI elements, highlighting on various table objects, as well as four different animation modes. The GUI elements, visual effects and animation modes will be described consecutively.

ViSQLizer GUI elements

Figure 7 shows the preliminary GUI for viSQLizer during animation from the original table to the result table. The GUI contains information about the database tables, saved queries, decomposition and table visualization. The various GUI elements are described below.

Text animation box

This box contains the four different animation modes provided in the preliminary design of the prototype. This box was provided to make it easy for the students to change between the different animation modes during the preliminary user test. This box were removed in the final version of the prototype.



SELECT student_no, year, score FROM exam WHERE yea

Step 1 of 3

SELECT * FROM exam *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3

Text animation

● Normal animation

○ Synchronous animation

□ Drag-out animation

□ Big text animation

Database tables

★ Saved queries

SELECT * FROM student ▶

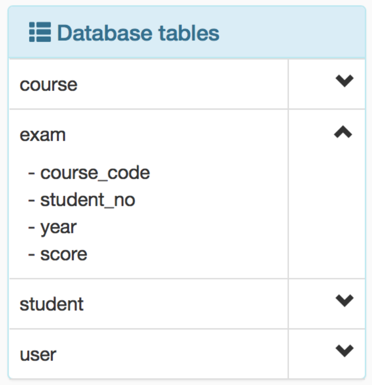
SELECT student_no, year, score FROM exam WHERE year > 2013 OR score < 4 ▶

SELECT * FROM exam WHERE year BETWEEN 2013 AND 2015 ▶

Figure 7: Preliminary GUI for viSQLizer

Database tables

A table is displayed, containing the information about the available database tables and column names. This was implemented to minimize the students' cognitive load, as discussed in Chapter 2.1, so they they do not have to remember this information when using the prototype. This table is shown in Figure 8.



Database tables

course

exam

- course_code
- student_no
- year
- score

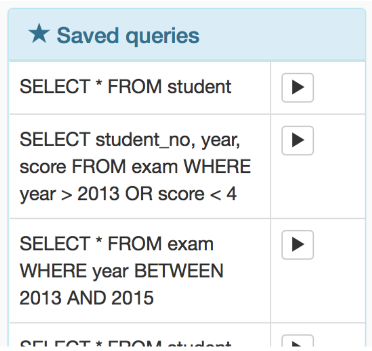
student

user

Figure 8: All available tables and column-names

Saved queries

As shown in Figure 9, another table is displayed, showing saved queries; queries that can be submitted, decomposed and visualized. These queries are various example queries that are supported by the prototype.



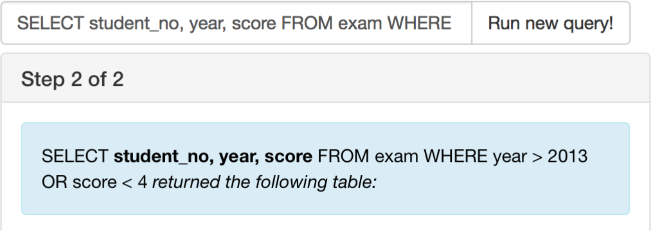
★ Saved queries

SELECT * FROM student	▶
SELECT student_no, year, score FROM exam WHERE year > 2013 OR score < 4	▶
SELECT * FROM exam WHERE year BETWEEN 2013 AND 2015	▶
SELECT * FROM student	▶

Figure 9: Table containing saved/sample queries

Decomposition information

This design for the decomposition information is shown in Figure 10. When a query is submitted, it gets decomposed into multiple steps. The number of steps that the decomposer creates are displayed, and the part of the query currently visualized is displayed in a blue information box. New text appearing in this information box throughout the decomposed steps is marked in bold, so that the student easily understands what is being visualized in the current step.



SELECT student_no, year, score FROM exam WHERE

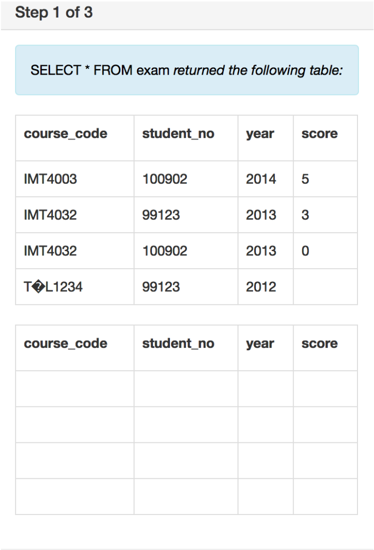
Step 2 of 2

SELECT **student_no, year, score** FROM exam WHERE year > 2013 OR score < 4 *returned the following table:*

Figure 10: Information about the query and decomposition

Table visualization

The GUI shows two different tables, as presented in Figure 7. The table at the top is the original table as represented in the database. This is from which the animation originates. If the current step contains a JOIN, multiple original tables are displayed, one for each table mentioned in the JOIN statement. Another table is placed underneath the original table and appears empty. This will contain the result data, when the animation is completed. This table is referred to as the result table. During animation, the cells are animated from the original table to the result table. Figure 11-14 shows the animation sequence, animating objects from the original table to the result table.



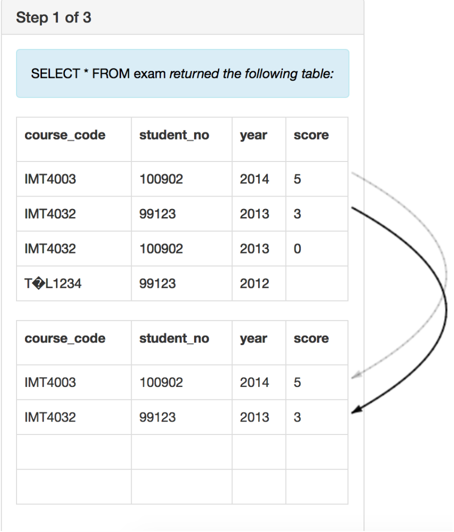
Step 1 of 3

SELECT * FROM exam *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

course_code	student_no	year	score

Figure 11: Animation sequence A



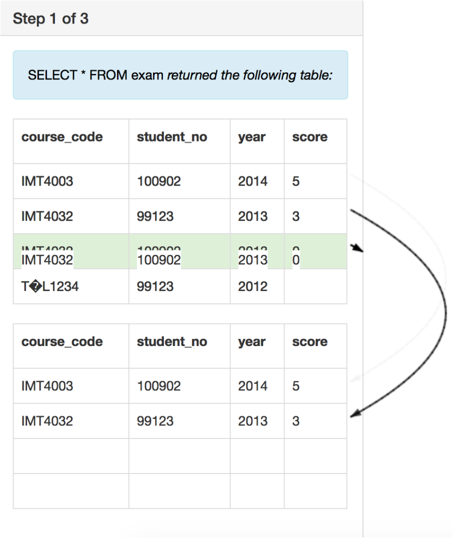
Step 1 of 3

SELECT * FROM exam *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3

Figure 12: Animation sequence B



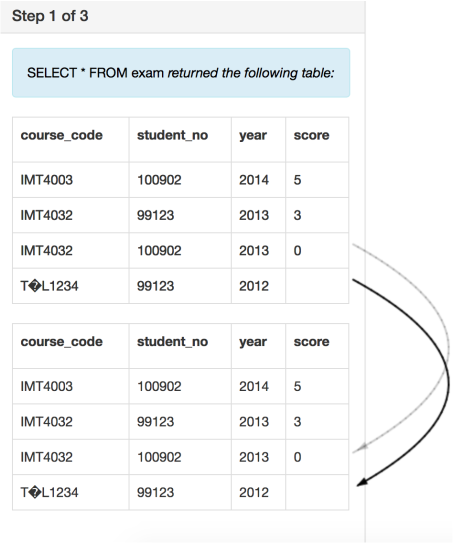
Step 1 of 3

SELECT * FROM exam *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3

Figure 13: Animation sequence C



Step 1 of 3

SELECT * FROM exam *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

Figure 14: Animation sequence D

Choosing decomposed step

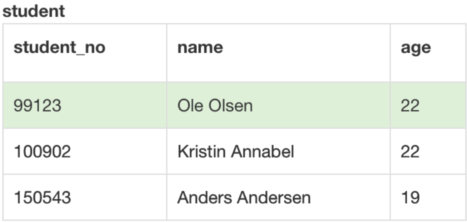
Buttons are provided to the user, to make it possible to switch between the steps in the decomposition. If the visualization only contains one step, these buttons are not present as they do not serve a purpose.

Highlighting table objects

Visual effects are used to give students a statically overview of what is happening in the learning tool at all times. Highlighting is added on the animated objects, the selected columns in the original table, and the origin of the cells in the result table. These three visualizations are described below.

Animated objects

During animation, the animated objects are highlighted in green in the original table. This is done to help the students understand what cells in the original table are being animated. The green color is chosen because of good visibility; no other justifications is done for this color choice. The green highlighting is removed when the animation is complete. An example of the highlighting during animation is shown in Figure 15.



student

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

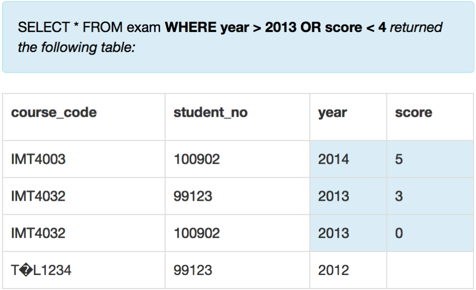
Figure 15: Highlighting of a row during animation

Selected columns

If the query contains an ON/WHERE/ORDER BY clause, the chosen columns are highlighted in blue. This color is the same as the information box containing the query, to make it easier for the students to understand the link between the ON/WHERE/ORDER BY clause in the query and the highlighted column. The blue color was chosen because wanted to use another color than the green color used on animated objects, and because it is the same color as the information box. An example of column highlighting using WHERE is shown in Figure 16.

Cell origin

When one of the cells in the result table is hovered over with the mouse pointer, the target cell is highlighted with a yellow color. The associated cell in the original table is highlighted as well,

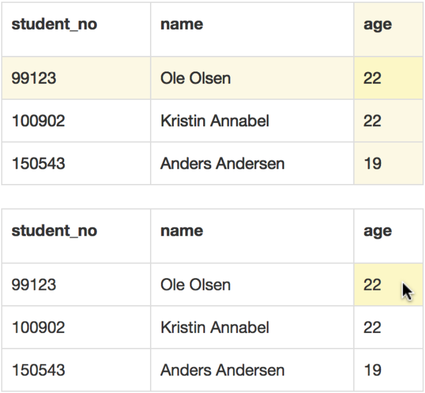


SELECT * FROM exam **WHERE year > 2013 OR score < 4** *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
T�L1234	99123	2012	

Figure 16: Highlighting on columns with WHERE in query

together with that specific column and row. This visualization idea is implemented to give the students a possibility to see where the various cells originate from and what column and row that cell belongs to, even after the animation has completed. The bright yellow color is chosen because it is easy to see and easy to differ from the green and blue color chosen for animated objects and selected columns. This highlighting is shown in Figure 17.



student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

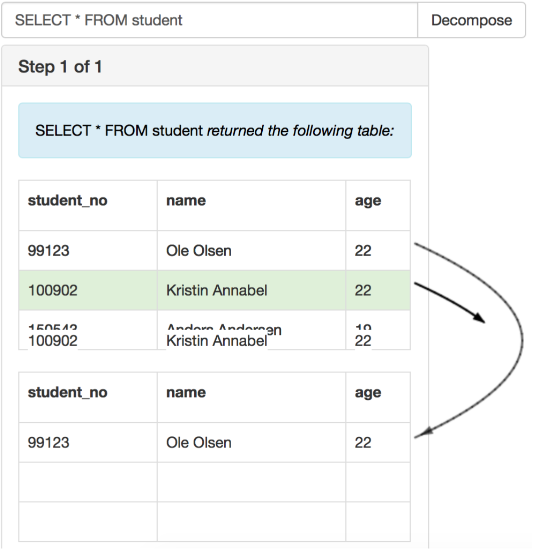
Figure 17: Highlighting on the cell, row and column

Animation modes

Four different animation modes were implemented in the preliminary design of the prototype; "normal", "synchronous", "drag-out" and "big text". This was done to determine if the students participating in a preliminary user test preferred some of the visualizations, to decide which animation ideas to use in further development of the prototype. The four different implemented animation mode are described below.

The "normal" animation mode

For the "normal" animation mode, the cells are animated from the original table to the result table, row by row, without any formatting of the text. The cells are animated vertically, dropping down from the original table and into the correct position in the result table. This is a straightforward visualization and is therefore included as one of the visualization ideas for the viSQLizer prototype. An arrow is drawn to the right of the table, to indicate the animation staring point and ending point. This is done to make the starting point and destination of the animated cells clearer for the students. Figure 18 is showing how the row is animated from the original table to the result table in the "normal" animation mode.



SELECT * FROM student

Step 1 of 1

SELECT * FROM student *returned the following table:*

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
100902	Kristin Annabel	22

student_no	name	age
99123	Ole Olsen	22

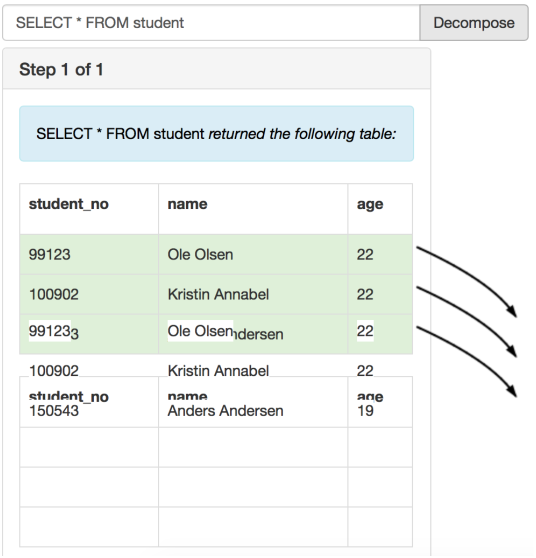
Figure 18: The "normal" animation mode

The "synchronous" animation mode

In the "synchronous" animation mode, all cells are animated at the same time. This visualization idea is included in the prototype because we want to determine if the students prefer an asynchronous or an synchronous animation. An arrow is displayed to the right of the table. An example of this animation mode can be seen in Figure 19.

The "drag-out" animation mode

In the "drag-out" animation mode, the cells are animated horizontally to the right of the original table, vertically down and horizontally left into the correct position in the result table. This visualization idea is implemented to determine if it could help students follow the animation flow better than with the arrows in both the "normal" and "synchronous" animation mode. This animation mode can be seen in Figure 20. During the preliminary user test, the students were also asked to test the "drag-out" animation mode together with the "synchronous" animation mode, as shown in Figure 21. This is done to determine if the students likes asynchronous or synchronous animation best in coherence with the "drag-out" animation mode.



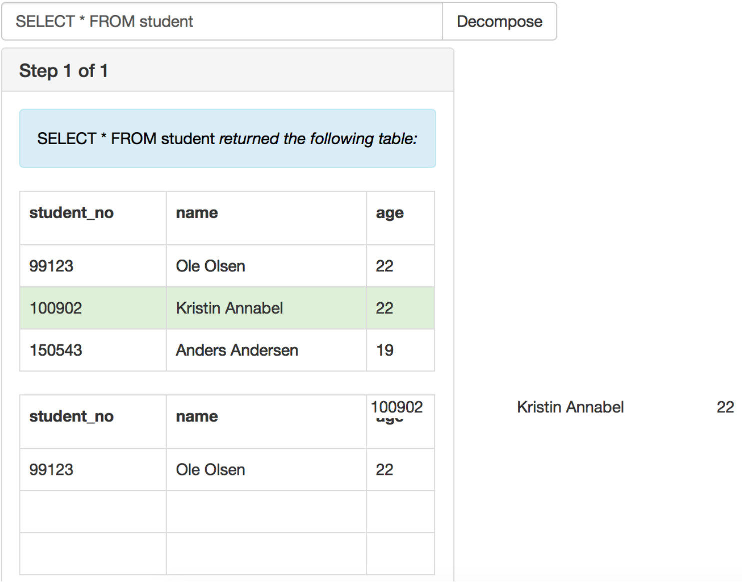
SELECT * FROM student

Step 1 of 1

SELECT * FROM student *returned the following table:*

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
991233	Ole Olsenndersen	22
100902	Kristin Annabel	22
student_no 150543	name Anders Andersen	age 19

Figure 19: The "synchronous" animation mode



SELECT * FROM student

Step 1 of 1

SELECT * FROM student *returned the following table:*

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

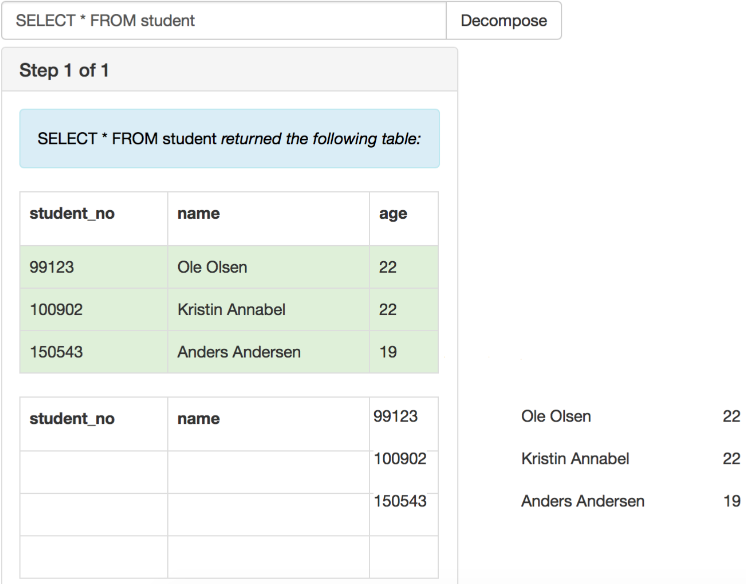
student_no	name	100902 age
99123	Ole Olsen	22

Kristin Annabel 22

Figure 20: The "drag-out" animation mode

The "big text" animation mode

In the "big text" animation mode, the cell text is enlarged before the cells are animated from the original table to the result table. The cells are then returned back to their original size. This visualization idea is implemented to determine if enhanced cell text during animation could help students see what is being animated and where the start and end position of the cells are located.



Step 1 of 1

SELECT * FROM student *returned the following table:*

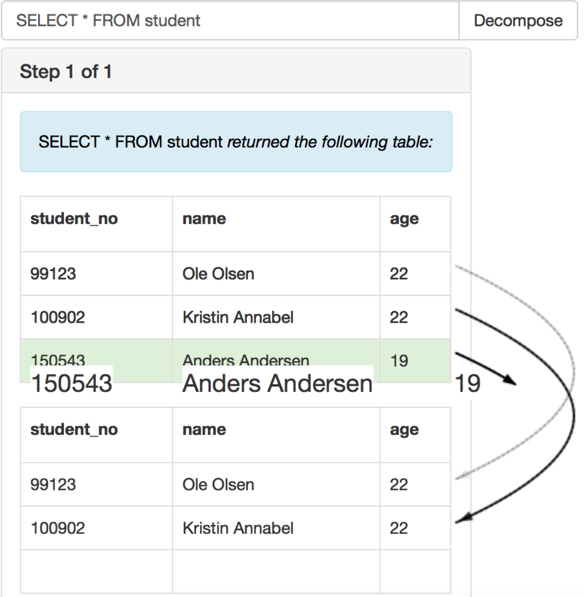
student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

student_no	name	
		99123
		100902
		150543

Ole Olsen	22
Kristin Annabel	22
Anders Andersen	19

Figure 21: The "drag-out" mode with "synchronous" animation

In the preliminary user test, the students were asked to view this animation mode together with the "normal" animation mode. This animation mode can be seen in Figure 22.



Step 1 of 1

SELECT * FROM student *returned the following table:*

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

150543 Anders Andersen 19

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22

Figure 22: The "big text" animation mode

5.1.3 Testing of the prototype

To get a better understanding of what students think about the different visualizations that were suggested, a preliminary user test was conducted. An email was sent out to the second year Game Development students at NTNU Gjøvik. These students are acquainted with SQL, as they have had an introductory database course in their previous semester. This class was selected because of their degree in development of software, and because game students often show a special interest in various visualizations and animations in tools and games. The class contains twenty-seven students; twenty-six male and one female.

The attending students were briefly introduced to the prototype and were informed about their participation role in the user test. A paper version of the survey, created with the online survey tool SurveyMonkey[2], was given out to all participating students as well as an URL for the online version of the survey. The questions from the survey can be viewed in Appendix E. The main role of the paper version was to minimize the students' cognitive load; the students could read the questions in the survey without switching between tabs in the browser window.

5.2 Results

5.2.1 User test

Six students attended the preliminary testing-session of the visualizations in the viSQLizer prototype. Two of the students chose to use the paper version of the survey, while three of the students chose to answer the online version of the survey. One of the attending students did not submit any answer to the survey, presumably because he did not bring a personal computer. All students answering the survey were male, with an age-span from 20 to 24.

General visualizations

The students were asked about the general elements of the visualization. As seen in Table 3, the students were positive to the general visual elements in the prototype. *N/A* in the table means that no answer were provided by the student.

"Normal" animation mode

The students tested the "normal" animation mode in the viSQLizer prototype. The results from the user test gives the overall idea that this animation mode is understandable and easy to follow. These results can be seen in Table 4.

[2] `https://www.surveymonkey.com`

	Attendants				
Question	1	2	3	4	5
Extent of usefulness of the partial query text	N/A	Very useful	Very useful	Very useful	In some extent
Mouseover shows origin data row and column	N/A	Good	Good	Good	Good
Go through steps multiple times	N/A	Good	Good	Good	Good
Intuitive link between ON-columns and query	N/A	Very intuitive	Somewhat intuitive	Very intuitive	Somewhat intuitive
Highlighting shows columns for ON-statement	N/A	Good	N/A	Good	Good
Marked columns for WHILE-statement	N/A	Good	N/A	N/A	Should use some other color
Intuitive link between WHILE-columns and query	N/A	Very intuitive	Somewhat intuitive	Very intuitive	Somewhat intuitive

Table 3: Survey results for general visualizations

	Attendants				
Question	1	2	3	4	5
Text animation speed	Good	Good	Good	Good	Good
Highlighting of animated row	Good	Good	Good	Good	Good
Usage of arrows	Unnecessary but not disruptive	Good	Good	Good	Good
Visualization of arrows	Disappears too late	Good	Good	Good	Good

Table 4: Survey results for "normal" animation-mode

"Synchronous" animation mode

As seen in Table 5, the results indicate that the previously mentioned "normal" animation mode was more understandable than the "synchronous" animation mode. Their attitude towards the synchronous arrows were spread, but indicates that they were clumsy/hard to follow and unclear.

"Drag-out" animation-mode

The results from the user test of the "drag-out" animation mode, shown in Table 6, indicate that the students did not mind the absence of arrows and that the "drag-out" animation mode were more understandable than the "normal" animation mode. The students did try the drag-out animation mode together with the synchronous animation mode, but stated that this type

	Attendants				
Question	1	2	3	4	5
Text animation speed	Good	Good	Good	Good	To fast animation
Synchronous better than normal	No	No	No	Both	No
Usage of arrows	Looks only at text	Clumsy	N/A	Good	Hard to follow
Arrow path	Unclear	Unclear	Unclear	Good	Unclear

Table 5: Survey results for "synchronous" animation-mode

of animation was unclear, and that it was difficult to follow the animated cells when multiple original tables were present, as is the case for JOIN.

	Attendants				
Question	1	2	3	4	5
Understanding without arrows	Yes	Yes	Yes	Yes	Yes
Drag-out better than normal	Yes	Yes	N/A	Yes	Yes
Animation of data	Good	Good	Good	Good	Good
Straightforward with "synchronous"	Easier with drag-out. Too much happening at once	Not when data are switching places	Some	Yes	Not when rows are overlapping
Animation of data with "synchronous"	Too fast. Should go slower out of the table	Unclear	Unclear	Unclear. Too fast	Too fast

Table 6: Survey results for "drag-out" animation-mode

"Big text" animation mode

As seen in Table 7, the students stated that there were no difference in how they perceived the path of the cells in the "big text" animation mode. Most of the students seemed satisfied with this type of visualization of the cell text, as long as they did not have to read the text during the animation.

User test conclusion

The preliminary user test was done to get some insights into how the students perceived the different visualizations and the solution as a whole. The user test can be looked upon as a small feedback between the two development iterations. Based on the results from the preliminary user test as well as our own previous suspicions and beliefs towards the different visualizations, some decisions were made in regards to what visualizations to keep and what visualizations to remove in the further development of the prototype.

	Attendants				
Question	1	2	3	4	5
More clear path of data	Not really	No difference	No difference	No difference	Sees text better
White box visualization	Good	Did not notice	OK	Did not notice, but seems useful	Good, but could need a black outline
Visualization of data	OK, but too fast if one is too read the text	Unclear	Good	Good	Good

Table 7: Survey results for "big text" animation-mode

General visualizations

The results show that overall the students are satisfied with the different general visualizations in the prototype. The results show that the students liked the mouse-over highlighting for the cell origin and the highlighting for the selected columns, and the students stated that the link between the query text and the highlighted columns were intuitive/somewhat intuitive. These results made us decide to keep the visual features in the further development of the prototype.

"*Normal*" *animation mode*

The results show that the students are satisfied with the "normal" animation mode. The students expressed that the animation speed and the highlighting of the animated objects is good. Because of the results from the other animation modes, described below, we did not keep this animation mode in the further development of viSQLizer.

"*Synchronous*" *animation mode*

Results showed that four of the five students participating in the user test did not like the "synchronous" animation mode better than the "normal" animation mode, and that the students did not like the use of arrows in the "synchronous" animation mode. The students stated that the arrows were confusing in this animation mode, especially when multiple original tables were present. Because we agreed with the students on these facts, we did not keep this animation mode in the further development of the prototype.

"*Drag-out*" *animation mode*

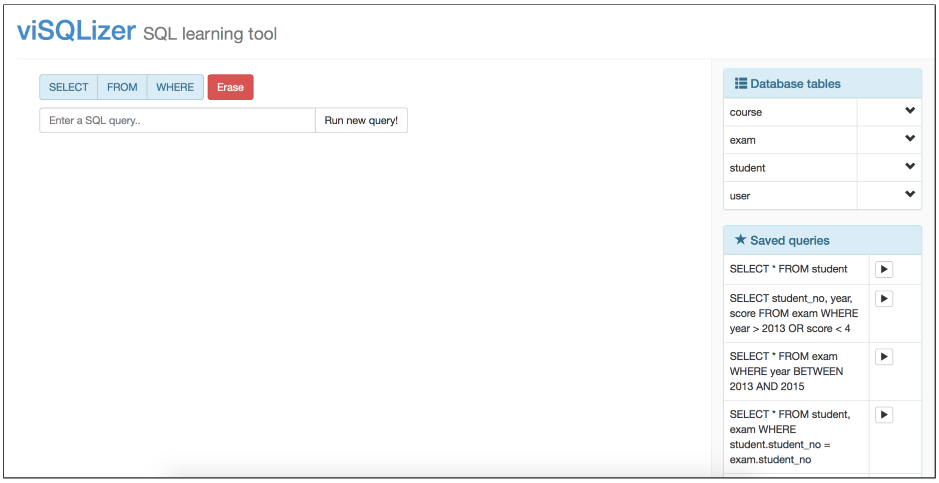
The results show that the students are satisfied with the visualizations in the "drag-out" animation mode. We believe that the "drag-out" mode makes it clearer to the student what cell is being animated, and where it is placed. This was also stated by some of the students during the user test. Because of this, the "drag-out" animation mode is kept in further development of the prototype.

"*Big text*" *animation mode*

The results show that three students did not notice that the text got bigger during the animation, but that it was not perceived as a negative element. We think that this feature could help some students to better notice the animated text. Because the "big text" animation mode had no obvious negative effect, this feature is kept in the further development of the prototype.

5.2.2 The prototype

Design



SELECT * FROM student

SELECT student_no, year, score FROM exam WHERE year > 2013 OR score < 4

SELECT * FROM exam WHERE year BETWEEN 2013 AND 2015

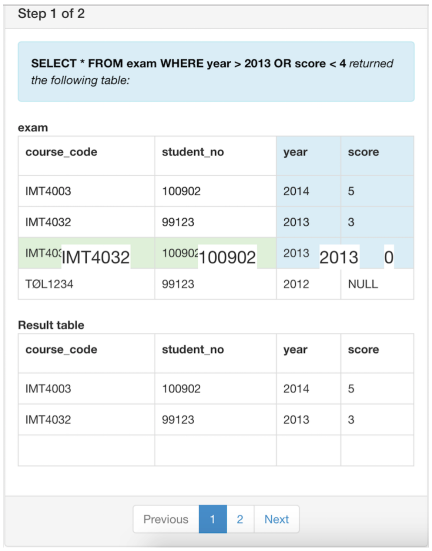
SELECT * FROM student, exam WHERE student.student_no = exam.student_no

Figure 23: The front-page of viSQLizer

The viSQLizer prototype uses the design implemented in the Decomposer project [12]. The design is shown in Figure 23. Buttons were added in viSQLizer to allow the students to quickly create a query by using the most frequent operations SELECT, FROM and WHERE. The students can also choose to quickly erase the written query from the input field. These buttons were added to optimize the effectiveness of using the learning tool. The final animation modes that were selected, as shown in Figure 24, were "big text" combined with "drag-out" animation mode.

The columns that are part of an ON/WHERE/ORDER BY clause are highlighted. Two students participating in the preliminary user test stated in the interview that the column highlighting could be clearer, which made us decide to highlight the column header as well as the cells underneath the column to make it clearer to the user what columns and cells are part of the clause.

Headings representing the table names were implemented above the original table(s) and the result table to make it clear to the user what tables are present in the visualization. If the upper table contains the result from the previous step, this table has no name, as it is not a table with data exactly as represented in the database. This can be seen in Table 25, where the last step of a decomposed query is being visualized.



Step 1 of 2

SELECT * FROM exam WHERE year > 2013 OR score < 4 *returned the following table:*

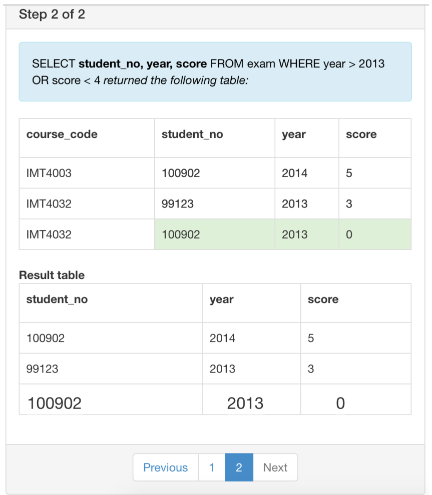
exam

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT403 IMT4032	100902 100902	2013 2013	0
TØL1234	99123	2012	NULL

Result table

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3

Figure 24: A row animating from original table to result table



Step 2 of 2

SELECT **student_no, year, score** FROM exam WHERE year > 2013 OR score < 4 *returned the following table:*

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0

Result table

student_no	year	score
100902	2014	5
99123	2013	3
100902	2013	0

Figure 25: Final step of the query, final row animating

HTML/CSS classes and IDs

To be able to highlight specific columns/cells in the tables and marking the animated cells correctly, various HTML/CSS classes and IDs are used. These classes are described later and can be organized into three groups:

- **Classes for highlighting selected columns:** "*where*", "*onColumn*" and "*orderByColumns*".
- **Classes for animating cells:** "*animThis*" and "*notInUse*".

- **Classes for marking an animated cell:** "*usedInRow*", "*usedInRow_#number#*", "*duplicate*" and "*used*".

Architecture

The architecture for the viSQLizer prototype consists of a server side implementation in PHP and a client side implementation in HTML5/JavaScript. The architecture is visualized in Figure 26, and explained below.

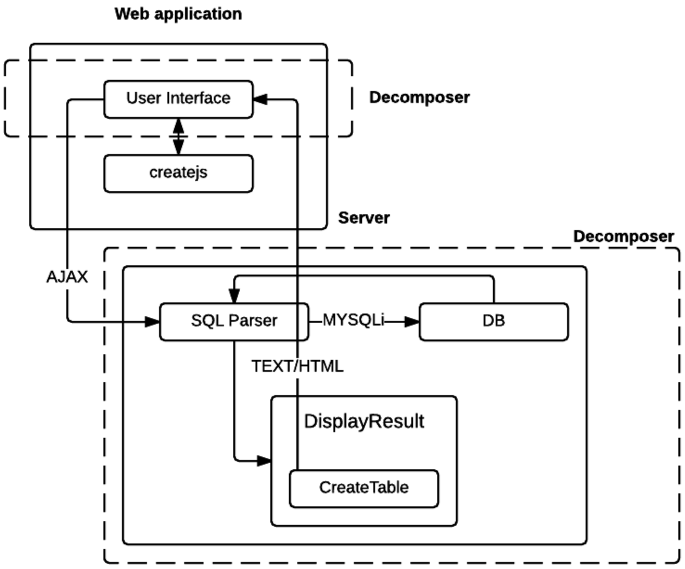


Figure 26: The architecture of the viSQLizer system

- Web application – The application was published online to make it easy to access for professors and students on their personal computers. The web application consist of HTML5/CSS, JavaScript/jQuery and the JavaScript-libraries EaselJS[3] and TweenJS[4]. The animations and all the events are handled with JavaScript/jQuery. HTML classes and id attributes are used to locate various elements for the visualization. HTML classes are set on HTML elements server side and used with CSS and JavaScript client side or set client side during animation.
- Server – The server is running PHP 5.6.2, with MySQLi connection to the mySQL databases. The use of PDO was considered, but not implemented as MySQLi was already implemented in the Decomposer application [12] (see information below for information about Decomposer). Requests from the web application to the server are sent through Ajax and the HTTP Request

[3] http://www.createjs.com/easeljs
[4] http://www.createjs.com/tweenjs

Method POST. The data that is sent from the server to the web application is pure text/HTML via PHP echo. The tables are created on the server, and sent with HTML mark-up to the web application DOM.

- SQL Parser – For parsing the SQL, a pure PHP SQL parser for MySQL has been included, PHP-SQL-Parser [13]. The parser divides the query into smaller parts and acquire the result table for the specific part of the query from the database. The query is divided into smaller parts by creating an individual query for each operation and places these in their natural order in an array.
- Decomposer – The Decomposer is developed by Brovold, Johnsen & Folland [12]. The viSQLizer prototype uses the same design as the Decomposer, and builds on the code created in the Decomposer project. All code related to the process of decomposing the queries into multiple steps was created in the Decomposer project. The Decomposer displays only the result table for each step without any visualization of the original table(s) or animations.
- viSQLizer databases – The solution contains two databases. One for storing information about the saved queries, and one for the available data sets. User restrictions are set so that a student cannot delete or change the provided sample data.
- DisplayResult – DisplayResult is a PHP method inside the Parser class. The method does two important jobs:
    1. It checks what has been visualized in the previous steps, and highlights the new part of the current query.
    2. It displays all the surrounding elements, like the highlighted query, the number of steps, and the buttons for each step. To display the original table and result table, the function make_table is called inside the DisplayResult method.

The flow on the viSQLizer client- and server-side is shown in figure 27. The query is sent through a post request to the server, getting information from the database and parses the result. The result is then placed in a HTML table which is sent back to the client as text/HTML content. The HTML table is placed in the HTML class "*streammode-panel*", and the table cells are animated. More information for the server and client side implementation can be found below.

Server-side technology

During development of the viSQLizer prototype, a number of different features was implemented and some issues had to be dealt with on the server-side implementation. These features and issues are listed below and the implemented solution to these issues will be described later in this section.

- *Original tables:* The data in the original table is changing throughout the decomposed steps. In the first step, the original table should contain the data exactly as represented in the

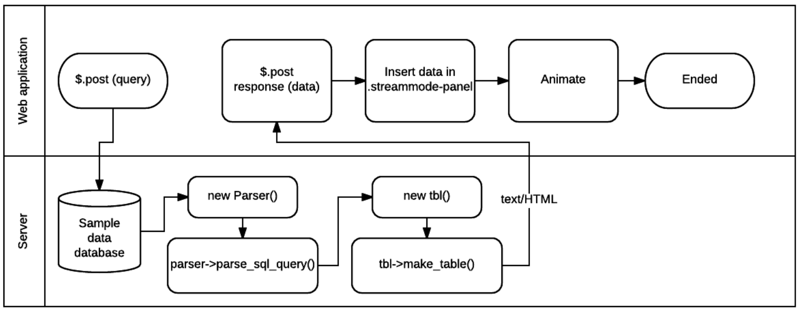


Figure 27: The flow of the viSQLizer system

database. In all but the first step, the original table should contain the result table from the previous step. If a decomposed step contains a JOIN, two original tables should be displayed, containing the data from the database.

- *Column highlighting:* If the current decomposed query contains an ON/WHERE/ORDER BY clause, the selected columns should get highlighted in the visualization. We needed to determine which columns is part of such a clause to mark these with HTML classes for easy access client side.
- *Column names:* A problem occurred when JOINing multiple tables contained the same column name. The reason for this is that an array can not hold two identical keys, and the array containing the result data from the JOIN is a key value array where the key is the column name and the value is the data.

All these issues/features are a part of the make_table function server side. The make_table function gets a number of parameters, and the parameter list is briefly described below:

```
function make_table($result=array(), $step, $tables=array(),
    $whereColumns=array(), $onColumns=array(), $onOrderBy=array(),
    $previousTable=array())
```

- *result:* This array contains the data for the result table for the current step. The array is a key-value array, where the column names in the table is the key, and the value is multiple other arrays, containing the data for all cells in the table.
- *step:* This variable contains the current decomposed step encoded as step number. This is used when accessing some arrays that are created by the parser [13].
- *tables:* This array contains all the tables referred to in the submitted query. This is used when displaying the original table as represented in the database.
- *whereColumns:* This array contains all the column names that are referred to in the WHERE clause in the current part of the query. This is used to determine which columns are part of the WHERE clause which are going to be highlighted in the visualization. If a column name

is present in this array, this column gets the HTML class "*where*" in the table.

- *onColumns:* This array is used as the whereColumns array, but for the columns that are part of the ON clause in a JOIN. These columns get the class "*onColumn*" in the HTML table.
- *onOrderBy:* This array is used as the two previous arrays, but for the columns that are part of the ORDER BY clause. These columns get the HTML class "*orderByColumn*" in the table.
- *previousTable:* This array contains the data for the result table from the previous step. This array is used for the original table after the data from the database has been shown in the first step.

Original tables
The original table could could be displayed in three different ways: the table as presented in the database, two tables as presented in the database when the query contains JOIN, or the result table from the previous step.

The first type of original table is shown in the first step of all query decompositions that does not contain a JOIN. The table name is used in a simple query that fetches all data from that particular table in the database.

```
//Get data for original table, directly from database
if($step <= count($tables[0])){
    $counter = $step-1;
    $query = "SELECT * FROM " . $tables[0][$counter];
    $tableResult = mysqli_query($con, $query);

    for($j = 0; $j < $tableResult->num_rows; $j++){
        $finalTbResult[] = mysqli_fetch_array($tableResult);
    }
    echo "<table>";
    for($m=0; $m < count($finalTbResult); $m++){ ?>
        echo "<tr>";
                for($n=0; $n<count($finalTbResult[$m])/2; $n++) {
                        echo "<td>".$finalTbResult[$m][$n]."</td>";
                }
                echo "</tr>";
        }
    echo "</table>";
}
```

When joining tables, the two tables that are joined are displayed as original tables as presented in the database. When JOIN is present in the query, this is the first step in the decomposition. This is done in the same way as in the previous example, but to display all tables the implementation loops through each table to display it.

```
//Step = the number of tables in the query + 1
for($i = 0; $i < $step; $i++){
    if($i < $numberOfTables){
        $query = "SELECT * FROM " . $tables[0][$i];
        $tableResult = mysqli_query($con, $query);
        for($j = 0; $j < $tableResult->num_rows; $j++){
            $finalTbResult[] = mysqli_fetch_array($tableResult);
        }
        echo "<table>";
        for($m=0; $m < count($finalTbResult); $m++){ ?>
            echo "<tr>";
                        for($n=0; $n<count($finalTbResult[$m])/2; $n++) {
                                echo "<td>".$finalTbResult[$m][$n]."</td>
                                    ";
                        }
                        echo "</tr>";
                }
        echo "</table>";
    }
}
```

As described above, the first step consists of one or multiple original table(s) with objects from the database and a result table containing e.g. the result from a JOIN. All following original tables contain the result from the previous decomposed step. This was done to make it easier for the students to understand how the results in the decomposed steps are related to each other.

```
if(($step > (count($tableName[0])+1)) && (!empty($prevTable))){
    foreach($previousTable[$step-1] as $output) {
        if ($output[’type’]==’table’) {
            $prevTable = $output[’contents’];
        }
    }
    echo "<table>";
    for($i=0; $i < count($prevTable); $i++){
        echo "<tr>";
        for($j=0; $j<count($prevTable[$i])/2; $j++) {
            echo "<td>".$prevTable[$i][$j]."</td>";
        }
        echo "</tr>";
    }
    echo "</table>";
}
```

Column highlighting

The array created by the PHP SQL Parser [13] contains information about which columns in the table are part of an ON/WHERE/ORDER BY clause. This information has to be retrieved to highlight these columns in the visualization.

```
$this->onColumns[] = $this->parser->parsed[’FROM’];
$this->whereColumns[] = $select["WHERE"][$k]["no_quotes"]["parts"];
$this->onOrderBy = $this->parser->parsed[’ORDER’];
```

In the make_table function, we loop through each column in the original table. If the column name (key) is equal to a string present in one of the onColumns/whereColumns/onOrderBy arrays, a class is set for this column in the HTML tag <th> in the original table. This makes it possible to hook on to this HTML element later on the client side to add highlighting on these columns.

```
//Set the WHERE class name
$whereClassName = "";
for($r = 0; $r < count($whereColumns); $r++){
    if(count($whereColumns[$r]) > 1){
        //Needs to be done because the array is different if there are
           multiple tables in this array
        $numInArr = 1;
    }
    else {
        $numInArr = 0;
    }

    if($keys[$i] == $whereColumns[$r][$numInArr]){
        $whereClassName = "where";
    }
}

//Set the ON class name
$onClassName = "";
if(!empty($onColumns[0][$step][’ref_clause’])){
    for($r = 0; $r < count($onColumns[0][$step][’ref_clause’][0][’
       sub_tree’]); $r++){
        if (($pos = strpos($onColumns[0][$step][’ref_clause’][0][’
           sub_tree’][$r][’base_expr’], ".")) !== FALSE) {
            $thisOnColumn = substr($onColumns[0][$step][’ref_clause’][0][
               ’sub_tree’][$r][’base_expr’], $pos+1);
            if($keys[$i] === $thisOnColumn){
                $onClassName = "onColumn";
            }
        }
    }
}

//Set the OrderBy class name
$OrderByClassName = "";
if(!empty($onOrderBy)){
    for($a = 0; $a < count($onOrderBy); $a++){
        if($keys[$i] === $onOrderBy[$a]["base_expr"]){
            $OrderByClassName = "orderByColumn";
        }
    }
}

//Create the header for the original table with appropriate classes.
   Classes are empty if the column is not a part of the ON/WHERE/ORDER BY
    clause.
```

```
echo '<th class="' .$whereClassName . ' ' . $OrderByClassName . ' ' .
    $onClassName . ' '. $finfo->name . '" id="' . $finfo->orgtable . '"><p
    >' . $finfo->name . '</p></th>';
```

Column names

To fix the problem with column names not appearing after a JOIN, we had to get the query from the current step and run it against the database to get the information about the wanted column by using mysqli_fetch_field_direct(). This mySQLi method returns information about the table schema, as table name, column names etc. The result array could look different depending on the placement of the duplicate keys and the number of duplicate keys. Because of this, we had to both check if a array key was not set and if the array key was a number instead of a column name.

```
//We get the query from the current step
$var = (string)$previousTable[$step][1]['contents'];
//Remove the words and HTML tags that should not be a part of the query
$removeWords = array("returned", "the", "following", "table:", "<b>", "</
    b>", "<i>", "</i>");
$tempString = str_replace($removeWords, "", $var);
$tempQueryOne = strstr($tempString, 'FROM');
$tempQuery = $tempQueryOne . " LIMIT 1";
$tempResult = mysqli_query($con, $tempQuery);
mysqli_close($con);

//We loop through each key in the result array
for($i=1, $c=0; $i <= count($keys); $i+= 2, $c++) {
    //If the key is not set in the array
    if($i == count($keys) && !isset($keys[$i])){
        $finfo = mysqli_fetch_field_direct($tempResult, $c);
        echo '<th>' . $finfo->name . '</th>';
    }
    //If the key is an integer instead of a column name
    else if(is_int($keys[$i])){
        $finfo = mysqli_fetch_field_direct($tempResult, $keys[$i]-1);
        echo '<th>' . $finfo->name . '</th>';
    }
}
```

Client-side technology

As mentioned earlier, createjs[5] makes it easy to animate HTML DOM elements inside the HTML5 Canvas element. This is done by creating a new *createjs.DOMelement*, and then adding this *createjs.DOMelement* to the stage. The stage is an container containing a display list of all elements that are added to the stage. This display list will be rendered and redrawn for every tick, "a heartbeat broadcast at a set interval" [36]. The Canvas is resized to be the same size as the HTML div element containing the original table(s) and the result table, and the canvas and the

[5] http://www.createjs.com

div element is placed at the same screen coordinates. The *createjs.DOMelement* is placed above the canvas in z-direction, to make all buttons clickable and all content interactable for the user. An illustration of the placement and the various layers of the elements are shown in Figure 28. The implementation for the Stage and Canvas setup can be seen in Appendix N.

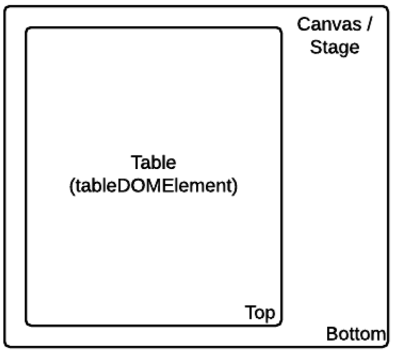


Figure 28: Screen layout

The flow of the client-side implementation is presented in Figure 29. A target cell in the original table is located for animation, the start and end position for the animation is calculated, the animation is done and the cell is marked as animated. The various processes and decisions are marked with the letters A through E, and are described below. See Appendix N for implementation of each process.

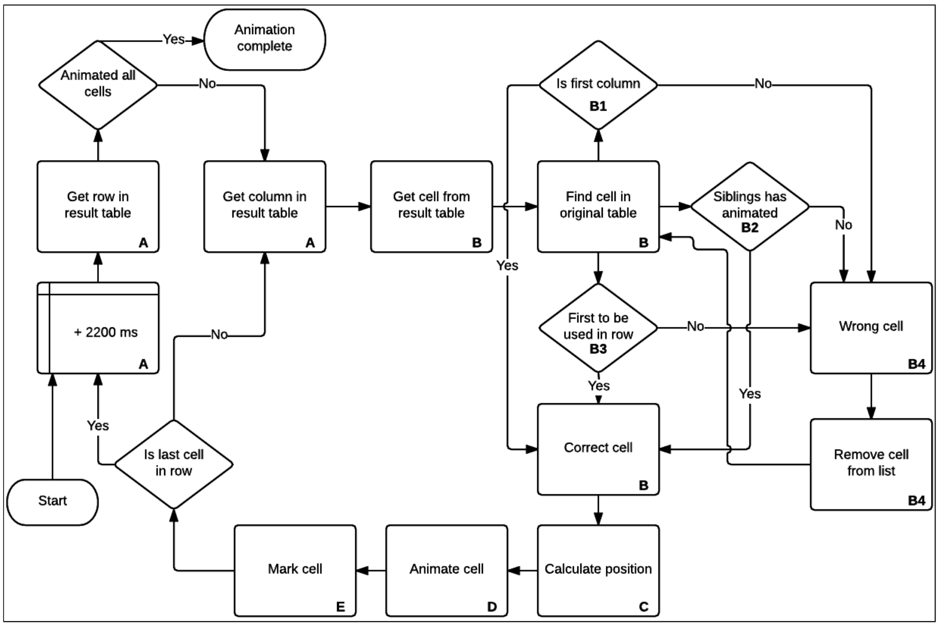


Figure 29: Flowchart for client side implementation

Process A: Looping through rows and columns

When the animation start, the first row in the result table is animated, then the second etc. Two for-loops are looping through each row and each column in the row. This makes it easy to handle one cell at a time in the implementation.

Process B: Locating target cell for animation

The result table object initially contains all data from the database. This data is hidden so that the result table appears empty for the students. This is done to make it easy to determine which cell is to be animated from the original table to the result table. To locate the target cell to animate, the first cell in the result table is found (top left corner of the table). This cell text is used to locate the target cell in the original table(s). But multiple cells could contain the same text and multiple original tables could exist.

All columns in the result table have an HTML ID, representing the table origin for that column. This is done to easily determine which original table the cell in the result table originates from, if there are multiple original tables present. All columns in the original table(s) also contains an ID with the name of that original table. These IDs have to match for the cell to be a possible target cell for the animation.

The method *checkIfDuplicatedData* goes through each cell in the list, one by one, and determines if this is the target cell we want to animate or not. There are four scenarios, deciding if the cell is the target cell or not;

B1 If this cell has the correct table name (ID), is located in the first column of the original table, and no other cells have been animated in this row, it is the correct cell to animate.

B2 The selected cell is checked to see if it has any siblings that has been animated. These siblings are any cell in the same table row. If this is true and the table name is correct, this cell has the correct position to animate.

B3 A cell could be the first to be animated in a row, even though it is not located in the first column in that row. If this is the case, the previous cell in the original table is not the same as the previous cell in the result table. If this is true, and the table name of this cell is correct, this is the correct cell to animate.

B4 The last possibility for the cell is that it is not the correct cell to animate. We have checked if the cell is the first cell in a row (B1), if any of its siblings has been animated (B2), and if it is the first to be animated in a row, even though it is not located in the first column in the row (B3). One of these conditions have to be met to animate a cell, and since this cell did not fulfill any of the conditions it is not the correct one. This cell seems to be a cell with the same name, but is not the correct one. In this case, the cell gets the class "notInUse" and is filtered out of the list with possible matches. At this point, the test continues to the new first cell in the list (which was the second from the beginning), and checks if this is the target cell to animate. This continues until the next target cell has been found. This cell is animated (but waiting until all the cells in that row can be animated at the same time), and we find the next cell in the result table.

Process C: Find cell position

The X and Y position of the cell in the result table had to be calculated, to decide the destination of the target cell after animation. Then the X position had to be calculated depending on the placement of the other cells in that table row. This is because the cells get scaled up in size, and should not crash with their sibling cells. An illustration of this issue is shown in Figure 30.

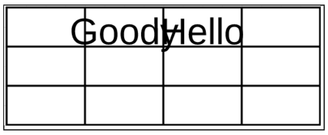


Figure 30: Issue when scaling up cell text

To avoid crashing sibling cells when resizing the cell text, values from the previous animated cell is used. This is because this cell is placed directly to the left of the target cell in the original table. The width of the previous cell is multiplied by 1.5, and added to the additional X positions calculated for the previous cell. This solution has been illustrated in Figure 31.

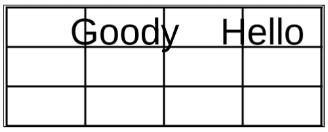


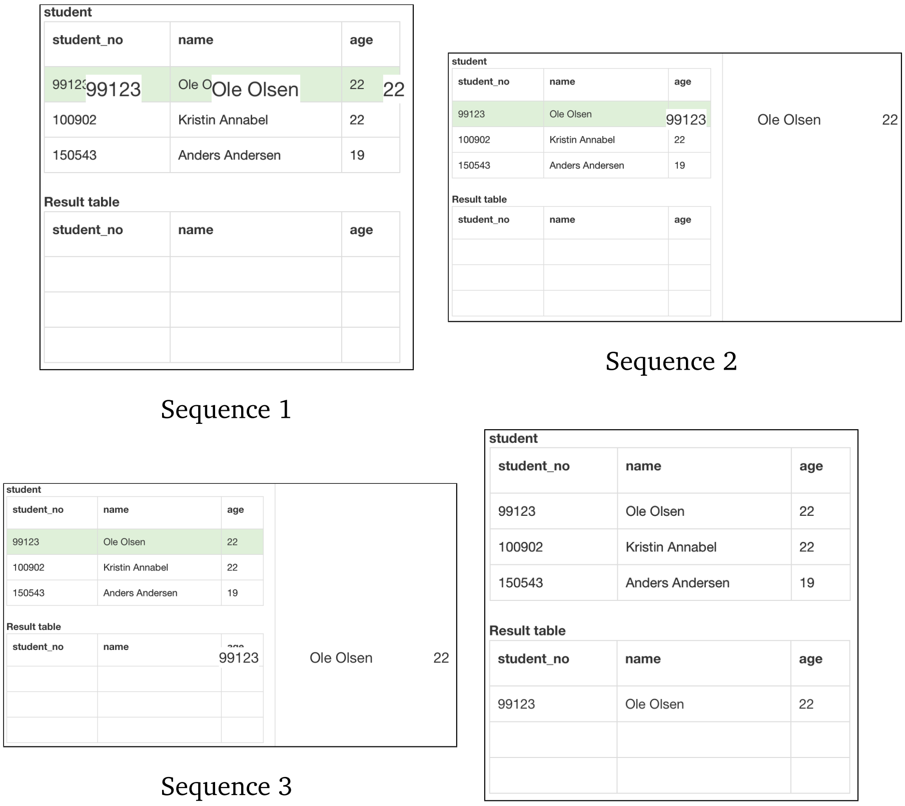
Figure 31: Solution when scaling up cell text

Process D: Animation

The target cell gets animated when the different positions for the animation have been calculated. The animation sequence is illustrated in Figure 32. The target cell is:

- **Sequence 1:** Scaled up and moved to the calculated X position
- **Sequence 2:** Animated the additional X coordinates
- **Sequence 3:** Animated down to the correct Y position in the result table
- **Sequence 4:** Scaled down and placed in the correct X position

When the animation is completed, the cell is moved back to its original position in the original table because it might need to be animated again later. The target cell text in the result table gets visible to the user, giving the impression that the cell stays in the result table.



student

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

Result table

student_no	name	age

Sequence 1

student

student_no	name	age
99123	Ole Olsen	
100902	Kristin Annabel	22
150543	Anders Andersen	19

99123 Ole Olsen 22

Result table

student_no	name	age

Sequence 2

student

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

Result table

student_no	name	age

99123 Ole Olsen 22

Sequence 3

student

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

Result table

student_no	name	age
99123	Ole Olsen	22

Sequence 4

Figure 32: Animation sequence

Each cell is set to animate with a waiting time for 2200 ms for the first row. As such, all the cells in one row are animated at the same time. When the next row is active, the number 2200 is added to the waiting time. The number 2200 were selected by trial and error; the waiting time were not to be to slow or to fast, and the students should be able to watch the animation of a row and be able to see it finish before the next row is animated.

600 ms is added to the waiting time, which is the time it takes to scale the cells up and down during animation. This number was determined through trial and error, and was chosen because it fits well with the animation speed of moving the cells.

Process E: Mark animated cell correctly

Whenever a cell has been animated, it is crucial that this cell is not animated again. This, however, depends on the query. If the query creates a cartesian product, all cells will be animated multiple times and this case also had to be covered by the implementation. Each cell is marked using HTML classes to indicate if the cell is to be animated again or not.

When the animation is completed, the cell gets the HTML class "*usedInRow*" + the row number in the result table. The parent in the DOM tree also gets the HTML class "*usedInRow*". This was done to avoid the same cell being animated multiple times during animation of one row and to help locate the target cell in scenario B2 and B3.

The cell is marked differently depending on the number of original tables present in the view, the number of duplicates in the original table(s) and the type of JOIN in the submitted query. These scenarios are shown in Figure 33, and described below.

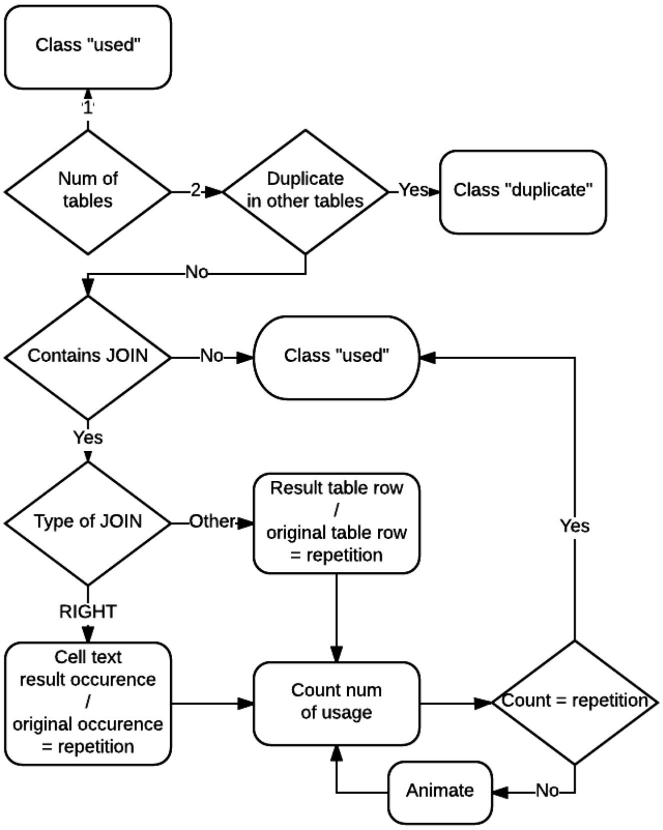


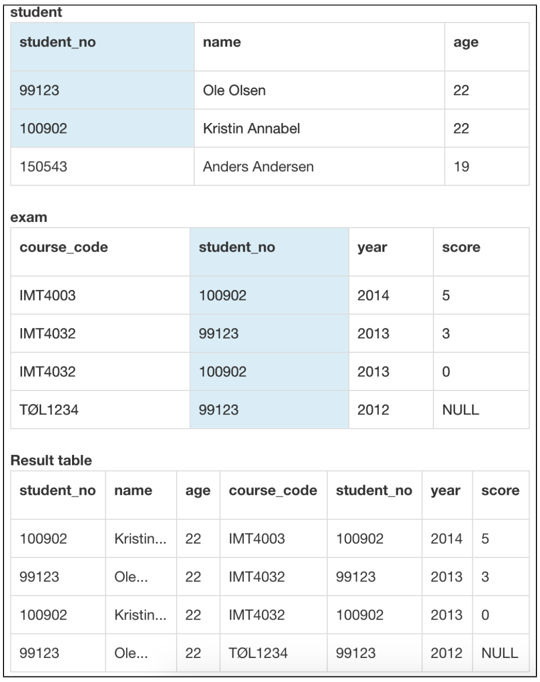
Figure 33: Flow chart for cell marking

One original table: If only one original table is present in the view, the animated cell gets the class "*used*" and is no longer available in the list of cells matching the result table cell text.

Multiple original tables: If there are multiple original tables present, we count the number of duplicates of that cell text in the original table that the animated cell originated from and compares this number to duplicates in all original tables. If there are duplicates in the other tables as well, the cell is given the class "*duplicate*" and can be used again depending on the query.

Query containing a JOIN: If the query does not contain a JOIN, the marking ends and the cell is given the class "*used*". If there are no duplicates in other tables, and if the current query contains a JOIN, we calculate the number of times the cell is going to be animated, depending on the query:

- **RIGHT JOIN:** If the query contains a RIGHT JOIN, the number of cells matching the text in the result table are divided by the number of cells matching the text in the original table. Figure 34 shows an example of a RIGHT JOIN, with the query "*SELECT * FROM student RIGHT JOIN exam ON student.student_no = exam.student_no*". In this example, we see that the student_no "100902" occurs two times in the first column in the result table. This column is originally from the "student" table where the student_no "100902" only occurs once. We do the math, 2 occurrences in the result table divided by the one occurrence in the "student" table; 2/1 = 2. The cell "100902" in the "student" table have been animated two times.



student

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

exam

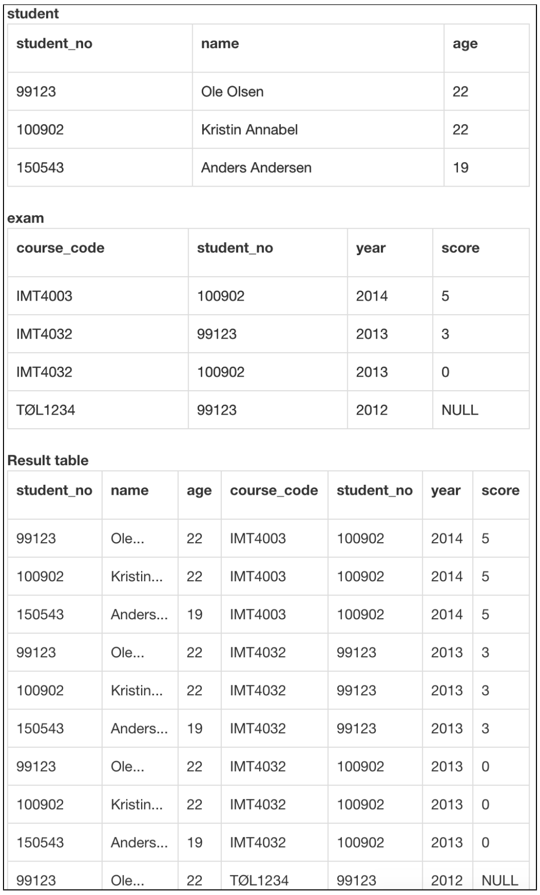
course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
TØL1234	99123	2012	NULL

Result table

student_no	name	age	course_code	student_no	year	score
100902	Kristin...	22	IMT4003	100902	2014	5
99123	Ole...	22	IMT4032	99123	2013	3
100902	Kristin...	22	IMT4032	100902	2013	0
99123	Ole...	22	TØL1234	99123	2012	NULL

Figure 34: Result from a RIGHT JOIN query

- **Other type of JOIN:** When the query contains another type of JOIN than RIGHT JOIN, the number of times the cell is to be animated is calculated by dividing the number of rows in the result table by the number of rows in the original table. We now get the number of times each cell is to be animated. If the number is not an integer number, it is rounded upwards or downwards. Figure 35 shows an example of a CROSS JOIN, with the query "*SELECT * FROM student, exam*". In the example, the result table contains 12 rows (not all is shown in the figure), and the "student" table contains 3 rows. If we e.g. want to know how many times the student_no "99123" from the table "student" is animated, we divide 12 rows in the result table by the 3 rows in the "student" table; 12/3 = 4. Each cell in the "student" table have been animated four times, and this includes the cell "99123".



student

student_no	name	age
99123	Ole Olsen	22
100902	Kristin Annabel	22
150543	Anders Andersen	19

exam

course_code	student_no	year	score
IMT4003	100902	2014	5
IMT4032	99123	2013	3
IMT4032	100902	2013	0
TØL1234	99123	2012	NULL

Result table

student_no	name	age	course_code	student_no	year	score
99123	Ole...	22	IMT4003	100902	2014	5
100902	Kristin...	22	IMT4003	100902	2014	5
150543	Anders...	19	IMT4003	100902	2014	5
99123	Ole...	22	IMT4032	99123	2013	3
100902	Kristin...	22	IMT4032	99123	2013	3
150543	Anders...	19	IMT4032	99123	2013	3
99123	Ole...	22	IMT4032	100902	2013	0
100902	Kristin...	22	IMT4032	100902	2013	0
150543	Anders...	19	IMT4032	100902	2013	0
99123	Ole...	22	TØL1234	99123	2012	NULL

Figure 35: Result from a CROSS JOIN query

Count the number of cell animations: The hidden HTML value attribute is given to the cell with the value of zero. This value is counted upwards every time the cell is animated. When this value is equal to the number of repetitions calculated, the class "*used*" is given to the cell and the cell is no longer in the list of available cells to be animated.

5.3 Discussion

5.3.1 Decomposition

Both Cembalo et al. [7] and Kearns et al. [4] uses decomposition in their learning tools. Because SQL cannot be divided into procedural steps, decomposition helps us divide the query into temporal sets, creating temporary result tables for each set. We wanted to help the students create a mental model to understand the underlying logic of SQL, and as Cembalo et al. stated, decomposition might help students "*overcome the mental visualization problem*" [7]. This is why we chose to implement decomposition in our visual learning tool.

5.3.2 Data sets

Both Kearns et al. [4] and Cembalo et al. [7] used small data sets in their learning applications. This was because they had issues with visualization when the data sets were to big. We encountered the same problem, where big data sets created big tables with a lot of data. We thought that students would not get the full potential of the animations when they would have to scroll the web view to see all tables and data, and decided to use small data sets in our prototype.

5.3.3 Visualization

We chose to visualize original tables and a result table. This was also done in the learning tools developed by Cembalo et al. [7], Kearns et al. [4] and Murray& Guimaraes [9]. In SAVI [7], two original tables are visualized in the first step of the decomposition during a JOIN. We chose to implement this in our prototype as well, because this seems like an understandable way of visualizing a JOIN statement. The SAVI application also visualizes the projection as the last step of the decomposition. We hypothesis that this makes it easier for the students to understand the underlying logic by getting the correct rows in the result table before selecting specific columns, and implemented this in our prototype. Each decomposed step uses the result table from the previous step as an original table. We chose to implement this in our prototype, because we hypothesise that this will give the students a better understanding of where the data in the result table is from.

During animation, Cembalo et al. [7] highlights the row that is being animated from the original table to the result table. We chose to implement this feature in our prototype, because this makes it easier for the students to see what is being animated.

We chose to animate each row/cell in the original table by scaling up the text, and animating the text horizontally before it was animated vertically down to the result table. This animation solution was not encountered in any of the learning tools we have discussed, and was an idea that we came up with during the brainstorming session. We argue that this type of animation makes it clearer to the students where the animated cells are animated from and where they are to be located in the result table. Cembalo et al. [7] animates each row vertically from the original table to the result table. We implemented this solution as well, and it was tested in the preliminary user test. The results from the preliminary user test showed that the students liked both visualization ideas, but that the horizontal animation was preferred. This is why we chose to implement this visualization idea in the final version of our prototype.

5.3.4 Selection and projection

The SAVI application [7] visualizes all SELECT statements by animating column wise. We chose not to use their solution to selection/projection, but to animate the cells from the original table to the result table row by row. If the SELECT statement contains a projection, only the selected columns in each row are animated to the result table. If the SELECT statements contains a selection, all columns in each row are animated. This was done because we wanted to follow the same visualization as the rest of the decomposed steps uses and we hypothesis that this could be more understandable for the students than animating column wise.

6 Student experiment

6.1 Methodology

After the development of the prototype was completed, an experiment was executed with first year programming bachelor students at NTNU Gjøvik. The main purpose of the experiment was to compare the learning outcome in a group of students using an online tutorial, Vertabelo Academy [25], with another group of students using the viSQLizer prototype, per time unit. Vertabelo Academy is different from the viSQLizer prototype; it does not use decomposition or animations, and it provides the users with explanatory text and exercises as well as a structured set of tutorial topics with increasing complexity. Sub goals of the experiment was to see if the use of a learning tool motivates the students to learn, and if and how they want to apply the use of such a tool to traditional methods for learning SQL.

We wanted to conduct the experiment with development students who had not been enrolled in a database course but who are to follow such a course the following semester. These students might give us the best overview of the learning outcome after given time unit since they have no previous experience with SQL. Students from the first year of the bachelor programmes Game Development and Software Engineering are selected, because these are classic developer degrees that does not only teach database management, but also SQL, which makes this a natural choice of students for this experiment.

6.1.1 Sampling

By our request, the programme coordinators for the bachelor programmes Software Engineering and Game Development, Tom Røise and Simon McCallum, sent out an email to all their first year bachelor students asking them to voluntarily participate in the SQL experiment. The programme coordinators also encouraged the students on several occasions to participate in the experiment.

In total, the Game Development bachelor program has twenty-seven students enrolled, and amongst these, one girl. The Software Development bachelor program has in total thirty-two students enrolled, and amongst these, two girls. The email informed the students that participation could give them a great head start at learning SQL and databases which would be of great value to them in the upcoming semester when they were to be enrolled in a database course. The students also were promised pizza if they attended the experiment, and one of the participating students would be randomly drawn to win a gift certificate of 500 NOK.

6.1.2 Experiment design

The experiment followed the 2-group post-test experimental design. No pre-test was conducted because the students in our sample had none or limited experience with SQL. The students participating in the experiment were randomly divided into two groups and placed in two different rooms. One group was the control group, using Vertabelo Academy [25]. The other group was the experimental group, using the viSQLizer prototype[1]. If the number of participating students were an odd number, the experimental group would get one more student than the control group. This was decided because we primarily were determining the learning outcome for students using the viSQLizer prototype. All information were given to each group separately; it was important that none of the groups felt as if they were the control group and were never informed about what the other group was doing. The information given to the groups contained a short introduction to the subject of the master thesis, a short description of what the students were to do during the experiment, and a short demonstration of the learning system. A sign was placed outside each door, informing about experiment in progress to avoid interference.

6.1.3 Experiment setup

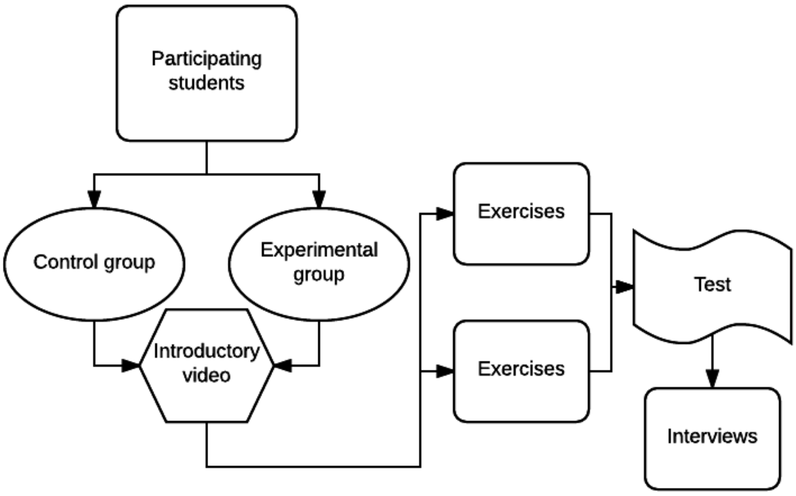


Figure 36: Experiment structure

Introductory video

Rune Hjelsvold, supervisor for this master thesis, and the responsible professor in the database course for the second year bachelor students, was asked by us to record a short introductory video on databases and SQL. This was done to give the students a quick introduction to databases, tables and SQL, because they lacked basic knowledge of databases and SQL. Hjelsvold recorded

[1] `http://andmark.no/kristin/`

the video [2], which lasted about 12 minutes and available online, viewable in all major browsers. All students in both groups watched the introductory video by themselves before proceeding with the experiment.

Exercises

The exercises were available to the students on paper. This made it easy for the students to use the learning tool and at the same time read the instructions for the exercises without having to switch between tabs in the browser window.

The exercises are limited to SELECT statements. Both groups had the same exercises, but with different data sets. We based the exercises on the exercises available in the Vertabelo Academy [25] learning tool. Vertabelo Academy offers several different subjects within SQL. We chose exercises from various topics that were supported by the viSQLizer prototype. A total of nine exercises were chosen, with associated sub-exercises.

The control group uses the learning tool Vertabelo Academy [25]. The chosen exercises were listed on a paper with information about the course name, topic and exercise. Six questions were written at the end of the exercise folder. These questions referred to how the students had perceived the tutorial, and how they liked this type of method for learning. The exercises for the control group can be found in Appendix I.

The experimental group uses the viSQLizer prototype. Each exercise was printed on paper. Seven questions were written at the end of the exercise folder. These questions referred to how the students had perceived the viSQLizer-prototype and how they liked this type of method for learning. The exercises for the experimental is can be found in Appendix J.

We allowed students working together and helping each other with the exercises and the learning tool during the exercise part of the experiment. Working together might be of great value for some students and could give them a greater learning outcome. In a natural setting, students might work together with exercises and/or in learning systems, and this is why we allowed working together in the experiment as well. It was however important that they kept their conversations as quiet as possible, to not disturb other students. No help with the exercises was given to the students.

Test questions

A test was created to test both groups in their understanding of SQL and the logic of SQL. The test was delivered to the participating students on paper. The questions in the test were based on the topics from the exercises. The test was divided into two parts. In the first part, the students had to fill in the correct data in an empty table based on a query and explaining text. The complexity of the query and the table increased with the questions. In the second part, the students had to

[2] Video available at: `https://screencast.uninett.no/relay/ansatt/runehjhig.no/2016/30.03/741333/Kort_intro_til_relasjonsdatabaser_og_SQL_-_20160330_110713_39.html`

write the correct query based on a result table containing data and a text explaining the table content. Also here the complexity of the query and the table increased with the questions. Both the control group and the experiment group answered the same test with the same sample data. The test can be found in Appendix K, and the correct answers to the test questions can be found in Appendix L.

When the students were to answer the test, they were seated separately without their personal computer. This was done to make sure that the students did not help each other or talk during the test, and to make sure that they did not look up the answers in the learning tool or any other place online using their computers.

Interviews

Questions for the interviews were prepared in advance. These questions were created to get qualitative results, giving us an in-depth view of how the learning tools were experienced and if/how they helped the students understand the logic of SQL. The questions were about what the students thought when they answered the various parts of the test, their opinion about the learning system and the visualization, and how they would like to use the learning system; in or outside of class. The interview questions can be found in Appendix M. Five students from each group were picked out randomly to answer the interview questions.

6.2 Results

Seventeen students participated in the experiment, and all seventeen completed all parts. We do not have any knowledge of how many there were from each of the two classes, but both students from Game Development and Software Development were represented in the selection. One girl was present in the experiment,out of the three girls in total in the two classes. Nine students got assigned to the experimental group and eight students got assigned to the control group. During the experiment, the students were asked not to rush through the exercises, but to use lots of time to fully learn the examples and the logic. The experiment lasted for about three hours in total.

Three students did not bring their own laptop, and were placed in a nearby computer lab to conduct the experiment. These three students were not observed at all times during the experiment, but was checked in on on a regular basis so that we could observe them interacting with the learning tool. These three students got assigned to the control group using Vertabelo Academy [25].

When the students got the test questions, they were asked to answer as best they could. Also here the students were asked not to rush, but to take their time. Students that finished the test were asked to participate in a four - six minutes interview.

The students participating in the experiment were asked if they had some prior knowledge of databases and/or SQL from earlier. This was done because it could have an effect on the test results. The results show that two people in each group have some prior knowledge of SQL and/or databases, and that the remaining students participating have no prior knowledge.

6.2.1 Results from the questions relating the learning system

The students were asked some questions about their opinion of the learning system, and about the usage of such systems for learning SQL, both instead of and together with traditional lectures. The results indicate that the control group, using Vertabelo Academy [25], liked the use of exercises in the tutorial and the text explaining the various operations in SQL. "*It is essential. It was easy to read. And it was nice to get a confirmation that you had understood everything*". "*It is a good method for learning. Doing practical exercises really helps the memory*". The students mentioned that they were motivated, interested, and engaged in the learning process, and that this type of learning was quicker than traditional methods for learning SQL. "*I learn quicker, than by reading an entire chapter in a book without understanding how the syntax works*". Students mentioned that the exercises was a good contribution to learning, and that it followed a meaningful path where each exercise is building on the knowledge from the previous exercise. "*It is nice, because it is piecewise learning, and you can decide your own pace*". The control group indicated that they would like to use the tutorial as an addition to traditional lectures.

Some students in the control group stated that they did not notice the representation of the result table in the Vertabelo Academy tool, and thus did not fully understand how the tables was created. Some also had trouble with finding information about the tables and the column-names.

Vertabelo Academy	
Code	**Students (8 in total)**
Like to do exercises	7
Nice as an addition to lectures	7
Easy to understand how the result is created	6

Table 8: Vertabelo Academy learning system results

The experimental group, using the viSQLizer prototype, seemed positive about learning SQL practically and expressed that they got some understanding of the logic behind the SQL queries. "*This type of learning seems really nice. It is informative, intuitive and interactive. Is is easier than reading a book or watching a video*". "*It is much easier to learn and understand when the process is visualized*". "*It is much more fun to see it in practice, and to see the logic behind the queries*". The experimental group wanted to use the learning tool as an addition to traditional lectures.

The students' opinions varied when it came to animation speed in viSQLizer. Some of the students stated that they experienced the animation of the text as too slow, while some experienced it was too fast, and some experienced it as having the correct pace. Some students stated that they wanted an explanatory text in the learning tool, explaining the syntax and various operations.

Two students said that they struggled with the decomposing in viSQLizer, and that they did not understand what happened to the data when it was displayed in more than one step.

viSQLizer	
Code	**Students (9 in total)**
Easy to understand how the result is created	9
Nice as an addition to lectures	9
Difficult with decomposing	2

Table 9: viSQLizer learning system results

6.2.2 Results from the test

Most students got all questions in the first part of the test correct. Minor mistakes did occur, but are not important for our findings. These questions were present to make the first part of the test easier for the students, to boost their self-esteem, and to get the opportunity to ask the students what they pictured in their mind when they answered these questions. This could be important when trying to determine if the visualization help the students with their memory and understanding. The first part of the text also contained textual explanations of the given query, which might make the query redundant and make it possible for the students to answer correct without any knowledge about the SQL syntax. This is why our analysis only focuses on the second part of the test, which indicate how well the students have understood the syntax and logic of SQL and what parts of SQL they are still struggling with.

The tests were graded with a A-F grading style. The gradings were done in cooperation with Rune Hjelsvold, a professor in a database course at NTNU Gjøvik. The results from the test conducted in the experiment shows that the students in both groups have done a really good job. The control group, using Vertabelo Academy [25], got an average of 89% correct answers on part two of the test, and the experimental group, using the viSQLizer prototype, got an average of 90% correct answers. This is shown in Figure 38. A confidence interval of 95% is added to the graph. This shows that the average student is getting 86-92% correct answers, which is a B to A grade. The grade span is from A to C, and there are no students who got a lower grade than C in any of the groups. The grade distribution for both groups is shown in Figure 37.

In the control group, one student answered 100% correctly on all six questions. In the experimental group, three students answered all the six questions 100% correctly. An overview of the number of students answering incorrect on the various questions for the two groups can be seen in Table 10. This table does show all answers that did not get a 100% score on the test.

As we can see in Table 10, there are an observable difference between the students' incorrect answers in the two groups on question 3, 5 and 6:

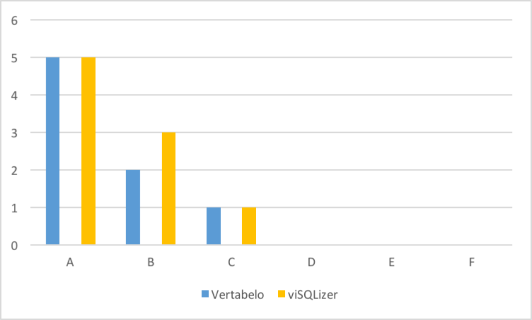


Figure 37: Grade distribution in both groups

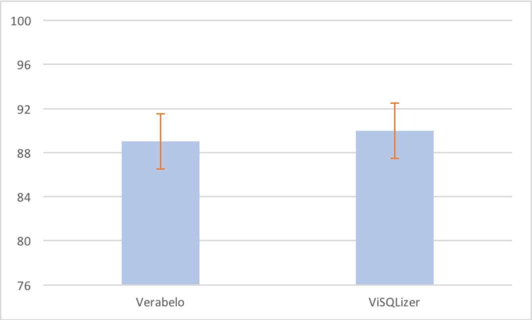


Figure 38: Average percentage value of the results from the two groups

Question	Vertabelo Academy (out of 8 students)	viSQLizer (out of 9 students)
Q1	2 (25%)	2 (22%)
Q2	2 (25%)	2 (22%)
Q3	0 (0%)	2 (22%)
Q4	1 (12.5%)	1 (11%)
Q5	5 (62.5%)	3 (33%)
Q6	7 (87.5%)	4 (44%)

Table 10: Incorrect answers per question

Question 3

All students in the Vertabelo Academy group answered question 3 correctly, while two students in the viSQLizer group answered this questions incorrectly. These were not serious mistakes; one put the bracket the wrong way, while the other student put an equality sign in the query. Both students still got a 90% score on question 3.

Question 5 and 6

Students in both group answered similar incorrect answers for question 5 and 6. The incorrect answers can be organized into eight error types, as described below. The distribution of error types answered by the students in both groups for each question can be seen in Table 11, and will be discussed furter in Section 6.3.

T1 Switching the WHERE and ON clause

T2 Missing a WHERE/ON clause

T3 Missing table name prefix in ON clause

T4 Switching the table names in ON clause

T5 Using WHERE and AND in the query with a JOIN clause

T6 Missing the WHERE clause after ON clause

T7 Missing AND in the WHERE clause

T8 No answer provided

Type of error	Question 5		Question 6	
	Vertabelo Academy	viSQLizer	Vertabelo Academy	viSQLizer
T1	2	1		1
T2	1			
T3	1	1	1	
T4	1	1		
T5			3	1
T6			2	1
T7			1	
T8				1

Table 11: Error types for question 5 and 6

6.2.3 Results from the interviews

Visualization in mind

The interviewed students were asked some questions about what they were picturing in their mind when they were answering the two different parts in the test. This was done to determine if the students using the viSQLizer learning tool easily could visualize the solution than the control group when answering the test, because of the visualization of the tables in the prototype.

Vertabelo Academy:

Amongst the interviewed students in the control group, two students stated that they were thinking in a logical way when answering part one of the test, where they were to fill data into different tables. Two students stated that they were picturing how the data was placed in the resulting table. One student did not look at the tables when working with the Vertabelo Academy learning tool, because he did not notice its presence, and thus felt that he had some problems answering part one of the test.

viSQLizer:
In the experimental group, three students said they were thinking in a logical way when answering the test. Two students stated that they were picturing how the data was flowing and being placed into the tables in part one of the test.

Experiment group	Logical	Picturing	Did not notice table representation
viSQLizer	3	2	0
Vertabelo Academy	2	2	1

Table 12: What students were visualizing in mind during experiment

Learning styles

The results from the interviews show that the students prefer various types of learning styles. One student stated that he did not feel the need to use a learning tool. "*I learn better by reading than by watching visualizations*". Some students stated that SQL was very logical to them. "*How the queries are build seems very logical to me*". A student from the control group, where exercises were a big part of the tutorial, stated "*I would like the learning tool to be customized for each student. If a student has problems, he could get simpler exercises, and if some other student has no issues, he could get more difficult exercises*". Another student wanted additional interactive features in the learning tool. "*Some short video clips in the learning tool would have been nice as well*".

Additional help/information

The students were asked if they were missing lectures, and/or a teacher to ask questions while working with the learning tool.

Vertabelo Academy:
Students in the control group stated that they did not miss a teacher to ask questions during the experiment, but four of the interviewed students stated that they would want this kind of help if they were to use the learning tool during a database course.

One student from the control group answered "*It would have been nice to have the opportunity to ask a teacher for help if you are wondering. But it works very well as is, because you can work in your own pace*". Another student stated that he would like to use the learning tool in the database course. "*I hope we get to use a learning tool like this one ourselves, when we are taking the database* *course next semester*".

viSQLizer:
Students in the experimental group stated that they did not miss a teacher to ask questions during the experiment, but three of the interviewed students stated that it would be nice to have more information about the theory.

One student from the experimental group answered "*Both yes and no. Some of the underlying theory are missing; what actually happens and why. This learning tool could be a supplement to lectures. In lectures, we get more theory behind the practical approach*". Another student stated that he would like some textual explanations in the learning tool. "*I would like the visualizations to be explained, not just visualized. The current scenario could be explained shortly in text*".

Positive effects of using a learning tool

The students were asked what they liked about using a learning tool, like the one they had tried during the experiment. This question was related to learning tools in general, and the answers from the students were not grouped when analysing the results. Most students stated that they learned quicker by using the learning tool and the practical approach. "*I am learning quicker, and remember it better*". "*I was very engaged, and I felt that I did learn better*". "*You get to practice. You are not just sitting in a lecture to get knowledge; when it goes through one ear and out the other. You get to use it and try it out for yourself*". "*You get a practical approach to the subject, you get to practice and repeat, and then the knowledge sticks better*". "*By doing a practical approach, you get it into your fingers, better than by hearing about the theory; when you know the theory but not how to apply it.*"

Some of the students in the experimental group mentioned some things that are specific for visualization in the viSQLizer prototype. A student from the experimental group stated "*By using such a tool, it is easy to understand how SQL works*". Another student from the same group seemed satisfied with how the visualization was organized. "*When you wrote a query, you got all the information you had asked for. And the visualization told you how much data was going to be fetched, before the animation began, by showing the empty result table with the correct number of rows and columns*".

Negative aspects of using the learning tool

Vertabelo Academy: When asked about negative aspects when using the learning tool, some students in the control group answered that it was hard to find information about the database, as column names. The students did not mention any other negative aspects.

ViSQLizer: Some students in the experimental group stated that there were too small data sets in the sample database. "*If there are only four rows in a table, you cannot really see what the query is doing*".

Decomposition of the queries

Some students from the experimental group mentioned that they did not quite understand the decomposition in the viSQLizer prototype. One student stated "*The steps were confusing in the beginning. I did not notice that I could press 'next'. This should maybe be clearer, or the tool could possibly show the entire solution with animation first, then show it step by step. That would be the optimal solution, at least for me*". Another student stated "*I was confused when there were multiple steps. You had the star in the beginning, in step one, and that created confusion in the beginning, when the final answer to the query was in step two*".

One of the students stated that he was satisfied with how the tool processed the queries, and stated "*It is nice to see, when you are comparing things and asking about stuff, what you have asked for and what is processed. If you want to get some columns, the tool tells you what you asked for. And if you asked for the wrong things, then I see 'Oops, I have to change my query'*".

Experiment setup satisfaction

All ten of the students attending the interviews answered that they felt they got enough time with the exercises and the learning tool before the test, that the test was designed in an understandable way, and that the level of the test questions were appropriate based on the type of exercises they had done in advance.

6.3 Discussion

The average test scores differed just by 1% between the two groups. Vertabelo Academy [25] has exercises and textual information for the various SQL operations, but does not have any particular visualizations. viSQLizer does not have exercises or an explanatory text, but visualizes the original table(s), the result table and the underlying logic of SQL by decomposing the queries. The two learning tools use different features to teach SQL. Our hypothesis is that the visualization represented in the viSQLizer prototype has great potential, and could give students a better learning outcome than ordinary online tutorials if some of the features present in the Vertabelo Academy tutorial were added to the viSQLizer prototype. We discuss future possibilities for viSQLizer further in Chapter 10.

No major difference were found when comparing the two groups' error types in Table 11. The information provided to the students during the exercise session were equal for the two groups which might explain the similar error types in the two groups; the information provided could lack some crucial information needed to fully understand JOIN, or this type of explanation could seem confusing for the students. The number of students answering incorrect on question 5 and 6 is almost doubled for the Vertabelo Academy group when compared to the viSQLizer group. We hypothesis that one reason might be that these students focused more on answering the exercises in the Vertabelo Academy tool than understanding the logic. Another reason might be that JOIN is a threshold concept for the students and difficult for novice students to understand just after approximately one hour of effort with the learning tool.

Results from the experiment indicate that the participating students, using viSQLizer, understood some of the underlying logic of SQL. Students in the control group, using Vertabelo Academy [25], were mostly interested in answering correct on the exercises provided in the tutorial. None of the students gave us any indications that they were pondering over how the result table was constructed. Students in the experimental group mentioned that it was easy to understand how SQL work, and commented on elements like small data sets being an issue. We think this indicates that the students understood some of the underlying logic, because they commented on the data flow of table data. The experimental group understood the how and why, and commented on issues that might only be clear for students who understand some of the underlying logic.

The results indicate that the students would like to have lectures in addition to using the learning tool so that they have the opportunity to ask a professor for help if they need to. The students stated that they wanted to learn the SQL basics in lectures and use the learning tool for a hands-on approach to learn how SQL works in practice.

In our experiment, the participating students were divided into a control group and a experimental group because of the high number of participants; seventeen students. We established a back up plan to be used if less than sixteen people were to participate in the experiment. This limit was set, because we decided that seven people in one group would be too few to get a good result. The plan was to keep all participant in one group, and ask them to use the viSQLizer prototype. The same exercises and test questions would be handed out to all the participating students as seen in Appendix J and K. All students would have been interviewed at the end of the experiment, to get as much qualitative information from the experiment as possible because of the limited number of participants.

7 Expert feedback

7.1 Methodology

The three database professors that attended the expert interviews in the beginning of the master thesis project were asked to give some feedback on the final version of the developed prototype.

The professors were given information about the results from the conducted experiment. They were asked to test the prototype with their own queries, or the same queries as the students in the experiment. They were asked to comment on what they liked or disliked with the prototype, what features they would like to see implemented in the future, as well as if/how they would like to use this tool in their own database course. The communication with the professors regarding the expert feedback was done by mail.

7.2 Results

The feedback from the professors was positive. "*I think this learning tool is good and instructive. It has a good explanation of basic SQL*". "*This is exiting, and it is a tool that I would like to use for teaching*". "*Testing the learning tool was fun*". One of the professors stated that the prototype would be great for learning and cramming SQL syntax. "*Furthermore, I think the animation is good as an illustration, and from a pedagogical perspective. This is a tool with great teaching potential as I see it*". Another professor stated that the learning tool would be an appropriate tool for novice students learning SQL. "*It is a useful tool in the beginning of learning SQL*".

Regarding future development, the professors stated that better coverage of multiple types of queries and functions, more information about the tables with type definitions, a possibility for the students to create their own database/tables and data, and a possibility for the students to sketch a data model that automatically generates tables would be of interest to them. One professor stated that he/she would like to have exercises implemented in the learning tool with points for correct answers. Another professor stated that he would like to pause, choose speed and rewind/fast forward the animations. "*Students are really different*". One of the professors thought that the decomposition of queries was difficult to understand and suggested that the various operations could be done explicit. It was also stated by one of the professors that the animation worked great for selection and join, but not as great for projection, when only a few of the columns are animated.

7.3 Discussion

Results show that the professors would like to use viSQLizer in their database course, and that it is suitable for novice students learning basic SQL. One professor stated that the animations could be beneficial from an pedagogical perspective, which could indicate that the professor sees the tool as beneficial both for visualization in lectures as well as a learning tool used by students to learn the SQL syntax.

The professors had various ideas for improving the prototype. These ideas might indicate that the professors want the learning tool to be used by the students as an addition to traditional learning methods at all levels of knowledge and not only for novice students. The professors also mentioned that the tool should include more clauses in the future, which could indicate that they are enthusiastic about the possibilities for this type of learning tool. One of the professors stated that the decomposition was difficult to understand, which supports the results from the experiment. Solutions to this issue are discussed in Chapter 10.

8 Analysis and discussion

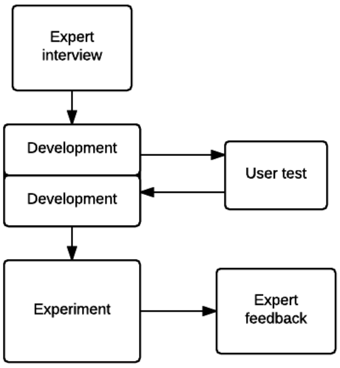


Figure 4: The project structure (repeated from page 16)

The expert interviews conducted gave us insight into how different database courses are being organized, and what threshold concepts students might have. Database professors often start with single table operations; selection and projection, and build on this knowledge when teaching JOIN on multiple tables. JOIN was said to be one of the students' threshold concepts, and research has shown that students often have problems with syntax rules [4]. One of the professors choose to teach relational algebra in the beginning of the database course. This could be an alternative to teaching SQL with decomposition. We want to develop a tool that does not require previous knowledge of relational algebra, which is why we chose to use decomposition in our learning tool.

SELECT statements were focused on during development of the viSQLizer prototype, with an emphasis on selection, projection and JOIN. This was done because preliminary research indicated that these were students' threshold concepts and core topics in SQL. Decomposition and animations were implemented to help students create a mental model, thus understanding the underlying logic of SQL. The development of the viSQLizer prototype was done in two intervals with a preliminary user test between the two. The preliminary user test gave us insight into what visualization ideas and animation modes the students preferred. We chose to keep the "drag-out" and "big text" animations modes, based on students feedback and our observations both in advance and during the user test.

We conducted an experiment which gave us insight into the learning outcome for students using viSQLizer contra students using the Vertabelo Academy tutorial [25]. Results show that students in both group got the same average score, which indicates that the students got the same learning outcome per time unit. The experiment also gave us some insight into the students' understanding of the underlying logic. Results from the interviews indicate that the students using viSQLizer understood some of the underlying logic, because they amongst other things commented on the data flow in the tables. This is something that needs to be further researched in the future. Results from the experiment showed that some of the students had difficulties understanding the decomposing of SQL. Some database professors teach relational algebra which could help the students understand decomposing, while other database professors does not focus on this in their database course. We propose that students need an introduction to decomposition before using the learning tool if they do not have previous knowledge of relational algebra.

The expert feedback gave us some insight into how database professors perceived the learning tool and how they would like to use it in a database course setting. The database professors were positive towards the learning tool, and stated that they would like it to be used in lectures for visualization as well as by students as an addition to lectures for them get a practical approach.

8.1 Limitations and weaknesses

Methodology and execution in all parts of this thesis have some limitations and weaknesses. These will be addressed in this section.

8.1.1 Expert interviews

Three professors were interviewed at project start. We think that we could have gotten stronger results if more database professors were interviewed. With the results from additional database professors, we could have determined if the course organization and students' threshold concepts are something that most professors agree on or if the professors' answers vary.

In our results, we included the database professor that participated in the pilot interview. This is a weakness, because some of the answers were not detailed enough for us to analyze. The reason for this is that the first version of the interview questions asked at the pilot interview were not specific enough on that SQL was the topic, and thus the interviewed professor did not answer all questions related to SQL in particular. Thus in the future we would have liked to conduct the pilot interview on an additional professor, not part of the results and analysis.

8.1.2 Development

Four visualization ideas were tested in the preliminary user test: "*normal*", "*synchronous*", "*drag-out*" and "*big text*". Two of these were further chosen to visualize table data in viSQLizer: "*drag-out*" and "*big text*". By testing additional visualization ideas, we could have gotten a different result on the preliminary user test. This could also strengthen our experiment results because of a more thorough examination of possible visualizations.

Some improvements could have been done to the architecture and implementation in the Decomposer project [12], but were not executed because we focused on implementing core functionality instead of improving the Decomposer. One improvement that could have been done to the architecture, is the format of the data sent from the server to the client. Today, this data is pure HTML, written with PHP:echo, and sent directly to the client. A better approach would be to send this data in JSON format, because it allows for client side processing which is quicker because there is less data to transfer for consecutive requests.

Another improvement to the architecture could be to change the mySQLi extension to the PDO extension in PHP. This is because PDO supports twelve different database drivers [1], while mySQLi only support mySQL [2]. A different database or multiple database servers might be used in a later version of viSQLizer, and there would be no need to change the interface for accessing the databases if PDO is used.

When locating a target cell for animation, we use the column and table name. *Alias* could not be supported by the prototype by using this approach. A solution could be to get the original names from the mysqli_fetch_array method, where both alias name and original name is provided for both the table and the columns. This should be done server side before assigning these names as classes to the table and columns. We did not focus on supporting alias in the prototype because we wanted to focus on students' threshold concept and on developing a learning tool for novice students.

The client side implementation is messy and unstructured. We did not focus on writing well-structured code, because we wanted to focus on implementing core functionality. Because of this, it might be difficult for a developer to understand the implemented code in the prototype to further develop viSQLizer. In retrospect, we see that by sacrificing some time on structuring the code, the viSQLizer could be further developed by another developer in an effective way. The unstructured implementation does however not have any impact on the results found in this project.

8.1.3 User test

In retrospect, we think that a pilot user test could benefit the user test results. A pilot could have given us an indication if some of the questions were not understandable, or if the test setup should be improved on before the actual user test. To our knowledge, all the participating students understood all the questions, but some students stated that there were too many questions in the questionnaire. This could have been improved if a pilot was conducted. During the user test, the participating students were talking to each other and with the course coordinator. This made them distracted from the task at hand which could have an impact on the results. A solution could have been to conduct the user test in a private room without distractions.

[1] http://php.net/manual/en/pdo.drivers.php

[2] http://php.net/manual/en/intro.mysqli.php

Six students attended the user test, but only five answered the survey. This might have been because some of the students did not bring their own computers. We should have have reminded the students to bring their personal computer to the user test, after the first email was sent. If more students had participated in the user test, we could have gotten more results to analyse, and a better basis for choosing visualization ideas. In retrospect, we see that we should have allocated more time on recruiting students to participate in the user test.

8.1.4 Experiment

In part one of the test provided to the students participating in the experiment, students were to fill in the correct data in the result table for a given query. These tasks did not only contain a query, but also some text explaining the query. For example, for the query "*SELECT * FROM student*", the explanatory text "*Show all data for the table student*" was given to the students. This text might make the query redundant, and make it easy for the students to fill data into the result table without any knowledge about SQL queries. This was one of the reasons why we chose only to analyse the results from the second part of the test.

By conducting a pilot experiment in advance, some issues could have been prevented. During a pilot, we could have tested the experiment setup to make sure that everything was working as expected; students' understanding of the introductory video, the information in the exercises, and the test questions. We did however not encounter any serious issues during the experiment, but know that the risk is greater when a pilot experiment has not been conducted in advance.

The information provided to the students during the exercise session were equal for the two groups, except for the data sets. This information text were provided by Vertabelo Academy [25], but we did not do any research to determine if this text were structured and presented in such a way that the students understood the provided information. If this had been done, we could have analysed the incorrect test answers further knowing that the information text had no impact on the test results.

The participation in the experiment was voluntary. The students participating could have been students most eager to learn SQL in their respective classes, and could therefore be more engaged or motivated to do a good job than the average student in their class. Some students could also be participating in the experiment only because of the free pizza, and are not necessarily students that are eager to learn SQL or interested in learning it. These factors could lead to some selection bias.

The total number of students in the two classes are fifty-nine. Because only seventeen students participated in the experiment, the experiment results could have some validity issues. This is because the impact that the participating student group has on the results could be skewed, relative to the entire student group. If more students, or other students from the two classes, would have participated in the experiment, the results could have been different.

8.1.5 Expert feedback

The expert feedback was obtained by mail. We hypothesise that we could have gotten more detailed feedback from the database professors if we had been interviewing them in person for the expert feedback, as we did for the preliminary expert interviews. The information sent to the professors were just an email. By structuring this email more thoroughly, the results could have been more useful and interesting results. We could also have observed the professors using viSQLizer to see how they interacted with it and if there were some issues with using the prototype.

9 Conclusion

Our results indicate that a learning tool with visualizations may help students create a mental model, and thus understanding the underlying logic of SQL. This is based on the fact that these students comment on the data flow in SQL, and not just on the visualization itself. The viSQLizer prototype uses "drag-out" and "big text" animations on table data as well as decomposition, for helping students create a mental model for SQL. This answers the first research question, "*How can visualizations be integrated in a learning tool to help students create a mental model, and thus understand the underlying logic of SQL?*", but further research needs to be conducted to determine if other types of visualization could help students create a mental model more efficiently, and how much of an impact visualizations in a learning tool could have on the students' understanding of the underlying logic.

Results from the conducted experiment indicates that students think of the viSQLizer prototype as motivating and engaging. Students have stated that they learn more and are more motivated to learn through this learning tool than through traditional lectures and textbooks. This partly answers part one of the second research question; "*What are the effects on motivation and learning outcome when using a learning tool with visualizations compared to using an ordinary online tutorial?*", but more research need to be conducted to determine the effects on students' motivation when using a visual learning tool.

When comparing the students' learning outcome when using a visual learning tool contra using an online tutorial, we found that the learning outcome were similar for both groups per time unit. This indicates that visualizations in an SQL learning tool could contribute to a better learning outcome if certain elements from an online tutorial are implemented in the visual learning tool, and thus answer the second part of the second research question; "*What are the effects on motivation and learning outcome when using a learning tool with visualizations compared to using an ordinary online tutorial?*". We suggest more research to be done to determine the long term effects on students' learning outcome when using a visual learning tool.

Interviews with both students and database professors have shown that a visual learning tool such as the viSQLizer prototype could both be used by professors in lectures, to visualize database tables and the data flow, and by students as a practical addition to traditional teaching methods such as lectures. This gives us an answer to the third research question; "*How can interactive learning tools be applied in relation to traditional methods for teaching SQL?*".

Feedback from database professors have been positive towards the prototype, and the database professors have shown real interest in using the tool in their own database course. Students have shown great engagement and seems to be motivated by the viSQLizer learning tool. Students stated that they were interested in using the learning tool in a database course.

10 Future work

10.1 What remains unsolved

Learning outcome

During this thesis, we have tested the learning outcome when using a learning tool with decomposition and animations against an online tutorial with explanatory text and exercises. The tests were conducted in a three hour session on students who have not been attending a database course at university level. We think students' learning outcome needs to be researched further, and propose a long term study of students' learning outcome when using a visual learning tool. The study could be done throughout a whole semester during a database course. By dividing the students into two groups, we get the opportunity to compare the exam results of the students who use a visual learning tool to the students who do not.

Understanding of underlying logic

Students' understanding of the underlying logic and their mental model when using a visual learning tool should be researched further. We suggest conducting some experiments to determine if the visualizations present in viSQLizer could help students create a good mental model that could help them understand the underlying logic of SQL.

Provided textual information

Textual information regarding SQL syntax provided to students in a learning tool should be researched further, to get a better understanding of how such information should be structured and presented to give the students the best possible understanding of the SQL syntax.

Alternative visualization ideas

We suggest that alternative visualization ideas should be implemented and tested to seek better visualizations regarding learning outcome and students' understanding of the underlying logic of SQL. During this thesis work, four different animation modes have been implemented and tested. There are several other possibilities to visualize SQL, and we support further research into alternate visualization ideas for representing SQL. Research could also be done to determine what colors to use when highlighting animated cells, selected columns and the cell origin, since we did not do any preliminary research on which colors were suitable for these purposes.

Understanding of decomposition

We think that students' understanding of decomposition should be researched further. The experiment results show that some students in the experimental group struggled with understanding the decomposition of the queries. We suggest that this issue could be prevented by offering an introduction to how decomposition works. As one of the professors told us in the preliminary expert interviews, he teaches relational algebra to the students before teaching SQL, to "operationalize the direction of implementation". We want our learning tool to be understandable and accessible for all students, both those who have been tough relational algebra and those who have not, and thus suggest some further research to be done on how relational algebra could give students a better understanding of decomposition and the decomposed steps of a query and on how to teach students how decomposition works without relational algebra.

10.2 Future development

viSQLizer is just a prototype developed to answer the research questions stated in this paper. Improvements to make viSQLizer a valuable learning tool capable of teaching students the underlying logic of SQL are suggested below. Some of these improvements are based on feedback from students and database professors, and some of the improvements are features that are natural in a SQL learning tool but not implemented in the current version of viSQLizer.

10.2.1 Exercises

The control group, using Vertabelo Academy [25] in the experiment, seem motivated because of the exercises. These findings indicate that implementing exercises in a learning tool could help motivate students, thus resulting in them using more time with the learning tool, and giving them an appropriate path to follow throughout their learning. Some of the database professors also state that they would like to have exercises in the learning tool. Students might find exercises motivating which might help them proceed through all the required subjects in an SQL course.

We propose an example case for implementing exercises in the learning tool; customers. Graphical representation of different customers could be added to the application. These graphical representations could be an image of a person, a customer, asking for assistance on some SQL related issue. The design could be somewhat similar to the design of the GalaXQL game [14], but with different narratives that give the students tasks to be solved. The customer could introduce the student to a SQL related issue and inform of any new syntax related information that the student might need to know to solve the task. The customer could say "*I want to know if it is the birthday of some of my staff today*", and the student would have to submit the correct query to the customer to complete the task. When all tasks are completed by the student for one customer, a new customer could show up and new tasks and subjects could be presented to the student. Points could also be implemented to give the students extra motivation to complete the tasks.

10.2.2 Explanatory text

Some of the students in the experimental group, using viSQLizer in the experiment, state that they would like textual information in the learning tool about what was going on in the different decomposed steps and the SQL syntax. Students in the control group state that they liked the explanatory text given to them because it was easy to understand. Some of the database professors also state that they would like explanatory text in the prototype. Kearns et al. [4] uses decomposing in their learning tool, and explains each step of the decomposed query to the user. We hypothesise that the viSQLizer prototype could result in a better learning outcome for the students if exercises and explanatory text are present in the learning tool. Thus, in further development of the prototype we suggest implementing explanatory text, explaining each decomposed step and its operators.

10.2.3 Player controls for animation

As suggested by one of the database professors, player controls for the animation could be implemented in the prototype. This is an approach that could benefit the students, because our results show that students experience the animation speed differently. To allow students to pause, fast forward or rewind the animations, students get more control over the visualization. A time line could also be displayed, allowing the students to go back and forth through the animation. This feature could also be beneficial for the database professors, having the ability to pause and go back and forth through the animation during visualization in lectures.

10.2.4 Decomposition

A number of students had difficulties with understanding the decomposition in the viSQLizer prototype during the experiment. One of the students participating in the experiment suggested that showing the query execution with animations and without decompositions as a first step could be a way to help students get a better understanding of SQL. The decomposition could be shown after the first step described above. This is something that might be looked more into, to see if this could be a possible way of teaching students SQL using decomposition.

10.2.5 Interaction/Manipulation of data

To make students more engaged in the learning tool, we suggest developing a more interactive learning tool by giving the students the possibility to directly manipulate the provided data. This could be done through exercises where the students would have to drag-and-drop the correct data into the correct cell in the result table and thus predict what will happen in the next step of the decomposed query. Students ability to predict the placement of the data for the next step could indicate that the students understand some of the underlying logic of SQL.

10.2.6 Different learning modes

We suggest that the viSQLizer learning tool could consist of different learning modes. One of these modes could contain exercises, as explained earlier. Another mode could have a more practical approach, a "try it yourself" kind of learning, where students get to write their own queries, and watch the visualization. This mode could be as the viSQLizer prototype is presented today. This mode could also contain textual information, explaining the different operations and the different decomposed steps.

A mode could also contain tasks submitted by the course coordinator of a database course. This solution was implemented in the learning tool developed by Kearns et al. [4]. By solving these tasks, students could either get grades as in an exam or just solve these tasks to get a better understanding of SQL. The students know that the tasks are relevant to the course which could make them more motivated to use the tool for learning SQL.

10.2.7 Customization

The students could get their own log in account in the learning tool, where they could save their own queries and create their own tables and data sets. As one of the professors suggested, students could also have the possibility to sketch a data model that automatically generates tables in the learning tool.

10.2.8 Facilitating learning on a higher level

The viSQLizer prototype was developed as a learning tool for novice students learning SQL. As a future development, the learning tool could facilitate a higher level of learning by implementing more advanced SQL functions and operations.

Aggregation functions

Aggregation functions are difficult to animate, because the result cell for the aggregation function is dependent on the original table data and is not present in the original table itself. Thus, the cells for the aggregation functions cannot be animated from the original table to the result table as the animation is implemented in the current version of viSQLizer.

A solution to this issue could be to add another type of animation to viSQLizer. If the aggregate function is AVG, SUM or COUNT, the aggregated column could get its value when all the cells have been animated into the result table. The average/sum/count could be calculated, and the result could be shown in the aggregate cells. By making the calculations visible to the students, students could understand how the aggregated value is created.

For aggregate functions such as MAX and MIN, the aggregated column in the result cell could get its value when all cells have been animated from the original table to the result table. An arrow could go through each row in the result table, and find the cell which is the MAX/MIN cell for the specified column. This design could be similar to the design implemented by Cembalo et al. [7], where an arrow is going through each row to determine if the row/cell fulfills the specified conditions.

Sub-queries

Sub-queries are not implemented in the viSQLizer prototype, because visualization of sub-queries need multiple tables to be represented. We suggest that sub-queries could be implemented in the prototype by visualizing each part of the sub-query separately and comparing the two queries to get the result table. The design of the prototype has to be modified so that there are room for multiple tables and multiple result tables at the same time.

LIKE expressions

We suggest a design for implementing LIKE expressions in viSQLizer. The LIKE expression could go trough (and hover) the different cells in the original table to find a match. Once a match is found, the cell is animated from the original table to the result table and the search continues for more matches. By visualizing LIKE expressions this way, students might understand why certain cells fulfill the LIKE expressions' criteria and thus are animated into the result table.

Table operations

Table operations could also be implemented in the future. To prevent the database data to be altered or deleted by the students, a copy could be created for each individual student. This approach was done by Grillenberger & Brina [21]. The visualization of the different table operations might depend on the type of operation. For CREATE TABLE, two different solutions could be implemented. The new table could be displayed, animating information from the query to the table itself, or the database model could be displayed, and the new table could be added in the model.

For ALTER TABLE, the original table could show the table before the change, and the result table could show the table after the change. Table information could be animated from the query to the result table. For DROP TABLE, the table could be removed from the database model, or the table could be visualized to the student, before it is removed using some animation technique.

When visualizing an INSERT/DELETE/UPDATE query, both the original and the result table could be visualized to the student. The original table displays the table before any change is done, and an animation inserts some new rows into the result table, deletes some rows or updated the data in some rows.

10.2.9 Visualizing bigger data sets

Some of the students participating in the experiment said that they would want bigger data sets in the database. viSQLizer only contains small data sets because this would produce wider and longer original tables and result tables. Big tables makes it necessary to scroll down on the web page to view the entire animation path for a cell from the original table to the result table. This issue could be solved by implementing some sort of automatic scroll, where the web view always follow the animated cell. Another solution could be to use the whole width of the view, and display multiple tables horizontally instead of vertically.

Bibliography

[1] Atchison, W. F., Conte, S. D., Hamblen, J. W., Hull, T. E., Keenan, T. A., Kehl, W. B., McCluskey, E. J., Navarro, S. O., Rheinboldt, W. C., Schweppe, E. J., Viavant, W., & Young, Jr., D. M. March 1968. Curriculum 68: Recommendations for academic programs in computer science: A report of the acm curriculum committee on computer science. *Commun. ACM*, 11(3), 151–197.

[2] Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM, New York, NY, USA, 999133.

[3] Mitrovic, A. 1998. A knowledge-based teaching system for sql. In *Proceedings of ED-MEDIA*, volume 98, 1027–1032.

[4] Kearns, R., Shead, S., & Fekete, A. 1996. A teaching system for sql. In *Proceedings of the 2Nd Australasian Conference on Computer Science Education*, ACSE '97, 224–231, New York, NY, USA. ACM.

[5] Hansen, K. T. & Mallaug, T. 2008. *Databaser*. Gyldendal Norsk Forlag - Gyldendal akademisk, 2 edition.

[6] Sadiq, S., Orlowska, M., Sadiq, W., & Lin, J. 2004. Sqlator: An online sql learning workbench. *SIGCSE Bull.*, 36(3), 223–227.

[7] Cembalo, M., De Santis, A., & Ferraro Petrillo, U. 2011. Savi: A new system for advanced sql visualization. In *Proceedings of the 2011 Conference on Information Technology Education*, SIGITE '11, 165–170, New York, NY, USA. ACM.

[8] Prior, J. C. 2003. Online assessment of sql query formulation skills. In *Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20*, ACE '03, 247–256, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

[9] Murray, M. & Guimaraes, M. 2008. Animated database courseware: Using animations to extend conceptual understanding of database concepts. *J. Comput. Sci. Coll.*, 24(2), 144–150.

[10] Allenstein, B., Yost, A., Wagner, P., & Morrison, J. 2008. A query simulation system to illustrate database query execution. *SIGCSE Bull.*, 40(1), 493–497.

[11] Patwardhan, M. & Murthy, S. 2015. When does higher degree of interaction lead to higher

learning in visualizations? exploring the role of 'interactivity enriching features'. *Computers & Education*, 82, 292 – 305.

[12] Brovold, M., Johnsen, F., & Folland, K. A. T. 2014. Decomposer. `https://dev.imt.hig.no/confluence/display/SQLD/SQL+Decomposer+Home`. (Visited Apr. 2016).

[13] Swanhart, J. & Rothe, A. Php-sql-parser. `https://github.com/greenlion/PHP-SQL-Parser`. (Visited Feb. 2016).

[14] GalaXQL. Galaxql 3.0. `http://sol.gfxile.net/g3/`. (Visited Feb. 2016).

[15] Brusilovsky, P., Sosnovsky, S., Lee, D. H., Yudelson, M., Zadorozhny, V., & Zhou, X. June 2008. An open integrated exploratorium for database courses. *SIGCSE Bull.*, 40(3), 22–26.

[16] KhanAcademy. Intro to sql: Querying and managing data. `https://www.khanacademy.org/computing/computer-programming/sql`. (Visited Feb. 2016).

[17] Schemaverse. The schemaverse. `https://schemaverse.com`. (Visited Feb. 2016).

[18] Danaparamita, J. & Gatterbauer, W. 2011. Queryviz: Helping users understand sql queries and their patterns. In *Proceedings of the 14th International Conference on Extending Database Technology*, EDBT/ICDT '11, 558–561, New York, NY, USA. ACM.

[19] Belov, A. Serious game for learning relational databases. Master's thesis, Gjøvik University College, Norway, 2004.

[20] W3Schools. Tryit editor v1.5. `http://www.w3schools.com/sql/trysql.asp?filename=trysql_select_all`. (Visited Feb. 2016).

[21] Grillenberger, A. & Brinda, T. 2012. eledsql: A new web-based learning environment for teaching databases and sql at secondary school level. In *Proceedings of the 7th Workshop in Primary and Secondary Computing Education*, WiPSCE '12, 101–104, New York, NY, USA. ACM.

[22] SQLFiddle. Sql fiddle. `http://sqlfiddle.com`. (Visited Feb. 2016).

[23] SQLCourse. Interactive online sql training. `http://www.sqlcourse.com`. (Visited Feb. 2016).

[24] CodeAcademy. Learn sql. `https://www.codecademy.com/learn/learn-sql`. (Visited Feb. 2016).

[25] VertabeloAcademy. Sql queries. `https://academy.vertabelo.com/course/sql-queries`. (Visited Feb. 2016).

[26] Cumming, A. Sqlzoo. `http://sqlzoo.net`. (Visited May 2016).

[27] Cigas, J. & Kushan, B. May 2010. Experiences with online sql environments. *J. Comput. Sci. Coll.*, 25(5), 251–257.

[28] Teradata Corporation. Teradata university network. `http://tunweb.teradata.ws/trial/`. (Visited May 2016).

[29] VertabeloAcademy. Sql queries. `https://academy.vertabelo.com/course/sql-queries/multiple-tables/join/join`. [Screenshot] (Visited Apr. 2016).

[30] Cembalo, M., De Santis, A., & Ferraro Petrillo, U. 2011. Savi. `http://savisql.appspot.com`. [Screenshot] (Visited Apr. 2016).

[31] Danaparamita, J. & Gatterbauer, W. 2011. Sql query visualization. `http://queryviz.com/online/`. [Screenshot] (Visited May 2016).

[32] Brinkmann, S. & Tanggaard, L. 2012. *Kvalitative metoder. Empiri og teoriutvikling*. Gyldendal Norsk Forlag - Gyldendal akademisk, 1 edition.

[33] NSD. Nsd - norwegian centre for research data. `http://www.nsd.uib.no/nsd/english/index.html`. (Visited Apr. 2016).

[34] Meyer, J. & Land, R. 2003. *Threshold concepts and troublesome knowledge: Linkages to ways of thinking and practising within the disciplines*. University of Edinburgh Edinburgh.

[35] Folland, K. A. T. 2015. Increasing the student's motivation when visualizing sql curricula.

[36] gskinner. Easeljs v0.8.2. `http://www.createjs.com/docs/easeljs/classes/Ticker.html`. (Visited May 2016).

A Final version of interview questions

Explain the project: *A visual learning system for SQL. The target audience is bachelor students who is learning SQL for the first time. The students should be motivated and understand the underlying logic of SQL. The final goal is a learning system that could be used across educational institutions.*

1. What is your position and responsibility at your current workplace?
2. How many years of experience do you have with SQL and databases? How many years of experience do you have with teaching SQL and databases?
3. The progression in a database course could be different from course to course, but a course often start out with the basics, as modification, and continues with queries, aggregating and, at the end, working with multiple tables. Do you agree? Why/why not?

   How would/have you organize the progression in a SQL course?

   Is there any other topics/subtopics you would/have in the curricula in a SQL course?
4. Literature searches has shown that the most difficult topics for the students to understand in SQL are joins, grouping, aggregating and union. Do you agree? Why/why not?

   Do you know of any other topics that the students find more difficult than other topics in the SQL course, and why?
5. What threshold concepts do you think the students need to learn/overcome in SQL before they can continue with more difficult curricula?
6. Literature searches has shown that learning systems should contain the three learning styles auditorial, visual, and kinesthetic. This could be done eg. by showing information to the user in clear text, showing the data in tables and have drag-and-drop to manipulate the data in the tables. What do you think adding the learning styles and using them in a such a way?

   What do you do in your own teaching?

   What would you wish to do if you had the necessary resources, and why?
7. A hypothesis is that it is better to solve a tasks in a learning system and explore, than to write a pre-set query and see the results. What do you think of this?

   What do you do in you teaching? (Example queries versus giving them tasks to solve themselves)

   Would you want the possibility to make queries that the students could try out, have the students create the queries themselves and explore on their own, or a mixture of both? Why would you prefer this?
8. A solution to motivate the students could be to give them points for solving the different tasks

in a correct way, and for using time in the learning system. The points could eg. be used in a high-score list for each individual class. What do you think about using pointification like this to motivate the students?

If NO: Do you still think that pointification or competition could help motivate the students in a learning system?

9. Do you think that another form of gamification could motivate the student to use should a learning system? Why/why not?
10. What do you do in your teaching to create/hold on to the students' motivation?
11. Do you test your students in databases and SQL on motivation? Why/why not?
    If YES: How do you do this?
    If NO: How would you test the students' motivation if you where to do it? Why?
12. Do you test you students understanding of the logic today? Why/why not?
    If YES: How do you do this?
    If NO: How would you, if you where to, test the students' understanding of the logic in SQL? Why?
13. Any last comments?

B Request for interview participation in research project " viSQLizer: An interactive visualizer for learning SQL"

Background and Purpose: The project is a master's project at NTNU Gjøvik – the Department of Computer Science and Media Technology. The purpose of the project is to make a prototype of a visual e-learning system for databases that improves the students' motivation and understanding. The main research topics that is to be analyzed is based on students' threshold concepts when they are learning SQL, if a visual learning system could increase the students' motivation, and if an interactive system could improve the students understanding of SQL. You have been requested to participate because of your knowledge about databases and because of your position as a professor in one/more database-courses. You have been contacted because of your affiliation with project supervisor, Rune Hjelsvold.
What does participation in the project imply? The project requires active participation in the form of a interview, that will take about half an hour. No other collection of data about the participant will be collected from other sources. The questions will concern database-curricula and your professional opinions about difficult subjects for the students, threshold concepts that the students should overcome, thoughts about visualization of databases and other details about visualization of database-curricula. The data from the interview will be collected by notes and audio recordings.
What will happen to the information about you? All personal data will be treated confidentially. Only the students and supervisor will have access to the personal data. The list of names and interview results will be kept separately, so not connections can be made between the participant and the results. The participant will not be recognizable in the publication, but the university, faculty and area of responsibility of the participant might be mentioned in the publication. The project is scheduled for completion by 26. May. Any personal data and recordings will be deleted within that date, and the results will be anonymous.
Voluntary participation: It is voluntary to participate in the project, and you can at any time choose to withdraw your consent without stating any reason. If you decide to withdraw, all your personal data will be made anonymous. If you have questions about the project, please contact
Project lead Kristin Annabel Torjussen Folland
Tlf: 47657330, Email: kristin.folland@hig.no
or supervisor Rune Hjelsvold, Email: rune.hjelsvold@hig.no

Consent for participation in the study
I agree that the interview will be audio recorded □
I have received information about the project and am willing to participate □
———————————————— (Signed by participant, date)

C Notification form NSD user test

Norsk samfunnsvitenskapelig datatjeneste AS
NORWEGIAN SOCIAL SCIENCE DATA SERVICES
NSD

MELDESKJEMA

Meldeskjema (versjon 1.4) for forsknings- og studentprosjekt som medfører meldeplikt eller konsesjonsplikt (jf. personopplysningsloven og helseregisterloven med forskrifter).

1. Intro

Samles det inn direkte personidentifiserende opplysninger?	Ja ○ Nei ●	En person vil være direkte identifiserbar via navn, personnummer, eller andre personentydige kjennetegn. Les mer om hva personopplysninger. NB! Selv om opplysningene skal anonymiseres i oppgave/rapport, må det krysses av dersom det skal innhentes/registreres personidentifiserende opplysninger i forbindelse med prosjektet.
Hvis ja, hvilke?	□ Navn □ 11-sifret fødselsnummer □ Adresse □ E-post □ Telefonnummer □ Annet	
Annet, spesifiser hvilke		
Samles det inn bakgrunnsopplysninger som kan identifisere enkeltpersoner (indirekte personidentifiserende opplysninger)?	Ja ● Nei ○	En person vil være indirekte identifiserbar dersom det er mulig å identifisere vedkommende gjennom bakgrunnsopplysninger som for eksempel bostedskommune eller arbeidsplass/skole kombinert med opplysninger som alder, kjønn, yrke, diagnose, etc. NB! For at stemme skal regnes som personidentifiserende, må denne bli registrert i kombinasjon med andre opplysninger, slik at personer kan gjenkjennes.
Hvis ja, hvilke	Kjønn, alder, studieretning og skole	
Skal det registreres personopplysninger (direkte/indirekte/via IP-/epost adresse, etc) ved hjelp av nettbaserte spørreskjema?	Ja ● Nei ○	Les mer om nettbaserte spørreskjema.
Blir det registrert personopplysninger på digitale bilde- eller videoopptak?	Ja ○ Nei ●	Bilde/videoopptak av ansikter vil regnes som personidentifiserende.
Søkes det vurdering fra REK om hvorvidt prosjektet er omfattet av helseforskningsloven?	Ja ○ Nei ●	NB! Dersom REK (Regional Komité for medisinsk og helsefaglig forskningsetikk) har vurdert prosjektet som helseforskning, er det ikke nødvendig å sende inn meldeskjema til personvernombudet (NB! Gjelder ikke prosjekter som skal benytte data fra pseudonyme helseregistre). Dersom tilbakemelding fra REK ikke foreligger, anbefaler vi at du avventer videre utfylling til svar fra REK foreligger.

2. Prosjekttittel

Prosjekttittel	ViSQLizer - en SQL visualizer	Oppgi prosjektets tittel. NB! Dette kan ikke være «Masteroppgave» eller liknende, navnet må beskrive prosjektets innhold.

3. Behandlingsansvarlig institusjon

Institusjon	NTNU	Velg den institusjonen du er tilknyttet. Alle nivå må oppgis. Ved studentprosjekt er det studentens tilknytning som er avgjørende. Dersom institusjonen ikke finnes på listen, har den ikke avtale med NSD som personvernombud. Vennligst ta kontakt med institusjonen.
Avdeling/Fakultet	NTNU i Gjøvik	
Institutt	Avdeling for informatikk og medieteknikk	

4. Daglig ansvarlig (forsker, veileder, stipendiat)

Fornavn	Rune	Før opp navnet på den som har det daglige ansvaret for prosjektet. Veileder er vanligvis daglig ansvarlig ved studentprosjekt. Veileder og student må være tilknyttet samme institusjon. Dersom studenten har ekstern veileder, kanbiveileder eller fagansvarlig ved studiestedet stå som daglig ansvarlig. Arbeidssted må være tilknyttet behandlingsansvarlig institusjon, f.eks. underavdeling, institutt etc. NB! Det er viktig at du oppgir en e-postadresse som brukes aktivt. Vennligst gi oss beskjed dersom den endres.
Etternavn	Hjelsvold	
Stilling	Professor	
Telefon	61135184	
Mobil		
E-post	rune.hjelsvold@ntnu.no	
Alternativ e-post	rune.hjelsvold@ntnu.no	
Arbeidssted	NTNU Gjøvik	

Adresse (arb.) Postnr./sted (arb.sted) Sted (arb.sted)	Teknologivegen. 22 2815 Gjøvik Gjøvik	

5. Student (master, bachelor)

Studentprosjekt	Ja ● Nei ○	Dersom det er flere studenter som samarbeider om et prosjekt, skal det velges en kontaktperson som føres opp her. Øvrige studenter kan føres opp under pkt 10.
Fornavn Etternavn Telefon Mobil E-post Alternativ e-post Privatadresse Postnr./sted (privatadr.) Sted (arb.sted) Type oppgave	Kristin Annabel Folland 47657330 kristin.folland@hig.no kristinannabel.folland@gmail.com Hovdevegen 4A 2821 Gjøvik Gjøvik ● Masteroppgave ○ Bacheloroppgave ○ Semesteroppgave ○ Annet	

6. Formålet med prosjektet

Formål	Lage en prototype av et system for å lære studenter SQL ved bruk av animasjoner og andre visualiseringer. Prototypen skal testes for å se om dette hjelper til forståelse av logikken i SQL hos studenter som ikke kan SQL fra tidligere.	Redegjør kort for prosjektets formål, problemstilling, forskningsspørsmål e.l.

7. Hvilke personer skal det innhentes personopplysninger om (utvalg)?

Kryss av for utvalg	□ Barnehagebarn □ Skoleelever □ Pasienter □ Brukere/klienter/kunder □ Ansatte □ Barnevernsbarn □ Lærere □ Helsepersonell □ Asylsøkere ■ Andre	
Beskriv utvalg/deltakere	Studenter fra første års bachelorprogram	Med utvalg menes dem som deltar i undersøkelsen eller dem det innhentes opplysninger om.
Rekruttering/trekking	Trekkes gjennom universitetet. Studieprogram-ansvarlig (Simon McCallum) for en klasse tar kontakt med klassen og spør om de kan delta i en brukertest.	Beskriv hvordan utvalget trekkes eller rekrutteres og oppgi hvem som foretar den. Et utvalg kan trekkes fra registre som f.eks. Folkeregisteret, SSB-registre, pasientregistre, eller det kan rekrutteres gjennom f.eks. en bedrift, skole, idrettsmiljø eller eget nettverk.
Førstegangskontakt	Simon McCallum tar kontakt med studentene som ønsker å bli med i utvalget første gang, deretter også Kristin Annabel Folland.	Beskriv hvordan kontakt med utvalget blir opprettet og av hvem. Les mer om dette på temasidene.
Alder på utvalget	□ Barn (0-15 år) □ Ungdom (16-17 år) ■ Voksne (over 18 år)	Les om forskning som involverer barn på våre nettsider.
Omtrentlig antall personer som inngår i utvalget	5	
Samles det inn sensitive personopplysninger?	Ja ○ Nei ●	Les mer om sensitive opplysninger.

Hvis ja, hvilke?	□ Rasemessig eller etnisk bakgrunn, eller politisk, filosofisk eller religiøs oppfatning □ At en person har vært mistenkt, siktet, tiltalt eller dømt for en straffbar handling □ Helseforhold □ Seksuelle forhold □ Medlemskap i fagforeninger	
Inkluderes det myndige personer med redusert eller manglende samtykkekompetanse?	Ja ○ Nei ●	Les mer om pasienter, brukere og personer med redusert eller manglende samtykkekompetanse.
Samles det inn personopplysninger om personer som selv ikke deltar (tredjepersoner)?	Ja ○ Nei ●	Med opplysninger om tredjeperson menes opplysninger som kan spores tilbake til personer som ikke inngår i utvalget. Eksempler på tredjeperson er kollega, elev, klient, familiemedlem.

8. Metode for innsamling av personopplysninger

Kryss av for hvilke datainnsamlingsmetoder og datakilder som vil benyttes	■ Papirbasert spørreskjema ■ Elektronisk spørreskjema ■ Personlig intervju □ Gruppeintervju ■ Observasjon □ Deltakende observasjon □ Blogg/sosiale medier/internett □ Psykologiske/pedagogiske tester □ Medisinske undersøkelser/tester □ Journaldata	Personopplysninger kan innhentes direkte fra den registrerte f.eks. gjennom spørreskjema,intervju, tester, og/eller ulike journaler (f.eks. elevmapper, NAV, PPT, sykehus) og/eller registre (f.eks.Statistisk sentralbyrå, sentrale helseregistre). NB! Dersom personopplysninger innhentes fra forskjellige personer (utvalg) og med forskjellige metoder, må dette spesifiseres i kommentar-boksen. Husk også å legge ved relevante vedlegg til alle utvalgs-gruppene og metodene som skal benyttes. Les mer om registerstudier her. Dersom du skal anvende registerdata, må variabelliste lastes opp under pkt. 15
	□ Registerdata	
	□ Annen innsamlingsmetode	
Tilleggsopplysninger		

9. Informasjon og samtykke

Oppgi hvordan utvalget/deltakerne informeres	■ Skriftlig ■ Muntlig □ Informeres ikke	Dersom utvalget ikke skal informeres om behandlingen av personopplysninger må det begrunnes. Les mer her. Vennligst send inn mal for skriftlig eller muntlig informasjon til deltakerne sammen med meldeskjema. Last ned en veiledende mal her. NB! Vedlegg lastes opp til sist i meldeskjemaet, se punkt 15 Vedlegg.
Samtykker utvalget til deltakelse?	● Ja ○ Nei ○ Flere utvalg, ikke samtykke fra alle	For at et samtykke til deltakelse i forskning skal være gyldig, må det være frivillig, uttrykkelig og informert. Samtykke kan gis skriftlig, muntlig eller gjennom en aktiv handling. For eksempel vil et besvart spørreskjema være å regne som et aktivt samtykke. Dersom det ikke skal innhentes samtykke, må det begrunnes.

10. Informasjonssikkerhet

Hvordan registreres og oppbevares personopplysningene?	■ På server i virksomhetens nettverk □ Fysisk isolert PC tilhørende virksomheten (dvs. ingen tilknytning til andre datamaskiner eller nettverk, interne eller eksterne) □ Datamaskin i nettverkssystem tilknyttet Internett tilhørende virksomheten □ Privat datamaskin □ Videoopptak/fotografi ■ Lydopptak □ Notater/papir ■ Mobile lagringsenheter (bærbar datamaskin, minnepenn, minnekort, cd, ekstern harddisk, mobiltelefon) □ Annen registreringsmetode	Merk av for hvilke hjelpemidler som benyttes for registrering og analyse av opplysninger. Sett flere kryss dersom opplysningene registreres på flere måter. Med «virksomhet» menes her behandlingsansvarlig institusjon. NB! Som hovedregel bør data som inneholder personopplysninger lagres på behandlingsansvarlig sin forskningsserver. Lagring på andre medier - som privat pc, mobiltelefon, minnepinne, server på annet arbeidssted - er mindre sikkert, og må derfor begrunnes. Slik lagring må avklares med behandlingsansvarlig institusjon, og personopplysningene bør krypteres.
Annen registreringsmetode beskriv		

Hvordan er datamaterialet beskyttet mot at uvedkommende får innsyn?	Datamaskin beskyttes med brukernavn og passord, lydopptak overføres til datamaskin og slettes fra tilhørende lydopptager, og tilgang til server beskyttes med brukernavn og passord. Eventuelle utskrifter låses i et låst skap, i et låst rom.	Er f.eks. datamaskintilgangen beskyttet med brukernavn og passord, står datamaskinen i et låsbart rom, og hvordan sikres bærbare enheter, utskrifter og opptak?
Samles opplysningene inn/behandles av en databehandler?	Ja ○ Nei ●	Dersom det benyttes eksterne til helt eller delvis å behandle personopplysninger, f.eks. Questback, transkriberingsassistent eller tolk, er dette å betrakte som en databehandler. Slike oppdrag må kontraktsreguleres.
Hvis ja, hvilken		
Overføres personopplysninger ved hjelp av e-post/Internett?	Ja ○ Nei ●	F.eks. ved overføring av data til samarbeidspartner, databehandler mm. Dersom personopplysninger skal sendes via internett, bør de krypteres tilstrekkelig. Vi anbefaler for ikke lagring av personopplysninger på nettskytjenester. Dersom nettskytjeneste benyttes, skal det inngås skriftlig databehandleravtale med leverandøren av tjenesten.
Hvis ja, beskriv?		
Skal andre personer enn daglig ansvarlig/student ha tilgang til datamaterialet med personopplysninger?	Ja ○ Nei ●	
Hvis ja, hvem (oppgi navn og arbeidssted)?		
Utleveres/deles personopplysninger med andre institusjoner eller land?	● Nei ○ Andre institusjoner ○ Institusjoner i andre land	F.eks. ved nasjonale samarbeidsprosjekter der personopplysninger utveksles eller ved internasjonale samarbeidsprosjekter der personopplysninger utveksles.

11. Vurdering/godkjenning fra andre instanser

Søkes det om dispensasjon fra taushetsplikten for å få tilgang til data?	Ja ○ Nei ●	For å få tilgang til taushetsbelagte opplysninger fra f.eks. NAV, PPT, sykehus, må det søkes om dispensasjon fra taushetsplikten. Dispensasjon søkes vanligvis fra aktuelt departement.
Hvis ja, hvilke		
Søkes det godkjenning fra andre instanser?	Ja ○ Nei ●	F.eks. søke registereier om tilgang til data, en ledelse om tilgang til forskning i virksomhet, skole.
Hvis ja, hvilken		

12. Periode for behandling av personopplysninger

Prosjektstart Planlagt dato for prosjektslutt	29.02.2016 01.06.2016	Prosjektstart Vennligst oppgi tidspunktet for når kontakt med utvalget skal gjøres/datainnsamlingen starter. Prosjektslutt: Vennligst oppgi tidspunktet for når datamaterialet enten skalanonymiseres/slettes, eller arkiveres i påvente av oppfølgingsstudier eller annet.
Skal personopplysninger publiseres (direkte eller indirekte)?	□ Ja, direkte (navn e.l.) ■ Ja, indirekte (bakgrunnsopplysninger) □ Nei, publiseres anonymt	NB! Dersom personopplysninger skal publiseres, må det vanligvis innhentes eksplisitt samtykke til dette fra den enkelte, og deltakere bør gis anledning til å lese gjennom og godkjenne sitater.
Hva skal skje med datamaterialet ved prosjektslutt?	■ Datamaterialet anonymiseres □ Datamaterialet oppbevares med personidentifikasjon	NB! Her menes datamaterialet, ikke publikasjon. Selv om data publiseres med personidentifikasjon skal som regel øvrig data anonymiseres.Med anonymisering menes at datamaterialet bearbeides slik at det ikke lenger er mulig å føre opplysningene tilbake til enkeltpersoner. Les mer om anonymisering.

13. Finansiering

Hvordan finansieres prosjektet?	Trenger ingen finansiering da det er et masterprosjekt.	

14. Tilleggsopplysninger

Tilleggsopplysninger		

D Request for participation in user test for research project "viSQLizer"

Forespørsel om deltakelse i forskningsprosjektet

"viSQLizer"

Bakgrunn og formål
Dette studiet er et master-studie ved NTNU Gjøvik, der formålet er å teste om et visuelt system med animasjoner og andre visuelle elementer kan hjelpe studenter til å forstå logikken bak SQL på en bedre måte enn kun ved bruk av tradisjonelle undervisningsmetoder.

Du er blitt forespurt om å delta i brukertesten da du er en student som tidligere har gjennomgått et kurs i databaser og SQL. Da du er student i spill-programmering med kompetanse på visuelle elementer, er det ønskelig at du ser på visualiseringene i denne prototypen.

Hva innebærer deltakelse i studien?
Deltakelse i studiet innebærer at du tester ut prototypen, og svarer på en rekke spørsmål angående de forskjellige visualiserings-elementene. Spørsmålene skal svares på via et spørreskjema, enten i papir-versjon eller på nett. Brukertesten vil ta omtrent en time, og du vil bli observert når du tester ut prototypen. Noen i utvalget vil også bli forespurt om å delta i et kort intervju, som vil bli tatt opp med en enkel lydopptager, der spørsmålene vil omhandle prototypen og spørreskjemaet.

Hva skjer med informasjonen om deg?
Alle personopplysninger vil bli behandlet konfidensielt. Kun masterstudenten bak denne masteroppgaven (Kristin Annabel Folland) vil ha tilgang til personopplysninger, og disse vil ikke bli lagret på noen måte. Kun bakgrunnsinformasjon som alder, kjønn, studieretning og skole vil bli lagret, og disse vil ikke kunne bli koblet mot de øvrige dataene. Du vil ikke kunne gjenkjennes i publikasjonen.

Prosjektet skal etter planen avsluttes 01.06.2016. Personopplysninger og opptak vil bli slettet fortløpende etter innhenting, men senest innen denne datoen.

Frivillig deltakelse
Det er frivillig å delta i studien, og du kan når som helst trekke ditt samtykke uten å oppgi noen grunn. Dersom du trekker deg, vil alle opplysninger om deg bli anonymisert.

Dersom du ønsker å delta eller har spørsmål til studien, ta kontakt med Kristin Annabel Folland, 47657330, eller veileder, Rune Hjelsvold.

Studien er meldt til Personvernombudet for forskning, Norsk samfunnsvitenskapelig datatjeneste AS.

Samtykke til deltakelse i studien

Samtykke hentes muntlig.
Jeg har mottatt informasjon om studien, og er villig til å delta

----
(Signert av prosjektdeltaker, dato)

E Visualization survey

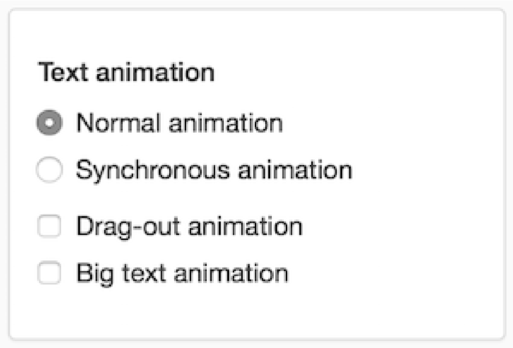
Informasjon

Informasjon om prototypen og spørreundersøkelsen

URL til prototype: andmark.no/kristin

Gå igjennom et og et spørsmål samtidig som du tester ut den visualiseringen spørsmålet omhandler. Svar så detaljert som du kan, om mulig.
Bruk enten nettleseren Google Chrome eller Safari, da prototypen ikke er blitt testet på andre nettlesere enda.

Animasjons-moduser



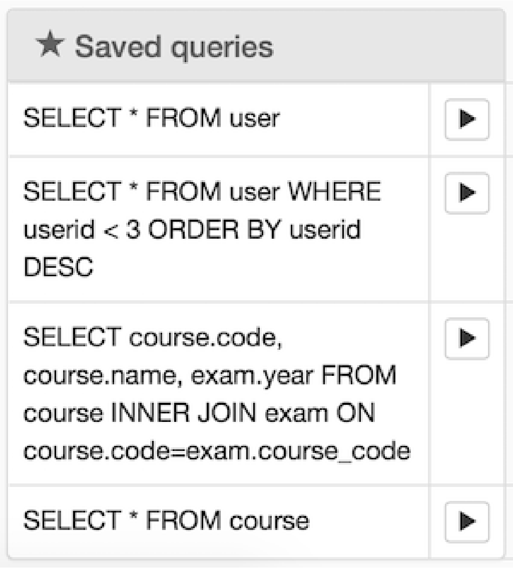
Text animation

● Normal animation
○ Synchronous animation

□ Drag-out animation
□ Big text animation

Velg animasjons-modus relevant til spørsmålet (Animasjons-modus er overskrift over hver del av spørsmåls-heftet)

Lagrede queries



★ Saved queries

SELECT * FROM user ▶

SELECT * FROM user WHERE userid < 3 ORDER BY userid DESC ▶

SELECT course.code, course.name, exam.year FROM course INNER JOIN exam ON course.code=exam.course_code ▶

SELECT * FROM course ▶

Velg en lagret query og klikk på "play"-knappen for å se visualiseringene. Ikke kjør egne queries, da dette er en tidlig versjon av prototypen, som ikke fungerer optimalt på alle queries enda.

Personalia

Informasjon om deg

1. Kjønn:

○ Mann

○ Kvinne

2. Alder:

□

3. Studieretning:

□

"Normal" animasjons-modus

Visualiseringer spesifikke for "normal" animasjonsvalg øverst til høyre på websiden

Velg øverste query under "Saved queries" først. Du kan deretter prøve med de andre, lagrede queriene om du ønsker.

4. Hva syns du om hastigheten på animasjon av teksten?

	For sakte animasjon	Passe hastighet på animasjon	For rask animasjon
Hastighet	○	○	○

Hvorfor?

5.

Hjelper markering av raden som animeres deg til å følge med på hvilken rad som er i fokus? Hvorfor/Hvorfor ikke?

6. Hjelper pilene som viser start- og slutt-posisjon for hver rad deg til å bedre forstå hvor radene hentes fra? Hvorfor/Hvorfor ikke?

7. Hvordan opplever du visualiseringen av pilene?

	Helt bortkastet	Burde ikke bli borte	Blir for raskt borte	Blir for sent borte	Oversiktlig
Visualisering av piler	□	□	□	□	□

Annet (vennligst spesifiser)

"Synchronous" animasjons-modus

Animasjoner spesifikke for "Synchronous" animasjonsvalget øverst til høyre på websiden

8. Hva syns du om hastigheten på animasjon av teksten?

For sakte animasjon	Passe hastighet på animasjon	For rask animasjon
○	○	○

Hvorfor?

9. Er det bedre at alle rader flyttes samtidig i motsetning til å flytte en og en rad, for å få forståelse for hvor data i resultattabellen hentes fra? Hvorfor/Hvorfor ikke?

10. Er det oversiktlig hvor dataene hentes fra og hvor de skal ved bruk av piler? Hvorfor/Hvorfor ikke?

11. Hvordan opplever du visualiseringen av pilene i dette animasjons-moduset?

	Helt bortkastet	Uoversiktlig når alle kommer samtidig	Burde ikke bli borte	Blir for raskt borte	Blir for sent borte	Oversiktlig
Visualisering av piler	□	□	□	□	□	□

Annet (vennligst spesifiser)

"Drag-out" animasjons-modus med "Normal"

Animasjoner spesifikke for "drag-out" animasjonsvalget øverst til høyre på websiden

Kjør normal-animasjons-modus sammen med drag-out animasjonen

12. Forstår du hvor radene hentes fra, selv om ingen piler blir visualisert?
Hvorfor/Hvorfor ikke?

13. Er det enklere å se hvor dataene hentes fra ved at dataene blir dratt ut til høyre på denne måten, i motsetning til når dataene animeres som på forrige side? Hvorfor/Hvorfor ikke?

14. Hvordan opplever du animasjonen av dataene?

	Uoversiktlig	For sakte animasjon	For rask animasjon	Oversiktlig
Animasjon normal	□	□	□	□

Annet (vennligst spesifiser)

Bytt til synkron-modus for animasjon sammen med drag-out animasjonen

15. Er det oversiktlig hvor dataene hentes fra og skal plasseres, når alle dataene animeres samtidig?

16. Hvordan opplever du animasjonen av dataene i synkron animasjons-modus?

	Uoversiktlig	For sakt animasjon	For rask animasjon	Oversiktlig
Animasjon synkron	□	□	□	□

Annet (vennligst spesifiser)

□

"Big text" animasjons-modus med "Normal"

Animasjoner spesifikke for "big text" animasjonsvalget øverst til høyre på websiden

17. Er det mer oversiktlig hvor dataene hentes fra og skal plasseres når teksten blir stor? Hvorfor/Hvorfor ikke?

18. Hjelper den hvite boksen bak teksten deg til å skille på hva som animeres og hva som er i bakgrunnen? Hvorfor/Hvorfor ikke?

19. Hvordan opplever du visualiseringen av teksten i dette animasjons-moduset?

	Uoversiktlig	For liten tekst	For stor tekst	For sakte animasjon av størrelses-ending	For rask animasjon av størrelses-endring	Oversiktlig
Visualisering	□	□	□	□	□	□

Annet (vennligst spesifiser)

Generelt

Generelle visualiseringer i prototypen

Se på den øverste lyseblå boksen over tabellene, når du kjører en valgfri query med flere steg og animasjon

20. I hvilken grad hjelper denne teksten deg til å forstå hvilken del av querien som visualiseres, og hva som er nytt siden forrige steg?

I ingen grad	I noen grad	I stor grad
○	○	○

Hvorfor?

21. Har du eventuelle forslag til hvordan dette kunne blitt gjort bedre?

Kjør en av de lagrede queriene. Hold musepekeren over en av cellene i resultat-tabellen, og se markeringene i orginal-tabellen(e)

22. Hjelper markering av rader og kolonner i orginal-tabellen(e) deg med å forstå hvor cellen du holder musepekeren over hentes fra? Hvorfor/Hvorfor ikke?

Hold musepeker over den første cellen i en rad i resultat-tabellen. Trykk på pilen som kommer opp i cellen

23. Hvor nyttig er det å markere hele raden på denne måten?

Ikke nyttig	Noe nyttig	Veldig nyttig
○	○	○

Hvorfor?

Kjør den lagrede querien som utfører en JOIN (tredje query i lista)

24. Virker det positivt å kunne gå frem og tilbake i stegene for å se animasjonene flere ganger? Hvorfor/Hvorfor ikke?

Gå til steg nr. 3 og se ferdig animasjonen

25. Gir det mening at kolonnene i hver original-tabell som er med i ON-delen av querien er markert i samme blåfarge som query-visningsboksen? Hvorfor/Hvorfor ikke?

Her menes de to kolonnene som nevnes etter ordet "ON" i querien

26. Hvor intuitiv er denne linken mellom kolonnene og querien?

Ikke intuitivt	Litt intuitivt	Veldig intuitivt
○	○	○

Hvorfor?

Marker første kolonne i en rad i resultat-tabellen på 3. steg i JOIN querien

27. Da dette er en av de to kolonnene i ON-delen av querien, markeres også den andre kolonnen i resultat-tabellen. Hjelper dette deg til å forstå hvordan de to original-tabellene er blitt joinet sammen? Hvorfor/Hvorfor ikke?

Å joine to tabeller sammen betyr å sette dataene i begge tabeller sammen til en resultat-tabell, her basert på en kolonne i hver tabell som viser hvilke rader som passer sammen

Gå til steg nr. 4 i JOIN querien

28. Hjelper markering av cellen som animeres deg til å forstå hvilke data som animeres, i motsetning til å markere hele raden? Hvorfor/Hvorfor ikke?

Kjør den lagrede querien som har en WHERE-statement (nr. 2 i lista). Gå til steg nr. 2 og se på animasjonen

29. Gir det mening at de radene og kolonnen(e) som oppfyller WHERE-statementet blir markert med samme blåfarge som query-visningsboksen? Hvorfor/Hvorfor ikke?

WHERE fungerer som en IF-test i programmering; den tester på om hver rad oppfyller kravet for å bli overført til resultat-tabellen

30. Hvor intuitivt er denne linken mellom kolonnene og querien?

Ikke intuitivt	Litt intuitivt	Veldig intuitivt
○	○	○

Hvorfor?

31. Andre kommentarer eller forslag for forbedring av prototypen og visualiseringene?

F Notification form NSD experiment

MELDESKJEMA

Meldeskjema (versjon 1.4) for forsknings- og studentprosjekt som medfører meldeplikt eller konsesjonsplikt (jf. personopplysningsloven og helseregisterloven med forskrifter).

1. Intro		
Samles det inn direkte personidentifiserende opplysninger?	Ja ● Nei ○	En person vil være direkte identifiserbar via navn, personnummer, eller andre personentydige kjennetegn. Les mer om hva personopplysninger. NB! Selv om opplysningene skal anonymiseres i oppgave/rapport, må det krysses av dersom det skal innhentes/registreres personidentifiserende opplysninger i forbindelse med prosjektet.
Hvis ja, hvilke?	■ Navn □ 11-sifret fødselsnummer □ Adresse ■ E-post □ Telefonnummer □ Annet	
Annet, spesifiser hvilke		
Skal direkte personidentifiserende opplysninger kobles til datamaterialet (koblingsnøkkel)?	Ja ○ Nei ●	Merk at meldeplikten utløses selv om du ikke får tilgang til koblingsnøkkel, slik fremgangsmåten ofte er når man benytter en databehandler
Samles det inn bakgrunnsopplysninger som kan identifisere enkeltpersoner (indirekte personidentifiserende opplysninger)?	Ja ● Nei ○	En person vil være indirekte identifiserbar dersom det er mulig å identifisere vedkommende gjennom bakgrunnsopplysninger som for eksempel bostedskommune eller arbeidsplass/skole kombinert med opplysninger som alder, kjønn, yrke, diagnose, etc. NB! For at stemme skal regnes som personidentifiserende, må denne bli registrert i kombinasjon med andre opplysninger, slik at personer kan gjenkjennes.
Hvis ja, hvilke	Skole, skolested, alder og kjønn	
Skal det registreres personopplysninger (direkte/indirekte/via IP-/epost adresse, etc) ved hjelp av nettbaserte spørreskjema?	Ja ● Nei ○	Les mer om nettbaserte spørreskjema.
Blir det registrert personopplysninger på digitale bilde- eller videoopptak?	Ja ○ Nei ●	Bilde/videoopptak av ansikter vil regnes som personidentifiserende.
Søkes det vurdering fra REK om hvorvidt prosjektet er omfattet av helseforskningsloven?	Ja ○ Nei ●	NB! Dersom REK (Regional Komité for medisinsk og helsefaglig forskningsetikk) har vurdert prosjektet som helseforskning, er det ikke nødvendig å sende inn meldeskjema til personvernombudet (NB! Gjelder ikke prosjekter som skal benytte data fra pseudonyme helseregistre). Dersom tilbakemelding fra REK ikke foreligger, anbefaler vi at du avventer videre utfylling til svar fra REK foreligger.
2. Prosjekttittel		
Prosjekttittel	ViSQLizer: en brukertest av et SQL læringsverktøy for høgskole-studenter.	Oppgi prosjektets tittel. NB! Dette kan ikke være «Masteroppgave» eller liknende, navnet må beskrive prosjektets innhold.
3. Behandlingsansvarlig institusjon		
Institusjon	NTNU	Velg den institusjonen du er tilknyttet. Alle nivå må oppgis. Ved studentprosjekt er det studentens tilknytning som er avgjørende. Dersom institusjonen ikke finnes på listen, har den ikke avtale med NSD som personvernombud. Vennligst ta kontakt med institusjonen.
Avdeling/Fakultet	NTNU i Gjøvik	
Institutt	Avdeling for informatikk og medieteknikk	
4. Daglig ansvarlig (forsker, veileder, stipendiat)		

Fornavn	Rune	Før opp navnet på den som har det daglige ansvaret for prosjektet. Veileder er vanligvis daglig ansvarlig ved studentprosjekt. Veileder og student må være tilknyttet samme institusjon. Dersom studenten har ekstern veileder, kanbiveileder eller fagansvarlig ved studiestedet stå som daglig ansvarlig. Arbeidssted må være tilknyttet behandlingsansvarlig institusjon, f.eks. underavdeling, institutt etc. NB! Det er viktig at du oppgir en e-postadresse som brukes aktivt. Vennligst gi oss beskjed dersom den endres.
Etternavn	Hjelsvold	
Stilling	Professor	
Telefon	61135184	
Mobil		
E-post	rune.hjelsvold@ntnu.no	
Alternativ e-post	rune.hjelsvold@ntnu.no	
Arbeidssted	NTNU Gjøvik	
Adresse (arb.)	Teknologivegen. 22	
Postnr./sted (arb.sted)	2815 Gjøvik	
Sted (arb.sted)	Gjøvik	

5. Student (master, bachelor)

Studentprosjekt	Ja ● Nei ○	Dersom det er flere studenter som samarbeider om et prosjekt, skal det velges en kontaktperson som føres opp her. Øvrige studenter kan føres opp under pkt 10.
Fornavn	Kristin Annabel	
Etternavn	Folland	
Telefon	47657330	
Mobil	47657330	
E-post	kristinannabel.folland@gmail.com	
Alternativ e-post	kristin.folland@hig.no	
Privatadresse	Hovdevegen 4A, H0202	
Postnr./sted (privatadr.)	2821 Gjøvik	
Sted (arb.sted)	Gjøvik	
Type oppgave	● Masteroppgave ○ Bacheloroppgave ○ Semesteroppgave ○ Annet	

6. Formålet med prosjektet

Formål	Formålet er å teste om en visualisering av databasetabeller kan hjelpe studenter med å forstå den underliggende logikken av SQL. Dette er et eksperiment som skal utføres på slutten av en periode med utvikling av et læringssystem, og studenter skal teste ut systemet, gi tilbakemeldinger, og svare på en test for at vi skal kunne fastslå om studentene forstår den underliggende logikken bedre etter at de har jobbet med systemet, enn før.	Redegjør kort for prosjektets formål, problemstilling, forskningsspørsmål e.l.

7. Hvilke personer skal det innhentes personopplysninger om (utvalg)?

Kryss av for utvalg	□ Barnehagebarn □ Skoleelever □ Pasienter □ Brukere/klienter/kunder □ Ansatte □ Barnevernsbarn □ Lærere □ Helsepersonell □ Asylsøkere ■ Andre	
Beskriv utvalg/deltakere	Studenter fra 1. år bachelorutdanning innen data/spill-programmering	Med utvalg menes dem som deltar i undersøkelsen eller dem det innhentes opplysninger om.

Rekruttering/trekking	Rekrutteres gjennom skole. Mail sendes ut, der det blir bedt om deltagelse av frivillige.	Beskriv hvordan utvalget trekkes eller rekrutteres og oppgi hvem som foretar den. Et utvalg kan trekkes fra registre som f.eks. Folkeregisteret, SSB-registre, pasientregistre, eller det kan rekrutteres gjennom f.eks. en bedrift, skole, idrettsmiljø eller eget nettverk.
Førstegangskontakt	Først via mail, deretter på brukertest-dagen mellom forsker og studenter.	Beskriv hvordan kontakt med utvalget blir opprettet og av hvem. Les mer om dette på temasidene.
Alder på utvalget	□ Barn (0-15 år) □ Ungdom (16-17 år) ■ Voksne (over 18 år)	Les om forskning som involverer barn på våre nettsider.
Omtrentlig antall personer som inngår i utvalget	Mulig mellom 10-30	
Samles det inn sensitive personopplysninger?	Ja ○ Nei ●	Les mer om sensitive opplysninger.
Hvis ja, hvilke?	□ Rasemessig eller etnisk bakgrunn, eller politisk, filosofisk eller religiøs oppfatning □ At en person har vært mistenkt, siktet, tiltalt eller dømt for en straffbar handling □ Helseforhold □ Seksuelle forhold □ Medlemskap i fagforeninger	
Inkluderes det myndige personer med redusert eller manglende samtykkekompetanse?	Ja ○ Nei ●	Les mer om pasienter, brukere og personer med redusert eller manglende samtykkekompetanse.
Samles det inn personopplysninger om personer som selv ikke deltar (tredjepersoner)?	Ja ○ Nei ●	Med opplysninger om tredjeperson menes opplysninger som kan spores tilbake til personer som ikke inngår i utvalget. Eksempler på tredjeperson er kollega, elev, klient, familiemedlem.

8. Metode for innsamling av personopplysninger

Kryss av for hvilke datainnsamlingsmetoder og datakilder som vil benyttes	■ Papirbasert spørreskjema ■ Elektronisk spørreskjema ■ Personlig intervju □ Gruppeintervju ■ Observasjon □ Deltakende observasjon □ Blogg/sosiale medier/internett □ Psykologiske/pedagogiske tester □ Medisinske undersøkelser/tester □ Journaldata	Personopplysninger kan innhentes direkte fra den registrerte f.eks. gjennom spørreskjema,intervju, tester, og/eller ulike journaler (f.eks. elevmapper, NAV, PPT, sykehus) og/eller registre (f.eks.Statistisk sentralbyrå, sentrale helseregistre). NB! Dersom personopplysninger innhentes fra forskjellige personer (utvalg) og med forskjellige metoder, må dette spesifiseres i kommentar-boksen. Husk også å legge ved relevante vedlegg til alle utvalgs-gruppene og metodene som skal benyttes. Les mer om registerstudier her. Dersom du skal anvende registerdata, må variabelliste lastes opp under pkt. 15
	□ Registerdata	
	□ Annen innsamlingsmetode	
Tilleggsopplysninger	Observasjon av testing av systemet, intervju av noen av studentene, og spørreskjema, enten elektronisk eller papirbasert.	

9. Informasjon og samtykke

Oppgi hvordan utvalget/deltakerne informeres	■ Skriftlig ■ Muntlig □ Informeres ikke	Dersom utvalget ikke skal informeres om behandlingen av personopplysninger må det begrunnes. Les mer her. Vennligst send inn mal for skriftlig eller muntlig informasjon til deltakerne sammen med meldeskjema. Last ned en veiledende mal her. NB! Vedlegg lastes opp til sist i meldeskjemaet, se punkt 15 Vedlegg.
Samtykker utvalget til deltakelse?	● Ja ○ Nei ○ Flere utvalg, ikke samtykke fra alle	For at et samtykke til deltakelse i forskning skal være gyldig, må det være frivillig, uttrykkelig og informert. Samtykke kan gis skriftlig, muntlig eller gjennom en aktiv handling. For eksempel vil et besvart spørreskjema være å regne som et aktivt samtykke. Dersom det ikke skal innhentes samtykke, må det begrunnes.

10. Informasjonssikkerhet

Spesifiser	Disse skal kun oppbevares inntil brukertesten er gjennomført, og dataene er samlet inn. Etter dette, skal disse destrueres og ikke oppbevares av noen parter.	NB! Som hovedregel bør ikke direkte personidentifiserende opplysninger registreres sammen med det øvrige datamaterialet.
Hvordan registreres og oppbevares personopplysningene?	■ På server i virksomhetens nettverk □ Fysisk isolert PC tilhørende virksomheten (dvs. ingen tilknytning til andre datamaskiner eller nettverk, interne eller eksterne) □ Datamaskin i nettverkssystem tilknyttet Internett tilhørende virksomheten □ Privat datamaskin □ Videoopptak/fotografi □ Lydopptak □ Notater/papir □ Mobile lagringsenheter (bærbar datamaskin, minnepenn, minnekort, cd, ekstern harddisk, mobiltelefon) □ Annen registreringsmetode	Merk av for hvilke hjelpemidler som benyttes for registrering og analyse av opplysninger. Sett flere kryss dersom opplysningene registreres på flere måter. Med «virksomhet» menes her behandlingsansvarlig institusjon. NB! Som hovedregel bør data som inneholder personopplysninger lagres på behandlingsansvarlig sin forskningsserver. Lagring på andre medier - som privat pc, mobiltelefon, minnepinne, server på annet arbeidssted - er mindre sikkert, og må derfor begrunnes. Slik lagring må avklares med behandlingsansvarlig institusjon, og personopplysningene bør krypteres.
Annen registreringsmetode beskriv		
Hvordan er datamaterialet beskyttet mot at uvedkommende får innsyn?	PC vil låses inn i skap når den ikke er i bruk, og står ellers inne på et låst rom i virksomheten.	Er f.eks. datamaskintilgangen beskyttet med brukernavn og passord, står datamaskinen i et låsbart rom, og hvordan sikres bærbare enheter, utskrifter og opptak?
Samles opplysningene inn/behandles av en databehandler?	Ja ○ Nei ●	Dersom det benyttes eksterne til helt eller delvis å behandle personopplysninger, f.eks. Questback, transkriberingsassistent eller tolk, er dette å betrakte som en databehandler. Slike oppdrag må kontraktsreguleres.
Hvis ja, hvilken		
Overføres personopplysninger ved hjelp av e-post/Internett?	Ja ○ Nei ●	F.eks. ved overføring av data til samarbeidspartner, databehandler mm. Dersom personopplysninger skal sendes via internett, bør de krypteres tilstrekkelig. Vi anbefaler for ikke lagring av personopplysninger på nettskytjenester. Dersom nettskytjeneste benyttes, skal det inngås skriftlig databehandleravtale med leverandøren av tjenesten.
Hvis ja, beskriv?		
Skal andre personer enn daglig ansvarlig/student ha tilgang til datamaterialet med personopplysninger?	Ja ○ Nei ●	
Hvis ja, hvem (oppgi navn og arbeidssted)?		
Utleveres/deles personopplysninger med andre institusjoner eller land?	● Nei ○ Andre institusjoner ○ Institusjoner i andre land	F.eks. ved nasjonale samarbeidsprosjekter der personopplysninger utveksles eller ved internasjonale samarbeidsprosjekter der personopplysninger utveksles.

11. Vurdering/godkjenning fra andre instanser

Søkes det om dispensasjon fra taushetsplikten for å få tilgang til data?	Ja ○ Nei ●	For å få tilgang til taushetsbelagte opplysninger fra f.eks. NAV, PPT, sykehus, må det søkes om dispensasjon fra taushetsplikten. Dispensasjon søkes vanligvis fra aktuelt departement.
Hvis ja, hvilke		
Søkes det godkjenning fra andre instanser?	Ja ○ Nei ●	F.eks. søke registereier om tilgang til data, en ledelse om tilgang til forskning i virksomhet, skole.
Hvis ja, hvilken		

12. Periode for behandling av personopplysninger

Prosjektstart Planlagt dato for prosjektslutt	29.03.2016 01.06.2016	Prosjektstart Vennligst oppgi tidspunktet for når kontakt med utvalget skal gjøres/datainnsamlingen starter. Prosjektslutt: Vennligst oppgi tidspunktet for når datamaterialet enten skalanonymiseres/slettes, eller arkiveres i påvente av oppfølgingsstudier eller annet.
Skal personopplysninger publiseres (direkte eller indirekte)?	□ Ja, direkte (navn e.l.) □ Ja, indirekte (bakgrunnsopplysninger) ■ Nei, publiseres anonymt	NB! Dersom personopplysninger skal publiseres, må det vanligvis innhentes eksplisitt samtykke til dette fra den enkelte, og deltakere bør gis anledning til å lese gjennom og godkjenne sitater.

Hva skal skje med datamaterialet ved prosjektslutt?	■ Datamaterialet anonymiseres □ Datamaterialet oppbevares med personidentifikasjon	NB! Her menes datamaterialet, ikke publikasjon. Selv om data publiseres med personidentifikasjon skal som regel øvrig data anonymiseres.Med anonymisering menes at datamaterialet bearbeides slik at det ikke lenger er mulig å føre opplysningene tilbake til enkeltpersoner. Les mer om anonymisering.

13. Finansiering

Hvordan finansieres prosjektet?	Personlig finansiering, forsker betaler selv utgifter.	

14. Tilleggsopplysninger

Tilleggsopplysninger	Skal dele deltagerne opp i to grupper, med kontrollgruppe i tillegg til hovedgruppen. Kontrollgruppen vil ikke bli gitt samme informasjon som testgrupper, og vil heller ikke teste det samme systemet som testgruppen. Jeg leverer dermed to skjemaer for informasjonsskriv.	

G Information letter for control group in experiment

Forespørsel om deltakelse i forskningsprosjektet

"SQL"

Bakgrunn og formål
I denne masteroppgaven ved NTNU Gjøvik skal læring av SQL undersøkes. Et verktøy for å lære SQL fra internett vil bli undersøkt for å se om dette fungerer optimalt som et læringsverktøy ved siden av ordinær undervisning ved universitetet. I dette studiet vil 1. års bachelorstudenter bli bedt om å bruke litt tid med verktøyet, og løse noen oppgaver, for å se om dette er et optimalt verktøy å bruke for SQL-læring. Deltagere av studiet vil ha muligheten til å evaluere verktøyet og komme med egne meninger om bruken av et slikt verktøy ved siden av ordinær undervisning av SQL.

Du har blitt trukket ut som utvalg av dette studiet pga. din nåværende utdanning innen data/IT/spill, og fordi ditt kull enda ikke har gjennomgått et SQL/database-kurs ved universitetet. Du forespørres om å delta da du er i brukergruppen av studenter som mulig vil ha brukt for et slikt verktøy.

Hva innebærer deltakelse i studien?
Deltakelse i studien innebærer å se på en kort intro-video for å lære basekunnskaper om SQL, å jobbe med oppgaver i angitt læringssystem på nett, svare på noen korte, skriftlige spørsmål, svare på en kort SQL-test og eventuelt delta på et kort intervju på slutten.

Spørsmålene vil omhandle din oppfattelse av læringssystemet, og din mening om bruken av et slikt system ved siden av ordinær undervisning. Testen vil ha enkle SQL-queries, der du skal svare det du mener blir riktig svar.

Studiet vil foregå ved bruk av din personlige datamaskin. Dersom du blir trukket ut til å bli med på et intervju, vil dette bli tatt opp med en enkel lydopptaker. Testen vil foregå på papir. All data vil være anonym. Data innhentet om deg, som navn og e-post, vil bli slettet etter at studiet er fullført, og dine svar vil ikke kunne spores tilbake til deg som person.

Hva skjer med informasjonen om deg?
Alle personopplysninger vil bli behandlet konfidensielt. Kun forsker vil ha tilgang til personopplysningene, og personopplysninger/opptak vil bli lagret på en sikker måte for å ivareta konfidensialitet. Du vil ikke kunne gjenkjennes i publikasjonen.

Prosjektet skal etter planen avsluttes 1. Juni 2016. Innen den dato vil alle personopplysninger og opptak slettes.

Frivillig deltakelse
Det er frivillig å delta i studien, og du kan når som helst trekke ditt samtykke uten å oppgi noen grunn. Dersom du trekker deg, vil alle opplysninger om deg bli anonymisert.

Dersom du har spørsmål til studien, ta kontakt med prosjektleder
Kristin Annabel Folland, tlf: 47657330, e-post: kristinannabel.folland@gmail.com
Eller veileder
Rune Hjelsvold, rune.hjelsvold@ntnu.no

Studien er meldt til Personvernombudet for forskning, NSD - Norsk senter for forskningsdata AS.

Har du erfaring med SQL fra tidligere?
(Sett ring rundt det svaret du føler er mest riktig for deg)

Nei Noe erfaring Mye erfaring

Samtykke til deltakelse i studien

Jeg har mottatt informasjon om studien, og er villig til å delta

----
(Fult navn i blokkbokstaver)

(Signert av prosjektdeltaker, dato)

H Information letter for experimental group in experiment

Forespørsel om deltakelse i forskningsprosjektet

”ViSQLizer”

Bakgrunn og formål
I denne masteroppgaven ved NTNU Gjøvik skal læring av SQL undersøkes. Et verktøy for å lære SQL, utviklet av prosjektleder/masterstudent, vil bli undersøkt for å se om dette fungerer optimalt som et læringsverktøy ved siden av ordinær undervisning ved universitet. I dette studiet vil 1. års bachelorstudenter bli bedt om å bruke litt tid med verktøyet, og løse noen oppgaver, for å se om dette er et optimalt verktøy å bruke for SQL-læring. Deltagere av studiet vil ha muligheten til å evaluere verktøyet og komme med egne meninger om bruken av et slikt verktøy ved siden av ordinær undervisning av SQL.

Du har blitt trukket ut som utvalg av dette studiet pga. din nåværende utdanning innen data/IT/spill, og fordi ditt kull enda ikke har gjennomgått et SQL/database-kurs ved universitetet. Du forespørres om å delta da du er i brukergruppen av studenter som mulig vil ha brukt for et slikt verktøy.

Hva innebærer deltakelse i studien?
Deltakelse i studien innebærer å se på en kort intro-video for å lære basekunnskaper om SQL, å jobbe med oppgaver i angitt læringssystem på nett, svare på noen korte, skriftlige spørsmål, svare på en kort SQL-test og eventuelt et kort intervju på slutten.

Spørsmålene vil omhandle din oppfattelse av læringssystemet, og din mening om bruken av et slikt system ved siden av ordinær undervisning. Testen vil ha enkle SQL-queries, der du skal svare det du mener blir riktig svar.

Studiet vil foregå ved bruk av din personlige datamaskin. Dersom du blir trukket ut til å bli med på et intervju, vil dette bli tatt opp med en enkel lydopptaker. Testen vil foregå på papir. All data vil være anonym. Data innhentet om deg, som navn og e-post, vil bli slettet etter at studiet er fullført, og dine svar vil ikke kunne spores tilbake til deg som person.

Hva skjer med informasjonen om deg?
Alle personopplysninger vil bli behandlet konfidensielt. Kun forsker vil ha tilgang til personopplysningene, og personopplysninger/opptak vil bli lagret på en sikker måte for å ivareta konfidensialitet. Du vil ikke kunne gjenkjennes i publikasjonen.

Prosjektet skal etter planen avsluttes 1. Juni 2016. Innen den dato vil alle personopplysninger og opptak slettes.

Frivillig deltakelse
Det er frivillig å delta i studien, og du kan når som helst trekke ditt samtykke uten å oppgi noen grunn. Dersom du trekker deg, vil alle opplysninger om deg bli anonymisert.

Dersom du har spørsmål til studien, ta kontakt med prosjektleder
Kristin Annabel Folland, tlf: 47657330, e-post: kristinannabel.folland@gmail.com
Eller veileder
Rune Hjelsvold, rune.hjelsvold@ntnu.no

Studien er meldt til Personvernombudet for forskning, NSD - Norsk senter for forskningsdata AS.

Har du erfaring med SQL fra tidligere?
(Sett ring rundt det svaret du føler er mest riktig for deg)

Nei Noe erfaring Mye erfaring

Samtykke til deltakelse i studien

Jeg har mottatt informasjon om studien, og er villig til å delta

----
(Fult navn i blokkbokstaver)

(Signert av prosjektdeltaker, dato)

I Exercises for the control group in experiment

Øvingsoppgaver for viSQLizer

I SQL skriver du en setning sammensatt av betingelser, klausuler og operasjoner. Denne setningen kalles en spørring (query på engelsk), og det er denne du skal skrive i øverste input-felt på viSQLizer-nettsiden. Skriv inn spørringene som nevnes i de forskjellige oppgavene nedenfor, se på animasjonen og prøv å forstå sammenhengen mellom spørringen, resultat-tabellen og original-tabellen(e). Dersom du ønsker å se en animasjon igjen, kan du trykke på det aktive steget under resultat-tabellen, eller kjøre spørringen på nytt.

Select fra en tabell

1. **Se hele tabellen:**
   SELECT forteller databasen din at du ønsker å velge data. **FROM** student forteller databasen din at du ønsker å velge data fra tabellen student. **Asterisken (*)** forteller databasen at vi ønsker å se **alle kolonner** i denne tabellen.

   a. Velg alle kolonner for alle studenter:
   *SELECT * FROM student*

2. **Velg noen kolonner i tabellen:**
   Dersom vi ikke ønsker å velge **alle kolonner** fra tabellen, kan vi skrive inn **kolonne-navnet** i stedet for asterisken (*).

   a. Velg student-nummer for alle studenter:
   SELECT student_no FROM student

   b. Velg kolonnene navn og alder for alle studenter:
   SELECT name, age FROM student

3. **Filtrere rader:**
   Dersom vi ønsker å få ut spesifikke rader, kan vi legge til en **WHERE** og en **betingelse**. En betingelse bestemmer hvilke brukere vi ønsker å se, basert på deres bruker-id.

   Den enkleste betingelsen er **=**, der vi ønsker å finne brukeren med bruker-id 2, så vi bruker likhet (userid = 2).

   a. Velg brukere med bruker-id 2:
   *SELECT * FROM user WHERE userid = 2*

Det finnes også andre betingelser som du kan bruke. I stedet for likhetstegnet (=) bruker vi **mindre-enn-tegnet (<)**. Nå kan vi velge brukere som har en bruker-id mindre enn 3. Vi kan bruke flere operasjoner på samme måte:
< (mindre enn)
> (større enn)
<= (mindre enn eller lik)
>= (større enn eller lik)

b. Velg alle brukere med bruker-id større enn 3:
*SELECT * FROM user WHERE userid > 3*

Det finnes en til viktig betingelses-operasjon, **ulikhets-tegnet (!=).** Vi kan velge alle brukere som ikke har bruker-id 1. I SQL kan man også bruke **operasjonen <>** for å få samme resultat.

c. Velg alle brukere som ikke har bruker-id 1:
*SELECT * FROM user WHERE userid != 1*

La oss nå kombinere det vi har lært om betingelser med å velge spesifikke kolonner. I stedet for å bruke asterisken (*), velger vi bare de spesifikke kolonnene som vi er interessert i.

d. Velg bruker-id og brukernavn på alle brukere med bruker-id lavere eller lik 2:
SELECT userid, username FROM user WHERE userid <= 2
(Dette skjer i flere steg)

4. **Logikk:**

Noen ganger ønsker vi å være veldig spesifikke og bruke flere betingelsen sammen. En **OR-klausul** lar oss bruke flere betingelser. Vi ønsker kun å vise de eksamenene som ble utført etter år 2013 **eller** har en score lavere enn 4. Men andre ord, en rad blir valgt dersom den første betingelsen **eller** den andre betingelsen er **sann**. Dersom begge betingelsen er sanne, vil også raden bli valgt.

a. Velg student-nummer, år og score for alle eksamener som er utført etter år 2013 eller har en score lavere enn 4:
SELECT student_no, year, score FROM exam WHERE year > 2013 OR score < 4
(Dette skjer i flere steg)

AND er en annen logisk operator. Vi ønsker å vise de eksamenene som ble utført før år 2014 **og** har en score høyere enn 3. Med andre ord, en rad blir vist dersom **begge** betingelser blir **sanne**.

b. Velg student-nummer, år og score for alle eksamener som er utført før år 2014 og har en score høyere enn 3:
SELECT student_no, year, score FROM exam WHERE year > 2013 AND score > 3
(Dette skjer i flere steg)

Når vi ønsker å vise eksamener som ble urført i årene mellom 2013 og 2015, kan vi selvfølgelig skrive følgende: *SELECT * FROM exam WHERE year < 2015 AND year > 2013*. Men det finnes en annen måte å skrive denne spørringen, med **BETWEEN**, som enkelt betyr at vi ønsker å vise rader som har data i year-kolonnen som er alt **innenfor** 2013 og 2015**, inkludert disse verdiene**.

c. Velg alle kolonner for alle eksamener som er utført mellom år 2013 og 2015:
*SELECT * FROM exam WHERE year BETWEEN 2013 AND 2015*

Det finnes også en annen logisk operator, **NOT**. Alt som blir skrevet etter NOT vil bli opphevet. Når vi setter NOT forann BETWEEN, vises alle rader i tabellen **bortsett fra** de med year 2013 til 2015.

d. Velg alle kolonner for alle eksamener som ikke er utført mellom år 2013 og 2015:
*SELECT * FROM exam WHERE year NOT BETWEEN 2013 AND 2015*

5. **Udefinerte-verdier (NULL):**
I alle rader kan det være **NULL** verdier, m.a.o. felt uten noen verdi/felt som er udefinerte. I exam-tabellen finnes det en eksamen med score NULL, som betyr at studenten ikke møtte opp på eksamen, m.a.o. vi kjenner ikke til scoren. For å sjekke om en kolonne har en verdi, bruker vi en spesiell instruksjon **IS NOT NULL**. Dermed velger vi kun de eksamenene som har en score, altså hvor kolonnen score er kjent.

a. Velg alle kolonner for alle eksamener hvor score ikke har en NULL-verdi:
*SELECT * FROM exam WHERE score IS NOT NULL*

NULL er en spesiell verdi. Du kan ikke bruke er-lik tegnet for å sjekke om noe er NULL. Det motsatte av IS NOT NULL er **IS NULL**. Dersom vi bruker denne instruksjonen, vil alle eksamener hvor score er ukjent bli vist.

b. Velg alle kolonner for alle eksamener hvor score har en NULL-verdi:
*SELECT * FROM exam WHERE score IS NULL*

Select fra flere tabeller

6. **Mer enn en tabell:**
I store databaser bruker vi alltid flere tabeller. Dette betyr også at vi ofte ønsker å hente data fra mer enn en tabell på samme tid. Vi vet hvilken student som tok hver eksamen, fordi det er en kolonne student_no i tabellen exam.

Den enkleste måten å hente data fra flere tabeller på er ved bruk av **komma** mellom de to tabellene. Du skriver SELECT * FROM som vi har gjort tidligere, og navngir to tabeller i stedet for en, men en komma som separerer de to.

  a. Velg alle kolonner for alle studenter og eksamener:
  *SELECT * FROM student, exam*

I en slik type sammensetting av tabeller, tar SQL og setter hver rad fra student-tabellen sammen med hver rad fra eksamens-tabellen, noe som gir hele 12 (3*4) rader! Dette skjer fordi SQL ikke vet hva som skal gjøres med resultatet fra de to tabellene, så den gir oss **alle mulige sammenkoblinger**. Dette kan vi endre ved å sette en betingelse i **WHERE**-klausulen. Vi vil kun se de sammenkoblingene hvor student_no fra student er den samme som student_no i exam. Dersom du har flere tabeller, burde du referere til spesifikke kolonner ved å angi navnet på tabellen og kolonnen, separert med et **punktum**.

  b. Velg alle kolonner for alle studenter, slik at hver student er vist sammen med sine eksamener:
  *SELECT * FROM student, exam WHERE student.student_no = exam.student_no*
  (Dette skjer i flere steg)

7. **JOIN:**

Det å sette sammen to tabeller er veldig populært og gjøres så ofte at SQL har et spesielt ord for det: **JOIN**. Det finnes mange versjoner av JOIN, men for nå skal vi bare fokusere på den mest grunnleggende versjonen. Vi bytter ut kommaet fra forrige oppgave med JOIN for å sette sammen de to tabellene. SQL burde også vite hvordan de to tabellene skal settes sammen, så vi bruker et annet ord: **ON**. Etter dette setter vi vår **betingelse**; join bare de radene hvor student_no i student er det samme som student_no i exam.

  a. Velg alle kolonner for alle studenter og eksamener ved bruk av JOIN, slik at alle studenter blir vist sammen med sine eksamener:
  *SELECT * FROM student JOIN exam ON student.student_no = exam.student_no*

8. **Referere til kolonner i JOIN:**

La oss nå si at vi bare trenger noen få kolonner i resultatet vårt. Vi ønsker bare å vite studentnavnet og studentenes score på eksamen. I stedet for asterisken (*) putter vi inn **kolonne-navnene**. Siden vi nå har mer enn en tabell, putter vi **tabellnavnet foran kolonnenavnet**, og separerer de med en **punktum**. På denne måten vet SQL at kolonnen name tilhører tabellen student osv.

  a. Velg student-navn og studentenes score på eksamen for alle studenter og eksamener, slik at alle studenter blir vist med sine eksamener:
  SELECT student.name, exam.score FROM student JOIN exam ON student.student_no = exam.student_no
  (Dette skjer i flere steg)

Nå ønsker vi å filtrere resultatene enda mer. Vi kan legge til en **WHERE** klausul, slik at vi bare ser etter koblinger mellom studenter og deres eksamener hvor eksamenene er gjennomført i år 2012. Husk å **inkludere tabell-navnet i betingelsen** (exam.year).

b. Velg alle kolonner for alle studenter og eksamener, slik at hver student blir vist med sine eksamener. Vis bare studenter som har gjennomført en eksamen i 2012:
*SELECT * FROM student JOIN exam ON student.student_no = exam.student_no WHERE exam.year = 2012*
(Dette skjer i flere steg)

9. Left JOIN:
Når man skriver en enkel JOIN, som vi har gjort i oppgave 7 og 8, kalles dette for en INNER JOIN. Nå skal vi lære en ny type JOIN: **LEFT JOIN**, eller **LEFT OUTER JOIN** som det også kan bli kalt. LEFT JOIN fungerer som følgende: den returnerer alle rader fra **den venstre (left) tabellen** (den første tabellen i spørringen) pluss alle matchende rader fra den venstre tabellen (den andre tabellen i spørringen). Dersom det ikke er noen matchende eksamen for en student, vil studenten likevel bli vist i resultatet.

a. Husker du resultatet fra oppgave **7a**? Sammenlign dette resultatet med resultatet fra denne oppgaven. Velg alle kolonner for alle student. Vis studentene selv om de ikke har hatt noen eksamen enda:
*SELECT * FROM student LEFT JOIN exam ON student.student_no = exam.student_no*

Avsluttende spørsmål

Disse spørsmålene omhandler visualiseringa i viSQLizer-prototypen du nettopp har testet. Dersom du ikke husker noen av detaljene relatert til spørsmålene, er du velkommen til å teste prototypen mer før du svarer på spørsmålene under.

1. Hva syns du om animasjonen av teksten?
   □ - Uoversiktlig
   □ - For sakte animasjon
   □ - For rask animasjon
   □ - Oversiktlig
   Annet: (Spesifiser)

   ____

   ____

   a. Klarer du å følge hvor dataene kommer fra og hvor de plasseres i resultat-tabellen? Hvorfor/Hvorfor ikke?

      ____

      ____

      ____

2. Virker det positivt å kunne gå frem og tilbake for å se animasjonene flere ganger? Hvorfor/Hvorfor ikke?

   ____

   ____

   ____

3. Hvor intuitiv er linken mellom kolonner i WHERE/ON-klausul og spørringen, ved bruk av blå markering av celler?

   ____

   ____

   ____

4. Hva syns du om denne formen for læring?

   ____

____

____

____

a. Syns du det er et bra alternativ til å sitte i forelesninger?

____

____

____

b. Hva gjør dette til et så bra/dårlig alternativ i forhold til ordinær undervisning/forelesning?

____

____

____

5. Andre kommentarer til prototypen?

____

____

____

____

J Exercises for the experimental group in experiment

Øvingsoppgaver SQL med VertabeloAcademy

Gå inn på https://academy.vertabelo.com/course/sql-queries for å starte med øvingsoppgavene. Du må logge inn for å bruke løsningen, men du kan velge å logge inn med Facebook, Twitter, google+ eller GitHub, om du heller ønsker dette. Bla ned på siden til Table of contents, og velg kurs der. Kursene er markert i denne øvingsoppgaven med blå tekst. Gå på menyen og velg de oppgavene som står nedenfor. Teksten markert med et tall er hoved-kategori i kurset, og teksten markert med bokstaver er oppgaver under disse kategoriene. Du kan velge å gjøre andre oppgaver i tillegg, men det er svært viktig at du gjennomfører disse først.

Selecting from one table

1. See the whole table
   a. Get all data

2. Select some columns
   a. Select one column
   b. Select many columns

3. Filtering rows
   a. Select only a few rows
   b. Conditional operators
   c. The not equal sign (!=)
   d. Conditional operators and selecting columns

4. Logic
   a. Logical operators – OR
   b. Logical operators – AND
   c. The BETWEEN operator
   d. Logical operators – NOT

5. To be or NULL to be
   a. Looking for NOT NULL values
   b. Looking for NULL values

Querying more than one table

6. How to query more than one table
   a. Data from multiple tables
   b. Get to know the tables
   c. Multiple tables in the clause FROM
   d. JOIN tables on a contion

7. Creating JOINs
    a. The keyword JOIN
    b. Join tables using JOIN

8. Referencing columns
    a. Display specific columns
    b. Filter the joined tables

More on JOINs

9. LEFT JOIN
    a. LEFT JOIN explained

Avsluttende spørsmål

Disse spørsmålene omhandler visualiseringa i viSQLizer-prototypen du nettopp har testet. Dersom du ikke husker noen av detaljene relatert til spørsmålene, er du velkommen til å teste prototypen mer før du svarer på spørsmålene under.

1. Hva syns du om designet på siden?
   □ - Uoversiktlig
   □ - Oversiktlig
   Annet: (Spesifiser)

   ____

   ____

   a. Er det enkelt å forstå hvordan resultatet blir laget? Hvorfor/Hvorfor ikke?

      ____

      ____

      ____

2. Virker det positivt å kunne gjøre små øvingsoppgaver underveis i læringen? Hvorfor/Hvorfor ikke?

   ____

   ____

   ____

3. Hva syns du om denne formen for læring?

   ____

   ____

   ____

   ____

a. Syns du det er et bra alternativ til å sitte i forelesninger?

____

____

____

b. Hva gjør dette til et så bra/dårlig alternativ i forhold til ordinær undervisning/forelesning?

____

____

____

4. Andre kommentarer til læringssystemet?

____

____

____

____

K Test questions for both groups in experiment

Test

Del 1

Denne testen består av 2 deler. I første del skal du fylle inn riktig data i tabellene, basert på spørringen som er skrevet ovenfor hver tomme tabell. I hver tabell skal du fylle inn riktig kolonne-navn i øverste rad, og deretter riktig data i radene nedover i tabellene. All informasjon du trenger for å løse oppgavene ligger i tabelldataene under.

Tabelldata

Tabell: bok			
ID	**TITTEL**	**ÅR**	**FORFATTER**
1	Snømannen	2007	Jo Nesbø
2	Etter deg	2016	Jojo Moyes
3	Politi	2013	Jo Nesbø
4	Great expectations	1861	Charles Dickens
5	Boiling a frog	2000	Christopher Brookmyre
6	Amatøren	1977	Lars S. Christensen

Tabell: utleie		
ID	**BOK_ID**	**UTLEIER**
1	4	Lars
2	1	Ole
3	5	Lise
4	1	NULL

Oppgaver: Skriv inn riktig data i tabellene

1. Vis tittel og forfatter på alle bøker:
   SELECT tittel, forfatter FROM bok

2. Vis alle bøker skrevet av Jo Nesbø:
   SELECT * FROM bok WHERE forfatter = ’Jo Nesbø’

3. Vis alle bøker med id mindre enn 4:
   SELECT * FROM bok WHERE id < 4

4. Vis alle bøker som er skrevet mellom år 2000 og 2016:
   SELECT * FROM bok WHERE year BETWEEN 2000 AND 2016

5. Vis all informasjon om alle utleiere som har oppgitt et navn:
   SELECT * FROM utleie WHERE utleier IS NOT NULL

6. Vis alle utleier, og alle bøkene som er blitt utleid:
   SELECT * FROM utleie JOIN bok ON utleie.bok_id = bok.id

Del 2

I del 2 får du oppgitt ferdige utfylte resultat-tabeller, og en kort tekst som beskriver scenarioet. Din oppgave her er å skrive spørringene som lager resultat-tabellene som er oppgitt.

Ny tabelldata

Tabell: regissør

ID	NAVN
1	Tim Burton
2	Michael Bay
3	Kirk Jones

Tabell: film

ID	TITTEL	ÅR	REGISSØR_ID
1	Mitt store fete greske bryllup 2	2016	3
2	Alice i eventyrland	2010	1
3	Armageddon	1998	2
4	Edward Saksehånd	1990	1
5	Enfilm	2001	NULL

Oppgaver: Skriv inn riktig spørring ut ifra tabell-dataene

Ps: Flere spørringer kan gi riktig svar! Prøv å skriv den du mener er mest ryddig.

1. Vis navnene på alle regissører:

NAVN
Tim Burton
Michael Bay
Kirk Jones

SVAR: ____

2. Vis alle regissører med ID mindre enn eller lik 2:

ID	NAVN
1	Tim Burton
2	Michael Bay

SVAR: ____

3. Vis alle filmer som er utgitt før år 2000:

ID	TITTEL	ÅR	REGISSØR_ID
3	Armageddon	1998	2
4	Edward Saksehånd	1990	1

SVAR: ____

____

4. Vis alle filmer hvor informasjonen om regissøren er udefinert:

ID	TITTEL	ÅR	REGISSØR_ID
5	Enfilm	2001	NULL

SVAR: ____

5. Vis alle filmer og informasjon om hver tilhørende regissør:

ID	TITTEL	ÅR	REGISSØR_ID	ID	NAVN
1	Mitt store fete greske bryllup 2	2016	3	3	Kirk Jones
2	Alice i eventyrland	2010	1	1	Tim Burton
3	Armageddon	1998	2	2	Michael Bay
4	Edward Saksehånd	1990	1	1	Tim Burton

SVAR: ____

____

6. Vis alle filmer og tilhørende regissører hvor regissørens navn er Tim Burton:

ID	TITTEL	ÅR	REGISSØR_ID	ID	NAVN
2	Alice i eventyrland	2010	1	1	Tim Burton
4	Edward Saksehånd	1990	1	1	Tim Burton

SVAR: ____

____

L Answers for test questions in experiment

Test FASIT

Tabelldata

Tabell: bok

ID	TITTEL	ÅR	FORFATTER
1	Snømannen	2007	Jo Nesbø
2	Etter deg	2016	Jojo Moyes
3	Politi	2013	Jo Nesbø
4	Great expectations	1861	Charles Dickens
5	Boiling a frog	2000	Christopher Brookmyre
6	Amatøren	1977	Lars S. Christensen

Tabell: utleie

ID	BOK_ID	UTLEIER
1	4	Lars
2	1	Ole
3	5	Lise
4	1	NULL

Oppgaver: Skriv inn riktig data i tabellene

1. Vis tittel og forfatter på alle bøker:
   SELECT tittel, forfatter FROM bok

TITTEL	FORFATTER
Snømannen	Jo Nesbø
Etter deg	Jojo Moyes
Politi	Jo Nesbø
Great expectations	Charles Dickens
Boiling a frog	Christopher Brookmyre
Amatøren	Lars S. Christensen

2. Vis alle bøker skrevet av Jo Nesbø:
   SELECT * FROM bok WHERE forfatter = ’Jo Nesbø’

ID	TITTEL	ÅR	FORFATTER
1	Snømannen	2007	Jo Nesbø
3	Politi	2013	Jo Nesbø

3. Vis alle bøker med id mindre enn 4:
   SELECT * FROM bok WHERE id < 4

ID	TITTEL	ÅR	FORFATTER
1	Snømannen	2007	Jo Nesbø
2	Etter deg	2016	Jojo Moyes
3	Politi	2013	Jo Nesbø

4. Vis alle bøker som er skrevet mellom år 2000 og 2016:
   SELECT * FROM bok WHERE year BETWEEN 2000 AND 2016

ID	TITTEL	ÅR	FORFATTER
1	Snømannen	2007	Jo Nesbø
2	Etter deg	2016	Jojo Moyes
3	Politi	2013	Jo Nesbø
5	Boiling a frog	2000	Christopher Brookmyre

5. Vis all informasjon om alle utleiere som har oppgitt et navn:
   SELECT * FROM utleie WHERE utleier IS NOT NULL

ID	BOK_ID	UTLEIER
1	4	Lars
2	1	Ole
3	5	Lise

6. Vis alle utleier, og alle bøkene som er blitt utleid:
   SELECT * FROM utleie JOIN bok ON utleie.bok_id = bok.id

ID	BOK_ID	UTLEIER	ID	TITTEL	ÅR	FORFATTER
1	4	Lars	4	Great expectations	1861	Charles Dickens
2	1	Ole	1	Snømannen	2007	Jo Nesbø
3	5	Lise	5	Boiling a frog	2000	Christopher Brookmyre
4	1	NULL	1	Snømannen	2007	Jo Nesbø

Ny tabelldata

Tabell: regissør

ID	NAVN
1	Tim Burton
2	Michael Bay
3	Kirk Jones

Tabell: film

ID	TITTEL	ÅR	REGISSØR_ID
1	Mitt store fete greske bryllup 2	2016	3
2	Alice i eventyrland	2010	1
3	Armageddon	1998	2
4	Edward Saksehånd	1990	1
5	Enfilm	2001	NULL

Oppgaver: Skriv inn riktig query ut ifra tabell-dataene

Ps: Flere querier kan gi riktig svar! Prøv å skriv den du mener er mest ryddig.

1.

NAVN
Tim Burton
Michael Bay
Kirk Jones

SVAR: ____

SELECT navn FROM regissør

2.

ID	NAVN
1	Tim Burton
2	Michael Bay

SVAR: ____

SELECT * FROM reggisør WHERE id <= 2

3.

ID	TITTEL	ÅR	REGISSØR_ID
3	Armageddon	1998	2
4	Edward Saksehånd	1990	1

SVAR: ____

____

SELECT * FROM film WHERE år < 2000

4.

ID	TITTEL	ÅR	REGISSØR_ID
5	Enfilm	2001	NULL

SVAR: ____

SELECT * FROM film WHERE regissør_id IS NULL

5.

ID	TITTEL	ÅR	REGISSØR_ID	ID	NAVN
1	Mitt store fete greske bryllup 2	2016	3	3	Kirk Jones
2	Alice i eventyrland	2010	1	1	Tim Burton
3	Armageddon	1998	2	2	Michael Bay

4	Edward Saksehånd	1990	1	1	Tim Burton

SVAR: ____

____

SELECT * FROM film JOIN regissør ON film.regissør_id = regissør.id

6.

ID	**TITTEL**	**ÅR**	**REGISSØR_ID**	**ID**	**NAVN**
2	Alice i eventyrland	2010	1	1	Tim Burton
4	Edward Saksehånd	1990	1	1	Tim Burton

SVAR: ____

____

SELECT * FROM film JOIN regissør ON film.regissør_id = regissør.id WHERE regissør.navn = ’Tim Burton’

M Interview questions for the experiment

Utkast intervjuguide/temaliste

1. Hvilket system testet du?
2. Når du skulle svare på del 1 av testen; så du for deg hvordan dataene ”flyter”/plasseres i resultat-tabellene når du svarte?
    a. Hva så du eventuelt for deg/tenkte du for å svare riktig på testen?
    b. Så du det på samme måte da du skrev spørringene?
    c. Hvorfor/Hvorfor ikke?
    d. Hvilke elementer i læringssystemet hjalp deg best til å svare på spørsmålene i testen? På hvilken måte?
    e. Dersom du skulle svart på testen på nytt, tror du det ville hjulpet deg til å svare bedre ved å se for deg hvordan dataene flyter?
    f. Hvorfor tenker du dette?

3. Hva syns du om å bruke et slikt læringssystem? Savnert du ordinær undervisning/en lærer å spørre om spørsmål?
4. Hva syns du om å bruke dette systemet ved siden av ordinær undervisning av databaser/SQL?
5. Ser du noen positive sider ved å bruke et slikt system? Hva?
6. Ser du noen negative sider ved å bruke et slikt system? Hva?
7. Virker systemet oversiktlig og enkelt å bruke? Hvorfor/hvorfor ikke?
8. Er det enkelte deler av systemet/visualiseringen som fremstår som nyttig? Hva?
9. Er det enkelte deler av systemet/visualiseringen som fremstår som unyttig/uforståelig? Hva?
10. Følte du at du fikk nok tid på å lære/øve deg på SQL før testen?
11. Følte du at testen var utformet på en forståelig måte?
12. Hvordan følte du nivået på test-spørsmålene var? (For lett/for vanskelig?)

N Client-side implementation

Stage and Canvas setup

```
var stage = new createjs.Stage("demoCanvas");
var canvas = document.getElementById("demoCanvas");
var table = document.getElementById("main-panel streammode-panel");
var tableDOMElement = new createjs.DOMElement("main-panel streammode-
    panel");

// Set canvas size to size of tables in SQL view
stage.canvas.width = tableDOMElement.htmlElement.clientWidth + 2 + 200;
stage.canvas.height = tableDOMElement.htmlElement.clientHeight + 2;

// Move the DOMElement at the center of the content
tableDOMElement.regX = table.offsetWidth * 0.5 + 100;
tableDOMElement.regY = table.offsetHeight * 0.5;

// Move the DOMElement to right position
tableDOMElement.x = canvas.width * 0.5;
tableDOMElement.y = canvas.height * -0.50;
$("#main-panel.streammode-panel").css("z-index", "2");
stage.addChild(tableDOMElement);
```

Process A: Looping through rows and columns

```
// Lopp through each row in empty-table (the last table in DOM-view with
    query result)
for (var i = 0; i < tableRows; i++)(function(i) {
    // Loop through each column on this (i) row in empty table
    for (var j = 0; j < tableColumns; j++)(function(j) {
        ...
    })(j);
    // Counting up the time counter for the animations by 2200ms
    timeCount += 2200;
})(i);
```

Process B: Locate target cell for animation

```
var thisGetTextOrigin = $(".original-table").find("span.textOrigin:not(.
    used):not(.notInUse)").filter(function() {
    return $(this).html() === textContent;
});

var getTextOrigin = checkIfDuplicatedData(textContent, thisGetTextOrigin)
    ;
```

```
while (isRightElem == false) {
    if((thisEmptyId == thisOriginalId) && (thisGetTextOrigin.first().
        parent().parent().parent().has(".usedInRow").length == 0) && (
        thisGetTextOrigin.first().parent().is(':first-child'))) {

        // If this is the first element in this row
        isRightElem = true;
        return $(thisGetTextOrigin[0]).parent().find("span.extraSpan").
            attr("id", "animThis").next();
        //animThis is the HTML ID to use for selecting the target cell
    }
...
}
```

Listing N.1: Process B1

```
else if ((thisGetTextOrigin.first().parent().parent().has(".usedInRow").
    length != 0) && (thisEmptyId == thisOriginalId)) {

        // If this has siblings with class usedInRow, siblings that have
            been used already on this row
        isRightElem = true;
        return $(thisGetTextOrigin[0]).parent().find("span.extraSpan").
            attr("id", "animThis").next();
}
```

Listing N.2: Process B2

```
else if((isNotFirstColumn) && (prevEmptyTableColumn !=
    prevOriginalTableColumn) &&  (!thisGetTextOrigin.first().parent().
    siblings().find("span:not(#original-span)").hasClass("used")) && (
    thisEmptyId == thisOriginalId)) {

        // If this element is first to be used in row even if it is not
            first element
        isRightElem = true;
        return $(thisGetTextOrigin[0]).parent().find("span.extraSpan").
            attr("id", "animThis").next();
}
```

Listing N.3: Process B3

```
else {
    // This element is not the right one, find next occurence
    thisGetTextOrigin.first().addClass("notInUse");
    var thisGetTextOrigin = $(".original-table").find("span.textOrigin:
        not(.used):not(.notInUse)").filter(function() {
            return $(this).html() === textContent;
    });
}
```

Listing N.4: Process B4

Process C: Find cell position

```
var resultPosX = $("#empty-table").find("tr:nth-child(" + rowCount + ")")
    .find("td:nth-child(" + columnCount + ")").find("span:not(.textOrigin)
    ").position().left;
var originalPosX = $(".original-table").find("span#animThis").position().
    left;
// Calculated position X from the original position and result position
var calcPositionX = resultPosX - originalPosX;

var originalPosY = $(".original-table").find("span#animThis").first().
    position().top;
var resultPosY = $("#empty-table").find("tr:nth-child(" + rowCount + ")")
    .find("td:nth-child(" + columnCount + ")").find("span:not(.textOrigin)
    ").position().top;
// Calculated position Y from the original position and result position
var calcPositionY = resultPosY - originalPosY;

//Move the cell if it crashes with the previous cell when resized
if ((animThisIndex > 0 && j > 0) && (((($allChangePositionX +
    $prevOrgPosXElem) + ($prevOrgWidth * 2)) >= (orgPosXElem +
    calcPositionX)) || ((($prevCalcPositionX + $prevOrgPosXElem) + (
    $prevOrgWidth * 2)) >= (orgPosXElem + calcPositionX)))) {
     var changeXPosition = $prevChangeXPosition + $prevOrgWidth * 0.5;
} else {
     var changeXPosition = 0;
}
```

Process D: Animation

```
var calcScalePositionX = calcPositionX;
var scaleX = 1.5;
var scaleY = 1.5;
var scaleTime = 300;
var newTimeCount = 600;
if (changeXPosition > 0) {
    if ((($prevOrgPosXElem + $prevOrgWidth) > orPosX) && (
        $prevOrgPosXElem < orPosX)) {
         changeXPosition = $prevOrgWidth - $prevOrgPosXElem + 30;
    }
    calcPositionX = calcPositionX + changeXPosition;
}
createjs.Tween.get(textDOM, {loop: false})
//Waiting for appropriate time
.wait(timeCount - newTimeCount).call(tweenStart)
//Scales up, and moves the appropriate X position
.to({
    scaleX: scaleX,
    scaleY: scaleY,
    x: changeXPosition
}, scaleTime)
//Moves the cell to the right, using 300 ms
.to({
    x: calcPositionX + 300
}, 300, createjs.Ease.getPowIn(1))
//Moves the cell down to the correct Y position for the result table,
    using 900 ms
.to({
    y: calcPositionY
}, 900, createjs.Ease.getPowIn(1))
//Moves the cell to the correct X position for the result table, using
    300 ms
.to({
    x: calcPositionX
}, 300, createjs.Ease.getPowIn(1))
//Scales the cell down to normal size, and removes the additional X
    values.
.to({
    scaleX: 1,
    scaleY: 1,
    x: calcScalePositionX
}, scaleTime)
//Makes the cell invisible for the user
.to({
    alpha: 0
}, 0, createjs.Ease.getPowIn(1))
//Moves the cell back to its original position, and makes it visible
    again
.to({
    alpha: 1,
    y: 0,
    x: 0
}).call(tweenComplete);
//Show the target cell text in the result table
getTextOrigin.show();
// call to function tweenComplete after animation is completed
```

Process E: Mark animated cell correctly

```
$(".original-table").find("span#animThis").parent().find(".textOrigin").
    parent().addClass("usedInRow").addClass("usedInRow_" + i);

if($(".original-table").length == 1){
    $(".original-table").find("span#animThis").parent().find(".textOrigin
        ").addClass("used");
}

var countOfDuplicates = $(thisTableId).find("span:not(#original-span):not
    (.textOrigin)").filter(function() {
    return $(this).text() === textContent;
}).length;

var duplicatesInAllTables = $(".original-table").find("span:not(.used):
    not(.textOrigin)").filter(function() {
    return $(this).text() === textContent;
}).length;
```

```
if (duplicatesInAllTables > countOfDuplicates) {
    $(".original-table").find("span#animThis").parent().find(".textOrigin
        ").addClass("duplicate");
}
```

```
 var emptyRowLength = $(".empty-table").find("td").find("span.span_"+
     emptyRowIndex).prev().filter(function() {
    return $(this).text() === thisText;
}).length;
var thisRowLength = $(thisTable).fingd("td").find("span.span_"+
    thisRowIndex).filter(function() {
    return $(this).text() === thisText;
}).length;

//If JOIN is RIGHT JOIN
if($(".alert-info-decomposer").find("b:contains(RIGHT)").length > 0){
    var numOfReps = emptyRowLength / thisRowLength;
}
//If JOIN is not a RIGHT JOIN
else {
    var numOfReps = tableRows/thisRowCount;
    if(numOfReps%1 > 0.5){
        numOfReps = numOfReps + 1 - (numOfReps%1);
    }
    else {
        numOfReps = numOfReps - (numOfReps%1);
    }
}
```

Listing N.5: Marking of cell depending on type of JOIN

```
if(typeof $(".original-table").find("span#animThis").parent().find(".
    textOrigin").attr("value") === 'undefined'){
    $(".original-table").find("span#animThis").parent().find(".textOrigin
        ").attr("value", "0")
}
var numUsed = parseInt($(".original-table").find("span#animThis").parent
    ().find(".textOrigin").attr("value"));
if(numUsed < numOfReps){
    numUsed = numUsed+1;
    $(".original-table").find("span#animThis").parent().find(".textOrigin
        ").attr("value", numUsed);
    if(numUsed == numOfReps){
        $(".original-table").find("span#animThis").parent().find(".
            textOrigin").addClass("used");
    }
}
else {
    $(".original-table").find("span#animThis").parent().find(".textOrigin
        ").addClass("used");
}
```

Listing N.6: Defining the number of animations for each cell