Towards a Validated Self-Efficacy Scale for Data Management

Wensheng Wu
University of Southern California
Los Angeles, CA, USA
wenshenw@usc.edu

ABSTRACT

We propose a self-efficacy scale for data management. The scale assesses students' perceived capabilities in mastering the breadth and depth of modern data management, as well as hands-on skills for effective management of data. Such capabilities are critical to computing and data science students. We have conducted factor analysis to validate the scale. The analysis produced a factor model with high internal consistencies. Group analyses using the factor solution and statistical testing show that (1) males and females have similar self-efficacy, except for the depth of knowledge where females showed higher confidences; and (2) CS students had much higher self-efficacy than non-CS students. To the best of our knowledge, this is the first self-efficacy scale for data management.

CCS CONCEPTS

• **Information systems** → **Data management systems**; • **Social and professional topics** → **Computing education**.

KEYWORDS

Self efficacy, data management, factor analysis, group analysis

ACM Reference Format:
Wensheng Wu. 2023. Towards a Validated Self-Efficacy Scale for Data Management. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2023), March 15–18, 2023, Toronto, ON, Canada.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3545945.3569767

1 INTRODUCTION

Self-efficacy refers to an individual's belief about his or her capabilities to produce designated levels of performance [1]. A person with a strong sense of efficacy is more likely to take initiatives to tackle a challenging task, apply greater efforts to achieve it, and remain task-oriented in the face of pressing situational demands [1, 35]. Self-efficacy beliefs of students are strongly correlated with their academic performance and achievement [31, 32, 46].

Past research has proposed self-efficacy scales for varied subjects, ranging from mathematics [32], social studies [46], programming [23, 34, 35], algorithms [13], to other computing knowledge and skills [30]. However, there has been little work on developing self-efficacy scale for data management. Data management knowledge and skills are increasingly important to computing majors in this era of big data. They are also at the core of every data science curriculum [3, 14]. A self-efficacy scale for data management would be tremendously useful to assess students' perceived capabilities on their data management knowledge and skills, uncover deficiencies, and help identify at-risk students early on, e.g., by administering the scale in the beginning or middle of the semester.

Towards this goal, we propose a self-efficacy scale for data management. The scale consists of 41 items which are formulated as general as possible so that the scale can be adapted to suit the need of other programs. The scale addresses both the breadth and depth of modern data management, covering relational, NoSQL, and big data. The depth aspect of the scale is targeted at the graduate-level database knowledge. The scale emphasizes hands-on skills necessary for the effective management of data, including data in the cloud. The scale also includes four items on self-regulation, adapted from the self-efficacy scale for computer programming developed by Ramalingam and Wiedenbeck [35].

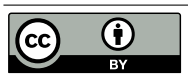
We have administered the scale to the *diverse* students in the graduate database courses of our applied data science program [42, 43] and conducted factor analysis on students' responses. The analysis produced a six-factor model for the scale: two factors on the breadth, two on hands-on skills, one on the depth, and one on self-regulation. All the factors have high internal consistencies, with alpha coefficients ranging from .89 to .94.

We have utilized the factor solution to perform group analyses where students were grouped based on their gender and undergraduate major. The analyses did not find statistically significant differences between male and female students in all aspects of the scale, except for the depth of data management knowledge where females showed higher self-efficacy. However, CS students (students with an undergraduate CS major) showed much higher self-efficacy than non-CS students in all aspects of data management and skills.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 describes the design of the scale. Sections 4 and 5 present the details of factor analysis and group analyses respectively. Section 6 discusses limitations of the study. Finally, Section 7 concludes the paper.

2 RELATED WORK

Self-efficacy scales and applications: Pajares and Graham [32] developed a mathematics self-efficacy scale and showed that it can be used to predict students' mathematics performance. Their study found no gender differences in self-efficacy beliefs. Zimmerman et. al. [46] used students' self-efficacy and personal goals at the beginning of the semester to predict students' final course grades in social studies. The study found that students' self-efficacy on self-regulated learning influenced their goal settings and final academic achievements. Ramalingam et. al. [34] showed that self-efficacy for programming was influenced by students' previous programming experience and increased as students progressed through an introductory programming course. Blaney and Stout [5] examined the



SIGCSE 2023, March 15–18, 2023, Toronto, ON, Canada

ACM ISBN 978-1-4503-9431-4/23/03.
https://doi.org/10.1145/3545945.3569767

self-efficacy of first-generation college women and found out their experiences in introductory computing courses had strong impact on their self-efficacy and sense of belongings.

Ramalingam and Wiedenbeck [35] developed a self-efficacy scale for computer programming. The scale consists of 32 items covering simple and complex programming tasks, independence and persistence in completing projects, and self-regulation. Similar to [35], our scale distinguishes basic data management tasks from more advanced tasks. We also reuse the items in [35] on students' self-regulation, after properly rephrasing them for data management tasks (see Section 3).

Computing attitude scales: Attitude toward computing is another critical factor in predicting class performance and career outcomes of computing students [8]. As such, there have been several efforts in developing scales for measuring students' computing attitudes [6, 16, 17]. In particular, Dorn and Elliott Tew [16] developed CAS, a scale that contains 26 questions to measure students' attitudes and beliefs about computer science. The scale is divided into 5 factors covering students' personal interests in computer science, real-world connections, problem-solving strategies, fixed mindsets, and knowledge transfer. The items on fixed mindset also measure students' self-regulation to some degree.

Bockmon et. al. [6] extended the CAS scale with 14 new questions to address gender bias, gender equity, and utility of CS in students' lives and careers. The study found a positive correlation between gender bias and fixed mindset in problem solving among the students in introductory computing courses.

Database competency and curriculum: According to the data science skills competency model [21], a data scientist must demonstrate SQL skills for querying databases and joining tables, the ability to work with data from multiple data sources such as SQL and NoSQL databases, and the ability to handle data of different formats such as data in relational databases, CSV, and JSON files.

A recent report on the "Curriculum Guidelines for Undergraduate Programs in Data Science" [14] stated that a data science undergraduate major must be able to apply the knowledge of data query languages to relational databases and emerging NoSQL data systems, access data from less structured systems through web services, and transform data into structured forms required for exploration, visualization, and analysis.

3 DESIGN OF THE SELF-EFFICACY SCALE

There are several desiderata in designing the scale.

- *Comprehensive*: The scale should cover students' self-efficacy in both fundamental and advanced data management knowledge, as well as important hands-on skills for the effective management of data.
- *Focused*: The scale should focus on the data management knowledge and skills that are frequently demanded by the IT industry and job market [45], and commonly covered in the database curriculum [3, 7, 12, 15, 25, 33, 38, 44].
- *General*: The items in the scale should be written as general as possible so that the scale can be easily adapted for other database courses or programs. For example, items may use general terms such as relational database and cloud platform instead of directly referring to specific system (e.g., MySQL) and platform (e.g., Amazon Web Service).

Based on the above desiderata, Table 1 shows the design of the scale. It consists of 41 items that assess students' self-efficacy in modeling and querying relational and NoSQL data, understanding the working of database and big data processing systems, interacting with computing systems, and writing codes to solve data management problems. The scale also contains four items (items 38–41) on students' self-regulation adapted from [35].

Breadth: The scale measures students' confidence in managing a variety of data, including relational, XML, and JSON data; working with different types of data management systems, such as relational databases (e.g., MySQL) and NoSQL databases (e.g., MongoDB); and mastering common query languages, such as SQL and XPath.

Depth: The scale also measures students' belief in their understandings of the internals of database and big data processing systems. In particular, processing of an SQL query by a relational database system, searching and updating of a B+-tree index, design and cost evaluation of different join algorithms, shuffling process in Hadoop MapReduce, reading and writing files in Hadoop HDFS, and parallel execution of transformations and actions in Spark. Hadoop [41] and Spark [9] are two software widely used for parallel data processing and analytics.

Note that the depth aspect of the scale may be removed if the scale is targeted at undergraduate students.

Hands-on skills: Hands-on skills on computing systems and software are critical to effective data management. In particular, *cloud computing* has become indispensable to data management. However, increasingly the students taking the database courses might not have strong CS backgrounds and might lack these important skills. For example, typically only about one-third of the students in our graduate database course have an undergraduate major in CS. Non-CS students may be required to take pre-requisite courses, e.g., on programming and computational thinking.

The scale contains items to measure students' perceived capability in setting up virtual machines on a cloud computing platform (e.g., Amazon AWS), interacting with virtual machines (e.g., through Linux command lines), installing and configuring database software and parallel data processing systems on the machines. The scale also measures students' capability in writing codes (e.g., in Python and Java) for data wrangling and data analytics, working with a cloud database through its API, as well as self-sufficiency in troubleshooting software and programming errors.

4 FACTOR ANALYSIS

4.1 Data Collection

We created a survey with questions on students' demographics and academic backgrounds, such as gender and major, and all 41 items in the self-efficacy scale described in Section 3. Students who have taken our graduate-level database course in the past three semesters, that is, from Spring 21 to Spring 22, were asked to fill out the survey. The survey was voluntary.

For each item in the self-efficacy scale, students were asked to indicate their confidence in the 5-point Likert scale, from absolutely confident, mostly confident, neutral, mostly not confident, to not at all confident. We received responses from 111 students: 85 of

them took the course in Spring 22 (the class size was 165); 21 in Fall 21; and 5 in Spring 21. For each response, we converted student's answers to numerical scores, with 5 for absolutely confident and 1 for not at all confident.

Among the 111 students, 66 (about 60%) were male, and 45 were female. We categorized students into CS and non-CS students, based on their undergraduate majors. We define CS students broadly to include students who majored in computer engineering, software engineering, data science, information technology, and informatics. There were 38 (34%) CS students. The rest of the students came from varied backgrounds, including mathematics, statistics, economics, biology, chemistry, and environmental science.

4.2 Exploratory Factor Analysis

We conducted exploratory factor analysis on students' responses. The goal is to discover a small set of latent factors that underlie a relatively large number of items in the scale, thereby revealing the inherent structure of the scale. Each (common) factor corresponds to a group of closely related items in the scale, and there are relatively smaller correlations among the items from different factors.

Formally, the response to each item i in the scale may be represented as a random variable X_i, where $1 \leq i \leq p$ and p is the total number of items. For example, $p = 41$ in the proposed scale. As Equation 1 shows, the factor model assumes that each items X_i is linearly dependent on a set of m unobservable random variables, F_1, ..., F_m, called *common* factors, and an additional variable ϵ_i , called *unique* factor, which captures variation (e.g., due to unique characteristics of the item or measurement error) specific to the item [22]. The common variance of an item, also called the communality of the item, is the part of its variance contributed by the common factors. Note that μ_i in Equation 1 is the mean of X_i, and ℓ_{ij} is the loading of the i-th variable on the j-th factor.

$$\begin{aligned} X_1 &= \mu_1 + \ell_{11}F_1 + \ell_{12}F_2 + \cdots + \ell_{lm}F_m + \epsilon_1 \\ X_2 &= \mu_2 + \ell_{21}F_1 + \ell_{22}F_2 + \cdots + \ell_{2m}F_m + \epsilon_2 \\ &\cdots \\ X_p &= \mu_p + \ell_{p1}F_1 + \ell_{p2}F_2 + \cdots + \ell_{pm}F_m + \epsilon_p \end{aligned} \tag{1}$$

The matrix of loadings, with its i-th row containing the loadings of X_i over the m factors, is called *pattern matrix* (e.g., see columns I–VI of Table 1). A key step in the factor analysis is to estimate the loadings of items on the common factors and item-specific variances from a sample data for X_i's. For example, in our analysis, the sample data may be stored in a *data matrix* where each row represents a student's responses to all scale items.

We applied the *principal axis* method with an *oblimin* rotation to find the above estimates, using the *psych* package in R [37]. The principal axis method finds the estimates by iteratively performing an eigen-decomposition of the *reduced* correlation matrix of items where the diagonal entries of the matrix are replaced by the estimates on the communalities of items based on the loadings found in the previous iteration. An initial estimate on the communality of item X_i may be given by the square of the multiple correlation coefficient between X_i and the other $p - 1$ items [22]. Since the original correlation matrix of items has 6 eigenvalues greater than 1, we set the number of factors to be extracted to 6 [39].

Factor rotation rotates the axes of the factor space so that factor loadings become easier to interpret [27]. For example, it is easy to decide which factor to assign an item to if the item loads highly on a single factor, but has relatively smaller loadings on the remaining factors. The oblimin rotation is a type of oblique methods for factor rotation. After the rotation, the axes of the factor space may not be orthogonal to each other any more. This is desirable since we expect different factors in the same scale to be correlated.

4.3 Results and Discussions

The first two columns of Table 1 (except for the last row) show the number and description of items in the six factors. As described earlier, the scale consists of 41 items, numbered from 1 to 41.

- Factor 1 consists of 12 items: items 1–5, 8–11, 15, 17, and 18. It is labeled as "system and software skills". Factor 4 "administration skills" consists of two items: items 6 and 7. Factor 1, together with factor 4, represent the *hands-on skills*.
- Factor 2 "data modeling and query language" contains 8 items, and factor 6 "NoSQL data management" contains 4 items. These two factors focus on the *breadth* aspect.
- Factor 3 "query execution and big data processing" consists of 11 items: items 26–27 and 29–37. This factor addresses the *depth* aspect of the scale.
- Finally, factor 5 contains the four items on students' *self-regulation*.

Columns I–VI of the table show the loading scores of items on the factors (i.e., the pattern matrix), where the scores in bold indicate the assignment of items to factors. Note that scores whose absolute values are smaller than .1 are not shown for clarity. All items were assigned to the factors on which they have the largest loadings, except for items 13, 14, and 28. Items 13 and 14 have slightly larger loadings on factor 3, but were assigned to factor 2, to emphasize their focus on data modeling. Item 28 has almost the same loading on both factors 5 and 6, and was assigned to factor 6 since it addresses the NoSQL data management skills.

Column h^2 shows the communality of scale items. We can see that the communality of items ranges from .42 (item 11) to .88 (item 20). The average communality of items is .73. Note that the communality of an item was computed from the loading scores of the item on the factors *before* the oblique rotation.

For each factor, the table also shows its Cronbach's alpha score in parenthesis. The Cronbach's alpha [4] measures the *internal consistency* of a factor, with a normal range between 0 and 1. A factor with a high alpha score indicates that the items in the factor are closely related to each other. The Cronbach's alphas of factors in the scale range from .89 (factors 5 and 6) to .96 (factor 3).

The last row of the table shows the proportion of variances of items explained by the common factors. We can see that factor 3 explains the largest proportion of the variances (19%), followed by factor 2 (16%) and factor 1 (15%). Factors 4, 5, and 6 explain 10%, 7%, and 6% of the variances respectively. These 6 factors together explain 73% of the total variance.

The last two columns of the table show the mean (i.e., estimates of μ_i's in Equation 1) and standard deviation of raw scores for each item, and the mean of raw scores of *all* items in each factor (shown in bold and italic font). From these numbers, we can observe that:

Item	Description of items in each factor	I	II	III	IV	V	VI	h^2	Mean	SD
	Factor 1: System and software skills (alpha = .94)								***4.33***	
1	Create virtual machine instances (running Linux) in the cloud	**0.47**	0.25	-0.13	0.33	0.11		0.71	4.38	0.76
2	Connect to virtual machine instances via command lines	**0.53**		-0.22	0.32	0.32		0.74	4.41	0.83
3	Manage (locate, copy, move, etc.) files in laptops	**0.55**	0.23		0.13	0.11		0.65	4.64	0.63
4	Execute Linux commands/programs (e.g., rm, wget)	**0.49**	0.12		0.33		0.10	0.64	4.38	0.71
5	Edit files in Linux system (e.g., using nano or vi)	**0.44**			0.41	0.28		0.65	4.43	0.82
8	Troubleshoot software problems and errors	**0.51**		0.11	0.41	-0.10		0.68	4	0.89
9	Utilize online resources (e.g., stackoverflow) in troubleshooting	**0.78**					-0.11	0.65	4.52	0.72
10	Write Python codes for homework assignments	**0.73**						0.72	4.59	0.69
11	Interpret and modify Java programs	**0.34**		0.14	0.30	-0.18	0.16	0.42	3.52	1.23
15	Structure JSON data in a NoSQL database (e.g., Firebase)	**0.50**	0.37		-0.14		0.27	0.77	4.34	0.8
17	Convert CSV files into JSON/XML format	**0.59**		0.29			0.10	0.61	4.43	0.82
18	Retrieve data from a cloud database through REST API	**0.57**					0.44	0.74	4.31	0.82
	Factor 2: Data modeling and query language (alpha = .94)								***4.38***	
12	Design ER (entity-relationship) model for an application	0.33	**0.36**	0.34		-0.15		0.58	4.11	0.84
13	Design schema for a relational database (e.g., MySQL)	0.34	**0.35**	*0.47*			-0.19	0.75	4.22	0.86
14	Create tables with foreign key constraints	0.28	**0.45**	*0.50*		-0.11	-0.14	0.77	4.32	0.82
19	Write SQL queries for projection and filtering of data		**0.79**			0.14		0.82	4.54	0.6
20	Write SQL queries for data grouping and aggregation	0.12	**0.84**	-0.17		0.12		0.88	4.55	0.61
21	Write SQL queries that involve subqueries		**0.86**					0.83	4.48	0.7
22	Write SQL queries to join tables		**0.84**				0.11	0.80	4.52	0.69
23	Answer queries using views	-0.23	**0.74**	0.17			0.20	0.75	4.32	0.76
	Factor 3: Query execution and big data processing (alpha = .96)								***3.97***	
26	Design a Hadoop MapReduce program for data processing			**0.56**	0.26		0.35	0.82	3.9	1
27	Write a Spark script (using data frame API) for data analysis			**0.42**	0.41		0.29	0.75	4.03	0.88
29	Explain how a database system processes an SQL query		0.32	**0.52**	0.23	-0.11		0.65	3.96	0.88
30	Describe searching and updating of a B+-tree index	0.16		**0.69**		0.15		0.73	4.09	0.93
31	Design nested-loop, sorting, and hashing-based join algorithms			**0.82**	0.15	0.14		0.79	3.96	0.96
32	Evaluate the costs of different join algorithms			**0.75**		0.23		0.73	4.02	0.94
33	Explain the architecture of Hadoop			**0.76**	-0.11	0.12	0.15	0.77	4.04	0.87
34	Describe the process of reading & writing files in Hadoop	0.22		**0.59**		0.15	0.17	0.76	4.08	0.86
35	Describe the shuffling process in Hadoop			**0.81**			0.17	0.78	3.98	0.94
36	Explain parallel execution of transformations and actions in Spark	-0.18		**0.65**	0.32	-0.13	0.13	0.71	3.77	1
37	Illustrate the client-server computing model			**0.64**	0.17	0.15	0.18	0.72	3.86	0.95
	Factor 4: Administration skills (alpha = .92)								***4.19***	
6	Install database software (e.g., MySQL, MongoDB)	0.12	0.23		**0.77**			0.85	4.3	0.89
7	Install and configure big data software (Hadoop&Spark)				**0.85**	0.14		0.87	4.07	1.02
	Factor 5: Self-regulation (alpha = .89)								***4.39***	
38	Concentrate on the assignments, despite many distractions		0.19	0.14		**0.71**		0.78	4.38	0.71
39	Motivate myself, even if the problem is of no interest to me	-0.13	0.15	0.16	0.21	**0.56**		0.60	4.32	0.75
40	Manage my time efficiently when there is a pressing deadline		0.29	0.24		**0.53**	-0.29	0.70	4.42	0.68
41	Create a suitable strategy for the assignment in a timely fashion	0.13	0.39	0.14		**0.48**	-0.14	0.73	4.44	0.72
	Factor 6: NoSQL data management (alpha = .89)								***4.17***	
16	Use XML format to represent data	0.30	0.27	0.13	-0.17	0.19	**0.32**	0.57	4.22	0.78
24	Perform ad-hoc queries on JSON documents in MongoDB		0.19	0.26	0.17		**0.56**	0.81	4.14	0.87
25	Perform data aggregation (using aggregate pipeline) in MongoDB		0.16	0.26	0.26		**0.50**	0.80	4.12	0.83
28	Retrieve data from XML files using XPath	0.22	0.11	0.25		*0.38*	***0.37***	*0.76*	4.19	0.83
	Proportion of variance explained	***15%***	***16%***	***19%***	***10%***	***7%***	***6%***			

Table 1: Scale items, factor loadings, item statistics, Cronbach's alpha of factors, and variances explained by the factors

- Students tend to have high confidences in self-regulation (mean = 4.39), data modeling and query language (mean = 4.38), and system and software skills (mean = 4.33).
- Students were less confident on administration skills (mean = 4.19) and NoSQL data management (mean = 4.17).
- Students were least confident on query execution and big data processing (mean = 3.97). This is perhaps not surprising, given that this factor addresses the depth aspect and hence the more challenging part of data management.

Furthermore, item 11 "interpret and modify Java program" has the lowest mean (3.52) among all items. On the other hand, item 10 on writing Python codes for homework assignments has a much higher mean (4.59). This shows that many data science students might be more confident in their Python skills than other programming languages, such as Java.

We also note that (not shown in the table) the item scores are very skewed, with the skewness ranging from -2.37 (item 3: managing files) to -.48 (item 29: understanding SQL query processing).

	F1	F2	F3	F4	F5	F6
F1	1					
F2	**.53**	1				
F3	.34	**.45**	1			
F4	.32	.36	.39	1		
F5	.34	**.46**	.28	.18	1	
F6	.21	.23	.41	.34	.1	1

Table 2: Correlation matrix of rotated factors

Table 2 shows the correlation matrix of factors after the oblimin rotation. Suppose the oblimin method uses a rotation matrix Q to perform the factor rotation. Then the correlation matrix C of factors is given by $Q'Q$, where Q' is the transpose of Q [2]. We can see that all factors are correlated to some degree, with the correlation coefficients ranging from .1 (between factors 5 and 6) to .53 (between factors 1 and 2). In particular, factor 2 is modestly correlated with three factors: 1, 3, and 5. This can also be seen from the factor loadings in Table 1. For example, items 12, 13, and 14 of factor 2 load highly also on factors 1 and 3; and items 38–41 of factors 5 also load significantly on factor 2. Note that factor 2 "data modeling and query language" addresses the most fundamental knowledge and skills on data management. It would be interesting to examine if high confidence in the fundamentals boosts students' confidences in other aspects of data management.

5 GROUP ANALYSES

The group analyses are guided by the following research questions.

- **RQ1:** Does the self-efficacy on data management differ between male and female students?
- **RQ2:** Does the self-efficacy on data management differ between CS and non-CS students?

The answers to these questions are based on student's scores on the six factors in the self-efficacy scale. The null hypothesis for both questions is that factor scores for the students from different groups (male vs. female and CS vs. non-CS) have statistically similar distributions. That is, high and low scores are approximately evenly distributed among the different groups. The alternative hypothesis is that scores of different groups have different distributions.

5.1 Computing Factor Scores

There are a number of ways of estimating factor scores (i.e., F_j's in Equation 1) for a student based on estimated factor loadings of items (ℓ_{ij}'s) and the student's raw scores on the items in the scale (X_i's). In our analysis, we used the *tenBerge* estimation method [20, 36, 40] which ensures that the correlation of factors computed from the estimated factor scores matches the factor correlation given by the model (i.e., the matrix C described in Section 4.3).

The factor scores computed by the tenBerge method are standardized, with a zero mean and standard deviation of 1. The *overall* columns of Table 3 show the statistics of scores for each factor, which include min, max, median, and skew. We can see that scores of factors 1, 4, and 6 are highly skewed (skewness < -1); and scores for the rest of the factors are modestly skewed (-1 < skewness < -.5). Scores for all factors are negatively or left skewed, which means that, while the majority of the scores are above zero, there are a large number of low negative scores (long left tail). It is not surprising that the scores for none of the factors are normally distributed. The non-normality of the factor scores was also confirmed using the Shapiro-Wilk test.

5.2 Performing Statistical Tests

To answer research question RQ1, we divided the students into two groups. Group 1 consists of n_1 male students and group 2 n_2 female students, where n_1 = 66 and n_2 = 45. We denote the total sample size as N where $N = n_1 + n_2$.

We then performed a *two-tailed* Mann-Whitney U test [11] for each factor to determine if factor scores of students from different groups are statistically different. The Mann-Whitney test considered all pairs of students (x, y)'s where x is a student from group 1 and y is a student from group 2. Note that there are $n_1 * n_2 = 2970$ such pairs in this gender-based test. The test then computed a statistic U which is the number of pairs where the student x has a score not less than the score of student y.

When the null hypothesis is true and the sample size is sufficiently large ($N > 20$), the U values are approximately normally distributed with mean = $n_1 * n_2/2$ [11]. Accordingly, the U statistic may be converted into a z-score and its corresponding p-value may be obtained from a standard normal distribution table.

Similarly, for RQ2, we divided students into CS and non-CS groups (n_1 = 38 and n_2 = 73) and performed the same tests.

The results for the gender-based and major-based tests are shown in the *male vs. female* and *CS vs. non-CS* columns of Table 3 respectively. Note that for each factor, the table also shows the median score of students in different groups. For example, the *m(male)* column shows the medium score of male students.

In addition, we have computed effect size for each test to quantify the difference between the groups. We considered two types of effect size: Cohen's r [18] and Glass's rank biserial correlation coefficient, denoted as rg [19, 26]. Both are widely used in research [18, 24]. The effect sizes for the tests are shown in the *ES: r* and *ES: rg* columns

Factor	Overall				Male vs. female						CS vs. non-CS					
	min	max	median	skew	m(male)	m(female)	U	p-value	ES: r	ES: rg	m(CS)	m(non-CS)	U	p-value	ES: r	ES: rg
F1	-4.29	1.61	0.32	-1.77	0.26	0.44	1361	0.46	-0.071	-0.084	0.41	0.21	1751	**0.02**	0.215	0.262
F2	-3.85	1.17	0.55	-0.92	0.50	0.69	1403	0.62	-0.047	-0.055	0.76	0.07	1730	**0.03**	0.202	0.247
F3	-3.28	1.34	0.14	-0.74	0.01	0.29	1195	***0.08***	-0.165	-0.195	0.48	0.03	1729	**0.03**	0.202	0.247
F4	-3.86	1.37	0.16	-1.47	0.15	0.17	1530	0.79	0.026	0.030	0.39	0.09	1660	***0.09***	0.161	0.197
F5	-2.94	2.04	0.26	-0.69	0.10	0.52	1331	0.36	-0.088	-0.104	0.29	0.26	1315	0.66	-0.043	-0.052
F6	-4.13	2.25	0.1	-1.08	0.11	-0.04	1630	0.39	0.083	0.098	0.54	-0.04	1668	***0.08***	0.166	0.203

Table 3: Mann-Whitney U test results for the group analyses

of the table. Note that an r value of about .1 is considered to be a small effect, .3 medium, and .5 a large effect [10, 28].

5.3 Analyzing the Results

Answering RQ1: We can observe from the table that all gender-based tests have large p values, ranging from .36 to .79, except for factor 3 which has a p value of .08. Note that if the test for a factor has a high p-value (e.g., factors 2 and 4), its corresponding U (e.g., 1403 and 1530) will be very close to the expected mean (2970/2 = 1485) when the null hypothesis is true.

If the significance level is set to .05, then the null hypothesis can not be rejected for any of the factors. In other words, we could not find significant difference between males and females. However, if it were a one-tailed test, the null hypothesis would be rejected for the factor 3. In other words, the test would find that females tend to have higher confidences in the depth aspect of data management than males. Note that negative effect sizes indicate that on the average, students in the group 2 (females in this case) are scored higher than students in group 1 (males).

In sum, the analysis found that male and female students had similar self-efficacy on the breadth of data management, hands-on skills, and self-regulation. But female students had significantly higher self-efficacy on their advanced data management skills.

Answering RQ2: The last 6 columns of Table 3 show the results of major-based tests. We can see that all tests had very low p-values, ranging from .02 to .09, except for factor 5 where the p-value is .66. Recall that factor 5 is about students' self-regulation. So this indicates that the differences between CS and non-CS students on their self-regulations are not statistically significant.

With a significance level of .05, the null hypothesis will be rejected for factors 1, 2 and 3. In other words, the tests find that CS students tend to have higher self-efficacy than non-CS students on system and software skills (factor 1), data modeling and query language (factor 2), and query execution and big data processing (factor 3). The effect sizes of these three tests are close to the medium (.3).

Furthermore, the null hypothesis for factor 4 (p = .09) and factor 6 (p = .08) would be rejected for one-tailed test. In other words, CS students were much more confident on their administration skills (factor 4) and NoSQL data management skills (factor 6) than non-CS students. This can also be seen by comparing their median scores. For example, the medium score of factor 6 is .54 for CS, compared to -.04 for non-CS students.

In sum, the analysis found that CS students had much higher self-efficacy on *all* aspects of data management knowledge and skills than non-CS students. However, they had similar beliefs in their self-regulation abilities.

6 LIMITATIONS

First of all, about 25% of the students who participated in the survey have taken the database course at least one semester ago. So additional courses students have taken might have impacted their self-efficacy. Students typically take machine learning, data mining, and data visualization after taking the database course. Furthermore, we did not compare the efficacy of students before and after they have taken the database course. Such a comparative analysis would be useful to assess the effects of the course on students' self-efficacy [16]. We plan to conduct this study in the coming semesters. We do expect student's self-efficacy to greatly improve after taking the course, as suggested by the theory and past studies [35].

Secondly, since the survey was voluntary, about 60% of the students taking the course in Spring 22 and only a small number of students in earlier semesters responded. It is likely that many students in the past semesters may have graduated and thus the students who have responded to the survey might not be good representatives of the students in these semesters. However, note that the minimum sample size for the factor analysis is largely dependent on the communality of items and the structure of factor model [29]. In particular, study [29] shows that when the ratio of the number of variables to the number of factors is 20:3 (similar to our 41:6), factor solutions will converge with a sample size of 100 or less if the communality of items is consistently high (above .6), and most of the factors contain a large number of items.

7 CONCLUSION

We have developed a self-efficacy scale to measure students' perceived capabilities in modeling and querying of relational and NoSQL data, understanding the internals of database and big data systems, and working with computing systems and software for data management. We have validated the scale through exploratory factor analysis and demonstrated its utility in group analyses.

Besides further evaluation, we are pursuing several directions to expand our study: (1) conduct pre-post tests to examine the shift in students' self-efficacy from the beginning to the end of the semester; (2) examine the correlation between student's self-efficacy and their performance in database courses; (3) investigate the underlying factors that led females to have higher self-efficacy than males on the depth of data management knowledge.

REFERENCES

[1] Albert Bandura and Sebastian Wessels. 1994. *Self-efficacy (in Encyclopedia of Human Behavior)*. Vol. 4. Academic Press.

[2] Coen Bernaards, Robert Jennrich, and Maintainer Paul Gilbert. 2022. Package GPArotation. (2022).

[3] Ismail Bile Hassan, Thanaa Ghanem, David Jacobson, Simon Jin, Katherine Johnson, Dalia Sulieman, and Wei Wei. 2021. Data science curriculum design: A case study. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 529–534.

[4] J Martin Bland and Douglas G Altman. 1997. Statistics notes: Cronbach's alpha. *Bmj* 314, 7080 (1997), 572.

[5] Jennifer M Blaney and Jane G Stout. 2017. Examining the relationship between introductory computing course experiences, self-efficacy, and belonging among first-generation college women. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education*. 69–74.

[6] Ryan Bockmon, Stephen Cooper, Jonathan Gratch, and Mohsen Dorodchi. 2020. Validating a CS attitudes instrument. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. 899–904.

[7] Richard A Brown. 2009. Hadoop at home: large-scale computing at a small college. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*. 106–110.

[8] Tor Busch. 1995. Gender differences in self-efficacy and attitudes toward computers. *Journal of educational computing research* 12, 2 (1995), 147–158.

[9] Bill Chambers and Matei Zaharia. 2018. *Spark: The definitive guide: Big data processing made simple.* " O'Reilly Media, Inc.".

[10] Jacob Cohen. 2013. *Statistical power analysis for the behavioral sciences*. Routledge.

[11] Gregory W Corder and Dale I Foreman. 2014. *Nonparametric statistics: A step-by-step approach*. John Wiley & Sons.

[12] Sarah Dahlby Albright, Titus H Klinge, and Samuel A Rebelsky. 2018. A functional approach to data science in CS1. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 1035–1040.

[13] Holger Danielsiek, Laura Toma, and Jan Vahrenhold. 2018. An instrument to assess self-efficacy in introductory algorithms courses. *ACM Inroads* 9, 1 (2018), 56–65.

[14] Richard D De Veaux, Mahesh Agarwal, Maia Averett, Benjamin S Baumer, Andrew Bray, Thomas C Bressoud, Lance Bryant, Lei Z Cheng, Amanda Francis, Robert Gould, et al. 2017. Curriculum guidelines for undergraduate programs in data science. *Annu Rev Stat Appl* 4 (2017), 15–30.

[15] Debzani Deb, Muztaba Fuad, and Keith Irwin. 2019. A module-based approach to teaching big data and cloud computing topics at cs undergraduate level. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. 2–8.

[16] Brian Dorn and Allison Elliott Tew. 2015. Empirical validation and application of the computing attitudes survey. *Computer Science Education* 25, 1 (2015), 1–36.

[17] Allison Elliott Tew, Brian Dorn, and Oliver Schneider. 2012. Toward a validated computing attitudes survey. In *Proceedings of the ninth annual international conference on International computing education research*. 135–142.

[18] Catherine O Fritz, Peter E Morris, and Jennifer J Richler. 2012. Effect size estimates: current use, calculations, and interpretation. *Journal of experimental psychology: General* 141, 1 (2012), 2.

[19] Gene V Glass. 1965. A ranking variable analogue of biserial correlation: Implications for short-cut item analysis. *Journal of Educational Measurement* 2, 1 (1965), 91–95.

[20] James W Grice. 2001. Computing and evaluating factor scores. *Psychological methods* 6, 4 (2001), 430.

[21] IBM. 2020. The Data Science Skills Competency Model. https://www.ibm.com/downloads/cas/7109RLQM.

[22] Richard Arnold Johnson, Dean W Wichern, et al. 2014. *Applied multivariate statistical analysis*. Vol. 6. Pearson London, UK:.

[23] Maria Kallia and Sue Sentance. 2019. Learning to use functions: The relationship between misconceptions and self-efficacy. In *Proceedings of the 50th ACM technical symposium on computer science education*. 752–758.

[24] Dave S Kerby. 2014. The simple difference formula: An approach to teaching nonparametric correlation. *Comprehensive Psychology* 3 (2014), 11–IT.

[25] Suneuy Kim. 2020. Seamless Integration of NoSQL class into the Database Curriculum. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*. 314–320.

[26] Bruce M King, Patrick J Rosopa, and Edward W Minium. 2018. *Statistical reasoning in the behavioral sciences*. John Wiley & Sons.

[27] James M Lattin, J Douglas Carroll, and Paul E Green. 2003. *Analyzing multivariate data*. Vol. 1. Thomson Brooks/Cole Pacific Grove, CA.

[28] W Lenhard and A Lenhard. 2016. Computation of effect sizes. Psychometrica. http://www.psychometrica.de/effect_size.html.

[29] Robert C MacCallum, Keith F Widaman, Shaobo Zhang, and Sehee Hong. 1999. Sample size in factor analysis. *Psychological methods* 4, 1 (1999), 84.

[30] Christine A Murphy, Delphine Coover, and Steven V Owen. 1989. Development and validation of the computer self-efficacy scale. *Educational and Psychological measurement* 49, 4 (1989), 893–899.

[31] Frank Pajares. 1997. Current directions in self-efficacy research. *Advances in motivation and achievement* 10, 149 (1997), 1–49.

[32] Frank Pajares and Laura Graham. 1999. Self-efficacy, motivation constructs, and mathematics performance of entering middle school students. *Contemporary educational psychology* 24, 2 (1999), 124–139.

[33] Ariel S Rabkin, Charles Reiss, Randy Katz, and David Patterson. 2012. Experiences teaching MapReduce in the cloud. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*. 601–606.

[34] Vennila Ramalingam, Deborah LaBelle, and Susan Wiedenbeck. 2004. Self-efficacy and mental models in learning to program. In *Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education*. 171–175.

[35] Vennila Ramalingam and Susan Wiedenbeck. 1998. Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy. *Journal of Educational Computing Research* 19, 4 (1998), 367–381.

[36] RDocumentation. 2022. factor.scores: Various ways to estimate factor scores for the factor analysis model.

[37] William Revelle. 2022. How To: Use the psych package for Factor Analysis and data reduction.

[38] Mariam Salloum, Daniel Jeske, Wenxiu Ma, Vagelis Papalexakis, Christian Shelton, Vassilis Tsotras, and Shuheng Zhou. 2021. Developing an interdisciplinary data science program. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 509–515.

[39] Barbara G Tabachnick, Linda S Fidell, and Jodie B Ullman. 2007. *Using multivariate statistics*. Vol. 5. pearson Boston, MA.

[40] Jos MF Ten Berge, Wim P Krijnen, Tom Wansbeek, and Alexander Shapiro. 1999. Some new results on correlation-preserving factor scores prediction methods. *Linear algebra and its applications* 289, 1-3 (1999), 311–318.

[41] Tom White. 2012. *Hadoop: The definitive guide.* " O'Reilly Media, Inc.".

[42] Wensheng Wu. 2021. CS vs non-CS: Analyzing Online Social Behaviors of Data Science Students with Diverse Academic Backgrounds. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 2*. 643–643.

[43] Wensheng Wu. 2021. One Size Doesn't Fit All: Diversifying Data Science Course Projects by Student Background and Interests. In *Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1*. 67–73.

[44] Wensheng Wu. 2021. SQL2X: Learning SQL, NoSQL, and MapReduce via Translation. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 590–596.

[45] Wensheng Wu. 2022. Investigating Internship Experiences of Data Science Students for Curriculum Enhancement. In *Proceedings of the 27th ACM Conference on on Innovation and Technology in Computer Science Education Vol. 1*. 505–511.

[46] Barry J Zimmerman, Albert Bandura, and Manuel Martinez-Pons. 1992. Self-motivation for academic attainment: The role of self-efficacy beliefs and personal goal setting. *American educational research journal* 29, 3 (1992), 663–676.