Complementing Course Contents Between IT/CS: A Case Study on Database Courses

Jai W. Kang
Information Sciences and Technologies
Rochester Institute of Technology
Rochester, NY USA
jai.kang@rit.edu

Qi Yu
Information Sciences and Technologies
Rochester Institute of Technology
Rochester, NY USA
qi.yu@rit.edu

Edward P. Holden
Information Sciences and Technologies
Rochester Institute of Technology
Rochester, NY USA
edward.holden@rit.edu

Xumin Liu
Computer Science
Rochester Institute of Technology
Rochester, NY USA
xmlics@rit.edu

ABSTRACT

Information Technology (IT) and Computer Science (CS) are two well regarded computing programs that produce hundreds of thousands of graduates in each year to meet the diverse needs in the IT industry. Despite being treated as two distinct disciplines, IT and CS do share commonalities as with other computing disciplines (e.g., CE, SE, and IS). In this paper, we provide a detailed analysis on the commonalities and differences between IT and CS along with their respective teaching methodologies, aiming to identify opportunities to complement these two computing disciplines so that they can benefit from each other's unique advantages to better train future computing professionals. A case study on the database courses from both IT and CS helps reveal deeper insights on both the common and different aspects of these two disciplines. These insights are then leveraged to reach important recommendations for future computing curriculum development.

CCS Concepts: • Computing education • Computing education programs • Information technology education

Keywords: Information technology; computer science; course content; curriculum; teaching methodology; database.

ACM Reference format:

Jai W. Kang, Qi Yu, Edward P. Holden and Xumin Liu. 2019. Complementing Course Contents Between IT/CS: A Case Study on Database Courses. In *Proceedings of ACM Special Interest Group in IT Education Conference (SIGITE'19), October 3-5, 2019, Tacoma, Washington, USA.* ACM, New York, NY, *USA, 6 pages.* https://doi.org/10.1145/3349266.3351414.

SIGITE '19, October 3–5, 2019, Tacoma, WA, USA

ACM ISBN 978-1-4503-6921-3/19/10…$15.00
https://doi.org/10.1145/3349266.3351414

1. INTRODUCTION

Information Technology (IT) educators develop courses for baccalaureate programs to meet the IT curriculum guidelines, which were first published in 2008 (IT2008) [9] and revised in 2017 (IT2017) [10] by ACM/IEEE Computer Society. We maintain the course materials as much, promptly and best as we can. Then one can ask questions such as "are the sources we utilize to update the course materials exhaustive?", "are there any other sources available than the IT curriculum guidelines?", and "if there are other sources, how to use them to benefit the IT curriculum?". This paper attempts to answer these research questions.

The IT discipline has become a separate baccalaureate program over the past few decades. For the first time, the Computing Curricula 2005 [3] included IT as a separate disciple in addition to Computer Engineering (CE), Computer Science (CS), Software Engineering (SE) and Information System (IS). This Computing Curricular document provides commonalities and differences among these five computing disciplines by characterizing graphically what students typically do after graduation. It also offers numerical values ranging between 0 (lowest) and 5 (highest) per knowledge topic for each of the five disciplines. These numerical values represent the relative emphasis each discipline would expect to place on each knowledge topic.

Another method of identifying the commonalities and differences among the computing disciplines is applying Bloom's taxonomy levels per knowledge topic for each discipline [5]. The Bloom's taxonomy represents three levels of capability of a student after graduation:

- 'v' (vocabulary): a student should be to understand the terminology to delegate to others if necessary;
- 'c' (comprehension): a student should be able to intelligently discuss the topic and perform basic tasks;
- 'a' (application): a student should be able to apply knowledge to perform tasks competently as required in a work environment.

The remainder of the paper is organized as follows. We begin by discussing how to complement course contents between IT and CS using their commonalities and differences, and related teaching

methodology in Section 2. Section 3 uses a case study to demonstrate how database course contents are complemented between IT and CS. Section 4 discusses how the IT and CS curricula can benefit from each other before concluding the paper in Section 5.

2. COMPLEMENTING COURSE CONTENTS BETWEEN IT & CS

The IT discipline has become a separate baccalaureate program over the past few decades. Even though [14] has empirically found that the IT discipline is significantly different from CS, there are commonalities between IT and CS. This section discusses how complementing course contents between IT and CS benefits both students and educators.

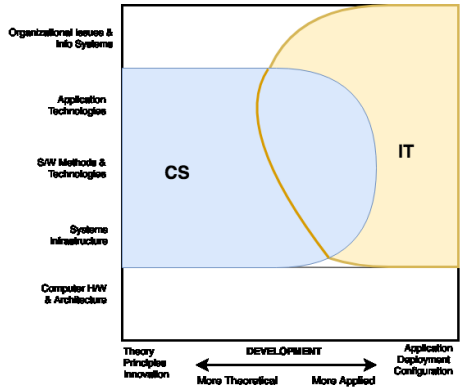

Figure 1. Problem space of IT & CS [3]

2.1 Motivation

Computing Curricula 2001 [2] divided the computing-related disciplines into four separate ones: Computer Engineering (CE), Computer Science (CS), Software Engineering (SE), and Information System (IS). Information Technology (IT) has joined the family of the computing as a separate discipline in the Computing Curricula 2005 (CC2005) [3]. An empirical study also supported that IT programs are significantly different from CS or IS [14].

Table 1. Bloom's taxonomy levels of technical knowledge [5].

ID	Technical Knowledge	CS	IT	General engineer/ software developer	Embedded systems designer	Business/ IT systems designer	Interaction/ UI designer	Service Support	Business analyst
D6	Programming language types	a	v	a	a	a	c	c	c
D7	Data Structure	a	c	a	a	a	a	a	a
D8	Algorithms	a	c	a	a	a	a	a	a
D9	Info & data modeling	a	c	a	a	a	a	a	a
D10	Databases	c	a	a	a	a	a	a	a
D11	Bus process & activity modeling	c	a	a	c	a	c	a	a
D13	Enterprise architecture & modeling	v	c	c	c	c	c	c	c
D14	Networking	c	v	c	c	c	c	c	c

Even though IT and CS are different enough to be offered as separate baccalaureate programs, it does not imply that their curricula are completely different from each other. CC2005 illustrated the commonalities and differences among computing disciplines graphically as shown in Figure 1, which displays what students in each discipline typically do after graduation. The X-axis displays from Theory, Principles, Innovation on the left, to Application, Development, Configuration on the right. CS graduates would fit to work in a discipline toward the left, and students who graduate from IT would fit to work in the discipline toward the right. The Y-axis runs from Computer Hardware and Architecture at the bottom, to Organizational Issues and Information Systems at the top. Figure 1 demonstrates commonalities and differences between IT and CS based upon the types of work that graduates from each discipline conduct. In other words, IT courses can be complemented by theoretical concepts, and conversely more applied topics can apply to CS courses, which provide students with more insights or motivations for not only understanding the course contents better but also performing better at work after graduation.

Another way of identifying commonalities and differences among computing disciplines is applying Blooms taxonomy levels: 'v', 'c' or 'a' to a specific discipline or education program, where the Bloom's knowledge levels are presented as follows [5]:

- Vocabulary (v): a graduate understands the terminology in a conversation about the topic enough so that graduate can delegate to others.
- Comprehension (c): a graduate should be able to intelligently discuss the topic and perform basic tasks.
- Application (a): a graduate can apply the knowledge in order to perform tasks in the area with a level of competence as expected in a work environment.

Table 1 includes a partial list of Blooms taxonomy levels of a Body of Knowledges (BoK) that defines the technical knowledge for CS and IT. A complete list of taxonomy levels can be found in [5] for eight knowledge areas of Professionalism & Ethics, Law & Regulations, Mathematics Foundations, Technical Knowledge, Quality Issues, Process Knowledge, and Business Knowledge for Computing and IT. For example, as shown in Table 1, CS graduates go out to the real world with the Bloom's taxonomy level of 'a' on Algorithms (D8), and IT graduates start their career with 'a' level of technology knowledge on Database (D10). We can complement course contents of Algorithms and Database between IT and CS to extend students' capacity towards job performance or graduate study.

2.2 IT & CS Course Resources for Complementation

Among resources like textbooks, reference books, websites, journal and conference papers, ACM SIGITE and SIGCSE conference papers published for IT and CS, respectively, are important for complementing course contents. For example, Table

2 lists a number of papers related to technical knowledge of databases published in SIGITE and SIGCSE by Pedagogical Objectives: curriculum, course, lab, research, and assessment over the past 15 years.

IT database papers published in SIGITE for potential complementation for CS courses include: Using Oracle & SQL Server to teach SQL [13]; In-process object-oriented database design for .NET [17]; and Integrating mobile storage into database courses [11].

CS database papers published in SIGCSE for potential complementation for IT courses include: Integrating XML into a database systems course [16]; A query simulation system to illustrate database query execution [4]; and Design patterns for database pedagogy [12].

Table 2. DB publications in SIGITE & SIGCSE ('03-'18).

Pedagogical Objectives	SIGITE	SIGCSE
Curriculum	0	3
Course	7	14
Lab	1	0
Research	3	3
Assessment	1	1
Total	12	21

2.3 Teaching Methodology

[1] discusses teaching methodologies when the focus of the course is based on Theory; Practice; Project; or All-encompassing/comprehensive (Theory + Practice + Project). This approach is especially appropriate as discussed in Section 2.1: CS courses are more theoretical than IT, and IT courses are more practical than CS.

In summary, complementing course contents among different degree programs such as IT and CS benefit from enhancing/broadening students' understanding of knowledge for better career opportunities including pursuing higher education; helping students become a lifelong learner; and having educators offer additional topics to challenge overachieving students.

3. CASE STUDY

This section will demonstrate how the IT and CS programs complement each other. We will use the database curricula in both programs at our university as a case study and discuss the application of the skills learned in the programs after graduation.

3.1 DB Curriculum in IT and CS

3.1.1 IT DB Curriculum. Among the BS programs offered by our IT discipline is the BS-Computing and Information Technologies (CIT). The original basis for the program was "The five-pillars of IT" later referenced in Information Technology 2008 (IT2008) [9]. The CIT program conforms to the Information Technology Curricula 2017 (IT2017) [10]. The core includes two database (DB) courses and it has a strong DB concentration.

The first core DB course, Introduction to Database and Data Modeling (ISTE-230), covers the basics of the DB environment. Topics include the theoretical foundations of data organization and the relational model, relational mapping and normalization, basic EER modeling and SQL. The second core DB course, Information Requirements Modeling (ISTE-430), includes topics such as requirements gathering, lifecycles, documenting requirements, conceptual, logical and physical views, as well as, process, data, and state analysis and modeling.

After the core database courses, students choosing to concentrate in database applications take three of the following courses to deepen their skills.

Database Connectivity and Access (ISTE-330) covers topics such as cloud computing, database drivers, basic operations and error handling, prepared statements, transactions and using stored procedures, security and integrity including SQL injection and audit trails, security, and basic extract, transformation and load (ETL) operations.

This course could be followed by Database Application Development (ISTE-432), which includes software testing, locking schemes, handling dirty data, performance and statement pooling, domain specific issues, authorization and authentication, and working with multiple data and business layers.

Data Warehousing (ISTE434) covers topics including the star schema and more advanced topics such as advanced dimension design, special types of fact tables, slowly changing dimensions, and in-depth coverage of the ETL process. Implementation of a data warehouse is also required.

Data Management and Access (ISTE-436) is a course aimed at database administration. It includes an examination of the physical and logical views of a database, logical and physical database layout, access to metadata, users, privileges and roles, and backup and recovery.

Contemporary Databases (ISTE-438) covers a variety of topics that are evolving in the big data world including big data storage, NoSQL databases (document, graph, memory resident, column). It includes the advantages and disadvantages of using a NoSQL database, geographic data storage and retrieval, and storage of large objects.

Data Mining and Exploration (ISTE-470) covers data mining and visualization, data-driven discovery, ethical concerns, association rule mining, clustering and classification methods, anomaly detection, and the analysis of results.

The CIT core database courses cover the IT2017 Essential Domain of ITE-IMA Information Management while the concentration courses cover the IT2017 Supplemental domain of ITS-DSA Data Scalability and Analytics.

3.1.2 CS DB Curriculum. The CS Department offers the BS program in Computer Science (CS/BS) with the specializations in areas such as artificial intelligence, computer graphics, computer theory, networking, security, robotics, parallel computing, data management and mining, computer architecture, and system software. The database courses that correspond to Information

Management (IM) in ACM CS curricula 2013 [6] include, Principles of Data Management (CSCI-320) and Database System Implementation (CSCI-621). CSCI-320 is a core course for all CS majors and CSCI-621 is an elective course for both graduate and undergraduate students.

CSCI-320 provides a broad introduction to the principles and practice of modern data management. It focuses on relational database models with the combination of theoretical and practical knowledge. Topic-wise, it covers the entire data management cycles, starting from data management concepts, database systems, database applications, data modeling, to indexing, query language, security and access control, transaction management, information retrieval, and data mining.

After taking CSCI-320, students can take CSCI-621 to learn database systems in a different perspective, i.e., from how to use it to how to design and develop it. Similar to CSCI-320, CSCI-621 focuses on relational database systems. It covers the topics related to the design of core DBMS functions related to physical data storage, file systems and buffer management, query processing and optimization, transaction management, access methods, and recovery. It also covers the topics related to managing large volumes of data through NoSQL databases and MapReduce.

The core CS database course, CSCI-320, covers the fundamental aspects of Information Management (IM) component of ACM CS curricular 2013 [6], i.e., the core-tier 1 hours, core-tier 2 hours, and some electives related to the understanding and using databases. The elective CS database course, CSCI-621, covers the theoretical and implementation aspects of the IM component with the major focus on physical database design, query language, and transaction processing.

3.2 Complementing IT and CS

Anecdotal evidence from conversations with employers in the US and Europe indicates that employers with certain entry positions prefer IT graduates because the students can be productive immediately. CS students, on the other hand, take more time to learn the application of the technology. In the long view, they can be more useful in other technologies because CS graduates have a deeper understanding of the theory behind the technology.

Figure 1 shows the reason for this. If we look at the IT discipline, the program emphasizes the "more applied" end of the x-axis; application, deployment and configuration. In the database courses that the CIT program offers, we emphasize those skills using MySQL, Oracle, MongoDB, Neo4j and other database products. When first on the job, graduates are already well versed in using the technology and can be immediately productive. Also, since the integration of technologies and the "human side" of computing is the focus of the IT degree, the IT graduates can be more immediate in looking at the information systems approach than the CS graduates as shown on the y-axis. On the other hand, the CS graduates have a better understanding of the theory and its application to the technology. They will be better at writing algorithms and examining complex problems. In this way, the two disciplines will complement each other on the job. The goal of each should be for the IT graduate to pull the CS graduate up and to the right on the figure 1 graph while the CS graduate pulls the IT graduate to the left, as they work together on the job.

The way that each program is set up enhances the skills in their respective roles; CIT applied and CS theoretical. One example is the CIT course, Introduction to Database and Data Modeling (ISTE-230). This course covers basic database theory and practice. The theory is covered by lectures and reading material. This would include topics such as normalization and relational algebra. Practice exercises, often completed in class, reinforce the theory. Lecture by example and practice exercises in areas like SQL, which are completed before practical homework, emphasizes the application of the technology. The emphasis in the course is mostly on the practical application of the database concepts.

As motivated in section 2, the skills learned in the IT program complement those of a CS program. There are areas where changes in the IT curriculum could benefit the communication among IT and CS professionals in the workforce. A database example would be looking at SQL query efficiency. In the CIT course, there is a discussion of query efficiency but it is not emphasized. In Bloom's taxonomy, we would say that the IT student would know the vocabulary or possibly comprehension of query efficiency. Today's servers and RDBMS query optimizers are so effective that efficiently coding the query is less important in many applications. That is, unless the query runs repeatedly in a large number of iterations. The CS student would understand the application of query efficiency.

3.2.1 Complementing IT DB with CS.

Various papers suggest that query efficiency is important in CS courses and the theory is covered. [4] suggests that CS students need to be familiar with the query execution process because it affects the query optimization process and other critical database factors. [8] goes on to show comparison of response times from queries yielding the same results but coded differently. Different ways to write selected queries to improve performance are then demonstrated, using WHERE instead of HAVING where applicable in GROUP BY queries, for example. [7] discuss the factors that affect the efficiency of the queries such as missing or excessively fragmented indexes, inexact database statistics, poorly written queries, deadlocks, and frequent query compilation. IT database courses touch these topics, but do not cover them extensively. Again, IT students are at the vocabulary or comprehension level of Bloom's taxonomy.

We would suggest that there is a benefit to cover topics such as indexing, database statistics, optimization of query coding practices and analyzing query execution plans more heavily in the IT database courses so that an IT professional's communication with CS graduates can be more productive in improving database performance or comparing different database products. The IT courses would still be very heavy on practice but provide a deeper understanding of the theory underlying the practice. In terms of Bloom's taxonomy for improving query efficiency, the IT students may move into the comprehension or application knowledge levels from the current vocabulary level.

3.2.2 Complementing CS DB with IT.

It is common that students in IT database courses are also taught specific databases. [13] shows that teaching a specific database, such as Oracle, and SQL together can help students better understand ANSI-standard SQL. By learning how an implementation can be deviated from SQL standard, students would not only better understand the general SQL knowledge, but be better prepared to use any specific database after the class.

We would suggest that the CS database course, while still focusing on teaching general database concepts and techniques, could benefit from including topics related to specific database systems. In addition to learning a concrete implementation of SQL standard, students can benefit from learning specific data management features offered by individual databases. For example, Oracle supports optimization tuning, allowing users to proactively monitor a system's performance, analyze statistics related to database, application, operating system, network, and disk I/O, and tune performance through the SQL tuning advisor. Introducing those topics into CS courses would help students better understand data queries through the link from those queries to their concrete executions and help them learn the strategies of improving the quality of those queries.

4. DISCUSSION

From the above analysis, we have reached several important observations. We first summarize these observations and then offer our suggestions on how the IT and CS curricula can benefit from each other, which may help shape the future IT and CS curricular development.

First, both CS and IT students start to learn a similar set of foundational topics in the entry level courses and then deviate from each other when it comes to more advanced topics. For example, the topic list of the introduction to database courses from both CS and IT shows a significant overlap with over 90% of topics appear in both lists. This is reasonable as students need to get to know the basic vocabularies, fundamental concepts, and standard functionalities to gain a good overall understanding on how a database management system (DBMS) works. However, the focus becomes very different in an advanced elective course in database. In the database implementation course that is usually offered to CS students in their senior year, topics show a strong algorithmic flavor. In many institutions, CS students need to implement a small-scale DBMS or its major technical components. In contrast, the advanced elective database course in the IT curriculum (e.g., the Database Management and Access course as discussed in this paper) typically focuses on the practical usage of a popular DBMS, such as Oracle or SQL Server. In such a course, students are usually exposed to a popular DBMS that is commonly used in the industry and gain practical skills of how to use and manage the system. The distinct training that students receive from these two different curricula may help determine the workforce readiness and lead to different career paths for IT and CS students, which will be discussed later in this section.

Second, the delivery methods also tend to be different. In particular, hands-on exercises usually play an important role when teaching courses in an IT curriculum. The difference in the teaching methodology is essentially driven by the different focus of the IT and CS curricula. As discussed in Section 2, IT courses tend to have a more practical focus as compared to CS courses which are more theoretical. The difference in the focus can be directly reflected from the actual subjects covered in the corresponding courses. For example, in the advanced database courses as discussed above, IT courses usually cover specific technologies through one or two popular DBMSs while CS courses focus more on the fundamental building blocks and their foundational underpinnings and tend not to discuss any specific database products. Additional evidence can also be found in the published SIGITE and SIGCSE papers. Most of the database related SIGITE papers discuss specific technologies or database products, including Oracle, SQL Server, and .NET while corresponding SIGCSE papers tend to discuss general concepts/theories, such as XML, query simulation/execution, and design patterns, which are not restricted to any specific system or product. A key benefit of the hands-on delivery of teaching materials is that students learn the best practice from the course instructors and get exposed to the right way of doing things. By following the course instructor's well-designed examples, students can effectively avoid making many common mistakes. However, one potential downside of such teaching approach is that students may get used to following course instructors' way of doing things instead of being challenged to develop their own version of best practice through a trial-and-error process. In addition, as the hands-on practice will take a large portion of the class, students may not have the opportunity to learn more and deeper concepts that can further extend their knowledge.

Third, the difference in the curricular content and teaching methodology will lead to IT and CS graduates with quite different skill sets. In particular, for the IT graduates, due to the practical focus of the IT curriculum and the rich hands-on experience that they have developed during the learning process, they are very well prepared for the job that they are hired to do and require little or no additional training. As a result, IT graduates have a high reputation in terms of their workforce readiness. In contrast, most CS graduates need some transition time and/or training when moving from school to their job environment. Since the CS curriculum primarily focuses on the underlying theories and the general foundation, it is usually hard for CS students to develop systematic knowledge on a specific system or product. Nonetheless, since the theories and foundation are typically shared across different systems and products, after the transition and training, most CS graduates will be able to perform very well in their jobs. Furthermore, thanks to their deeper knowledge in various subjects and systematic training and practice in algorithm development, CS students usually demonstrate stronger capability to adapt to new technological advances, which can significantly benefit their career path in a longer term.

Based on the above observations and our own experience as IT and CS educators, we would like to offer some suggestions that

can help improve the design of future IT and CS curricula and better prepare graduates from both programs for their feature career. First, it appears that complementing both the course content and the teaching methodology from IT and CS will bring additional benefit to both IT and CS students. In fact, in the authors' institution, there are always a good number of CS students in each term that choose to take some IT courses and the same situation also applies to the IT students. In essence, students are trying to using their own ways to receive complementing training from both IT and CS curricula. Second, extra caution should be given when choosing topics for complementing these two curricula because it is still of critical importance to keep the unique characteristics of each curriculum so that they can properly serve students with different background and interest. Ultimately, IT and CS graduates should not possess completely overlapping skills so that they can serve different needs from their future employers. Last but not least, the complementing design should provide enough flexibility that accommodates diverse needs from students with different background and skill sets. It may be beneficial to follow a modular design that groups similar topics into course modules, where some modules may have a theoretical focus and other may have a practical one. Given the needs and requirements from the students, different modules can be offered in a flexible way. Some courses could be intentionally designed to offer to both CS and IT students, where they have the opportunity to work together on some team project to learn from each other. Such team projects will also help better prepare students for their future working environment, where people with different skills collaborate to accomplish a large-scale project.

5. CONCLUSION

In this paper, we use database courses offered by both IT and CS curricula as a case study to identify commonalities and differences between these two important computing disciplines. Our analysis further helps reveal the root reason that causes those differences, which eventually show significant impact on the graduates' future career paths. In particular, a strong emphasis on the practical skills better prepares IT graduates for their workforce readiness while a deeper theoretical knowledge base puts CS graduates in a better position to adapt to technological advances that benefits them in a longer term. Given the identified differences in terms of both topics and teaching methodologies, we provide important recommendations that may help shape the future IT and CS curricular development.

REFERENCES

[1] Adnan Abid, Muhammad Shoaib Farooq, Ishaq Raza, Uzma Farooq, and Kamran Abid. 2015. Variants of Teaching First Course in Database Systems, Bulletin of Education and Research, December 2015, Vol. 37, No. 2 pp. 1-17.

[2] ACM/IEEE-Curriculum 2001 Task Force. Computing Curricula 2001, Computer Science. IEEE Computer Society Press and ACM Press, December 2001. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2001.pdf.

[3] ACM/IEEE-Curriculum 2005 Task Force. Computing Curricula 2005, Computer Science. IEEE Computer Society Press and ACM Press, September 2005. https://ieeecs-media.computer.org/assets/pdf/CC2005-March06Final.pdf.

[4] Brett Allenstein, Andrew Yost, Paul Wagner, and Joline Morrison. 2008. A query simulation system to illustrate database query execution. In Proceedings of the 39th SIGCSE technical symposium on Computer science education (SIGCSE '08). ACM, New York, NY, USA, 493-497. DOI: https://doi.org/10.1145/1352135.1352301.

[5] Canada's Association of Information Technology (IT) Professionals, "A Guide to the Common Body of Knowledge for Computing and IT (CBOK)", Canadian Information Processing Society, Canada retrieved May 24, 2019 from http://www.cips.ca/CBOK.

[6] Computer Science Curricula 2013 - Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. The Joint Task Force on Computing Curricula Association for Computing Machinery (ACM), Dec. 2013.

[7] Costel Gabriel Corlăţan, Marius Mihai Lazăr, Valentina Luca, and Octavian Teodor Petricică. 2014. Query Optimization Techniques in Microsoft SQL Server. Database Systems Journal vol. V, no. 2/2014.

[8] Jean Habimana. 2015. Query Optimization Techniques - Tips for Writing Efficient And Faster SQL Queries. International Journal of Scientific & Technology Research Volume 4, Issue 10, October 2015.

[9] IT2008 Task Group. 2008. Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology. http://pin.epsig.uniovi.es/informatica/GradoII-TI/IT2008%20Curriculum.pdf.

[10] IT2017 Task Group. 2017. Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology. DOI: http://it2017.acm.org.

[11] Qusay H. Mahmoud, Shaun Zanin, and Thanh Ngo. 2012. Integrating mobile storage into database systems courses. In Proceedings of the 13th annual conference on Information technology education (SIGITE '12). ACM, New York, NY, USA, 165-170. DOI: https://doi.org/10.1145/2380552.2380602.

[12] Thomas J. Marlowe, Cyril S. Ku, and James W. Benham. 2005. Design patterns for database pedagogy: a proposal. In Proceedings of the 36th SIGCSE technical symposium on Computer science education (SIGCSE '05). ACM, New York, NY, USA, 48-52. DOI: https://doi.org/10.1145/1047344.1047375.

[13] Gary B. Randolph. 2003. The forest and the trees: using oracle and SQL server together to teach ANSI-standard SQL. In Proceedings of the 4th conference on Information technology curriculum (CITC4 '03). ACM, New York, NY, USA, 234-236. DOI: https://doi.org/10.1145/947121.947174.

[14] Han Reichgelt, Barry Lunt, Tina Ashford, Andy Phelp, Erik Slazinski, and Cheryl Willis. 2004. A comparison of baccalaureate programs in Information Technology with baccalaureate programs in Computer Science and Information Systems, Journal of Information Technology Education, 3, 19-34, 2004.

[15] Russell Shackelford, Andrew McGettrick, Robert Sloan, Heikki Topi, Gordon Davies, Reza Kamali, James Cross, John Impagliazzo, Richard LeBlanc, and Barry Lunt. 2006. Computing Curricula 2005: The Overview Report. In Proceedings of the 37th SIGCSE technical symposium on Computer science education (SIGCSE '06). ACM, New York, NY, USA, 456-457. DOI: https://doi.org/10.1145/1121341.1121482.

[16] Paul J. Wagner and Thomas K. Moore. 2003. Integrating XML into a database systems course. SIGCSE Bull. 35, 1 (January 2003), 26-30. DOI: https://doi.org/10.1145/792548.611924.

[17] Yanhao Zhu, James Crouch, and Mohammad H. N. Tabrizi. 2005. In-process object-oriented database design for .NET. In Proceedings of the 6th conference on Information technology education (SIGITE '05). ACM, New York, NY, USA, 139-142. DOI=http://dx.doi.org/10.1145/1095714.1095747.