Healthcare Database Management for Health Informatics and Information Management Students: Challenges and Instruction Strategies—Part 1

by Ray Hylock, PhD, and Susie T. Harris, PhD, MBA, RHIA, CCS

Abstract

Enactment of the Health Information Technology for Economic and Clinical Health (HITECH) Act under the American Recovery and Reinvestment Act (ARRA) of 2009 forever altered the data management landscape in healthcare. The volume, breadth, depth, and pace at which health information is collected have surged. Effective data management is crucial for patient treatment, regulatory compliance, population health, cost management, and quality control. As a result, the health informatics and health information management disciplines have required greater educational emphasis on data management, governance, quality, and analysis. In this article, the first of a two-part series, the authors present strategies for creating a comprehensive healthcare database course. Factors influencing course design, such as themes that differ from those of traditional database courses and accreditation requirements, are addressed. The authors propose minimum and expanded construct sets for database modeling and interrogation, define a minimum feature set for database management systems used for instruction, discuss hosted and individually maintained systems, and evaluate commonly used database management systems. The result of this work is a framework and methodology for devising a healthcare database management course specifically intended for a health informatics and health information management audience.

Keywords: healthcare databases, healthcare database instruction, health information management (HIM), health informatics (HI), health information technology, database course development/design, HI/HIM instructional design, HI/HIM course development, health information technology instructional design

Introduction

Healthcare information systems have become a seemingly ubiquitous component of modern healthcare largely because of the electronic health record mandate resulting from the Health Information Technology for Economic and Clinical Health (HITECH) Act. They are instrumental in the treatment of patients, hospital administration, regulatory compliance, population health, cost management, quality control, research, and other purposes. Therefore, the accuracy, timeliness, relevancy, validity, and integrity of the information are of utmost importance.[1, 2] Thus, the roles of health informatics (HI) and health information management (HIM) professionals have had an increasing focus on health information technology.[3–6]

At the heart of a healthcare information system is the database management system (DBMS). Virtually every critical work area defined by the American Health Information Management Association (AHIMA) HIM Workforce Study[7] necessitates at least a cursory understanding of database systems. This need is echoed in the global health information technology workforce training program framework of Cortelyou-Ward et al., in which database management is cited as one of three integral elements of the data sphere.[8] Although topics such as databases, data mining, and analytics obviously require in-depth knowledge, a general understanding is also important to adequately manage personnel and projects, support research and analytic initiatives, evaluate future scenarios and products, and comply with state and federal laws, regulations, and statutes.[9–11] Therefore, a proper background in database management is imperative for HI and HIM students and professionals.

Of the nine HI and HIM master's programs accredited by the Commission on Accreditation for Health Informatics and Information Management Education (CAHIIM), six require a database management course, and one allows it as an elective.[12] All nine comply with the minimum database and data management requirements for accreditation, yet by necessitating a stand-alone database management course, the majority of accredited institutions have signaled the importance of these skills for the profession.

This article, the first of a two-part series, presents the methodology behind the creation of the graduate-level course in healthcare database systems at East Carolina University. The methodology is applicable to baccalaureate-level courses as well. The construction of the course draws from the authors' 14 years of combined experience designing, implementing, and instructing on databases in professional, research, and academic settings. First, the merits of offering a healthcare-centered database course are discussed, followed by discussion of the course design itself. Every lecture is mapped to CAHIIM accreditation standards, with justification provided for each topic. Database modeling and structured query language (SQL) are elaborated upon, and minimum and expanded construct sets are proposed for each of these topics. The discussion then moves to the selection of a database management system for the course as well as the benefits and drawbacks of hosted versus individual installations. Part 2 of the series will provide greater specifics on designing a flexible course, deconstruction strategies for database modeling and interrogation instruction, and healthcare-focused sample data sets.

The Divergence of Healthcare Database Courses from Traditional Database Courses

Administrators of each HI and HIM program must decide whether to require a database course and, if the course will be required, whether to offer a healthcare-oriented course within the program or use a non-healthcare-focused course from outside the program. Common substitutes include database courses in computer science (CS) and management information systems (MIS) departments. Course content for these replacements is typically stable across colleges and universities because they rely on curriculum guidelines set forth by their respective professional organizations (e.g., Association for Computing Machinery [ACM] and IEEE for the computer science, computer engineering, information technology, and software engineering curricula;[13] ACM and Association for Information Systems [AIS]for the information systems curriculum; [14] and Information Resources Management Association [IRMA] and DAMA for the information technology curriculum[15]). These courses generally cover database theory (e.g., relationship characterization, data modeling, functional dependencies, normalization, and concurrency controls) and database application design (e.g., database implementation and querying).[16, 17] However, on the basis of experience in teaching and studying databases from CS, MIS, and healthcare perspectives, the authors of this article recommend against outsourcing database courses for the following reasons.

First, whether the course is offered in a CS or MIS department, HI and HIM students will constantly be seeking healthcare analogs to the examples and problems presented in the textbook and lectures. Although this task appears innocuous, it adds an undue burden to the student and can lead to confusion and even incorrect application. Second, most CS textbooks portray examples and definitions in the form of equations, proofs, and set theory, which are unfamiliar to typical HI and HIM students, decreasing

retention and discernment.[18–20] Third, healthcare laws and regulations pertaining to data and information security, such as the Health Insurance Portability and Accountability Act (HIPAA) and the HITECH Act, are not discussed in non-healthcare-related courses. Although these topics may be addressed in other courses, applying these rules to actual data sets solidifies the concepts for the student. Lastly, the data sets, databases, and data warehouses used in these courses are not as complex and standardized as those used in healthcare. That is, healthcare databases store highly diverse and heterogeneous data, maintain standard data sets for reporting and compliance, utilize various data standards and vocabularies, transmit data using different standards (e.g., HL7), require immense privacy and security oversight, and are typically an amalgamation of multiple sources. Combined, these shortcomings provide a strong fundamental argument in favor of a healthcare-focused database course. However, if offering such a course is not feasible or practical, then a MIS database course is generally a suitable alternative. In fact, of the seven CAHIIM-accredited programs with a database option, three outsource the course to MIS; the remaining programs offer a healthcare-focused course.

Course Design

The course design follows the minimalist theory of database instruction.[21] The name of this theory is a misnomer because it does not imply less substantive content; rather, it refers to a course progression that builds on self-contained minimal activities, using error recognition and recovery to reinforce concepts in realistic, interactive environments. Such environments include the concepts of active learning, theory-to-practice, and scaffolding, which have been repeatedly shown to increase student comprehension and retention of database material.[22–28] Mohtashami and Scher refer to this pedagogical theory as a transition from "sage on the stage" to "guide on the side"[29]—that is, from rote lecturing to hands-on topic immersion and discovery. Minimalist objectives incorporate pre- and postmodule learning outcomes. For instance, one would first describe single-relation operations prior to describing multirelation operations. Error recognition and recovery can be a powerful pedagogical tool in this area.

Error recognition and recovery is a process of providing positive and negative examples, forcing the student beyond problem solving to solution finding. To continue the previous example, the demonstration of single-relation anomalies segues seamlessly into normalization and multirelation operations. Additionally, one could demonstrate the perils of an unconstrained database by adding, for example, encounters for patients who do not exist, and then trying to produce a bill. These errors reinforce the concepts of referential integrity. Stated simply, the goal of error recognition and recovery is to move from simply solving a problem to identifying errors (whether logical or syntactic) and providing solutions.

Conforming to Accreditation Standards

Accreditation signals to the public that a program has met and maintained an acceptable level of quality as determined by an external accrediting body in the field of study. CAHIIM provides this function for the HI and HIM disciplines. CAHIIM produces curriculum competencies for associate through master's degrees in HIM, master's degrees in HI, and five specializations.[30] As a HI master's program (in candidacy) offering RHIA eligibility (accredited), East Carolina University must comply with both the master's in HI (MHI) and baccalaureate in HIM (BHIM) requirements. Thus, although this article focuses on the graduate-level curriculum, it is also applicable to baccalaureate healthcare database course design.

Table 1 presents lecture topics for the Database Systems in Health Care course at East Carolina University. Several factors contributed to the material and its order. First, the minimum content level was derived from the explicit database, data management, data quality, and modeling requirements of the accreditation standards. From there, a rough course outline was prepared and expanded to include concepts such as privacy and security, de-identification, database merging, concurrency controls, and NoSQL (nonrelational) databases. Each topic is mapped to the BHIM, MHI, and, for completeness, master's in HIM (MHIM) curriculum requirements using the current versions of the requirements (year denoted in parentheses in the table). Adjacent to the domain or facet is the portion of the standard covered. It is the responsibility of each program to ensure that competencies are met following the

guidelines specified in the accreditation documents. Therefore, the competencies may be considered to be met if full consideration of the section is included in instruction. A justification for each featured competency is given in Table 2, which was compiled on the basis of the authors' years of experience teaching and working with database management systems, as well as input from the program's advisory board.

Database Modeling and SQL Constructs

Research has repeatedly shown that database modeling (topic 3) and interrogation (topics 5 to 7) provide the greatest difficulty for students and instructors alike.[31–34] From the students' perspective, modeling requires the production of interconnected, abstract, logically related elements, while interrogation (typically through SQL) necessitates procedural thinking. The literature focuses on students in computer science, information systems/technology, and related technical fields, where students have more exposure to these topics through ancillary courses than HI or HIM students typically have because of their varied backgrounds (e.g., business, biology, chemistry, nursing, psychology, and information technology). Thus, the students' difficulties in HI and HIM courses are generally greater than the research suggests. This difficulty exacerbates instructional challenges.

The instructor must provide a minimum level of education not only to meet explicit accreditation requirements but to prepare the student for work in the field and future classes (e.g., project management, policy evaluation and creation, data governance and analysis, systems analysis and design, and decision support), while reducing extraneous features that may be less frequently utilized and generally confusing for students in introductory courses. Thus, the exact set and depth of constructs to cover is not always clear. In the following sections, a detailed breakdown of constructs for each topic and justification for inclusion of the topic in the minimum and/or expanded construct sets is given. A minimum construct set defines the narrowest collection of constructs conveyed to students to satisfy accreditation and basic system utilization requirements. The expanded construct set presents additional features that are desirable to include if time and student progress permit.

Database Modeling

A data model is a collection of concepts used to describe the structure of a system, and is widely considered to be the most important component of systems development. Data models characterize the facts, rules, and integrity controls of a system; analyze data rather than processes (which are the most complex element in modern information systems); provide a guide for information inquiry, analysis, and summary; and are generally stable compared to business processes, experiencing greater lifetimes and usefulness.[35] Thus, data modeling is an integral component of systems analysis and design, project management, software engineering, and database development, and other areas, as well as in HI and HIM education.

Pertaining to database design, the most pervasive model is the entity-relationship (ER) model. Standards of ER diagramming do not exist; therefore, many forms are available.[36–38] Regardless of the chosen form, all support the defined minimum and expanded construct sets (Table 3and Table 4).

Structured Query Language

SQL is the de facto database language standard. As such, it is the most commonly taught language for database interrogation. SQL is generally deconstructed into three main languages. The first is Data Definition Language (DDL). DDL commands define a database by creating, altering, and dropping schemas, tables, attributes, constraints, indices, and views. Without knowledge of such commands, students would be unable to design and manage databases. Table 5 and Table 6 present the proposed minimum and expanded construct sets related to DDL.

The next language is the one typically associated with SQL—Data Manipulation Language (DML). DML commands are those that query and maintain data; these functions are referred to as CRUD (create, read, update, delete) operations. HI and HIM students should be familiar with at least the minimum

construct set presented in Table 7to populate, manipulate, and extract information from databases. Time and student success permitting, elements from the expanded construct set in Table 8can be incorporated.

The final language, Data Control Language (DCL), administers a database. These commands control a database and transactions by way of administering privileges, bounding transactions, and committing data, to name a few. DCL commands are essential functions to ensure data integrity and database security; thus they are mandatory for instruction. Table 9 and Table 10 present the minimum and expanded construct sets for DCL statements.

Recommended Database Management System Features

Many DBMSs are available for use in learning environments. Common systems include Microsoft Access, Oracle, MySQL (owned by Oracle Corporation), Microsoft SQL Server, and PostgreSQL. In this section, a minimum DBMS feature set for instruction is proposed. This list considers CAHIIM accreditation requirements and draws on the authors' years of experience teaching and utilizing various DMBSs in professional, instructional, and research capacities.

The DBMS should include the following features:

- Full support of the DDL, DML, and DCL minimum and expanded construct sets outlined in Tables 5, 6, 7, 8, 9, and 10.
- A database catalog (also referred to as metadata tables): This feature provides information about database objects such as relations, indices, constraints, and views, and system data such as statistics and privileges.
- Concurrent user support: The DBMS should provide an environment where user-based and role-based access controls (DCL) are necessary.
- Aggregate functions: At a minimum, the standard five aggregate functions (average, count, max, min, and sum) are necessary, but support for basic measures of spread for analysis of populations and samples (e.g., standard deviation and variance) are desirable to assist with integrity controls, quality monitoring, data analytics, and reporting.
- A robust SQL interface with profiling support (e.g., explaining and tracing queries): Drag-and-drop query design is suitable in basic situations. However, many healthcare queries involve complex joins (e.g., nested outer and recursive unary joins—refer to Table 7 and Table 8 for more details) and subqueries, requiring manual SQL programming. A robust query interface allows for quick connections, database navigation, syntax highlighting, query profiling, and error detection and reporting.
- Large data set support: Healthcare data by nature are historical and verbose, necessitating vast quantities of storage. This need for storage translates into secondary systems as well (e.g., registries, indices, and de-identified data sets). Support is generally measured at the file (e.g., table) and database level.
- Entire curriculum support: The chosen DBMS must be capable of supporting the activities of the educational program in general. That is, the choice of DBMS cannot be limited to the needs of a single course. If the chosen system fails to meet the requirements of other courses or topics (e.g., quality, data analytics, and decision support), student learning opportunities will be lost to software training in different courses.
- System compatibility: The software must be compatible with student hardware and must be affordable (free if possible). Most students use low to midrange laptops of the Windows or, with increasing frequency, Mac variety; occasionally, a student will use some form of the Linux operating system. All platforms, therefore, should be supported unless the academic program specifically states that a student must own or have access to a particular platform, for example, a Windows-based machine.

Table 11 compares five common database systems intended for individual users on the basis of the features identified. Here, the Express editions of Oracle and SQL Server are discussed. The results

indicate that MySQL and PostgreSQL are prime contenders for course selection. Although Oracle XE and SQL Server Express are free, neither supports the full range of potential operating systems (Windows and Mac at a minimum), and the imposed hardware restrictions can degrade performance with larger data sets. In the authors' experience (as both instructor and student), it is difficult to fully uninstall Oracle because it leaves residual files in the system. SQL Server Express is an attractive solution because of the prevalence of SQL Server implementations in healthcare. However, the increased presence of Mac computers on campuses removes it from consideration. Third-party SQL interfaces such as Oracle SQL Developer[39] and SQuirreL SQL[40] are available, but their use limits the benefits of training on a particular system. For instance, relying on SQL Developer to interact with SQL Server restricts the student's exposure to the SQL Server ecosystem (e.g., database browsing, built-in functions, database linkages, and profiling tools unique to SQL Server Management Studio), diminishing the intended effect.

Microsoft Access requires a more detailed examination. This software is widely used to teach databases in non–computer science settings. Many basic database courses or those with a database module rely on Access to provide a user-friendly tool with a shallow learning curve. In these environments, the use of Access is acceptable or even encouraged. However, a comprehensive healthcare database course typically exceeds the limits of Access.[41–43] Supported operating systems aside, Access is designed to take a one-database-to-a-file approach and, as of version 2010, does not support user-level security. From a database management and administration instructional perspective, the lack of a database catalog and the lack of privileges to manipulate make this software less than desirable. Additionally, query performance in Access diminishes rapidly as the number of tuples (records) increases to even moderate sizes. Finally, to invoke the query profiler, the user must modify the Windows registry, which is not to be attempted by novice users; furthermore, this hack cannot be used with the cloud version of the software. Therefore, Access should be avoided in a comprehensive healthcare database course.

Hosting a database for a course is also an option. An instructor might consider doing so, for example, to make use of a commercial tool for which the university, college, or department has available licenses; to maintain control of shared DBMS resources (e.g., adding and removing data, altering schemas, and having a central "grading" repository); or to eliminate the need for student installation (thereby alleviating the need for the instructor to provide technical support). Table 12 represents the same feature set comparison between DBMS server options. It should be noted that Table 12 compares the commercial versions of Oracle and SQL Server, not their Express versions. An additional attribute—client operating systems—is added to report operating systems supported by the native interface of the DBMS. Here, only SQL Server, which supports only Windows clients, is ruled out as a viable option. Because Oracle requires a commercial license, MySQL and PostgreSQL are again the top contenders.

As instructors, the authors have experimented with both hosted and user-based DBMSs in various database courses; a list of observations regarding key installation and support activities is presented in Table 13. In the authors' experience, the choice generally comes down to the instructor's preference. Most institutions have the capability to host databases and provide some level of support. After attempting both methodologies, the authors prefer a user-based approach, mostly out of the desire to maintain control. That is, utilizing hosting sources almost certainly requires the instructor to relinquish administrator privileges, limiting his or her ability to, for example, troubleshoot small problems (those that are less pressing from the standpoint of the information technology department) and grant or revoke privileges (from a security and instructional point of view).

Conclusions

The high-volume, data-intensive healthcare landscape has thrust data management to the forefront of the HI and HIM professions. As a result, preparatory programs have placed a greater emphasis on instruction in data management. In this article, the design of a graduate-level course in healthcare database management is presented, along with discussion of the importance of offering a healthcare-specific version of the course rather than outsourcing it to another department. The course is mapped in its entirety to CAHIIM accreditation standards for BHIM, MHI, and MHIM programs to indicate topic coverage if sufficiently implemented. Data modeling and SQL constructs are discussed in finer detail

because of the elusive nature of balance in HI and HIM courses resulting from the diversity of student backgrounds. Minimum and expanded construct sets are proposed for these topics to support the course and perceived needs within an accredited program at large. Lastly, a minimum DBMS feature set for instruction is proposed, and traditional DBMSs used in the classroom in individual and hosted environments are compared. The authors of this article recommend using either MySQL or PostgreSQL, with hosted or individually managed configurations at the discretion of the instructor.

The second part of this series will present instructional strategies used in East Carolina University's healthcare database course. Tactics for providing a dynamic course, deconstructing complex problems, and ensuring advanced student engagement will be detailed. Additionally, the article will provide access to multiple data sets that the authors have constructed and curated specifically for healthcare database education.

Ray Hylock, PhD, is an assistant professor in the College of Allied Health Sciences at East Carolina University in Greenville, NC.

Susie T. Harris, PhD, MBA, RHIA, CCS, is an associate professor and director of the Master of Science in Health Informatics and Information Management Program at East Carolina University in Greenville, NC.

Notes

1. Warner, Diana. "IG 101: What Is Information Governance?" *Journal of AHIMA*, December 4, 2013. Available at http://journal.ahima.org/2013/12/04/ig-101-what-is-information-governance/.
2. Warner, Diana. "IG 101: Enterprise Information Governance." *Journal of AHIMA*, December 11, 2013. Available at http://journal.ahima.org/2013/12/11/ig-101-enterprise-information-governance/.
3. Bates, Mari, et al. "Perceptions of Health Information Management Educational and Practice Experiences." *Perspectives in Health Information Management* (Summer 2014).
4. Palkie, Brooke. "The Perceived Knowledge of Health Informatics Competencies by Health Information Management Professionals." *Educational Perspectives in Health Informatics and Information Management* (Winter 2013).
5. Fenton, Susan H., et al. "Health Information Technology Employer Needs Survey: An Assessment Instrument for HIT Workforce Planning." *Educational Perspectives in Health Informatics and Information Management* (Winter 2013).
6. Warner, Diana. "IG 101: Information Asset Management." *Journal of AHIMA*, December 13, 2013. Available at http://journal.ahima.org/2013/12/13/ig-101-information-asset-management/.
7. American Health Information Management Association. "Embracing the Future: New Times, New Opportunities for Health Information Managers. Summary Findings from the HIM Workforce Study." 2005. Available at http://library.ahima.org/xpedio/groups/public/documents/ahima/bok1_027397.hcsp?dDocName=bok1_027397.
8. Cortelyou-Ward, Kendall, Summerpal Kahlon, and Alice Noblin. "Competencies for Global Health Informatics Education: Leveraging the US Experience." *Educational Perspectives in Health Informatics and Information Management* (Winter 2013).
9. Jacob, Julie A. "HIM's Evolving Workforce: Preparing for the Electronic Age's HIM Profession Shake-Up." *Journal of AHIMA* 84, no. 8 (August 2013): 18–22.
10. American Health Information Management Association. "Embracing the Future: New Times, New Opportunities for Health Information Managers. Summary Findings from the HIM Workforce Study."
11. White, Susan, and Sandra Nunn. "Two Educational Approaches to Ensuring Data Quality." *Journal of AHIMA* 85, no. 7 (July 2014): 50–51.
12. Commission on Accreditation for Health Informatics and Information Management Education. "Welcome to CAHIIM." Available at http://www.cahiim.org/index.html (accessed October 1, 2015).
13. Association for Computing Machinery. "Curricula Recommendations." Available at https://www.acm.org/education/curricula-recommendations (accessed October 1, 2015).
14. Ibid.
15. Cohen, Eli B. "Rationale for the IRMA/DAMA 2000 Model Curriculum." *Information Resource Management Association International Conference* (2001): 612–16.
16. Mohtashami, Mojgan, and Julian M. Scher. "Application of Bloom's Cognitive Domain Taxonomy to Database Design." *Proceedings of the Information Systems Education Conference* (2000).

17. Connolly, Thomas M., and Carolyn E. Begg. "A Constructivist-based Approach to Teaching Database Analysis and Design." *Journal of Information Systems Education* 17, no. 1 (2005): 43–53.
18. Garcia-Molina, Hector, Jeffrey D. Ullman, and Jennifer D. Widom. *Database Systems: The Complete Book*. Upper Saddle River, NJ: Prentice Hall, 2001.
19. Silberschatz, Abraham, Henry Korth, and S. Sudarshan. *Database System Concepts*. 5th ed. Boston: McGraw-Hill, 2005.
20. Ullman, Jeffrey D., and Jennifer Widom. *A First Course in Database Systems*. 3rd ed. Upper Saddle River, NJ: Pearson/Prentice Hall, 2008.
21. Kwan, A. C. M. "Towards a Minimalist Approach to Lesson Planning of an Introductory Database Course." *2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering* (2013): 782–85.
22. Boyeena, M., and P. Goteti. "Promoting Active Learning through Case Driven Approach: An Empirical Study on Database Course." *2010 IEEE Students' Technology Symposium* (2010): 191–95.
23. Etlinger, Henry A. "Adding a Contributing Student Pedagogy Component to an Introductory Database Course." In *SIGCSE '13: Proceedings of the 44th ACM Technical Symposium on Computer Science Education*. New York: ACM, 2013, 299–304.
24. Kwan, A. C. M. "Towards a Minimalist Approach to Lesson Planning of an Introductory Database Course."
25. Rashid, T. A., and R. S. Al-Radhy. "Transformations to Issues in Teaching, Learning, and Assessing Methods in Databases Courses." *2014 International Conference on Teaching, Assessment and Learning* (2014): 252–56.
26. Wang, Jingmin, and Haoli Chen. "Research and Practice on the Teaching Reform of Database Course." In *International Conference on Education Reform and Modern Management (ERMM 2014)*. Paris, France: Atlantis Press, 2014, 229–31.
27. Sastry, Musti K. S. "An Effective Approach for Teaching Database Course." *International Journal of Learning, Teaching and Educational Research* 12, no. 1 (2015): 53–63.
28. Chen, Hsuan-Hung, Yau-Jane Chen, and Kim-Joan Chen. "The Design and Effect of a Scaffolded Concept Mapping Strategy on Learning Performance in an Undergraduate Database Course." *IEEE Transactions on Education* 56, no. 3 (2013): 300–307.
29. Mohtashami, Mojgan, and Julian M. Scher. "Application of Bloom's Cognitive Domain Taxonomy to Database Design."
30. Commission on Accreditation for Health Informatics and Information Management Education. "Welcome to CAHIIM."
31. Connolly, Thomas M., and Carolyn E. Begg. "A Constructivist-based Approach to Teaching Database Analysis and Design."
32. Kung, Hsiang-Jui, and Hui-Lien Tung. "A Web-based Tool for Teaching Data Modeling." *Journal of Computing Sciences in Colleges* 26, no. 2 (2010): 231–37.
33. Brusilovsky, Peter, et al. "Learning SQL Programming with Interactive Tools: From Integration to Personalization." *Transactions on Computing Education* 9, no. 4 (2010): 19:1–19:15.
34. Kwan, A. C. M. "Towards a Minimalist Approach to Lesson Planning of an Introductory Database Course."
35. Elmasri, Ramez, and Shamkant Navathe. *Fundamentals of Database Systems*. 6th ed. Boston: Addison Wesley, 2010.
36. Ibid.
37. Hoffer, Jeffrey A., Mary B. Prescott, and Fred R. McFadden. *Modern Database Management*. 6th ed. Upper Saddle River, NJ: Prentice Hall, 2002.

38. Ullman, Jeffrey D., and Jennifer Widom. *A First Course in Database Systems*. 3rd ed.
39. Oracle Corporation. "Oracle SQL Developer." Available at http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index-097090.html (accessed October 1, 2015).
40. "SQuirreL SQL." Available at http://squirrel-sql.sourceforge.net/ (accessed October 1, 2015).
41. White, Susan. *A Practical Approach to Analyzing Healthcare Data*. 2nd ed. Chicago: American Health Information Management Association, 2013.
42. Fenton, Susan H., and Sue Biedermann. *Introduction to Healthcare Informatics*. Chicago: American Health Information Management Association, 2013.
43. Hoffer, Jeffrey A., Mary B. Prescott, and Fred R. McFadden. *Modern Database Management*. 6th ed.

Table 1

Commission on Accreditation for Health Informatics and Information Management Education (CAHIIM) Competencies Met by Lecture for Baccalaureate in Health Information Management (BHIM), Master's in Health Information Management (MHIM), and Master's in Health Informatics (MHI)

Lecture Topic (Number of lectures)	**Competencies Covered in Full or Part**		
	BHIM Curriculum Requirements (2014)	**MHI Curriculum Requirements (2012)**	**MHIM Curriculum Requirements (2014)**
Introduction to Database Systems in Health Care (1)	• Laws and regulations: II.A.1, III.H.2	• Privacy and security: I.10, I.11, III.10	• Privacy and security, assessment: II.B.1 • Regulatory and legal: III.B.1
Database Development, Systems Development Life Cycle (SDLC), and Prototyping (1)	• Capture, structure, and modeling: I.D.1, I.D.4, III.A.5, III.A.6 • SDLC: III.B.2	• Data representation: III.7 • Design: II.10, III.9 • Modeling: II.5	• Data representation: I.D.3 • SDLC and performance: III.A.5
Conceptual Design (2)	• Capture, structure, and modeling: I.D.1, I.D.4, III.A.5, III.A.6 • Data dictionary: I.C.2, I.D.4, III.A.6	• Data representation: III.7 • Design: II.10, III.9 • Modeling: II.5	• Data representation: I.D.3 • Design: III.A.7
Logical Design: Relational Model, Entity-Relationship (ER) Models to Relations, and Normalization (2)	• Capture, structure, and modeling: I.D.1, I.D.4, III.A.5, III.A.6	• Data representation: III.7 • Design: II.10, III.9 • Modeling: II.5	• Data representation: I.D.3 • Design: III.A.7
Physical Design: Databases, Tables, and Indices (1)	• Capture, structure, and modeling: I.D.1, I.D.4, III.A.5, III.A.6	• Architecture and design: III.9 • Data representation: III.7 • Design: II.10	• Data representation: I.D.3 • Design: III.A.7 • Performance: III.A.5
Structured Query Language (SQL) (2)	• Capture and structure: I.D.1 • Quality and integrity: III.H.4 • SQL: III.C.5, III.C.6	• Data representation: III.7 • Design: II.10, III.9 • Quality: I.4 • SQL: III.2	• Data representation: I.D.3 • Design: III.A.7 • SQL: III.A.7
Data and Database Administration (1)	• Administration: III.A.6 • SQL: III.C.5, III.C.6	• Security: I.11, III.10 • SQL: III.2	• Administration: III.A.7 • SQL: III.A.7

Data Quality and Integration (2)	• Integration: I.C.1 • Laws and regulations: II.A.1, III.H.2 • Quality and integrity: III.H.4 • SQL: III.C.5, III.C.6	• Privacy and security: I.10, I.11, III.10 • Quality: I.4 • SQL: III.2	• Database conversions: III.A.6 • Integration: I.C.1 • Privacy and security, assessment: II.B.1 • Regulatory and legal: III.B.1 • Threats to quality and validity: III.H.1
Archival and Retrieval Systems and Data Warehousing (1)	• Archival and retrieval: III.A.1 • Data warehousing: I.D.4, III.A.6 • Laws and regulations: II.A.1, III.H.2 • Modeling: I.D.1, I.D.4, III.A.5, III.A.6	• Data representation: III.7 • Data warehousing and design: III.9 • Modeling: II.5	• Archival: III.A.7 • Data representation: I.D.3 • Data mapping: I.E.1 • Data warehousing: I.E.1, III.A.7 • Database conversions: III.A.6 • Performance: III.A.5 • Regulatory and legal: III.B.1

Table 2
Justification for Each Covered Topic

No.	Topic	Justification
1	Healthcare laws and regulations	Databases maintain electronic protected health information; thus an understanding of the Privacy and Security Rules are paramount. Examples include Safe Harbor de-identification (45 CFR §164.514(b)(2)), access controls and management, audit controls, data integrity, authentication, encryption, retention, and data backup and recovery.
2	Database development, SDLC, and Prototyping	Because the course is about databases and database management, a detailed understating of development is required. Topics include the Systems Development Life Cycle (SDLC), prototyping, and abstract data modeling. SDLC and prototyping are two commonly used systems development frameworks. The former is for large-scale systems requiring detailed planning (e.g., electronic health record systems), while the latter is for rapid, iterative production of smaller projects (e.g., registries). Because database design falls into both categories, an understanding of the processes and appropriate adoption are necessary. Abstract data modeling (e.g., enterprise data modeling) is customary during initial planning and analysis to gain a high-level understanding of the project and is the foundation for the subsequent topic.
3	Conceptual design, logical design, and normalization	Conceptual design, logical design, and normalization are standard processes in the creation of a database definition; thus these concepts are a necessity. Conceptual design captures graphically (in the form of Entity-Relationship [ER] diagrams) the information to be stored within a database. It records basic entities (e.g., Patients), attributes, and relationships between entities (e.g., between Patients and Encounters), as well as integrity control properties (refer to database modeling in the next section for details). Logical design translates ER diagrams into database implementation specifications. These specifications are refined to remove anomalies (insert, update, and delete anomalies) through normalization. The result is a database definition primed for physical design.
4	Physical design	The last of the design steps, physical design applies platform-specific details to the logical model, for example, the application of database management system (DBMS)-specific data types. Additionally, appropriate file organization and indexing structures are selected to maximize performance. File organization pertains to data storage on disk and thus directly affects performance. An index, if implemented correctly, can decrease query execution times by storing bits of information commonly used (e.g., primary and foreign keys) in small, high-performance data structures generally residing in memory. The designated indexing structure can have a profound effect on performance. Once this process is complete, the database is ready for implementation.
5	SQL	Structured query language (SQL) is the standard interface language for DBMSs and is thus another critical skill for students to learn. The creation, management, and manipulation of data and databases are controlled via various types of SQL commands (refer to Structured Query Language in the subsequent section for details).
6	Administration	Database administration covers a wide swath of activities pertaining to the management, backup, recovery, performance tuning, and security of a database. Entwined with privacy and security, database administration protects the data from unauthorized access (45 CFR §164.312(a)(2)(i) and 164.312(d)), modification, injection, and deletion (45 CFR §164.312(c)(1)), while backup and recovery assist with contingency planning (45 CFR §164.308(a)(7)(ii)(A)-(B))—all vital knowledge and skills for health informatics and health information management professionals. Administration also includes concurrency controls, transactions and conflicts (e.g., lost updates, dirty reads, and incorrect summaries), and locks (e.g., shared and exclusive).
7	Quality and integration	Students must understand that capturing and storing information in an efficient manner does not imply that the data will be of high quality; controls and permissions conceived during the design process and updated throughout the database's lifetime are required. Additionally, knowledge of data cleansing and transformation techniques required for data integration from other sources is necessary, given the nature of healthcare systems and continued advancement in information exchange. Topics include data governance, extraction, quality characteristics, profiling, cleaning, transformation, and loading.
8	Archival and retrieval systems, data warehousing, and ETL	Archival and retrieval systems and data warehouses are key technologies in healthcare. Both store vast quantities of historical data and are used for advanced interrogation. Data are loaded into these systems via the process of extracting, transforming, and loading (ETL) information from multiple independent and heterogeneous data sources. This process requires extreme care (see previous topic) and generally results in de-normalized entities. Students are exposed to concepts in this area such as multidimensional modeling, various ETL types (e.g., consolidated, federated, and propagated), the appropriate use of these systems, and the laws pertaining to data retention.

Table 3

Database Modeling Minimum Construct Set

Modeling Constructs	**Justification**
Strong and weak entities	Strong and weak entities are the most basic types and therefore necessary. They represent independent data (e.g., Patients) and dependent data (e.g., Encounter and Medication pairings), respectively.
Attributes: primary keys, foreign keys, basic types, and decorations	The data are stored in attributes that must be modeled. Decorations delineate attribute types such as composite, multivalued, derived, and key. Data integrity relies on concepts such as primary and foreign key definitions.
Relationships	Associations between entities.
• Unary, binary, and *n*-ary	Unary relationships are those between an entity and itself, for example, patient and spouse (both patients). Binary relationships, connecting two entities, are the most common, for example, Patients and Encounters. In some situations, three or more entities participate in a relationship (e.g., Encounters, Procedures, and Providers).
• Minimum and maximum cardinalities	Cardinality is the specification of the minimum and maximum number of tuples in a relationship—a crucial data integrity control. For example, an encounter is for exactly one (*min* = *max* = 1) patient. If more than one patient or no patient were allowed in an encounter, for instance, then to whom would the bill be sent?

Table 4

Database Modeling Expanded Construct Set

Modeling Constructs	**Justification**
Aggregate and composite entities	Aggregate and composite entities are those created by constituent elements, the difference being the existence of the sub-element (child) independent of the super-element (parent). If independent, then aggregate, else composite. These are advanced modeling techniques to associate seemingly disparate tuples. For instance, a surgery can be modeled as an aggregation of personnel, equipment, facilities, and medication. The human body, however, is a composition of elements that alone cannot exist (e.g., a hand does not exist absent a body).
Super- and sub-types	A topic beyond basic introduction, super-types and sub-types allow for specialization of entities. For instance, a generic Providers table can be subclassified into surgeons, primary care providers, specialists, and nurses. This is necessary when a sub-portion of an entity is involved in additional modeling (e.g., relationships) in order to support data integrity and optimize storage requirements.
• Constraints: completeness and disjointedness	Completeness and disjointedness specify the minimum and maximum sub-type membership, which is a data integrity control.

Table 5

SQL Data Definition Language (DDL) Minimum Construct Set

DDL Constructs	**Justification**
Schemas	A schema defines a portion of a database owned by a particular user. Access can be limited by schemas; therefore, they are important constructs not only for storing logically related tables, but for ensuring security as well.
Tables and attributes	The basic data structures for information storage.
Constraints: primary key, foreign key, default values, optional, and domain	Constraints are integral to supporting data integrity. Primary keys enforce entity integrity (a unique record identifier such as patient ID), and foreign keys enforce referential integrity (a reference value in one table must exist in another, e.g., an encounter must be for an existing patient). Default values, optional modifiers, and domain constraints bound and control data entry.
Indices	An index is a data structure designed to improve the performance of query processing. Without indices, entire tables would need to be read from the hard disk, incurring massive overhead. Therefore, indices are vital to the successful implementation of a database, especially on a large scale (such as the typical size of an electronic health record) and during analysis (e.g., analytics, decision support, and data warehousing). At a minimum, students should understand the necessity, performance benefits, basic creation, and default instantiation (e.g., primary keys) of indices.
Views	Views, which are essentially stored queries. provide two main advantages. First, complex queries requiring real-time data can be stored and accessed the way a table would be; this process minimizes query writing and errors. Second, views are used as a security apparatus, in which a user has access to, for example, only a portion of a table. At a minimum, students should understand the basic uses and definition of views.

Table 6

SQL Data Definition Language (DDL) Expanded Construct Set

DDL Constructs	**Justification**
Indices	In advanced settings, students must go beyond merely understanding the basic function and intent of indices, to actively utilizing them to optimize performance. This use includes implementing more complex options (e.g., composite, join, partial, and unique) and understanding the strengths and limitation of the underlying data structures (e.g., bitmaps and the various B-trees).
Views	Views are a much more potent tool than their definition in the minimum construct set suggests. They can be scoped (i.e., exist only during the current session, referred to as a temporary view), updatable (not a statement, but a set of definitional conditions), and stored (i.e., the data can actually be stored with a defined maintenance frequency, referred to as a materialized view). These techniques allow for performance and security gains when properly implemented.

Table 7

SQL Data Manipulation Language (DML) Minimum Construct Set

DML Constructs	Justification
Selecting data	The means by which to extract data from a database management system (DBMS).
• Basic predicates: simple operators, [NOT] IN, [NOT] NULL, and DISTINCT	These predicates are employed for the most basic of SQL transactions, from basic querying to profiling to analytics.
• Aggregate functions	Aggregate functions such as average, count, max, min, and sum are readily used in reporting and basic data analysis and profiling.
• Advanced predicates: regular expressions, ranges, GROUP BY, HAVING, ORDER BY, and LIMIT	Many attributes are text-based and therefore require some form of regular expression for querying. Additionally, GROUP BY and HAVING are essential for aggregating and limiting subsets of data. ORDER BY is used for sorting, and LIMIT is used to constrain the number of results displayed. Each of these functions is necessary for database interrogation.
• Parentheses and SQL order of execution	Many questions in healthcare contain two or more sets of conditions, for example, patients greater than 65 with at least one disease from set A and one disease from set B. If the user does not have a proper understanding of the order of SQL execution and the effects of parentheses, the DBMS will incorrectly interpret the query, returning logically incorrect (yet syntactically correct) results.
• Joins: inner, natural, and Cartesian	One of the most basic SQL design patterns, a join allows a relation to be combined with another in a meaningful way. Inner joins are those most commonly associated with the term *join*. That is, the results contain only those tuples (records) existing in both sets. Natural joins achieve the same goal using an implicit join predicate. That is, a natural join relies on attribute names and data types from the relations being joined to determine how to combine them. Cartesian joins return the product of the relations (i.e., each tuple in one relation is matched with every tuple in the other). While Cartesian joins have legitimate uses, novice students should be aware of their existence because the result of a Cartesian join is effectively what they would see if they failed to include a join predicate for an inner join. Thus, exposing students to Cartesian joins is necessary for error recognition and recovery.
• Noncorrelated subqueries	Subqueries allow the results of one query to be used as a predicate value in another. For example, to determine the patient with the greatest weight, the maximum weight must first be known (the subquery). That value is then used to locate the encounter(s) with that recorded weight (outer query). This example is a noncorrelated subquery, in which the results of the subquery do not depend on the outer query.
• Views	Declaring a view is a Data Definition Language (DDL) command, but the query itself is any SELECT statement. From a DML perspective, views simplify employing queries and enhance programming productivity while maintaining access to current information (not true for materialized views). In an introductory setting, many students may be sufficiently challenged by basic SELECT statements, limiting the need to focus on views. At a minimum, student should have an understanding of what views accomplish and how to create them.
Inserting: single tuples, from SELECT	The means by which to populate tables. Data can be added one item at a time or from the results of a query. Although the former method is useful, the latter is commonly used for constructing secondary data sets, copying tables or results, and de-normalization.
Updating and deleting	The means by which to manage existing tuples.

Table 8

SQL Data Manipulation Language (DML) Expanded Construct Set

DML Constructs	**Justification**
Selecting data	The means by which to extract data from a database management system.
• Joins: outer and semi	In addition to the joins in the minimum construct set, outer and semi joins are necessary for advanced tasks and are generally more confusing in terms of appropriate application. Outer joins include tuples that do not have a match during a join. For example, they can be instrumental in database merging and extracting, transforming, and loading (ETL) operations that require all data to be migrated. Semi joins provide significant performance gains when data are transmitted and joined over a network. Examples include distributed and federated databases.
• Correlated subqueries and [NOT] EXISTS	The result of a correlated subquery is dependent upon an element within the current tuple in the outer query. For instance, to determine the maximum weight recorded for *each* patient, the subquery locates the maximum weight for the current patient in the outer query. Many questions in healthcare rely on applying subquestions to elements within sets, substantiating the inclusion of this construct in this set. Existence is essentially a Boolean response to the correlated subquery problem; that is, values exist or they do not.
• Views	While the minimum construct set promotes basic knowledge, expanded instruction should place greater emphasis on view construction to assist in complex query environments and optimization.
Combining sets: UNION [ALL], INTERSECT [ALL], and EXCEPT [ALL]	When working with multiple sets (e.g., query results or relations), it is sometimes necessary to combine or limit given elements in another. For instance, a researcher might be interested in patients with certain diseases who do not take medications in a given class. Assuming that diseases and medications are assigned to encounters and are not associated with one another, it would be impossible to join on the two in a meaningful way (other than a Cartesian join). Thus, query A would produce the set of patients meeting the disease requirement, and query B would produce the inverse medication constraint (i.e., NOT IN becomes IN to collect those in the set for removal). Patients in set A who also exist in set B are removed (EXCEPT), producing the desired patient set. This scenario is just one of many healthcare examples in which these set combination operators are necessary. However, because set combination requires a detailed understanding of set theory and basic query design, it is considered an advanced topic.

Table 9

SQL Data Control Language (DCL) Minimum Construct Set

DCL Constructs	Justification
Granting, altering, and revoking Data Definition Language (DDL), Data Manipulation Language (DML), and Data Control Language (DCL) privileges	Views are essential for database security and integrity because they control data access rights and available features, for example, revoking all modifying privileges to a record when it is placed under a legal hold, or allowing researchers to have read-only access to records.
User- and role-based access controls	Two privilege control techniques. The first allows for individual management; the second allows for group management. Role-based management is common to allow uniform access based on job description. That is, a nurse will have the same privileges as other nurses. Thus a "nurse" role is created with privileges, and nurses are assigned to that role. In special cases, a user might have additional privileges attached specifically to the individual, that is, user-based privileges.
Views	One intent of a view is to control access to data; thus, it is also a DCL construct. For instance, Social Security numbers may be recorded in the Patients table, but because the Social Security number has nothing to do with treatment, a view is created without this attribute. Providers are then given access to the view instead of to the full Patients table, protecting the information. As stated in Table 7, an understanding of views is sufficient to allow greater focus on basic database interrogation.

Table 10

SQL Data Control Language (DCL) Expanded Construct Set

DCL Constructs	Justification
Views	While the minimum construct set promotes a basic level of understanding, expanded instruction should place greater emphasis on view construction as a control measure.
Transaction controls	Transaction-level controls to ensure data integrity.
• COMMIT and ROLLBACK	Commits instruct the database to permanently write changes. If a transaction is not committed and a system error occurs (e.g., loss of power), it will be lost. Also, prior to a commit, a transaction can be rolled back (undone). These features allow the user to control the effect transactions have on data integrity and recovery.
• BEGIN and END	An advanced technique, a transaction block is an atomic wrapper telling the database that all of the enclosed SQL must successfully execute or the effects must be rolled back. This technique is another essential tool for data integrity.
• CHECKPOINT	Checkpoints flush all data files to disk without waiting for a regularly scheduled event. This control is an explicit call to "save" the data yet to be transferred from temporary space to the disk. It ensures the integrity of the data up to that point in case of system failure.

Table 11

Common Database Management Systems for the Individual User

Feature	Access 2013	MySQL 5.7	Oracle XE 11*g*	PostgreSQL 9.4	SQL Server Express 2014
Catalog tables	No	Yes	Yes	Yes	Yes
Data Definition Language (DDL)	Yes	Yes	Yes	Yes	Yes
Data Manipulation Language (DML)	Yes	Yes	Yes	Yes	Yes
Data Control Language (DCL)	No[a]	Yes	Yes	Yes	Yes
Concurrent users	Yes (limited)[b]	Yes	Yes	Yes	Yes (limited)
Aggregate functions	Yes	Yes	Yes	Yes	Yes
Structured query language (SQL) interface	Built-in	MySQL Workbench—universal	SQL Developer—universal	pgAdminIII—universal	SQL Server Management Studio Express—Windows
SQL profiling	Basic[c]	Yes	Yes	Yes	Yes
Maximum file size	None	16 TB	None	32 TB	None
Maximum database size	2 GB	64 PB	11 GB	Unlimited	10 GB
Supported operating systems[d]	Windows	Windows, Linux, Mac	Windows, Linux	Windows, Linux, Mac	Windows
License	Commercial; generally provided to students	Free Community Edition—GPL	Free (1 GB of RAM and 1 CPU)	Free—PostgreSQL License	Free (1 GB of RAM, 1 socket, and 4 cores max)

[a] As of Access 2010, user-level security has been removed.
[b] Without a database server or SharePoint site, users can simultaneously access a file via a shared folder.
[c] Requires the user to add a Windows registry key; therefore, this feature is only available with a local installation.
[d] Indicates native support for an operating system. It might be possible to use a virtual machine (e.g., VirtualBox), compatibility software (e.g., Wine), or a college/university-supported virtual computing environment to run the tool on an unsupported operating system.

Sources:

Microsoft Corporation. "Database Software and Applications | Microsoft Access." Available at https://products.office.com/en-us/access (accessed October 1, 2015).
Microsoft Corporation. "JETSHOWPLAN in Access 2013." Available at https://social.msdn.microsoft.com/Forums/office/en-US/b786a029-fa8c-4556-a40c-e749ba73499e/jetshowplan-in-access2013 (accessed October 1, 2015).
Microsoft Corporation. "Microsoft SQL Server Express Edition." Available at https://www.microsoft.com/en-us/server-cloud/products/sql-server-editions/sql-server-express.aspx (accessed October 1, 2015).
Microsoft Corporation. "Ways to Share an Access Database." Available at https://support.office.com/en-za/article/Ways-to-share-an-Access-database-b7f250cc-5413-4fc4-a54a-8e2b54db252c (accessed October 1, 2015).
Microsoft Corporation. "What Happened to User-Level Security?" Available at https://support.office.com/en-au/article/What-happened-to-user-level-security--828a8be1-a9cd-463c-a07f-22d5abf78659?ui=en-US&rs=en-AU&ad=AU (accessed October 1, 2015).
Oracle Corporation. "MySQL." Available at http://www.mysql.com/ (accessed October 1, 2015).
Oracle Corporation. "Oracle Database Express." Available at http://www.oracle.com/technetwork/database/database-technologies/express-edition/overview/index.html (accessed October 1, 2015).
Oracle Corporation. "Oracle VM VirtualBox." Available at https://www.virtualbox.org/ (accessed October 1, 2015).
PostgreSQL Global Development Group. "PostgreSQL." Available at http://www.postgresql.org/ (accessed October 1, 2015).
"WINE HG." Available at https://www.winehq.org/ (accessed October 1, 2015).

Table 12

Common Hosted Database Management Systems

Feature	**MySQL 5.7**	**Oracle 12*c***	**PostgreSQL 9.4**	**SQL Server 2014**
Catalog tables	Yes	Yes	Yes	Yes
Data Definition Language (DDL)	Yes	Yes	Yes	Yes
Data Manipulation Language (DML)	Yes	Yes	Yes	Yes
Data Control Language (DCL)	Yes	Yes	Yes	Yes
Concurrent users	Yes	Yes	Yes	Yes
Aggregate functions	Yes	Yes	Yes	Yes
Structured query language (SQL) interface and profiling	Yes	Yes	Yes	Yes
Maximum file size	16 TB	32 TB	32 TB	16 TB
Maximum database size	64 PB	Unlimited	Unlimited	512 PB
Supported operating systems	Windows, Linux, Mac	Windows, Linux	Windows, Linux, Mac	Windows
Client operating systems	Window, Linux, Max	Windows, Linux, Mac	Windows, Linux, Mac	Windows
License	Free Community Edition—GPL	Commercial	Free—PostgreSQL License	Commercial

Note: MySQL 5.7 and PostgreSQL 9.4 are duplicated from Table 11 for ease of comparison.

Sources:

Microsoft Corporation. “Microsoft SQL Server.” Available at https://msdn.microsoft.com/en-us/library/bb545450.aspx (accessed October 1, 2015).

Oracle Corporation. “MySQL.” Available at http://www.mysql.com/ (accessed October 1, 2015).

Oracle Corporation. “Oracle Database.” Available at https://www.oracle.com/database/ (accessed October 1, 2015).

PostgreSQL Global Development Group. “PostgreSQL.” Available at http://www.postgresql.org/ (accessed October 1, 2015).

Table 13

User-based versus Hosted Database Management Systems (DBMSs): Breakdown by Installation and Support Activities

Feature or Requirement	**User-based**	**Hosted**
Hardware	At least 4 GB of RAM, 5 GB of hard drive space, and two cores.	Requires a server or virtual machine with at least 16 GB of RAM, 100 GB of hard drive space, and four cores. The supporting department may charge a fee.
Software licensing	None if a free DBMS or hosted DBMS is involved.	None unless using a commercial tool without available licenses; then it can be very expensive. The supporting department may charge.
Installation	The student is required to install the software.	The student installs only a client. The instructor must ensure that the DBMS is installed and configured correctly. The supporting department may charge.
Networking/firewall pass-through/VPN (virtual private network) access	None for a local installation.	The server must be accessible off-campus (especially with the growth of distance education in health informatics and health information management programs). This need requires collaborating with the information technology (IT) department to allow access to the machine, whether by direct firewall pass-through or VPN access. Also, if the network is down, students cannot access the system. The supporting department may charge.
Technical support	The instructor provides technical support because a university/college resource is not in use.	Generally, the hosting department, such as IT, will provide support, which frees the instructor from those responsibilities. However, the supporting department may charge.
Assignment setup	The student is responsible for properly importing files.	The instructor can preload data into student schemas.
Assignment assistance	Questions require emailing structured query language (SQL) files and ensuring that data files are correctly imported.	Questions are generally easier to deal with because a shared space is available (i.e., the instructor has access to the student's data), but SQL files are still emailed.
Database administration instruction	Simple because the student is the system administrator.	Difficult because a protected environment must be established for each individual student. This task requires extensive planning and configuration by the instructor and the supporting department, which, again, may charge.
DBMS security	Nothing additional because the installation is stand-alone.	Each student must be provided a fully encapsulated environment. Their privileges must constrain them to their space (e.g., they cannot be allowed to crosswalk into other students' schemas or databases), yet allow the functioning of Data Definition Language (DDL) and Data Control Language (DCL) commands.