DATABASE DESIGN FOR THE COLLECTION AND VISUALIZATION OF SUSTAINABILITY DATA

by

HENRY A. KOZLOWSKI

NIKHILA PERUGUPALLI

A Senior Project submitted

in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Industrial Engineering

California Polytechnic State University

San Luis Obispo

Graded by:___________________________ Date of Submission:____________________

Checked by:__________________________ Approved by:__________________________

Contents

Abstract

The goal of the project was to create a database that would be a repository for the high volume of sustainability data Illumina would like to collect. To couple with the database the project team's goal was to design a system for visualizing the data graphically. Illumina required a database system that would be user-friendly, easy to add, delete, and edit data inputs, as well as a program that would allow the user to easily sort and graph data. The intent of the graphical representation of the data is to publish the graphs in a monthly report that would be circulated internally, as well as some graphs that would be published through public domains. A number of difficulties along the way led to some crucial design changes, including Microsoft Access being the only well-established program capable of meeting the project goals, and Microsoft Excel being the only system capable of creating dynamic graphs that can be easily exported.

Introduction

Illumina lacked a process for tracking sustainability metrics and reporting. The team was asked to help them solve this problem. This report will outline exactly what the objectives were, how they were achieved, and the lessons learned throughout the process.

The project originated because Newsweek magazine contacted Illumina in 2014 asking them for certain environmental data in order to report-out on their impact. In 2014, Illumina realized that they did not have this data organized and that it was all scattered in different areas of the company. The need then arose for the creation of a central location that the company could store their data for energy consumption, greenhouse gas emissions, water consumption, and waste production. The team was then contacted to help with establishing a process for collecting and reporting data.

The objectives of the project are as follows:

- Design and implement a database that will house all of the sustainability data
- Establish a process for analysis and reporting of the data (e.g. creatively designed graphs)
- Communicate effectively with the Health & Safety team as well as region representatives to meet expectations
- Create user and administrator guides for the system so that the company can continue to maintain it after the team finishes the project
- Ensure that stakeholder expectations and objectives are met through each step of the process to deliver a quality product

The team met these objectives by following a detailed process. First, the team created a list of all of the features that the database design should contain. Then, proper research was done to plan out the system design so that all of these features are present in the most efficient method. Next, the front-end of the database was created so that it was user-friendly. Through each step of this creation process, the stakeholders played a large role by constantly providing feedback. After the database was designed, the reporting aspect was researched, planned, and created. Finally, user guides were written for system administrators and users to properly utilize the database.

The deliverables for this project included weekly meetings with the company, a database (with multiple prototypes), a reporting tool and user guides. The scope of the project did not include the creation of a database management process to be used in future once the amount of data stored in the system goes beyond the capacity of the system.

Background

The impetus for this project is the global focus on how mankind affects the environment. The world is putting more focus on how to reverse some of the damage we have caused to our environment, and minimize future harm. There are entire fields of study devoted to sustainability, and companies are changing the way they interact with the world on a sustainability front, even if the profit motive is not present. There is true social value these days in being a greener company and giving back to the community.

One of the companies feeling the changes is Illumina, located in San Diego, CA. They are an industry leading biotechnology firm focused on revolutionizing the way DNA sequencing is done. As this young company continues to mature and develop, they are looking for new ways to set themselves apart from their competitors, and one way to do that is by being on the leading edge of new trends, like sustainability. In 2014, Newsweek magazine published their Green rankings of the Fortune 500 companies, in which Illumina place 435. This was very troubling to the company, so they decided to make a plan to change that.

The main reason Illumina received such a low ranking was due in part to their failure to publish a formal sustainability report. Within the company, the Health, Safety, and Environment (HSE) team decided to form a group of employees to handle developing a report for the upcoming rankings in 2015. A few representatives were sent up to San Luis Obispo, CA to propose a senior project to a group of industrial engineering students. The project would involve creating a system to help gather and represent data that would have high visibility with the public and internally. Two students, Henry Kozlowski and Nikhila Perugupalli, agreed to partner with Illumina on the project.

The structure of the project would be based around gathering data from the three main regions, Americas (AMR), Asia-Pacific (APAC), and Europe, Middle East, and Africa (EMEA). Henry's responsibility was the APAC region, Nikhila's responsibility was the AMR region, and the EMEA region was split between the two. The team had weekly meetings with headquarters in San Diego, as well as monthly meetings with each member's respective region. There were no

prior systems in use, so everything that was done for this project was new and unique. It was also clear from the beginning that there would need to be some sort of database to store the data that would be collected, therefore much research was done involving sustainability reporting, database design, and user-friendly software designs.

Literature Review

This literature review will cover the basic elements of a relational database and sustainability reporting and assurance. This review explores the basic levels of database design as they relate to complexity, user interaction, and flexibility. The parts of sustainability reporting and assurance covered relate to how companies go about producing the reports and validating them. Both topics tie in with the research topic at hand by providing a background for how databases are designed, and an idea of the issues that go into creating and collecting data for a sustainability report.

Sustainability Reporting and Assurance

A sustainability report is a lot different that the annual reports we are used to seeing from companies. We are used to seeing the pure financial quantitative performance data in those reports, but a sustainability report takes on the task of combining quantitative and qualitative information into one holistic report (Searcy, Buslovich, 2013). These reports are often called corporate social responsibility (CSR) reports, which include indicators of environmental, governance, economic, and ethical performance. Sustainability reports can be used to understand a company's sustainability issues, to benchmark existing processes, to learn how the company

performs, to establish accountability internally, and to gather market information about the company (Searcy, Buslovich, 2013). In doing so the report also helps to avoid criticism from the public while also improving the credibility of the reporting company to its stakeholders and the general public (Searcy, Buslovich, 2013).

Although the report can be used to improve credibility and avoid criticism the major types of motivations for companies to report on sustainability are internal pressures, external pressures, and the opportunity for the company to tell its story (Searcy, Buslovich, 2013). The sources of external pressures for reporting are usually legal requirements, stakeholder requests, competitive pressures, and transparency requirements (Searcy, Buslovich, 2013). The main sources of internal pressures for reporting come from management requests, policy commitments, employee expectations, and the need to improve over time (Searcy, Buslovich, 2013). There are other added benefits of reporting; the report is often used by recruiters to attract top talent to the company, due to the fact that the new generation of working talent is socially and environmentally aware (Searcy, Buslovich, 2013). One of the companies that were interviewed in a study on sustainability reporting said that adopting the practices of CSR helps to track performance which becomes the impetus for change and creates a sense of responsibility internally instead of relying on external pressures (Searcy, Buslovich, 2013).

The current uses of sustainability reporting found by a study on reporting were enhancing employee awareness and engagement, and for use as an internal reference tool to the company and its employees (Searcy, Buslovich, 2013). Having the report available allows employees to

get involved in the process of improving company sustainability. For example, an employee with the knowledge of the company's sustainability efforts might make more environmentally and socially aware decisions. In the process of reporting, a majority of companies consult external stakeholders, and refer to guidelines like the AA1000 and Global Reporting Initiative (GRI), and consider some form of integration with the annual report (Searcy, Buslovich, 2013). It has become increasingly more common for companies to follow the GRI guidelines for reporting because it is becoming increasingly institutionalized, and as more companies begin to follow it, other companies feel compelled to follow suit (Searcy, Buslovich, 2013).

Another trend in the way companies report has seen many of them simply combine their annual report and sustainability report into one big report rather than integrating them together (Searcy, Buslovich, 2013). According to a report by KPMG done in 2007 approximately 80% of larger corporations published some form of a sustainability report, however only 40% of larger corporations integrated with their annual report (Manetti, Toccafondi, 2011). The reasons companies do not do this is because they believe that the two reports have difference audiences, they need to keep the annual report to a manageable size for the public, and in some cases the reporting cycles of the annual report are different than the sustainability report (Searcy, Buslovich, 2013). For example, the bills for annual water usage and other sustainability metrics may be gathered in February, while the fiscal year may end in September. This prevents the reports from being integrated, and therefore often the reports are completed separately.

Another major component of sustainability reporting is sustainability reporting assurance. This is the process of an external entity verifying the contents of the report are adequate to publish and tell the whole truthful story of the company sustainability. The sustainability reporting process is a difficult one as previously mentioned, pertaining the combination of quantitative and qualitative data, and thus the process of sustainability reporting assurance is difficult as well. Sustainability reporting assurance is still very immature and in its embryonic stages of development (Manetti, Toccafondi, 2011). In the process of moving toward a more mature assurance system the GRI has created a way to improve reporting credibility with the advent of 'A+ GRI Checked' (Manetti, Toccafondi, 2011). This is a formal check done by the GRI once the company has compiled its report to verify it is valid, while also contributing to the credibility of GRI. The 'A' represents a company who has reported on all the indicators and other numbered elements required by GRI. The '+' means the company has sought professional assurance services by an independent external auditor. The 'GRI Checked' means the company has also had a formal check of the report contents by the GRI.

A lot of the challenges in the reporting process arise due to overlapping timelines, data collection, selection of content, and striking an appropriate balance when reporting (Searcy, Buslovich, 2013). Additional minor challenges come from having to coordinate people and resources, the changing focus of stakeholders, and the management team's lack of experience and knowledge relating to sustainability reporting (Searcy, Buslovich, 2013). It can also be hard to justify the use of resources going toward sustainability because many investors are financially driven, and do not consider the non-financial performance of a company (Searcy, Buslovich,

2013). As well, assurance firms have difficulties because the process of assurance itself is difficult to formalize (Manetti, Toccafondi, 2011).

As the process of sustainability reporting and assurance develops there have been some predictions of what the future holds. One of the main uses of sustainability reporting companies would like to develop is the ability to track real-time performance (Searcy, Buslovich, 2013). The ways they think this can be done is by developing better data standardization and tools to pull data together (Searcy, Buslovich, 2013). Experts on the topic predict that in the future sustainability reporting and compliance may become mandatory for companies if they would like to be listed on the various stock exchanges. They also think that the institutions like AA1000 and GRI will need to create more comparable metrics because it may be undermining the legitimacy of the reports (Searcy, Buslovich, 2013).

Database Design

If one is to understand the process of designing a functioning database, one must first understand the basic building blocks of databases. There are three traditional models of databases; relational, hierarchical, and network (Chan, Wei, Siau, 1993). For the purposes of this review we will focus mainly on the relational database, as it is the most common current database. A relational database is made up of tables and relationships between those tables, or entities with relationships. Data is stored in the tables, and within the tables there are records, which are essentially rows of data that have different attributes (the columns). Basic database operations can be performed on the data using the relationships between the various tables. The static

properties of a database include the database structure - that is, the stored data items and their logical organization (Leitheiser, March, 1996). Within databases, there are things called queries that allow the user to perform commands used to update, delete, and retrieve data (Leitheiser, March, 1996). A query language is defined by its semantic and syntactic rules (Leitheiser, March, 1996). The semantic rules refer to the meaning of the language, so in other words, what the actual words mean when writing the language. The syntactic rules of the language refer to the way things are written. These rules are often rigid and require the user to write in a very exact manner or else the faulty syntax will cause the queries not to work.

Databases can have different levels of abstraction, which essentially refers to the level of complexity with parts of a software or program. The three levels of abstraction when it comes to a database are the physical, logical, and conceptual levels (Chan, Wei, Siau, 1993). The lowest level, the physical level requires the user to know the details of the data structures in the computer memory and storage (Chan, Wei, Siau, 1993). The logical level requires the user to know the layout of the logical data and relationship among data elements in the database world (Chan, Wei, Siau, 1993). Examples of a logical level understanding of databases are the languages of SQL, QBE, and relational algebra. The conceptual level deals with objects in the user's world, where the database is supposed to understand the user's world of entities and relationships (Chan, Wei, Siau, 1993). An example of the three levels of abstraction can be seen in Figure 1. Studies have shown that higher levels of abstraction make it easier for the user to learn how to use a database because there is less the user needs to know about database anatomy. However the study also shows the higher level of abstraction does not help users who are already experts at the logical level (Chan, Wei, Siau, 1993).

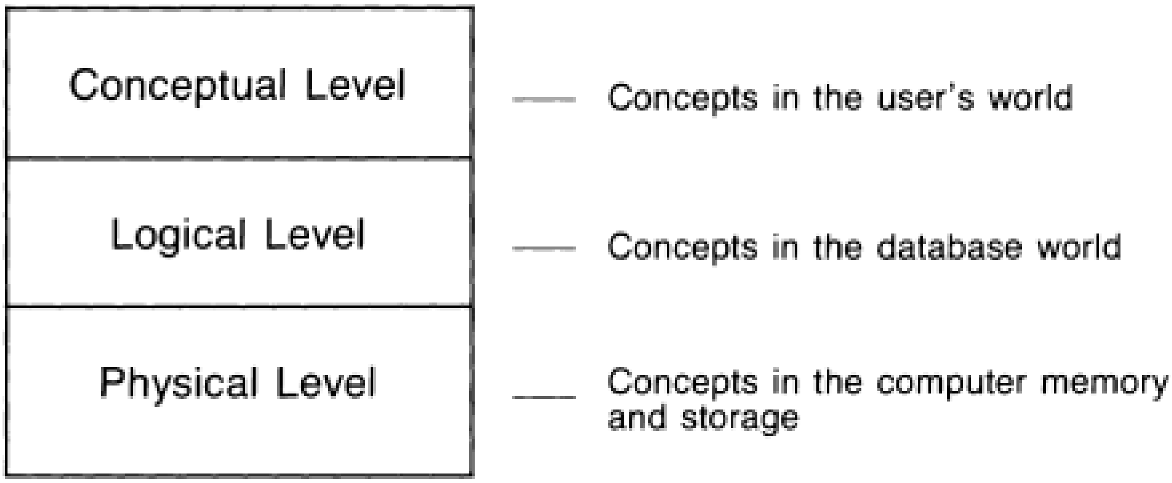


Figure 1: The Three Levels of Database Abstraction

There are four steps of database design; the four steps are (1) requirements specification, (2) conceptual design, (3) logical design, and (4) physical design (Storey, Goldstein, 2013). Requirements specification is concerned with identifying the information needs of various user groups (Storey, Goldstein, 2013). A designer must understand how the users of the database will interact with the database in order to create a sufficient product. Conceptual design models the users' and applications' views of information and includes a specification of the processing of user information. The objective is to produce a precise, formal representation of the requirements that is independent of any particular database management system or approach (Storey, Goldstein, 2013). The logical design phase sees the general conceptual model translated into the logical data model of the selected database management system (Storey, Goldstein, 2013). For example, the conceptual design may be translated in to the relational database representation using tables. The final phase, physical design, transforms the logical data model design into a form that is suitable for the specific hardware and database management system that are going to

be used (Storey, Goldstein, 2013). This is quite an in-depth process, and requires a lot of skill and prior knowledge in database implementation. Currently, the process is usually performed in an ad-hoc style by a so called ‘database design expert’ (Storey, Goldstein, 2013).

There are some weaknesses in the current approach to database design, including the fact that there are not very many expert database designers. As more is understood about the design of databases, a new level of database education is springing up. An effective database requires the user to learn how to read and use databases, which can be an intensive and difficult process (Leitheiser, March, 1996). In the learning process it has been found that users writing queries at a conceptual level (with a user interface) were more accurate and faster than users writing queries at the logical level (hard coding), which some say is attributed to SQL being easier to make mistakes unknowingly (Chan, Wei, Siau, 1993). SQL requires the user to be very exact with syntax, and any errors can cause the entire code not to work. A conceptual interface allows the user to visualize what the query is performing, and often has automatic correcting functions. If learning the SQL language is necessary it may be better to stay away from the explicit representations of relationships found at the conceptual level, because studies show it can negatively impact logical query language learning and use (Leitheiser, March, 1996). It also seems that any advantages object-oriented representations provide to database users will come from the semantics rather than from symbolic notation (Leitheiser, March, 1996). In other words, the user learns more from the meaning of the objects on the interface of the conceptual level, rather than from the actual use of the objects themselves.

The future of database design is very exciting, as new ways to use artificial intelligence (AI) systems to aid in the creation of database management systems is researched and employed. The artificial intelligence that is being proposed would be something that uses a set of requirements to generate a database design automatically or semi-automatically. It would incorporate rules and heuristics used by human designers and apply them in a systematic way to each design task (Storey, Goldstein, 2013). Most AI systems will feature knowledge-based approaches that facilitate the use of heuristics, learning from experience, domain-specific knowledge, and predetermined choices of design strategy (Storey, Goldstein, 2013). There are a few knowledge-based tools being used for work and study currently on the market now outlined by Storey and Goldstein. The intelligent interview system (I2S) is a way of interviewing the end users and storing information as it interviews more people and learns from past interviews. This will be useful in the first step of the design process, requirements specification. Its main features include the ability to build and store information, use a natural language (English), and ability to create an 'attention list' that can be discussed in person after the interview. The view creation system (VCS) is a tool that allows a user who does not know anything about database creation to place predefined inputs into the system and create an entity-relationship (ER) diagram that moves the user to the logical design phase of the database design steps. The system is supposed to detect and resolve inconsistencies, ambiguities, and redundancies in the user input. For example a user is not allowed to enter the same value for two or more records in a primary key field. If a user does, the VCS would be able to locate the offense and offer advice on how to remedy it. Another useful tool is the Modeller, which helps to produce an extended ER diagram. It is able to help select primary keys from candidate keys, and also explode aggregate attributes, such as addresses, into their component parts (e.g. street number, street name, street type, etc.). There are

many other tools that assist the user in database creation and it is only a matter of time before a more robust system is designed that can design and implement a database with complete autonomy.

Sustainability Reporting and Assurance

One of the most important parts of designing a database is making sure it functions correctly. Not only does is it supposed to fulfill the user's needs, but also fulfill all the requirements of the system. There are many ways to go about testing your database to make sure it works correctly, but this section will cover a few major forms of testing as it relates to general software packages. The testing methods here should be a good beginning to validation of most relational databases, and is a good introduction to the different schools of thought when it comes to database testing.

The three main types of testing that will be covered in this section are exploratory testing, acceptance testing, and scenario testing. Exploratory testing is a form of testing that does not rely on detailed pre-specified test cases, but rather allows for simultaneous learning, test design, and test execution, in an open environment (Itkonen, Rautiainen, 2005). Acceptance testing is a written test used to determine if the system is exhibiting the expected results when you perform certain actions on the system (Miller, Collins, 2001). Scenario testing is a way of testing in which the tester follows a prescribed set of steps deemed necessary to verify and validate the software. Verification of the software comes when the software meets the predefined specification of the software, while validation of the software means it fulfills its intended purpose (Ryser, Glinz, 1999).

Exploratory testing can be a useful form of testing in the database development process because it is a means to keep testing software after executing the scripted tests and avoid investing effort carefully designing test cases when the software is still in the early unstable stages (Itkonen, Rautiainen, 2005). It is also optimized for finding defects in the software, a crucial part of the database design process (Itkonen, Rautiainen, 2005). In order to know if the testing process being employed is an exploratory test, a few criteria must be fulfilled: "(1) tests cannot be defined in advance, (2) the test is guided by results of previous tests and gained knowledge from them, (3) the focus is on finding defects, (4) there is simultaneous learning of system under test, and (5) the effectiveness of testing relies on the tester's knowledge, skills and expertise" (Itkonen, Rautiainen, 2005). These criteria are employed when one follows a predefined approach to exploratory testing called the Session-Based Test Management (SBTM). The SBTM is an approach to planning, managing, and controlling exploratory testing in short, fixed-length sessions (Itkonen, Rautiainen, 2005). The SBTM also uses a charter sheet that defines a mission for testing and works as a guideline for testing. Following this procedure can help one reap the benefits of exploratory testing. Some of the benefits of exploratory testing are the ability to increase the effectiveness of testing in terms of number of found defects, the opportunity for simultaneous learning of the system, the ability minimize preparation documentation before executing tests, and the rapid flow of feedback from testing to both the developers and the testers (Itkonen, Rautiainen, 2005). If one would like to fully utilize the benefits of exploratory testing there are a few ways it can be done. There is the functional testing of specific individual features. There is also exploratory smoke testing, which is there a 'head level' list of areas to be tested is used (Itkonen, Rautiainen, 2005). There is exploratory regression testing of small portions of the

system (Itkonen, Rautiainen, 2005). There is subcontracted testing of the systems by experienced professional users of the system who may be more inclined to find the bugs a developer might not find (Itkonen, Rautiainen, 2005). The final way is fairly simple, freestyle exploratory testing, in which the tester is free to test however and wherever he or she wants (Itkonen, Rautiainen, 2005).

Acceptance testing can be especially helpful because it is a way to capture user requirements in a directly verifiable way, and measures how well the system meets those requirements, as well as provide an idea of how finished the database is (Miller, Collins, 2001). It is way to gauge the completion of the system because percentage of the acceptance tests that pass is an easy rough metric of what percent of the system is finished. Acceptance tests, which can be automated or manual, are very dynamic, and able grow as the system grows by capturing user requirements as they evolve (Miller, Collins, 2001). As far as who should write the tests, it should be the business side of the team, which in the case of the database would be the QA personnel or the business analysts (Miller, Collins, 2001). As far as when the test should be written, it should always be written before the system is complete, as it can cause miscommunication between the administrators and the user (Miller, Collins, 2001).

Scenarios, which are sometimes call use cases, are used to describe the functionality and behavior of a system in a user-centered perspective (Ryser, Glinz, 1999). They can be used to derive test cases for system test, where validation and verification are generally recognized as two vital activities in developing a system, as previously mentioned (Ryser, Glinz, 1999). These

are generally useful because they run through the system from an external perspective since a scenario is an ordered set of interactions between partners, usually between a system and a set of actors external to the system. An actor is a role played by a user or an external system interacting with the system specified (Ryser, Glinz, 1999).

Design

This section will cover the design of our project, including the requirements, specifications, constraints, and steps. It will start from the beginning and go to the end explaining the theory and logic behind the decisions that were made.

Illumina Requirements

Illumina's mission to create a full sustainability report for the public has led to the necessity for a system to gather data. The system Illumina requires must be able to store large amounts of data points in an organized fashion. The company anticipates that there will be upwards of two thousand data points going into the system, so the system must be able to store and manage a lot of information without becoming sluggish or unresponsive. The requirement for organization of the data also becomes crucial due to the room for error when entering large amounts of data. Another functionality requirement is that an integral part of the sustainability reporting process is the ability to manipulate the data that is stored in ways that will help the process of reporting. The system will have to have the ability to input data, edit the data, and delete the data. As well as these three functionalities the user must also have the ability to filter the data in ways that will help them to tell the story of the data. On top of the main functionality, there must be a user-friendly interface that is also fool proof. As previously mentioned in the literature review, it is important not only for the user to become proficient in the use of the software, but also for them to accept it and find satisfaction in its use. The first step of that process is to make it easy for the user to work with the program.

As the data is collected, it will become important to have ways to visualize the data and be able to tell a story with it. Illumina would like to visualize the data is in the form of graphs. These graphs will feature comparisons between the various sustainability metrics and their respective regions, locations, and years. In the creation of the graphs Illumina requires they have dynamic axes first and foremost. Illumina anticipates there will be a need for a great many graphs, so it is crucial that there is a way to use as few graphs as possible to tell the story. Along with the dynamic requirements of the graphs, Illumina wants a very easily expandable system for graphing in the anticipation that more graphs or different types of graphs will be necessary in the future. This means that there must be a well-defined process for manipulating graphs and creating new ones that are consistent with the existing graphs.

Since the senior project group will not be working for Illumina indefinitely, Illumina would like an encompassing system that is also easy to maintain and expand for any future users and administrators. There must be documentation to explain to future users how to use the program interface and work with its back end, in whatever capacity that may be, as well as use existing framework to add on to the program should the scopes change later on. It is clear from the beginning that this will call for a user guide of some sort to help explain how to use the system and how to manipulate it.

Knowing that the system will have to be used by future employees without the help of its creators, the system must be designed to last. The user guide should help with that, however it is crucial the system is clean of bugs, and that minimal maintenance will have to be done to keep the program up and running. It has also been outlined by Illumina this program, if successful,

may be the future prototype for a move to a more professional database system provided by companies such as SAP and Oracle. The program could save Illumina a lot of money, time, and other resources in the future if the system the project team creates is well thought out and easily translatable to other platforms.

Approach

The following are the steps the project team followed in order to solve the problem. The steps include the reasoning and justification for the decisions that were made in order to deliver a product to Illumina that meets all of the requirements.

Database Alternatives

There were initially fast approaching deadlines for data collection when the project began in November. Illumina wanted to have a majority of the data collected system completed by the end of December. The brainstorming for how to create a system that would meet all the requirements began immediately. The system has to be able to handle a large amount of data, must have an easy to use interface, be able to filter and organize data, and represent the data graphically. On top of that, Illumina expressed interest in a system that is well-known as it will have to be a system that many stakeholders can access on the internal network of Illumina. After intensive research, four alternatives were evaluated; Microsoft Excel, Microsoft Access, AxisBase, and LibreOffice Base.

Microsoft Excel is a very familiar program for the project group, so this was the first option that was evaluated. The program has qualities that satisfy some of the requirements of Illumina, but there are also some shortcomings in its design. The system can handle very large inputs of data; Microsoft Excel can handle up to 20,000 rows of data in one sheet, which is far more rows of data than is anticipated to be needed. The interface is generally pretty well known by most users, which makes it easier to learn and teach the program. The program can also be manipulated to look the way it has to. There are also very many resources online on how to use Excel. However, it does not meet the requirement of being able to filter data and organize it very easily. There are sort functions for the data; however they are inadequate for the level of complexity that Illumina requires. For these reasons, the project group ruled Excel out as an option moving forward.

After reevaluating the options that could meet all the requirements, it was determined that a formal relational database model would be the most appropriate option. In the interest of exploring all the options three major database systems were evaluated; AxisBase, LibreOffice Base, and Microsoft Access. AxisBase is free software with a simple user interface that offers options for visual database manipulation while also providing the opportunity to hard code the database using the SQL language. LibreOffice Base is another free software that performs all the same functions as AxisBase, however has a slightly easier to use interface, as well as offering linkages to other more well established programs such as Microsoft Access. However, neither of these databases are very well known, or used, in industry which precludes it from meeting the requirements set forth by Illumina.

Microsoft Access was chosen because it was the only program that met all the requirements of Illumina. First of all, Microsoft is a very well-known and established program across industry and academia. Illumina already has Access preinstalled on all of its employee’s computers, which allows everyone to open the program without the risk of installing new software on the internal system. Also, the fact that it is such a well-known program lends to its ease of access to information online. There are countless forums and help websites devoted solely to using Microsoft Access. Not only is it a very well-known program, but also a very well developed one. It can store massive amounts of data points, and in a very well organized fashion. The program offers the ability to filter data through queries, and organizes each data point in a system of records that are linked between data, making it very easy to relate the sustainability metrics and their respective regions, locations, and years. Access also has a very easy to understand system for designing a user interface for inputting data points. With all of these features, it was no contest that Microsoft Access would be the best option to achieve the goal of a robust system for managing the sustainability data.

Building the Database

The first step after having chosen Microsoft Access as the best option for the database was to build a prototype. With a limited knowledge of how to use Microsoft Access, the project group set out to build a rough version of the product. The initial focus was creating a simple system that would accept the inputs into a very user friendly interface. The initial interface had to have about 30 boxes for inputs, so it was important they be well organized and easy to navigate. When designing the interface the focus was on making sure all of the titles, captions, and text boxes

were aligned correctly, colors chosen to match the classic Illumina gray and orange, and sized according to heading level (see Appendix A).

The back end was designed to account for the necessity of multiple tables for the data that needed to be related in order to make reporting easier. An initial database design was created with minimal queries, tables, and a single form. To promote usability all objects in the database were named using a system of antecedents and other naming conventions, in the anticipation that there would be many changes and additions necessary in further iterations of the database (see Appendix B). The project team wanted to make it easy to delete, add, and rename objects in the future.

In order to continue to develop the database toward a finished product, weekly meetings were held between Illumina and the project team. The weekly meetings provided feedback to the project team on what changes were necessary to keep moving forward. As the database was shaped the team needed to go through many short-term iterations along the way. What these iterations did was help build and improve the product step by step. Hence the defects can be dealt with at early stages. The iterative model also allows for less planning and more active problem solving. As more and more iterations were cycled through, quality increased, as well as productivity. Moving toward the end of the development process of the database, designs were finalized, and the scope frozen. At this point only minor changes would be accepted.

Graphing

Along with the task of gathering the data, the visualization phase brings a sense of understanding to the data. The data can be stored in Access; however, in order to make sense of it, Illumina required graphs. The reason graphing is so crucial is because the intent is to use the graphs to publish in company-wide internal reports on monthly sustainability metrics, as well as to use internally for the HSE team to try to improve. Two alternatives were evaluated in the graphing process; Microsoft Access and Microsoft Excel. Of the two alternatives, Microsoft Access was chosen first, simply because it already contained the program's data. This way there would be no data loss that would come from switching over to another program.

Microsoft Access has very dynamic reporting and graphing features that are very easy to link to various queries and tables. The graphs allow the user to create graphs that can easily be switched between different locations, regions, years, and metrics. This makes it very favorable because it allows one to switch the data in the graphs and avoid making countless graphs that are similar. However there were a few pitfalls when using Microsoft Access for graphing. The most concerning issue is the lack of customization when dealing with the aesthetics of the graphs. There are not many options when it comes to coloring, as well the styles of graphs possible are very limited. The problem with this is that Illumina would like to publish these graphs in monthly reports; therefore they need to look appropriate for company-wide viewership. Even if the company did want to use the graphs, the graphs do not export well to other formats, thus making them essentially useless. With that said, the project team did not move forward with the graphing functions in Microsoft Access.

The next logical option was the graphing abilities of Microsoft Excel. Excel features very dynamic functionality pertaining to graph style. The stylistic customization made it viable to use Excel for the project. Another positive aspect of Excel is that graphs export very well to other programs that could be used to make publications, such as Microsoft Word or Microsoft Publisher. Not only are the graphs very easy to style, they are also very dynamic if they are linked to the correct locations in the spreadsheets. One kind of chart placed on a sheet can be linked to a range of cells, and when the value of that cell changes, so to does the values in the graphs. Another plus for the team is that Excel can be a very powerful tool when VBA is employed in the background. An example is how chart axes titles work for charts that are based on a single location. Instead of making a new graph for each location, the graph can be linked to a set range of cells, and the titles for the chart can be coded using VBA to match the input in a cell (see Appendix C)

There are a few negative aspects of using Microsoft Excel for graphing however. Small inconveniences like the fact that scaling the axes, should one location be drastically different than another, is a bit of a difficulty. However, this is a minor inconvenience compared to the problems faced with Access. Perhaps the most dubious reason to doubt Excel is the fact that it is a separate program from Access. The changeover of data from one program to the other presented a bit of a challenge, however the translation of table data in Access queries to spreadsheets in Excel allowed for a solution. The data would be queried in Microsoft Access, using the graphing tools that the team has built in to the database, copied by the user, and pasted into the correct location in Excel. This way all the data can be transferred for Access to Excel with minimal risk of error. Once the data is in Excel it must be sorted, so a button that sorts the

data automatically once it has been pasted in can be pressed by the user causing a chain reaction. The VBA code from the button sorts the data (see Appendix D), which in turn cause the graphs to update to the appropriate values.

User Guide/Admin Guide

Knowing the database and graphing utility will need to be passed on to the next users and administrators once the students leave, Illumina sees a need for a user guide and an admin guide. The purpose of the user guide is to teach the user guide how to interact with the database on a surface level. It is meant to be a one stop resource for users trying to add, update, and delete data. It has explanations of how to input and edit data in bulleted lists that reference diagrams of what the user guide is trying to portray. This is the best way to help the user with any issues they may face along the way. The admin guide is a resource for database administrators to learn how to work with the tables, queries, and code in the background. The way this guide was set up was mostly focused around scenarios. This model was chosen because it was determined there are only a few scenarios in which the administrator will need to edit the functionality in the background. For that purpose, the scenarios show the administrator how to do the task they need to, but nothing more, in order to protect the system from human error. There are also explanations of the code for the administrator to use and possibly learn from for the future (see Appendix D).

Methods

After the database was created, the project team applied different methods of testing the system to make sure that it was error-proof and properly met stakeholder expectations. The first method was error testing by the team and the second was end-user testing. The team began by identifying different types of errors that the user could potentially encounter while using the system. The possible errors included being taken into the back-end code of the database, receiving an error message from the software, a button not working or working incorrectly, the instructions on the form not being easy to understand, and etc. Once identified, solutions were brainstormed for the potential errors and were incorporated into the system in many different ways. The first was by putting error-proofs in the code in which the code "stops" once an error is identified. In this way, the user does not ever know that an error has occurred and does not have to be bothered with the annoyances of the system. Also, error messages or pop-ups were incorporated into the system. For example, if the user picks an incorrect combination from the drop-downs on the form, then a message will appear asking them to change it (see Appendix E). Both of these ways make sure that the user does not encounter parts of the code or any aspect of the database that is beyond their knowledge level.

The second method was error-testing by the end-users. The team had weekly meetings with the Health & Safety team at Illumina along with representatives from the different world regions. Before the meetings, the database was sent out to all of them and they were asked to test the forms and tables that would be used on a daily basis. Then, during the meetings, they provided the team with feedback in terms of areas for improvement or features that had to be changed. The team then took this feedback and incorporated it into the database using the different methods

described above. An important aspect that the team had to keep in mind is that although the user provided good feedback, it could not always be applied properly in parallel with rules of database design. The team encountered situations in which the users had feedback that was either impossible to incorporate into the database or was completely unnecessary. Therefore, the team realized that in end-user testing, it is important to properly judge which aspects of user feedback can actually be used in a value-added way.

Results

The team successfully completed the project with most of the results meeting expectations. At the beginning of the project, the team set out to achieve the set of expectations Illumina outlined and the team's own goals. The success of the project was defined by the quality with which the team met these expectations and goals.

The first goal was to design a good quality and efficient database that had a user-friendly front end form with a sturdy back-end design. This goal was achieved after creating multiple iterations of the database through thorough testing and applying user feedback. The final design withstood all user errors and shows custom pop-ups when an error does occur. This ensures that the system is constantly working and does not cause the user any extra stress. The second goal was incorporating all of the features that Illumina had asked for. This was achieved during the database design phase where the team had these features as the foundation for which the tables and queries were based on. The form design was straight-forward and allowed the user to access exactly what they are looking for. The third goal was keeping in constant communication with the company and receiving their feedback in each step of the way. The team felt it was extremely important to deliver a quality product and the best way to do this was to show the company their design through each step of the process.

Additionally, during the process of completing the project, there were some unusual conditions that the team encountered. For example, Illumina had never before done data collection of sustainability metrics. Therefore, they only had a broad idea of their expectations from the project. The team had no foundation for a database to go off of since Illumina had not previously

used one. In this way, there was no benchmark for the database and the team had to define brand new standards. Also, since the company's expectations were vague, they remained dynamic in that they constantly changed through the project phases. Illumina started with a broad sense of what they were looking for but as the project went on, they became more concise in their expectations. By the end of the database creation, the team and the company both had a good view on how the database should be designed and its reasons for utilization.

The project was mostly completed through Microsoft Access. In theory, this software is best used for small to medium sized data storage. Currently, Illumina is collecting data for only a small number of years but eventually, they will expand which will mean that a large amount of data will need to be stored in the database. This progression from collecting small amounts of data to large amounts should be aided by purchasing a third party database software. The team's current model and database design is a good foundation on which the new software should be based. In this way, the fees for purchasing the software will also be reduced since a bottom-up design will not need to be done. Also, the team is unsure of exactly how long the company will use the database that they created.

Some questions remain unanswered and will only be answered with time. For example, the database is currently stable but as time goes on and it is utilized more, there are risks (as with any software) that it could break down. The team created the user and administrator guides but unless they are properly used to constantly maintain the database, it could incur bugs and stop working. Unfortunately, the team cannot help the company to continuously manage this and the company will need to designate a system administrator to make sure the system is properly maintained.

Also, many legitimate problems might crop up after the team is finished with the project and submits it to Illumina. First, as stated before, they might not be able to completely maintain the system. Although they have the user and administration guides, situations could arise in which the solution to a problem is not stated in the guide. In this case, they must find another answer to the problem or eventually stop using the database entirely if enough major problems persist. The system administrators also do not have all of technical knowledge to properly manage it (in terms of the back-end coding that was done).

Finally, there are many predictions that the team has for the future of the system based on the results that were seen. First, this database will allow Illumina to drastically improve their sustainability tracking. It will enable them to gather and output the data in an organized manner for analysis. Eventually their company-wide environmental performance will improve since Illumina will be able to identify and implement areas for advancement. Next, the database system will save the company money and time when a larger third party system is invested in. They will be able to easily translate the team's database design and information into the new software which will decrease the amount of money that is spent on that system.

Conclusion

The project was successful as seen through the end results. First, the database is dynamic in that it allows Illumina to manage the system and add to it as the data collection process grows. The system also met, and in some ways exceeded, all of the stakeholder expectations. The accompanying guides for the database also allow the administrators to properly maintain the system and continue to make it easy for the user to utilize. Second, the team learned how important it is to be flexible. Since the MS Access reporting functionality was inadequate, the team created an Microsoft Excel document to supplement the database. Although the team did not achieve the reporting functionality goal as they had originally planned, it was still done well and exceeded the objective. Finally in terms of soft skills, the team learned how to cross-culturally collaborate and communicate. As stated before, the team had weekly meetings with different region representatives in EMEA, AMR and APAC. Each of these regions had different business cultures and methods of operation. Being able to collaborate with these representatives allowed the team to gather skills that they can carry forward with them in industry.

Throughout the phases of the project, the team also learned about database design and business goals in general. For example, the final database was made after multiple iterations of the database were created and analyzed. In this way, the team learned that system design is based off testing and a proper foundation rather than correctly making it on the first try. Also, by utilizing the graphing functionality of Microsoft Excel, the team realized the importance of testing out functionalities of different software before picking one. It is necessary to properly understand how different software works and interact with one another, before settling on one way of accomplishing a goal. Next, the graphs were difficult to read in terms of the units and numbers. It

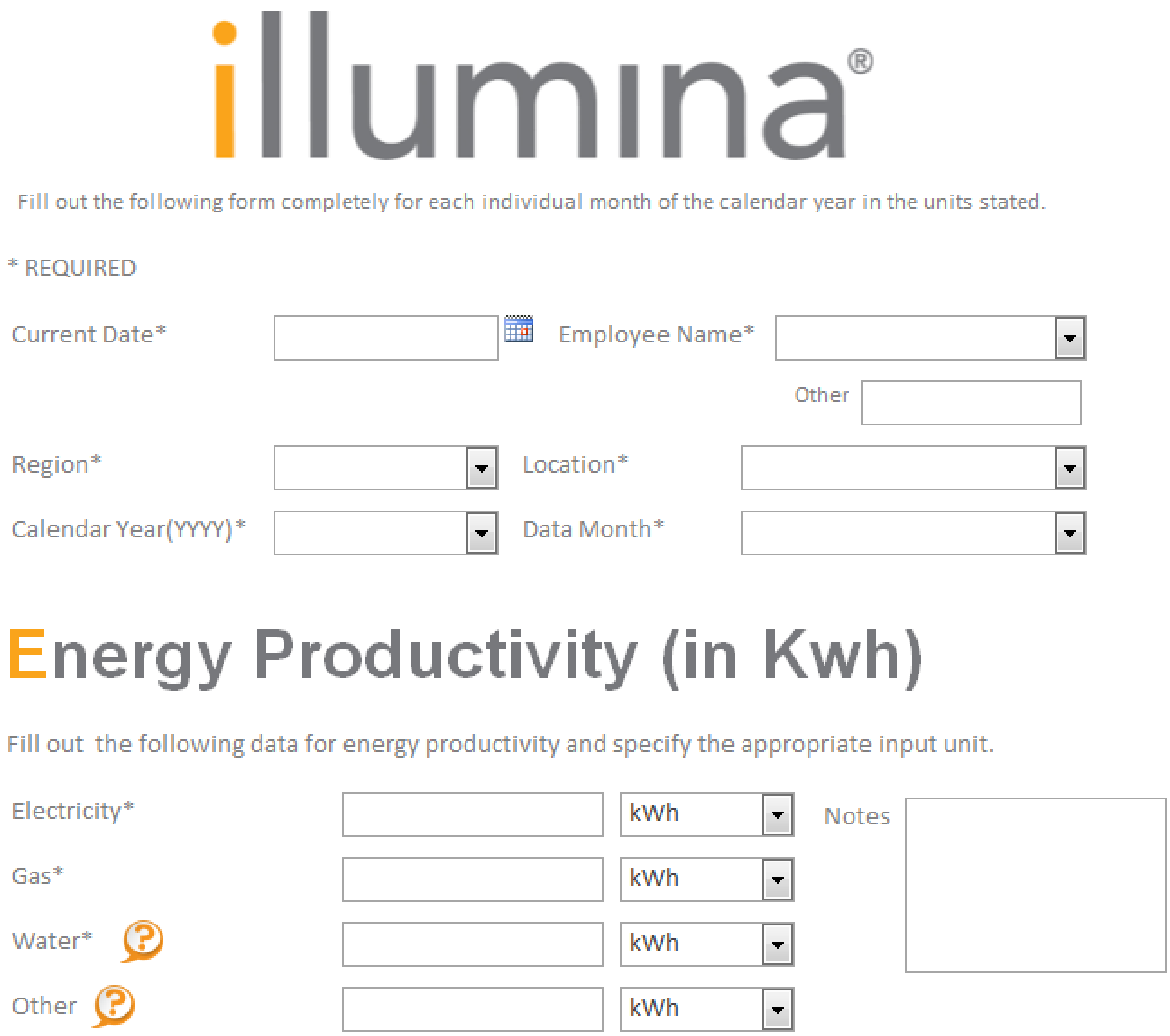
is essential to create a graph that can be easily read and analyzed by the user. In a graph, it is extremely easy to be misguided based on the units of the axes or shape of the x & y coordinates. Therefore, designing a graph that can be easily analyzed is essential to good reporting. Finally, it is necessary to constantly keep stakeholder expectations in mind while designing the database so that re-work is not necessary.

The project was completed well but as with all projects, there are always areas of impovement. If the project was to be re-done, there are many aspects that the team would do differently. First, the stakeholder expectations should be more clearly defined from the beginning of the project rather than being dynamic through the phases. Next, the team would explore the software more in-depth and thoroughly understand it before starting the design phase of the database creation.

In conclusion, the team believes that the project had a successful impact on the Illumina's process for tracking their sustainability metrics. The database created a long-standing central location that will be used company-wide to track, analyze, and report on Illumina's environmental impact.

Appendix

Appendix A: Userform


illumina®

Fill out the following form completely for each individual month of the calendar year in the units stated.

* REQUIRED

Current Date*

Employee Name*

Other

Region*

Location*

Calendar Year(YYYY)*

Data Month*

Energy Productivity (in Kwh)

Fill out the following data for energy productivity and specify the appropriate input unit.

Electricity* kWh

Gas* kWh

Water* kWh

Other kWh

Notes

Waste Productivity (in metric tonnes)

Fill out the following data for waste productivity and specify the appropriate input unit.

Mixed Recycled Material*		tonnes
Hazardous*		tonnes
Landfill*		tonnes
Diverted*		tonnes
Incinerated		tonnes

Notes

Individual Waste Streams: If your recyled material is sorted into individual waste streams, please provide data for the specific streams below.

Cardboard		tonnes
Mixed Papers		tonnes
Metal		tonnes
Aluminum		tonnes
Steel		tonnes
WEEE Products		tonnes
Glass		tonnes
Wood		tonnes
Polystyrene		tonnes
Plastics		tonnes
Polyurethane		tonnes
Batteries		tonnes
Fluorescent Bulbs		tonnes
Compostable		tonnes
Other		tonnes

Water Productivity (in m^3)

Fill out the following data for water productivity and specify the appropriate input unit.

Total Water*		m3	Notes
Potable Water		m3	
Reclaimed Water		m3	

Submit

Close Form

Clear Form

Appendix B: Naming Conventions

- Tables
    - Descriptive Phrase + “T”
      Ex. EnergyDataT
- Queries
    - Descriptive Phrase + “Q”
      Ex. MonthlyEnergyQ
- Forms
    - “CSR_” + Descriptive Phrase
      Ex. CSR_Form
- Graph Queries: these are the particular queries used to generate graphs in the sister excel program
    - # + “I”/”R”/”L” (Illumina/Region/Location) + Type of Data + Time Period + “Q”
      Ex. 10_R_CategorySums_MonthlyQ
    - This means it is the 10th graph, showing monthly data for the energy, GHG, waste, and water broken down by region
- Labels
    - “lbl” + Descriptive Phrase
- TextBoxes
    - “txt” + Descriptive Phrase
- ComboBoxes
    - “cbo” + Descriptive Phrase
- Command Buttons
    - “cmd” + Descriptive Phrase

Appendix C: Different Location Graphs

Appendix D: VBA Code Explanation

Button Coding

This section will show an explanation of the code that accompanies the sort buttons on each sheet

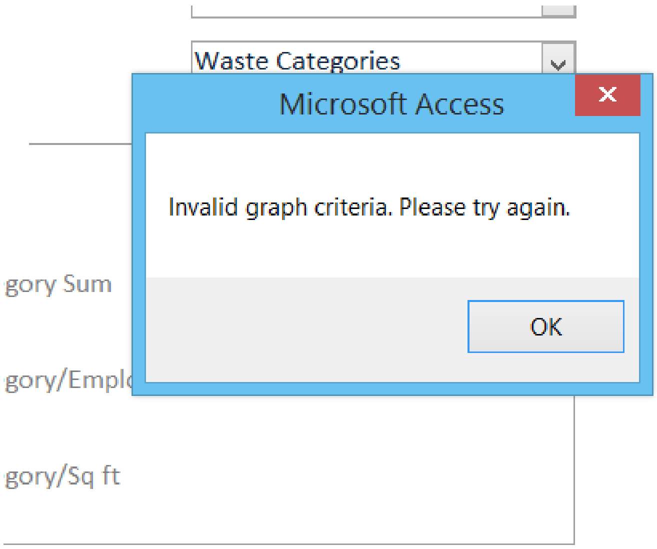
- You must make sure each sort button has an assigned macro by using the following procedure
  - **Right-click** the *Sort* button → **Click** *Assign Macro* → **Select** the appropriate macro
- The following code explains how the columns are sorted using the Sort button

```
Sub combo1_Click() 'The name of the button on the sheet

    Range("A1:F25").Select
    'Select all the data including titles

    ActiveWorkbook.Worksheets("1_I_CategorySums_Monthly").Sort.SortFields.Clear
    'Unsort all the fields in that range

    ActiveWorkbook.Worksheets("1_I_CategorySums_Monthly").Sort.SortFields.Add Key:=Range("B2:B25") _
        , SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    'Sort the field in ascending order (in this case it is the year)
        'The name '1_I_CategorySums_Monthly' should be replaced with whatever the name of the sheet is
        'The range 'B2:B25' should be replaced with the series of data you want to sort

    ActiveWorkbook.Worksheets("1_I_CategorySums_Monthly").Sort.SortFields.Add Key:=Range("A2:A25") _
        , SortOn:=xlSortOnValues, Order:=xlAscending, CustomOrder:= _
        "January,February,March,April,May,June,July,August,September,October,November,December" _
        , DataOption:=xlSortNormal
    'Sort the field in ascending order (in this case it is the month)
        'The name '1_I_CategorySums_Monthly' should be replaced with whatever the name of the sheet is
        'The range 'A2:A25' should be replaced with the series of data you want to sort

    With ActiveWorkbook.Worksheets("1_I_CategorySums_Monthly").Sort
        .SetRange Range("A1:F25")
        .Header = xlYes
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
    'Not entirely sure what this does but it is necessary

End Sub
```

- The following code explains how certain chart titles are automatically updated to reflect unique regions or locations

```
With Worksheets("6_I_EmplCategorySums_Yearly").ChartObjects(1).Chart
'Select the chart you would like to work with

    .HasTitle = True
    'Setting the fact that the chart has a title to true

    .ChartTitle.Text = "Illumina Energy Consumed Per Employee" & vbLf & Cells(2, 1).Value & " vs " & Cells(3, 1).Value
    'You must use this syntax. 'vbLf' means 'return'.
    'Cells(2, 1).Value' represents the value in the cell you would like to reference

End With

With Worksheets("6_I_EmplCategorySums_Yearly").ChartObjects(2).Chart
    .HasTitle = True
    .ChartTitle.Text = "Illumina GHG Produced Per Employee" & vbLf & Cells(2, 1).Value & " vs " & Cells(3, 1).Value
End With
```

Appendix E: Error-Proofing Example



Bibliography

Chan, Hock Chuan, Kwok Kee Wei, and Keng Leng Siau. "User-Database Interface: The Effect of Abstraction Levels on Query Performance." Mis Quarterly 17.4 (1993): 441-64. Print.

Itkonen, Juha, and Kristian Rautiainen. "Exploratory testing: a multiple case study." Empirical Software Engineering, 2005. 2005 International Symposium on. IEEE, 2005.

Leitheiser, Robert L., and Salvatore T. March. "The Influence of Database Structure Representation on Database System Learning and Use." Journal of Management Information Systems 12.4 (1996): 187-213. Print.

Manetti, Giacomo, and Simone Toccafondi. "The Role of Stakeholders in Sustainability Reporting Assurance." Journal of Business Ethics 107.3 (2011): 363-77. Print.

Miller, Roy, and Christopher T. Collins. "Acceptance testing." Proc. XPUniverse(2001).

Ryser, Johannes, and Martin Glinz. "A scenario-based approach to validating and testing software systems using statecharts." Proc. 12th International Conference on Software and Systems Engineering and their Applications. 1999.

Searcy, Cory, and Ruvena Buslovich. "Corporate Perspectives on the Development and Use of Sustainability Reports." Journal of Business Ethics 121.2 (2013): 149-69. Print.

Storey, Veda C., and Robert C. Goldstein. "Knowledge-Based Approaches to Database Design." Mis Quarterly 17.1 (2013): 25-43. Print.