COURSE SUMMARY

This is the final week already! We hope that you have enjoyed the database development course and that you have enjoyed learning about the various theoretical and practical aspects involved in database development, from developing an understanding of client data, to the design of a representation, the iterative steps in this process of design, to the initial construction of the database itself, through to fine-tuning and the steps of normalisation etc., indeed, if a relational database is the right option, or perhaps you might have opted for a noSQL database, who knows! …

You should use this week to read through the below, and reflect on the various things you have learned. Please feel free to refer to some of the material you have already processed – there are several separate subjects covered within this database development course, which is challenging, and you should give yourself every opportunity to reflect on the material, and re-visit some of it as you progress.

As this is the final week there is only this course summary, there are no specific learning outcomes. 'revision' week therefore consists of a course summary.

Good luck with you future database endevours!

Table of Contents

WEEK 12: COURSE SUMMARY

12.1 Week 1: Introduction to database systems

We introduced the idea of data and the potential for the vast amounts of data that are available to be exploited, i.e., converted into useful information. We distinguished between structured data unstructured data and mentioned that structured data is far easier to exploit, computationally, compared with unstructured data.

We discussed how structured data organization can take different forms and, typically, that most data within organizations is organised according to some form of file system storage. Issues, such is the duplication of data, it's scalability, accuracy and integrity are examples of limiting factors related to file-based storage, and which motivates the use of databases.

We noted the relational databases exist alongside other database types (e.g., hierarchical, object-orientated and noSQL), but that relational databases remain the most widely used database technology today. Furthermore, relational databases are relatively easy databases learn when compared with some of the other database systems in existence. Most of the database development course was therefore dedicated to relational databases.

12.2 Week 2: Data organization and some basic relational ideas

We learnt that database development should not just be undertaken by jumping straight into creating a database. There is a very important process of design, which must take place. Using examples, we considered relationships between datatypes. We also highlighted the problem of unique access to data and, generally, future changes to data should only add data to an existing, designed data structure, rather than change the data structure itself. We introduced the relational approach to tackling such issues and listed some basic relational database definitions.

We then considered real-world file-based dataset known as Code Point Open. The reason we did this, was to develop an appreciation for an existing File-based system, and the often-used comma separated values (.csv) format. Very often a database developer will be asked to move an existing set of data into a database format. While considering this existing data set, we made a brief critique of how it is organised – looking at an existing system with a critical eye is a skill that all database developers will need to develop.

We then considered how we might write a program to access the data within such a system. This is an important consideration because it is one of the key motivations the creation of database systems, to make a access to data much easier in terms of the programming interface. This led us to the important distinction between declarative programming and algorithmic programming. Query languages are very much based on the idea of declarative programming. There is no need to develop complicated algorithms to retrieve data. If the database is well designed, relatively simple declarative statements are enough to octane the data required.

12.3 Week 3: Relational data modelling I

The term, 'modelling' is very widely used, in everyday life, but also heavily used within the computational sciences. We thus explored the meaning of models and the modelling process. This allowed us to specifically define and discuss 'data modelling' and 'database modelling' in the context of relational databases. We considered database development as a sequence of iterative steps involving conceptual modelling, logical data modelling, followed by physical data modelling. We introduced some standard visual diagramming techniques related to relational database modelling. A key purpose of these diagrams is to relate specific entities in the modelling domain to others, using different times of relationships one-to-one one-to-many many-to-many etc.

We then introduced the language of set theory, which is the mathematical underpinning of the query language of rational databases, known as SQL. We also then covered some definitions from set theory, which are, at the same time keywords within the SQL language, demonstrating the importance of the former to an understanding of the latter.

12.4 Week 4: Relational data modelling II

We learnt, in more detail, features of entity relationship modelling, also offering an alternative diagrammatic notation. Continuing the logistics example highlighting previous weeks we arrived at a simple design for a logistics database.

We then moved on to some of the practical issues regarding the creation of tables for a relational database designer. In contrast to the simple representation tables, for example, as represented in previous weeks as a spreadsheet boring set of comma separated value files, the specific datatypes must be explicitly declared when creating tables. We highlighted some of the datatypes available in the MySQL implementation of SQL.

We then discussed primary keys and synthetic keys before returning to the issue of creating relationships between tables, using link-tables, to specify different kinds of relationships between tables/entities. Again, this was done by drawing on the logistics database design.

12.5 Week 5: Normalising relational data

We covered the topic of Relational data optimization, aka ‘normalising relational data’. We provided some textbook definitions of the procedure of normalization, which is a ‘bottom-up’ approach to the design of the data structures.

Several ‘anomalies’ we discussed: the ‘update’, ‘insertion’, ‘deletion’ anomalies, before the sequential steps in the process of data optimization were considered. These consist of the process of moving from and on normalized database to one that corresponds to first normal form, then the second normal form, ending (for our purposes) in third normal form: UNF – 1NF – 2NF – 3NF.

We then covered the concept of functional dependency by firstly defining relations more formally and, again, regarding some of the concepts available within the underpinning language of sets.

We then went through an example scenario concerning a fictional company (G&P Ltd). However, the scenario was intended to provide you with I more pragmatic feeling for the process of normalization, and to indicate to you that the process of normalization is, in fact, are highly pragmatic process, which needs to be undertaken, to produce workable real-world databases. Each step in the normalization process was therefore exemplified.

12.6 Week 6: Database management systems and the MySQL RDMS

Introduced the idea of database management systems. Several general properties of database management systems were covered, relating to the *internal* level, the *conceptual* level, and *external* level. Relational database management systems were that introduced, in terms of the more basic desktop applications available, but also the various industrial-strength relational database management system available by company vendor.

We then summarized some of the software that is relevant to the current course. For your practical sessions, the configurations/ installations of such software will have been completed. However, as a budding database developer it is expected of you within industry that you have a good knowledge not only if the theory of databases, but the pragmatic aspects of database development also, which includes being familiar with web service existing database engines, existing database GUI tools, and other relevant technologies, such as languages used for server side programming and languages used the clients are programming and, of course, the related software environments.

Finally, we presented a brief introduction to MySQL workbench, a GUI for MySQL databases.

12.7 Week 7: DB development essentials: The AMP stack and MySQL querying

In week seven we covered a topic related to web applications, and the various software tiers relevant - ‘client’, ‘web’, ‘business’, ‘data’.

Following on from last week, recovered something known as the AMP stack and the various options available. We briefly covered the main steps of the AMP stack installation flow.

Following this introduced SQL querying with the number of Basic MySQL commands to create a database, create tables with a database, select, sort data, and join data tables together.

12.8 Week 8: Server-side database programming with PHP

We learned about the PHP programming language, both is a server side scripting language but also as a programming language inside right. We looked at the structure PHP has an embedded within HTML, the main use case considered on this course, and then returned to the topic of a database developer’s software environment, i.e. the depiction of the local machine as localhost, web server, etc.

We described the required configurations PHP, to access the mysqli Library. Variables and datatypes in PHP were introduced, along with the *array* datatype and the *associated* array datatype.

Various functions within PHP then summarized, before a set of PHP examples were covered, each one building on the last, to produce different kinds of output to the browser. However, the name lessons from this week I’m not related to the simple PHP examples provided, but an overall understanding a PHP, how to install it, how to connect to database PHP, then to start doing basic programming within the environment described.

12.9 Week 9: Client-side database programming with Java

We learned about those aspects of the java programming language related to MySQL programming. In order to appreciate clients are programming, we discussed my client side logistics program to set the background.

We then focused on Java generally, highlighting important features other typical Java project - i.e., project folder, 'source' and 'build' directories, 'main' and 'test' directories etc. we then covered the steps involved in Installing the MySQL connector, and adding this to the classpath of a Java project.

We then covered a number Java examples, which led to a very simple application to run a number of MySQL queries on an existing database and output the results from that database to the command line / console.

Again, as with the previous week (Week 8), the purpose of this week is to provide you with practical skills, which enable you to begin Java development that involves the use other database, not to provide you with an in-depth min-course on Java.

12.10 Week 10: Hibernate

Java programmers are used to creating software using object-orientated principles. The use of embedded queries (this is true of any query language, not just MySQL) makes object-orientated development quite messy and, to a typical java developer, this spoils the codebase, making it; more awkward to code, less consistently based on natural o-o methods, and more tied to a specific API.

You might imagine that this last issue also becomes more pressing as the growth of different databases increases and more API choices available. For a java developer keeping the code as clean as possible will help refactoring of the codebase, make quicker the ability to switch to different API (of even just test them out).

Hibernate is a framework that allows embedded queries to be removed, and object-orientated methods of database interaction possible, and removing of the problem of 'fragmentation', which has been referred by previous commentators on these issues.

The purpose of this week, was to give you a peek under the bonnet of Hibernate, but mainly to present to your these 'issues' of accessing databases from within an object-orientated perspective.

12.11 Week 11: noSQL

Finally, we learned about the considerably diverse field known as the 'noSQL movement'. Although noSQL is sometimes packaged as a replacement to the relational database approach, a more realistic view of this field is one that treats database management and database design as being related to the task/tasks demanded of data management and design to hand. The amount and variety data has exploded in recent decades. Consequently, the relational database approach, which was first invented decades ago is, inevitably, out of date to some purposes. Therefore, the noSQL movement consists of the hundreds of different kinds of databases that should be taken as meaning that you can do database development, *not only* using SQL, but also using other kinds of databases.

For general classes of database system were given (‘key-value’, ‘wide column’, document-based, Graph)’, before we focused on a specific example of graph database known as neo4J, a leading industrial strength graph database. We introduced the concept of a graph and its components, why you might use a graph database structure, emphasised the flexibilities of this know noSQL example, and (very) briefly looked at Cypher, neo4J’s query language.