Database design and its implementation

Report

FEBRUARY 23, 2019

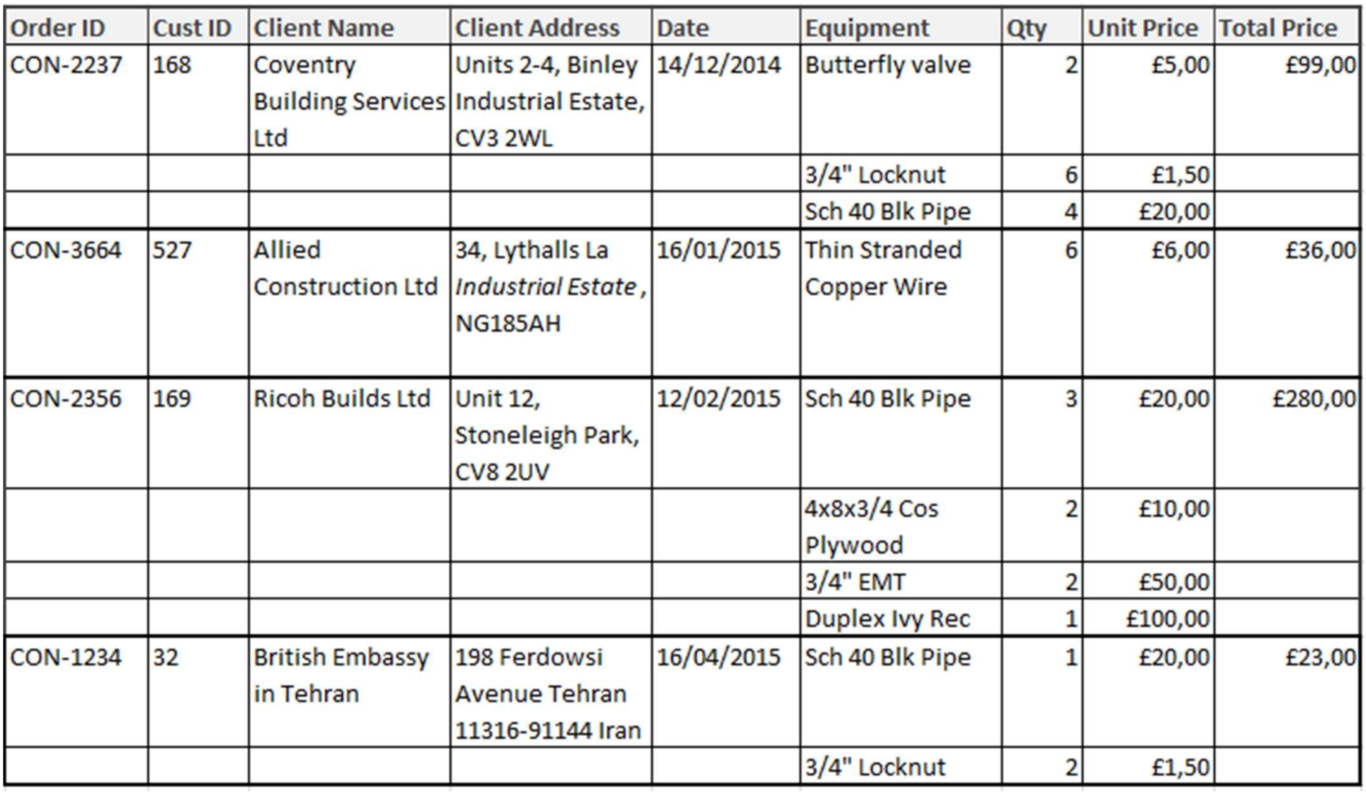
COVENTRY UNIVERSITY STUDENT
Computer science – Year 1

Table of contents

Table of figures

Company situation

Hamzepur Building Services Ltd is a long established construction supplier for companies and the public. The company specialises in supplying equipment such as piping, timber, wiring, plumbing UPVC frames and doors and masonry. As construction demand is picking up Hamzepur's has regular orders for equipment to meet the needs of its client base. In order to support this, an orders book, see table below, is maintained in which equipment orders for its clients are recorded. For each order, the client's details, date, equipment, quantity, unit price and overall price of the order is logged.



Order ID	Cust ID	Client Name	Client Address	Date	Equipment	Qty	Unit Price	Total Price
CON-2237	168	Coventry Building Services Ltd	Units 2-4, Binley Industrial Estate, CV3 2WL	14/12/2014	Butterfly valve	2	£5,00	£99,00
					3/4" Locknut	6	£1,50	
					Sch 40 Blk Pipe	4	£20,00	
CON-3664	527	Allied Construction Ltd	34, Lythalls La *Industrial Estate*, NG185AH	16/01/2015	Thin Stranded Copper Wire	6	£6,00	£36,00
CON-2356	169	Ricoh Builds Ltd	Unit 12, Stoneleigh Park, CV8 2UV	12/02/2015	Sch 40 Blk Pipe	3	£20,00	£280,00
					4x8x3/4 Cos Plywood	2	£10,00	
					3/4" EMT	2	£50,00	
					Duplex Ivy Rec	1	£100,00	
CON-1234	32	British Embassy in Tehran	198 Ferdowsi Avenue Tehran 11316-91144 Iran	16/04/2015	Sch 40 Blk Pipe	1	£20,00	£23,00
					3/4" Locknut	2	£1,50	

Figure 1: Data table

Should we establish this table in a traditional database form?

In the modern era of databases, we are dealing with a big amount of information that is going to be stored somewhere on the servers. As data consumption increases, we need to ensure that database design is efficient, safe and error-free. "Normal forms in a database or the concept of Normalization makes a Relation or Table free from insert/update/delete anomalies and saves space by removing duplicate data" (Pandey 2015). As it is being said, traditional database form is not the best choice for use because it duplicates some of the attributes. For instance, we can see that every new order makes us store order id and client information multiple times. Moreover, a traditional database may lead us to an update, delete and insert anomalies. Consequently, to tackle this issue we must perform normalization.

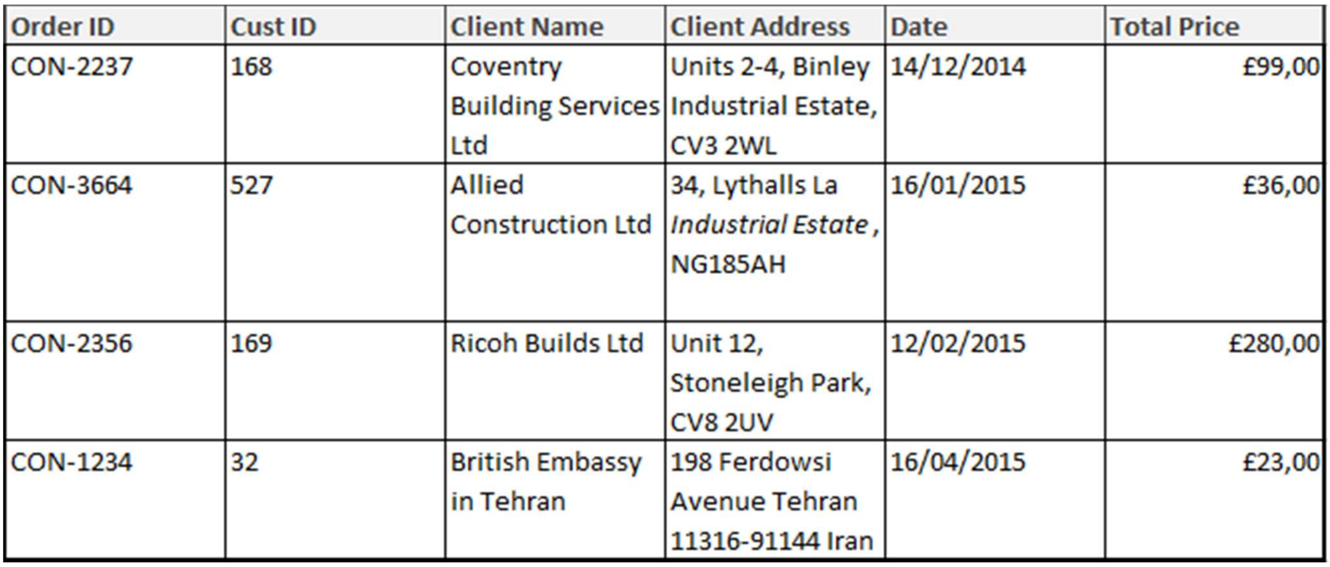
Implementing table normalization

The normalization process is usually made of 3 forms. Each 1st, 2nd and 3rd form requires us to satisfy specific criteria and also to be in the previous form. For example, we cannot be at 2nd form if we have not been in 1st form. Thus, it has to be done coherently. Those steps can be described as this (Hintea 2019):

- a table(also called an entity) is in 1NF (normal form) if: there are no repeating attributes/fields or groups of attributes
- 2NF if: no attribute is dependent on only part of the Primary Key (PK). This only applies to entries with concatenated PK
- 3NF if: all attributes are strictly dependent on the PK and also not on any other attribute that is not a part of the PK

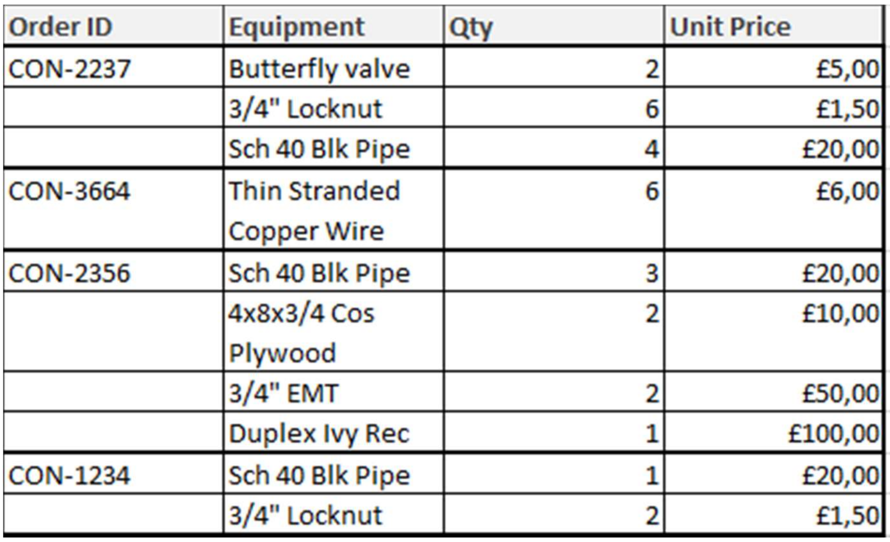
First normal form

Firstly, it is immediately obvious that we have a repeating attribute value such as *OrderID* whereas *Equipment* attribute values are different in every record. This leads us to a separation of the table like in the example below *(Figure 2 and 3)*. *OrderID* helps us to identify which order the item belongs to.



Order ID	Cust ID	Client Name	Client Address	Date	Total Price
CON-2237	168	Coventry Building Services Ltd	Units 2-4, Binley Industrial Estate, CV3 2WL	14/12/2014	£99,00
CON-3664	527	Allied Construction Ltd	34, Lythalls La *Industrial Estate* , NG185AH	16/01/2015	£36,00
CON-2356	169	Ricoh Builds Ltd	Unit 12, Stoneleigh Park, CV8 2UV	12/02/2015	£280,00
CON-1234	32	British Embassy in Tehran	198 Ferdowsi Avenue Tehran 11316-91144 Iran	16/04/2015	£23,00

Figure 2: Orders table

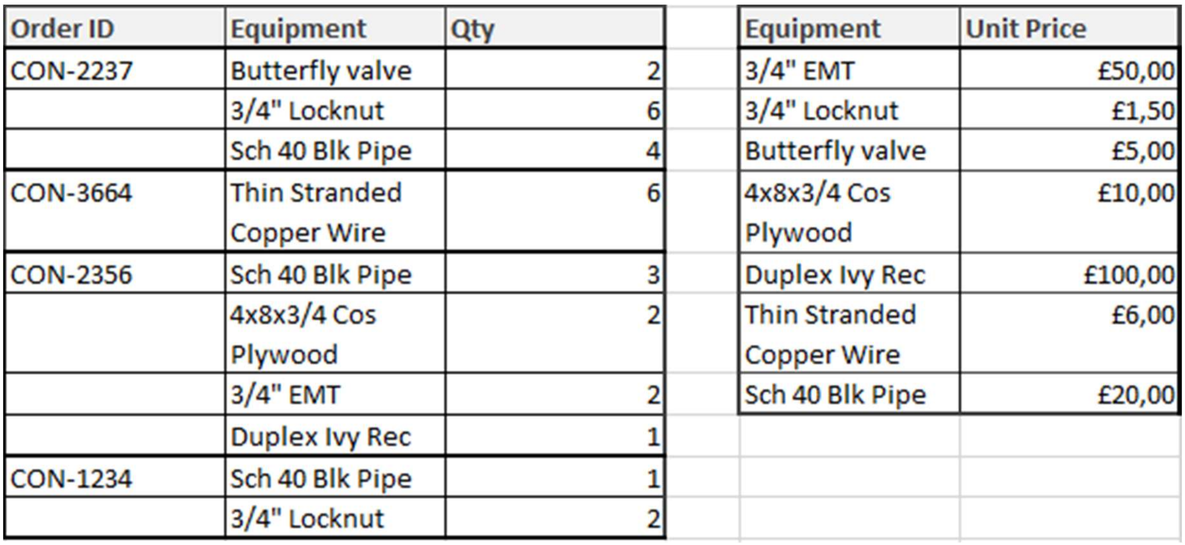


Order ID	Equipment	Qty	Unit Price
CON-2237	Butterfly valve	2	£5,00
	3/4" Locknut	6	£1,50
	Sch 40 Blk Pipe	4	£20,00
CON-3664	Thin Stranded Copper Wire	6	£6,00
CON-2356	Sch 40 Blk Pipe	3	£20,00
	4x8x3/4 Cos Plywood	2	£10,00
	3/4" EMT	2	£50,00
	Duplex Ivy Rec	1	£100,00
CON-1234	Sch 40 Blk Pipe	1	£20,00
	3/4" Locknut	2	£1,50

Figure 3: OrderedItems table

Second normal form

As the 1NF is established, we notice that multiple items appear in *OrderedItems* table and this creates a redundancy. To overcome this issue, it is possible to separate individual item with its price to a new table linking the item/equipment by its name. Overall, an ordered item can be linked with an *Order* table by joining *OrderID* and *Equipment* foreign keys to a Composite Key. It is essential to do so because reference by *OrderID* is not consistent and it may refer to multiple order items. The following picture illustrates the changes *(Figure 4)*.



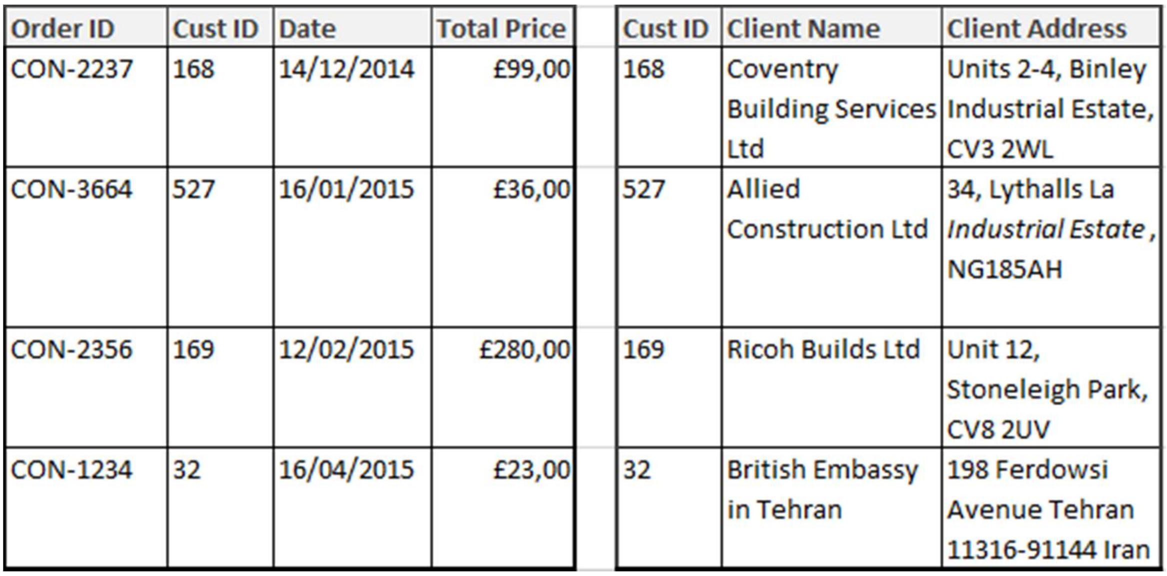
Order ID	Equipment	Qty
CON-2237	Butterfly valve	2
	3/4" Locknut	6
	Sch 40 Blk Pipe	4
CON-3664	Thin Stranded Copper Wire	6
CON-2356	Sch 40 Blk Pipe	3
	4x8x3/4 Cos Plywood	2
	3/4" EMT	2
	Duplex Ivy Rec	1
CON-1234	Sch 40 Blk Pipe	1
	3/4" Locknut	2

Equipment	Unit Price
3/4" EMT	£50,00
3/4" Locknut	£1,50
Butterfly valve	£5,00
4x8x3/4 Cos Plywood	£10,00
Duplex Ivy Rec	£100,00
Thin Stranded Copper Wire	£6,00
Sch 40 Blk Pipe	£20,00

Figure 4: OrderedItems and Item tables

Third normal form

Finally, as the 3NF rule requires, all attributes have to be strictly dependent on the PK. Cleary, it is easy to comprehend that a customer‘s data is not related to an order entity. Ultimately, we achieve the following result by creating a *Client* table and referencing it by *CustID (Figure 5)*.



Order ID	Cust ID	Date	Total Price
CON-2237	168	14/12/2014	£99,00
CON-3664	527	16/01/2015	£36,00
CON-2356	169	12/02/2015	£280,00
CON-1234	32	16/04/2015	£23,00

Cust ID	Client Name	Client Address
168	Coventry Building Services Ltd	Units 2-4, Binley Industrial Estate, CV3 2WL
527	Allied Construction Ltd	34, Lythalls La *Industrial Estate*, NG185AH
169	Ricoh Builds Ltd	Unit 12, Stoneleigh Park, CV8 2UV
32	British Embassy in Tehran	198 Ferdowsi Avenue Tehran 11316-91144 Iran

Figure 5: Orders and Client tables

Identifying attributes of an entity

"Attributes are those pieces of information on an entity that are required for processing performed by the business functions. By carefully examining the business functions, you can determine which attributes need to be maintained for each entity in the database" (CA Technologies 2016). As being said, we must identify the attributes for each table/entity to achieve robust and reliable business functions processing. Obviously, we have a unique *Order*, *Client* and *Equipment/Item* elements (*Figure 6*)."A key is a field, or combination of fields, that is guaranteed to have a unique value for every record in the table" (Churcher 2012: 100). Therefore, we must have those 3 entities to contain PK as they are unique. Moreover, its a must to have a composite/concatenated key in *OrderedItems* entity to link client ordered items with an actual *Order* without losing the quantity. To make a composite key, we will use 2 foreign keys: *OrderID* and *Equipment* (*OrderedItems* table).

Designing entity-relationship diagram (ERD)

ERD is a fundamental graphical tool for effective database design. "Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization" (Rouse, Biscobing and Aberle 2014). There are 3 different ERD types: Conceptual, Logical and Physical. Each of these are served for a different use. According to *Visual Paradigm International Ltd* (2019), the logical model depicts information gathered from business requirements but it is more complex than the conceptual model. For example, it defines a data type used for each attribute and identifies the attribute. From what is said above, the company's logical ERD is visualized in *Figure 6*.

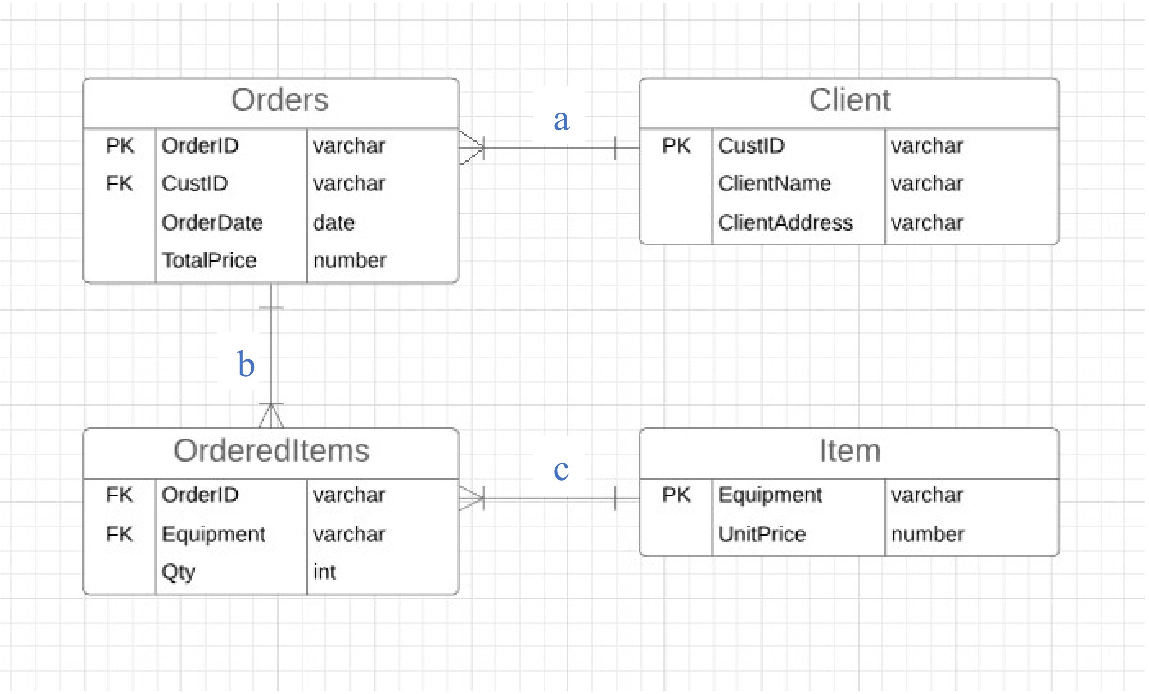


Figure 6: ERD

a. Every *Order* has only one *Client,* but every *Client* can have 1 or more *Order*s.
b. *Order* may have 1 or more *OrderedItems* but 1 *OrderedItem* can only be an instance of 1 *Order*
c. Every *OrderedItem* is part of only 1 *Item,* but the same *Item* can be an instance of 1 or more *OrderedItem*

SQL queries to create and populate a table with records

It is crucial to understand how to control a database. To accomplish that we have to construct SQL query commands to manipulate the data. The following are used to **create** tables:

Entity name	SQL query
Orders	CREATE TABLE orders (orderid VARCHAR PRIMARY KEY, orderdate DATE NOT NULL, custid VARCHAR NOT NULL, totalprice NUMBER(8, 2) NOT NULL, FOREIGN KEY (custid) REFERENCES client(custid));

Client	CREATE TABLE client (
	custid VARCHAR PRIMARY KEY,
	clientname VARCHAR NOT NULL,
	clientaddress VARCHAR NOT NULL);

OrderedItems	CREATE TABLE ordereditems (orderid VARCHAR NOT NULL, equipment VARHCAR NOT NULL, qty INTEGER NOT NULL, FOREIGN KEY (orderid) REFERENCES orders(orderid), FOREIGN KEY (equipment) REFERENCES item(equipment));

Item	CREATE TABLE item (
	equipment VARCHAR PRIMARY KEY,
	unitprice NUMBER(8, 2) NOT NULL);

Furthermore, we have commands to **populate** tables with data:

Entity name	SQL query
Orders	INSERT INTO orders (orderid, custid, orderdate, totalprice) VALUES ('CON-2356', '169', '12/02/2015', 280);

Client	INSERT INTO client (custid, clientname, clientaddress)
	VALUES ('169', 'Ricoh Builds Ltd', 'Unit 12, Stoneleigh Park, CV8 2UV');

OrderedItems	INSERT INTO ordereditems (orderid, equipment, qty)
	VALUES ('CON-2356', 'Butterfly valve', 2);

Item	INSERT INTO item (equipment, unitprice)
	VALUES ('Butterfly valve', 5);

Summary

An obvious conclusion to be drawn is that contemporary databases should be more than a traditional database. It requires proficient database design and it is possible to achieve this by designing ERD and applying normalization processes. “The database needs to be of high quality so that the users of the database would have confidence in using it. The design of the database should be done with an emphasis on each of the design phase“ (Hamad 2016: 7331). Once ERD is made, we can create SQL queries to feed the database with data and launch it for successful use.

Reflection

During this module, I learned a lot of things that I will use in my further carrier. It gave me an understanding of table normalization, SQL queries and ERD. Despite this, I wish to supplement my skill set with additional normalization forms. Moreover, I would love to build a larger and more complex database to make use of more advanced queries. Overall, this module gave me an excellent understanding of modern databases and I found it certainly useful.

List of references

- CA Technologies (2016). *Identifying the Attributes of Each Entity.* [online] available from <https://docops.ca.com/ca-idms/19/en/administrating/administrating-database-design/identifying-attributes/identifying-the-attributes-of-each-entity> [24 February 2019]
- Churcher, C. (2012). *Beggining Database Design*. 2nd edn. New York: Paul Manning
- Hamad, D.R. (2016). 'Logical Database Design'. *International journal of computers & technology* [online] 15 (12), 7329-7332. Available from <https://cirworld.com/index.php/ijct/article/view/4344> [26 February 2019]
- Hintea, D. (2019) *Database systems* [lecture] module 4005CEM, 5 February 2019. Coventry: Coventry University
- Margaret Rouse, Jacqui Biscobing, Laura Aberle (MR, JB, LA) (2014). *Entity relationship diagram.* [online] available from <https://searchdatamanagement.techtarget.com/definition/entity-relationship-diagram-ERD> [24 February 2019]
- Pandey, M. (2015). *Quora forum answer: Why the need for normalization?* [online] available from <https://www.quora.com/Why-the-need-for-normalization> [25 February 2019]
- Visual Paradigm International Ltd (n.d.). *Documentation, part 10, Database design and engineering, chapter 2.* [online] available from <https://www.visual-paradigm.com/support/documents/vpuserguide/3563/3564/85378_conceptual,l.html> [24 February 2019]