University of London
Computing and Information Systems/Creative Computing
CO2209 Database systems
Coursework assignment 2 2019–2020

Your coursework assignment should be submitted as a single PDF file, using the following file-naming conventions:

YourName_SRN_COxxxxcw#.pdf (e.g. GraceHopper_920000000_CO2209cw2.pdf)

- **YourName** is your full name as it appears on your student record (check your student portal); **Please use just your name, and not any appellation such as 'MR' or 'MS'.**
- **SRN** is your Student Reference Number, for example 920000000
- **COXXXX** is the course number, for example CO2209; and
- **cw#** is either cw1 (coursework 1) or cw2 (coursework 2).

REMINDER: It is important that your submitted assignment is your own individual work and, for the most part, written in your own words. You must provide appropriate in-text citation for both paraphrase and quotation, with a detailed reference section at the end of your assignment (this should not be included in the word count).Copying, plagiarism and unaccredited and/or wholesale reproduction of material from books, online sources, etc. is unacceptable, and will be penalised (see our guide on how to avoid plagiarism on the VLE). You can also look at the end of any journal or conference paper to get an idea of how to cite your reference material appropriately.

Introduction

In this coursework you will download a real database. In **Part A** you examine its structure at various levels. In **Part B**, you run some queries on it. In **Part C**, you normalize a relation. In **Part D**, you draw an Entity/Relationship (E/R) diagram and derive a relational schema from it. In **Part E**, you define some basic database terms. In **Part F**, you choose from among several topics and write an essay on one of them.

Preparation and pre-assignment tasks

In coursework assignment 1, we created a 'toy' database, consisting of a small number of tables and just a few rows. Real databases are typically many orders of magnitude larger, in terms of numbers of relations, and the size of each relation. This impacts a crucial aspect of database usage, 'performance' (basically, how fast queries are processed). In this coursework assignment, our practical task will be to download a large database, and use it to learn more about real databases (but note that even this database is small compared to many existing ones).

We are going to download the 'Mondial' database, which carries information about the countries of the world. This is supposedly based on the CIA's **World Fact Book** and a few other data sources. As with many real-world applications*, there may be inconsistencies in its documentation*.

Task 1

Go to this website: http://www.dbis.informatik.uni-goettingen.de/Mondial/ and find the paragraph titled "Generating the Database under MySQL".

Click on each of the following links:
http://www.dbis.informatik.uni-goettingen.de/Mondial/OtherDBMSs/mondial-schema-mysql.sql
http://www.dbis.informatik.uni-goettingen.de/Mondial/OtherDBMSs/mondial-inputs-mysql.sql

This will download two files. You do not download the data in the database directly. Instead, you download these two text files, which consist of SQL statements. The SQL statements in the first file (called **mondial-schema-mysql.sql**) create your tables, and then statements in the second file (called **mondial-inputs-mysql.sql**) will populate (add the data to) the tables that the statements in the first file created.

The second file is about 1.5 megabytes in size, and consists of over 20,000 INSERT INTO statements.

To create the Mondial database, we must first download and then execute the SQL commands in these files.

These files may be downloaded and submitted to a "front end" processor for your database, if you are using one (that is, the statements themselves will be processed).

Or, the contents of each of the files may be copied and pasted into a text editor (you will probably need to add a new file name extension to the files to do this, for example, changing *mondial-inputs-mysql.sql* to *mondial-inputs-mysql.sql.**txt***). The SQL statements in those files can then be copied and pasted directly to the MySQL or MariaDB command-line processor if you are running them in command-line mode (of course, you will *first* do this with the table-creation file, and *then* with the data inputs file).

Tip: When running MySQL or MariaDB from the command line, if your operating system is Windows, you may want to make the DOS Command Prompt window larger than the default value. Follow the link below to see how to do this:
http://www.isunshare.com/windows-10/change-command-prompt-window-size-in-windows-10.html

There are similar sites for earlier versions of Windows. You may have to reboot after doing this for the new window size to take effect.

Note: This is a large database, which may take up to half an hour to download. If you are using Linux, and you encounter a problem with the download, come to the course discussion board and see if your problem has been answered there.

Task 2
From this same website (http://www.dbis.informatik.uni-goettingen.de/Mondial/), download the documents that display the logical structure of the database. There are three, all of which carry the same information, but show it in increasingly 'physical' ways. The first one – the E/R diagram – shows abstract entity-types and their attributes, and how the entity-types are related to each other. The second one shows actual relations, but illustrates their links visually. The last file just lists relations and their attributes. In theory, all of them hold the same information, but present it in different ways.

An E/R diagram:
http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial-ER.pdf

A "Referential Dependency" diagram:

http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial-abh.pdf

A Relational Schema:
http://www.dbis.informatik.uni-goettingen.de/Mondial/mondial-RS.pdf

These documents should let us see how the data in the separate tables is related, namely which tables hold data relating to the same things. However, as in many real-world applications, what is on paper and what exists in reality may not match perfectly.

A note on vocabulary: In reading about relational databases, the following words are usually synonyms: that is, they mean the same thing: ‘relation’ = ‘table’ (and sometimes, = ‘file’); ‘attribute’= ‘field’, = ‘column’; ‘tuple’= ‘row’ (and sometimes = 'record'). The first word is the ‘official’ name. Note that technically, a ‘relation’ is a special kind of ‘table’; that is, ‘table’ is a broader term than ‘relation’. But in general, or in casual discussion, they mean the same thing. When dealing with non-technical clients, you will find it easier to talk about ‘tables’ and ‘fields’ or 'columns', and ‘rows’, rather than ‘relations’, ‘attributes’ and ‘tuples’.

The E/R diagram is a very simple one, which omits cardinality and participation constraints. Don't worry too much about these documents until you think you have understood the concepts of data dependency, functional dependency, keys and normalization. They will be useful for constructing SQL statements to answer queries which require you to know which tables are linked to which other tables. Be aware that E/R diagrams assume that the world can be conceived in terms of entity-types, which have attributes, and which are related to each other. Sometimes this is only a very rough fit to reality, and we are forced to think of certain things as ‘entity-types’ that we might not naturally think of in this way.

Note that the Relational Schema shown here is only a broad outline schema. It doesn't show datatypes for the attributes, or which attributes are Primary or Foreign Keys, or other constraints.

Important note about the Mondial Database: The value of this database is that it is not a toy one. However, it is definitely out-of-date, and was inaccurate even when first put up on the web. Remember, all large data sets must be assumed to be ‘dirty’. **All documentation of large systems must be assumed to have inconsistencies, errors or gaps until proven otherwise**.

Additionally, its designers made at least one poor choice: they have **field names** (attributes, or columns) that are the same as **relation names**. So, there is a *relation* called ‘Country’, and in some of the other relations, there is an *attribute* called ‘Country’ as well. The ‘Country’ field (attribute) of these other relations is a Foreign Key for Country *Code*. That is, it matches an attribute in the Country table which has a different name. It would have been a better idea, arguably, to label *all* of these attributes ‘Country-Code’. There are other attribute-relation names which have the same problem. Don’t let these confuse you. If you consult the Referential Dependency Diagram you can see which attributes in which relations are actually ‘the same’ (from the same domain) and are thus possible meaningful natural JOIN links, despite having different names.

Also, note that the Mondial database, as implemented using MySQL or MariaDB, does NOT enforce Foreign Key – ‘referential’ - integrity. That is, it would be possible to have a ‘Country’ field (attribute) in a relation like Religion, which has no matching ‘Code’ field (attribute) in the relation Country, although the relation Country should list every country which appears in the other tables of the database – it should be a 'master' relation.

Note: If you have questions about any part of this assignments, or need help in completing any of the tasks, use the course discussion board.

Coursework assignment 2

Part A

Getting information about the Database

Background reading
In preparation, and as part of your background reading, you should read the **subject guide, volume 1**, pp.106–123**.**

If you ever get a job as a Database Administrator, or have to work with a database that you yourself did not create, you will have to understand your database's structure, i.e. what tables are there, what data they hold, and how the data in each table is related to the data in other tables. In addition, you will have to understand to what extent this data represents reality. We approach this problem using the method of abstraction. First, we try to get an overall picture of the database – the kinds of things it's holding data on, and how they are related – and then we get closer and closer to the actual data.

Getting information about the database at the semantic/conceptual level
When we use the words 'semantic', or 'conceptual' or 'logical' in connection with data, we are referring to the *information* the data is supposed to convey, abstracted away from *how* it does this, from its actual physical implementation on paper, or magnetic disc, or any other physical medium.

In the Mondial database, we are given three levels of non-physical descriptions of the data: an E/R diagram – sometimes called the 'conceptual' or 'semantic' level; a Referential Dependency diagram, and a Relational Schema, proceeding from a higher level of abstraction to a lower one.

An E/R diagram is at a higher level of abstraction than a Referential Dependency diagram, which in turn is more abstract than a Relational Schema. Each new level gives us more detailed information.

Note that at each level, there is always a choice about how much information to include – the examples here are fairly minimal – thus, the E/R diagram you download here doesn't have cardinality and participation constraints, and the relational schema used here does not include data types (see p.106 of the **subject guide, volume 1** for further discussion of the levels of information design).

Question 1
Look at the **E/R diagram** that you have downloaded, and then look at the way E/R diagrams can (not necessarily "must") represent cardinality, as described in the **subject guide, volume 1**, pp.114–115). 'Cardinality' refers to the situation where there is a relationship between instances of two entity types: cardinality is the number – the 'count' – of one type which can participate in a relationship with a single instance of another type.

For example, a given lecturer may be assigned several students to tutor; a given student might be limited to one lecturer as his or her tutor. The relationship would be 'one-to-many' (lecturer-to-student) in the first case, and 'one-to-one' (student-to-lecturer) in the second. We might want to be more explicit. Our client might specify that a lecturer had to have a minimum of two students to tutor, and a maximum of eight. We might want to show this on our E/R diagram and incorporate a check in the database to make sure this constraint was followed. If our client does not require there to be a maximum, we typically use the letters "N" or "M", to mean simply "more than one".

i. With respect to representing cardinality, how is the E/R diagram that you have downloaded *different* from the E/R diagrams in the **subject guide?**

ii. Review p.109 of the **subject guide, volume 1**, where 'weak' entity types are discussed. What is the difference between a 'strong' and a ‘weak' entity type?

iii. Looking at the E/R diagram, can we identify any 'weak' entity types?

iv. Look at the actual tables for COUNTRY and RELIGION (Use the DESCRIBE COUNTRY and DESCRIBE RELIGION commands, and enter SELECT * FROM COUNTRY LIMIT 5; and SELECT * FROM RELIGION LIMIT 5;). Suppose we had to enter a new country into the COUNTRY table. Is it possible to enter a new country in the COUNTRY table, without entering any data for it into the RELIGION table? Could we do the same for a religion? That is, could we enter a new religion in the RELIGION table, without entering any countries for it? If you're not sure about this, try to do it in both cases.

v. Suppose we wanted to make 'religion' a strong entity type. That is, we wanted to be able to record information about each religion, even if a religion was not represented in any country. What would we have to do?

[5 marks]

Question 2
Look at the **Referential Dependency Diagram** (‘Mondial.abh.pdf’), and answer the following:

i. How many tables are shown in this diagram?

ii. How many tables are listed when you run the SHOW TABLES command in **Part A, Question 3** of this coursework assignment?

iii. According to the Referential Dependency Diagram, what are the attributes (fields) of the POLITICS table?

iv. What are the attributes (fields) of this table that are listed when you run the DESCRIBE POLITICS command?

v. What conclusion can we draw about the database documentation that may be supplied with an organisation's database?

[5 marks]

Getting information about the database at the logical level

Although documentation describing the database at the conceptual and logical level can be useful, the only ultimate ‘documentation’ is the data itself. Here, we will get the actual relational schema, along with a bit of purely ‘physical’ information (namely, information about how the tables are indexed).

Compiling a schema of the tables you have downloaded

This compilation will provide basic information about the database tables you have set up. Unlike the documentation in the previous section, this will show you what the database ‘intension’ actually is.

Combined with the Referential Dependency diagram, the E/R diagram, and the rather general Relational Schema you have already downloaded, you would have the materials you need to start to understand your database's structure. The document called Mondial.RS.pdf is a Relational Schema, but we can generate one with more information using MySQL itself.

Question 3
You don't actually have to write anything here, but rather, just copy in the results of running some commands. We will get some basic information about all of the tables, and then some detailed information.

To do this, you will need three SQL commands:

SHOW TABLES;   *and*
DESCRIBE <tablename>; *and*
SELECT COUNT(*) FROM <tablename>.

SHOW TABLES just gives us the **name** of every table (relation) in our database. The other two commands let us get data about a particular table: its structure, and how many rows (tuples) it has.

Note that for '<tablename>' you will need to substitute the name of one of the tables listed by **SHOW TABLES.** We will look at just a few of them. Use of a word processor or a simple text editor can make this a very quick operation if you are working with MySQL directly from the DOS prompt. Note that for earlier versions of Windows, to paste into the Command Prompt, you may need to right-click – CTRL-V may not work.

It will be useful to be able to output what you see on the screen to an output file; you can use the command "TEE" to do this, as follows (user input in bold):

MariaDB [Mondial]> > **TEE D: OutputLog.txt** – whatever shows on the screen is also copied to the file OutputLog.txt which I have placed on my D: disc in this example, but which can be located anywhere you like, provided you give the full Path name.

MariaDB [Mondial]> **SHOW TABLES;** – the names of all of the tables will be sent to OutputLog.txt as well as being shown on the screen;

MariaDB [Mondial]> > **DESCRIBE COUNTRY**;
MariaDB [Mondial]> >**SELECT COUNT(*) FROM COUNTRY;**
MariaDB [Mondial]> > **SHOW INDEX FROM COUNTRY;**

- and so on ... for the following relations in this database;
- (In other words, substitute the names below, one at a time, for the name COUNTRY in the commands above.)
- ECONOMY
- ISMEMBER
- LANGUAGE
- ORGANIZATION
- POPULATION

mysql> **NOTEE;** – turns it off;

Include the requested information for the tables listed above, plus the size of each of these tables, which you can find out by executing the SQL statement below.

To see the size in megabytes of each of your tables, do this:

```
SELECT
      table_schema as `Database`,
      table_name AS `Table`,
      round(((data_length + index_length) / 1024 / 1024), 2) `Size in MB`
FROM information_schema.TABLES
WHERE table_schema = 'mondial'
ORDER BY (data_length + index_length) DESC ;
```

[5 marks]

Question 4
Answer the following questions:

i. What is the total size of the Mondial database in megabytes?

ii. What are the largest, and the smallest, relations in the Mondial database in terms of total bytes? There may be 'ties'.

iii. For any two relations (in any database) is it the case that the relation with the largest cardinality – number of tuples (rows) – must be the largest in terms of total bytes of data? If not, can you give an example from this database which shows this?

iv. For any two relations, is it the case that the relation with the largest degree (number of columns) must be the largest in terms of total bytes of data? If not, can you give an example from this database which shows this?

v. If, given two relations, one is larger than the other in terms of both cardinality and degree, is it *necessarily* larger than the second one, in terms of total bytes of data? If *not*, can you give an example from this database that shows this?

[5 marks]

What to submit: Answers to **Question 1 (i–v), Question 2 (i–v), Question 3** the results of running the commands for the tables named in Question 3 above) **and Question 4 (i–v).** Please follow the numbering conventions in this coursework assignment and start each answer on a new line. Thus, your answers should look like this:

> **Part A, Question 1**
> **i *Your answer…***
> **ii *Your answer …***
>
> **Part A, Question 2**
> **i. *Your answer…***
> **ii. *Your answer …***

[Total for Part A: 20 marks]

Part B

Queries on the Mondial database

A note about SQL: SQL's tables do not conform completely to the definition of relations. In particular, the tables which result from a query can have duplicate tuples (rows), which in most cases is not what we want, and violates the definition of 'relation'. To avoid this, always use the **DISTINCT** keyword, as in **SELECT DISTINCT …**

Show not just your query, but the data set that results. **Use 'LIMIT 5' at the end of your**

query if your query returns more than five results. Note that queries that involve COUNTs and SUMs, etc., will return just one value. Each answer should include:

- the original natural language query
- the SQL commands to answer it
- the data set that results (limited to the first five tuples if there are more than five in the result).

Example:

Query number

Query in natural language

B(x) What is the query that will tell us the name and population of each country?

```
SELECT DISTINCT Name, Population
  FROM country
  LIMIT 5;
```

SQL – Do not put your whole query on one line – use several lines, and indent for readability.

```
+---------------------+
| Name | Population  |
+---------------------
| Austria | 8023244 |
| Afghanistan | 22664136 |
| Antigua and Barbuda | 65647 |
| Albania | 3249136 |   |
| American Samoa | 65628 |
+---------------------+-
```

Results

Question 1
What is the query that will list the names of all countries that have a GDP less than 15,000 and an infant mortality greater than 6? **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 2
What is the query that will list the names of all cities that have populations greater than the population of New York? **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 3
What is the query that will list the world's lakes, and for each one, the total number of countries where each is found? (Hint: use **GROUP BY**). **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 4
What is the query that will list the world's rivers that pass through at least two countries, and for each one, the total number of countries where it runs? (Hint: use **GROUP BY** and **HAVING**). **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 5
What is the query that will list the names of all countries that are full members (not

observers or non-regional members) of the World Health Organization and have GDP's greater than 3,000,000? **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 6
What is the query that will list the names of all countries that are NOT members (or observers or non-regional members) of the World Health Organization and have GDP's greater than 100,000? **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 7
What is the query that will list the total population for all countries (all countries' populations added together)? **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 8
What is the query that will list the name of the country with the highest population density? (population density = population/area). **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 9
What is the query that will list the world's languages, and for each one, the total number of countries where each is represented? (Hint: use **GROUP BY**). **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

Question 10
What is the query that will list the world's languages that are present in at least 5 countries, and for each one, the total number of countries where it is represented? (Hint: use **GROUP BY** and **HAVING**). **Show not just your query, but the data set that results. Remember to use SELECT DISTINCT** and **LIMIT 5;**

What to submit: Answers to **Questions 1–10,** in the requested format.

[20 marks]

[Total for Part B: 20 marks]

Part C

Normalizing an Unnormalized Relation

Background reading
You may wish to read the **subject guide volume 1**, pp.131–141, and to re-read **Appendix IV** of coursework assignment 1 before attempting the questions in **Part C**.

The basic idea of database normalization is that a single 'fact' – a dependency among data items – should be stored in one place only, and independently of other facts. The word 'normalize' has many completely different meanings. Don't confuse database normalization with other uses. When we design a set of tables, we sometimes find that we have stored the same fact in more than one place. For instance, a customer's address may be stored *both* in a table dealing with billing, and *also* in a table dealing with marketing. If the customer changes their address, we can end up with inconsistent data.

Sometimes we find that it is necessary to violate the rules of normalization for performance purposes. That is, we deliberately store the same data in more than one place, in order to avoid the performance hit of doing a JOIN to retrieve the data. But the initial design of a database should be according to the principles of normalization, so that any violation of these principles is a conscious one of which we are aware.

We begin the process of normalization by identifying the functional (one-valued from one data item to another) dependencies among the data, and also noting any dependencies that are multi-valued in both directions. Remember, we are only interested in dependencies that we cannot deduce from other dependencies, and which do not depend on other data items for their existence. In the table below, for example, BandTown and AgentTown have a relationship, but it is not one we need to record independently.

Having identified the functional dependencies and the both-way multi-valued dependencies, we then try to arrange the data so that every determinant (the attribute or combination of attributes on the left-hand side of a functional dependency diagram) is a Candidate Key of a table. If there are mutual multi-valued dependencies, each of them goes into its own table. There can cause situations in which further work may be required, or in which we have to choose between two less-than-optimal arrangements, but they are rare.

Consider the following (unnormalized) relation, which contains information on concerts by bands, organised by their agents. It gives the date of the concert and the venue (where the concert will be held). It also gives the town in which the agent is located, and the hometown of the band. Agents and bands have only one town. A band plays at most at one venue per evening, although a given venue can host more than one band on a given evening. A band can play at the same venue, on different dates. An agent can manage more than one band, but a band will have only one agent.

CONCERTS
PRIMARY KEY: BANDNAME + DATE

BANDNAME	BANDTOWN	AGENT	AGENTTOWN	DATE	VENUE
The Discords	Nashville	Perry Smollet	Odessa	27.05.2019	Smallville
The Discords	Nashville	Perry Smollet	Odessa	02.07.2019	Smallville
The Discords	Nashville	Perry Smollet	Odessa	03.07.2019	Kleinstadt
Kakaphony	Mumbai	Ahmed Chaudry	Paris	27.05.2019	Smallville
Infidelity	Wuhan	Ahmed Chaudry	Paris	04.07.2019	PequenoPueblo
Nostalgic	Wuhan	Perry Smollet	Odessa	04.07.2019	Obodonta
The Discords	Nashville	Perry Smollet	Odessa	06.07.2019	Gradche
Infidelity	Wuhan	Ahmed Chaudry	Paris	09.07.2019	Chisana Machi
Kakaphony	Mumbai	Ahmed Chaudry	Paris	10.07.2019	Chisana Machi
The Discords	Nashville	Perry Smollet	Odessa	09.07.2019	PetiteVille

Question 1

i. Identify the functional dependencies in the above table. Your answer should conform to the following format:

If, taken together, attributes A and B determine C, show it this way: A + B → C.

ii. Identify the partial and transitive dependencies in the original relation. A Partial dependency is when there is a composite Primary Key (a Primary Key made up of more than one attribute), but only part of it is a determinant for another attribute. A Transitive dependency is when one attribute is a determinant of a second, and the second is a determinant for a third.

[5 marks]

Question 2
This table is susceptible to what are called insertion, deletion, and modification 'anomalies'. An 'anomaly' in database terms is the possibility of not being able to add information, or of inadvertently losing information, or of having inconsistent information, due to designing a table in which a single data dependency – a link between two data items – is not recorded independently, and/or is recorded more than once). Give an example of each kind.

[5 marks]

Question 3
Recast this relation as a set of relations, each of which is in Boyce-Codd Normal Form, which together hold the same information that this relation does.

[5 marks]

What to submit: Answers to **Questions 1 (both sub-parts), 2** and **3,** in the requested format.

[Total for Part C: 15 marks]

<u>Part D</u>

E/R diagrams

A company that makes telescopes receives telescope **parts** (such as lenses, mounts, tripods, and other parts) from **suppliers**, and then uses these parts to put together a number of different **models** of telescope. These telescopes are then sold to **shops** which specialise in optical products.

The company always wants to have more than one supplier for each different type of part, so that it will not become dependent on any single supplier. A model of telescope may exist but not yet be stocked in any shop.

Each part-type can be used in the assembly of several different models of telescope. A telescope is made up of many different types of part. A given retail shop can receive and re-sell many different models of telescope. No retail shop has a monopoly on re-selling any given telescope model.

A supplier is only recorded if it supplies at least one part to the manufacturer. No shop will be recorded unless it is selling at least one model of telescope.

Question 1
Draw an Entity-Relationship Diagram to illustrate the relationships among telescope models, part types, part suppliers, and retail shops.

Use the convention for E/R diagrams explained on pp.114–115 of **volume 1** of the **subject guide**, i.e. use the (min, max) notation. Note that on the illustration on p.115, there should be a comma between '5' and 'm' on the courses-to-students relationship, and that the tutor-to-student relationship should be [5,20]).

[5 marks]

Question 2
Prepare a normalized relational schema that can record the relationships illustrated in **Question 1.** Assume that telescope-models are identified by *Model-Numbers*, suppliers by *Supplier-Names*, shops by *Shop-Names*, and part-types by *Part-Codes*. Be sure to indicate the **key** of each relation.

[5 marks]

Question 3
Extend the relational schema in **Question 2** by showing a (new) schema for a relation that could record how many of which kind of parts were shipped to us, and by which supplier, and on what date, for each delivery we received. For example, the relation should be able to record that supplier 'Cheng Optics' shipped 325 of part-type XB-330 and 20 of part-type XC602 on 22 March 2018, and that Supplier 'Tatu Industries' shipped 125 of part-type XC602 on 12 January 2019. **Be sure to indicate the primary key of this relation**.

[5 marks]

What to submit: Answers to **Questions 1, 2,** and **3** in the requested format.

[Total for Part D: 15 marks]

<u>Part E</u>

General knowledge

Question 1
Write brief (one or two sentence) definitions of the following terms to show how they are used when applied to relational databases:

i. Relation
ii. Tuple
iii. Intension
iv. Extension
v. Candidate Key
vi. Primary Key
vii. Compound (or composite) Key
viii. NULL value
ix. Functional dependency
x. Determinant
xi. Degree
xii. Cardinality
xiii. Transaction
xiv. Deadlock
xv. View

[15 marks]

What to submit: Answers to **i–xv,** in the requested format.

[Total for Part E: 15 marks]

Part F

Essay

Choose **one** of the following topics (i–iii), and write a brief essay summarising the topic. Your essay should not be more than about 500 words. (For comparison, this coursework assignment as a whole is about 5000 words). Some links have been provided to get you started, but don't confine yourselves to them.

i. **Database hacking**
https://www.wikihow.com/Hack-a-Database
https://www.hackers-arise.com/post/2016/12/08/hacking-databases-part-1-getting-started-with-terms-and-technologies
https://www.darknet.org.uk/category/database-hacking/

ii. **Database indexing**
http://www.databaseguides.com/what-are-database-indexes
https://stackoverflow.com/questions/1108/how-does-database-indexing-work
https://www.geeksforgeeks.org/indexing-in-databases-set-1/

iii. **Improving Database performance**
https://www.keycdn.com/blog/database-performance
https://www.eversql.com/5-easy-ways-to-improve-your-database-performance/
https://www.c-sharpcorner.com/UploadFile/ff2f08/tips-to-improve-sql-database-performance/
https://www.c-sharpcorner.com/article/why-is-my-sql-server-query-slow/

[15 marks]

What to submit: An essay of about 500 words.

[Total for Part F: 15 marks]

[Total for assignment: 100 marks]

[END OF COURSEWORK ASSIGNMENT 2]