CSC5007Z SQL Assignment 1 Part 1

1. Instructions

In this assignment you will:

1. Create and load your own MySQL database using the SQL file '`classicModels.sql`', available from the Amathuba assignment page.
2. Write SQL queries to answer the questions below and run them against your database. **Your queries must be correct for any instance of the database schema, and not just for the given sample data**.

A diagram for the database schema may be found at the end of this document.

Please use the Amathuba Discussions forum for all questions on this assignment, so all information is in 1 place accessible to all. Emails will not be answered. **Check Amathuba and the forum frequently** for messages on this assignment.

2. Submission

Submit one zip file containing separate files for each query to the automatic marker.

Call the file with your answer to question 1 '`query1.sql`', the one with your answer to question 2 '`query2.sql`', and so forth.

To ensure successful automatic marking, make sure that you use lowercase and uppercase letters exactly as in the assignment question - the output of each of your files will be compared with the expected output.

NOTE:

- Your ZIP file must only contain your answer files. It should not contain a folder containing your answer files.
- Do ***not*** have any comments in these files, only the SQL statements (automatic marker expects no comments).
- You do not need to complete all questions before trying out your answers on the automatic marker – it will just report that it can't find some.
- The automatic marker is used by all CS students and at times can be under heavy load. If there is not an immediate response when submitting your work, please be patient.

Automarker submission process:

a) Click on the Automatic Marker item on the left menu of the Amathuba site. The assignment page,including the list of *Open Assignments,* is displayed. Each open assignment should have a Browse and a Submit button.

b) Click on the Browse button of the assignment you want to submit and find and select the zip file you want to upload.

c) Click the Submit button to submit the assignment to the marker.

d) The automarker will mark the assignment and provide feedback.

e) Examine the feedback, make the necessary changes to your program and follow this procedure again to submit an updated solution.

Continued

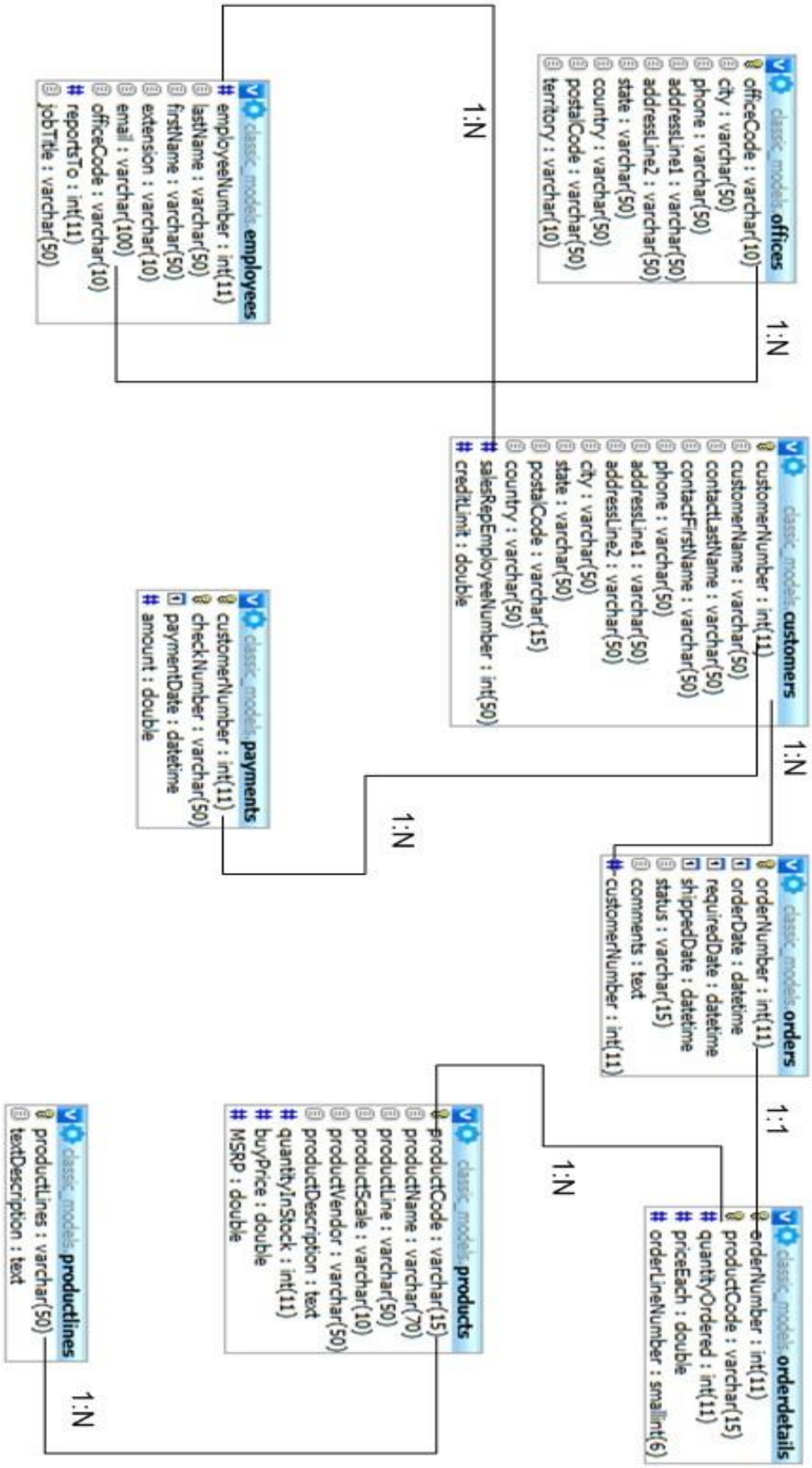
3. Questions (queries)

Each correct query earns 5 marks.

1. Show all information in the **offices** relation.
2. Show the **city** and **phone** of all **offices**, in alphabetical order of **city** – if there is more than one office in a city then give their phone numbers in decreasing order.
3. Show all information in the **offices** table for **offices** located in countries 'UK' and 'SA'.
4. Show all information in the **orderdetails** relation for large orderlines. A large orderline is one where the cost is more than 10 000 (cost is **quantityOrdered** times **priceEach**).
5. We have doubled our **quantityInStock** of every product from **productVendor** 'Exoto Designs'. Show **productVendor**, **productCode**, and new (doubled) **quantityInStock** for all 'Exoto Designs' products - I'm unsure how "Exoto" is spelt, but I know it starts "Ex" and has a "to" somewhere. Call the last column newStock.
6. In what cities do we have **offices**? Call the answer column `**city**'.
7. Show all data for **offices** where there is an **addressLine2** value, but the state is missing (NULL).
8. How many tuples (rows) are there in **employees**? Call the result column '**numEmps**'.
9. What is the average **buyPrice** in the database? Call the result '**avPrice**' and show 2 decimal places.
10. How many different **creditLimit** values are there in our database? Call the result '**numLimits**'.
11. Show **orderNumber**, **status**, **priceEach**, **quantityOrdered** and **productName** for all **products** from **productVendor** 'Exoto Designs'.
12. Show the **OrderNumber**, **Comments** and **customerName** of all orders that have a **status** of 'Disputed'.
13. Show the **productCode** of all **products** that have never been ordered.
14. Show how many **employees** there are in each office. Call the 1st column '**officeCode**' and the 2nd column '**numEmps**'.
15. Show how many Sales Reps there are in each office. Call the 1st column '**officeCode**' and the 2nd column '**numReps**'.
16. Show how many customers each employee is associated with (as **salesRepEmployeeNumber**), but only for employees who are the **salesRepEmployeeNumber** for at least 10 customers and who report to employee 1143. Call the 1st column '**salesRepEmployeeNumber**' and the 2nd column '**numCustomers**'.
17. Which pairs of **offices** are in the same country? Show the **country** and the 2 officeCodes, making sure that the 2nd column's **officeCode** is less than the 3rd column's **officeCode** so information is not repeated. Call the 2nd column '**oneOffice**' and the 3rd column '**otherOffice**'.
18. What percentage of the **offices** are located in the 'USA' **country**? Call the result '**percentUSA**'.
19. Show the **customerNumber** of the customer/s with the largest single check (cheque) payment **amount.**
20. Give a SQL statement to output "YES" if any attribute storing a price (and thus specified as NOT NULL) contains a value that is zero – i.e., if **priceEach** in any **orderdetails** row, or **buyPrice** in any products row, is zero. If there is no zero in those columns, then it should output an empty table. Call the result column '**anyProblems**'.

Continued

4. Appendix


classic_models offices
officeCode : varchar(10)
city : varchar(50)
phone : varchar(50)
addressLine1 : varchar(50)
addressLine2 : varchar(50)
state : varchar(50)
country : varchar(50)
postalCode : varchar(50)
territory : varchar(10)
classic_models employees
employeeNumber : int(11)
lastName : varchar(50)
firstName : varchar(50)
extension : varchar(10)
email : varchar(100)
officeCode : varchar(10)
reportsTo : int(11)
jobTitle : varchar(50)
classic_models customers
customerNumber : int(11)
customerName : varchar(50)
contactLastName : varchar(50)
contactFirstName : varchar(50)
phone : varchar(50)
addressLine1 : varchar(50)
addressLine2 : varchar(50)
city : varchar(50)
state : varchar(50)
postalCode : varchar(15)
country : varchar(50)
salesRepEmployeeNumber : int(50)
creditLimit : double
classic_models payments
customerNumber : int(11)
checkNumber : varchar(50)
paymentDate : datetime
amount : double
classic_models orders
orderNumber : int(11)
orderDate : datetime
requiredDate : datetime
shippedDate : datetime
status : varchar(15)
comments : text
customerNumber : int(11)
classic_models orderdetails
orderNumber : int(11)
productCode : varchar(15)
quantityOrdered : int(11)
priceEach : double
orderLineNumber : smallint(6)
classic_models products
productCode : varchar(15)
productName : varchar(70)
productLine : varchar(50)
productScale : varchar(10)
productVendor : varchar(50)
productDescription : text
quantityInStock : int(11)
buyPrice : double
MSRP : double
classic_models productlines
productLines : varchar(50)
textDescription : text
1:N
1:N
1:N
1:N
1:1
1:N
1:N