Student ID: 730091091
ECM1420DA Exam

1.

a. The first benefit is that an RDBMS allows you to consolidate and centralise all student data into a single, well-organised repository. Instead of various spreadsheets across different workstations, you have one unified database. This means that everyone will have access to the same accurate and up-to-date information and that changes only have to be made in one place. Another benefit is that RDBMS systems use SQL for data retrieval, which allows staff to query the database quickly and efficiently. Complex queries can be stored and executed to extract specific information. The final benefit is that RDBMS systems provide robust security features. Access control can be set at various levels, ensuring only authorised personnel can view or modify specific data. Most RDBMS systems also have encryption and authentication mechanisms built, to protect sensitive information.

b.

i. The user has decided to process the deletion as a transaction, so if there are any errors during execution, the deletion can be rolled back, reversing the effect of the transaction.

ii. This section of the code declares the catch-block. This is an error-handling mechanism which will only be executed if an error occurs during the execution of the try block. In this case, if an error occurs, the deletion will be rolled back.

iii. This commits the transaction. This means that all changes that have been made during the transaction are applied to the actual database. This essentially completes the transaction, meaning it can no longer be rolled back, and the effect of the deletion can now be seen by all users of the database.

iv. The WHERE clause specifies which records to delete. If the WHERE clause was not present, all records in the corresponding tables would be deleted.

v. Databases supporting OLTP must deliver ACID compliance to ensure the integrity of the data. In this case, the SQL code complies with ACID. This is because the transaction is only executed once (atomic), consistency is ensured by deleting the models before deleting the manufacturers (making sure there are no clashes with referential integrity), the transaction is isolated from other transactions, and the transaction is durable since it checks for errors before committing.

c.

i. The IDENTITY keyword allows a unique number to be generated automatically when a new record is inserted into the table. In this

case, it is used for the primary key of each table since we would like this to be created automatically every time a new record is inserted. Where no values are specified, as is the case in this SQL code, the incrementation will start with 1 and increase by 1 for each record.

ii. Add a NOT NULL clause. This can be added to the CREATE TABLE statement or once the table has been created using ALTER TABLE.

iii. SELECT t.empid, empname, taskdate, hours FROM TimesheetEntry t, Employee e WHERE t.empid=e.empid AND tasked=60 ORDER BY empname, date;

iv. SELECT t.taskid, taskname FROM Task t LEFT JOIN TimesheetEntry te ON t.taskid=te.taskid WHERE timeentid IS NULL;

v. Setup a foreign key constraint where the foreign key taskid in the TimesheetEntry table references the primary key taskid in the Task table. This will cause an error to occur whenever someone attempts to execute a change that causes there to be a TimesheetEntry that refers to a task that doesn't exist.

vi. You can add the DISTINCT keyword when defining the taskname field in the task table. You can do this in the original CREATE TABLE statement or an ALTER TABLE statement. This will reject any insertions to the table where a record has a duplicate task name.

2.

a.

b.

i. Item table – ItemID, ItemName, Unit, PricePerUnit
Customer table – CustomerID, CustomerName, PhoneNumber
Order table – OrderID, CustomerID, Date, Status
OrderLine table – OrderID, ItemID, Qty

ii. In practice, the price of items fluctuates and changes over time due to a variety of reasons, such as stock, supply, demand, sales, and inflation. Whilst an order is in the 'Open' status, the price of the order should be affected by these price changes, since it has not been finalised. Once the customer has paid and the order has been finalised, the price of the items should be locked in and now cannot change. This will be when the order is in the 'Delivered' or 'Completed' status.

3.

a.

i. My first recommendation would be to train the staff to test the database for errors before backups and to test the backups. This is extremely important since there is no point in taking backups if

they aren't in a restorable state. In the event of an issue such as the one described, there can be confidence that restoring the back will bring it back to its original, usable state. This process would consist of four steps. First, use the T-SQL command DBCC CHECKDB to check the existing database for errors. Then use a checksum to check that the backup file is not physically corrupt. Next, verify the backup by checking the header data which is needed for the restore process. Finally, regularly test backups by conducting a full restore on a separate dedicated server to test that all of the parts fit together and work correctly. My second recommendation would be to modify the storage system. Firstly, the file server that is located in the stationary cupboard is unlikely to be secure from intruders and hackers. This means that the backups are vulnerable to attacks. Make sure the file server is in a physically secure location, such as behind locks and security systems. It is also important to make sure that the external drives are accessed securely. This could be through end-to-end encryption and/or a dedicated, secure network channel. It may also be a good idea to encrypt the backups, so that if they are obtained by hackers, they will be unable to decrypt the backups and therefore the data will be safe. Keeping so many backups simply is not needed, so I think the company should implement a retention policy, deleting all full backups that are older than 6 months or 12 months, and only keeping the 3 or 4 latest differential backups. The company could also consider storing backups on the cloud, for its many benefits such as cost and in-built security and redundancy. My final recommendation would be to switch from a simple recovery model to a full or bulk-logged recovery model. This will mean that transaction log backups will be required. These transaction log backups should be taken regularly, such as at the end of every day. This will allow for a more up-to-date recovery of data when something goes wrong. A bulk-logged recovery model should be considered if there is a high number of bulk operations. This model reduces log space but can only be recovered to the end of the backup, whilst in a full recovery model, the database can be recovered to any specific point in time.

ii. First, I would restore the file crmful152.bak, since it is the latest full backup and it was taken before the database was corrupted. Then, I would restore the file crmdiff676.bak, since it is the latest differential backup that was taken before the corruption and after the latest full backup.

iii. A simple recovery model means there are no log backups. Changes since the most recent full/differential backup are unprotected. In the event of a disaster, those changes must be redone. This also means there is no possibility for point-in-time restores, and it is likely that some data will be lost if a disaster occurs.

b.

i. OLTP systems must be able to handle a large number of users accessing the database at one time. The database should be optimised for INSERTs, UPDATEs, and DELETEs. This is done through normalisation through to the 3rd normal form and potentially further. Finally, each transaction in the database should be ACID-compliant.

ii. OLAP systems should be optimised for SELECTs. This makes relational databases unsuitable since they often require complex queries and joins to select data. OLAP systems are often a centralised hub for data from a variety of sources. This means the data won't always follow a consistent schema, making relational databases not suitable. Finally, data in OLAP systems often undergo the ETL process, where the data is staged and transformed before being properly stored. Relational databases do not allow for staging of data and they are not designed to handle transformations.