



INFS1200 Assignment 2 – Module 3

Code Due: 9 May 2025 @ 3:00 PM AEST
Oral Assessment: Week 12, 19-23 May 2025

Weighting: 25%

Full Name	Student ID (8 digits)
Thomas Summerton	s4960678

Overview

The purpose of this assignment is to test your ability to use and apply SQL concepts to complete tasks in a real-world scenario. Specifically, this assessment will examine your ability to use SQL Data Manipulation Language to return specific subsets of information that exist in a database and Data Definition Language to create a new relational schema. The assignment is to be completed **individually**.

Submission

Assignment 2 is made up of two parts.

- **Part 1** will be submitted through an electronic marking tool called Gradescope, which will also be used for providing feedback.
- **Part 2** is an oral assessment that will be completed during an in-person interview with a tutor during a practical session in Week 12 (after your Gradescope submission).

Details below:

Part 1: Answer the questions on this task sheet and submit them through an electronic marking tool called Gradescope. For this assignment, you will need to submit two types of files to the portal:

- **Query Files:**
 - For each question in Sections A, B and C, and D where indicated, you are required to submit a separate *.sql* or *.txt* file which contains your SQL query solution for that question (submit only one of these file types).
 - Each file should only contain the SQL query(s) and no additional text or comments.
 - Each file should be named as per the *Filename* description in the question.
 - When submitting files to the autograder, select all of your *.sql* or *.txt* files as well as your *.pdf* file.
 - It is recommended to write queries in a SQL/text file and test them in your phpMyAdmin zones before copying them into the Word document.
 - Your queries must compile using **MySQL version 8.0**. This is the same DBMS software as is used on your *phpMyAdmin* zones.
 - You may use any MySQL function that have been used in class in addition to those specified in the questions.
- **Assignment PDF:**
 - Insert your answers for all Sections A-D into the template boxes on the Microsoft Word version of this assignment task sheet where appropriate. Export this document to a PDF and also upload it to the Gradescope autograder portal.
 - Only subsections of Section D will be partially hand-marked from your PDF submission, however this is also a backup for Sections A, B and C in case of autograder failure.
 - For Sections A, B and C, include a screenshot of the output of your query for each question in the space provided. Use your zones to generate the output.
 - **Please name your file 'Assignment_2.pdf'**. Please do not alter the format or layout of this document and ensure the name and SID boxes are completed.

Part 2 is an oral assessment, to verify your understanding of the code you submitted in Part 1 Sections A, B and C.

- This will be an oral critique of your submitted code. In a short meeting with a member of the teaching staff during Week 12 practical sessions, you will explain the work you have submitted in Part 1 and discuss your choices.
- All oral assessments must be given live and will be recorded by the teaching team (i.e. on Zoom) for archiving purposes.

Marking

Assignment 2 is worth **25 course marks**, and marking is made up of two parts. First, the marks available per section of Part 1 are as follows (note that INFS1200 differs from INFS7900):

	INFS1200
Section A – SQL DML (SELECT)	14 marks
Section B – SQL DML (UPDATE, INSERT, DELETE)	3 marks
Section C – SQL DDL	3 marks
Section D – Critical thinking	5 marks

Given these available marks, **students must also achieve a pass (+/-) in Part 2**, the oral critique, to be eligible to pass Assignment 2. Failure in Part 2 will result in your mark being capped at 12.5 marks (50% for this assignment).

Grading and Autograder feedback:

Sections A, B, C and parts of D of this assignment will be graded via an autograder deployed on Gradescope. However, we reserve the right to revert to hand marking using the PDF submission should the need arise.

When you submit your code, the autograder will provide you with two forms of immediate feedback:

- **File existence and compilation tests:** Your code will be checked to see if it compiles correctly. If it fails one or more compilation tests, the errors returned by the autograder will help you debug. Note that code that fails to compile will receive 0 marks. No marks are given for passing the compilation tests.
- **Column Domain checking:** The autograder will check whether the domain of the projected values for each attribute in your query results matches the expected domain. Make sure to carefully read the 'explanation' section in each question of the assignment, as it provides specific details about the expected domains for attributes in the result.

Submit your work to Gradescope early so you can use the test results to ensure your queries are compiling and you are on the right track. You will be able to resubmit to the autograder an unlimited number of times before the deadline.

Materials provided: You will be provided with the database schema and a simple data instance that will yield a result for every correct DML (SELECT) question. Visible test results in the autograder submission portal will check that the query compiles successfully. Because the autograder uses the same DBMS as your zones, you are encouraged to use your zones to develop your assignment answers.

Late penalties: Please consult the course profile for late penalties that apply to this assessment item.

Plagiarism

The University has strict policies regarding plagiarism. Penalties for engaging in unacceptable behaviour range from loss of grades in a course through to expulsion from UQ. You are required to read and understand the policies on academic integrity and plagiarism in the course profile (Section 6.1). If you have any questions regarding an acceptable level of collaboration with your peers, please see either the lecturer or your tutor for guidance. Remember that ignorance is not a defence!

You are permitted to use generative AI tools to help you complete this assessment task. However, if you do, please provide complete copies of your interactions with the AI tool in the space provided at the end of your submission. Please note that if you use generative AI but fail to acknowledge this by attaching your interaction to the end of the assignment, it will be considered misconduct, as you are claiming credit for work that is not your own.

Task

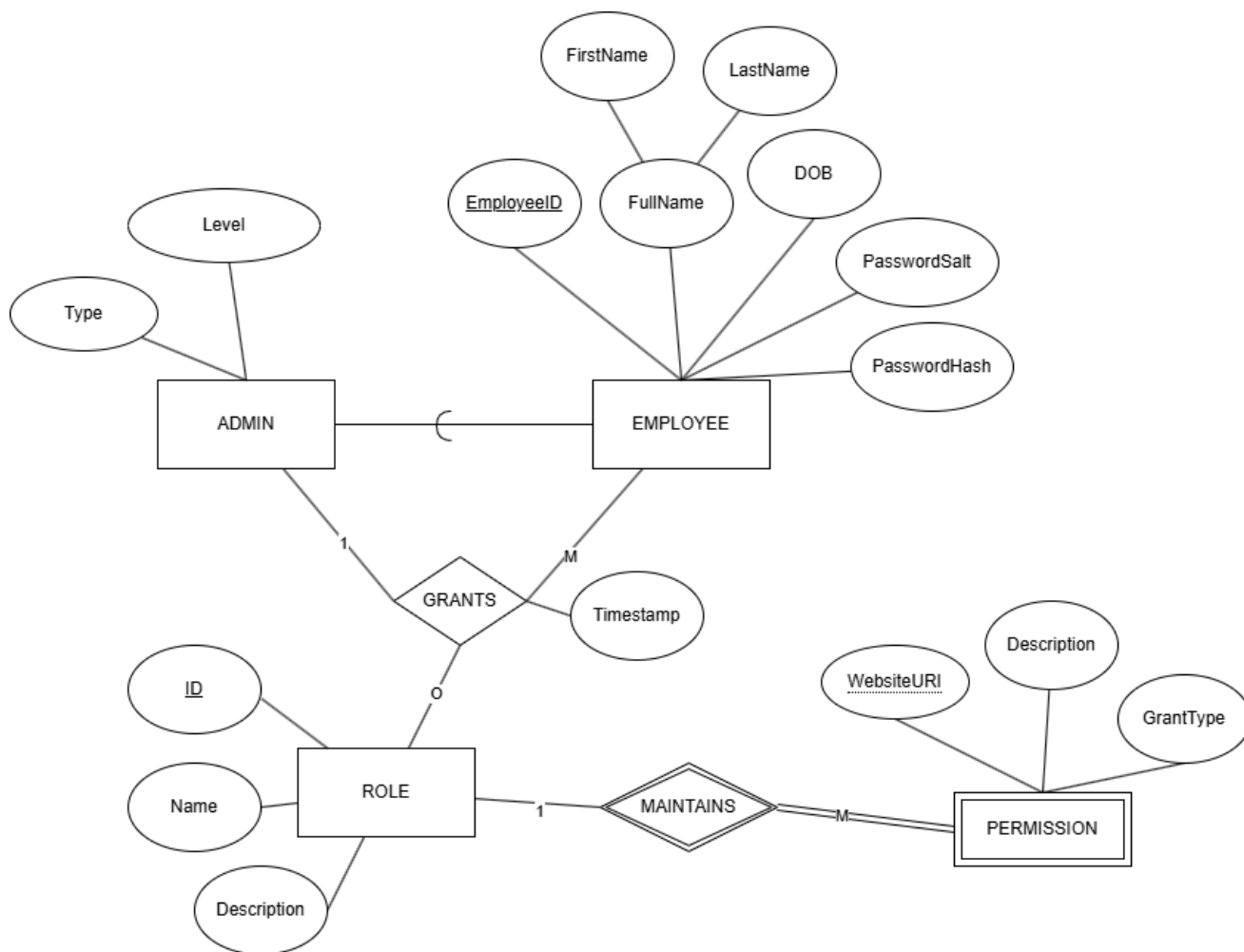
For this assignment, you will be presented with the simplified schema of an authorisation application.

BestTechLtd have designed and developed a simplistic authorisation management system to provide secure and sufficient access control within their organisational network.

When a new employee joins an organisation using SecureAccess, they are registered and their personal details are stored in the Employee table. Administrative employees, whose additional information is stored in the AdministrativeEmployee table, can then assign appropriate roles to these employees through a role-granting process. Each role, defined in the Role table, comes with specific permissions that determine which company websites and resources the employee can access. This hierarchical permission structure ensures that employees only have access to the resources necessary for their job functions. BestTechLtd manage their authorisation by storing employee and permission data in a relational database management system (RDBMS) with the following schema:

- The **Employee** table records staff-specific information.
- **Administration** employees are a specific type of employee that have authority to grant authorisation roles to employees.
- The **Role** table records typical roles that might exist within a business and is associated with the necessary permissions that a user (Employee) of the role will need.
- The **Permission** table maintains a record of all the websites in a company's network. A website URI and RoleID associated with the website means that the Role is granted permission to access, retrieve resources and perform operations on the website.

ERD for the BestTechLtd schema:



Relational Schema:

Employee [EmployeeID, FirstName, LastName, DOB, PasswordHash, PasswordSalt]

AdministrativeEmployee [EmployeeID, Level, Type]

Role [RoleID, Name, Description]

RoleGranting [EmployeeID, RoleID, AdministrationID, Timestamp]

Permission [WebsiteURI, RoleID, GrantType, Description]

Foreign Key References:

AdministrativeEmployee.EmployeeID references Employee.EmployeeID

RoleGranting.EmployeeID references Employee.EmployeeID

RoleGranting.RoleID references Role.RoleID

~~RoleGranting.AdministrationID references AdministrativeEmployee.AdministrationID (correction 23/04/25)~~

RoleGranting.AdministrationID references AdministrativeEmployee.EmployeeID

Permission.RoleID references Role.RoleID

For this assignment you will be required to write SQL queries to complete the tasks below.

- Answer the queries using only the information provided in the **Task** box.
- Use the **SQL Solution** box provided to record your answer code. It is recommended to write queries in the SQL/text file and test them in your phpMyAdmin zones before pasting them here.
- Use the **Output Screenshot** box to record the output of your query (generated in your zones before submission).

Example Query																																																												
Task	Retrieve all information about all employees.																																																											
SQL Solution	SELECT * FROM Employee;																																																											
Output Screenshot	<table><tr><th>EmployeeID</th><th>FirstName</th><th>LastName</th><th>DOB</th><th>PasswordHash</th><th>PasswordSalt</th></tr><tr><td>E0001</td><td>Mehdi</td><td>Rahman</td><td>1985-06-15</td><td>3d6f0a5b31287f5b2b31d764a3e6b09a9c2bfa6e5d1e3455b7...</td><td>xYz89&@p</td></tr><tr><td>E0002</td><td>Mei</td><td>Chen</td><td>1990-08-20</td><td>a8b9c0d4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b617...</td><td>kLm34^7pQz</td></tr><tr><td>E0003</td><td>Aarav</td><td>Chen</td><td>1988-03-10</td><td>b2cd4ef5b61718293ab2cd4ef5b61718293ab2cd4ef5b61718...</td><td>mNop8!@hG</td></tr><tr><td>E0004</td><td>Fatima</td><td>NULL</td><td>1995-11-25</td><td>c4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b61718293a...</td><td>qRsT56%\$w</td></tr><tr><td>E0005</td><td>Sofia</td><td>Gonzalez</td><td>1987-09-12</td><td>ed8cd892c82eb261f662663dcae5c22f1fca2c60c78f0dc9...</td><td>!%pgH51bl1</td></tr><tr><td>E0006</td><td>John</td><td>Stevens</td><td>1992-04-03</td><td>41854b9416e7d992fca5f489ad5a5334eb4eda636cf0c4314...</td><td>LW8cUj33Qxw</td></tr><tr><td>E0007</td><td>Elena</td><td>Popov</td><td>1989-12-30</td><td>bc9256e23093d8f31e418931e3e68da5031bb3aac1c1ae8201...</td><td>3b3SZgmAlp</td></tr><tr><td>E0008</td><td>Omar</td><td>NULL</td><td>1996-07-21</td><td>2d03d5085f6b9c78a1625302a452cbfdc5a11f7b3cd7e26708...</td><td>ibOWLI*u</td></tr></table>						EmployeeID	FirstName	LastName	DOB	PasswordHash	PasswordSalt	E0001	Mehdi	Rahman	1985-06-15	3d6f0a5b31287f5b2b31d764a3e6b09a9c2bfa6e5d1e3455b7...	xYz89&@p	E0002	Mei	Chen	1990-08-20	a8b9c0d4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b617...	kLm34^7pQz	E0003	Aarav	Chen	1988-03-10	b2cd4ef5b61718293ab2cd4ef5b61718293ab2cd4ef5b61718...	mNop8!@hG	E0004	Fatima	NULL	1995-11-25	c4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b61718293a...	qRsT56%\$w	E0005	Sofia	Gonzalez	1987-09-12	ed8cd892c82eb261f662663dcae5c22f1fca2c60c78f0dc9...	!%pgH51bl1	E0006	John	Stevens	1992-04-03	41854b9416e7d992fca5f489ad5a5334eb4eda636cf0c4314...	LW8cUj33Qxw	E0007	Elena	Popov	1989-12-30	bc9256e23093d8f31e418931e3e68da5031bb3aac1c1ae8201...	3b3SZgmAlp	E0008	Omar	NULL	1996-07-21	2d03d5085f6b9c78a1625302a452cbfdc5a11f7b3cd7e26708...	ibOWLI*u
EmployeeID	FirstName	LastName	DOB	PasswordHash	PasswordSalt																																																							
E0001	Mehdi	Rahman	1985-06-15	3d6f0a5b31287f5b2b31d764a3e6b09a9c2bfa6e5d1e3455b7...	xYz89&@p																																																							
E0002	Mei	Chen	1990-08-20	a8b9c0d4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b617...	kLm34^7pQz																																																							
E0003	Aarav	Chen	1988-03-10	b2cd4ef5b61718293ab2cd4ef5b61718293ab2cd4ef5b61718...	mNop8!@hG																																																							
E0004	Fatima	NULL	1995-11-25	c4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b61718293a...	qRsT56%\$w																																																							
E0005	Sofia	Gonzalez	1987-09-12	ed8cd892c82eb261f662663dcae5c22f1fca2c60c78f0dc9...	!%pgH51bl1																																																							
E0006	John	Stevens	1992-04-03	41854b9416e7d992fca5f489ad5a5334eb4eda636cf0c4314...	LW8cUj33Qxw																																																							
E0007	Elena	Popov	1989-12-30	bc9256e23093d8f31e418931e3e68da5031bb3aac1c1ae8201...	3b3SZgmAlp																																																							
E0008	Omar	NULL	1996-07-21	2d03d5085f6b9c78a1625302a452cbfdc5a11f7b3cd7e26708...	ibOWLI*u																																																							

Section A – SQL DML (SELECT)

Question A1																			
Task	Return the distinct first name and last name of all employees, in descending alphabetical order of their last name.																		
Filename	A1.sql or A1.txt																		
Columns returned	This query must return two columns; the first name and the last name of employees, in that order.																		
SQL Solution	<pre>SELECT FirstName, LastName from Employee ORDER BY LastName ASC;</pre>																		
Output Screenshot	<table><thead><tr><th>FirstName</th><th>LastName</th></tr></thead><tbody><tr><td>Fatima</td><td>NULL</td></tr><tr><td>Omar</td><td>NULL</td></tr><tr><td>Mei</td><td>Chen</td></tr><tr><td>Jeremy</td><td>Chen</td></tr><tr><td>Sofia</td><td>Gonzalez</td></tr><tr><td>Elena</td><td>Popov</td></tr><tr><td>Mehdi</td><td>Rahman</td></tr><tr><td>John</td><td>Stevens</td></tr></tbody></table>	FirstName	LastName	Fatima	NULL	Omar	NULL	Mei	Chen	Jeremy	Chen	Sofia	Gonzalez	Elena	Popov	Mehdi	Rahman	John	Stevens
FirstName	LastName																		
Fatima	NULL																		
Omar	NULL																		
Mei	Chen																		
Jeremy	Chen																		
Sofia	Gonzalez																		
Elena	Popov																		
Mehdi	Rahman																		
John	Stevens																		



Question A2																											
Task	For each website URI, find the number of roles that can access it.																										
Columns returned	This query must return two columns; the website URI, and the number of Roles which access a website URI, in that order.																										
Filename	A2.sql or A2.txt																										
SQL Solution	<pre>SELECT WebsiteURI, COUNT(*) as NumberOfRoles FROM Permission GROUP By WebsiteURI;</pre>																										
Output Screenshot	<table><thead><tr><th>WebsiteURI</th><th>NumberOfRoles</th></tr></thead><tbody><tr><td>https://aurion.besttechltd.com</td><td>2</td></tr><tr><td>https://bamboohr.besttechltd.com</td><td>2</td></tr><tr><td>https://besttechltd/financialperformance.tech</td><td>1</td></tr><tr><td>https://datadog.besttechltd.com</td><td>2</td></tr><tr><td>https://github.besttechltd.com</td><td>2</td></tr><tr><td>https://grafana.besttechltd.com</td><td>2</td></tr><tr><td>https://hubspot.besttechltd.com</td><td>2</td></tr><tr><td>https://jenkins.besttechltd.com</td><td>2</td></tr><tr><td>https://quickbooks.besttechltd.com</td><td>2</td></tr><tr><td>https://sonarqube.besttechltd.com</td><td>2</td></tr><tr><td>https://workday.besttechltd.com</td><td>2</td></tr><tr><td>https://xero.besttechltd.com</td><td>2</td></tr></tbody></table>	WebsiteURI	NumberOfRoles	https://aurion.besttechltd.com	2	https://bamboohr.besttechltd.com	2	https://besttechltd/financialperformance.tech	1	https://datadog.besttechltd.com	2	https://github.besttechltd.com	2	https://grafana.besttechltd.com	2	https://hubspot.besttechltd.com	2	https://jenkins.besttechltd.com	2	https://quickbooks.besttechltd.com	2	https://sonarqube.besttechltd.com	2	https://workday.besttechltd.com	2	https://xero.besttechltd.com	2
WebsiteURI	NumberOfRoles																										
https://aurion.besttechltd.com	2																										
https://bamboohr.besttechltd.com	2																										
https://besttechltd/financialperformance.tech	1																										
https://datadog.besttechltd.com	2																										
https://github.besttechltd.com	2																										
https://grafana.besttechltd.com	2																										
https://hubspot.besttechltd.com	2																										
https://jenkins.besttechltd.com	2																										
https://quickbooks.besttechltd.com	2																										
https://sonarqube.besttechltd.com	2																										
https://workday.besttechltd.com	2																										
https://xero.besttechltd.com	2																										


Question A3													
Task	For every role in the database, find the number of websites they have access to.												
Columns returned	This query must return two columns; the RoleID, and the number of permissions they have, in that order.												
Filename	A3.sql or A3.txt												
SQL Solution	<pre>SELECT RoleID, COUNT(*) as NumberOfPermissions FROM Permission GROUP BY RoleID</pre>												
Output Screenshot	<table> <tr> <th>RoleID</th><th>NumberOfPermissions</th></tr> <tr> <td>1</td><td>11</td></tr> <tr> <td>2</td><td>5</td></tr> <tr> <td>3</td><td>3</td></tr> <tr> <td>4</td><td>3</td></tr> <tr> <td>5</td><td>1</td></tr> </table>	RoleID	NumberOfPermissions	1	11	2	5	3	3	4	3	5	1
RoleID	NumberOfPermissions												
1	11												
2	5												
3	3												
4	3												
5	1												

Question A4																		
Task	Return the distinct role ID, name and descriptions of all roles which permit access to commercial websites (where the URI ends with “.com”).																	
Columns returned	This query must return three columns; the role ID, the role’s name, and the role’s Description, in that order.																	
Filename	A4.sql or A4.txt																	
SQL Solution	<pre>SELECT * FROM Role WHERE RoleID in(SELECT RoleID FROM Permission WHERE WebsiteURI LIKE ‘%.com’);</pre>																	
Output Screenshot	<table><tr><th>RoleID</th><th>Name</th><th>Description</th></tr><tr><td>1</td><td>DBA</td><td>Manages database structures and access control</td></tr><tr><td>2</td><td>Developer</td><td>Responsible for writing, testing, and maintaining ...</td></tr><tr><td>3</td><td>HR</td><td>Manages employee relations, recruitment, and organ...</td></tr><tr><td>4</td><td>Finance</td><td>Handles financial planning, budgeting, and company...</td></tr></table>			RoleID	Name	Description	1	DBA	Manages database structures and access control	2	Developer	Responsible for writing, testing, and maintaining ...	3	HR	Manages employee relations, recruitment, and organ...	4	Finance	Handles financial planning, budgeting, and company...
RoleID	Name	Description																
1	DBA	Manages database structures and access control																
2	Developer	Responsible for writing, testing, and maintaining ...																
3	HR	Manages employee relations, recruitment, and organ...																
4	Finance	Handles financial planning, budgeting, and company...																

Question A5																																																
Task	Return all information about all employees that have not been granted a role from an Administrator with type ‘ProductEngineer’.																																															
	Restriction: You must use a sub-query to answer this question.																																															
Columns returned	This query must return all fields from the employee table.																																															
Filename	A5.sql or A5.txt																																															
	<pre>Select * FROM Employee WHERE EmployeeID IN(SELECT EmployeeID FROM RoleGranting WHERE AdministrationID != “E0002”);</pre>																																															
Output Screenshot	<table><tr><th>EmployeeID</th><th>FirstName</th><th>LastName</th><th>DOB</th><th>PasswordHash</th><th>PasswordSalt</th></tr><tr><td>E0001</td><td>Mehdi</td><td>Rahman</td><td>1985-06-15</td><td>3d6f0a5b31287f5b2b31d764a3e6b09a9c2bfa6e5d1e3455b7...</td><td>xYz89&@p</td></tr><tr><td>E0003</td><td>Jeremy</td><td>Chen</td><td>1988-03-10</td><td>b2cd4ef5b61718293ab2cd4ef5b61718293ab2cd4ef5b61718...</td><td>mNop8!@hG</td></tr><tr><td>E0004</td><td>Fatima</td><td>NULL</td><td>1995-11-25</td><td>c4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b61718293a...</td><td>qRsT56%\$w</td></tr><tr><td>E0005</td><td>Sofia</td><td>Gonzalez</td><td>1987-09-12</td><td>ed8cd892c82ebeb261f662663dcae5c22f1fca2c60c78f0dc9...</td><td>!%pgH51b!l</td></tr><tr><td>E0007</td><td>Elena</td><td>Popov</td><td>1989-12-30</td><td>bc9256e23093d8f31e418931e3e68da5031bb3aac1c1ae8201...</td><td>3b3SZgmAlp</td></tr><tr><td>E0008</td><td>Omar</td><td>NULL</td><td>1996-07-21</td><td>2d03d5085f6b9c78a1625302a452cbfdc5a11f7b3cd7e26708...</td><td>ibOWLI*u</td></tr></table>						EmployeeID	FirstName	LastName	DOB	PasswordHash	PasswordSalt	E0001	Mehdi	Rahman	1985-06-15	3d6f0a5b31287f5b2b31d764a3e6b09a9c2bfa6e5d1e3455b7...	xYz89&@p	E0003	Jeremy	Chen	1988-03-10	b2cd4ef5b61718293ab2cd4ef5b61718293ab2cd4ef5b61718...	mNop8!@hG	E0004	Fatima	NULL	1995-11-25	c4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b61718293a...	qRsT56%\$w	E0005	Sofia	Gonzalez	1987-09-12	ed8cd892c82ebeb261f662663dcae5c22f1fca2c60c78f0dc9...	!%pgH51b!l	E0007	Elena	Popov	1989-12-30	bc9256e23093d8f31e418931e3e68da5031bb3aac1c1ae8201...	3b3SZgmAlp	E0008	Omar	NULL	1996-07-21	2d03d5085f6b9c78a1625302a452cbfdc5a11f7b3cd7e26708...	ibOWLI*u
EmployeeID	FirstName	LastName	DOB	PasswordHash	PasswordSalt																																											
E0001	Mehdi	Rahman	1985-06-15	3d6f0a5b31287f5b2b31d764a3e6b09a9c2bfa6e5d1e3455b7...	xYz89&@p																																											
E0003	Jeremy	Chen	1988-03-10	b2cd4ef5b61718293ab2cd4ef5b61718293ab2cd4ef5b61718...	mNop8!@hG																																											
E0004	Fatima	NULL	1995-11-25	c4e5f61718293ab2cd4ef5b61718293ab2cd4ef5b61718293a...	qRsT56%\$w																																											
E0005	Sofia	Gonzalez	1987-09-12	ed8cd892c82ebeb261f662663dcae5c22f1fca2c60c78f0dc9...	!%pgH51b!l																																											
E0007	Elena	Popov	1989-12-30	bc9256e23093d8f31e418931e3e68da5031bb3aac1c1ae8201...	3b3SZgmAlp																																											
E0008	Omar	NULL	1996-07-21	2d03d5085f6b9c78a1625302a452cbfdc5a11f7b3cd7e26708...	ibOWLI*u																																											

Question A6									
Task	Return the Date of Birth (DOB), first and last name of the youngest employee(s)								
Columns returned	This query must return three columns; the date of birth, the first name, and the last name of the employee(s), in that order.								
Filename	A6.sql or A6.txt								
SQL Solution	<pre>SELECT FirstName, LastName, DOB FROM Employee WHERE DOB IN(SELECT MAX(DOB) FROM Employee);</pre>								
Output Screenshot	<table><tr><th>FirstName</th><th>LastName</th><th>DOB</th></tr><tr><td>Omar</td><td>NULL</td><td>1996-07-21</td></tr></table>			FirstName	LastName	DOB	Omar	NULL	1996-07-21
FirstName	LastName	DOB							
Omar	NULL	1996-07-21							

Question A7									
Task	<p>Find and return the first name and last name of all employees that have access to at least all the website URIs that Elena Popov has access to.</p> <p>You may assume only one Elena Popov exists in the database.</p>								
Columns returned	This query must return two columns; the first name and the last name of the employees, in that order.								
Filename	A7.sql or A7.txt								
	<pre>SELECT FirstName, LastName FROM Employee Where EmployeeID IN(SELECT EmployeeID FROM RoleGranting WHERE RoleID IN(SELECT RoleID FROM RoleGranting WHERE EmployeeID = 'E0007')) and EmployeeID != 'E0007';</pre>								
Output Screenshot	<table> <thead> <tr> <th>FirstName</th><th>LastName</th></tr> </thead> <tbody> <tr> <td>Mei</td><td>Chen</td></tr> <tr> <td>Fatima</td><td>NULL</td></tr> <tr> <td>John</td><td>Stevens</td></tr> </tbody> </table>	FirstName	LastName	Mei	Chen	Fatima	NULL	John	Stevens
FirstName	LastName								
Mei	Chen								
Fatima	NULL								
John	Stevens								

Question A8	
Task	Return the EmployeeID of the Administrator(s) that have granted the most permissions of any other Administrator. Restriction: You must use one or more views in your answer
Columns returned	This query must return two columns; the employeeID(s) and the number of permissions granted for the employee(s).
Filename	A8.sql or A8.txt
Note	If one Administrator granted the same role twice, then all the permissions of that role must be counted twice.
SQL Solution	<pre> CREATE VIEW NumPermissionsGivenByRoles AS SELECT RoleID, COUNT(*) as NumberOfPermissions FROM Permission GROUP BY RoleID; CREATE VIEW RolesGivenByAdmin as SELECT AdministrationID, RoleID, NumberOfPermissions From RoleGranting JOIN NumPermissionsGivenByRoles USING (RoleID); CREATE VIEW TOTALS AS SELECT AdministrationID, SUM(NumberOfPermissions) as ROLESGRANTED FROM RolesGivenByAdmin GROUP BY AdministrationID; SELECT AdministrationID FROM TOTALS WHERE ROLESGRANTED IN (SELECT MAX(ROLESGRANTED) FROM TOTALS) </pre>
Output Screenshot	

Question A9

Task	The co-founders of bestTechLtd can be identified as either: <div>1. Administrative Employees with a ‘LegacyEngineer’ type;</div> <div>2. Employees who have been granted the “DBA” role; or</div> <div>3. Permissions to access only one website URI, which is https://besttechltd/financialperformance.tech</div> Return the EmployeeID, first and last name of all co-founders. Restriction: You must use at least one set operation in your answer.														
Columns returned	This query must return three columns; the employeeID, the first name and the last name of employee(s), in that order.														
Filename	A9.sql or A9.txt														
Hint	You may want to use one or more views in your answer.														
SQL Solution	<pre>SELECT EmployeeID, FirstName, LastName From Employee WHERE EmployeeID IN (SELECT EmployeeID FROM AdministrativeEmployee WHERE Type = 'LegacyEngineer') OR EmployeeID IN (SELECT EmployeeID FROM RoleGranting WHERE RoleID = '1') OR EmployeeID IN (SELECT EmployeeID FROM RoleGranting WHERE RoleID IN (SELECT RoleID FROM Permission WHERE WebsiteURI = 'https://besttechltd/financialperformance.tech'))</pre>														
Output Screenshot	<table><tr><th>EmployeeID</th><th>FirstName</th><th>LastName</th></tr><tr><td>E0001</td><td>Mehdi</td><td>Rahman</td></tr><tr><td>E0006</td><td>John</td><td>Stevens</td></tr><tr><td>E0008</td><td>Omar</td><td>NULL</td></tr></table>			EmployeeID	FirstName	LastName	E0001	Mehdi	Rahman	E0006	John	Stevens	E0008	Omar	NULL
EmployeeID	FirstName	LastName													
E0001	Mehdi	Rahman													
E0006	John	Stevens													
E0008	Omar	NULL													

Section B – SQL DML (UPDATE, DELETE, INSERT)

Question B1	
Task	Revoke the role that was granted to John Stevens by Mei Chen at any time during the day of 22 nd July, 2024. You can assume such a role exists, and only one John Stevens and Mei Chen exist in the database.
Filename	B1.sql or B1.txt
SQL Solution	<pre>DELETE FROM RoleGranting WHERE EmployeeID IN (Select EmployeeID From Employee WHERE FirstName = 'John' and LastName ='Stevens') and AdministrationID IN (Select EmployeeID From Employee WHERE FirstName = 'Mei' and LastName ='Chen') and Timestamp LIKE '%2024-07-22%'</pre>

Question B2	
Task	<p>A new employee has been onboarded to BestTechLtd. The employee (ID E0024) is named James Moran, Born 21 November 2001, with a password hash e7cf3ef8d8aac2c1c93963e7a58b7b62ade24d0d0ba2c8ae0f7fb6c8b0aa0332 and passwordSalt D;%yL9TS:5PaIS/d</p> <p>James Moran is granted all the roles that employee Sofia Gonzalez has, and is granted them by the admin E0001.</p> <p>Insert new records with James' details and roles as given above. You can assume only one Sofia Gonzales exists in the database.</p>
Filename	B2.sql or B2.txt
SQL Solution	<pre>INSERT INTO Employee (EmployeeID, FirstName, LastName, DOB, PasswordHash, PasswordSalt) VALUES ('E0024', 'James', 'Moran', '2001-11-21', 'e7cf3ef8d8aac2c1c93963e7a58b7b62ade24d0d0ba2c8ae0f7fb6c8b0aa0332', 'D;%yL9TS:5PaIS/d'); INSERT INTO RoleGranting (EmployeeID, RoleID, AdministrationID, Timestamp) SELECT 'E0024', RoleID , 'E0001', '2025-05-08 09:00:00' From RoleGranting WHERE EmployeeID = 'E0005';</pre>

Section C – SQL DDL

Question C1	
Task	<p>Before BestTechLtd created their authorisation system, employees were assigned only a single role in their company, in addition to a unique access token which granted access for that role. To maintain compatibility with their older technologies, a new relation must be made to track legacy employees and their tokens.</p> <p>Create a new relation LegacyEmployee, which is a <i>specialised</i> type of employee which maintains additional characteristics of an Employee and must be stored in a separate relation. The details are as follows:</p> <ul style="list-style-type: none">• <i>LegacyEmployeeID</i>: A unique reference to the employeeID stored in the authorisation database.• <i>GrantAccessToken</i>: an SHA-256 token to grant access to old websites.• <i>RoleID</i>: A reference to the original role of the legacy employee. This reference cannot be null. <p>Write a SQL DDL query to implement the relation LegacyEmployee. Make sure to follow the naming syntax exactly and add all required constraints.</p>
Filename	C1.sql or C1.txt
SQL Solution	<pre>CREATE TABLE LegacyEmployee (LegacyEmployeeID varchar(255), GrantAccessToken Binary(32), RoleID int NOT NULL, PRIMARY KEY (LegacyEmployeeID), FOREIGN KEY (LegacyEmployeeID) REFERENCES Employee(EmployeeID))</pre>

Question C2	
Task	Add a constraint to ensure that the highest administration level is 10. Name the constraint <i>AdministrationMax</i> . (You can assume that no existing employees have an admin level higher than 10.)
Explanation	The following resources may be useful when answering this question: Check constraints
Filename	C2.sql or C2.txt
SQL Solution	<pre>ALTER TABLE AdministrativeEmployee ADD CONSTRAINT AdministrationMax CHECK (Level <= 10)</pre>

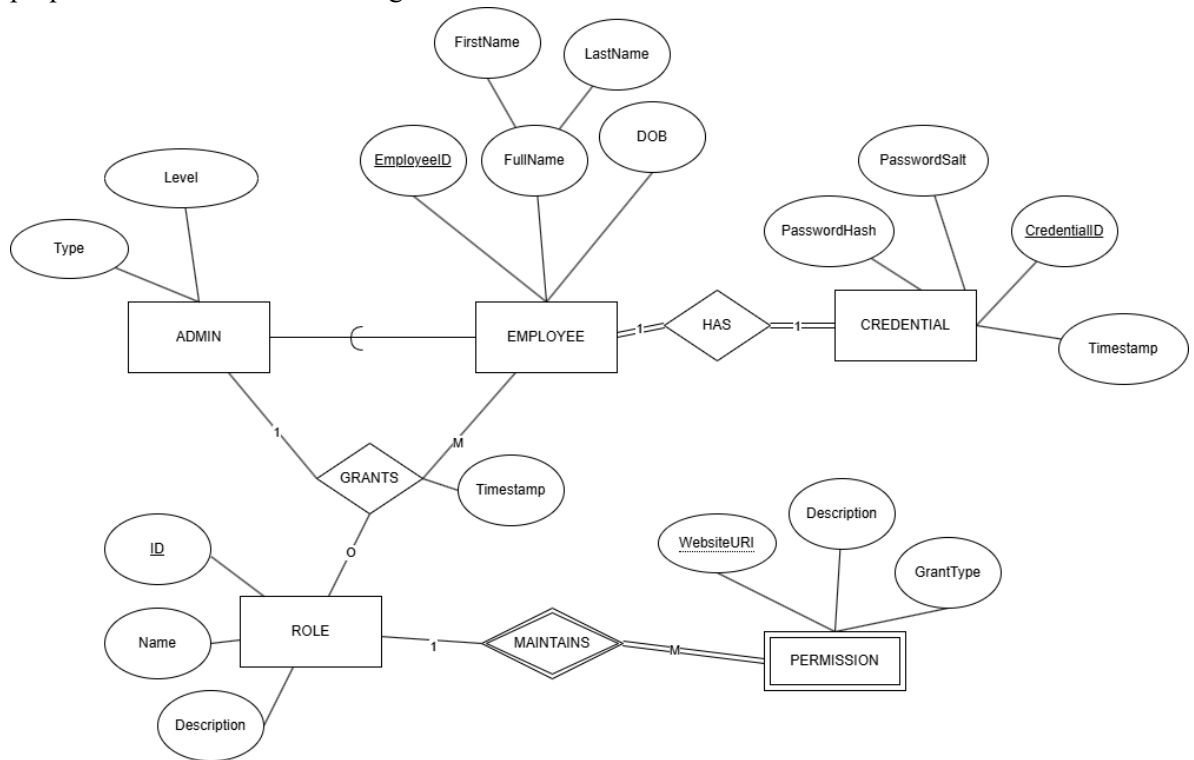
Section D – Critical Thinking

In this section, you will receive theoretical situations related to the UoD mentioned in the task description. Your task is to offer strategies to tackle the situation and write SQL queries to execute the approaches.

Question D.1

Task

The database architects at BestTechLtd wish to change the authorisation relational schema and have proposed a new design to improve security management. In this new design, password credentials will be migrated from employee data into a separate relation called **Credential**. A new ERD is proposed to illustrate these changes:



The database architects have provided the following instructions for implementing both structural changes and the migration of existing data:

Required Operations:

- Create a new table called **Credential** with appropriate fields as outlined in the ERD. These fields can be in any order in the table;
- Migrate all existing credential data from the Employee table to the new Credential table, where:
 - Each *CredentialID* should be formed by prepending 'C' to the corresponding EmployeeID (e.g., EmployeeID E001 has *CredentialID* CE001)
 - A reference to the employee must be added to the **Credential** relation and named *EmployeeID*
 - The *PasswordSalt* must be transferred from the original passwordSalt in the **Employee** table
 - The *PasswordHash* must be transferred from the original passwordHash in the **Employee** table
 - The *Timestamp* is the date and time when these new records are immediately migrated into the database
 - Ensure the resulting relational schema matches the proposed ERD. *Hint:* Pay particular attention to the structure and constraints given in the new ERD.

	<p>Task:</p> <p>Write sequence of SQL statements that perform both the structural changes to the relations and the data migration. Your solution must ensure that no credential data is lost during this transition.</p> <p>Paste your SQL solution into D1.sql and the first text box below.</p> <p>In the second text box, briefly describe (in 200 words or less) how the changes you have made maintain relational integrity.</p>
Filename (SQL only)	D1.sql or D1.txt
SQL Solution	<pre> CREATE TABLE Credential(CredentialID varchar(255), EmployeeID varchar(255), PasswordSalt varchar(255), PasswordHash varchar(255), Timestamp DATETIME DEFAULT CURRENT_TIMESTAMP, PRIMARY KEY (CredentialID), FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)); INSERT INTO Credential (CredentialID, EmployeeID, PasswordSalt, PasswordHash) SELECT CONCAT('C',EmployeeID), EmployeeID, PasswordSalt, PasswordHash FROM Employee; ALTER TABLE Employee DROP COLUMN PasswordHash, DROP Column PasswordSalt </pre>
Explanation	<p>The changes made were to create a new table for the credentials with the fields CredentialID, EmployeeID, PasswordSalt, PasswordHash and Timestamp. CredentialID was considered the primary key as the UOD called for it to be unique. The EmployeeID was added as there had to be a foreign key within the table and a foreign key cannot be modified so to get the credentialID with the prepended 'c' two separate fields were required. Timestamp is set to default to the time of entity insertion. Finally, the two password fields were removed from the employee table so that they were only present within the Credentials table. Constraints were upheld in this transfer as aforementioned the EmployeeID was preserved and a new field was created to prepend the C for the credential ID ensuring no foreign key constraints were violated. Additionally, PasswordSalt and PasswordHash were not added as foreign keys as the UOD stated that they were to be removed from the Employee table and therefore a foreign key constraint would be violated if they were added as foreign keys.</p>

Question D.2	
Task	<p>BestTechLtd's current authorization system tracks roles granted to employees through their RoleGranting relation but lacks the ability to record when roles are revoked.</p> <p>The company now wishes to enhance the existing schema to retain a complete history of both role assignments and revocations. This means when a role is granted to an employee, it will continue to be recorded, but additionally, when a role is taken away from an employee, a record of this removal should also be preserved in the database.</p> <p>In 200 words or less, describe how you would extend the current implementation to meet these new business requirements while maintaining relational integrity constraints. Your solution should build upon rather than replace the existing functionality.</p> <p>SQL scripts are not required but you may include them if they help illustrate your proposed changes.</p>
Filename	No SQL submission for this question
Solution	<p>To start I would create a new relation which is RolesRevoked, similar to RolesGranted it would contain all the same fields (EmployeeID, RoleID, AdministrationID, Timestamp). To transfer the data over I would use</p> <pre>SELECT INTO RolesRevoked FROM RolesGranted</pre> <p>which would get the fields from RolesGranted and transfer them over to the new RolesRevoked table. Then instead of transferring the Timestamp that was there initially I would use Current_Timestamp to get the date and time of when the revocation occurred.</p> <p>Then to remove the entity from RolesGranted I would use</p> <pre>DELETE FROM RolesGranted WHERE EmployeeID = "" and RoleID = "" and AdministrationID = "" and Timestamp = ""</pre> <p>The values within the where filter would be inserted with the specifics of the entity that is being revoked</p> <p>This would maintain all relational integrity as RolesRevoked is not interacting with any data outside of RolesGranted and RolesGranted does not have any dependants within the database, hence if an entity is removed from RolesGranted, no constraint will be violated.</p>

--	--