Project report

Database management systems – Course task

Table Of contents

Contents

Introduction

This project report is part of the 'Database Management System' course and aims to clarify the tasks I have completed and the actions I have taken in relation to the databases. I have organized this document so that each task is addressed in its own chapter, with explanations and code solutions provided under each section. I have completed all assigned tasks for this project, as well as all homework assignments, and I am expecting a grade of 5 from this course.

Instead of adding code to this document, I have made a GitHub repository, where you can find all my SQL commands. I did this to improve the readability of the SQL commands. Each task and its solution have a file related to it.

Note: I did use AI as a tool with the writing of this report. Generative AI was used to help with my texts readability and to fix grammatical errors. The use of the AI tools is in line with LUT's code.

Tasks and Solutions

Task 1

Task: "(8 %) Create four views to provide for the company superiors and management."

I added five different views that hopefully align with the task's specifications. I created an extra view just in case, since I wasn't sure about what "not including linking tables" meant, since most of the tables are linked by a foreign key. Nonetheless, all these views I made give valuable information to management of a company. Here's the views:

- employee_details
- department_details
- employee_statistics
- contract_details
- project_details

For more information about these views, please refer to the "management_views.sql" file.

Task 2

Task: "(15 %) Create three triggers for the database:"

I successfully created all three triggers: 'check_same_skill', 'assign_employees_trigger', and 'check_contract'.

The first trigger, 'check_same_skill', queries the database to check if there are any existing skills that match the new skill's name. If there are any matches, it raises an exception.

The second trigger, 'assign_employees_trigger', retrieves the country of the customer and randomly selects three employees from the same country to be inserted to the 'project_role' table.

The last trigger, 'check_contract', checks if the ‘employee’ tables ‘contract_type’ has been updated. If so, it sets the contract start date to the current date. If the contract type is temporary, it sets the end date to two years from now. Initially, there was a conflict with the procedure that added three months to every temporary contract. This conflict was resolved by modifying the triggers to only trigger when the 'contract_type' column is updated, rather than the entire 'employee' table update.

The SQL commands for these triggers can be found in the 'triggers.sql' file."

Task 3

Task: “(9 %) Create three procedures for the database:”

I was able to successfully implement all 3 procedures, which work as intended.

The first procedure simply updates every employees salary to what matches the base salary of their job title.

The second procedure updates the contract end column to add three months if the employees contract type is temporary. This was the procedure which didn’t work due to a bug with the 3rd trigger. this bug has been fixed since, like I said in the previous chapter.

The last procedure increases everybody's salary bye give an percentage this percentage is given as how much you want to increase it for example if you want to increase the salaries by 10% you will give 0.1 as a parameter there is also a parimeter for salary limit in which if the new salary is higher than it then the salary will be set to the salary limit. To check which one is lower it uses that LEAST() function.

The SQL commands for these procedures can be found in the procedures.sql' file."

Task 4

Task: “(8 %) Partition two of the following tables to at least three partitions (excluding default partition):”

This one was the hardest task, because I couldn't realise A practical solutions where these partitions would have an actual purpose without modifying the original tables. Eventually, I decided to create 3 partitions for the employee and the project table in which both the partitions take 500 entries by their respective ideas from the original table.

After creating the partitions, I insert the data from the original tables according to the partitions specifications.

The SQL commands for this task can be found in the partitions.sql.

Task 5

Task: “(6 %) Create access rights:”

I managed to create all 3 roles. I granted Admin with superuser permisison, the employee with ‘SELECT’ permission on all table on the schema. Finally, I gave the trainee role to ‘SELECT’ permission on the specified tables and columns.

The SQL commands for this task can be found in the file role_creation.sql.

Task 6

Task: “(4 %) Do the following changes to the database:”

This task was the easiest and all its functionality is self-explanatory from the task description. Please refer to the file basic_changes.SQL.