**Stored Procedure: sp\_GenerateExam**

**Purpose**

This stored procedure **generates a new exam** for a specified course and instructor. It creates an exam record and populates it with a random selection of multiple-choice (MCQ) and true/false (T/F) questions from the Question\_Bank.

**Parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Data Type** | **Description** | **Default Value** |
| @CourseID | INT | **Required.** The unique identifier (Course\_ID) of the course for which the exam is being generated. | *None* |
| @InstructorID | INT | **Required.** The unique identifier (Instructor\_ID) of the instructor creating or assigned to the exam. | *None* |
| @NumMCQ | INT | The number of multiple-choice questions to include in the exam. | 7 |
| @NumTF | INT | The number of true/false questions to include in the exam. | 3 |
| @ExamDuration | INT | The duration of the exam in minutes. | 60 |
| @ExamType | NVARCHAR(50) | The type of exam (e.g., 'Normal', 'Corrective'). Must match allowed values in the Exam table constraint. | N'Normal' |

**Functionality**

1. **Validation:**
   * Checks the Question\_Bank to ensure there are enough available MCQ and T/F questions for the specified @CourseID based on the requested @NumMCQ and @NumTF.
   * If insufficient questions exist for either type, it raises an error message and stops execution.
2. **Exam Creation:**
   * Starts a database transaction to ensure all steps complete successfully or none are saved.
   * Determines the next available Exam\_ID by finding the maximum current ID and adding 1.
   * Inserts a new record into the Exam table with the provided parameters and the current date (GETDATE()) as the Exam\_Date.
3. **Question Selection:**
   * Randomly selects the specified number (@NumMCQ) of multiple-choice questions for the @CourseID from the Question\_Bank.
   * Randomly selects the specified number (@NumTF) of true/false questions for the @CourseID from the Question\_Bank.
   * Uses ORDER BY NEWID() for randomization.
   * Inserts the selected Question\_IDs along with the newly generated Exam\_ID into the Exam\_Questions table, linking the questions to the exam.
4. **Transaction Commit:** If all insertions are successful, the transaction is committed, saving the changes.
5. **Confirmation:**
   * Prints a success message indicating the newly generated Exam\_ID.
   * Executes a SELECT query to display details of the newly created exam and the questions included in it (Exam ID, Course Name, Instructor Name, Question ID, Question Type, Question Description).

**Error Handling**

* If not enough questions of either type (MCQ or T/F) are available in the Question\_Bank for the specified course, the procedure raises a descriptive error and terminates.
* If any database error occurs during the insertion process within the TRY block (e.g., constraint violation, connection issue), the CATCH block rolls back the entire transaction, ensuring no partial exam is created. The original error is then re-thrown.

**Usage Example**

**Generate a standard 10-question exam (7 MCQ, 3 T/F) for Course ID 1 by Instructor ID 1:**

SQL

EXEC sp\_GenerateExam

@CourseID = 1,

@InstructorID = 1;

**Generate a shorter, 5-question corrective exam (3 MCQ, 2 T/F) with a 30-minute duration:**

SQL

EXEC sp\_GenerateExam

@CourseID = 5,

@InstructorID = 2,

@NumMCQ = 3,

@NumTF = 2,

@ExamDuration = 30,

@ExamType = N'Corrective';

**Output**

* **On Success:**
  + A printed message: Successfully generated Exam ID: <NewExamID>
  + A result set (table) containing details of the new exam and its questions.
* **On Failure (e.g., not enough questions):**
  + An error message (raised via RAISERROR) explaining the problem.