**Online Examination System – Expanded README Explanation**

**Project Overview**

This project is designed to simulate the working of a real-world online exam platform. It's built on database principles and simulates functionalities such as:

* Registration & login
* Exam creation by instructors
* Question assignment
* Exam participation by students
* Auto-calculated results

It helps understand **entity relationships**, **data normalization**, **query design**, and **foreign key constraints**.

**Objective**

The objective of this mini project is to:

* Design a normalized relational database structure
* Maintain data consistency using constraints and relationships
* Write SQL queries for real-world use cases like reports, analysis, and tracking
* Apply DBMS concepts like **ER modeling**, **relational schema**, and **data integrity**

**Types of Users**

1. **Admin**
   * Adds subjects
   * Manages users
   * Oversees all operations
2. **Teacher**
   * Creates exams
   * Adds questions
   * Views student submissions
3. **Student**
   * Takes assigned exams
   * Views their own results

**Key Features**

| **Feature** | **Description** |
| --- | --- |
| User Management | Registration, login, role assignment |
| Subject Handling | Admin adds and edits subjects |
| Exam Setup | Teachers can create exams and assign questions |
| Question Bank | Multiple-choice questions stored and categorized |
| Answer Submission | Students select answers during exams |
| Auto-Result Generation | System evaluates answers and calculates scores |
| Reporting | Admins and teachers can view student performance |

**Database Design**

7 key tables have been used to store and manage all information:

| **Table** | **Purpose** |
| --- | --- |
| Users | Stores all user login and role info |
| Roles | Stores types of roles in system |
| Subjects | List of subjects like Math, Science |
| Exams | Exam schedule and details |
| Questions | Contains question text and options |
| Answers | Stores student responses |
| Results | Final scores and exam status |

Each table is related using **foreign keys** (e.g., ExamID, StudentID), ensuring proper linking.

**Table Creation Scripts**

You'll have a file like Table\_Creation.sql with all the CREATE TABLE statements. These define:

* Data types (e.g., VARCHAR, INT)
* Constraints (e.g., NOT NULL, UNIQUE)
* Primary and foreign keys

These ensure your database is **reliable**, **searchable**, and **relational**.

**Sample SQL Queries**

Use cases for the system include:

1. Get student names who participated in an exam
2. Fetch results with exam titles and pass/fail status
3. Calculate class average for performance review
4. Upcoming exams for a specific subject

You can include these in a Sample\_Queries.sql file for demonstration.

**Technologies Used**

* **MySQL** or **PostgreSQL** – to build and query the DB
* **SQL** – to handle data manipulation and reporting
* **ERD tools** – like **Draw.io**, **dbdiagram.io** or **Lucidchart** for visual design

**Conclusion**

This project helps you understand:

* How a full DBMS project is structured
* Designing real-life systems using entity modeling
* Implementing relational databases with queries
* Use of SQL for backend logic and user interactions

It’s an ideal **college-level project** and lays the foundation for more advanced DBMS work in applications like learning platforms, portals, or certification systems.