ABC Primary School Database System Documentation

1. Database Schema Design

Why This Design?

To effectively manage the school's teaching process, we structured the database with a clear separation of entities like students, instructors, courses, and semesters. The schema ensures:

- A 5-year academic system with multiple courses per year.
- Each instructor can teach multiple classes across semesters.
- Students are linked to courses within their academic year.
- Attendance and exam marks are tracked at the class level.

Key Entities:

- AcademicYear: Represents each school year (e.g., 2023-2024).
- Instructor: Stores instructor details, with an IsDeleted flag for soft deletion.
- Course: Holds course names and duration.
- Students: Connects students to their academic year.
- Semester: Tracks different semesters and which one is active.
- Class: Represents a group of students assigned to an instructor for a course.
- Exam: Holds exam details linked to a class.
- Attendance: Records student attendance per class per date.
- ExamMarks: Stores student grades.
- Enrollment: Links students to classes.
- CoursesAcademicYear: Defines which courses belong to which academic year.

2. Stored Procedures

Managing Instructors

AddOrUpdateInstructor

- **Why?** So we don't create duplicate instructors.
- **How?** If the instructor exists, we update their details; otherwise, we insert a new one.

DeleteInstructors

- **Why?** Instead of permanently deleting instructors, we mark them as deleted.
- **How?** Uses an IsDeleted flag to keep records but hide them from normal queries.

SearchInstructors

- Why? Quickly find instructors based on name, email, course, or class.
- How? Joins multiple tables to return how many courses and classes an instructor is handling.

Managing Attendance

RecordAttendance

- **Why?** We need to track whether students are present, absent, or late.
- **How?**
    - Prevents duplicate entries by checking if an attendance record already exists.
    - Updates the record if the student already has attendance for the given date.
    - Allows marking students as Late and storing the number of late minutes.

GetAttendanceRecords

- **Why?** We need to retrieve attendance details for a class on a specific date.
- How?
    - Fetches student names, attendance status, and late minutes.
    - Joins Students, Class, and Attendance tables for better readability.

---

3. User-Defined Functions (UDFs)

Why Use Functions?

Instead of repeating logic inside multiple queries, we created functions that:

- Get the currently active semester **(GetActiveSemester)**.
- Count the number of courses an instructor teaches **(GetInstructorCourseCount).**
- Count the number of classes an instructor is assigned **to (GetInstructorClassCount).**

This keeps the stored procedures clean, efficient, and reusable.

---

4. Testing

How We Tested the System?

We inserted test data for instructors, students, and classes, then executed stored procedures with different inputs to verify their correctness.

Example Tests

- Adding or Updating an Instructor
- Recording Attendance
- Retrieving Attendance Records
- Searching for Instructors
- Verifying Data

---

5. Summary

This system ensures that school data remains organized, efficient, and scalable. It prevents duplicate records, maintains data integrity, and provides fast and accurate retrieval of information.

Next Steps:

- Implement a front-end system to interact with the database.
- Add reports for attendance tracking and student performance analysis.

This design sets the foundation for a fully automated school management system.