1. I need a student table which has StudentID, First name, Last name, School ID, person number, phone number, guardian, address , grade they attend now..

A student can attend only one school, enforce it with a key. Index on studentid or name for faster search.

I have a constraint here, how do we maintian old records of old schools a student has attended. I mean if he changed schools, how do we get those details.

Does a students table hold multiple values with different schoolids and only one active school flag?

1. I need a schools table where it has a unique Schoolid , organization number, name, address, type of school etc.. Index schools on SchoolID and create a relation with Workingdays table with workingdayID
2. WorkingDays table to hold list of WorkingdayID which is unique and SchoolID, dayDate, Isworkingday flag. This will help to have individual working days for each school. Ensure that the schoolid matches with Schoolid from schools table.
3. I need a Grades table which should be referenced by both a School and also students.

How do we store the previous grades and previous attendance records.

Grades shold relate to a grade a student is studying in, so ensure that a grade from this table is selected in Students table.

1. I need a attendance table where they have all the current year working days .
2. Archive of all the attendance records for previous years can be maintained in an archive with academic years.

Attendance table must hold attendanceid which is unique number and StudentID, his schoolid , WorkingdayID which gives us the day and then a flag to say if he was in the school.

We need workingDays table to create any future plans for school events or planning of events and classes at the school.

There must be only one entry in combination to a workingday to a studentid, which means a student can have one attendance record with a workingdayid and this workingdayid must belong to his school.

This completes my database creation. How do I generate Visual studio EF classes and create the DB from C#. So that all the DB is created if it does not exist and all the keys are enforced.

Partitioning: Consider partitioning large tables like Attendance based on school or time ranges to improve manageability and query performance.

Normalization: Use normalization to reduce redundancy and improve data integrity. This helps in efficiently managing and querying large datasets.