Table Schema For Quiz Project CS108

* [Sample] Repository Name
  + TableName1
    - Field1: Type + description
    - Field2: Type + description
  + TableName2
    - Field1: Type + description
    - Field3: Type + description
* Quiz Repository (Saving Quiz Information)
  + QuizSummary
    - QuizId: Primary Key, Auto-Increment
    - QuizName: varchar(100), Default: “Quiz ” + QuizzId
    - QuizDescription: varchar(100), Default: “Default Quiz”
    - CreatorId: Int, UserId of user that created quiz
    - CreateDate: DateTime, time quiz was created
  + QuizAttemptHistory
    - AttemptId: Primary Key, Auto-Increment
    - QuizId: Int
    - UserId: Int
    - AttemptScore: Int, score on attempt
    - AttemptPossible: Int, maximum possible score on quiz
    - ElapsedTime: DateTime, time between “start quiz” and “submit quiz”
    - AttemptDate: DateTime, time of “submit quiz”
  + QuizStats
    - QuizId: Int
    - QuizAttempt: Int, incremented per user submit action
    - SumActualScores: Int, incremented per user submit by AttemptScore
    - SumPossibleScore: Int, incremented per user submit by AttemptPossible
      * Note: Saving as sums because I believe it will be easier to track (just add a number versus recalculate averages in SQL)
    - UserAttempts: Int, incremented per UNIQUE user submit action
  + QuizQuestions
    - QuestionId: Primary Key, Auto-Increment
    - QuizzId: Int
    - QuestionType: VarChar(20), short code (such as “MultChoice”) describing type of question
    - QuestionText: VarChar(MAX), question text
  + QuizAnswers
    - AnswerId: Primary Key, Auto-Increment
    - QuestionId: Foreign key, links answers to their question (don’t need quiz id since QuestionId is unique)
    - AnswerType: VarChar(20), short code (such as “ShortAnswer”) describing type of answer
      * Note: I’m not sure this is actually needed, can just make correspondence between Question and Answer 1 to 1 and not need this…
    - AnswerText: VarChar(Max), possible answer to question (note: not necessarily correct)
    - AnswerCorrectFlag: Boolean, true/false about whether this answer is correct
* User Repository (Saving User Information)
  + UserDetail
    - UserId: Primary Key, Auto-Increment
    - UserName: VarChar(60), user name (what should maximum length be?)
    - UserPassword: VarChar(60), user password in hash
      * Note: Figure out the maximum return type length of Encryption Algorithm we are using
    - UserEmail: Not sure if we need this, but good to use as an extension (maybe add an admin email blast feature to send messages to all subscribed users?)
    - UserType: VarChar(10), “Admin” or “User”
    - UserCreateDate: DateTime, date of user account creation
  + UserFriends: Note that for this table, need to add a new friend transaction as two entries (as both the User and Friend)
    - UserId: Int
    - FriendId: Int
    - FriendAddDate: DateTime, date that the friend was added
  + UserSocial
    - MessageId: Priamy Key, Auto-Increment
    - UserId: Int
    - FriendId: Int, corresponds to UserId of this user’s friend
    - MessageType: VarChar(20), short code (such as “FriendRequest”) describing the type of the Message (Note, Friend Request, or Challenge)
  + UserNote
    - MessageId: Int
    - MessageText: VarChar(Max) – I think this would limit message to 256 characters, need to look into a different type if we want larger messages. Possible option is to break the message out into smaller pieces?)
  + UserChallenge
    - MessageId: Int
    - ChallengerNote: VarChar(Max), similar to above but I think it is fine to limit the size of this note
    - ChallengerScore: Int, will be populated by looking through the QuizzAttemptHistory table
    - ChallengerQuizId: QuizId of the challenge, use to create link in challenge UI
  + UserFriendRequest
    - MessageId:Int
    - RequestStatus: VarChar(20) code for status of request. Can be “Sent”, “Read”, “Completed”
    - FriendAccepted: Boolean, true/false if the request was accepted
      * Note: Remember to update both UserFriendRequest and UserSocial if added/deleted friend
  + UserAchievements
    - AchievementId: Primary Key, Auto-Increment
    - UserId: Int
    - AchievementName: VarChar(40)
    - AchievementDescription: VarChar(100), could be increased if needed
    - AchievementDate: DateTime, date that achievement was acquired
    - AchivementToolTip: VarChar(100), if we want, can store the location of the tooltip we use as the “badge” earned for an achievement
  + UserActivity: Note, this table needs to be updated every time a user does and “action”
    - ActivityId: PrimaryKey, Auto-Increment
    - UserId: Int
    - ActivityType: VarChar(20), code for the type of activity. Examples are “Achievement”, “QuizTaken”, or (extra credit?) “FriendOfFriend” for a User’s friend becoming friends with another of his/her friend
    - ActivityLinkId: Int, this will be an ID that will correspond as follows based on the ActivtyType:
      * “Achievement”: AchievementId
      * “QuizTaken”: AttemptId
    - ActivityDate:DateTime, time of the activity
* Site Repository (Info saved to be used throughout the website)
  + Announcement
    - AnnouncementId: Primary Key, Auto-Increment
    - AnnouncementCreatorId: UserId of creator of the assignment
    - AnnouncementCreateDate: DateTime, date of creating of the announcement
    - AnnouncementText: VarChar(Max), same issue as with notes before, how large should we allow this to be?