C#/ SQL+ RDBMS

1. C#
   1. **Delegates** = a reference type that refers to a particular method with a specific return type and parameters
      1. Properties:
         1. Can be static or instantiable
         2. Can be assigned any method that matches the delegate type and is accessible
         3. Fully OO and encapsulate an object instance and method
            1. When instantiated, they’re invokable as if it were the method
         4. “Chainable” – allowing for multiple methods for a single event
      2. Str:
         1. Pass precise methods as arguments for other methods
            1. **Callback functions**
         2. Flexible via **covariance** and **contravariance**
      3. Lim:
      4. **Action delegates** vs **Function delegates**
      5. **Async/await**
      6. Threads
   2. XML
   3. Event Handlers
   4. Lambda Expressions = an anonymous function that takes input parameters from the L of the lambda operator and executes an expression/operation on the R of the operator. (Usually in place of a delegate)
      1. Properties
         1. Lambda operator: =>
            1. Intuitively: “goes to” or “becomes”
            2. Parameter types must be explicit (int x/string y) or implicit
      2. Statement Lambda
         1. Cannot be used for **expression trees**
      3. Expression Lambda
      4. Str
      5. Lim
   5. **Predicate**
   6. **Anonymous types**
2. SQL + Database Management
   1. **Normalization**
   2. **DML**
   3. RDBMS
   4. ACID
   5. Repo Pattern
   6. Dependency Injection
   7. ORM
   8. EF
   9. Fluent API
   10. Scaffolding
   11. Input Validation
   12. View = SQL Table that is virtual and doesn’t allow you to actually change your data
       1. Computed Column = a column that is the result of some computation on another table. Not physically stored in a table unless its PERSISTED.
          1. Requires AS to make a column as a CC
       2. Computed Table
       3. Properties
          1. Compu
          2. Schema binding = prevents changes from happening to underlying tables
       4. Str
       5. Lim
   13. Procedures
   14. SQL Functions = Function stores in SQL that outputs a scalar or table as a result
       1. Properties
          1. Scalar Function: takes in 1 or more parameters, operates on a single value, and outputs a scalar{
             1. Accepts user parameters
             2. Performs an action
             3. Returns the result of the operation as a scalar or table}
             4. Can be used wherever a query is valid
          2. Aggregate functions
          3. Table-Valued Functions (https://bit.ly/3cz5I8j)
             1. Inline
             2. Multi-statement
       2. Str
          1. Instead of inputting a commonly-used complex function into every query, functions allow use to streamline code & be more efficient by invoking just the function
          2. Organized into DB under the programmability folder
          3. Can use IF and WHILE logic blocks
          4. Can call other functions
          5. Can be easily modified using ALTER
       3. Lim
          1. Have readonly access – cannot input or alter the database
          2. Cannot output multiple result sets
          3. Does not support error handling via “try..catch”, @error, or @raiseerror
          4. Cannot use SET statements
   15. Scalar = a single value of any datatype that resides in a table
   16. Table-valued parameter = a function parameter that is a table
   17. Creating a function:
       1. CREATE FUNCTION + name
       2. Parameter selection: Choose parameters/columns/factors to be used in operation
       3. RETURN datatype
       4. RETURN statement: operational content of the function
          1. Make sure the RETURN statement gives only **one** output by using parentheses correctly as needed
       5. End
   18. 