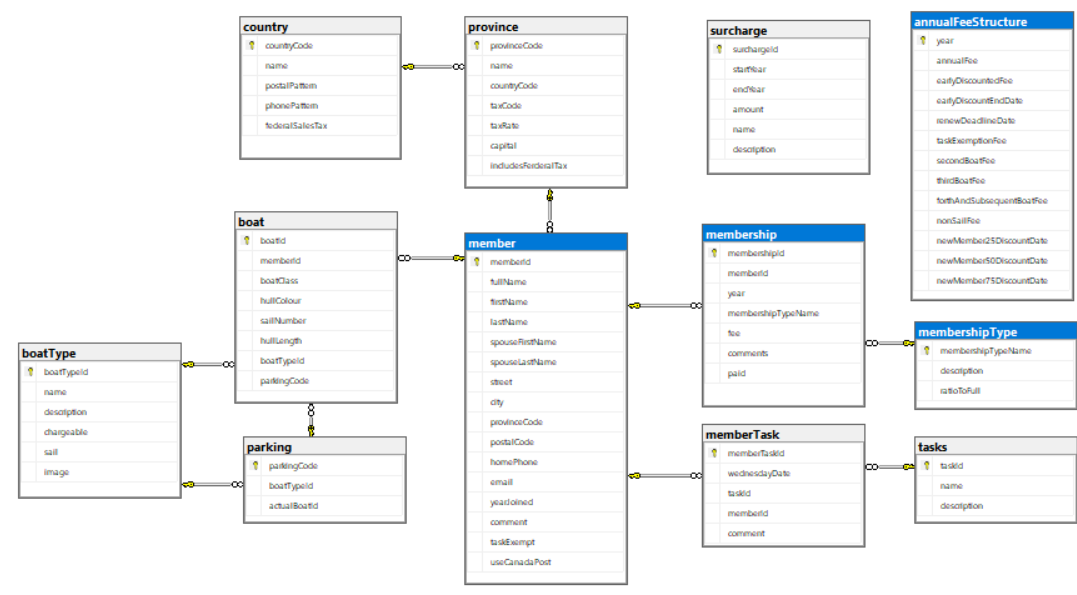
## Ms Web Tech Assignment – Persistence, sorting & filtering

In this assignment, you will pass & persist (remember) a key. When the user selects a club member, you will provide a link to list all memberships that member has had and to add a new membership for the current year. As always, where you see “XX”, substitute your name



### Project Setup

1. You can continue using your ***XXSail*** project from previous assignment.
2. On ***~/Views/Shared/\_layout.cshtml***, display the TempData variable “message” before the guest View, bold and red.
3. Put a nautical-themed logo (image) in the header with a height of 100 pixels, and a better site heading than just “XXSail”.
   1. The logo will enlarge the banner on each page. The body of the guest page passes under the banner: modify the site CSS so no content (like page headings or @TempData[“message”]) is overlaid.
4. Add support for Session variables to your site’s Startup.cs

### XXAnnualFeeStructureController

1. Generate a controller for the *annualFeeStructure* table, called ***XXAnnualFeeStructureController***, with full CRUD support and views.
2. On the Index view, just show the *year* and *annualFee* fields, and order the record by year … most recent first.
3. On the setup Create, preload the view with data from the most recent record on file & overlay the *year* with the current year. Don’t lose the user’s input if the post-back create encounters an error.
4. In the Edit actions, return to the Index view, with an error message in TempData, if the user tries to edit a prior year’s record.
5. Change the browser title and page headings on all views to reflect the table being modified and the view’s action: “Annual fees by year”, “Create a new Annual Fee record”, etc.

### XXMemberController

1. Generate a controller for the *member* table, called ***XXMemberController***, with full CRUD support and views. Make sure that create, edit, delete and details pages all work.
2. In the Index View:
   1. Change the page’s heading and the browser title to “Member List”.
   2. Add the *memberId* to the index view, then restrict the other fields to *fullName*, *homePhone*, and *eMail*. If someone wants to see more fields, they can click the Details link.
      1. Order the listing by *fullName.*
   3. Change the *fullName* of each member into a hyperlink that passes its *memberId* to the *XXMembership* controller’s Index action. You might also want to pass the member’s full name.

### XXMembershipController

1. Generate a controller for the *membership* table, called ***XXMembershipController***, with full CRUD support and views. Make sure that create, edit, delete and details pages all work.
2. In the *Index* action:
   1. If the *memberId* is in the URL or a QueryString variable, save it to a cookie or session variable.
   2. If there is no *memberId* passed in the URL or QueryString, see if there’s a cookie or session variable with it.
      1. If there’s no *memberId* in the cookie or session variables either, return to the ***XXMember*** controller with a message asking them to select a member to see its memberships.
   3. Regardless of where you got the *memberId* from, use it to filter the listing to only show memberships on file for that member, order by year (most recent first).
   4. If fullNname was passed as well, persist it the same way … otherwise, take the memberId, fetch the record from the member table, then extract & persist its name.
   5. Do not show either the memberId or membershipId … instead, show the member’s fullName in the browser’s title and the view’s heading … “Memberships for Chesterfield, Stanley & Mabel”
3. In the Create and Edit views:
   1. Do not show an input field for *memberId* … the user has requested memberships for a specific member, so stay in that mindset.
      1. When posted back, the *memberId* field still has to be inserted into the new/modified record, so hide it on the views. If a hidden field is not given a value, it posts back as an empty string, which kills numeric receiving parameters … even nullable ones.
   2. In the *membershipType* drop-down, display and return the *membershipTypeName* (that’s its primary key), in ascending order.
   3. In the Edit actions, do not permit changes to membership records for prior years.
   4. In the Create view & action:
      1. Default the *year* to the current year.
      2. On post-back, retrieve the *annualFeeStructure* record for the current year.
      3. Calculate the *fee* for the selected *year* and *memberShipType* before saving the record to the database.
4. In all views for this controller:
   1. Modify each browser title and page heading to indicate its purpose and the member it’s for, like “Add a Membership for Chesterfield, Stanley & Mabel”, or “Details of Membership for Chesterfield, Stanley & Mabel” and so on … watchyourpunctuation.
   2. Do not show the *memberId* or *membershipId* on any view.

### Hand In

1. Zip and upload your project folder to the drop box (course tools 🡪 assignments) for this assignment.