# ROM Estimator

# Database and C# Technical Documentation

## List of Resources, where and how to obtain them

### General Resource Information

* Deployed Version: <https://ecsocoms-proservices-rom.nprod.vaec.va.gov/>
* Azure PaaS SQL resource: [romestimator-db - Microsoft Azure Government](https://portal.azure.us/#@vaazuregov.onmicrosoft.com/resource/subscriptions/3b091b7b-37d8-4f17-aa4d-6fddfc5dd7c5/resourceGroups/VAEC-NPROD-INT-EAST-RG/providers/Microsoft.Sql/servers/romestimator-db/overview)
  + Azure PaaS SQL database: [romestimator](https://portal.azure.us/#@vaazuregov.onmicrosoft.com/resource/subscriptions/3b091b7b-37d8-4f17-aa4d-6fddfc5dd7c5/resourceGroups/VAEC-NPROD-INT-EAST-RG/providers/Microsoft.Sql/servers/romestimator-db/databases/romestimator/overview)
  + GFE prevents direct connection to the database due to local firewall rules
  + SSMS connection is accessible via RDP to the Azure jump box, running SSMS on the jump box, and connecting via the fully qualified SQL server name romestimator-db.database.usgovcloudapi.net
  + Database administrator name: romest-admin
  + Database administrator password:
  + Database access “panic button”: set your 0-account as the Database owner via the Azure Portal
* Azure App Service Link: [ecsocoms-proservices-rom - Microsoft Azure Government](https://portal.azure.us/#@vaazuregov.onmicrosoft.com/resource/subscriptions/3b091b7b-37d8-4f17-aa4d-6fddfc5dd7c5/resourceGroups/vaec-nprod-int-east-rg/providers/Microsoft.Web/sites/ecsocoms-proservices-rom/appServices)
* Git repository for the ASP.NET C# project code: <https://github.ec.va.gov/Owen-Emlen/romestimator>
  + **Note that the current published/latest version is in branch “version3”**
* App Service Web-Deploy username: $ecsocoms-proservices-rom
* App Service Web-Deploy password:
  + You will be prompted for this when publishing the C# code from Visual Studio.
  + Note: the original web publish profile, including this information, is included in [Appendix B](#_Appendix_B:_Web).

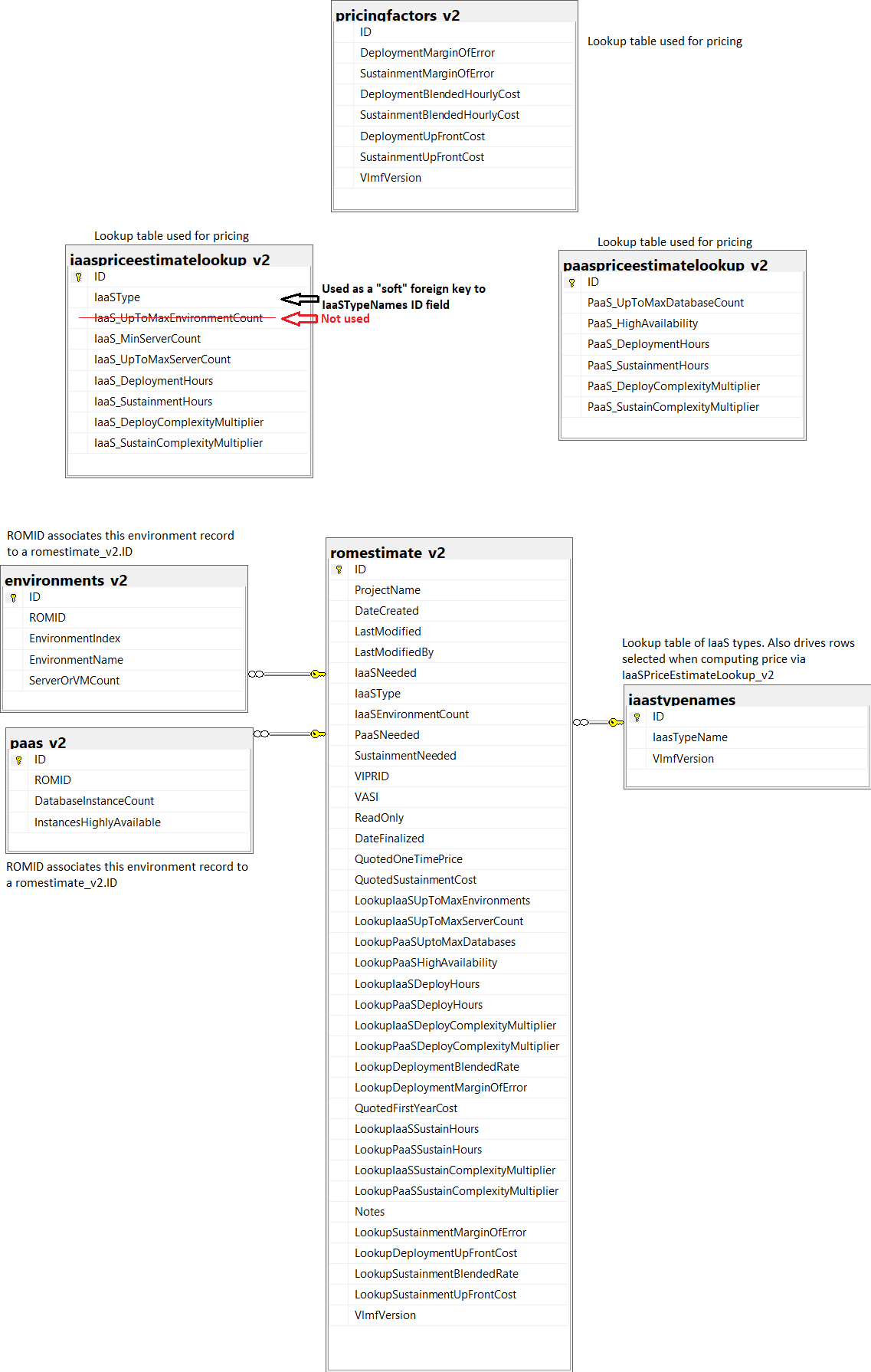
### Copying the PaaS database to your GFE

Should you need to make changes to the code, because the Entity Data Mapper requires connectivity to the ROM database, *I recommend making all modifications against a local copy of the ROM Estimator SQL database*. The connection string to switch between local and Azure PaaS is located within the web.config file. There are comments in there that show you how to switch connection strings. *Always make sure to switch back to the Azure PaaS database connection string before you publish the code to Azure* (which you can publish directly from your GFE laptop). To pull down a local copy of the database to your GFE:

1. RDP to the Azure jump box
2. Run SQL Server Management Studio (SSMS) and connect to romestimator-db.database.usgovcloudapi.net
3. Select the romestimator database
4. Right-click and under Tools, use the “Generate Database Scripts” option
   * Make sure you select the Advanced option and then select the “**script schema and data**” option
5. The generated script can be copied in plain text to your GFE with SQL server and SSMS installed
6. Use SSMS locally to run the script (New Query), and the “romestimator” database (along with all data) will be created as a database within your local SQL instance.

### V2 (Current) Database Model

Note that all queries and modification of the data, schema, or stored procedures (SPROCS) in the Azure PaaS database romestimator-db.database.usgovcloudapi.net must be performed from SSMS on the Azure jump box. A diagram of all relevant tables can be found on the following page.



### Entity Data Mapping

The ROM Estimator program (written in C# with an ASP.NET front-end) uses the standard Entity Data Model Mapping tool provided by Microsoft. **Communication with the database is done** **only through Stored Procedure Calls (SPROCS)**. In particular, the following stored procedures are used to read from and write data to the database:

* **CreateRom\_v2**
  + Takes a ROM name, username (or identifier), and VLMF version (currently 2)
  + Returns a new unique ROM ID (GUID)
* **ReadRom\_v2**
  + Takes a ROM ID and returns the corresponding romestimate\_v2 records
  + This returns multiple sparse records because it joins on the environments\_v2 table
* **UpdateRom\_v2**
  + Updates any or all of the user-modifiable fields for a given a ROM ID
* **ProvideEstimate\_v2**
  + Takes a ROM ID and provides a price estimate (including OneTimeCost, SustainmentPerYear, and QuotedFirstYearCost, used to populate the pricing portion of the UI)
* **FinalizeEstimate\_v2**
  + Takes a ROM ID and sets its **ReadOnly** field to true, causing the UI to display a non-modifiable ROM. **Also recalculates and locks in the price**, writing non-user-modifiable data to the record in romestimate\_v2
* **ReadIaaSTypeNames**
  + Takes a VLMF version number and returns the IaaS IDs and type names associated with the VLMF version (currently 2)

As a general note, all tables and classes with \_v2 appended to them supersede any table/class of the same name without the \_v2 suffix. This was done to retain the data from v1 ROMs, which still exist in the database, but may no longer accessible via the ROM Estimator UI (they may display pricing information, but I’ve never tested backwards compatibility with the original implementation where prices were simply looked up via spreadsheet values). The corresponding calls to the old tables (which can be retrieved directly via the database or by using the Common.cs class in code) now exist in the Cmmon\_v2.cs class, and Common.cs is now a deprecated class.

The entity data mapper (which can be accessed within Visual Studio when you double-click on RomModel.edmx and then select “Model Browser” tab, and then expand the Entity Types and Complex Types) shows the direct mapping of SPROC return value names to C# Properties and class. For information about Microsoft’s Entity Framework Object-Relational Mapper (ORM), see [Entity Framework documentation | Microsoft Docs](https://docs.microsoft.com/en-us/ef/).

### Example Use – Instantiating the Entity Data Model, calling a SPROC, and using the Result set

I use a loosely-termed ViewModel class “public class ReadRomViewModel\_v2 : ReadRom\_v2\_Result“, where the colon indicates that the class inherits from the Result set returned by the ReadRom\_v2 SPROC call and extends upon it, making relevant information more easily accessible to the User Interface.

The following code is located in ROMEstimator.aspx.cs in the Page\_Load method and demonstrates how the Entity Framework data layer is instantiated, the ReadRom\_v2 SPROC is called, and then converted into a ReadRomViewModel\_v2 class, suitable for consumption (and manipulation) by the UI:

// Read the ROM from the database and set up a ViewModel for UI interaction

**RomestimatorDataLayer** romestimatorDataLayer = new **RomestimatorDataLayer**(); *🡨 Sets up DB connection*

ObjectResult<ReadRom\_v2\_Result> objectResult2 = romestimatorDataLayer.**ReadRom\_v2**(romId);*🡨 SPROC call*

var list = objectResult2.ToList();

if (list.Count > 0)

{

// Create the main ROM View Model (v2) to display to the user

**RomViewModel\_v2** = **Common\_v2**.**FlattenRomFromDatabase\_v2**(list); *🡨 Creates a UI friendly model*  
}  
... else treat ROM as v1 or not found

This code, because it is located in the Page\_Load method, is called each time the page loads, and makes the populated variable **RomViewModel\_v2** available for all other methods in the ROMEstimator class, which is directly wired to the UI (and is associated with the HTML and markup contained in ROMEstimator.aspx).

### User Interface Structure

The user interface has one master page, titled **Site.master**, which contains the common UI header (ROM Estimator and the VAEC Logo), and two “pages”, both of which are displayed *within the context of Site.master*:

1. **Default.aspx** 🡨 The default page showing the user summary information and a button to “Create a New ROM”
2. **RomEstimator.aspx** 🡨 The ROM Estimator form page, which must always be passed the GUID of a valid ROM ID via query string – for example RomEstimator.aspx?<rom\_guid>

### Displaying What’s included in the ROM Estimate

The WhatsIncluded.ascx user control is embedded at the bottom of RomEstimator.aspx, as follows:

<section class="section-indent auto-height">

<uc1:WhatsIncluded runat="server" ID="WhatsIncluded" />

</section>

This control is responsible for showing What’s included in the ROM estimate based on what the user has selected. It either shows or hides each of the following user controls. The file names should be self-explanatory:

Always displayed:

* NotIncludedInRom.ascx 🡨 Displays text about what is NOT included in a ROM estimate

Displayed when any IaaS or PaaS deployment type is selected:

* StandardDeploymentServices.ascx 🡨 Displayed when any type of deployment (PaaS or IaaS) is requested
* IaaSDeployment.ascx 🡨 Displayed when an IaaS Deployment type is requested
* GreenfieldMigrationDeployment.ascx 🡨 Displayed when a Greenfield Deployment type is requested
* VMLiftAndShiftDeployment.ascx 🡨 Displayed when a Lift and Shift Deployment type is requested
* PaaSDeployment.ascx 🡨 Displayed when PaaS Deployment is requested

Displayed when Needs Support Services are selected:

* StandardSupportServices.ascx 🡨 Displayed when Needs Sustainment is selected
* IaaSSustainmentServices.ascx 🡨 Displayed when any IaaS Deployment type and Needs Sustainment is selected
* PaaSSustainment.ascx 🡨 Displayed when PaaS Deployment and Needs Sustainment is selected

### ROM Estimate Pricing Calculation

ROM Pricing has evolved from (1) a simple Excel spreadsheet lookup to (2) an equation involving complexity and discounts, to (3) a translation to an hour estimate, which is then used to estimate cost.

Note: An Excel spreadsheet (proprietary to Cognosante/COMS), which should be included as an attachment with this document, dictates the number of estimated hours spent to build each server (per IaaS type) for a certain number of servers. More detail on how hour estimates are calculated will be provided later in this document.

**All constants (prices and hours) are stored in the database as lookup table values, and all logic for calculating price is contained in stored procedures** (in particular the two stored procedures ProvideEstimate\_v2 and FinalizeEstimate\_v2, which contain identical price calculation logic). **Therefore, if pricing adjustments are needed, only the database tables and/or stored procedures need to be changed. Code only needs to be deployed if there are cosmetic changes needed.**

The equation for IaaS Estimated One-Time Deployment Cost is:

ROM IaaS Estimated Deployment Cost = (Total Estimated IaaS Deployment Hours \* IaaS Deployment Bucket Complexity Multiplier (1) \* Deployment Hourly Blended Rate $$ \* Deployment Margin of Error) + Deployment Up-Front Cost $$

The equation for IaaS Estimated Yearly Sustainment Cost is:

ROM IaaS Estimated Yearly Sustainment Cost = (Total Estimated IaaS Sustainment Hours per month \* 12 months \* IaaS Sustainment Bucket Complexity Multiplier (1) \* Sustainment Hourly Blended Rate $$ \* Sustainment Margin of Error) + Sustainment Up-Front Cost $$

The equation for PaaS Estimated One-Time Deployment Cost is:

ROM PaaS Estimated Deployment Cost = (Total Estimated PaaS Deployment Hours \* PaaS Deployment Bucket Complexity Multiplier \* Deployment Hourly Blended Rate $$ \* Deployment Margin of Error) + Deployment Up-Front Cost $$

The equation for PaaS Estimated Yearly Sustainment Cost is:

ROM PaaS Estimated Yearly Sustainment Cost = (Total Estimated PaaS Sustainment Hours per month \* 12 months \* PaaS Sustainment Bucket Complexity Multiplier \* Sustainment Hourly Blended Rate $$ \* Sustainment Margin of Error) + Sustainment Up-Front Cost $$

The final ROM estimate for **One-Time Deployment Cost** is calculated by adding the IaaS deployment estimate to the PaaS deployment estimate.

The final ROM estimate for **Yearly Sustainment Cost** is calculated by adding the IaaS sustainment per year estimate to the PaaS sustainment per year estimate.

The **Estimated First Year Cost** is simply the sum of the estimated one-time deployment cost plus one year of the estimated sustainment cost. This number is used for the Project Team’s budgeting purposes and for obtaining funding.

These three costs are “locked in” when the ROM is finalized, along with dates and the pricing numbers used to calculate the three estimated costs. This can be used as an audit trail between different versions of the application and can also be used to devise how the estimate was calculated. *Storing these values also serves to protect backwards compatibility;* ***once a ROM Estimate has been finalized, only these three stored cost estimates will be displayed in the UI, regardless of pricing algorithm changes that may have occurred after finalization.***

### VLMF 2.0 Hour Estimate Calculation

With VLMF 2.0+, the model for calculating estimated IaaS Deployment and Sustainment hours was made more complex. Please reference the attached Excel document for detailed information.  **Note that this is proprietary information, owned by COMS/Cognosante, and should not be shared with the VA.**

Basically, two “sizing buckets” need to be considered when calculating estimated hours. The best way to explain is to provide an example; let’s say the customer requests 25 IaaS servers in total for a simple IaaS Server Deployment project. Since the hour estimate matrix is as follows, the total calculated hours would be for 20 servers at 7 hours each, and the remaining 5 servers at 4 hours each (20\*7)+(5\*4) = 160 hours. The same calculation is done if the customer requests sustainment; the total calculated hours per month would be for 20 servers at 2 hours per month, and the remaining 5 servers also at 2 hours per month.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Deployment Margin of Error** | **Sustainment Margin of Error** | **Deployment Blended Hourly Cost** | **Sustainment Blended Hourly Cost** | **Deployment One Time Cost** | **Sustainment One Time Cost** |  |  |
| 1.5 | 1.5 | $180 | $150 | $3,000 | $3,000 |  |  |
|  |  |  |  |  |  |  |  |
| **IaaSTypeName** | **IaaS - Up To Max Server Count** | **IaaS - Deployment Hours** | **IaaS - Sustainment Hours/Month** | **Enter IaaS Servers Requested into any of the IaaS Deployment Type rows** | **ROM IaaS One-Time Cost** | **ROM Sustainment Cost (Yearly)** | IaaS hours calculated |
| IaaS Server Deployment | 1 | 8 | 2.0 | 25 | **$46,200** | **$138,000** | 160 |
| IaaS Server Deployment | 3 | 8 | 2.0 |  |  |  |  |
| IaaS Server Deployment | 10 | 7 | 2.0 |  |  |  |  |
| IaaS Server Deployment | 20 | 7 | 2.0 |  |  |  |  |
| IaaS Server Deployment | 100 | 4 | 2.0 |  |  |  |  |
| IaaS Server Deployment | 200 | 4 | 2.0 |  |  |  |  |

Unfortunately, though the logic seems simple, it makes for some nasty SQL code, having to add a condition for the special case where the customer asks for more servers than specified by the max bucket size, and the special case where a customer only asks for 1 VM.

-- SPECIAL CASE - only asking for 1 unit.

IF @IaaSTotalServerCount = 1

BEGIN

SELECT TOP 1 @IDTopRow = ID

,@DeployHoursPerUnit = IaaS\_DeploymentHours

,@LookupIaaSUpToMaxServerCount = 1

,@SustainHoursPerUnit = IaaS\_SustainmentHours

FROM [dbo].[**iaaspriceestimatelookup\_v2**]

WHERE IaaSType = @IaaSType AND @IaaSTotalServerCount = IaaS\_UpToMaxServerCount

ORDER BY IaaS\_UpToMaxServerCount  
... IaaS price calculation for only 1 unit ...  
END  
... otherwise more than 1 unit requested ...

-- Grab the max number of IaaS VMs in our model for yet another **corner case where IaaS VMs requested**

**exceeds the max in our model (currently 200)**

SELECT TOP 1 @MaxInModel = IaaS\_UpToMaxServerCount

from [dbo].[**iaaspriceestimatelookup\_v2**]

WHERE @IaaSType = IaaSType

ORDER BY [dbo].[iaaspriceestimatelookup\_v2].IaaS\_UpToMaxServerCount DESC

-- Determine the "Bucket" we're in (VMs requested >= Minimum VMs associated with the bucket), order

descending so we pick the highest matching bucket

SELECT TOP 1 @IDTopRow = ID

,@DeployHoursPerUnit = IaaS\_DeploymentHours

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount

,@SustainHoursPerUnit = IaaS\_SustainmentHours

,@MinUnitsCheaperBucket = IaaS\_MinServerCount

,@MaxUnitsCheaperBucket = IaaS\_UpToMaxServerCount

FROM [dbo].[**iaaspriceestimatelookup\_v2**]

WHERE IaaSType = @IaaSType AND @IaaSTotalServerCount > IaaS\_MinServerCount

ORDER BY IaaS\_MinServerCount DESC

**-- Handle the corner case where the customer requests >= (currently) 200 IaaS servers**

IF @IaaSTotalServerCount >= @MaxInModel

BEGIN

SET @DeployHoursPerUnitMoreExpensiveBucket = @DeployHoursPerUnit

SET @SustainHoursPerUnitMoreExpensiveBucket = @SustainHoursPerUnit

SET @UnitsAtLowerPrice = @IaaSTotalServerCount

SET @UnitsAtHigherPrice = 0

SET @LookupIaaSUpToMaxServerCountMoreExpensiveBucket = @MaxInModel

END

ELSE

**-- Handle the case where the customer requests a number of IaaS servers that fits into a "bucket"**

BEGIN

-- retrieve the "previous" bucket, since we will price all machines up to the bucket's max

VM count @IaaSTotalServerCount using that prior bucket

SELECT TOP 1 @DeployHoursPerUnitMoreExpensiveBucket = IaaS\_DeploymentHours

,@SustainHoursPerUnitMoreExpensiveBucket = IaaS\_SustainmentHours

,@LookupIaaSUpToMaxServerCountMoreExpensiveBucket = IaaS\_UptoMaxServerCount

,@LookupIaaSMinCountMoreExpensiveBucket = IaaS\_MinServerCount

FROM [dbo].[**iaaspriceestimatelookup\_v2**]

WHERE IaaSType = @IaaSType AND (ID = @IDTopRow - 1)   
**-- this is safe only because we already took care of the special case where 1 VM is requested**

END

-- "up to n units" in the more expensive bucket

SET @UnitsAtHigherPrice = @LookupIaaSUpToMaxServerCountMoreExpensiveBucket

-- remaining VMs above and beyond the maximum number in the higher priced bucket

SET @UnitsAtLowerPrice = @IaaSTotalServerCount - @LookupIaaSUpToMaxServerCountMoreExpensiveBucket

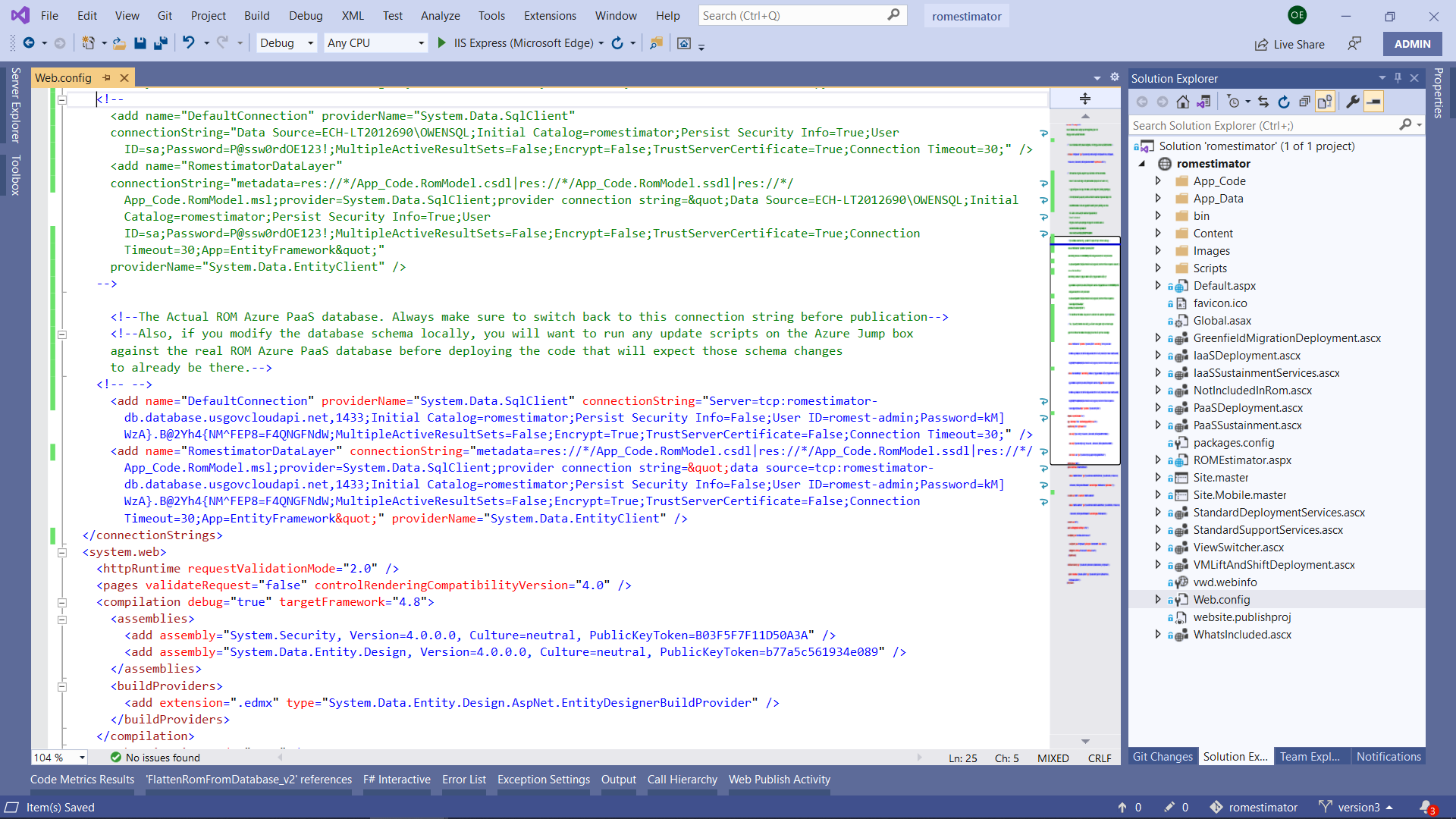
## Changing, Testing, and Publishing – Code and Database

Before modifying any code or making changes to database table contents or its schema, you will need to run the project locally against a local SQL database. To do this:

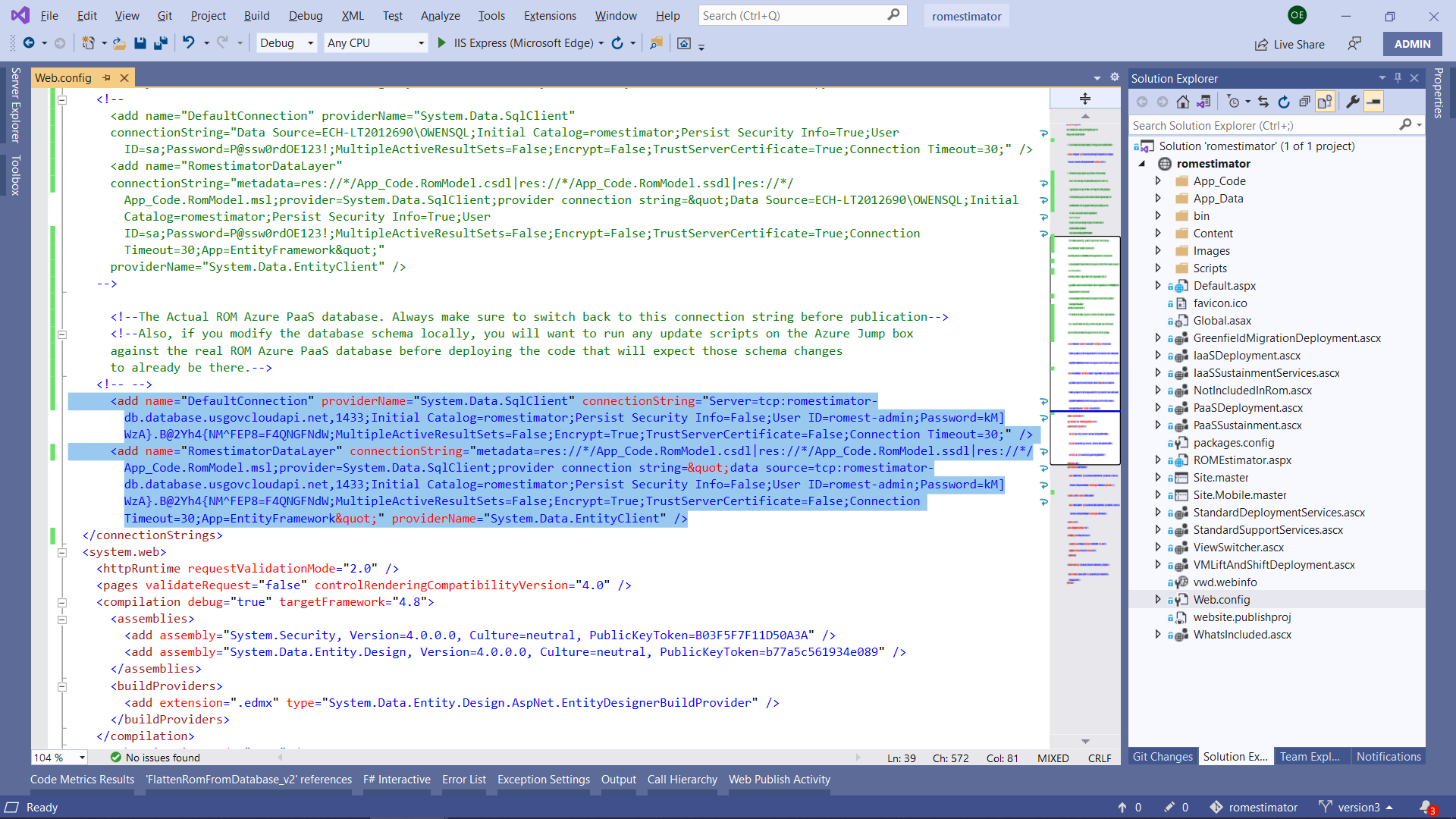
1. Download and install Visual Studio 2019, Community Edition (free) on your GFE.
   1. Download is located at [Download Visual Studio 2019 for Windows & Mac (microsoft.com)](https://visualstudio.microsoft.com/downloads/)
   2. Visual Studio Code may also work, but I have neither tried it, nor am I confident that the same level of integration exists between it and Microsoft’s database abstraction layer, also known as Microsoft Entity Framework (MEF).
2. Download SQL Server 2019 (free developer edition) or higher and install it on your GFE.
   1. The free developer version can be downloaded from [SQL Server Downloads | Microsoft](https://www.microsoft.com/en-us/sql-server/sql-server-downloads)
3. Download SQL Server Management Studio (SSMS) and install it on your GFE.
   1. [Download SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS) | Microsoft Docs](https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15)
4. Open SSMS and run the SQL script to create the romestimator database, along with all lookup table/pricing values, located in [Appendix C](#_Appendix_C:_SQL).
5. Clone the GitHub Enterprise repository, <https://github.ec.va.gov/Owen-Emlen/romestimator/tree/version3>, to a local directory.

### Testing Code changes locally and Publishing the Web App

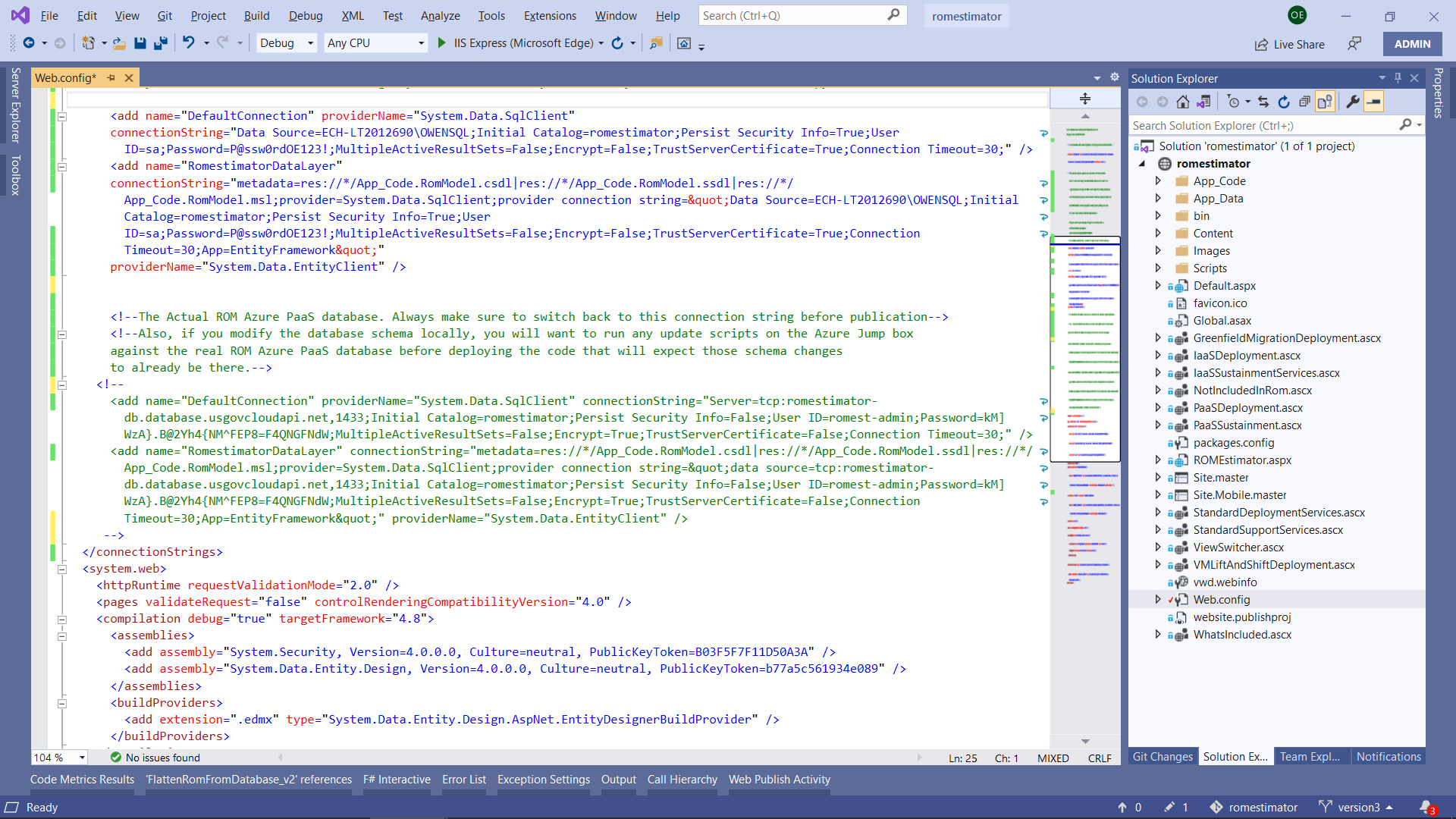
After you have performed the above 5 steps, open Visual Studio **as an administrator if you want to debug locally** **in IIS Express** and open the solution file romestimator.sln. Open the file web.config from the Solution Explorer, as shown below:



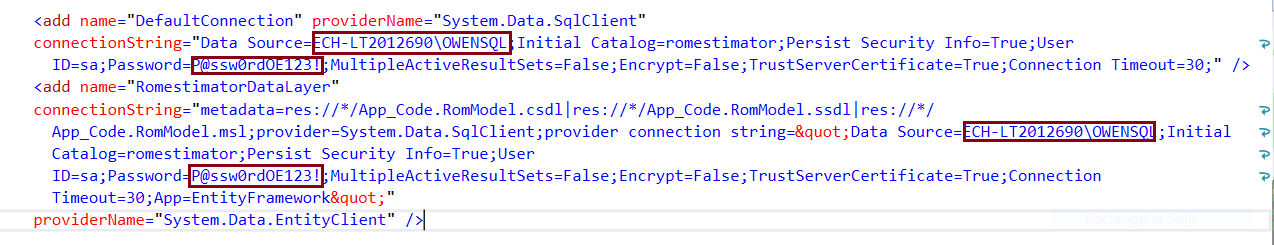
You’ll note that the DefaultConnection and RomestimatorDataLayer are both pointing to the production PaaS database, romestimator-db.database.usgovcloudapi.net. I’ve highlighted the lines below:



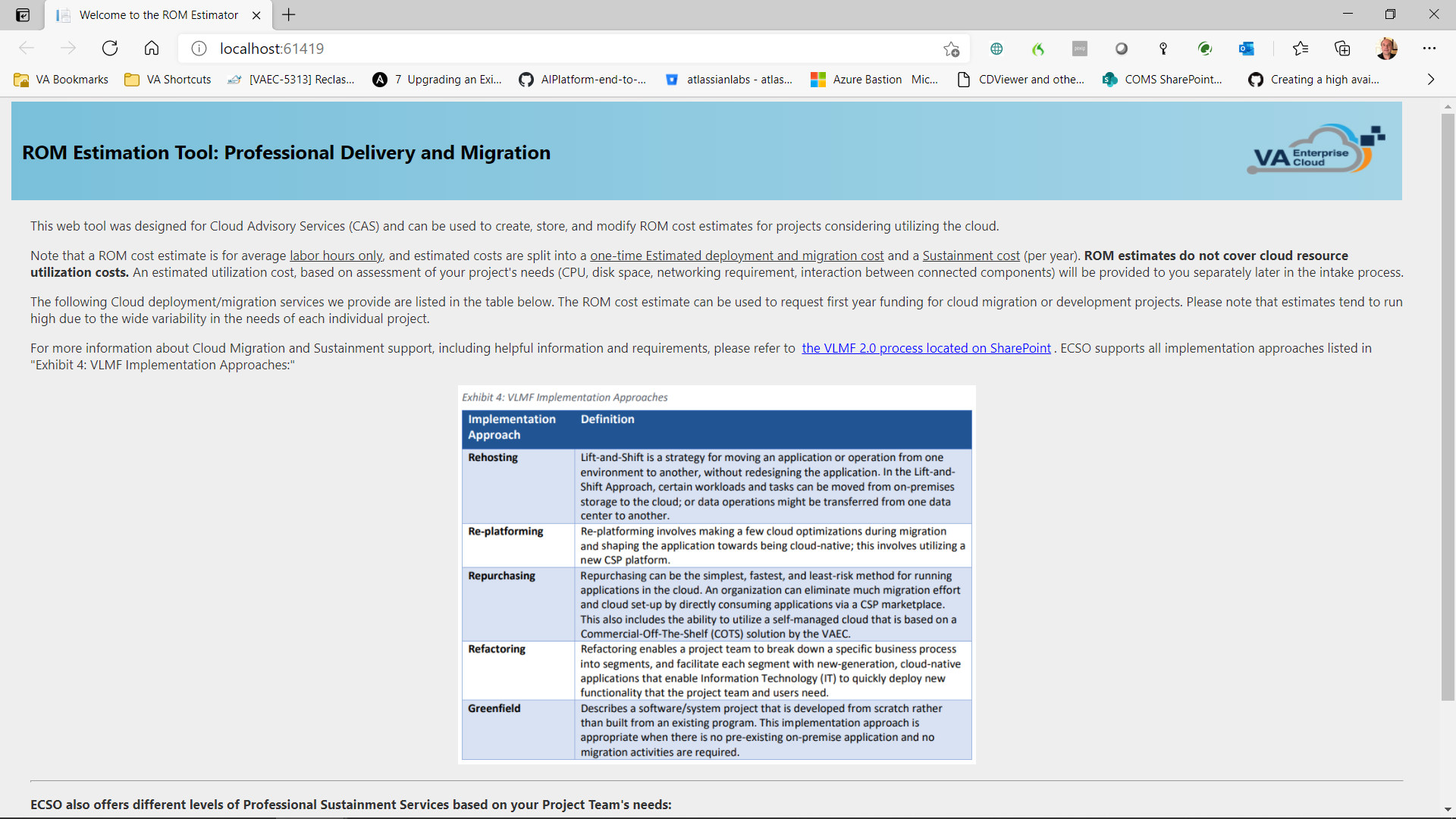
To work on the project locally, you’ll want to comment those lines and uncomment the section above, which looks similar but points to your/my local SQL server romestimator database (ECH-LT2012690\OWENSQL), so that the file looks like this. Make sure to adjust the machine name and the password for your sa SQL account to match your own (currently they match mine). This is illustrated as follows:



Zooming in, this is the data you will need to adjust:



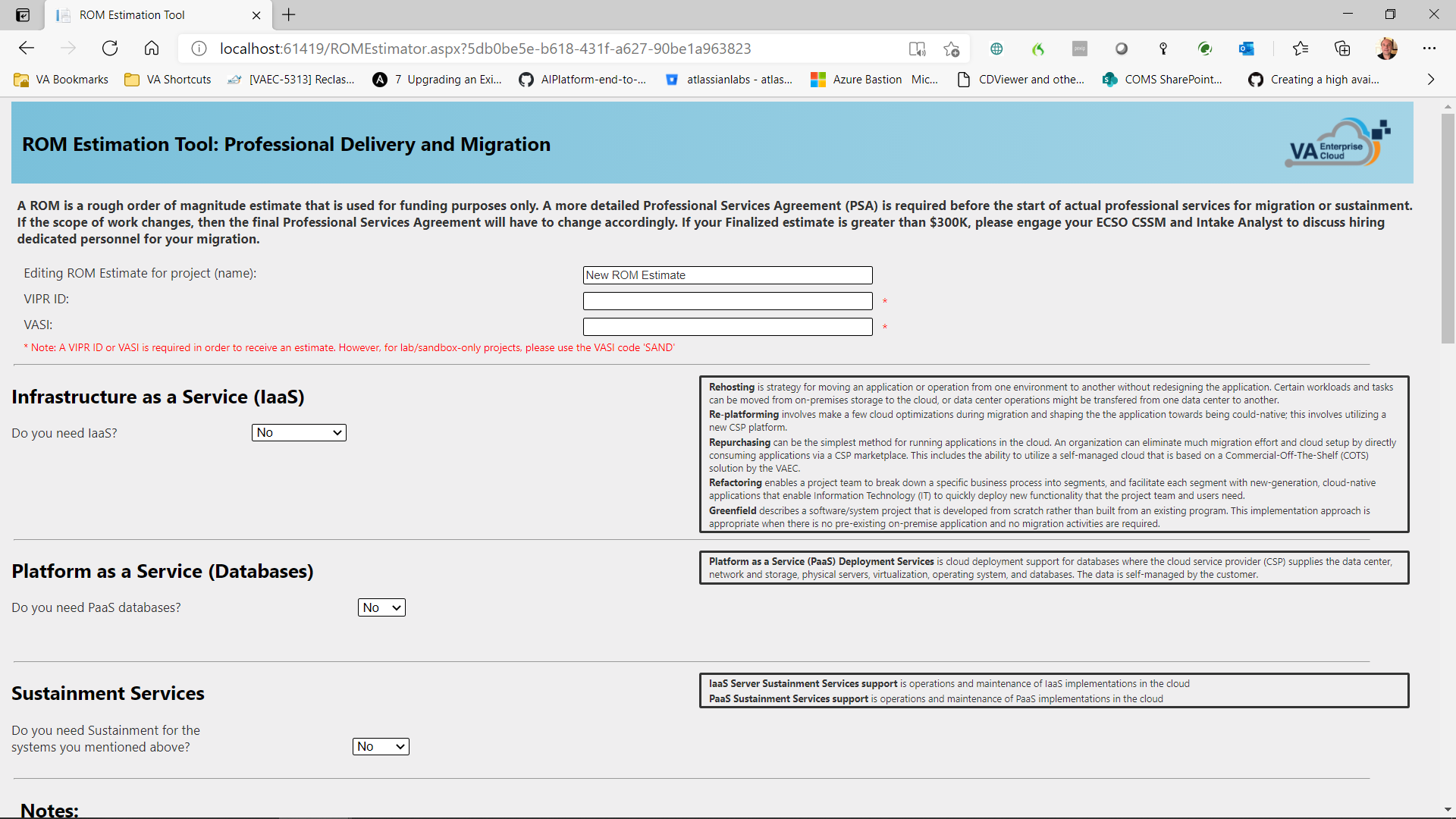
After you have changed the Data Source to match your machine name\SQL instance and the Password for the SQL “sa” account, you can build the project, run and debug the project (shortcut key: F5) against your local database in IIS Express. After a few seconds of compilation, a web browser window will open and you will be redirected to your locally-running copy of the ROM Estimator, shown below.



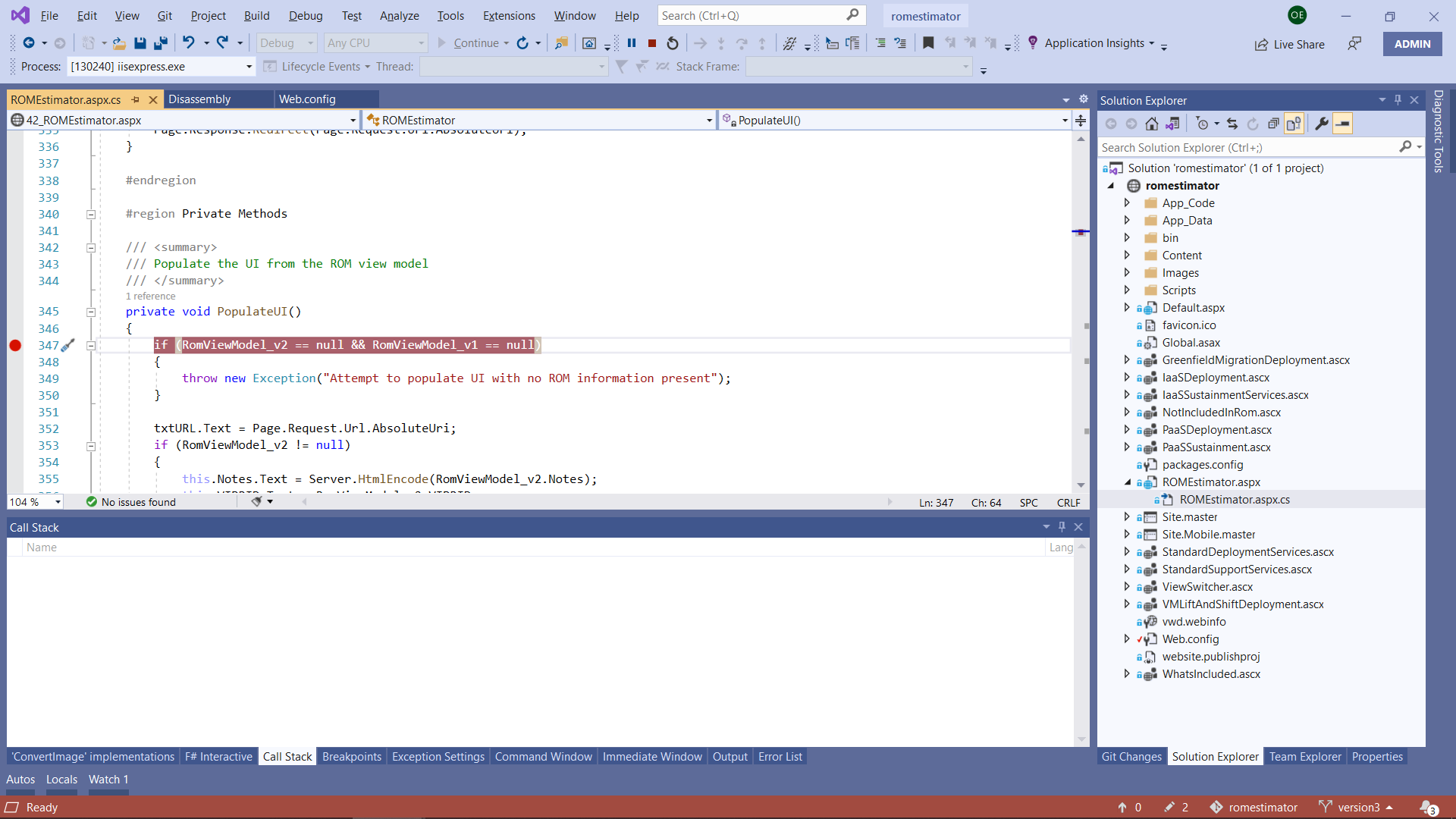
If you scroll down and hit the “Create New ROM” button, you should be taken to the main ROM Estimator page, and you’ll note a new entry if you go into SSMS and “select \* from romestimate\_v2”. If you receive an error page, it’s likely you need to:

1. Ensure your machine name and database instance name are correct in web.config
2. Ensure your sa password is correct in web.config
3. If those look correct, make sure user “sa” has access to the romestimator database in SSMS (it should).

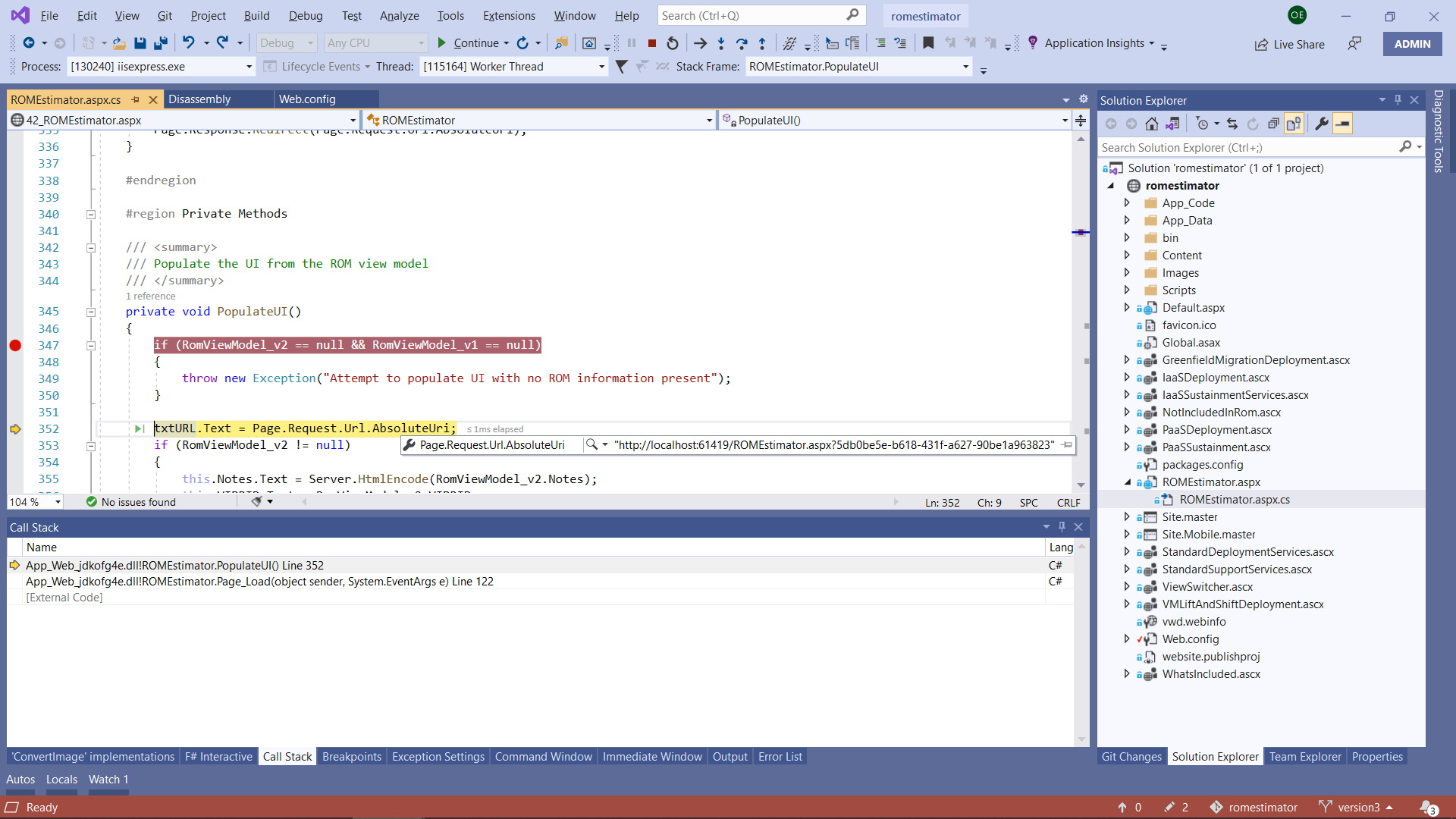
Assuming you’ve successfully created a new ROM, you will be taken to the following main page, locally, which is represented by ROMEstimator.aspx (markup) and ROMEstimator.aspx.cs (code behind).



Note that you can go back to Visual Studio and set Breakpoints (shortcut key: F9) and refresh the page. Visual Studio will break on your breakpoint and you can step through instructions (F10) or into method calls (F11). Here is an example of setting a breakpoint:

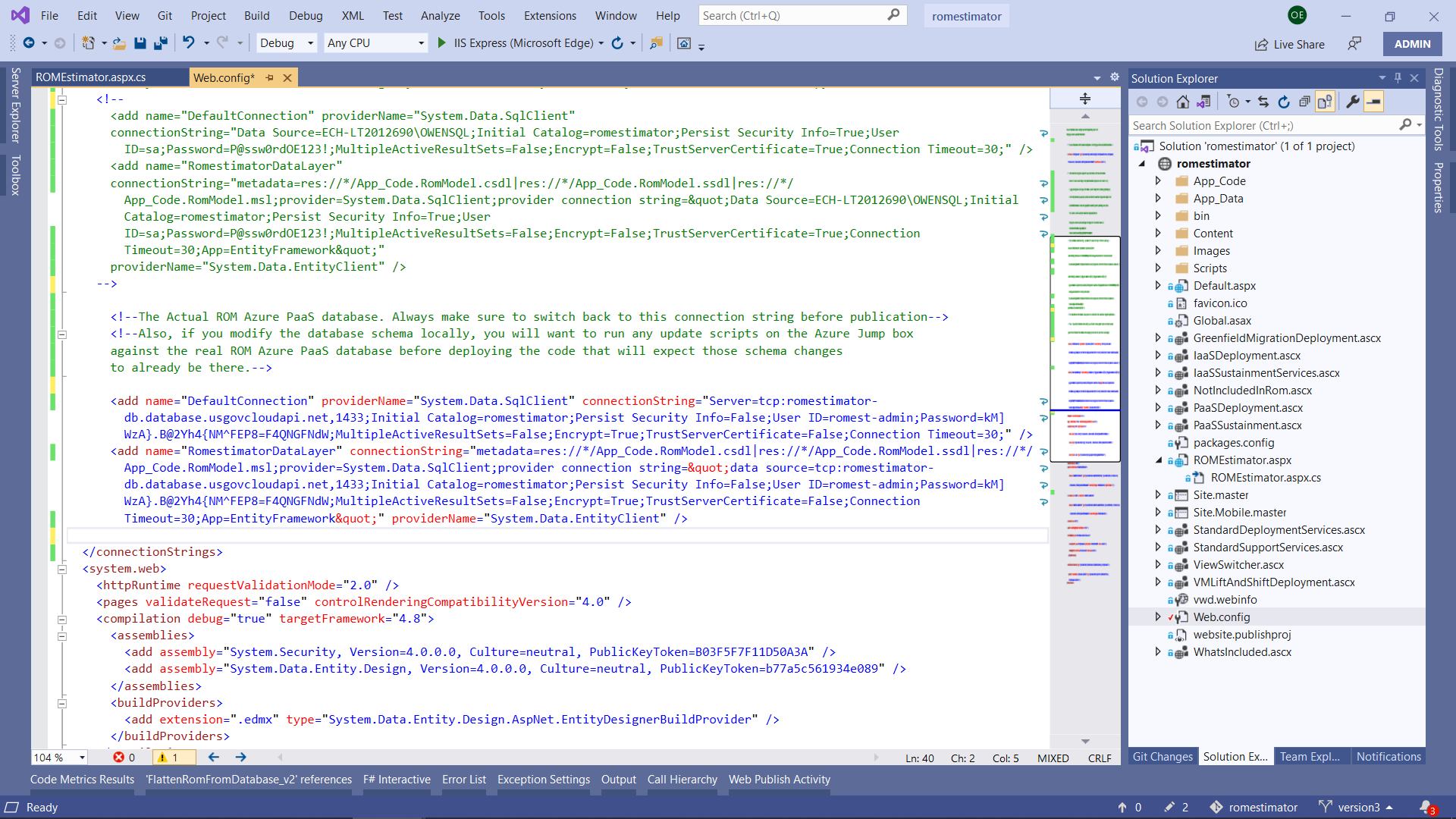


Going back to the browser and hitting refresh, Visual Studio will stop at your breakpoint and you can debug (or even modify code on-the-fly).

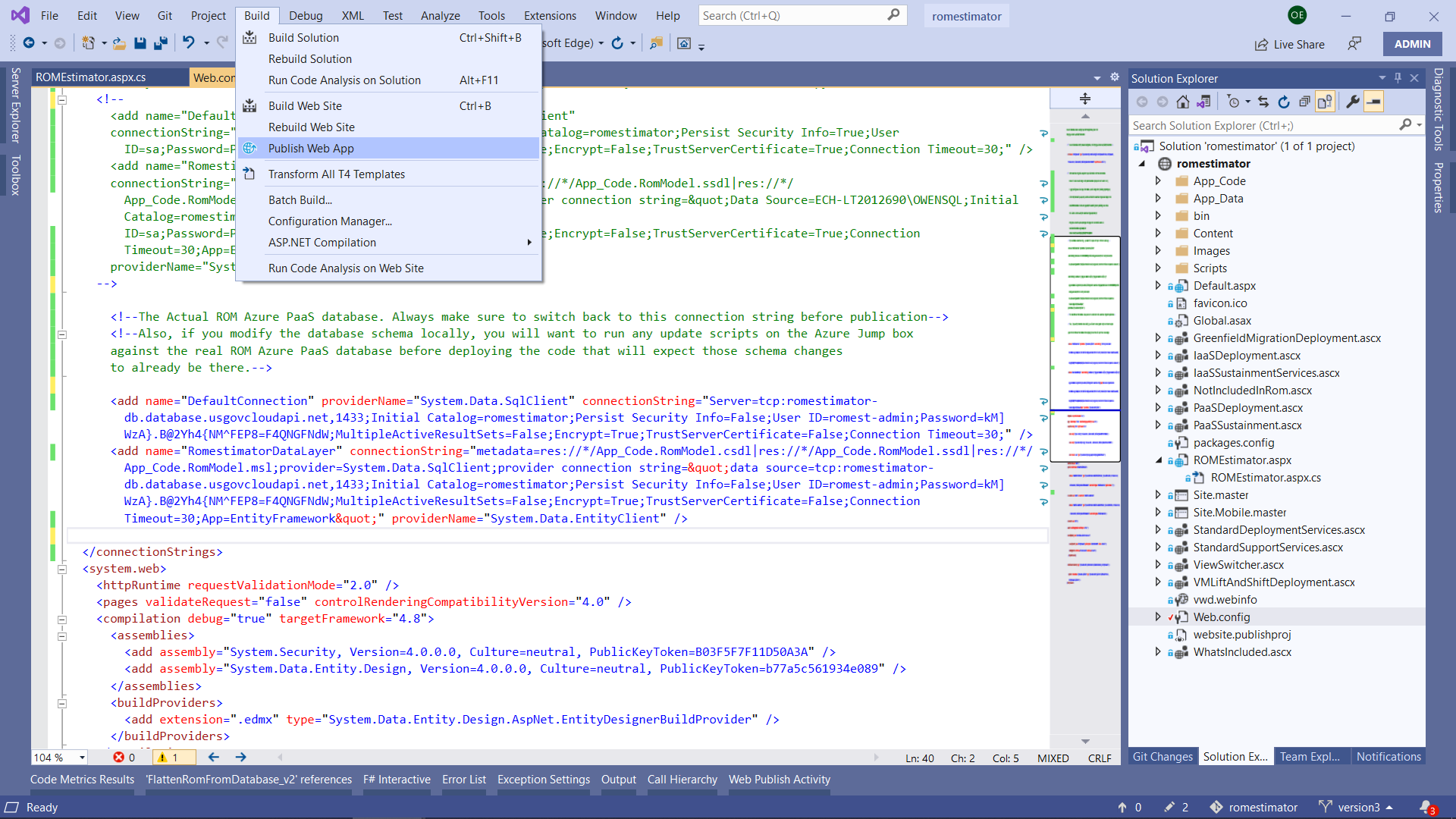


### Publishing your Code Changes to the Azure Web App

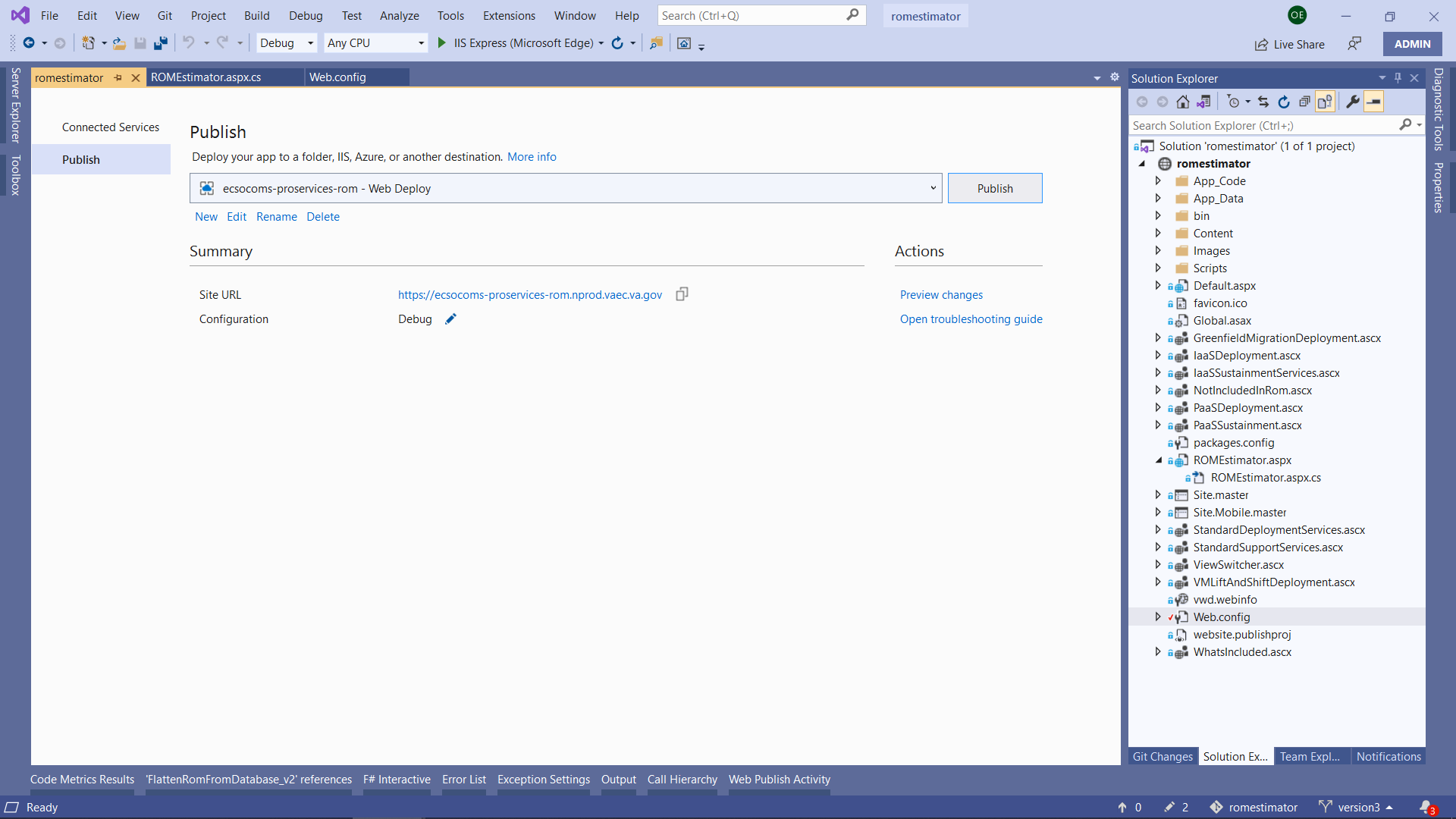
To publish your changes to the NPROD Azure Gov Web App, change your web.config file so that it points back to the Azure PaaS SQL database, as such. Note that you can hit Alt-E then V then M to comment a selected section of code or XML and Alt-E then V then E to uncomment a selected section of code or XML.



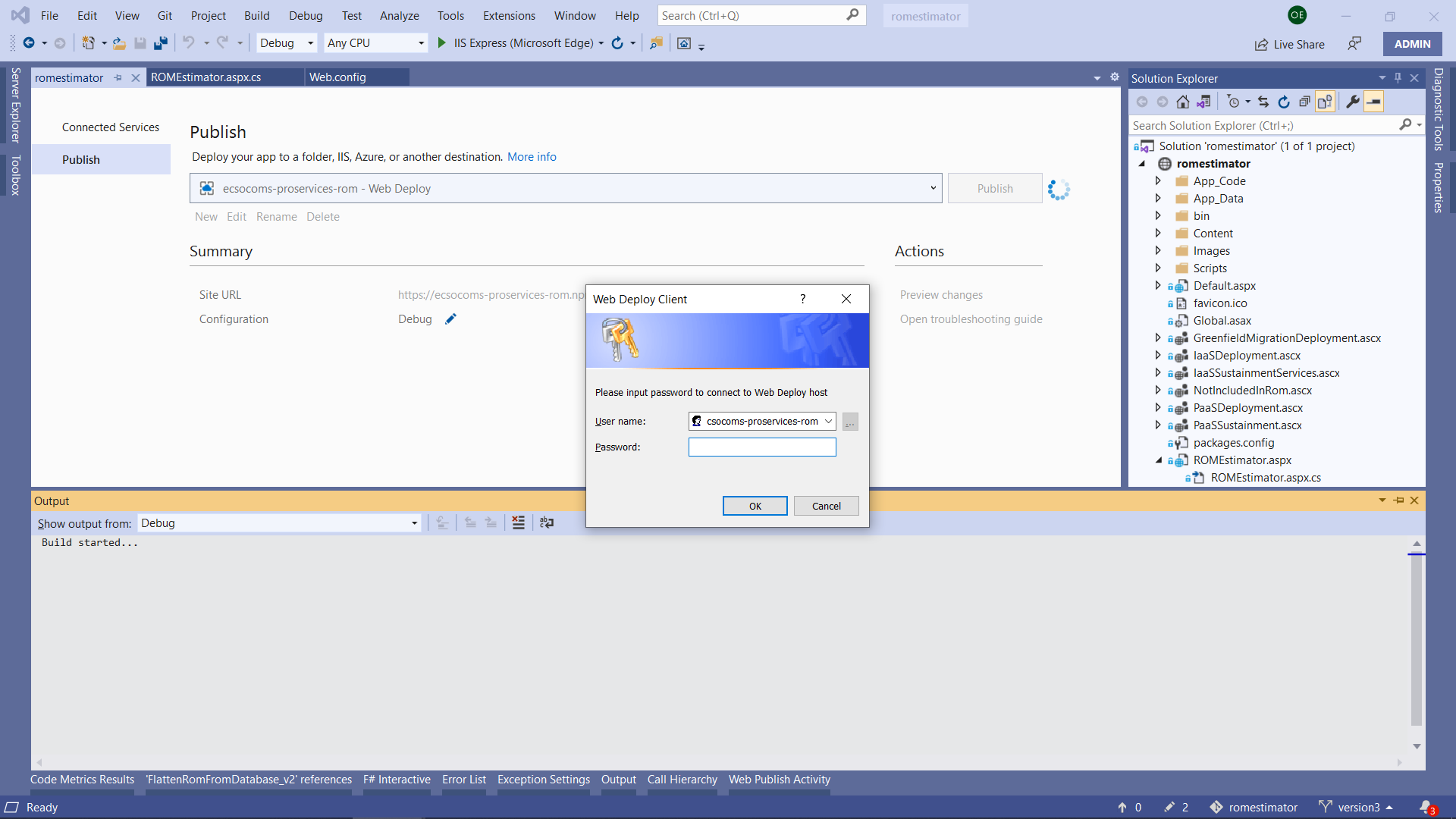
Select Build from the top menu and then Publish Web App (screenshot below). **Make sure you have saved your web.config file with the Azure PaaS Database connection string before you Publish.**



You will see the following screen. Ensure the “ecsocoms-proservices-rom – Web Deploy” publish profile is selected, then hit the Publish button.



You will be prompted for the deployment password, which you can copy/paste from this document, or find in [Appendix B](#_Appendix_B:_Web).



After a short compilation, your code will be deployed and will be immediately available for use at <https://ecsocoms-proservices-rom.nprod.vaec.va.gov/>

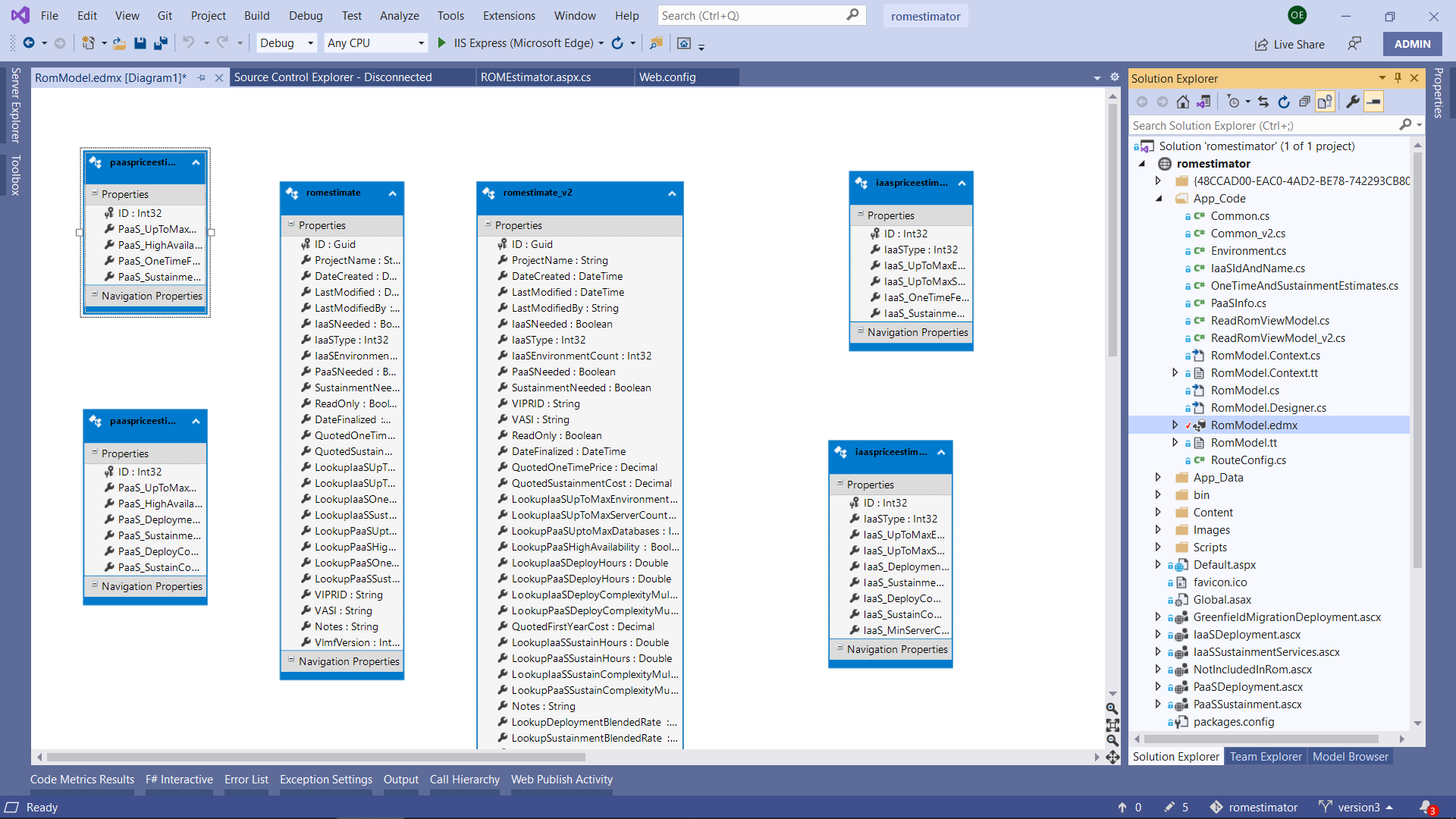
### Database Tables and Stored Procedures (SPROCS)

To make minor database changes (such as modifying pricing information constants and lookup values), I suggest you script out UPDATE statements against your local SQL database, test the pricing changes, and when satisfied, run the same script against the Azure PaaS database. (Note: to do this you will need to log in to the Azure Jump Box and use SSMS from there to connect, since GFE laptops do not allow direct connection over port 1433 to Azure SQL databases).

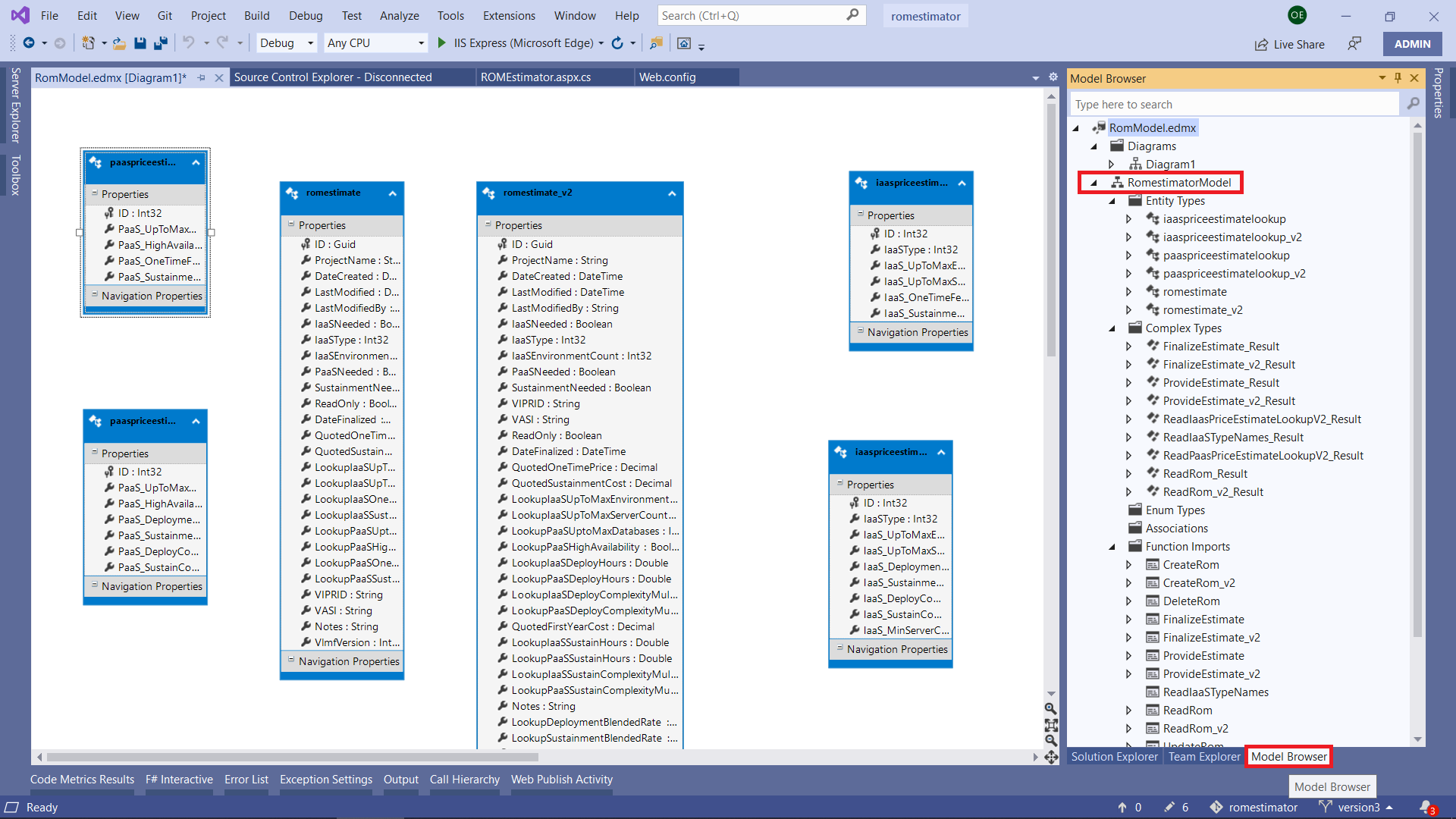
The same principal applies to modifying Stored Procedures. Modify the stored procedure locally, test it out locally, and then copy/paste the generated “Modify” script to the Azure Jump Box where you can use SSMS from there to run the same statement against the Azure PaaS database.

When making schema changes to tables, or altering the parameter or return values types (or names), you must think ahead and make sure:

1. You have considered backwards compatibility. None of the existing data will be automatically updated, so your code will have to handle those “old” records gracefully.
2. You will need to open the EDMX (Entity Data Mapping Model) and synchronize the changes made in the database to its corresponding generated code. To do this for a stored procedure:
   1. Make sure you are pointing to your local database in the web.config file.
   2. Open the EDMX file. You will see a diagram of database tables, as shown below.



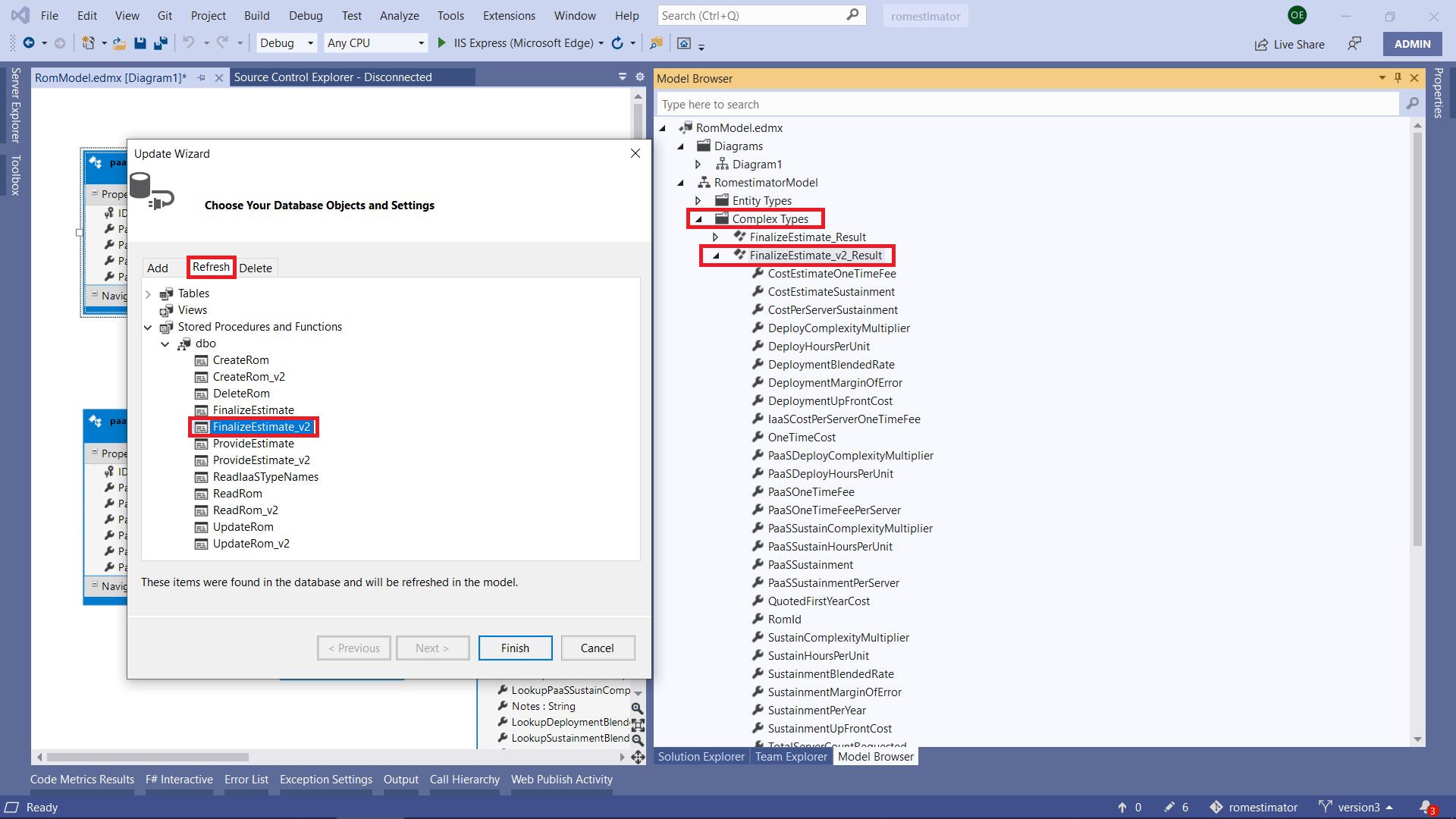
* 1. Click on the Model Browser Tab, highlighted in Red below. Notice that the right-hand half of the screen changes to show a different tree. The tree you are interested in is RomestimatorModel.



* 1. If I’ve made changes to the FinalizeEstimate\_v2 SPROC, I can right-click on Function Imports->FinalizeEstimate\_v2 and select “Update Model from Database,” as seen below:



* 1. You will be taken to the Update Wizard, where you can Refresh the data model (code) with the changes you’ve made to the database.
  2. If you have modified any of the **return type values of the SPROC**, you must update the C# return type, located in the “Complex Types” subitem as follows. Updating will force the Entity Framework’s generated C# code to update the return values based on what the database SPROC now returns. Always, ALWAYS have backwards compatibility in mind!



## Appendices

### Appendix A: All Files, Directory Structures and Methods

|  |  |  |  |
| --- | --- | --- | --- |
| Path | File Name | Method | Description |
| AppCode | Common.cs | Not Applicable | Deprecated. Only used for ROMS created Oct/Nov 2019 |
|  |  |  |  |
| AppCode | Common\_v2.cs | List<IaaSIdAndName> LoadIaaSTypes() | /// Loads all IaaS Types from the database and sorts them by ID |
|  |  |  |  |
|  | string GetUserIdentity(Page page) | | /// Common method to get a user identity. |
|  |  |  | /// For now we are simply using the user's host address with the user agent appended |
|  |  |  | /// <param name="page">The Page/Context from which the request originated</param> |
|  |  |  | /// <returns>A string representation of the user's identity</returns> |
|  |  |  |  |
|  | ReadRomViewModel\_v2 FlattenRomFromDatabase\_v2(List<ReadRom\_v2\_Result> list) | | /// Takes a non-flattened result set from ReadRom and flattens it so all fields contain useful information |
|  |  |  | /// <param name="list">A list of result rows returned from the ReadRom\_v2 sproc call</param> |
|  |  |  | /// <returns>A single ReadRom\_Result that has been flattened for use</returns> |
|  |  |  |  |
|  | FinalizeEstimate\_v2\_Result FinalizeRom\_v2(Page page, ReadRomViewModel\_v2 rom) | | /// A simple wrapper around the FinalizeRom\_v2 SPROC |
|  |  |  |  |
|  | OneTimeAndSustainmentEstimates ProvideEstimate\_v2(ReadRomViewModel\_v2 rom) | | /// A simple wrapper around the ProvideEstimate\_v2 SPROC |
|  |  |  |  |
|  | void UpdateRom\_v2(Page context, ReadRomViewModel\_v2 rom) | | /// A simple wrapper around the UpdateRom\_v2 SPROC |
|  |  |  | /// Flattens the array of environments into the parameters the SPROC is looking for |
|  |  |  |  |
| AppCode | Environment.cs | public class Environment | /// Represents an IaaS Environment - Index, NumberOfServers, and Name of Environment |
| AppCode | IaaSIdAndName.cs | public class IaaSIdAndName | /// Represents an IaaS Type (ID, Name, and associated VLMF Version (1 or 2) |
| AppCode | OneTimeAndSustainmentEstimates.cs | public class OneTimeAndSustainmentEstimates | /// Stores One Time and/or Sustainment Estimates |
| AppCode | PaaSInfo.cs | public class PaaSInfo | /// Information about PaaS - Number of databases Needed, and if High Availability was requested |
|  |  |  |  |
| AppCode | ReadRomViewModel.cs | public class ReadRomViewModel : ReadRom\_Result | /// DEPRECATED after Oct/Nov 2019 /// ReadRomViewModel contains a "flattened" (single-row) ROM that is used for rendering the UI |
|  |  |  |  |
| AppCode | ReadRomViewModel\_v2.cs | public class ReadRomViewModel\_v2 : ReadRom\_v2\_Result | /// ReadRomViewModel contains a "flattened" (single-row) ROM that is used for rendering the UI |
|  |  |  |  |
| AppCode | RomModel.Context.tt/.cs | Transformation template for Entity Data Model | Generated by .EDMX |
| AppCode | **RomModel.EDMX** | Primary Entity/Data Mapping for SPROCS and DB Tables | **Generates RomModel.Context.tt and RomModel.tt and their respective DB-to-C# class mappings** |
|  |  |  | **You must point to a local copy of the ROM SQL database to modify or update the Entity Data Model** |
| AppCode | RomModel.tt/.cs | Transformation template for Entity Data Model | Generated by .EDMX |
|  |  |  |  |
| AppCode | RouteConfig.cs | class RouteConfig | /// ASP.NET generic wireup code (generated). |
|  |  |  |  |
| AppData/PublishProfiles | ecsocoms-proservices-rom - Web Deploy.pubxml | | Contains all data (except the password) needed to publish the ROM Estimator Application to NPROD via the "Publish" menu option |
|  |  |  |  |
| bin |  |  | Do not touch, binary dependencies |
|  |  |  |  |
| Content | Site.css |  | CSS classes and styles for all HTML rendered |
|  |  |  |  |
| Images | ECSO Small Logo.png | | ECSO Logo |
| Images | VAEC Logo.png |  | VAEC Logo |
| Images | VLMF 2 IaaS Types.png | | An image showing VLMF 2.0 IaaS types and descriptions |
|  |  |  |  |
| Scripts |  |  | All useless. Contains jQuery and modernizr, standard with ASP.NET |
|  |  |  |  |
| ./ | Default.aspx |  | Contains the markup for the "Create New ROM" page |
| ./ | Default.aspx.cs | void btnNewROM\_Click | /// Button handler for "Create New ROM" |
|  |  |  |  |
| ./ | favicon.ico |  | Icon used for the web site and when adding to Favorites |
| ./ | Global.asax |  | Generated code by ASP.NET, no need to modify |
| ./ | GreenfieldMigrationDeployment.ascx | | Markup used in What's Included for Greenfield IaaS Types |
| ./ | IaaSDeployment.ascx | | Markup used in What's Included for IaaS Deployment Types |
| ./ | IaaSSustainmentServices.ascx | | Markup used in What's Included for standard sustainment services |
| ./ | NotIncludedInRom.ascx | | Markup used in What's NOT Included in a ROM estimate |
| ./ | PaaSDeployment.ascx | | Marked used in What's Included when PaaS Deployment is selected |
| ./ | PaaSSustainment.ascx | | Marked used in What's Included when PaaS Sustainment is selected |
| ./ | StandardDeploymentServices.ascx | | Markup used in What's Included when any Deployment services are requested |
| ./ | StandardSupportServices.ascx | | Markup used in What's Included when any Sustainment services are requested |
| ./ | VMLiftAndShiftDeployment.ascx | | Marked used in What's Included for Lift and Shift Deployment IaaS Types |
|  |  |  |  |
| ./ | packages.config | | NuGet managed dependencies |
| ./ | Site.Mobile.master | | Unused. We don't support mobile. |
| ./ | ViewSwitcher.ascx | | Unused. We aren't a single page app (SPA) |
|  |  |  |  |
| ./ | Site.master |  | All markup that is common to all pages, including javascript for (Default.aspx and RomEstimator.aspx) |
| ./ | Site.master.cs |  | Generated code. Have not modified/no need to modify. |
|  |  |  |  |
| ./ | ROMEstimator.aspx | | The HTML markup for the main ROM Estimator editing/viewing page |
| private ReadRomViewModel\_v2 RomViewModel\_v2; | | | /// Stores the v2 (most recent) view model of the current ROM being worked on in a way that |
|  |  |  | /// presents field-friendly interaction with the UI |
|  |  |  |  |
| private List<IaaSIdAndName> iaaSTypeNames; | | | /// Stores the IaaS Type Name classes (ID, Name, VLMF version) |
|  |  |  |  |
| private List<ListItem> iaaSTypeListItems; | | | /// Stores the list of List Items that will be rendered, showing which IaaS type names |
|  |  |  | /// are selectable for this VLMF version (VLMF 1.0 or 2.0+) |
|  |  |  |  |
| protected override void OnPreLoad | | | /// This method is executed regardless of circumstance. Its sole purpose is to populate |
|  |  |  | /// the IaaS type names (all of them) in the dropdown. When the ROM is then loaded, the type |
|  |  |  | /// names available may be trimmed down based on VLMF version v1 or v2+ |
|  |  |  |  |
| protected void Page\_Load | | | /// Executed whenever the page loads for a user or "Posts Back" data, such as when a button is clicked |
|  |  |  | /// or a dropdown changes. Therefore, we check the Page.IsPostBack to determine whether to maintain |
|  |  |  | /// the current form data or reload and repopulate the UI from the ROM data record in the database |
|  |  |  | /// This also trims out VLMF IaaS options that may not be available for the ROM's VLMF version |
|  |  |  |  |
|  |  |  |  |
| private void InvalidateCostEstimate(DropDownList controlChanged) | | | /// If any of the dropdown controls changed, invalidate the price so that we aren't left showing an invalid |
|  |  |  | /// price estimate when something like, number of environments, or IaaS Type, or Needs Sustainment is changed |
|  |  |  | /// <param name="controlChanged"></param> |
|  |  |  |  |
| protected void dropdownNeedIaaS\_SelectedIndexChanged | | | /// User indicated a change in whether they need IaaS (yes/no) |
|  |  |  |  |
| protected void dropdownNumberOfIaaSEnvironments\_SelectedIndexChanged | | | /// User indicated a change in number of environments (2-5 allowed) needed. |
|  |  |  |  |
| protected void dropdownPaaSNeeded\_SelectedIndexChanged | | | /// User indicated a change in whether PaaS is needed (Y/N) |
|  |  |  |  |
| protected void dropdownSustainmentNeeded\_SelectedIndexChanged | | | /// User indicated a change in whether Sustainment is needed (Y/N) |
|  |  |  |  |
| protected void dropdownIaaSType\_SelectedIndexChanged | | | /// User indicated a change in the TYPE of IaaS being requested (Greenfield vs. Iaas Deployment, etc.) |
|  |  |  |  |
| protected void btnUpdateEstimate\_Click | | | /// Handles the Update Estimate button click |
|  |  |  |  |
| protected void btnSaveROM\_Click | | | /// Handles the Save ROM button click |
|  |  |  |  |
| protected void btnFinalizeROM\_Click | | | /// Handles the Finalize ROM button click |
|  |  |  |  |
| private void PopulateUI() | | | /// Populate the UI from the ROM view model |
|  |  |  |  |
| private void MakeFormReadOnly() | | | /// Makes the form read-only. Blocks any user-initiated changes. |
|  |  |  |  |
| private bool CanProvideEstimate() | | | /// Determines if we can provide a Cost Estimate for this ROM |
|  |  |  |  |
| private void PopulateRomViewModelFromUI() | | | /// Reads the current UI changes the user has made back into the ROM ViewModel |
|  |  |  | /// Also performs an additional validation on fields that may not have been populated with valid values, |
|  |  |  | /// although the Validate methods should have been called prior to this call. |
|  |  |  |  |
| private bool ValidateCount(TextBox txtCount, Control focusOn, Label labelValidationError, bool showVisually) | | | /// Ensures the textbox contains a valid number (and not a letter or some such nonsense) |
|  |  |  |  |
| private bool PerformValidation(bool showVisually) | | | /// Performs validations on the ROM to make sure the user has not entered garbage into the text fields, etc. |
|  |  |  |  |
| private void ForceSetFocus(Control control) | | | /// Sets the selected focus on a specific Control/field |
|  |  |  |  |
| private bool IsValidNumber(string text) | | | /// Checks if the string is a valid number |
|  |  |  |  |
| private bool SaveROM() | | | /// Saves the ROM, but only if validation passes. This ensures that all ROMs in the database are either |
|  |  |  | /// incomplete or complete, but none contain bad values (such as numbers where an integer should be, etc.) |
|  |  |  |  |
| private bool PopulateEstimate() | | | /// Performs validation on newly input user values, and if the validation passes, saves the ROM and |
|  |  |  | /// attempts to get estimated costs |
|  |  |  |  |
| private void PopulateEstimateFromViewModel() | | | /// Populates the ROM Estimated Cost fields to the best of its ability from the current RomViewModel (\_v2) |
|  |  |  | /// Validation should've occurred prior to this call (i.e. this call should not be called directly, |
|  |  |  | /// but instead PopulateEstimate() should be used instead). |

**NOTE: All methods, fields, and properties are all commented in the code. Hovering over any call to the method will show the help for the method, including parameters and return value(s).**

### Appendix B: Web Publish Profile

The master Web Publish Profile can be found below:

<publishData>

<publishProfile profileName="ecsocoms-proservices-rom - Web Deploy" publishMethod="MSDeploy" publishUrl="ecsocoms-proservices-rom.scm.nprod.vaec.va.gov:443" msdeploySite="ecsocoms-proservices-rom" userName="$ecsocoms-proservices-rom" userPWD="" destinationAppUrl="https://ecsocoms-proservices-rom.nprod.vaec.va.gov" SQLServerDBConnectionString="" mySQLDBConnectionString="" hostingProviderForumLink="" controlPanelLink="https://manage.windowsazure.us" webSystem="WebSites">  
 <databases />  
</publishProfile>

</publishData>

### Appendix C: SQL Script to Create romestimator database locally

The following script can be run in a SQL Server Management Studio (SSMS) query window to create the latest version of the romestimator database, with pricing lookup data. Simply copy/paste the text into the query window and adjust the path name to adjust for your SQL database instance name (highlighted in yellow below).

USE [master]

GO

/\*\*\*\*\*\* Object: Database [romestimator] Script Date: 5/19/2021 1:49:53 PM \*\*\*\*\*\*/

CREATE DATABASE [romestimator]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'romestimator', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.OWENSQL\MSSQL\DATA\romestimator.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'romestimator\_log', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.OWENSQL\MSSQL\DATA\romestimator\_log.ldf' , SIZE = 73728KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

WITH CATALOG\_COLLATION = DATABASE\_DEFAULT

GO

ALTER DATABASE [romestimator] SET COMPATIBILITY\_LEVEL = 150

GO

IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled'))

begin

EXEC [romestimator].[dbo].[sp\_fulltext\_database] @action = 'enable'

end

GO

ALTER DATABASE [romestimator] SET ANSI\_NULL\_DEFAULT OFF

GO

ALTER DATABASE [romestimator] SET ANSI\_NULLS OFF

GO

ALTER DATABASE [romestimator] SET ANSI\_PADDING OFF

GO

ALTER DATABASE [romestimator] SET ANSI\_WARNINGS OFF

GO

ALTER DATABASE [romestimator] SET ARITHABORT OFF

GO

ALTER DATABASE [romestimator] SET AUTO\_CLOSE OFF

GO

ALTER DATABASE [romestimator] SET AUTO\_SHRINK OFF

GO

ALTER DATABASE [romestimator] SET AUTO\_UPDATE\_STATISTICS ON

GO

ALTER DATABASE [romestimator] SET CURSOR\_CLOSE\_ON\_COMMIT OFF

GO

ALTER DATABASE [romestimator] SET CURSOR\_DEFAULT GLOBAL

GO

ALTER DATABASE [romestimator] SET CONCAT\_NULL\_YIELDS\_NULL OFF

GO

ALTER DATABASE [romestimator] SET NUMERIC\_ROUNDABORT OFF

GO

ALTER DATABASE [romestimator] SET QUOTED\_IDENTIFIER OFF

GO

ALTER DATABASE [romestimator] SET RECURSIVE\_TRIGGERS OFF

GO

ALTER DATABASE [romestimator] SET DISABLE\_BROKER

GO

ALTER DATABASE [romestimator] SET AUTO\_UPDATE\_STATISTICS\_ASYNC OFF

GO

ALTER DATABASE [romestimator] SET DATE\_CORRELATION\_OPTIMIZATION OFF

GO

ALTER DATABASE [romestimator] SET TRUSTWORTHY OFF

GO

ALTER DATABASE [romestimator] SET ALLOW\_SNAPSHOT\_ISOLATION OFF

GO

ALTER DATABASE [romestimator] SET PARAMETERIZATION SIMPLE

GO

ALTER DATABASE [romestimator] SET READ\_COMMITTED\_SNAPSHOT OFF

GO

ALTER DATABASE [romestimator] SET HONOR\_BROKER\_PRIORITY OFF

GO

ALTER DATABASE [romestimator] SET RECOVERY FULL

GO

ALTER DATABASE [romestimator] SET MULTI\_USER

GO

ALTER DATABASE [romestimator] SET PAGE\_VERIFY CHECKSUM

GO

ALTER DATABASE [romestimator] SET DB\_CHAINING OFF

GO

ALTER DATABASE [romestimator] SET FILESTREAM( NON\_TRANSACTED\_ACCESS = OFF )

GO

ALTER DATABASE [romestimator] SET TARGET\_RECOVERY\_TIME = 60 SECONDS

GO

ALTER DATABASE [romestimator] SET DELAYED\_DURABILITY = DISABLED

GO

ALTER DATABASE [romestimator] SET ACCELERATED\_DATABASE\_RECOVERY = OFF

GO

EXEC sys.sp\_db\_vardecimal\_storage\_format N'romestimator', N'ON'

GO

ALTER DATABASE [romestimator] SET QUERY\_STORE = OFF

GO

USE [romestimator]

GO

/\*\*\*\*\*\* Object: Table [dbo].[environments] Script Date: 5/19/2021 1:49:53 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[environments](

[ID] [int] IDENTITY(1,1) NOT NULL,

[ROMID] [uniqueidentifier] ROWGUIDCOL NOT NULL,

[EnvironmentIndex] [int] NULL,

[EnvironmentName] [nvarchar](max) NULL,

[ServerOrVMCount] [int] NULL,

CONSTRAINT [PK\_environments] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[environments\_v2] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[environments\_v2](

[ID] [int] IDENTITY(1,1) NOT NULL,

[ROMID] [uniqueidentifier] ROWGUIDCOL NOT NULL,

[EnvironmentIndex] [int] NULL,

[EnvironmentName] [nvarchar](max) NULL,

[ServerOrVMCount] [int] NULL,

CONSTRAINT [PK\_environments\_v2] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[iaaspriceestimatelookup] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[iaaspriceestimatelookup](

[ID] [int] IDENTITY(1,1) NOT NULL,

[IaaSType] [int] NOT NULL,

[IaaS\_UpToMaxEnvironmentCount] [int] NOT NULL,

[IaaS\_UpToMaxServerCount] [int] NOT NULL,

[IaaS\_OneTimeFee] [money] NOT NULL,

[IaaS\_SustainmentCostPerYear] [money] NOT NULL,

CONSTRAINT [PK\_IaaSPriceEstimateLookup] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[iaaspriceestimatelookup\_v2] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[iaaspriceestimatelookup\_v2](

[ID] [int] IDENTITY(1,1) NOT NULL,

[IaaSType] [int] NOT NULL,

[IaaS\_UpToMaxEnvironmentCount] [int] NOT NULL,

[IaaS\_MinServerCount] [int] NULL,

[IaaS\_UpToMaxServerCount] [int] NOT NULL,

[IaaS\_DeploymentHours] [float] NULL,

[IaaS\_SustainmentHours] [float] NULL,

[IaaS\_DeployComplexityMultiplier] [float] NULL,

[IaaS\_SustainComplexityMultiplier] [float] NULL,

CONSTRAINT [PK\_IaaSPriceEstimateLookup\_v2] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[iaastypenames] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[iaastypenames](

[ID] [int] IDENTITY(1,1) NOT NULL,

[IaasTypeName] [nvarchar](max) NOT NULL,

[VlmfVersion] [int] NULL,

CONSTRAINT [PK\_iaastypenames] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[paas] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[paas](

[ID] [int] IDENTITY(1,1) NOT NULL,

[ROMID] [uniqueidentifier] NOT NULL,

[DatabaseInstanceCount] [int] NULL,

[InstancesHighlyAvailable] [bit] NULL,

CONSTRAINT [PK\_paas] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[paas\_v2] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[paas\_v2](

[ID] [int] IDENTITY(1,1) NOT NULL,

[ROMID] [uniqueidentifier] NOT NULL,

[DatabaseInstanceCount] [int] NULL,

[InstancesHighlyAvailable] [bit] NULL,

CONSTRAINT [PK\_paas\_v2] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[paaspriceestimatelookup] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[paaspriceestimatelookup](

[ID] [int] IDENTITY(1,1) NOT NULL,

[PaaS\_UpToMaxDatabaseCount] [int] NOT NULL,

[PaaS\_HighAvailability] [bit] NOT NULL,

[PaaS\_OneTimeFee] [money] NOT NULL,

[PaaS\_SustainmentCostPerYear] [money] NOT NULL,

CONSTRAINT [PK\_paaspriceestimatelookup] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[paaspriceestimatelookup\_v2] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[paaspriceestimatelookup\_v2](

[ID] [int] IDENTITY(1,1) NOT NULL,

[PaaS\_UpToMaxDatabaseCount] [int] NOT NULL,

[PaaS\_HighAvailability] [bit] NOT NULL,

[PaaS\_DeploymentHours] [float] NULL,

[PaaS\_SustainmentHours] [float] NULL,

[PaaS\_DeployComplexityMultiplier] [float] NULL,

[PaaS\_SustainComplexityMultiplier] [float] NULL,

CONSTRAINT [PK\_paaspriceestimatelookup\_v2] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[pricingfactors\_v2] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[pricingfactors\_v2](

[ID] [int] IDENTITY(1,1) NOT NULL,

[DeploymentMarginOfError] [float] NOT NULL,

[SustainmentMarginOfError] [float] NOT NULL,

[DeploymentBlendedHourlyCost] [money] NOT NULL,

[SustainmentBlendedHourlyCost] [money] NOT NULL,

[DeploymentUpFrontCost] [money] NOT NULL,

[SustainmentUpFrontCost] [money] NOT NULL,

[VlmfVersion] [int] NULL

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[romestimate] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[romestimate](

[ID] [uniqueidentifier] ROWGUIDCOL NOT NULL,

[ProjectName] [nvarchar](max) NOT NULL,

[DateCreated] [datetime] NULL,

[LastModified] [datetime] NULL,

[LastModifiedBy] [nvarchar](max) NULL,

[IaaSNeeded] [bit] NOT NULL,

[IaaSType] [int] NULL,

[IaaSEnvironmentCount] [int] NULL,

[PaaSNeeded] [bit] NOT NULL,

[SustainmentNeeded] [bit] NOT NULL,

[VIPRID] [nvarchar](max) NULL,

[VASI] [nvarchar](max) NULL,

[ReadOnly] [bit] NOT NULL,

[DateFinalized] [datetime] NULL,

[QuotedOneTimePrice] [money] NULL,

[QuotedSustainmentCost] [money] NULL,

[LookupIaaSUpToMaxEnvironments] [int] NULL,

[LookupIaaSUpToMaxServerCount] [int] NULL,

[LookupIaaSOneTimeFee] [money] NULL,

[LookupIaaSSustainmentCost] [money] NULL,

[LookupPaaSUptoMaxDatabases] [int] NULL,

[LookupPaaSHighAvailability] [bit] NULL,

[LookupPaaSOneTimeFee] [money] NULL,

[LookupPaaSSustainmentCost] [money] NULL,

[QuotedFirstYearCost] [money] NULL,

[Notes] [nvarchar](max) NULL,

[VlmfVersion] [int] NULL,

CONSTRAINT [PK\_romestimate\_1] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[romestimate\_v2] Script Date: 5/19/2021 1:49:54 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[romestimate\_v2](

[ID] [uniqueidentifier] ROWGUIDCOL NOT NULL,

[ProjectName] [nvarchar](max) NOT NULL,

[DateCreated] [datetime] NULL,

[LastModified] [datetime] NULL,

[LastModifiedBy] [nvarchar](max) NULL,

[IaaSNeeded] [bit] NOT NULL,

[IaaSType] [int] NULL,

[IaaSEnvironmentCount] [int] NULL,

[PaaSNeeded] [bit] NOT NULL,

[SustainmentNeeded] [bit] NOT NULL,

[VIPRID] [nvarchar](max) NULL,

[VASI] [nvarchar](max) NULL,

[ReadOnly] [bit] NOT NULL,

[DateFinalized] [datetime] NULL,

[QuotedOneTimePrice] [money] NULL,

[QuotedSustainmentCost] [money] NULL,

[LookupIaaSUpToMaxEnvironments] [int] NULL,

[LookupIaaSUpToMaxServerCount] [int] NULL,

[LookupPaaSUptoMaxDatabases] [int] NULL,

[LookupPaaSHighAvailability] [bit] NULL,

[LookupIaaSDeployHours] [float] NULL,

[LookupPaaSDeployHours] [float] NULL,

[LookupIaaSDeployComplexityMultiplier] [float] NULL,

[LookupPaaSDeployComplexityMultiplier] [float] NULL,

[LookupDeploymentBlendedRate] [money] NULL,

[LookupDeploymentMarginOfError] [float] NULL,

[QuotedFirstYearCost] [money] NULL,

[LookupIaaSSustainHours] [float] NULL,

[LookupPaaSSustainHours] [float] NULL,

[LookupIaaSSustainComplexityMultiplier] [float] NULL,

[LookupPaaSSustainComplexityMultiplier] [float] NULL,

[Notes] [nvarchar](max) NULL,

[LookupSustainmentMarginOfError] [float] NULL,

[LookupDeploymentUpFrontCost] [money] NULL,

[LookupSustainmentBlendedRate] [money] NULL,

[LookupSustainmentUpFrontCost] [money] NULL,

[VlmfVersion] [int] NULL,

CONSTRAINT [PK\_romestimate\_1\_v2] PRIMARY KEY CLUSTERED

(

[ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

SET IDENTITY\_INSERT [dbo].[environments\_v2] ON

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (982, N'23d131b4-3ddf-434b-9764-2d7611799257', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (983, N'23d131b4-3ddf-434b-9764-2d7611799257', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1002, N'171f29c5-95c5-49d8-86ec-4e8c30157ef6', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1003, N'171f29c5-95c5-49d8-86ec-4e8c30157ef6', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1008, N'76101459-573b-4a8a-bae1-24f777eeb98d', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1009, N'76101459-573b-4a8a-bae1-24f777eeb98d', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1012, N'108325d6-9f4e-4750-aa13-4ffceada7ac1', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1013, N'108325d6-9f4e-4750-aa13-4ffceada7ac1', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1016, N'7eaac442-3f52-46d7-9987-1fdee3798f79', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1017, N'7eaac442-3f52-46d7-9987-1fdee3798f79', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1018, N'9e7392f2-5b31-443b-ae6b-6079915083ae', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1019, N'9e7392f2-5b31-443b-ae6b-6079915083ae', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1026, N'7c1aa223-0258-4921-a772-c10a0e6f262c', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1027, N'7c1aa223-0258-4921-a772-c10a0e6f262c', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1030, N'93aee4f7-46dd-4ab9-b1e5-b86932b13b96', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1031, N'93aee4f7-46dd-4ab9-b1e5-b86932b13b96', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1034, N'3a211ef5-baf6-4bac-a8b8-43578d6bd83b', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1035, N'3a211ef5-baf6-4bac-a8b8-43578d6bd83b', 2, N'PREPROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1038, N'cf3964f2-65ee-4ccb-9398-276a0aba7639', 1, N'PROD', 1)

INSERT [dbo].[environments\_v2] ([ID], [ROMID], [EnvironmentIndex], [EnvironmentName], [ServerOrVMCount]) VALUES (1039, N'cf3964f2-65ee-4ccb-9398-276a0aba7639', 2, N'PREPROD', 1)

SET IDENTITY\_INSERT [dbo].[environments\_v2] OFF

GO

SET IDENTITY\_INSERT [dbo].[iaaspriceestimatelookup] ON

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (1, 1, 1, 1, 7677.0000, 34997.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (2, 1, 1, 3, 36847.0000, 157487.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (3, 1, 1, 10, 184236.0000, 699940.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (4, 1, 1, 20, 491296.0000, 2099820.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (5, 2, 3, 5, 44243.5000, 502455.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (6, 2, 3, 10, 88487.0000, 1507365.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (7, 2, 4, 15, 176974.0000, 4019640.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (8, 2, 5, 25, 370540.0000, 12561375.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (9, 3, 3, 5, 345443.0000, 502455.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (10, 3, 3, 10, 690885.0000, 1507365.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (11, 3, 4, 15, 1381770.0000, 4019640.0000)

INSERT [dbo].[iaaspriceestimatelookup] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_UpToMaxServerCount], [IaaS\_OneTimeFee], [IaaS\_SustainmentCostPerYear]) VALUES (12, 3, 5, 25, 2936262.0000, 12561375.0000)

SET IDENTITY\_INSERT [dbo].[iaaspriceestimatelookup] OFF

GO

SET IDENTITY\_INSERT [dbo].[iaaspriceestimatelookup\_v2] ON

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (1, 1, 1, 0, 1, 8, 2.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (2, 1, 1, 1, 3, 8, 2.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (3, 1, 1, 3, 10, 7, 2.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (4, 1, 1, 10, 20, 7, 2.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (5, 1, 1, 20, 100, 4, 3, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (6, 1, 1, 100, 200, 4, 3, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (7, 2, 1, 0, 1, 12, 1.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (8, 2, 1, 1, 3, 12, 1.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (9, 2, 1, 3, 10, 10, 1.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (10, 2, 1, 10, 20, 10, 1.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (11, 2, 1, 20, 100, 6, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (12, 2, 1, 100, 200, 6, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (13, 3, 1, 0, 1, 18, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (14, 3, 1, 1, 3, 18, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (15, 3, 1, 3, 10, 16, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (16, 3, 1, 10, 20, 16, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (17, 3, 1, 20, 100, 9, 2.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (18, 3, 1, 100, 200, 9, 2.5, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (19, 4, 1, 0, 1, 8, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (20, 4, 1, 1, 3, 8, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (21, 4, 1, 3, 10, 7, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (22, 4, 1, 10, 20, 7, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (23, 4, 1, 20, 100, 4, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (24, 4, 1, 100, 200, 4, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (25, 5, 1, 0, 1, 8, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (26, 5, 1, 1, 3, 8, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (27, 5, 1, 3, 10, 7, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (28, 5, 1, 10, 20, 7, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (29, 5, 1, 20, 100, 4, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (30, 5, 1, 100, 200, 4, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (31, 6, 1, 0, 1, 12, 1, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (32, 6, 1, 1, 3, 12, 1, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (33, 6, 1, 3, 10, 10, 1, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (34, 6, 1, 10, 20, 10, 1, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (35, 6, 1, 20, 100, 6, 1, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (36, 6, 1, 100, 200, 6, 1, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (37, 7, 1, 0, 1, 18, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (38, 7, 1, 1, 3, 18, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (39, 7, 1, 3, 10, 16, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (40, 7, 1, 10, 20, 16, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (41, 7, 1, 20, 100, 9, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (42, 7, 1, 100, 200, 9, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (43, 8, 1, 0, 1, 18, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (44, 8, 1, 1, 3, 18, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (45, 8, 1, 3, 10, 16, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (46, 8, 1, 10, 20, 16, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (47, 8, 1, 20, 100, 9, 2, 1, 1)

INSERT [dbo].[iaaspriceestimatelookup\_v2] ([ID], [IaaSType], [IaaS\_UpToMaxEnvironmentCount], [IaaS\_MinServerCount], [IaaS\_UpToMaxServerCount], [IaaS\_DeploymentHours], [IaaS\_SustainmentHours], [IaaS\_DeployComplexityMultiplier], [IaaS\_SustainComplexityMultiplier]) VALUES (48, 8, 1, 100, 200, 9, 2, 1, 1)

SET IDENTITY\_INSERT [dbo].[iaaspriceestimatelookup\_v2] OFF

GO

SET IDENTITY\_INSERT [dbo].[iaastypenames] ON

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (1, N'IaaS Server Deployment', 1)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (2, N'VM Image Lift and Shift', 1)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (3, N'Greenfield Migration', 1)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (4, N'Rehosting', 2)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (5, N'Re-platforming', 2)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (6, N'Repurchasing', 2)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (7, N'Refactoring', 2)

INSERT [dbo].[iaastypenames] ([ID], [IaasTypeName], [VlmfVersion]) VALUES (8, N'Greenfield', 2)

SET IDENTITY\_INSERT [dbo].[iaastypenames] OFF

GO

SET IDENTITY\_INSERT [dbo].[paaspriceestimatelookup] ON

INSERT [dbo].[paaspriceestimatelookup] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_OneTimeFee], [PaaS\_SustainmentCostPerYear]) VALUES (1, 1, 0, 3838.0000, 16748.5000)

INSERT [dbo].[paaspriceestimatelookup] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_OneTimeFee], [PaaS\_SustainmentCostPerYear]) VALUES (2, 3, 0, 18422.0000, 75368.0000)

INSERT [dbo].[paaspriceestimatelookup] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_OneTimeFee], [PaaS\_SustainmentCostPerYear]) VALUES (3, 10, 1, 92112.0000, 334970.0000)

INSERT [dbo].[paaspriceestimatelookup] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_OneTimeFee], [PaaS\_SustainmentCostPerYear]) VALUES (4, 20, 1, 245632.0000, 1004910.0000)

SET IDENTITY\_INSERT [dbo].[paaspriceestimatelookup] OFF

GO

SET IDENTITY\_INSERT [dbo].[paaspriceestimatelookup\_v2] ON

INSERT [dbo].[paaspriceestimatelookup\_v2] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_DeploymentHours], [PaaS\_SustainmentHours], [PaaS\_DeployComplexityMultiplier], [PaaS\_SustainComplexityMultiplier]) VALUES (1, 1, 0, 5, 3, 1, 1)

INSERT [dbo].[paaspriceestimatelookup\_v2] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_DeploymentHours], [PaaS\_SustainmentHours], [PaaS\_DeployComplexityMultiplier], [PaaS\_SustainComplexityMultiplier]) VALUES (2, 3, 0, 4.825, 2.9, 1.1, 1.033333)

INSERT [dbo].[paaspriceestimatelookup\_v2] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_DeploymentHours], [PaaS\_SustainmentHours], [PaaS\_DeployComplexityMultiplier], [PaaS\_SustainComplexityMultiplier]) VALUES (3, 10, 1, 4.6125, 2.8, 1.2, 1.066667)

INSERT [dbo].[paaspriceestimatelookup\_v2] ([ID], [PaaS\_UpToMaxDatabaseCount], [PaaS\_HighAvailability], [PaaS\_DeploymentHours], [PaaS\_SustainmentHours], [PaaS\_DeployComplexityMultiplier], [PaaS\_SustainComplexityMultiplier]) VALUES (4, 20, 1, 4.5, 2.7, 1.3, 1.1)

SET IDENTITY\_INSERT [dbo].[paaspriceestimatelookup\_v2] OFF

GO

SET IDENTITY\_INSERT [dbo].[pricingfactors\_v2] ON

INSERT [dbo].[pricingfactors\_v2] ([ID], [DeploymentMarginOfError], [SustainmentMarginOfError], [DeploymentBlendedHourlyCost], [SustainmentBlendedHourlyCost], [DeploymentUpFrontCost], [SustainmentUpFrontCost], [VlmfVersion]) VALUES (1, 3, 1.5, 150.0000, 150.0000, 3000.0000, 3000.0000, 1)

INSERT [dbo].[pricingfactors\_v2] ([ID], [DeploymentMarginOfError], [SustainmentMarginOfError], [DeploymentBlendedHourlyCost], [SustainmentBlendedHourlyCost], [DeploymentUpFrontCost], [SustainmentUpFrontCost], [VlmfVersion]) VALUES (2, 1.5, 1.5, 180.0000, 150.0000, 3000.0000, 3000.0000, 2)

SET IDENTITY\_INSERT [dbo].[pricingfactors\_v2] OFF

GO

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'7eaac442-3f52-46d7-9987-1fdee3798f79', N'New ROM Estimate', CAST(N'2021-05-04T14:20:05.587' AS DateTime), CAST(N'2021-05-04T14:20:16.370' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 0, N'', N'sand', 1, CAST(N'2021-05-04T14:20:16.370' AS DateTime), 8000.0000, 0.0000, 1, 3, NULL, NULL, 8, NULL, 1, NULL, 180.0000, 1.5, 8000.0000, 5, NULL, 1, NULL, N'', 1.5, 3000.0000, 150.0000, 3000.0000, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'76101459-573b-4a8a-bae1-24f777eeb98d', N'New ROM Estimate', CAST(N'2021-04-20T12:10:57.027' AS DateTime), CAST(N'2021-04-20T12:28:02.520' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36 Edg/89.0.774.48', 1, 4, 2, 0, 1, N'sand', N'', 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, N'', NULL, NULL, NULL, NULL, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'cf3964f2-65ee-4ccb-9398-276a0aba7639', N'New ROM Estimate', CAST(N'2021-05-04T14:32:58.903' AS DateTime), CAST(N'2021-05-04T14:33:09.020' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 0, N'', N'sand', 1, CAST(N'2021-05-04T14:33:09.020' AS DateTime), 8000.0000, 0.0000, 1, 3, NULL, NULL, 8, NULL, 1, NULL, 180.0000, 1.5, 8000.0000, 5, NULL, 1, NULL, N'', 1.5, 3000.0000, 150.0000, 3000.0000, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'23d131b4-3ddf-434b-9764-2d7611799257', N'New ROM Estimate', CAST(N'2021-03-29T07:49:52.447' AS DateTime), CAST(N'2021-03-29T07:50:16.807' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36 Edg/89.0.774.48', 1, 4, 2, 0, 1, N'', N'sand', 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, N'', NULL, NULL, NULL, NULL, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'3a211ef5-baf6-4bac-a8b8-43578d6bd83b', N'New ROM Estimate', CAST(N'2021-05-04T14:30:31.930' AS DateTime), CAST(N'2021-05-04T14:30:41.450' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 0, N'', N'sand', 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, N'', NULL, NULL, NULL, NULL, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'171f29c5-95c5-49d8-86ec-4e8c30157ef6', N'New ROM Estimate', CAST(N'2021-03-29T09:30:43.080' AS DateTime), CAST(N'2021-03-29T09:46:48.327' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36 Edg/89.0.774.48', 1, 7, 2, 0, 1, N'', N'sand', 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, N'', NULL, NULL, NULL, NULL, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'108325d6-9f4e-4750-aa13-4ffceada7ac1', N'New ROM Estimate', CAST(N'2021-05-04T14:14:10.210' AS DateTime), CAST(N'2021-05-04T14:14:22.340' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 0, N'', N'sand', 1, CAST(N'2021-05-04T14:14:22.340' AS DateTime), 8000.0000, 0.0000, 1, 3, NULL, NULL, 8, NULL, 1, NULL, 180.0000, 1.5, 8000.0000, 5, NULL, 1, NULL, N'', 1.5, 3000.0000, 150.0000, 3000.0000, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'9e7392f2-5b31-443b-ae6b-6079915083ae', N'New ROM Estimate', CAST(N'2021-05-04T14:21:56.357' AS DateTime), CAST(N'2021-05-04T14:22:04.317' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 0, N'', N'sand', 0, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, N'', NULL, NULL, NULL, NULL, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'93aee4f7-46dd-4ab9-b1e5-b86932b13b96', N'New ROM Estimate', CAST(N'2021-05-04T14:29:38.293' AS DateTime), CAST(N'2021-05-04T14:29:47.237' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 0, N'', N'sand', 1, CAST(N'2021-05-04T14:29:47.237' AS DateTime), 8000.0000, 0.0000, 1, 3, NULL, NULL, 8, NULL, 1, NULL, 180.0000, 1.5, 8000.0000, 5, NULL, 1, NULL, N'', 1.5, 3000.0000, 150.0000, 3000.0000, 2)

INSERT [dbo].[romestimate\_v2] ([ID], [ProjectName], [DateCreated], [LastModified], [LastModifiedBy], [IaaSNeeded], [IaaSType], [IaaSEnvironmentCount], [PaaSNeeded], [SustainmentNeeded], [VIPRID], [VASI], [ReadOnly], [DateFinalized], [QuotedOneTimePrice], [QuotedSustainmentCost], [LookupIaaSUpToMaxEnvironments], [LookupIaaSUpToMaxServerCount], [LookupPaaSUptoMaxDatabases], [LookupPaaSHighAvailability], [LookupIaaSDeployHours], [LookupPaaSDeployHours], [LookupIaaSDeployComplexityMultiplier], [LookupPaaSDeployComplexityMultiplier], [LookupDeploymentBlendedRate], [LookupDeploymentMarginOfError], [QuotedFirstYearCost], [LookupIaaSSustainHours], [LookupPaaSSustainHours], [LookupIaaSSustainComplexityMultiplier], [LookupPaaSSustainComplexityMultiplier], [Notes], [LookupSustainmentMarginOfError], [LookupDeploymentUpFrontCost], [LookupSustainmentBlendedRate], [LookupSustainmentUpFrontCost], [VlmfVersion]) VALUES (N'7c1aa223-0258-4921-a772-c10a0e6f262c', N'New ROM Estimate', CAST(N'2021-05-04T14:22:34.907' AS DateTime), CAST(N'2021-05-04T14:26:36.037' AS DateTime), N'::1/Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.114 Safari/537.36 Edg/89.0.774.76', 1, 4, 2, 0, 1, N'', N'sand', 1, CAST(N'2021-05-04T14:26:36.037' AS DateTime), 8000.0000, 17000.0000, 1, 3, NULL, NULL, 8, NULL, 1, NULL, 180.0000, 1.5, 25000.0000, 5, NULL, 1, NULL, N'', 1.5, 3000.0000, 150.0000, 3000.0000, 2)

GO

ALTER TABLE [dbo].[environments] ADD CONSTRAINT [DF\_environments\_EnvironmentIndex] DEFAULT ((0)) FOR [EnvironmentIndex]

GO

ALTER TABLE [dbo].[environments] ADD CONSTRAINT [DF\_environments\_ServerOrVMCount] DEFAULT ((0)) FOR [ServerOrVMCount]

GO

ALTER TABLE [dbo].[environments\_v2] ADD CONSTRAINT [DF\_environments\_v2\_EnvironmentIndex] DEFAULT ((0)) FOR [EnvironmentIndex]

GO

ALTER TABLE [dbo].[environments\_v2] ADD CONSTRAINT [DF\_environments\_v2\_ServerOrVMCount] DEFAULT ((0)) FOR [ServerOrVMCount]

GO

ALTER TABLE [dbo].[iaaspriceestimatelookup] ADD CONSTRAINT [DF\_IaaSPriceEstimateLookup\_IaaSType] DEFAULT ((1)) FOR [IaaSType]

GO

ALTER TABLE [dbo].[iaaspriceestimatelookup] ADD CONSTRAINT [DF\_Table\_1\_IaaSPriceUpToEnvironmentCount] DEFAULT ((0)) FOR [IaaS\_UpToMaxEnvironmentCount]

GO

ALTER TABLE [dbo].[iaaspriceestimatelookup] ADD CONSTRAINT [DF\_Table\_1\_IaaSPrice\_UpToServerCount] DEFAULT ((0)) FOR [IaaS\_UpToMaxServerCount]

GO

ALTER TABLE [dbo].[iaaspriceestimatelookup] ADD CONSTRAINT [DF\_Table\_1\_IaaS\_OneTimeFee\_] DEFAULT ((0)) FOR [IaaS\_OneTimeFee]

GO

ALTER TABLE [dbo].[iaaspriceestimatelookup] ADD CONSTRAINT [DF\_IaaSPriceEstimateLookup\_IaaS\_SustainmentCostPerYear] DEFAULT ((0)) FOR [IaaS\_SustainmentCostPerYear]

GO

ALTER TABLE [dbo].[iaastypenames] ADD CONSTRAINT [DF\_iaastypenames\_VlmfVersion] DEFAULT ((1)) FOR [VlmfVersion]

GO

ALTER TABLE [dbo].[paas] ADD CONSTRAINT [DF\_paas\_DatabaseInstanceCount] DEFAULT ((0)) FOR [DatabaseInstanceCount]

GO

ALTER TABLE [dbo].[paas] ADD CONSTRAINT [DF\_paas\_InstancesHighlyAvailable] DEFAULT ((0)) FOR [InstancesHighlyAvailable]

GO

ALTER TABLE [dbo].[paas\_v2] ADD CONSTRAINT [DF\_paas\_v2\_DatabaseInstanceCount] DEFAULT ((0)) FOR [DatabaseInstanceCount]

GO

ALTER TABLE [dbo].[paas\_v2] ADD CONSTRAINT [DF\_paas\_v2\_InstancesHighlyAvailable] DEFAULT ((0)) FOR [InstancesHighlyAvailable]

GO

ALTER TABLE [dbo].[paaspriceestimatelookup] ADD CONSTRAINT [DF\_Table\_1\_PaaSUpToMaxDatabaseCount] DEFAULT ((0)) FOR [PaaS\_UpToMaxDatabaseCount]

GO

ALTER TABLE [dbo].[paaspriceestimatelookup] ADD CONSTRAINT [DF\_Table\_1\_PaaSHighAvailability] DEFAULT ((0)) FOR [PaaS\_HighAvailability]

GO

ALTER TABLE [dbo].[paaspriceestimatelookup] ADD CONSTRAINT [DF\_paaspriceestimatelookup\_PaaS\_OneTimeFee] DEFAULT ((0)) FOR [PaaS\_OneTimeFee]

GO

ALTER TABLE [dbo].[paaspriceestimatelookup] ADD CONSTRAINT [DF\_paaspriceestimatelookup\_PaaS\_SustainmentCostPerYear] DEFAULT ((0)) FOR [PaaS\_SustainmentCostPerYear]

GO

ALTER TABLE [dbo].[paaspriceestimatelookup\_v2] ADD CONSTRAINT [DF\_Table\_1\_PaaSUpToMaxDatabaseCount\_v2] DEFAULT ((0)) FOR [PaaS\_UpToMaxDatabaseCount]

GO

ALTER TABLE [dbo].[paaspriceestimatelookup\_v2] ADD CONSTRAINT [DF\_Table\_1\_PaaSHighAvailability\_v2] DEFAULT ((0)) FOR [PaaS\_HighAvailability]

GO

ALTER TABLE [dbo].[romestimate] ADD CONSTRAINT [DF\_romestimate\_ID] DEFAULT (newid()) FOR [ID]

GO

ALTER TABLE [dbo].[romestimate] ADD CONSTRAINT [DF\_romestimate\_PaaSNeeded] DEFAULT ((0)) FOR [PaaSNeeded]

GO

ALTER TABLE [dbo].[romestimate] ADD CONSTRAINT [DF\_romestimate\_SustainmentNeeded] DEFAULT ((0)) FOR [SustainmentNeeded]

GO

ALTER TABLE [dbo].[romestimate] ADD CONSTRAINT [DF\_romestimate\_Finalized] DEFAULT ((0)) FOR [ReadOnly]

GO

ALTER TABLE [dbo].[romestimate] ADD CONSTRAINT [DF\_romestimate\_VlmfVersion] DEFAULT ((1)) FOR [VlmfVersion]

GO

ALTER TABLE [dbo].[romestimate\_v2] ADD CONSTRAINT [DF\_romestimate\_ID\_v2] DEFAULT (newid()) FOR [ID]

GO

ALTER TABLE [dbo].[romestimate\_v2] ADD CONSTRAINT [DF\_romestimate\_v2\_IaaSNeeded] DEFAULT ((0)) FOR [IaaSNeeded]

GO

ALTER TABLE [dbo].[romestimate\_v2] ADD CONSTRAINT [DF\_romestimate\_PaaSNeeded\_v2] DEFAULT ((0)) FOR [PaaSNeeded]

GO

ALTER TABLE [dbo].[romestimate\_v2] ADD CONSTRAINT [DF\_romestimate\_SustainmentNeeded\_v2] DEFAULT ((0)) FOR [SustainmentNeeded]

GO

ALTER TABLE [dbo].[romestimate\_v2] ADD CONSTRAINT [DF\_romestimate\_v2\_VlmfVersion] DEFAULT ((1)) FOR [VlmfVersion]

GO

ALTER TABLE [dbo].[environments] WITH CHECK ADD CONSTRAINT [FK\_environments\_romestimate] FOREIGN KEY([ROMID])

REFERENCES [dbo].[romestimate] ([ID])

GO

ALTER TABLE [dbo].[environments] CHECK CONSTRAINT [FK\_environments\_romestimate]

GO

ALTER TABLE [dbo].[environments\_v2] WITH CHECK ADD CONSTRAINT [FK\_environments\_v2\_romestimate\_v2] FOREIGN KEY([ROMID])

REFERENCES [dbo].[romestimate\_v2] ([ID])

GO

ALTER TABLE [dbo].[environments\_v2] CHECK CONSTRAINT [FK\_environments\_v2\_romestimate\_v2]

GO

ALTER TABLE [dbo].[paas] WITH CHECK ADD CONSTRAINT [FK\_paas\_romestimate] FOREIGN KEY([ROMID])

REFERENCES [dbo].[romestimate] ([ID])

GO

ALTER TABLE [dbo].[paas] CHECK CONSTRAINT [FK\_paas\_romestimate]

GO

ALTER TABLE [dbo].[paas\_v2] WITH CHECK ADD CONSTRAINT [FK\_paas\_v2\_romestimate\_v2] FOREIGN KEY([ROMID])

REFERENCES [dbo].[romestimate\_v2] ([ID])

GO

ALTER TABLE [dbo].[paas\_v2] CHECK CONSTRAINT [FK\_paas\_v2\_romestimate\_v2]

GO

ALTER TABLE [dbo].[romestimate] WITH CHECK ADD CONSTRAINT [FK\_romestimate\_iaastypenames] FOREIGN KEY([IaaSType])

REFERENCES [dbo].[iaastypenames] ([ID])

GO

ALTER TABLE [dbo].[romestimate] CHECK CONSTRAINT [FK\_romestimate\_iaastypenames]

GO

ALTER TABLE [dbo].[romestimate\_v2] WITH CHECK ADD CONSTRAINT [FK\_romestimate\_iaastypenames\_v2] FOREIGN KEY([IaaSType])

REFERENCES [dbo].[iaastypenames] ([ID])

GO

ALTER TABLE [dbo].[romestimate\_v2] CHECK CONSTRAINT [FK\_romestimate\_iaastypenames\_v2]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[CreateRom] Script Date: 5/19/2021 1:49:56 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/18/2019

-- Description: Creates a new ROM (which may only be partially filled-out). Returns a GUID for the new ROM that can be used

-- to gain future read/write access to the ROM

-- =============================================

CREATE PROCEDURE [dbo].[CreateRom]

@id uniqueidentifier OUTPUT,

@RomName varchar(MAX) = Unnamed,

@UserName varchar(MAX) = Unknown,

@IaaSNeeded bit = 0,

@IaaSType int = 0,

@IaaSEnvironmentCount int = 0,

@PaaSNeeded bit = 0,

@PaaSDBInstances int = 0,

@PaaSHighAvailability bit = 0,

@SustainmentNeeded bit = 0,

@Environment1 bit = 0,

@Environment1Name varchar(MAX) = null,

@Environment1ServerOrVMCount int = 0,

@Environment2 bit = 0,

@Environment2Name varchar(MAX) = null,

@Environment2ServerOrVMCount int = 0,

@Environment3 bit = 0,

@Environment3Name varchar(MAX) = null,

@Environment3ServerOrVMCount int = 0,

@Environment4 bit = 0,

@Environment4Name varchar(MAX) = null,

@Environment4ServerOrVMCount int = 0,

@Environment5 bit = 0,

@Environment5Name varchar(MAX) = null,

@Environment5ServerOrVMCount int = 0

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

-- SET NOCOUNT ON;

-- Inserts a new ROM, returns the GUID used for future access to the ROM. Also inserts into environments table, based

-- on the parameters passed in, sets the created date, last modified date, last modified by, etc.

set @id = NEWID()

INSERT INTO [dbo].[romestimate]

([ID]

,[ProjectName]

,[DateCreated]

,[LastModified]

,[LastModifiedBy]

,[IaaSNeeded]

,[IaaSType]

,[IaaSEnvironmentCount]

,[PaaSNeeded]

,[SustainmentNeeded]

,[ReadOnly]

,[DateFinalized]

,[QuotedOneTimePrice]

,[QuotedSustainmentCost]

,[LookupIaaSUpToMaxEnvironments]

,[LookupIaaSUpToMaxServerCount]

,[LookupIaaSOneTimeFee]

,[LookupIaaSSustainmentCost]

,[LookupPaaSUptoMaxDatabases]

,[LookupPaaSHighAvailability]

,[LookupPaaSOneTimeFee]

,[LookupPaaSSustainmentCost])

VALUES

(@id,

@RomName,

CURRENT\_TIMESTAMP,

CURRENT\_TIMESTAMP,

@UserName,

@IaaSNeeded,

@IaaSType,

@IaaSEnvironmentCount,

@PaaSNeeded,

@SustainmentNeeded,

0,

null,

null,

null,

null,

null,

null,

null,

null,

null,

null,

null)

-- now insert into the PaaS table, if appropriate

IF @PaaSNeeded = 1

BEGIN

INSERT INTO [dbo].[paas]

([ROMID]

,[DatabaseInstanceCount]

,[InstancesHighlyAvailable])

VALUES

(@id,

@PaaSDBInstances,

@PaaSHighAvailability)

END

-- now insert the environment table data, if appropriate

IF @IaaSEnvironmentCount > 0

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,1

,@Environment1Name

,@Environment1ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 1

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,2

,@Environment2Name

,@Environment2ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 2

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,3

,@Environment3Name

,@Environment3ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 3

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,4

,@Environment4Name

,@Environment4ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 4

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,5

,@Environment5Name

,@Environment5ServerOrVMCount)

END

-- Return the GUID for the ROM (as an output value)

SELECT @id

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[CreateRom\_v2] Script Date: 5/19/2021 1:49:57 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/18/2019

-- Updated: 3/26/2021

-- Description: Creates a new ROM (which may only be partially filled-out). Returns a GUID for the new ROM that can be used

-- to gain future read/write access to the ROM

-- =============================================

CREATE PROCEDURE [dbo].[CreateRom\_v2]

@id uniqueidentifier OUTPUT,

@RomName varchar(MAX) = Unnamed,

@UserName varchar(MAX) = Unknown,

@IaaSNeeded bit = 0,

@IaaSType int = 0,

@IaaSEnvironmentCount int = 0,

@PaaSNeeded bit = 0,

@PaaSDBInstances int = 0,

@PaaSHighAvailability bit = 0,

@SustainmentNeeded bit = 0,

@Environment1 bit = 0,

@Environment1Name varchar(MAX) = null,

@Environment1ServerOrVMCount int = 0,

@Environment2 bit = 0,

@Environment2Name varchar(MAX) = null,

@Environment2ServerOrVMCount int = 0,

@Environment3 bit = 0,

@Environment3Name varchar(MAX) = null,

@Environment3ServerOrVMCount int = 0,

@Environment4 bit = 0,

@Environment4Name varchar(MAX) = null,

@Environment4ServerOrVMCount int = 0,

@Environment5 bit = 0,

@Environment5Name varchar(MAX) = null,

@Environment5ServerOrVMCount int = 0,

@VlmfVersion int

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

-- SET NOCOUNT ON;

-- Inserts a new ROM, returns the GUID used for future access to the ROM. Also inserts into environments table, based

-- on the parameters passed in, sets the created date, last modified date, last modified by, etc.

set @id = NEWID()

INSERT INTO [dbo].[romestimate\_v2]

([ID]

,[ProjectName]

,[DateCreated]

,[LastModified]

,[LastModifiedBy]

,[IaaSNeeded]

,[IaaSType]

,[IaaSEnvironmentCount]

,[PaaSNeeded]

,[SustainmentNeeded]

,[ReadOnly]

,[DateFinalized]

,[QuotedOneTimePrice]

,[QuotedSustainmentCost]

,[LookupIaaSUpToMaxEnvironments]

,[LookupIaaSUpToMaxServerCount]

,[LookupPaaSUptoMaxDatabases]

,[LookupPaaSHighAvailability]

,[VlmfVersion])

VALUES

(@id,

@RomName,

CURRENT\_TIMESTAMP,

CURRENT\_TIMESTAMP,

@UserName,

@IaaSNeeded,

@IaaSType,

@IaaSEnvironmentCount,

@PaaSNeeded,

@SustainmentNeeded,

0,

null,

null,

null,

null,

null,

null,

null,

@VlmfVersion)

-- now insert into the PaaS table, if appropriate

IF @PaaSNeeded = 1

BEGIN

INSERT INTO [dbo].[paas\_v2]

([ROMID]

,[DatabaseInstanceCount]

,[InstancesHighlyAvailable])

VALUES

(@id,

@PaaSDBInstances,

@PaaSHighAvailability)

END

-- now insert the environment table data, if appropriate

IF @IaaSEnvironmentCount > 0

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,1

,@Environment1Name

,@Environment1ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 1

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,2

,@Environment2Name

,@Environment2ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 2

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,3

,@Environment3Name

,@Environment3ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 3

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,4

,@Environment4Name

,@Environment4ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 4

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@id

,5

,@Environment5Name

,@Environment5ServerOrVMCount)

END

-- Return the GUID for the ROM (as an output value)

SELECT @id

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[DeleteRom] Script Date: 5/19/2021 1:49:57 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/19/2019

-- Description: Deletes an existing ROM by GUID.

-- =============================================

CREATE PROCEDURE [dbo].[DeleteRom]

-- Add the parameters for the stored procedure here

@RomId uniqueidentifier

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- TODO - need to read PaaS and read and flatten Environment info associated with the ROM

DELETE FROM [dbo].[environments] WHERE ROMID=@RomId

DELETE FROM [dbo].[paas] WHERE ROMID=@RomId

DELETE FROM [dbo].[environments\_v2] WHERE ROMID=@RomId

DELETE FROM [dbo].[paas\_v2] WHERE ROMID=@RomId

DELETE FROM [dbo].[romestimate] WHERE ID=@RomId

DELETE FROM [dbo].[romestimate\_v2] WHERE ID=@RomId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[FinalizeEstimate] Script Date: 5/19/2021 1:49:57 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/24/2019

-- Description: Finalizes a rough order of magnitude (ROM) estimate and marks it as read-only. Also stamps the ROM

-- with the frozen-in-time estimate values that were used to calculate the estimate.

-- Note that the ROM values must be SAVED/Updated before calling this. i.e. the only parameter is a ROM id.

-- =============================================

CREATE PROCEDURE [dbo].[FinalizeEstimate]

@RomId uniqueidentifier,

@UserIdentifier varchar(MAX)

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- NOTE: we must go through a lot of the same logic to retrieve the one-time and sustainment

-- costs, since we want to store metadata about the rows used to perform the calculations

-- (i.e. the values we used to calculate these costs)

-- Retrieve a rough order of magnitude (ROM) one-time and sustainment cost/year estimate (based on the values in cost

-- estimate lookup tables). These will be stored in the "Quoted" one-time and sustainment cost price columns

DECLARE @EstimatedOneTimeCost money = 0

DECLARE @EstimatedSustainmentPerYear money = 0

-- First, retrieve the ROM

-- Declare local variables to store the information required to produce an estimate

DECLARE @IaaSNeeded bit

DECLARE @IaaSType int

DECLARE @PaaSNeeded bit

DECLARE @SustainmentNeeded bit

DECLARE @IaaSEnvironmentCount int

DECLARE @IaaSTotalServerCount int -- Total server count across all environments (since # of environments no longer plays any factor)

DECLARE @PaaSDatabaseCount int

DECLARE @PaaSHighAvailability bit

-- Declare local variables used to store metadata about the price estimate we provided

DECLARE @LookupIaaSUpToMaxEnvironments as int

DECLARE @LookupIaaSUpToMaxServerCount as int

DECLARE @LookupIaaSUpToMaxServerCountPerEnvironment as int

DECLARE @LookupIaaSOneTimeFee as money

DECLARE @LookupIaaSSustainmentCost as money

DECLARE @LookupPaaSUptoMaxDatabases as int

DECLARE @LookupPaaSHighAvailability as bit

DECLARE @LookupPaaSOneTimeFee as money

DECLARE @LookupPaaSSustainmentCost as money

DECLARE @PaaSCostPerServerOneTimeFee as money = 0

DECLARE @PaaSCostEstimateOneTimeFee as money = 0

DECLARE @PaaSCostPerServerSustainment as money = 0

DECLARE @PaaSCostEstimateSustainment as money = 0

DECLARE @CostPerServerOneTimeFee as money = 0

DECLARE @CostEstimateOneTimeFee as money = 0

DECLARE @CostPerServerSustainment as money = 0

DECLARE @CostEstimateSustainment as money = 0

-- Set variable values based on reading the ROM

SELECT @IaaSNeeded = IaaSNeeded

,@IaaSType = IaaSType

,@PaaSNeeded = PaaSNeeded

,@SustainmentNeeded = SustainmentNeeded

,@IaaSEnvironmentCount = IaaSEnvironmentCount

FROM [dbo].[romestimate] WHERE ID=@RomId

-- Now read the environments table to get the max number of servers per environment, if applicable

IF @IaaSNeeded = 1

BEGIN

SELECT @IaaSTotalServerCount = Sum(ServerOrVMCount) FROM [dbo].[environments]

WHERE ROMID = @RomId

END

-- Now collect info about any PaaS needed, including DB count and if they need to be Highly Available

IF @PaaSNeeded = 1

BEGIN

SELECT @PaaSDatabaseCount = DatabaseInstanceCount

,@PaaSHighAvailability = InstancesHighlyAvailable

FROM [dbo].[paas]

WHERE ROMID = @RomId

END

IF @IaaSNeeded = 1

BEGIN

DECLARE @OneTimeFee as money = 0

DECLARE @SustainmentCost as money = 0

SELECT TOP 1 @OneTimeFee = IaaS\_OneTimeFee

,@SustainmentCost = IaaS\_SustainmentCostPerYear

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

,@LookupIaaSOneTimeFee = IaaS\_OneTimeFee

,@LookupIaaSSustainmentCost = IaaS\_SustainmentCostPerYear

FROM [dbo].[iaaspriceestimatelookup]

WHERE IaaSType = @IaaSType AND

-- REQUEST BY JEN: Do not take into account environment count when

-- providing ROMs. This doesn't seem intuitively right, but it's the marching order

-- for now. @IaaSEnvironmentCount <= IaaS\_UpToMaxEnvironmentCount AND

IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount >= @IaaSTotalServerCount

ORDER BY

-- IaaS\_UpToMaxEnvironmentCount,

IaaS\_UpToMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

-- special case for catching the first "bucket"

if @OneTimeFee = 0 BEGIN

-- TODO: reconsider hardcoding these values

IF @IaaSTotalServerCount < 11 BEGIN

SELECT TOP 1 @OneTimeFee = IaaS\_OneTimeFee

,@SustainmentCost = IaaS\_SustainmentCostPerYear

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

,@LookupIaaSOneTimeFee = IaaS\_OneTimeFee

,@LookupIaaSSustainmentCost = IaaS\_SustainmentCostPerYear

FROM [dbo].[iaaspriceestimatelookup]

WHERE IaaSType = @IaaSType

ORDER BY IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

END

IF @IaaSTotalServerCount >= 11 BEGIN

SELECT TOP 1 @OneTimeFee = IaaS\_OneTimeFee

,@SustainmentCost = IaaS\_SustainmentCostPerYear

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

,@LookupIaaSOneTimeFee = IaaS\_OneTimeFee

,@LookupIaaSSustainmentCost = IaaS\_SustainmentCostPerYear

FROM [dbo].[iaaspriceestimatelookup]

WHERE IaaSType = @IaaSType

ORDER BY IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount DESC

END

END

-- this is the pricing "bucket" we'll use to calculate the per-server cost

SET @CostPerServerOneTimeFee = @OneTimeFee / @LookupIaaSUpToMaxServerCount

SET @CostEstimateOneTimeFee = @CostPerServerOneTimeFee \* @IaaSTotalServerCount

SET @CostPerServerSustainment = @SustainmentCost / @LookupIaaSSustainmentCost

SET @CostEstimateSustainment = @CostPerServerSustainment \* @IaaSTotalServerCount

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @OneTimeFee + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @SustainmentCost + 999) / 1000) \* 1000

END

END

IF @PaaSNeeded = 1

BEGIN

DECLARE @OneTimeFeePaaS as money = 0

DECLARE @SustainmentCostPaaS as money = 0

SELECT TOP 1 @OneTimeFeePaaS = PaaS\_OneTimeFee

,@SustainmentCostPaaS = PaaS\_SustainmentCostPerYear

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSHighAvailability = PaaS\_HighAvailability

,@LookupPaaSOneTimeFee = PaaS\_OneTimeFee

,@LookupPaaSSustainmentCost = PaaS\_SustainmentCostPerYear

FROM [dbo].[paaspriceestimatelookup]

WHERE PaaS\_UpToMaxDatabaseCount >= @PaaSDatabaseCount

ORDER BY PaaS\_UpToMaxDatabaseCount

-- special case for catching the first "bucket"

if @OneTimeFeePaaS = 0 BEGIN

-- TODO: reconsider hardcoding these values

IF @PaaSDatabaseCount < 11 BEGIN

SELECT TOP 1 @OneTimeFeePaaS = PaaS\_OneTimeFee

,@SustainmentCostPaaS = PaaS\_SustainmentCostPerYear

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSOneTimeFee = PaaS\_OneTimeFee

,@LookupPaaSSustainmentCost = PaaS\_SustainmentCostPerYear

FROM [dbo].[paaspriceestimatelookup]

ORDER BY PaaS\_UpToMaxDatabaseCount

END

IF @PaaSDatabaseCount >= 11 BEGIN

SELECT TOP 1 @OneTimeFeePaaS = PaaS\_OneTimeFee

,@SustainmentCostPaaS = PaaS\_SustainmentCostPerYear

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSOneTimeFee = PaaS\_OneTimeFee

,@LookupPaaSSustainmentCost = PaaS\_SustainmentCostPerYear

FROM [dbo].[paaspriceestimatelookup]

ORDER BY PaaS\_UpToMaxDatabaseCount DESC

END

END

-- this is the pricing "bucket" we'll use to calculate the per-server cost

SET @PaaSCostPerServerOneTimeFee = @OneTimeFeePaaS / @LookupPaaSUptoMaxDatabases

SET @PaaSCostEstimateOneTimeFee = @PaaSCostPerServerOneTimeFee \* @PaaSDatabaseCount

SET @PaaSCostPerServerSustainment = @SustainmentCostPaaS / @LookupPaaSUptoMaxDatabases

SET @PaaSCostEstimateSustainment = @PaaSCostPerServerSustainment \* @PaaSDatabaseCount

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @OneTimeFeePaaS + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @SustainmentCostPaaS + 999) / 1000) \* 1000

END

END

DECLARE @QuotedFirstYearCost as money = @EstimatedOneTimeCost + @EstimatedSustainmentPerYear

UPDATE romestimate

SET [romestimate].[ReadOnly] = 1

,LastModified = CURRENT\_TIMESTAMP

,LastModifiedBy = @UserIdentifier

,DateFinalized = CURRENT\_TIMESTAMP

,QuotedOneTimePrice = @EstimatedOneTimeCost

,QuotedSustainmentCost = @EstimatedSustainmentPerYear

,QuotedFirstYearCost = @QuotedFirstYearCost

,LookupIaaSUpToMaxEnvironments = @LookupIaaSUpToMaxEnvironments

,LookupIaaSUpToMaxServerCount = @LookupIaaSUpToMaxServerCount

,LookupIaaSOneTimeFee = @LookupIaaSOneTimeFee

,LookupIaaSSustainmentCost = @LookupIaaSSustainmentCost

,LookupPaaSUptoMaxDatabases = @LookupPaaSUptoMaxDatabases

,LookupPaaSHighAvailability = @LookupPaaSHighAvailability

,LookupPaaSOneTimeFee = @LookupPaaSOneTimeFee

,LookupPaaSSustainmentCost = @LookupPaaSSustainmentCost

WHERE ID = @RomId

-- Return the GUID for the ROM along with estimated costs (one time and sustainment per year)

SELECT @RomId as RomId,

@EstimatedOneTimeCost as OneTimeCost,

@EstimatedSustainmentPerYear as SustainmentPerYear,

@IaaSTotalServerCount AS TotalServerCountRequested,

@PaaSCostPerServerOneTimeFee as PaaSOneTimeFeePerServer,

@PaaSCostEstimateOneTimeFee as PaaSOneTimeFee,

@PaaSCostPerServerSustainment as PaaSSustainmentPerServer,

@PaaSCostEstimateSustainment as PaaSSustainment,

@CostPerServerOneTimeFee as IaaSCostPerServerOneTimeFee,

@CostEstimateOneTimeFee as CostEstimateOneTimeFee,

@CostPerServerSustainment as CostPerServerSustainment,

@CostEstimateSustainment as CostEstimateSustainment,

@QuotedFirstYearCost as QuotedFirstYearCost

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[FinalizeEstimate\_v2] Script Date: 5/19/2021 1:49:58 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/24/2019

-- Description: Finalizes a rough order of magnitude (ROM) estimate and marks it as read-only. Also stamps the ROM

-- with the frozen-in-time estimate values that were used to calculate the estimate.

-- Note that the ROM values must be SAVED/Updated before calling this. i.e. the only parameter is a ROM id.

-- =============================================

CREATE PROCEDURE [dbo].[FinalizeEstimate\_v2]

@RomId uniqueidentifier,

@UserIdentifier varchar(MAX)

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Retrieves a rough order of magnitude (ROM) one-time and sustainment cost/year estimate (based on the values in cost

-- estimate lookup tables)

-- The basic logic for IaaS is to select the row in iaaspriceestimatelookup where the

-- number of environments needed AND max number of servers needed in any environment is

-- Less Than Or Equal To the corresponding lookup row in the database.

-- For PaaS we look if high availability is needed to narrow down our query. Then we select

-- the row in paaspriceestimatelookup where the number of databases needed is

-- Less Than Or Equal To the values in that row

-- First, we must retrieve the ROM

-- First, retrieve the ROM

-- Declare local variables to store the information required to produce an estimate

DECLARE @IaaSNeeded bit

DECLARE @IaaSType int

DECLARE @PaaSNeeded bit

DECLARE @SustainmentNeeded bit

DECLARE @EstimatedOneTimeCost money = 0

DECLARE @EstimatedSustainmentPerYear money = 0

DECLARE @IaaSEnvironmentCount int

DECLARE @IaaSTotalServerCount int -- Total server count across all environments (since # of environments no longer plays any factor)

DECLARE @PaaSDatabaseCount int

DECLARE @PaaSHighAvailability bit

-- metadata about pricing lookup

DECLARE @LookupIaaSUpToMaxEnvironments as int

DECLARE @LookupIaaSUpToMaxServerCount as int

DECLARE @LookupIaaSUpToMaxServerCountPerEnvironment as int

DECLARE @LookupPaaSUptoMaxDatabases as int

DECLARE @LookupPaaSHighAvailability as bit

DECLARE @PaaSCostPerServerOneTimeFee as money = 0

DECLARE @PaaSCostEstimateOneTimeFee as money = 0

DECLARE @PaaSCostPerServerSustainment as money = 0

DECLARE @PaaSCostEstimateSustainment as money = 0

DECLARE @IaaSDeploymentCost as money = 0

DECLARE @IaaSDeploymentCostEstimate as money = 0

DECLARE @CostEstimateSustainment as money = 0

DECLARE @DeployMarginOfError as float

DECLARE @SustainMarginOfError as float

DECLARE @DeploymentBlendedRate as money

DECLARE @SustainBlendedRate as money

DECLARE @DeploymentUpFrontCost as money

DECLARE @SustainmentUpFrontCost as money

DECLARE @CostForSustainmentHours as money = 0

DECLARE @CostPerServerSustainment as money = 0

DECLARE @DeployHoursPerUnit as float = 0 -- this may take into account 1 or more "bucket"

DECLARE @SustainHoursPerUnit as float = 0 -- this may take into account 1 or more "bucket"

DECLARE @VlmfVersion as Int = 2

-- Set variable values based on reading the ROM

SELECT @IaaSNeeded = IaaSNeeded

,@IaaSType = IaaSType

,@PaaSNeeded = PaaSNeeded

,@SustainmentNeeded = SustainmentNeeded

,@IaaSEnvironmentCount = IaaSEnvironmentCount

,@VlmfVersion = VlmfVersion

FROM [dbo].[romestimate\_v2] WHERE ID=@RomId

SELECT @DeployMarginOfError = DeploymentMarginOfError

,@SustainMarginOfError = SustainmentMarginOfError

,@DeploymentBlendedRate = DeploymentBlendedHourlyCost

,@SustainBlendedRate = SustainmentBlendedHourlyCost

,@DeploymentUpFrontCost = DeploymentUpFrontCost

,@SustainmentUpFrontCost = SustainmentUpFrontCost

FROM [dbo].[pricingfactors\_v2]

WHERE [dbo].[pricingfactors\_v2].VlmfVersion = @VlmfVersion

-- Now read the environments table to get the max number of servers per environment, if applicable

IF @IaaSNeeded = 1

BEGIN

SELECT @IaaSTotalServerCount = Sum(ServerOrVMCount) FROM [dbo].[environments\_v2]

WHERE ROMID = @RomId

END

-- Now collect info about any PaaS needed, including DB count and if they need to be Highly Available

IF @PaaSNeeded = 1

BEGIN

SELECT @PaaSDatabaseCount = DatabaseInstanceCount

,@PaaSHighAvailability = InstancesHighlyAvailable

FROM [dbo].[paas\_v2]

WHERE ROMID = @RomId

END

IF @IaaSNeeded = 1

BEGIN

-- We need to collect up to two rows - the actual hours row for server count requested, and the prior row

DECLARE @MaxInModel as Integer = 0

DECLARE @IDTopRow as Integer = 0

DECLARE @DeployHoursPerUnitMoreExpensiveBucket as float = 0

DECLARE @SustainHoursPerUnitMoreExpensiveBucket as float = 0

DECLARE @LookupIaaSUpToMaxServerCountMoreExpensiveBucket as float = 0

DECLARE @UnitsAtLowerPrice as float = 0

DECLARE @UnitsAtHigherPrice as float = 0

DECLARE @MinUnitsCheaperBucket as float = 0

DECLARE @MaxUnitsCheaperBucket as float = 0

-- SPECIAL CASE - only asking for 1 unit

IF @IaaSTotalServerCount = 1

BEGIN

SELECT TOP 1 @IDTopRow = ID

,@DeployHoursPerUnit = IaaS\_DeploymentHours

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = 1

,@SustainHoursPerUnit = IaaS\_SustainmentHours

FROM [dbo].[iaaspriceestimatelookup\_v2]

WHERE IaaSType = @IaaSType AND @IaaSTotalServerCount = IaaS\_UpToMaxServerCount

ORDER BY IaaS\_UpToMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

SET @IaaSDeploymentCost = @DeployHoursPerUnit \* @DeploymentBlendedRate \* @DeployMarginOfError \* @IaaSTotalServerCount

SET @IaaSDeploymentCostEstimate = @IaaSDeploymentCost + @DeploymentUpFrontCost

SET @CostForSustainmentHours = @SustainHoursPerUnit \* @SustainBlendedRate \* @SustainMarginOfError \* 12

SET @CostEstimateSustainment = (@CostForSustainmentHours \* @IaaSTotalServerCount) + @SustainmentUpFrontCost

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @IaaSDeploymentCostEstimate + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @CostEstimateSustainment + 999) / 1000) \* 1000

END

END

IF @IaaSTotalServerCount > 1

BEGIN

DECLARE @LookupIaaSMinCountMoreExpensiveBucket as float = 0

-- Grab the max number of IaaS VMs in our model for yet another corner case where IaaS VMs requested exceeds the max in our model (currently 200)

SELECT TOP 1 @MaxInModel = IaaS\_UpToMaxServerCount

from [dbo].[iaaspriceestimatelookup\_v2]

WHERE @IaaSType = IaaSType

ORDER BY [dbo].[iaaspriceestimatelookup\_v2].IaaS\_UpToMaxServerCount DESC

-- Determine the "Bucket" we're in (VMs requested >= Minimum VMs associated with the bucket), order descending so we pick the highest matching bucket

SELECT TOP 1 @IDTopRow = ID

,@DeployHoursPerUnit = IaaS\_DeploymentHours

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount

,@SustainHoursPerUnit = IaaS\_SustainmentHours

,@MinUnitsCheaperBucket = IaaS\_MinServerCount

,@MaxUnitsCheaperBucket = IaaS\_UpToMaxServerCount

FROM [dbo].[iaaspriceestimatelookup\_v2]

WHERE IaaSType = @IaaSType AND @IaaSTotalServerCount > IaaS\_MinServerCount

ORDER BY IaaS\_MinServerCount DESC

-- Handle the corner case where the customer requests >= (currently) 200 IaaS servers

IF @IaaSTotalServerCount >= @MaxInModel

BEGIN

SET @DeployHoursPerUnitMoreExpensiveBucket = @DeployHoursPerUnit

SET @SustainHoursPerUnitMoreExpensiveBucket = @SustainHoursPerUnit

SET @UnitsAtLowerPrice = @IaaSTotalServerCount

SET @UnitsAtHigherPrice = 0

SET @LookupIaaSUpToMaxServerCountMoreExpensiveBucket = @MaxInModel

END

ELSE -- Handle the case where the customer requests a number of IaaS servers that fits into a "bucket"

BEGIN

-- retrieve the "previous" bucket, since we will price all machines up to the bucket's max VM count @IaaSTotalServerCount using that prior bucket

SELECT TOP 1 @DeployHoursPerUnitMoreExpensiveBucket = IaaS\_DeploymentHours

,@SustainHoursPerUnitMoreExpensiveBucket = IaaS\_SustainmentHours

,@LookupIaaSUpToMaxServerCountMoreExpensiveBucket = IaaS\_UptoMaxServerCount

,@LookupIaaSMinCountMoreExpensiveBucket = IaaS\_MinServerCount

FROM [dbo].[iaaspriceestimatelookup\_v2]

WHERE IaaSType = @IaaSType AND (ID = @IDTopRow - 1) -- this is safe because we took care of the special case where 1 VM is requested

END

SET @UnitsAtLowerPrice = @IaaSTotalServerCount - @LookupIaaSUpToMaxServerCountMoreExpensiveBucket -- remaining VMs above and beyond the maximum in the higher price bucket

SET @UnitsAtHigherPrice = @LookupIaaSUpToMaxServerCountMoreExpensiveBucket -- max units in the more expensive bucket

IF @UnitsAtHigherPrice < 0

BEGIN

SET @UnitsAtHigherPrice = 0

END

-- The price is determined by the "prior" bucket max (at the "prior" bucket prices) plus additional units beyond that max which are

-- priced at the higher bucket's prices

DECLARE @DeployHoursForLowerPriceBand as float = 0

DECLARE @DeployHoursForHigherPriceBand as float = 0

SET @DeployHoursForLowerPriceBand = @UnitsAtLowerPrice \* @DeployHoursPerUnit

SET @DeployHoursForHigherPriceBand = @UnitsAtHigherPrice \* @DeployHoursPerUnitMoreExpensiveBucket

-- For record keeping purposes - average per unit deployment hours

SET @DeployHoursPerUnit = (@DeployHoursForLowerPriceBand + @DeployHoursForHigherPriceBand) / @IaaSTotalServerCount

DECLARE @SustainmentForLowerPriceBand\_PerYear as float = 0

DECLARE @SustainHoursHigherPriceBand\_PerYear as float = 0

SET @SustainmentForLowerPriceBand\_PerYear = @UnitsAtLowerPrice \* @SustainHoursPerUnit \* 12

SET @SustainHoursHigherPriceBand\_PerYear = @UnitsAtHigherPrice \* @SustainHoursPerUnitMoreExpensiveBucket \* 12

-- For record keeping purposes - average per unit sustainment hours

SET @SustainHoursPerUnit = (@SustainmentForLowerPriceBand\_PerYear + @SustainHoursHigherPriceBand\_PerYear) / 12

SET @IaaSDeploymentCost = (@DeployHoursForLowerPriceBand + @DeployHoursForHigherPriceBand) \* @DeploymentBlendedRate \* @DeployMarginOfError

SET @IaaSDeploymentCostEstimate = @IaaSDeploymentCost + @DeploymentUpFrontCost

SET @CostForSustainmentHours = (@SustainmentForLowerPriceBand\_PerYear + @SustainHoursHigherPriceBand\_PerYear) \* @SustainBlendedRate \* @SustainMarginOfError

SET @CostEstimateSustainment = @CostForSustainmentHours + @SustainmentUpFrontCost

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @IaaSDeploymentCostEstimate + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @CostEstimateSustainment + 999) / 1000) \* 1000

END

END

END

IF @PaaSNeeded = 1

BEGIN

DECLARE @PaaSDeployHoursPerUnit as float = 0

DECLARE @PaaSDeployComplexityMultiplier as float = 0

DECLARE @PaaSSustainHoursPerUnit as float = 0

DECLARE @PaaSSustainComplexityMultiplier as float = 0

SELECT TOP 1 @PaaSDeployHoursPerUnit = PaaS\_DeploymentHours

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@PaaSDeployComplexityMultiplier = PaaS\_DeployComplexityMultiplier

,@PaaSSustainHoursPerUnit = PaaS\_SustainmentHours

,@PaaSSustainComplexityMultiplier = PaaS\_SustainComplexityMultiplier

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSHighAvailability = PaaS\_HighAvailability

FROM [dbo].[paaspriceestimatelookup\_v2]

WHERE PaaS\_UpToMaxDatabaseCount >= @PaaSDatabaseCount

ORDER BY PaaS\_UpToMaxDatabaseCount

-- special case for catching the first "bucket"

if @PaaSDeployHoursPerUnit = 0 BEGIN

SELECT TOP 1 @PaaSDeployHoursPerUnit = PaaS\_DeploymentHours

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@PaaSDeployComplexityMultiplier = PaaS\_DeployComplexityMultiplier

,@PaaSSustainHoursPerUnit = PaaS\_SustainmentHours

,@PaaSSustainComplexityMultiplier = PaaS\_SustainComplexityMultiplier

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSHighAvailability = PaaS\_HighAvailability

FROM [dbo].[paaspriceestimatelookup\_v2]

ORDER BY PaaS\_UpToMaxDatabaseCount DESC

END

-- this is the pricing "bucket" we'll use to calculate the per-server cost

SET @PaaSCostPerServerOneTimeFee = @PaaSDeployHoursPerUnit \* @PaaSDeployComplexityMultiplier \* @DeploymentBlendedRate \* @DeployMarginOfError

SET @PaaSCostEstimateOneTimeFee = (@PaaSCostPerServerOneTimeFee \* @PaaSDatabaseCount) + @DeploymentUpFrontCost

SET @PaaSCostPerServerSustainment = @PaaSSustainHoursPerUnit \* @PaaSSustainComplexityMultiplier \* @SustainBlendedRate \* @SustainMarginOfError \* 12

SET @PaaSCostEstimateSustainment = (@PaaSCostPerServerSustainment \* @PaaSDatabaseCount) + @SustainmentUpFrontCost

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @PaaSCostEstimateOneTimeFee + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @PaaSCostEstimateSustainment + 999) / 1000) \* 1000

END

END

DECLARE @QuotedFirstYearCost as money = @EstimatedOneTimeCost + @EstimatedSustainmentPerYear

DECLARE @ComplexityMultiplier as float = 1.0

UPDATE romestimate\_v2

SET [romestimate\_v2].[ReadOnly] = 1

,LastModified = CURRENT\_TIMESTAMP

,LastModifiedBy = @UserIdentifier

,DateFinalized = CURRENT\_TIMESTAMP

,QuotedOneTimePrice = @EstimatedOneTimeCost

,QuotedSustainmentCost = @EstimatedSustainmentPerYear

,QuotedFirstYearCost = @QuotedFirstYearCost

,LookupIaaSUpToMaxEnvironments = @LookupIaaSUpToMaxEnvironments

,LookupIaaSUpToMaxServerCount = @LookupIaaSUpToMaxServerCount

,LookupPaaSUptoMaxDatabases = @LookupPaaSUptoMaxDatabases

,LookupPaaSHighAvailability = @LookupPaaSHighAvailability

,LookupDeploymentBlendedRate = @DeploymentBlendedRate

,LookupSustainmentBlendedRate = @SustainBlendedRate

,LookupDeploymentMarginOfError = @DeployMarginOfError

,LookupSustainmentMarginOfError = @SustainMarginOfError

,LookupDeploymentUpFrontCost = @DeploymentUpFrontCost

,LookupSustainmentUpFrontCost = @SustainmentUpFrontCost

,LookupIaaSDeployHours = @DeployHoursPerUnit

,LookupPaaSDeployHours = @PaaSDeployHoursPerUnit

,LookupIaaSDeployComplexityMultiplier = @ComplexityMultiplier

,LookupPaaSDeployComplexityMultiplier = @PaaSDeployComplexityMultiplier

,LookupIaaSSustainHours = @SustainHoursPerUnit

,LookupIaaSSustainComplexityMultiplier = @ComplexityMultiplier

,LookupPaaSSustainHours = @PaaSSustainHoursPerUnit

,LookupPaaSSustainComplexityMultiplier = @PaaSSustainComplexityMultiplier

WHERE ID = @RomId

-- Return the GUID for the ROM along with estimated costs (one time and sustainment per year)

SELECT @RomId as RomId,

@EstimatedOneTimeCost as OneTimeCost,

@EstimatedSustainmentPerYear as SustainmentPerYear,

@IaaSTotalServerCount AS TotalServerCountRequested,

@PaaSCostPerServerOneTimeFee as PaaSOneTimeFeePerServer,

@PaaSCostEstimateOneTimeFee as PaaSOneTimeFee,

@PaaSCostPerServerSustainment as PaaSSustainmentPerServer,

@PaaSCostEstimateSustainment as PaaSSustainment,

@DeployHoursPerUnit as IaaSCostPerServerOneTimeFee,

@DeploymentUpFrontCost as CostEstimateOneTimeFee,

@CostPerServerSustainment as CostPerServerSustainment,

@CostEstimateSustainment as CostEstimateSustainment,

@QuotedFirstYearCost as QuotedFirstYearCost,

@DeploymentBlendedRate as DeploymentBlendedRate,

@SustainBlendedRate as SustainmentBlendedRate,

@DeployMarginOfError as DeploymentMarginOfError,

@SustainMarginOfError as SustainmentMarginOfError,

@DeploymentUpFrontCost as DeploymentUpFrontCost,

@SustainmentUpFrontCost as SustainmentUpFrontCost,

@DeployHoursPerUnit as DeployHoursPerUnit,

@ComplexityMultiplier as DeployComplexityMultiplier,

@PaaSDeployHoursPerUnit as PaaSDeployHoursPerUnit,

@PaaSDeployComplexityMultiplier as PaaSDeployComplexityMultiplier,

@SustainHoursPerUnit as SustainHoursPerUnit,

@ComplexityMultiplier as SustainComplexityMultiplier,

@PaaSSustainHoursPerUnit as PaaSSustainHoursPerUnit,

@PaaSSustainComplexityMultiplier as PaaSSustainComplexityMultiplier

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ProvideEstimate] Script Date: 5/19/2021 1:49:58 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/20/2019

-- Description: Retrieves a rough order of magnitude (ROM) one-time and sustainment cost/year estimate (based on the values in cost

-- estimate lookup tables). Note that the ROM values must be SAVED/Updated before calling this. i.e. the only parameter is a

-- ROM id.

-- =============================================

CREATE PROCEDURE [dbo].[ProvideEstimate]

@RomId uniqueidentifier

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Retrieves a rough order of magnitude (ROM) one-time and sustainment cost/year estimate (based on the values in cost

-- estimate lookup tables)

DECLARE @EstimatedOneTimeCost money = 0

DECLARE @EstimatedSustainmentPerYear money = 0

-- The basic logic for IaaS is to select the row in iaaspriceestimatelookup where the

-- number of environments needed AND max number of servers needed in any environment is

-- Less Than Or Equal To the corresponding lookup row in the database.

-- For PaaS we look if high availability is needed to narrow down our query. Then we select

-- the row in paaspriceestimatelookup where the number of databases needed is

-- Less Than Or Equal To the values in that row

-- First, we must retrieve the ROM

-- Declare local variables to store the information required to produce an estimate

-- metadata about pricing lookup

DECLARE @LookupIaaSUpToMaxEnvironments as int

DECLARE @LookupIaaSUpToMaxServerCount as int

DECLARE @LookupIaaSUpToMaxServerCountPerEnvironment as int

DECLARE @LookupIaaSOneTimeFee as money

DECLARE @LookupIaaSSustainmentCost as money

DECLARE @LookupPaaSUptoMaxDatabases as int

DECLARE @LookupPaaSHighAvailability as bit

DECLARE @IaaSNeeded bit

DECLARE @IaaSType int

DECLARE @PaaSNeeded bit

DECLARE @SustainmentNeeded bit

DECLARE @IaaSEnvironmentCount int

DECLARE @IaaSTotalServerCount int -- across all environments (since environment is no longer taken into account)

DECLARE @PaaSDatabaseCount int

DECLARE @PaaSHighAvailability bit

DECLARE @PaaSCostPerServerOneTimeFee as money = 0

DECLARE @PaaSCostEstimateOneTimeFee as money = 0

DECLARE @PaaSCostPerServerSustainment as money = 0

DECLARE @PaaSCostEstimateSustainment as money = 0

DECLARE @CostPerServerOneTimeFee as money = 0

DECLARE @CostEstimateOneTimeFee as money = 0

DECLARE @CostPerServerSustainment as money = 0

DECLARE @CostEstimateSustainment as money = 0

-- Set variable values based on reading the ROM

SELECT @IaaSNeeded = IaaSNeeded

,@IaaSType = IaaSType

,@PaaSNeeded = PaaSNeeded

,@SustainmentNeeded = SustainmentNeeded

,@IaaSEnvironmentCount = IaaSEnvironmentCount

FROM [dbo].[romestimate] WHERE ID=@RomId

-- Now read the environments table to get the max number of servers per environment, if applicable

IF @IaaSNeeded = 1

BEGIN

SELECT @IaaSTotalServerCount = SUM(ServerOrVMCount) FROM [dbo].[environments]

WHERE ROMID = @RomId

END

-- Now collect info about any PaaS needed, including DB count and if they need to be Highly Available

IF @PaaSNeeded = 1

BEGIN

SELECT @PaaSDatabaseCount = DatabaseInstanceCount

,@PaaSHighAvailability = InstancesHighlyAvailable

FROM [dbo].[paas]

WHERE ROMID = @RomId

END

IF @IaaSNeeded = 1

BEGIN

DECLARE @OneTimeFee as money = 0

DECLARE @SustainmentCost as money = 0

SELECT TOP 1 @OneTimeFee = IaaS\_OneTimeFee,

@SustainmentCost = IaaS\_SustainmentCostPerYear

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

,@LookupIaaSOneTimeFee = IaaS\_OneTimeFee

,@LookupIaaSSustainmentCost = IaaS\_SustainmentCostPerYear

FROM [dbo].[iaaspriceestimatelookup]

WHERE IaaSType = @IaaSType AND

-- REQUEST BY JEN: Do not take into account environment count when

-- providing ROMs. This doesn't seem intuitively right, but it's the marching order

-- for now. @IaaSEnvironmentCount <= IaaS\_UpToMaxEnvironmentCount AND

IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount >= @IaaSTotalServerCount

ORDER BY

-- IaaS\_UpToMaxEnvironmentCount,

IaaS\_UpToMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

-- special case for catching the first "bucket"

if @OneTimeFee = 0 BEGIN

-- TODO: reconsider hardcoding these values

IF @IaaSTotalServerCount < 11 BEGIN

SELECT TOP 1 @OneTimeFee = IaaS\_OneTimeFee

,@SustainmentCost = IaaS\_SustainmentCostPerYear

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

,@LookupIaaSOneTimeFee = IaaS\_OneTimeFee

,@LookupIaaSSustainmentCost = IaaS\_SustainmentCostPerYear

FROM [dbo].[iaaspriceestimatelookup]

WHERE IaaSType = @IaaSType

ORDER BY IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

END

IF @IaaSTotalServerCount >= 11 BEGIN

SELECT TOP 1 @OneTimeFee = IaaS\_OneTimeFee

,@SustainmentCost = IaaS\_SustainmentCostPerYear

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

,@LookupIaaSOneTimeFee = IaaS\_OneTimeFee

,@LookupIaaSSustainmentCost = IaaS\_SustainmentCostPerYear

FROM [dbo].[iaaspriceestimatelookup]

WHERE IaaSType = @IaaSType

ORDER BY IaaS\_UptoMaxServerCount \* IaaS\_UpToMaxEnvironmentCount DESC

END

END

-- this is the pricing "bucket" we'll use to calculate the per-server cost

SET @CostPerServerOneTimeFee = @OneTimeFee / @LookupIaaSUpToMaxServerCount

SET @CostEstimateOneTimeFee = @CostPerServerOneTimeFee \* @IaaSTotalServerCount

SET @CostPerServerSustainment = @SustainmentCost / @LookupIaaSSustainmentCost

SET @CostEstimateSustainment = @CostPerServerSustainment \* @IaaSTotalServerCount

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @OneTimeFee + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @SustainmentCost + 999) / 1000) \* 1000

END

END

IF @PaaSNeeded = 1

BEGIN

DECLARE @OneTimeFeePaaS as money = 0

DECLARE @SustainmentCostPaaS as money = 0

-- In this case, we only care about max database count. Ignore the HighAvailability flag completely

SELECT TOP 1 @OneTimeFeePaaS = PaaS\_OneTimeFee

,@SustainmentCostPaaS = PaaS\_SustainmentCostPerYear

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSHighAvailability = PaaS\_HighAvailability

FROM [dbo].[paaspriceestimatelookup]

WHERE PaaS\_UpToMaxDatabaseCount >= @PaaSDatabaseCount

ORDER BY PaaS\_UpToMaxDatabaseCount

-- special case for catching the first "bucket" or exceeding the max count

if @OneTimeFeePaaS = 0 BEGIN

-- TODO: reconsider hardcoding these values

IF @PaaSDatabaseCount < 11 BEGIN

SELECT TOP 1 @OneTimeFeePaaS = PaaS\_OneTimeFee

,@SustainmentCostPaaS = PaaS\_SustainmentCostPerYear

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

FROM [dbo].[paaspriceestimatelookup]

ORDER BY PaaS\_UpToMaxDatabaseCount

END

IF @PaaSDatabaseCount >= 11 BEGIN

SELECT TOP 1 @OneTimeFeePaaS = PaaS\_OneTimeFee

,@SustainmentCostPaaS = PaaS\_SustainmentCostPerYear

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

FROM [dbo].[paaspriceestimatelookup]

ORDER BY PaaS\_UpToMaxDatabaseCount DESC

END

END

-- this is the pricing "bucket" we'll use to calculate the per-server cost

SET @PaaSCostPerServerOneTimeFee = @OneTimeFeePaaS / @LookupPaaSUptoMaxDatabases

SET @PaaSCostEstimateOneTimeFee = @PaaSCostPerServerOneTimeFee \* @PaaSDatabaseCount

SET @PaaSCostPerServerSustainment = @SustainmentCostPaaS / @LookupPaaSUptoMaxDatabases

SET @PaaSCostEstimateSustainment = @PaaSCostPerServerSustainment \* @PaaSDatabaseCount

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @OneTimeFeePaaS + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @SustainmentCostPaaS + 999) / 1000) \* 1000

END

END

DECLARE @QuotedFirstYearCost as money = @EstimatedOneTimeCost + @EstimatedSustainmentPerYear

-- Return the GUID for the ROM along with estimated costs (one time and sustainment per year)

SELECT @RomId AS RomId,

@EstimatedOneTimeCost as OneTimeCost,

@EstimatedSustainmentPerYear as SustainmentPerYear,

@IaaSTotalServerCount AS TotalServerCountRequested,

@PaaSCostPerServerOneTimeFee as PaaSOneTimeFeePerServer,

@PaaSCostEstimateOneTimeFee as PaaSOneTimeFee,

@PaaSCostPerServerSustainment as PaaSSustainmentPerServer,

@PaaSCostEstimateSustainment as PaaSSustainment,

@CostPerServerOneTimeFee as IaaSCostPerServerOneTimeFee,

@CostEstimateOneTimeFee as CostEstimateOneTimeFee,

@QuotedFirstYearCost as QuotedFirstYearCost

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ProvideEstimate\_v2] Script Date: 5/19/2021 1:49:59 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

--- =============================================

-- Author: Owen Emlen

-- Create date: 9/20/2019

-- Last Modified: 3/24/2021

-- Description: Retrieves a rough order of magnitude (ROM) one-time and sustainment cost/year estimate (based on the values in cost

-- estimate lookup tables). Note that the ROM values must be SAVED/Updated before calling this. i.e. the only parameter is a

-- ROM id.

-- =============================================

CREATE PROCEDURE [dbo].[ProvideEstimate\_v2]

@RomId uniqueidentifier

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Retrieves a rough order of magnitude (ROM) one-time and sustainment cost/year estimate (based on the values in cost

-- estimate lookup tables)

-- The basic logic for IaaS is to select the row in iaaspriceestimatelookup where the

-- number of environments needed AND max number of servers needed in any environment is

-- Less Than Or Equal To the corresponding lookup row in the database.

-- For PaaS we look if high availability is needed to narrow down our query. Then we select

-- the row in paaspriceestimatelookup where the number of databases needed is

-- Less Than Or Equal To the values in that row

-- First, we must retrieve the ROM

-- First, retrieve the ROM

-- Declare local variables to store the information required to produce an estimate

DECLARE @IaaSNeeded bit

DECLARE @IaaSType int

DECLARE @PaaSNeeded bit

DECLARE @SustainmentNeeded bit

DECLARE @IaaSEnvironmentCount int

DECLARE @IaaSTotalServerCount int -- Total server count across all environments (since # of environments no longer plays any factor)

DECLARE @PaaSDatabaseCount int

DECLARE @PaaSHighAvailability bit

-- metadata about pricing lookup

DECLARE @LookupIaaSUpToMaxEnvironments as int

DECLARE @LookupIaaSUpToMaxServerCount as int

DECLARE @LookupIaaSUpToMaxServerCountPerEnvironment as int

DECLARE @LookupPaaSUptoMaxDatabases as int

DECLARE @LookupPaaSHighAvailability as bit

DECLARE @PaaSCostPerServerOneTimeFee as money = 0

DECLARE @PaaSCostEstimateOneTimeFee as money = 0

DECLARE @PaaSCostPerServerSustainment as money = 0

DECLARE @PaaSCostEstimateSustainment as money = 0

DECLARE @IaaSDeploymentCost as money = 0

DECLARE @IaaSDeploymentCostEstimate as money = 0

DECLARE @CostEstimateSustainment as money = 0

DECLARE @DeployMarginOfError as float

DECLARE @SustainMarginOfError as float

DECLARE @DeploymentBlendedRate as money

DECLARE @SustainBlendedRate as money

DECLARE @DeploymentUpFrontCost as money

DECLARE @SustainmentUpFrontCost as money

DECLARE @CostForSustainmentHours as money = 0

DECLARE @DeployHoursPerUnit as float = 0 -- this may take into account 1 or more "bucket"

DECLARE @SustainHoursPerUnit as float = 0 -- this may take into account 1 or more "bucket"

DECLARE @EstimatedOneTimeCost money = 0

DECLARE @EstimatedSustainmentPerYear money = 0

DECLARE @VlmfVersion as Int = 2

-- Set variable values based on reading the ROM

SELECT @IaaSNeeded = IaaSNeeded

,@IaaSType = IaaSType

,@PaaSNeeded = PaaSNeeded

,@SustainmentNeeded = SustainmentNeeded

,@IaaSEnvironmentCount = IaaSEnvironmentCount

,@VlmfVersion = VlmfVersion

FROM [dbo].[romestimate\_v2] WHERE ID=@RomId

SELECT @DeployMarginOfError = DeploymentMarginOfError

,@SustainMarginOfError = SustainmentMarginOfError

,@DeploymentBlendedRate = DeploymentBlendedHourlyCost

,@SustainBlendedRate = SustainmentBlendedHourlyCost

,@DeploymentUpFrontCost = DeploymentUpFrontCost

,@SustainmentUpFrontCost = SustainmentUpFrontCost

FROM [dbo].[pricingfactors\_v2]

WHERE [dbo].[pricingfactors\_v2].VlmfVersion = 2

-- Now read the environments table to get the max number of servers per environment, if applicable

IF @IaaSNeeded = 1

BEGIN

SELECT @IaaSTotalServerCount = Sum(ServerOrVMCount) FROM [dbo].[environments\_v2]

WHERE ROMID = @RomId

END

-- Now collect info about any PaaS needed, including DB count and if they need to be Highly Available

IF @PaaSNeeded = 1

BEGIN

SELECT @PaaSDatabaseCount = DatabaseInstanceCount

,@PaaSHighAvailability = InstancesHighlyAvailable

FROM [dbo].[paas\_v2]

WHERE ROMID = @RomId

END

IF @IaaSNeeded = 1

BEGIN

-- We need to collect up to two rows - the actual hours row for server count requested, and the prior row

DECLARE @MaxInModel as Integer = 0

DECLARE @IDTopRow as Integer = 0

DECLARE @DeployHoursPerUnitMoreExpensiveBucket as float = 0

DECLARE @SustainHoursPerUnitMoreExpensiveBucket as float = 0

DECLARE @LookupIaaSUpToMaxServerCountMoreExpensiveBucket as float = 0

DECLARE @UnitsAtLowerPrice as float = 0

DECLARE @UnitsAtHigherPrice as float = 0

DECLARE @MinUnitsCheaperBucket as float = 0

DECLARE @MaxUnitsCheaperBucket as float = 0

-- SPECIAL CASE - only asking for 1 unit

IF @IaaSTotalServerCount = 1

BEGIN

SELECT TOP 1 @IDTopRow = ID

,@DeployHoursPerUnit = IaaS\_DeploymentHours

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = 1

,@SustainHoursPerUnit = IaaS\_SustainmentHours

FROM [dbo].[iaaspriceestimatelookup\_v2]

WHERE IaaSType = @IaaSType AND @IaaSTotalServerCount = IaaS\_UpToMaxServerCount

ORDER BY IaaS\_UpToMaxServerCount \* IaaS\_UpToMaxEnvironmentCount

SET @IaaSDeploymentCost = @DeployHoursPerUnit \* @DeploymentBlendedRate \* @DeployMarginOfError \* @IaaSTotalServerCount

SET @IaaSDeploymentCostEstimate = @IaaSDeploymentCost + @DeploymentUpFrontCost

SET @CostForSustainmentHours = @SustainHoursPerUnit \* @SustainBlendedRate \* @SustainMarginOfError \* 12

SET @CostEstimateSustainment = (@CostForSustainmentHours \* @IaaSTotalServerCount) + @SustainmentUpFrontCost

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @IaaSDeploymentCostEstimate + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @CostEstimateSustainment + 999) / 1000) \* 1000

END

END

IF @IaaSTotalServerCount > 1

BEGIN

DECLARE @LookupIaaSMinCountMoreExpensiveBucket as float = 0

-- Grab the max number of IaaS VMs in our model for yet another corner case where IaaS VMs requested exceeds the max in our model (currently 200)

SELECT TOP 1 @MaxInModel = IaaS\_UpToMaxServerCount

from [dbo].[iaaspriceestimatelookup\_v2]

WHERE @IaaSType = IaaSType

ORDER BY [dbo].[iaaspriceestimatelookup\_v2].IaaS\_UpToMaxServerCount DESC

-- Determine the "Bucket" we're in (VMs requested >= Minimum VMs associated with the bucket), order descending so we pick the highest matching bucket

SELECT TOP 1 @IDTopRow = ID

,@DeployHoursPerUnit = IaaS\_DeploymentHours

,@LookupIaaSUpToMaxEnvironments = IaaS\_UptoMaxEnvironmentCount

,@LookupIaaSUpToMaxServerCountPerEnvironment = IaaS\_UptoMaxServerCount

,@LookupIaaSUpToMaxServerCount = IaaS\_UptoMaxServerCount

,@SustainHoursPerUnit = IaaS\_SustainmentHours

,@MinUnitsCheaperBucket = IaaS\_MinServerCount

,@MaxUnitsCheaperBucket = IaaS\_UpToMaxServerCount

FROM [dbo].[iaaspriceestimatelookup\_v2]

WHERE IaaSType = @IaaSType AND @IaaSTotalServerCount > IaaS\_MinServerCount

ORDER BY IaaS\_MinServerCount DESC

-- Handle the corner case where the customer requests >= (currently) 200 IaaS servers

IF @IaaSTotalServerCount >= @MaxInModel

BEGIN

SET @DeployHoursPerUnitMoreExpensiveBucket = @DeployHoursPerUnit

SET @SustainHoursPerUnitMoreExpensiveBucket = @SustainHoursPerUnit

SET @UnitsAtLowerPrice = @IaaSTotalServerCount

SET @UnitsAtHigherPrice = 0

SET @LookupIaaSUpToMaxServerCountMoreExpensiveBucket = @MaxInModel

END

ELSE -- Handle the case where the customer requests a number of IaaS servers that fits into a "bucket"

BEGIN

-- retrieve the "previous" bucket, since we will price all machines up to the bucket's max VM count @IaaSTotalServerCount using that prior bucket

SELECT TOP 1 @DeployHoursPerUnitMoreExpensiveBucket = IaaS\_DeploymentHours

,@SustainHoursPerUnitMoreExpensiveBucket = IaaS\_SustainmentHours

,@LookupIaaSUpToMaxServerCountMoreExpensiveBucket = IaaS\_UptoMaxServerCount

,@LookupIaaSMinCountMoreExpensiveBucket = IaaS\_MinServerCount

FROM [dbo].[iaaspriceestimatelookup\_v2]

WHERE IaaSType = @IaaSType AND (ID = @IDTopRow - 1) -- this is safe because we took care of the special case where 1 VM is requested

END

SET @UnitsAtLowerPrice = @IaaSTotalServerCount - @LookupIaaSUpToMaxServerCountMoreExpensiveBucket -- remaining VMs above and beyond the maximum in the higher price bucket

SET @UnitsAtHigherPrice = @LookupIaaSUpToMaxServerCountMoreExpensiveBucket -- "up to n units" in the more expensive bucket

IF @UnitsAtHigherPrice < 0

BEGIN

SET @UnitsAtHigherPrice = 0

END

-- The price is determined by the "prior" bucket max (at the "prior" bucket prices) plus additional units beyond that max which are

-- priced at the higher bucket's prices

DECLARE @DeployHoursForLowerPriceBand as float = 0

DECLARE @DeployHoursForHigherPriceBand as float = 0

SET @DeployHoursForLowerPriceBand = @UnitsAtLowerPrice \* @DeployHoursPerUnit

SET @DeployHoursForHigherPriceBand = @UnitsAtHigherPrice \* @DeployHoursPerUnitMoreExpensiveBucket

-- For record keeping purposes - average per unit deployment hours

SET @DeployHoursPerUnit = (@DeployHoursForLowerPriceBand + @DeployHoursForHigherPriceBand) / @IaaSTotalServerCount

DECLARE @SustainmentForLowerPriceBand\_PerYear as float = 0

DECLARE @SustainHoursHigherPriceBand\_PerYear as float = 0

SET @SustainmentForLowerPriceBand\_PerYear = @UnitsAtLowerPrice \* @SustainHoursPerUnit \* 12

SET @SustainHoursHigherPriceBand\_PerYear = @UnitsAtHigherPrice \* @SustainHoursPerUnitMoreExpensiveBucket \* 12

-- For record keeping purposes - average per unit sustainment hours

SET @SustainHoursPerUnit = (@SustainmentForLowerPriceBand\_PerYear + @SustainHoursHigherPriceBand\_PerYear) / 12

SET @IaaSDeploymentCost = (@DeployHoursForLowerPriceBand + @DeployHoursForHigherPriceBand) \* @DeploymentBlendedRate \* @DeployMarginOfError

SET @IaaSDeploymentCostEstimate = @IaaSDeploymentCost + @DeploymentUpFrontCost

SET @CostForSustainmentHours = (@SustainmentForLowerPriceBand\_PerYear + @SustainHoursHigherPriceBand\_PerYear) \* @SustainBlendedRate \* @SustainMarginOfError

SET @CostEstimateSustainment = @CostForSustainmentHours + @SustainmentUpFrontCost

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @IaaSDeploymentCostEstimate + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @CostEstimateSustainment + 999) / 1000) \* 1000

END

END

END

IF @PaaSNeeded = 1

BEGIN

DECLARE @PaaSDeployHoursPerUnit as float = 0

DECLARE @PaaSDeployComplexityMultiplier as float = 0

DECLARE @PaaSSustainHoursPerUnit as float = 0

DECLARE @PaaSSustainComplexityMultiplier as float = 0

SELECT TOP 1 @PaaSDeployHoursPerUnit = PaaS\_DeploymentHours

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@PaaSDeployComplexityMultiplier = PaaS\_DeployComplexityMultiplier

,@PaaSSustainHoursPerUnit = PaaS\_SustainmentHours

,@PaaSSustainComplexityMultiplier = PaaS\_SustainComplexityMultiplier

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSHighAvailability = PaaS\_HighAvailability

FROM [dbo].[paaspriceestimatelookup\_v2]

WHERE PaaS\_UpToMaxDatabaseCount >= @PaaSDatabaseCount

ORDER BY PaaS\_UpToMaxDatabaseCount

-- special case for catching the first "bucket"

if @PaaSDeployHoursPerUnit = 0 BEGIN

SELECT TOP 1 @PaaSDeployHoursPerUnit = PaaS\_DeploymentHours

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@PaaSDeployComplexityMultiplier = PaaS\_DeployComplexityMultiplier

,@PaaSSustainHoursPerUnit = PaaS\_SustainmentHours

,@PaaSSustainComplexityMultiplier = PaaS\_SustainComplexityMultiplier

,@LookupPaaSUptoMaxDatabases = PaaS\_UpToMaxDatabaseCount

,@LookupPaaSHighAvailability = PaaS\_HighAvailability

FROM [dbo].[paaspriceestimatelookup\_v2]

ORDER BY PaaS\_UpToMaxDatabaseCount DESC

END

-- this is the pricing "bucket" we'll use to calculate the per-server cost

SET @PaaSCostPerServerOneTimeFee = @PaaSDeployHoursPerUnit \* @PaaSDeployComplexityMultiplier \* @DeploymentBlendedRate \* @DeployMarginOfError

SET @PaaSCostEstimateOneTimeFee = (@PaaSCostPerServerOneTimeFee \* @PaaSDatabaseCount) + @DeploymentUpFrontCost

SET @PaaSCostPerServerSustainment = @PaaSSustainHoursPerUnit \* @PaaSSustainComplexityMultiplier \* @SustainBlendedRate \* @SustainMarginOfError \* 12

SET @PaaSCostEstimateSustainment = (@PaaSCostPerServerSustainment \* @PaaSDatabaseCount) + @SustainmentUpFrontCost

-- Add on the one time fee and possibly the sustainment cost (if sustainment needed) to the

-- corresponding local variables

SET @EstimatedOneTimeCost = floor((@EstimatedOneTimeCost + @PaaSCostEstimateOneTimeFee + 999) / 1000) \* 1000

IF @SustainmentNeeded = 1

BEGIN

SET @EstimatedSustainmentPerYear = floor((@EstimatedSustainmentPerYear + @PaaSCostEstimateSustainment + 999) / 1000) \* 1000

END

END

DECLARE @QuotedFirstYearCost as money = @EstimatedSustainmentPerYear + @EstimatedOneTimeCost

DECLARE @ComplexityMultiplier as float = 1.0

-- Return the GUID for the ROM along with estimated costs (one time and sustainment per year)

SELECT @RomId as RomId,

@EstimatedOneTimeCost as OneTimeCost,

@EstimatedSustainmentPerYear as SustainmentPerYear,

@IaaSTotalServerCount AS TotalServerCountRequested,

@PaaSCostPerServerOneTimeFee as PaaSOneTimeFeePerServer,

@PaaSCostEstimateOneTimeFee as PaaSOneTimeFee,

@PaaSCostPerServerSustainment as PaaSSustainmentPerServer,

@PaaSCostEstimateSustainment as PaaSSustainment,

@IaaSDeploymentCost as IaaSCostPerServerOneTimeFee,

@IaaSDeploymentCostEstimate as CostEstimateOneTimeFee,

@QuotedFirstYearCost as QuotedFirstYearCost,

@DeploymentBlendedRate as DeploymentBlendedRate,

@SustainBlendedRate as SustainmentBlendedRate,

@DeployMarginOfError as DeploymentMarginOfError,

@SustainMarginOfError as SustainmentMarginOfError,

@DeploymentUpFrontCost as DeploymentUpFrontCost,

@SustainmentUpFrontCost as SustainmentUpFrontCost,

@DeployHoursPerUnit as DeployHoursPerUnit,

@ComplexityMultiplier as DeployComplexityMultiplier,

@PaaSDeployHoursPerUnit as PaaSDeployHoursPerUnit,

@PaaSDeployComplexityMultiplier as PaaSDeployComplexityMultiplier,

@SustainHoursPerUnit as SustainHoursPerUnit,

@ComplexityMultiplier as SustainComplexityMultiplier,

@PaaSSustainHoursPerUnit as PaaSSustainHoursPerUnit,

@PaaSSustainComplexityMultiplier as PaaSSustainComplexityMultiplier

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ReadIaaSTypeNames] Script Date: 5/19/2021 1:50:00 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/26/2019

-- Description: Reads all IaaS type names from the lookup table

-- =============================================

CREATE PROCEDURE [dbo].[ReadIaaSTypeNames]

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- TODO - need to read PaaS and read and flatten Environment info associated with the ROM

SELECT \* FROM [dbo].[iaastypenames]

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ReadRom] Script Date: 5/19/2021 1:50:00 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/19/2019

-- Description: Reads an existing ROM (which may only be partially filled-out) by GUID. Returns IaaS environments

-- and PaaS information in a flattened structure for easy use

-- =============================================

CREATE PROCEDURE [dbo].[ReadRom]

@RomId uniqueidentifier

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- NOTE: we are not using PIVOT to flatten our results here (I'm not exactly a SQL guru)

-- The back-end will need to flatten PaaS and IaaS Environment info associated with the ROM

SELECT

ROM.ID

,ROM.ProjectName

,ROM.DateCreated

,ROM.LastModified

,ROM.LastModifiedBy

,ROM.IaaSNeeded

,ROM.IaaSType

,ROM.IaaSEnvironmentCount

,ROM.PaaSNeeded

,ROM.SustainmentNeeded

,ROM.ReadOnly

,ROM.DateFinalized

,ROM.QuotedOneTimePrice

,ROM.QuotedSustainmentCost

,ROM.LookupIaaSUpToMaxEnvironments

,ROM.LookupIaaSUpToMaxServerCount

,ROM.LookupIaaSOneTimeFee

,ROM.LookupIaaSSustainmentCost

,ROM.LookupPaaSUptoMaxDatabases

,ROM.LookupPaaSHighAvailability

,ROM.LookupPaaSOneTimeFee

,ROM.LookupPaaSSustainmentCost

,ROM.Notes

,ROM.VIPRID

,ROM.VASI

,ROM.QuotedFirstYearCost

,ENVS.EnvironmentIndex

,ENVS.EnvironmentName

,ENVS.ServerOrVMCount

,PAAS.DatabaseInstanceCount

,PAAS.InstancesHighlyAvailable

FROM [dbo].[romestimate] as ROM

LEFT OUTER JOIN [dbo].[environments] as ENVS ON ENVS.ROMID = ROM.ID

LEFT OUTER JOIN [dbo].[paas] AS PAAS ON PAAS.ROMID = ROM.ID

WHERE ROM.ID = @RomId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ReadRom\_v2] Script Date: 5/19/2021 1:50:00 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/19/2019

-- Description: Reads an existing ROM (which may only be partially filled-out) by GUID. Returns IaaS environments

-- and PaaS information in a flattened structure for easy use

-- =============================================

CREATE PROCEDURE [dbo].[ReadRom\_v2]

@RomId uniqueidentifier

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- NOTE: we are not using PIVOT to flatten our results here (I'm not exactly a SQL guru)

-- The back-end will need to flatten PaaS and IaaS Environment info associated with the ROM

SELECT

ROM.ID

,ROM.ProjectName

,ROM.DateCreated

,ROM.LastModified

,ROM.LastModifiedBy

,ROM.IaaSNeeded

,ROM.IaaSType

,ROM.IaaSEnvironmentCount

,ROM.PaaSNeeded

,ROM.SustainmentNeeded

,ROM.ReadOnly

,ROM.DateFinalized

,ROM.QuotedOneTimePrice

,ROM.QuotedSustainmentCost

,ROM.LookupIaaSUpToMaxEnvironments

,ROM.LookupIaaSUpToMaxServerCount

,ROM.LookupPaaSUptoMaxDatabases

,ROM.LookupPaaSHighAvailability

,ROM.Notes

,ROM.VIPRID

,ROM.VASI

,ROM.LookupDeploymentMarginOfError

,ROM.LookupSustainmentMarginOfError

,ROM.LookupDeploymentUpFrontCost

,ROM.LookupSustainmentUpFrontCost

,ROM.QuotedFirstYearCost

,ROM.LookupIaaSDeployHours

,ROM.LookupIaaSSustainHours

,ROM.LookupPaaSDeployHours

,ROM.LookupPaaSSustainHours

,ROM.LookupPaaSDeployComplexityMultiplier

,ROM.LookupPaaSSustainComplexityMultiplier

,ROM.LookupIaaSDeployComplexityMultiplier

,ROM.LookupIaaSSustainComplexityMultiplier

,ROM.LookupDeploymentBlendedRate

,ROM.LookupSustainmentBlendedRate

,ROM.VlmfVersion

,ENVS.EnvironmentIndex

,ENVS.EnvironmentName

,ENVS.ServerOrVMCount

,PAAS.DatabaseInstanceCount

,PAAS.InstancesHighlyAvailable

FROM [dbo].[romestimate\_v2] as ROM

LEFT OUTER JOIN [dbo].[environments\_v2] as ENVS ON ENVS.ROMID = ROM.ID

LEFT OUTER JOIN [dbo].[paas\_v2] AS PAAS ON PAAS.ROMID = ROM.ID

WHERE ROM.ID = @RomId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[UpdateRom] Script Date: 5/19/2021 1:50:00 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/18/2019

-- Description: Inserts a new ROM (which may only be partially filled-out). Returns a GUID for the new ROM that can be used

-- to gain future read/write access to the ROM

-- =============================================

CREATE PROCEDURE [dbo].[UpdateRom]

@RomId uniqueidentifier,

@RomName varchar(MAX) = Unnamed,

@UserName varchar(MAX) = Unknown,

@IaaSNeeded bit = 0,

@IaaSType int = 0,

@IaaSEnvironmentCount int = 0,

@PaaSNeeded bit = 0,

@PaaSDBInstances int = 0,

@PaaSHighAvailability bit = 0,

@SustainmentNeeded bit = 0,

@Notes varchar(MAX) = null,

@VIPRID varchar(MAX) = null,

@VASI varchar(MAX) = null,

@Environment1 bit = 0,

@Environment1Name varchar(MAX) = null,

@Environment1ServerOrVMCount int = 0,

@Environment2 bit = 0,

@Environment2Name varchar(MAX) = null,

@Environment2ServerOrVMCount int = 0,

@Environment3 bit = 0,

@Environment3Name varchar(MAX) = null,

@Environment3ServerOrVMCount int = 0,

@Environment4 bit = 0,

@Environment4Name varchar(MAX) = null,

@Environment4ServerOrVMCount int = 0,

@Environment5 bit = 0,

@Environment5Name varchar(MAX) = null,

@Environment5ServerOrVMCount int = 0

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Updates an existing ROM, including updates to the environments table and PaaS, based

-- on the parameters passed in, sets the last modified date, last modified by, etc.

UPDATE [dbo].[romestimate]

SET

[ProjectName]=@RomName

,[LastModified]=CURRENT\_TIMESTAMP

,[LastModifiedBy]=@UserName

,[IaaSNeeded]=@IaaSNeeded

,[IaaSType]=@IaaSType

,[IaaSEnvironmentCount]=@IaaSEnvironmentCount

,[PaaSNeeded]=@PaaSNeeded

,[SustainmentNeeded]=@SustainmentNeeded

,[VIPRID]=@VIPRID

,[VASI]=@VASI

,[Notes]=@Notes

,[ReadOnly]=0

WHERE ID = @RomId

DELETE FROM [dbo].[paas] WHERE ROMID=@RomId

-- now re-insert into the PaaS table, if appropriate

IF @PaaSNeeded = 1

BEGIN

INSERT INTO [dbo].[paas]

([ROMID]

,[DatabaseInstanceCount]

,[InstancesHighlyAvailable])

VALUES

(@RomId,

@PaaSDBInstances,

@PaaSHighAvailability)

END

DELETE FROM [dbo].[environments] WHERE ROMID=@RomId

-- now insert into the environment table data, if appropriate

IF @IaaSEnvironmentCount > 0

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,1

,@Environment1Name

,@Environment1ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 1

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,2

,@Environment2Name

,@Environment2ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 2

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,3

,@Environment3Name

,@Environment3ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 3

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,4

,@Environment4Name

,@Environment4ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 4

BEGIN

INSERT INTO [dbo].[environments]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,5

,@Environment5Name

,@Environment5ServerOrVMCount)

END

-- Return the GUID for the ROM

SELECT @RomId

END

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[UpdateRom\_v2] Script Date: 5/19/2021 1:50:00 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: Owen Emlen

-- Create date: 9/18/2019

-- Description: Inserts a new ROM (which may only be partially filled-out). Returns a GUID for the new ROM that can be used

-- to gain future read/write access to the ROM

-- =============================================

CREATE PROCEDURE [dbo].[UpdateRom\_v2]

@RomId uniqueidentifier,

@RomName varchar(MAX) = Unnamed,

@UserName varchar(MAX) = Unknown,

@IaaSNeeded bit = 0,

@IaaSType int = 0,

@IaaSEnvironmentCount int = 0,

@PaaSNeeded bit = 0,

@PaaSDBInstances int = 0,

@PaaSHighAvailability bit = 0,

@SustainmentNeeded bit = 0,

@Notes varchar(MAX) = null,

@VIPRID varchar(MAX) = null,

@VASI varchar(MAX) = null,

@Environment1 bit = 0,

@Environment1Name varchar(MAX) = null,

@Environment1ServerOrVMCount int = 0,

@Environment2 bit = 0,

@Environment2Name varchar(MAX) = null,

@Environment2ServerOrVMCount int = 0,

@Environment3 bit = 0,

@Environment3Name varchar(MAX) = null,

@Environment3ServerOrVMCount int = 0,

@Environment4 bit = 0,

@Environment4Name varchar(MAX) = null,

@Environment4ServerOrVMCount int = 0,

@Environment5 bit = 0,

@Environment5Name varchar(MAX) = null,

@Environment5ServerOrVMCount int = 0

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Updates an existing ROM, including updates to the environments table and PaaS, based

-- on the parameters passed in, sets the last modified date, last modified by, etc.

UPDATE [dbo].[romestimate\_v2]

SET

[ProjectName]=@RomName

,[LastModified]=CURRENT\_TIMESTAMP

,[LastModifiedBy]=@UserName

,[IaaSNeeded]=@IaaSNeeded

,[IaaSType]=@IaaSType

,[IaaSEnvironmentCount]=@IaaSEnvironmentCount

,[PaaSNeeded]=@PaaSNeeded

,[SustainmentNeeded]=@SustainmentNeeded

,[VIPRID]=@VIPRID

,[VASI]=@VASI

,[Notes]=@Notes

,[ReadOnly]=0

WHERE ID = @RomId

DELETE FROM [dbo].[paas\_v2] WHERE ROMID=@RomId

-- now re-insert into the PaaS table, if appropriate

IF @PaaSNeeded = 1

BEGIN

INSERT INTO [dbo].[paas\_v2]

([ROMID]

,[DatabaseInstanceCount]

,[InstancesHighlyAvailable])

VALUES

(@RomId,

@PaaSDBInstances,

@PaaSHighAvailability)

END

DELETE FROM [dbo].[environments\_v2] WHERE ROMID=@RomId

-- now insert into the environment table data, if appropriate

IF @IaaSEnvironmentCount > 0

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,1

,@Environment1Name

,@Environment1ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 1

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,2

,@Environment2Name

,@Environment2ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 2

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,3

,@Environment3Name

,@Environment3ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 3

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,4

,@Environment4Name

,@Environment4ServerOrVMCount)

END

IF @IaaSEnvironmentCount > 4

BEGIN

INSERT INTO [dbo].[environments\_v2]

([ROMID]

,[EnvironmentIndex]

,[EnvironmentName]

,[ServerOrVMCount])

VALUES

(@RomId

,5

,@Environment5Name

,@Environment5ServerOrVMCount)

END

-- Return the GUID for the ROM

SELECT @RomId

END

GO

EXEC sys.sp\_addextendedproperty @name=N'MS\_Description', @value=N'Environments associated with a ROM' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'environments', @level2type=N'CONSTRAINT',@level2name=N'FK\_environments\_romestimate'

GO

EXEC sys.sp\_addextendedproperty @name=N'MS\_Description', @value=N'Environments associated with a ROM (v2)' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'environments\_v2', @level2type=N'CONSTRAINT',@level2name=N'FK\_environments\_v2\_romestimate\_v2'

GO

EXEC sys.sp\_addextendedproperty @name=N'MS\_Description', @value=N'IaaS Type Name reference' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'romestimate', @level2type=N'CONSTRAINT',@level2name=N'FK\_romestimate\_iaastypenames'

GO

EXEC sys.sp\_addextendedproperty @name=N'MS\_Description', @value=N'IaaS Type Name reference' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'TABLE',@level1name=N'romestimate\_v2', @level2type=N'CONSTRAINT',@level2name=N'FK\_romestimate\_iaastypenames\_v2'

GO

USE [master]

GO

ALTER DATABASE [romestimator] SET READ\_WRITE

GO