AS500242

Planning Driven by Data: Organize, Evaluate, Illustrate

Robert Manna

Digital Practice Product / Project Manager | Stantec

Esra Abumounshar

Architectural Designer | Generative Design Lead | Stantec

Learning Objectives

- Learn how to assess raw data and identify opportunities to normalize the data for consumption by tools.
- Learn how to define a process for converting raw data to a usable data model.
- Discover how users can migrate data processing out of their scripts and into tools better suited for data manipulation
- Learn how to build a process for generating a variety of graphic outputs based on data

Description

With increasing focus on computation and data-driven solutions, architects and designers need to become more comfortable with data manipulation and preparation. One of the biggest risks that comes from the increased use of tools such as Dynamo is that a large amount of data manipulation is defined in the scripts themselves. This locks the data business logic into the script, which is accessible by only a subset of users. We have found that with large projects, it's better to utilize datacentric tools for formatting and manipulation prior to feeding the data into Dynamo.

This affords the possibility that the final data set can be used to drive multiple outputs.

In this class, we will focus on the key attributes of a good data process, using a recent project as an example. We will take you through the cleaning process, as well as the script development in Dynamo and the graphic production process in Revit software that was complimentary to other data-visualization tools. We'll also review the decision-making process that went into our choice between using Revit or FormIt software.

Speaker(s)

Robert leads a busy life. Trained as an architect he dove headfirst into Design Technology at the start of his career and hasn't looked back. General problem solver, occasional user interface designer and wanna-be DB admin Robert is currently a Product/Project (depends on the day of the week) Manager for Stantec Building's Digital Practice Team. Recently Robert has managed several project focused efforts dealing with data and data visualization and is currently the PM for a SaaS Solution being developed by Stantec. Somewhere he fits in being a husband, father of two, tri-athlete, avid skier and occasional backhoe operator.

Esra is the Generative & Computational Design Lead at Stantec with a background in architecture, interior design and facilities management. Esra focuses on developing digital design assist tools and driving adoption. With a passion for innovation Esra focuses her energy on researching and connecting with other technology driven individuals to exchange ideas and findings. On her downtime, she is usually reading a book on coding and data while juggling her second full time job as a mother of a 6-year-old.

Table of contents

Goals

What do you want to accomplish?

Clean data is the cornerstone to successful outcomes, whether discussing a DWG file, a Revit model, or a pure table of data. In master planning efforts data is often the cornerstone upon which “The Plan” is built, but while data may be the cornerstone, the consumers and even the planners of The Plan all want to focus on and look at the elegant façade, not the the underlying skeleton. Getting from tables of data to visually compelling outcomes is the challenge, how we construct the structure of our data can ultimately determine how much effort is required to have the beautiful façade in the end.

In this particular case study the client supplied data and offered assurances that the data was:

1. Clean and well organized.
2. All the data we would need.

This is a familiar story for anyone who has ever dealt with data from client(s) (internal or external). Needless to say, neither was 100% true or accurate. How quickly a team can evaluate the data provided to determine where the gaps are, and what is required, has a major impact on the success of the project. Building an appropriate data model from what is supplied, that affords flexibility and adaptability, is the key to successful data management and the ultimate goal of successful outputs from the data.

What were the project goals?

The work discussed in this course was the outcome of a Project Team that wanted to:

- understand existing conditions in the context of program distribution across twenty-five plus buildings in a dense urban campus
- propose a master plan for the healthcare campus, including:
  - options for the amount of program growth
  - options for new construction on multiple possible sites, including different massing proposals for each possible site
  - options for how the program would be distributed
- automatically generate geometry to illustrate the campus and its program components (blocking & stacking) in an overall context model
- navigate and explore the 3D model from any device
- easily update the 3D model, potentially in client meetings
- generate traditional 2D artifacts/architectural illustration, tables, charts, diagrams, etc.

In assessing the scope of the campus and the project as a whole, it was clear that more than a 3D model would be required to achieve any of their goals. A good data source would be required to help deliver any of the advanced goals while allowing the delivery of more traditional master plan artifacts.

Data

Everyone has it, what to do with it?

Data is not “hard,” the hard part about data is avoiding human error. Human error is avoided by limiting the frequency that data has to be inputted or manipulated by humans, reducing the risk of typos and errant clicks. Human error is one of the most frequent causes of problems in this type of work, avoiding repetitive data entry is one of the best methods to ensure data consistency, while also making it easy to change, filter or modify the data in the first place.

Normalized Stars

“Stars” are the key to a good data model[1]. The issue is that most of the time when data is received it is not formatted or organized in a way that lends itself to analysis and visualization (in Excel, Dynamo/Revit/Grashopper, or Power BI). The concept of a “star” in the context of a data model has to do with the graphic appearance of the data model in a relationship view, it should be possible to overlay a “star” on the relationship diagram.

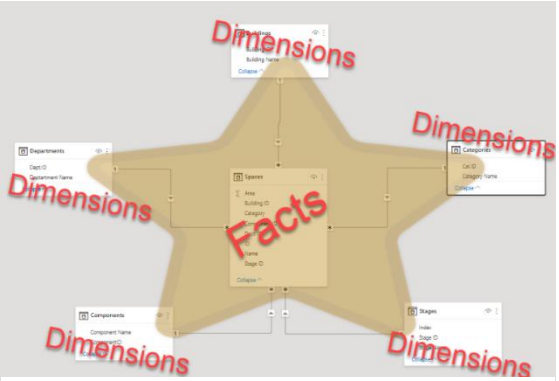


Figure 1 a simple data model that is a in a star schema format

Vocabulary

- Normalization[2] – Normalization is the process of reorganizing data so that it meets two basic requirements:
  1. There is no redundancy of data, all data is stored in only one place.
  2. Data dependencies are logical, all related data items are stored together.
- Fact Table[3] – a table that stores measures (values) that measure the subject of the data model; for example, sales, costs, profit, area, quantities, etc.
- Dimension Table(s)[4] – values that help to categorize or describe facts so that business questions can be answered, for example; “please create a graph that sums all Areas (fact) **BY** Department (dimension)”
- Relationship[5] – the connection between data in one table to data in another table, typically using “keys” or unique values that define the relationship. Relationships can have direction; one to one, one to many/many to one (most typical) and many to many. Relationships are the foundation to data normalization.

---

[1] Assuming the use of Power BI (or a similar product) and that the end user is not a data expert by trade.
[2] Adapted from Techopedia - (Normalization, 2020)
[3] Adapted from Wikipedia - (Fact Table, 2021)
[4] Adapted from Wikipedia - (Dimension (data warehouse), 2021)
[5] Adapted from Lifewire - (Chapple, 2021)

Normalization Process

Data normalization can be quick, or it can be arduous. More consistent data will be easier to normalize, inconsistent data either in terms of structure or values will take more effort and time to clean. Regardless of quantity or quality, these are the typical steps used to clean-up and normalize data in Excel:

- Identify Facts versus Dimensions. Facts will be values like Area, Volume, Room Number, Instance Mark; attributes that are unique to that particular item. Information like Building Name, Level, Type Name, Category, Department/Sub-Department are more often Dimensions and should be split into their own unique tables.[6]
    - Some values like Room Name could remain as part of the Fact Table or be split out as a dimension. It depends upon the frequency of repetition. In a healthcare setting it's possible that Room Name as part of a Dimension may make sense, in other data models "Name" may be considered unique to each row or instance.
    - Think about how the data will need to be analyzed. Will it be important to sum all of the area by a Room Name? Is there another Dimension that achieves a similar result. Do you need to know all of the area for all "Meeting Rooms"? Do you need to introduce a new Dimension like "Room Type"; which may also include Room Name?
- Separate the dimension data into unique tables
    - **Always** make sure to **name Tables**. Avoid overlapping names between Tables and Worksheets, use a prefix or suffix to delineate one from the other. Example "tbl.Rooms" for the table and "Rooms" for the worksheet name.
    - As you create columns in tables, follow a naming convention for consistent column names across all tables.
        - Adding the table name to the ID column can help avoid confusion.
        - Dot "." Notation is a useful way to write complex names while avoiding hyphens, spaces or under-scores. For example: Room.ID
        - CamelCase in Column and Table names combined with Dot Notation can help improve legibility and readability.
        - Column and Table names should help provide clarity about the structure of the Data Model. These values do not have to be visible in "outputs" (Charts, Graphs, Etc.). For example, Power BI will allow you to rename all the column names to more appropriate user friendly names.

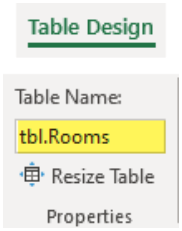


Figure 2 Excel Table Name

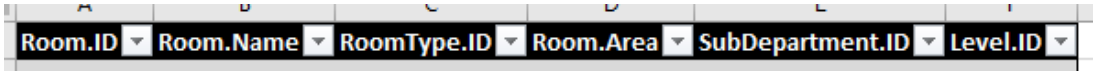


Figure 3 a sample of column names in a Fact table

[6] Not to confuse things further, sometimes a Dimension table may also be a Fact Table with its own star schema. For example a table "Buildings" could function as a Fact table with dimensions like "owner" and "type" and any other data that might be used to further describe or classify a building.

- Ensure that relationships between Fact Tables and Dimension Tables are maintained as you break out Dimensions:
  - Duplicate the Fact Table (or Reference[7] the query if working in Power Query) to create a new Dimension Table.
  - In the new Dimension table delete/remove unnecessary columns.
  - Format the data as a Table, and Name the table appropriately.
  - Use: Data -> Remove Duplicates to de-dupe the new Dimension Table

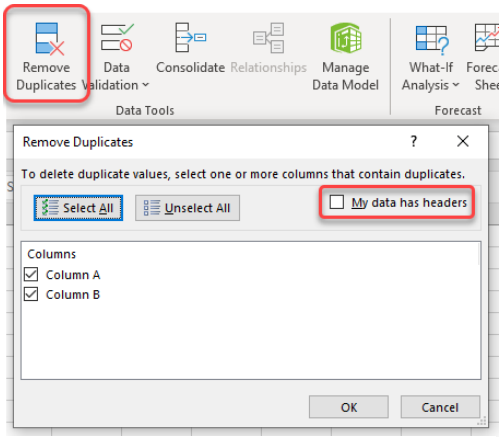


Figure 4 Removing Duplicate Values in Excel

  - Create an ID column and assign unique ID values
  - In the Fact Table use XLOOKUP (see aside) as a shortcut to write the new ID values from the new Dimension table to the Fact Table; Write the XLOOKUP formula, propagate to all rows, use Copy Values to Copy the values into the Table, delete original column with actual Dimension Value

The De-Duping process is where you will be able to detect/determine how consistent the data is and if there are typos, derivatives, or other deviations. For instance when dealing with Room Types you could encounter something like the table to the right.

Exam Room
Exam
Exam Rm
exam room

In this case you will need to determine if all the dimension valuess are the same and therefore should be consolidated. Consolidation can usually be best accomplished using Find/Replace. Large quantities of values can present more challenges in identifying overlap between dimension values and consolidation.

Dimension Tables may need to be further normalized:

- Ensuring that each Room has a relationship to a Level and each Level has a relationship to a Building.
- Data may be missing; for example determining the assigned Level.
  - The value of one dimension (Level) could be implied by the value in another column. Room Number (04100) may imply that the room is on a level (Level 4). Simple processing in Excel with the Left, Right or Mid functions allow for extraction of data such as the level value (key) from a room number.

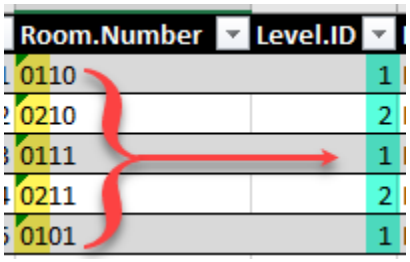


Figure 5 Extracting values

[7] "Reference" is a specific command in Power Query and an alternate to "Duplicate". Review more detailed information on Power Query to determine when to use Reference versus Dupilcate.

XLOOKUP

XLOOKUP is Microsoft's "new" feature meant to replace both the V and H look-up features. XLOOKUP can also replace Index & Match in some situations. XLOOKUP fully supports column references, which means it works very well with formatted tables.

Replacing Dimension Values with Dimension IDs:

When normalizing data it's typically preferable (though not required) to rely on a unique ID value to relate tables to each other and not rely on a human readable value (like a name, or description). Since de-normalized data will rarely have ID's that means as you normalize the data you'll need to define your own ID values, and then get the ID values into the appropriate tables in lieu of the original value (Name or such). Using a unique value gives you the ability to change the Name if needed (or add other "names") without having to worry about other tables becoming out-of-date.

This can all be quickly accomplished with XLOOKUP:

1. The generic XLOOKUP formula is:
   =XLOOKUP(
   <*Fact.TableName*>[@[<*Dimension.Value.ColumnName*>]],
   <*Dimension.TableName*>[<*Dimension.Value.ColumnName*>],
   <*DimensionTableName*>[<*Dimension.ID.Column*>],
   "Missing")
   - The "@" tells the formula to use the specific cell in the current row, as opposed to the entire column range.
   - "Missing" is a good troubleshooting technique. If there are any "Missing" values you know that something is not correct in the data table(s).
2. Write the formula in a new column in the Fact Spreadsheet, if the column is a Table Column the formula will automatically propagate for the entire table.
3. Select the entire column and Copy.
4. Choose Paste Special -> Values and paste the values into a second new column in the table named for the "ID" that is being defined.
5. Delete the formula column. Delete the original Dimension Values column in the Fact Table.*

* Or replace the values in the Dimension Value column with a similar XLOOKUP formula that looks-up based on the ID column and returns the Dimension Name Value. This reference can be handy when working in the table. See example spreadsheet for this in action.

Data Aggregation

Aggregation is the absolute key to successful data manipulation and reporting. For the type of high-level analysis typical of a master plan or early project planning it's unlikely that anyone wants to view reports, charts, graphs, diagrams, or visuals that have the granularity of individual rooms, suites, or even sub-departments. Executives and project leadership need high level summarization of more granular data. On the other hand granular data can help to ensure accuracy, particularly when people question the summarized values.

- Always work with the smallest granularity that you have data available for.
    - For example, if you are provided a spreadsheet that breaks down room types by quantities. Consider generating rows for each unique room instance (easily done using PowerQuery). Quantities or totals are aggregations of rows and by having each unique instance you will have more long-term flexibility in your data.
- Make sure that there are Dimensions available to aggregate the data appropriately. For example, if an executive wants to see total inpatient and outpatient areas by department then you need Department and Category (Inpatient/Outpatient, etc.) dimensions. You can then use Group By (in Power Query) to group and summarize the data by both Dimensions.
    - The approach of maintaining granularity allows the addition of new dimensions with relative ease. Either a completely new dimension can be added, or a new level inserted into an existing hierarchy/dimension.

Aggregating Data with Power Query & Excel

With data normalized and organized into a series of tables, Power Query can be used to aggregate data to provide summaries directly in Excel. At this stage your data is also prepared to be imported into Power BI which inherently aggregates data when building reports.

- Data first needs to be loaded into Power Query.
    - Select any cell in the target table (ex. tbl.Rooms).
    - Right click and choose “Get Data from Table/Range”

Power Query

Power Query has been in Excel in one form or another for quite awhile.

Power Query is a highly versatile tool for importing, manipulating and transforming data and is shared between both Excel and Power BI. While the two implementations are not identical, they are very similar and generally what you can do in one, you can do in the other; including often being able to copy/paste M-code (the underlying language of Power Query).

Detailed instructions on adopting and using Power Query is outside the scope of this document, some key pointers will be covered.

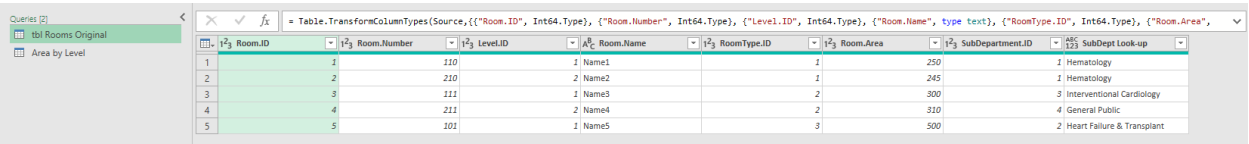


Figure 6 Example Room Table Query

- This will automatically load the Table into Power Query.
- Once in Power Query right click on the Query Listed to the left (the name will match the Table name) and choose “Reference”
  - Referencing the original query preserves it in its original state for future use for other operations.
  - Rename the original query to indicate that it is the original (ex. tbl Rooms Original).
  - If working in Power BI you may also want to set the original query to “not load” by right clicking on the query.
- Rename the new Referenced Query to reflect the intent, for example, “Area by Level”.
- Select the Column that represents how the data is to be grouped (in this example the Level.ID column), click on the “Group By” button in the Ribbon Transform panel.

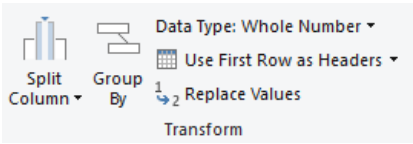


Figure 7 the Transform Panel from Power Query

- In the dialog that opens, the selected column should be selected in the drop-down. In the new Column Name field, enter “Area”, set the operation to “Sum” and choose the column for Area (for this example Room.Area), click Ok to finish the operation.

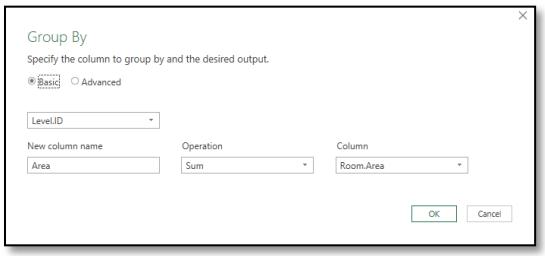


Figure 8 Group By dialog

- The Query will be modified to reflect the rows being grouped by Level and the Areas will be summed. Note in the far right the Applied Steps view, which is updated to show the step added “Grouped Rows”.
- Go to the “Close & Load” button in the top left. Click on the lower third to access the menu option and choose “Close & Load To…”, this is a more reliable option than the default, as it allows you to control what will happen with the queries that have been created.
  - In more complex operations there may be queries that you do not want to load into the actual Workbook.
- In the dialog that opens choose “Table” and “New Worksheet”.

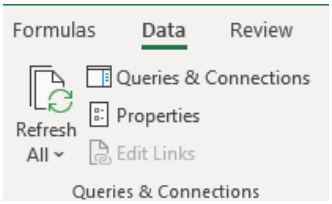


Figure 9 Power Query Refresh in Excel

The result is a new table and worksheet in the Workbook tied to the data in the data tables. By default tables generated by Power Query must be refreshed manually using the "Refresh" button. Power Query can be set-up to refresh based on various triggers. To verify that everything is working you can modify an Area value in the Fact Table and click on the refresh button to see the result.

Data Summary

Aggregations are the cornerstone of the work discussed in this handout and are the basis of being able to deliver good reports from the data. Taking the time to build a well-structured and normalized data source makes it much, much easier to engage a variety of tools for further analysis and reporting, be it Grasshopper, Power BI, Dynamo, Tableau or any other tool that is capable of consuming data. Whether the tool would prefer a data model (PowerBI) or a single aggregated table (Dynamo) the combination of the normalization and aggregation processes is key. Separating the data model from reporting and analysis also helps to ensure data consistency and can help to avoid data entry errors.

Learn More

To learn more about data models, star schemas, hierarchies and how to use Power Query, check-out these YouTube videos:

Still not sure what this talk about Facts, Dimensions and Normalization is all about, take a look at this course that was presented at a Microsoft conference:

Microsoft Power BI: The Do's and Don'ts of Power BI Relationships – BRK3019

Stuck with a big table of data (flat file) that you need to create a data model from, learn from the masters in the cube:

Guy in a Cube: Power BI Tutorial | From Flat File to Data Model

Grouping (aggregating) data in Power Query, and why it's more useful than Pivot Tables:

Grouping Data with Power Query

Slightly dated, still relevant. Power Query is located under the "Data" tab. "Newer" commands in Power Query should allow you to create the delimited column in lieu of hand coding M-Code.

Have multiple facts? Make sure you build a good model in Power BI:

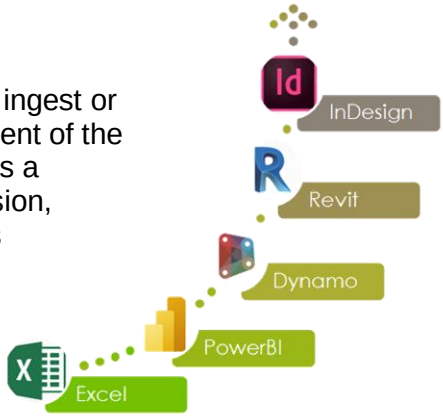
Guy in a Cube: Handling MULTIPLE fact tables in Power BI

General introduction to Power Query:
Note that this content is extracted from a longer paid course

Excel Power Query Course: Power Query Tutorial for Beginners

Data & Dynamo

Less = More



Many examples of visual scripting that ingest or use data often take on a large component of the data processing. In Dynamo this means a great deal of list management, conversion, filtering, etc. For a coding expert this is "easy" and all within a context they are familiar with, the drawback is that it locks the business logic of data manipulation and modeling into a format (a Dynamo graph in this case) which is generally not easily accessible to the majority of the team. This in effect turns the script into a magical black box, certain data goes in, and something else comes out. Validation and QA can only be performed on the final outcome; those with limited experience in coding may find it difficult to troubleshoot depending on the inputted data versus the generated results.

The processes of Data Normalization and Aggregation described in the first section of this document are the groundwork to avoiding the black box issue with scripting. In lieu of performing data manipulation in the script, data manipulation and modeling is carried out in data manipulation tools (in our case Excel and Power Query) that are more accessible to a broader audience. The specific data required to achieve a certain scripted result can be served to the coding platform of choice, this aids in making the scripts significantly simpler, and more agile. If the script only creates geometry based on inputs from Revit and Excel files respectively, then the script can be easily used on multiple projects as long as it is understood what the required columns in Excel are and what kind of Revit model is expected.

When coding to automate 3D modeling, the starting point should be establishing the desired end goals, followed by mapping out the logic required to meet them. Understanding objectives and goals allows the developer to structure the graph more efficiently. In this case the team had established that Dynamo would be used as the coding platform to automate 3D modeling of the data coming from Excel to visualize the program stacking and assist in diagram generation. As discussed earlier, given the requirements and the massive amount of data, the data incoming to Dynamo needed to be well organized in order to achieve the desired result.

ETL vs ELT

You might be saying "what's" the difference here? Why does it matter if I manipulate Data in Dynamo or in Excel? While on the surface it may seem very similar, philophsically it represents two different approaches.

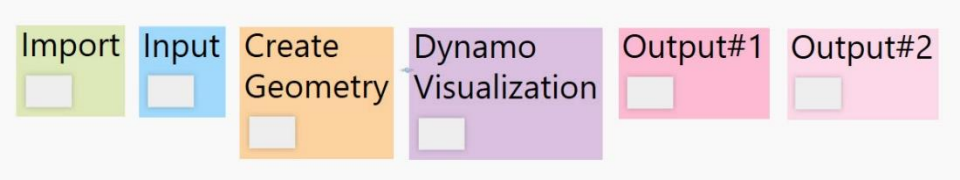
Loading data into Dynamo then transforming it would be "Extract, Load, Transform". You're taking data from a source, inserting it into Dynamo, then Transforming it.

What we are advocating for is Extract, Transform Load. Extract the data into Excel (really the same tool as the source in this case), Transform the data, then load into Dynamo.

ETL or ELT are appropriate in different places and in different scenarios. Our argument in this case is that Transforming (and enriching) data is best accomplished in a tool purpose-built for that task.

Script Strategy

Breaking the graph into key components is a helpful way to keep the graph organized and easily accessed by others. Understanding how the data will relate to the 3D model needs to be vetted out in order to optimize the workflow; the graph for this project was broken down as follows:



1. ***Import –*** Model Elements; objects supplying geometric data as the basis of new geometry creation.
    a. Levels
    b. Mass Floors
    c. Mass Families
    d. Family/Project/Shared Parameters
    e. Color Schemes & Filters
    f. Design Options
2. ***Input*** - Excel Data, simplifying data prior to bringing it into Dynamo minimizes errors and troubleshooting when it comes to pushing information into the Revit model. *Take only what you need! Less is More!*
    a. Unique identifier
    b. Percent of the gross floor plate.
    c. Usable Area
    d. Associated Building ID
    e. Associated Level ID
    f. Dimension Values; Area of Care, Category
    g. Total Gross Floor Plate Area
3. ***Creating Geometry –*** Transforming Data to 3D geometry, understanding the nature of the Revit model components that will host data (from Excel) and based upon imported geometry data to enable the fast production of stacking diagrams for the different scenarios.
4. ***Dynamo Visualization*** - A built-in feature within the graph to enable visualizing the output of the geometry quickly, prior to pushing it into Revit and its engine.
5. ***Output*** 1- 3D Geometry or Mass to represent data depending on the tool used Formit versus Revit.
6. ***Output 2***- Pushing data associated with geometry to the 3D model as Parameters.

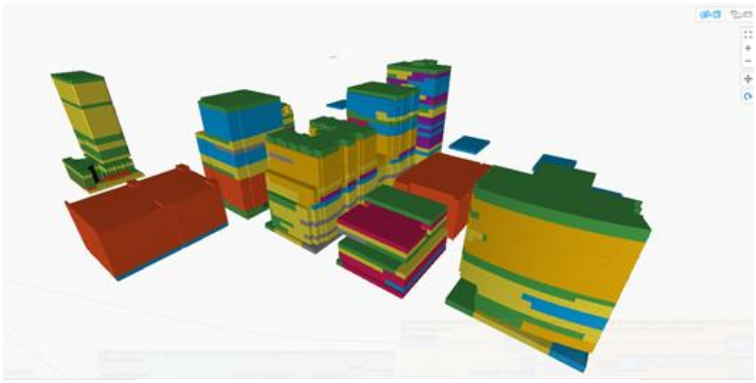


Figure 10 Pre-visualization in Dynamo

Process Example

In this case study, the primary objective was narrowed down to the creation of a three-dimensional blocking and stacking model to generate 2D and 3D images and diagrams for inclusion in static presentations and reports.

The grain of the data model was the Suites (collections of rooms) on each Floor Level in each Building. Project leadership wanted to see graphics that summarized the data at the level of what the client referred to as “Areas of Care,” for example, Oncology or Cardiology, etc. Leadership also wanted to see data summarized by what the team referred to as “Category”, for example; Inpatient, Outpatient, Diagnosis & Treatment, etc; as part of the data normalization exercise both dimensions were defined for each Suite.

As an example, to accomplish the desired result, it was neccessary to generate distinct pieces of geometry that represented:

- total area for Outpatient Cardiology on Floor 6 of Building 13
- total area for Inpatient Oncology on Floor 15 of Building 7.

To compound the challenge further, once the project moved from Existing Conditions validation to master planning, there would be multiple options for:

- the amount of program growth/change over time, 5, 10, 15 years.
- the distribution of the program over time to different buildings and levels.

Level of Detail

Just like in the geometry world, in the data world you can have “too much detail”. When you are combining both worlds, it can be worse.

The aggregated data in our case was being used to create geometry that would then be displayed in a variety of formats that would typically be at the scale of the entire campus or a large portion thereof.

This created a dilemma, what to do if, after aggregating the data there was a row that represented only a few hundred square feet? If geometry was generated for that row (a fractional percentage of the gross) it would at best appear as a “blob” or “graphic artifact” in the visual output.

It was determined that any summed area or percentage that fell below certain thresholds would be filtered out of the tables meant for Dynamo. This also meant that when calculating the percent of gross, the “missing” area has to be accounted for by weighting the percentages to ensure that the total percent still equaled 100%.

Revit Model Setup

To make it easy to manipulate and update individual buildings, the Project Model was composed of multiple Revit models:

- A context model for surrounding structures (purchased, converted and adjusted as required).
- Individual mass models for each building that are occupied by the client.
    - Breaking out each building as its own model makes it easy to manage unique levels per building as well as being able to: add “future” new buildings, have different options for new buildings and even propose modifications to existing buildings.

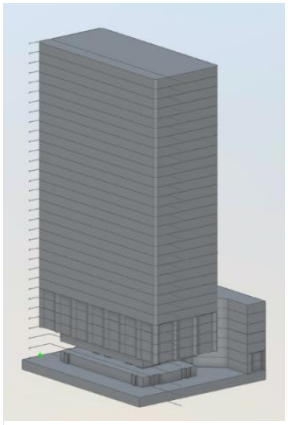


Figure 11 Example Building Mass Model

- An overall project model to link together the context and occupied buildings; this is the model where Dynamo generated and placed the geometric elements.

Unique IDs

Unique IDs are an important part of structuring data. Unique IDs give you the power of being able to identify a “thing” without having to use words or text to describe one differently from another.

When structuring raw data typically a sequential set of numbers is sufficient to be used as an ID or “Index”. When data is aggregated, particularly for the purposes of sending the data “out” (for example to Dynamo) unique ID’s can be trickier, particularly if you want to be able to refresh the data later on (the data may change).

Aggregations create “new” data and by it’s nature that data could vary as the underlying dataset is updated or modified. In this case the unique ID cannot be relied upon as a single unique sequential integer. Rather a unique ID needs to be constructed that will be “immutable”. That is the same ID will always be returned for the same set of conditions.

For example; if you Group by Building Level and Category, then a repeatable unique ID could be constructed by concatenating the ID values of Building Level and Category. Future changes or updates to the data will always result in unique ID’s that can be used to analyze changes or differences between what was and what is now. While the same set of conditions Building Level and Cateogriy will always result in the same ID value for a particular condition.

Dynamo Data Preparation

To develop the Dynamo script the key pieces of data that would be required to meet the objective need to be identified in order to avoid data transformation within the script itself.

The following data was determined to be required:

- Revit Model(s) (for each unique building):
    - Overall building envelope, simple mass model, with some envelope/fenestration articulation.
    - Unique Building ID – as specified by the data model
    - Unique Levels with Unique Names per building – as specified by the data model
- Data Table (Excel), for each row:
    - Unique identifier
    - Percent of the gross floor plate.
    - Usable Area
    - Associated Building ID
    - Associated Level ID
    - Dimension Values; Area of Care, Category
    - Total Gross Floor Plate Area
    - Other attributes/data to be written to Revit Parameters

The Revit models were fairly straight forward; each model name included the unique Building ID. Levels for each building were also uniquely named with the ID generated from the data model. The level name was a combination of the Level Identifier/Number and Building ID; for example, “16_14” would be Building 16 and the 14th Level as counted from the lowest level. Massing within the models was kept fairly rudimentary.

To generate the data for the script Power Query was used in an Excel file separate from the primary source Excel file to aggregate the granular data and calculate values as required. “Group By” was used to group individual Suites by unique building level then by dimension(s) as appropriate:

- Group By:
    - Level ID
    - Care Area
    - Category
- SUM Area
- Generate a unique ID for each row (required for dictionaries in Dynamo).

- Using “Merge Queries” additional data was attached as new columns to the Grouped Query:
    - Level Name
    - Building Name
    - Client Property Name
    - Building ID
    - Care Area Name
    - Category Name
    - Gross Area for the unique Level
    - Overall Building Gross (for tagging & scheduling in Revit)
- Add a custom column that calculated the percent of Gross Area that each aggregated row represented.

Multiple tables were generated to look at the data from different perspectives and generate different sets of geometry. Below is one example.

Space ID	Areas of Care.RevitGrouping	CategoryName	BldgID	Buildings.BuildingName	LevelID	LevelName
16	Support	Support	8	[redacted] Hospital	74	Lower Level
17	Support	Healthcare Support	8	[redacted] Hospital	74	Lower Level
18	Oncology	Outpatient	8	[redacted] Hospital	74	Lower Level
19	Other	Other	8	[redacted] Hospital	75	Level 1
20	Womens	Inpatient	8	[redacted] Hospital	75	Level 1

RevitLevel Name	GrossFloorArea	ProgramPerLevel	Area	Program Pct per Level
8_LL	65355	45635	2680	5.87%
8_LL	65355	45635	21403	46.90%
8_LL	65355	45635	9526	20.87%
8_1	45975	25537	16949	66.37%
8_1	45975	25537	8588	33.63%

Scripting

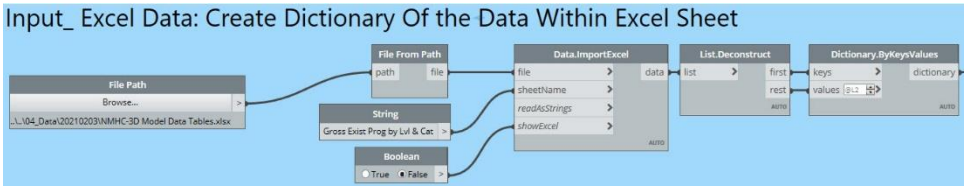
Sorting out the data for Dynamo was an important part of the effort. Just as important though was getting a working script that could process the data. The key objectives of the script were:

- Create geometry that represents a portion of a floor plate based on the specified percentage in Excel.
    - Leverage Revit mass models as the basis of the outline for geometry and vertical height (level).
- Process incoming data from Excel to determine the quantity and size of geometry pieces required per floor plate.
- Process incoming data from Excel to update Revit parameters.
- Ensure the script is easy to use and is re-usable.

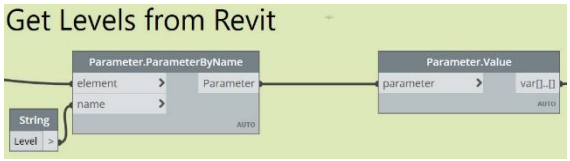
Key Script Components

To accomplish the target goals there were some specific lessons learned that can be shared:

- One key to a successful Dynamo graph that utilizes Excel data is to pull from a single data source and to structure the Dynamo List(s) in a way that allows users to carry data through the logic, limiting duplication and preventing errors.
    - In this case that goal was achieved at the very beginning through the development of a “Dictionary” in the Dynamo script.
    - A “Dictionary” is a one-stop-shop technique that ensures that the graph will reference a set of well-structured Lists to pull specific information for each individual space (row) within the Excel sheet and relate it directly to Revit components.
    - By making use of a Dictionary and having a unique ID established in the data source any column can be “looked-up” without significant List operations based on the ID value, much like XLOOKUP in Excel. Just like in Excel this helps to ensure data consistency and quality.
    - Using a dictionary makes it exceptionally easy to automate the population of family parameters with data that helped in generating the desired stacking diagrams. It enabled the designer to utilize filters and color schemes within Revit to show different stacking illustrations.



- In order to place the elements on the proper levels within Revit, consistency in naming between the data source (Excel) and Revit is a must!!
    - A dictionary is again useful to relate the Levels from the Revit source models to the Levels indicated in each row of the data source.
    - In this instance the team relied on imported Revit Mass Floors to host the geometry (Mass Families) representing spaces that were populated with additional data (Parameters) all created in Dynamo, as defined by the Excel data.



- Being mindful of the quantity of geometry to be produced in Dynamo and the subsequent duration required to create the Revit geometry; it was very important to maintain an efficient workflow.
    - To further simplify the scripting, the area for each space category was calculated (in Excel) as a percentage of the overall floor area, this meant that in Dynamo all that was required was to calculate the appropriate size block to match the percentage.

- It is good practice to author the script such that the geometry can be visualized in Dynamo prior to initiating the process of creating it in Revit.
- The ability to de-activate the Revit processes allows for quick troubleshooting within Dynamo without the overhead associated with a large number of Revit processes.
- This allows for the investigation of the Dynamo results for errors or nulls, rather than waiting for a longer period to realize that that something is not displaying correctly in Revit; in this case study it took Revit about 25 minutes to generate all of the geometry for all the space categories within all buildings on the site along with all of their parameters.

Location

A very important caveat to the work carried out on this project has to do with “where” program is.

In the case of this project we were not concerned about being able to indicate that the Onolcogy department is in the southwest corner of the Hospital Building. The major concern was to be able to say that Oncology consumed 45% of Level 10 in the Hospital Building.

This is a very important detail; accounting for “where” program was physically located was limited to Building & Floor Level.

If a higher degree of accuracy for location were required it would have singnificantly increased the complexity of the effort and would have required a more detailed geometry model to pair with the data analysis and aggregation efforts.

Illustrating Data

A picture is worth one thousand words

That last piece of the effort are the results that can help drive high level, data-based decisions. Illustrating and diagraming the data in a useful manner where stakeholders can look at the results and draw reasonable conclusions or make informed choices.

Telling a story

Tables of data are quite boring and rather dry for most people. To help the client understand the scope of their footprint across all of their buildings and Areas of Care it was important to find ways to tell compelling stories from the data that would help reveal important relationships or simply help the consumer compare and contrast significant parts of the program. Data validation by the client was also an important story that needed to be told. Data visuals helped to aggregate the detailed data into something the client could navigate as part of their review process.

Power BI

For the purposes of data validation and review in virtual meetings, the Power BI visuals developed from the data model were useful for early data interrogation and review, followed by using Excel exports to document specific feedback and requested changes. For data visualization a variety of traditional visuals; like bar charts and tree maps as well as some unique visuals or unique uses of visuals were used.

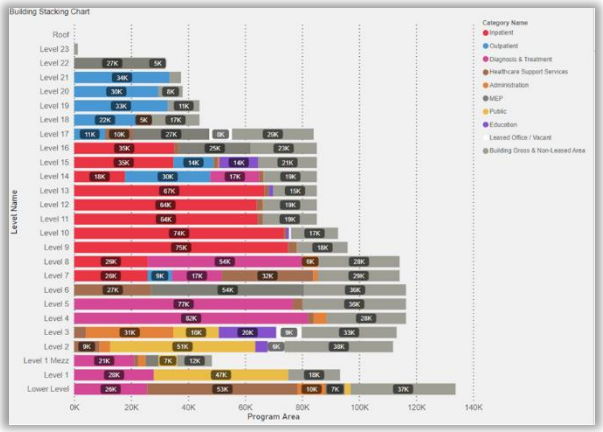


Figure 12 Building stacking can encompasse one or all buildings.

- Horizontal bar charts (Figure 12) make very good stacking diagrams for buildings as long as an architectural illustration of the building form is not required.
- Sankey diagrams (Figure 13) can also be useful visuals for illustrating how one group or category (Departments for instance) are spread between multiple physical locations, and what other groups are co-located in the same buildings.

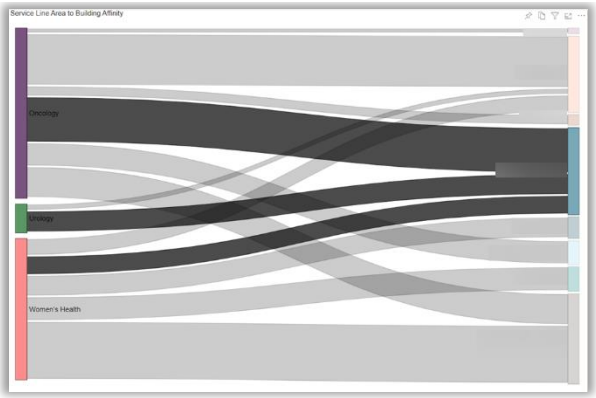


Figure 13 Sankey Diagrams can illustrate relationships. Groups on left, Buildings on right.

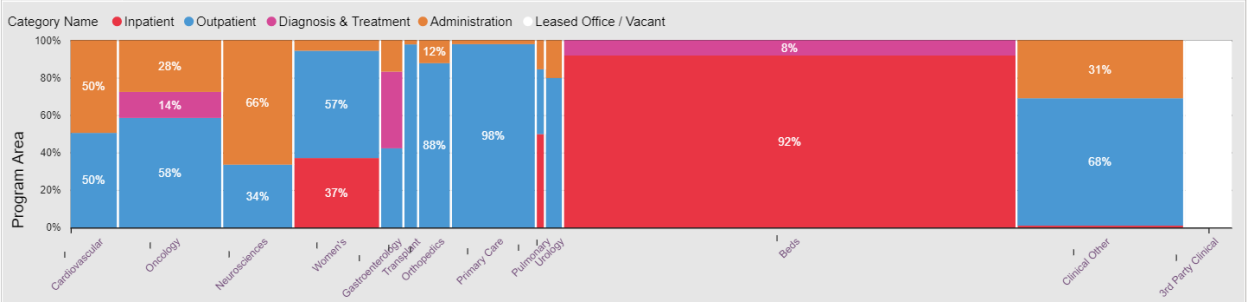


Figure 14 Mekko charts are stacked column charts with column width showing a proportional relationship.

- Mekko charts (Figure 13) are like advanced column charts, adding a third dimension in the form of the width of the column in addition to the typical X and Y axis and the break-down (stacking) of each column.

- Chord charts (Figure 15) can illustrate how groups migrate or move. Unfortunately since the team hasn‘t done any masterplanning yet, there is no decanting to illustrate.

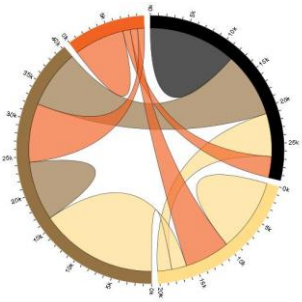


Figure 15 Chord Charts can show change.

Revit & Adobe

While Power BI is a powerful data visualization tool, architects and people that hire architects still want to (and need to) see data visualized in the context of the physical environment we inhabit, know, and are making decisions about. Architectural illustration has a long history, the question was what was the best way to use the data to produce compelling visualizations that could be generated without significant amounts of manual modeling and data entry. We all know the various products quite well, and there exists a vast amount of documentation on how to use any of the tools used for generating graphics. The question for this team was what was the best approach for this project to generate color coded 3D views, 2D campus sections, key plans and campus wide plans.

Originally the project team said they wanted a “3D model they could navigate with the client on any device”. For the purposes of documenting existing conditions this morphed into “2D and 3D illustrations embedded in Power Point or PDF”.

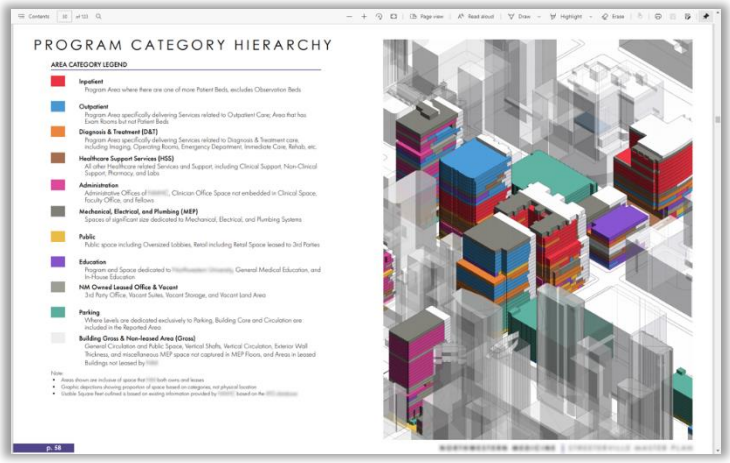


Figure 16 Cover page to the report section with diagrams

3D images of blocking and stacking look nice and make for nice accent graphics, the reality is that projection views (plan or section) make for a much more easily understood diagram depicting the major areas of program.

In that regard, Revit was a powerful and flexible tool that made it easy to create compelling images. The Dynamo script generated geometry with embedded data, the appearance of the geometry could then be manipulated:

- View Filters could be used to control visibility and apply color coding based on the embedded data.
- Tags could be quickly applied to report information that could not be conveyed with color/pattern.
- The support for jogging sections made it possible to easily create campus sections that highlighted critical buildings.
- View Templates made it easy to repeat and re-use filters to quickly generate multiple views that were consistent in appearance.
- Elements could be scheduled to produce summary tables.
    - It is important to note that these summary tables would not agree 100% with similar summaries in Power BI due to the data filtering applied to the aggregations for the Dynamo Script.
- Additional illustrative elements (city skyline, text, callouts, etc.) could be easily added or modified in the Revit or Adobe environment.
- Tools like plan regions could be used to “unify” floor levels even when levels did not align building to building.
- Revit Worksharing & BIM 360 provided for multiple users to work on the same model from multiple locations.
- BIM 360 still allowed basic model viewing without requiring the authoring application.

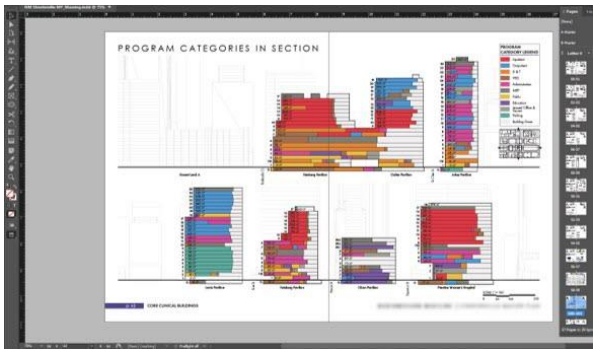


Figure 17 Campus Section in Adobe Post Production

Figure 18 Sheet for Specific Area of Care

Sheets and views were set-up in Revit following typical processes and procedures. Images were then exported from Revit for final post-processing in Adobe products to ensure text and graphic consistency with the other portions of the Existing Conditions Report.

Power BI Paginated Reports

While Power BI (the service) provided a good way for the immediate project team and client representatives to navigate and review all of the data, there was still a requirement to include summary data tables in the primary report as well as more detailed tabular breakdowns in attached appendices. Power BI itself does not export well to static formats, particularly for longer tables that need to break over multiple pages.

Since the project already had a well-organized Power BI Dataset it made sense to adopt Power BI Paginated Reports to generate well-formatted data tables that could be exported to PDF (and run on demand by the end users).

Benefits of Paginated Reports:

- Formerly SQL Server Reporting Services (SSRS) Report Builder; same application updated to work with Power BI.
    - Lots of documentation, online material, and expertise with SSRS Report Builder.
- Originally intended for generating reports from SQL Servers, can include:
    - Tables
    - Matrixes
    - Charts & Graphs
- Support for repeating formatting, embedding reports in other reports.
- All formatting, text, color fills, and line styles can be controlled through expressions that reference the data (conditional formatting).

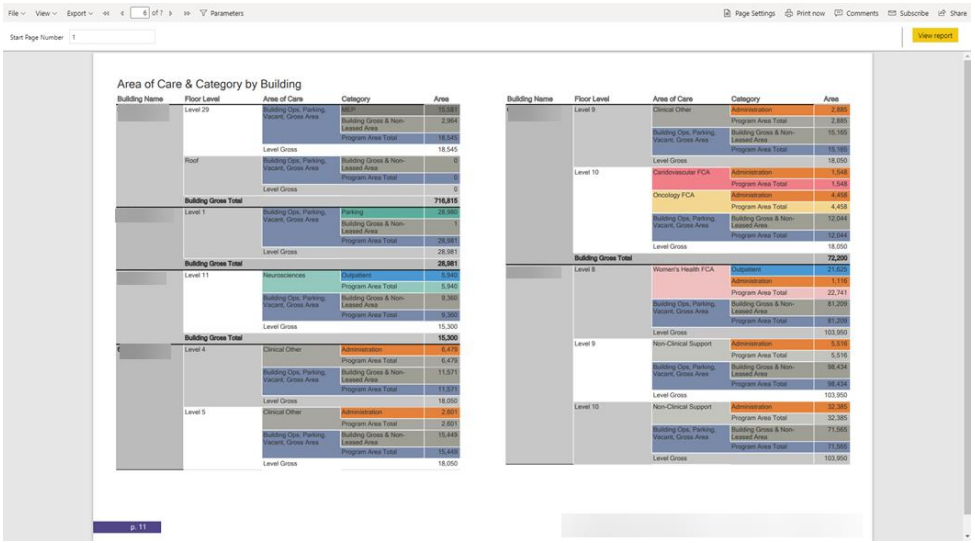


Figure 19 Fully formatted Paginated Report using Matrix and Conditoinal Formatting

- Reports can be hosted and run by end users on demand from the Power BI service.
- Exports to PDF, Excel, and other formats.

The result from Paginated Reports was more than satisfactory. Custom formatting options in Paginated Reports allowed the development of a generic page template to match the project team's overall report design as well as to set custom start page numbers, so that the pages could slide seamlessly into the larger Master Plan report.

Paginated reports is not without its own learning curve, for relatively simple summary tables and matrices, it is not to difficult to set up. More complex reports, such as Room Data Sheets, would incur more significant development time. All color data in the example was driven by the report data and was set to match the colors used in the Revit models. Unfortunately, unlike Paginated Reports or even Revit, Power BI itself does not support nearly as much data driven conditional formatting.

Scenario Planning

To date the project has progressed to finalizing a report that lays out existing conditions. In parallel to the development of that report the team executed a short charrette focused on high-level planning scenarios based on growth strategy information provided by the client.

In conjunction with the client, the team identified three possible sites and the team agreed to develop three different building size concepts for three different options: all outpatient, all inpatient, or mix of inpatient and outpatient.

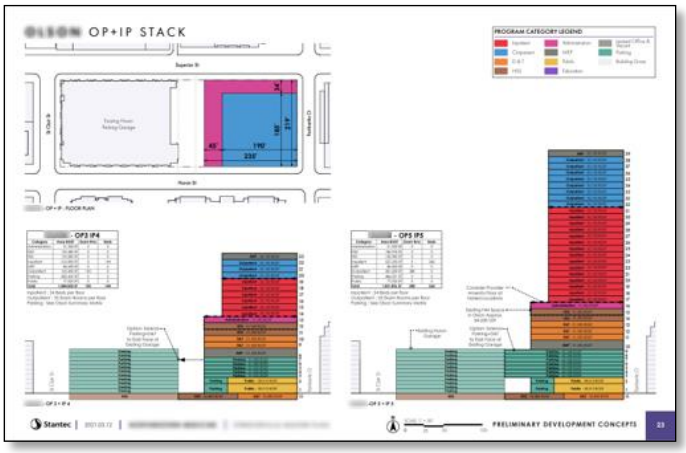


Figure 20 Example of one of the 27 scenarios

3 sites * 3 sizes * 3 options = 27 design options/schemes

Fortunately, the options only needed to be developed as a very basic overall blocking and stacking scheme, with no specific departments or program assigned beyond the three combinations of care services and placeholders for additional supporting program. All options at a minimum required basic exploration before whittling the twenty-seven down to a more reasonable quantity for final presentation to the client.

While this type of scheme development was not anticipated as part of master planning, the two week design charette still helps to showcase the versatility of the script and data model that was developed. The same script was quickly re-used with entirely different data in Excel to quickly build the twenty-seven different models based on three primary envelope configurations for

each site. The envelope options changed only slightly based on the proposed program size. Different data was quickly inputted into an Excel sheet with the same column names (critical for Dynamo) and run against the new envelopes. With the use of Revit View Templates a single architectural designer was able to work with a senior principal and a senior health-care planner to develop and illustrate all of the options, and fine tune as required for client presentation. From a time perspective the Excel and Script file certainly helped the process go faster, but more importantly, the script and use of data helped to ensure consistency and quality across all of the options being developed, which may be even more valuable than reducing the overall amount of time it takes to execute.

Concluding Thoughts

This project has helped to illustrate internally, and now externally, the value and validity of taking the time to develop a robust data model that lives outside of a particular design application, so that it can be easily used by multiple applications.

- The client originally handed over an Excel spreadsheet of 300 rows. The current data model has 1,000 rows in the primary Fact Table. A 300% plus increase in data is a strong argument for taking the time to properly structure and validate data.
- The use of Power BI & Excel helped to expedite data validation.
- Data validation then allowed the production of rich graphics from the vetted data source.
- The high-level scenario planning further validated our approach in Dynamo; that moving data modeling and transformation into Excel allowed for a more simplified script that was easily and quickly re-purposed.
- While this class has focused on the Existing Conditions portion, the Data model itself was developed with the goal of scenario planning for the long term. Simply put addressing actual programming would be a second class unto itself, when you dig into it while the basics remain true, scenario planning involves two more data models (new program and program distribution) as well as the complexity of multiple options, all changing over time, so the data is no longer static.
- Prior to the project pausing, we had begun analysis of patient and procedure volumes. Thanks to a robust vetted data source that had the grain of “Suites,” the financial/billing data was able to be married to the physical property data.
- The approach taken in this case sets the stage for employing tools like Autodesk Generative Design, or more generically “optioneering.” While not used on this project, it is conceivable that optioneering could have be combined with the scenario planning to more quickly generate significantly more options than the twenty-seven that the team generated. Those options likely would have been able to explore a larger variety of forms, based on the same functional program scenarios.