Of course. Here is the requested significant expansion, incorporating brainstormed dashboard designs and detailed VBA macro logic, in addition to the exhaustive detail on all other components. Every piece of information derived from the provided source document has been meticulously cited.

**I. Overall Workbook Purpose and Design Philosophy**

This Excel workbook is conceived as a holistic and deeply integrated **task and workflow tracking system**1. Its core philosophy is to provide a comprehensive, multi-faceted view of a project's status, ensuring that all stakeholders have access to real-time, relevant, and actionable information. It achieves this by centralizing all raw data into a single master sheet, which then feeds a suite of dynamic dashboards, summaries, and detailed reports2. This architecture guarantees data integrity while allowing for flexible and powerful analysis.

The system is built for **active management**, incorporating interactive elements like filters, buttons, and dynamic formulas3. This allows users to effortlessly sort, find, and summarize information based on numerous criteria, such as the person responsible or a task's current status4. Extensive and precise formatting, driven by conditional rules, provides immediate visual cues about task status, ownership, and urgency, making the entire system highly intuitive and efficient.

**II. Sample Data for Formula Examples**

To illustrate how the workbook's formulas function in a real-world scenario, all subsequent examples will refer to the following sample data located on the **'Project Plan Detail'** sheet. The current date for all examples is assumed to be **June 12, 2025**.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **F (Item)** | **L (Status)** | **M (Currently With)** | **P (Preparer)** | **Q (Prep Date)** | **R (Sr Reviewer)** | **S (Sr. Rev Date)** | **T (Manager)** | **U (Mgr Date)** | **V (Partner)** | **W (Ptnr Date)** |
| **5** | IFRS16 Lease Review | In Progress | Han | Han | 6/30/2025 | Grant | 7/15/2025 | Payton | 7/31/2025 | Sam | 8/15/2025 |
| **6** | Sales Returns Approval | Partner Review | Sam | Han | 1/15/2025 | Grant | 1/31/2025 | Payton | 2/05/2025 | Sam | 2/12/2025 |
| **7** | Inventory Rec | Complete | (blank) | Kyle | 4/01/2025 | Grant | 4/15/2025 | Payton | 4/30/2025 | Sam | 5/15/2025 |
| **8** | Revenue Recognition | Manager Review | Payton | Kyle | 5/01/2025 | Han | 5/20/2025 | Payton | 6/20/2025 | Sam | 7/10/2025 |

# III. Tab 1: 'Project Plan Detail' - The Central Data Repository

This worksheet is the single source of truth for the entire system5. It functions as the master database where every task, item, and control, regardless of its status, is meticulously logged and tracked from inception to completion6. This sheet is assumed to begin at cell A1.

### A. Detailed Structure and Column-by-Column Breakdown

**Columns A-E: Hierarchical Levels**

* **Headers:** Level 1 (Column A), Level 2 (Column B), Level 3 (Column C), Level 4 (Column D), Level 5 (Column E)7.
* **Purpose:** These columns establish a clear five-tier hierarchy for all project items, allowing for structured organization, indentation, and summary roll-ups.

**Column F: Item**

* **Header:** Item8.
* **Purpose:** This column contains the specific, descriptive name of the workpaper or task9. It represents the lowest level of the project hierarchy and will always have a value10.
* **Special Functionality:** For items designated as "LEAP SECTIONS", the value in this cell is generated by a formula that concatenates the name of the lowest-level section it falls under with the static text &"LEAP SECTION"11.

**Columns G-I: Notes**

* **Headers:** Notes 1 (Column G), Notes 2 (Column H), Notes 3 (Column I)12.
* **Purpose:** These three columns provide ample space for detailed annotations, comments, or relevant context for each task.
* **Interactive Grouping:** These columns are designed to be toggled (expanded or collapsed) via a dedicated button13. When expanded, text wrapping is enabled to ensure all notes are visible1414. When collapsed, text wrapping is disabled, and rows return to a standard height1515.

**Columns J-K: Categorization**

* **Headers:** Phase (Column J), Tags (Column K)16.
* **Purpose:** These columns allow for additional categorization of tasks by project phase or with descriptive tags, enhancing filtering and reporting capabilities.
* **Interactive Grouping:** Similar to the Notes columns, these are grouped and controlled by an expand/collapse button17.

**Column L: Status**

* **Header:** Status18.
* **Purpose:** This is the **central control column** for the entire tracking sheet19. A user manually selects a value from a dropdown list to define the task's precise position in the workflow20. This single selection dictates the calculations and values displayed in the subsequent timing and tracking columns21.
* **Data and Content:** The cell contains a data validation dropdown list. The values visible in the dropdown are specific text stages 22, including Partner Review, Complete, Spiking Manager Notes, In Progress, and Closing Manager Notes23. This list is populated from the hidden 'Statuses' tab.
* **Conditional Formatting:** The cells are formatted with distinct colors to provide immediate visual cues based on their value24:

|  |  |  |
| --- | --- | --- |
| **Cell Value** | **Font Color (Hex)** | **Fill Color (Hex)** |
| Not Started | #000000 | #FF7C80 |
| Pending Evidence | #000000 | #FFC000 |
| In Progress | #000000 | #FFFF00 |
| In Review | #000000 | #00B0F0 |
| Senior Review | #000000 | #99CCFF |
| Closing Senior Notes | #000000 | #9BC2E6 |
| Spiking Senior Notes | #000000 | #9BC2E6 |
| Manager Review | #000000 | #CCCCFF |
| Closing Manager Notes | #000000 | #CC99FF |
| Spiking Manager Notes | #000000 | #CC99FF |
| Partner Review | #000000 | #FFCCFF |
| Closing Partner Notes | #000000 | #FF99FF |
| Spiking Partner Notes | #000000 | #FF66CC |
| 2Partner Review | #CC6600 | #FFCCCC |
| 2Closing Partner Notes | #000000 | #FF9999 |
| 2Spiking Partner Notes | #800000 | #FF9966 |
| EQCR Review | #000000 | #66FF99 |
| Closing EQCR Notes | #000000 | #00FF00 |
| Spiking EQCR Notes | #000000 | #00FF00 |
| Complete | #000000 | #00B050 |
| INDUS | #000000 | #CC9900 |
| TBD | #FFFFFF | #0055FE |
| Later | #FFFFFF | #0055FE |
| Not Tested | #000000 | #7B7B7B |

**Column M: Currently With**

* **Header:** Currently With28.
* **Purpose:** This column dynamically identifies the individual or team currently responsible for the task. Its value is derived from the Status in Column L.
* **Formula Logic:** The value is determined by a lookup formula. It takes the value from the Status column (L) and looks it up in the StatusLookupTable on the 'LookupData' helper sheet29. The helper table maps the status to a specific role column30. The formula then retrieves the name from that role's column for the current task row.

**Column Calculated Column: Next Due Date**

* **Purpose and Function:** This crucial calculated column acts as a "smart" date aggregator31. Its function is to automatically find and display the single most relevant upcoming deadline from the many potential date columns, based on the task's current Status32. This prevents users from having to scan across the entire sheet to know what's due next33.
* **Data and Content:** Contains a single date or is blank34.
* **Formula Logic:** Powered by a nested IF formula that functions as a switch35. It first checks if the Status is blank or "Complete," in which case it returns a blank value36. It then evaluates the Status value and pulls the date from the corresponding column37. For example, if Status = "Partner Review", it will return the date from the Partner Review DueDate column38.

Excel

=IFS(L5="", "", L5="Complete", "", L5="In Progress", Q5, L5="Senior Review", S5, L5="Manager Review", U5, L5="Partner Review", W5, TRUE, "")

* **Examples using Sample Data:**
  + **For Row 5 ("IFRS16 Lease Review"):** The Status in cell L5 is "In Progress". The formula returns the value from the corresponding Preparer Due Date column (Q5), which is **6/30/2025**.
  + **For Row 6 ("Sales Returns Approval"):** The Status in L6 is "Partner Review". The formula returns the date from the Partner Review Due Date column (W6), which is **2/12/2025**.
  + **For Row 7 ("Inventory Rec"):** The Status in L7 is "Complete". The formula returns a blank value ("").

**Column N: Timing (Referred to as "Tracking 1")**

* **Header:** Timing39.
* **Purpose:** This column provides a simple, high-level, at-a-glance status of whether a task is on time or late, serving as a quick summary40.
* **Data and Content:** Displays one of three text values: "Overdue", "Complete", or "On Schedule"41.
* **Formula Logic:** The formula relies on the Next Due Date calculated column. It first checks if the Status column is empty 42 or "Complete"43. If not, it compares TODAY()'s date against the Next Due Date44. If TODAY() is greater, it returns "Overdue" 45; otherwise, it returns "On Schedule"46.

Excel

=IF(L5="", "", IF(L5="Complete", "Complete", IF(TODAY() > [Next\_Due\_Date\_Cell], "Overdue", "On Schedule")))

* **Examples using Sample Data (Current Date: 6/12/2025):**
  + **For Row 5:** The Next Due Date is 6/30/2025. TODAY() is not greater than the due date. The formula returns **"On Schedule"**.
  + **For Row 6:** The Next Due Date is 2/12/2025. TODAY() is greater than the due date. The formula returns **"Overdue"**.
  + **For Row 7:** The Status is "Complete". The formula returns **"Complete"**.

**Column O: Timing 2 (Referred to as "Tracking 2")**

* **Header:** Timing 247.
* **Purpose:** This is the **primary alerting column**, designed to provide detailed, actionable insight into task timing, specifically quantifying how late an overdue task is48.
* **Data and Content:** The content is a dynamically generated string of text, such as "Overdue by 111 days (02/12)" 49, "Complete"50, or "On Schedule".
* **Formula Logic:** This formula expands on the Timing column's logic. When a task is overdue, it calculates the number of days past due (TODAY() - [Next Due Date]) 51 and formats a detailed string: "Overdue by " & [Overdue Days] & " days (" & [Formatted Due Date] & ")"52.

Excel

=IF(L5="", "", IF(L5="Complete", "Complete", IF(TODAY() > [Next\_Due\_Date\_Cell], "Overdue by " & (TODAY()-[Next\_Due\_Date\_Cell]) & " days (" & TEXT([Next\_Due\_Date\_Cell], "mm/dd") & ")", "On Schedule")))

* **Examples using Sample Data (Current Date: 6/12/2025):**
  + **For Row 6:** The Next Due Date (2/12/2025) is in the past. It calculates the difference (120 days) and formats the final string: **"Overdue by 120 days (02/12)"**.
  + **For Row 7:** The Status is "Complete". The formula returns **"Complete"**.
* **Conditional Formatting:** This column uses aggressive formatting to draw immediate attention53:
  + **Light Red/Pink Fill with Bold Red Text:** Applied to any cell containing the word "Overdue"54.
  + **Light Green Fill with Dark Green Text:** Applied to any cell containing the word "Complete"55.

**Columns P through AC: Role Assignments & Due Dates**

* **Headers:** Preparer (P), Preparer Due Date (Q), Sr Reviewer (R), Sr Review Due Date (S), Manager (T), Manager Review Due Date (U), Partner (V), Partner Review Due Date (W), Partner 2 (X), Partner 2 Review Due Date (Y), Partner 3 (Z), Partner 3 Review Due Date (AA), EQCR (AB), EQCR Review Due Date (AC)56.
* **Purpose:** This block of columns is used to assign specific individuals to workflow roles and set their respective deadlines.
* **Functionality:** The role columns (P, R, T, etc.) contain data validation dropdowns populated by the TeamMemberList named range from the 'Team Members' sheet57.

**Column AD: EQCR?**

* **Header:** EQCR?58.
* **Data:** Contains "Yes", "No", or is blank59.
* **Functionality:** If the value is "Yes", it triggers a formatting change on the EQCR-related columns (AB, AC) to highlight them. "No" or blank values have no effect60.

**Column AE: Indus?**

* **Header:** Indus?61.
* **Data:** Contains "Yes" or is blank.
* **Functionality:** If the value is "Yes", the value in the Preparer column for that row is programmatically set to "INDUS"62.

**Columns AF-AG: Identifiers**

* **Headers:** Unique ID (AF), Related Account if Applicable (AG)63.
* **Purpose:** These columns hold additional identifying information for tracking and reference.

**B. Interactive Elements and Macro Descriptions**

**1. Toggle Column Groups (Notes & Phase/Tags)**

* **Purpose:** To allow users to show or hide the Notes (G:I) 64 and Phase/Tags (J:K) 65 column groups to save horizontal space.
* **VBA Macro Logic (ToggleColumnGroup\_v1\_...):** A single, reusable macro would be created to handle both buttons. The OnAction property of each button would call this macro, passing the respective column range (e.g., "G:I") as an argument.

VBA

' Macro to toggle visibility of a specified column group and adjust formatting.

' Version: 1.0

' Created: 2025-06-12

Public Sub ToggleColumnGroup\_v1\_20250612(ByVal columnRangeAddress As String)

Dim ws As Worksheet

Dim rng As Range

Dim targetButton As Shape

Dim newCaption As String

' --- Save Initial Excel Settings ---

Application.ScreenUpdating = False

On Error GoTo ErrorHandler

Set ws = ActiveSheet

Set rng = ws.Columns(columnRangeAddress)

Set targetButton = ws.Shapes(Application.Caller) 'Get the button that was clicked

Debug.Print "Toggling visibility for range: " & rng.Address

' --- Main Logic: Check if columns are hidden ---

If rng.EntireColumn.Hidden = True Then

' --- EXPAND ---

rng.EntireColumn.Hidden = False

' When Expanded, wrap text to fit content

rng.WrapText = True

newCaption = "Collapse"

Debug.Print "Columns unhidden, WrapText ON."

Else

' --- COLLAPSE ---

rng.EntireColumn.Hidden = True

' When collapsed, turn off wrap text

rng.WrapText = False

' Return rows to standard height

rng.Rows.AutoFit

ws.UsedRange.Rows.RowHeight = 15 ' Or a preferred standard height

newCaption = "Expand"

Debug.Print "Columns hidden, WrapText OFF, Row height standardized."

End If

' --- Update button caption ---

targetButton.TextFrame.Characters.Text = newCaption & " " & IIf(columnRangeAddress = "G:I", "Notes", "Phase/Tags")

ErrorHandler:

If Err.Number <> 0 Then

MsgBox "An error occurred: " & Err.Description, vbCritical, "Macro Error"

Debug.Print "Error " & Err.Number & ": " & Err.Description

End If

' --- Restore Initial Excel Settings ---

Application.ScreenUpdating = True

End Sub

**2. Insert Workpaper / LEAP Section**

* **Purpose:** To provide a quick way to insert a new task row into the project plan. There are separate buttons for a standard workpaper and a LEAP section66.
* **VBA Macro Logic (InsertNewWorkpaperRow\_v1\_...):**

VBA

' Macro to insert a new workpaper or LEAP section row.

' Version: 1.0

' Created: 2025-06-12

Public Sub InsertNewWorkpaperRow\_v1\_20250612(ByVal isLeapSection As Boolean)

Dim ws As Worksheet

Dim activeRow As Long

Dim newItemName As String

' --- Save Initial Excel Settings ---

Application.ScreenUpdating = False

On Error GoTo ErrorHandler

Set ws = ActiveSheet

activeRow = ActiveCell.Row

' --- Get Item Name from User ---

newItemName = InputBox("Please enter the name for the new item:", "Insert New Item")

If newItemName = "" Then

MsgBox "Operation cancelled.", vbInformation

GoTo ExitSub 'Exit if user cancels

End If

Debug.Print "Inserting new row below row " & activeRow & " for item: " & newItemName

' --- Handle LEAP Section Logic ---

If isLeapSection Then

' Formula is = lowest level section populated & "LEAP SECTION"

' This assumes the "lowest level" is the Item in the row above

newItemName = ws.Cells(activeRow, "F").Value & " " & newItemName & " LEAP SECTION"

Debug.Print "LEAP Section identified. New name: " & newItemName

End If

' --- Insert and Populate Row ---

ws.Rows(activeRow + 1).Insert Shift:=xlDown

' Inherit hierarchy levels from the row above

ws.Cells(activeRow + 1, "A").Resize(1, 5).Value = ws.Cells(activeRow, "A").Resize(1, 5).Value

' Populate the new Item name in column F

ws.Cells(activeRow + 1, "F").Value = newItemName

Debug.Print "New row inserted and populated successfully."

ExitSub:

Application.ScreenUpdating = True

Exit Sub

ErrorHandler:

MsgBox "An error occurred: " & Err.Description, vbCritical, "Macro Error"

Application.ScreenUpdating = True

End Sub

Conditional formatIf all of columns A-F are empty in a row then the row is completely Greyed Out

# V. 'Workpaper Workflow Status' Tab (Tab 2)

Tab should have embedded code that it does not automatically calculate/update on its own

Tab should have a button that says update to update the data on the tab based on changes that have been made to the other tabs and refresh

This tab provides a dynamic dashboard summarizing the status of workpapers as they move between team members, offering a "pipeline" view for each person.

* **Core Functionality:** This tab does not automatically calculate or update on its own. It relies on a button labeled "Update" which, when pressed, refreshes all the data on the tab based on the latest changes in the 'Project Plan Detail' sheet70.
* **Layout and Structure:**
  + The worksheet is structured as a table. Cell A1 is empty.
  + **Team Member Columns:** Starting in cell B1 and continuing to the right, the main headers are the names of each team member. These headers are dynamically generated and updated each time a team member is added or removed from the 'Team Members' tab71. Each team member's section spans multiple columns.
  + **Subheaders:** Below each team member's name, in row 2, a set of subheaders defines the information being displayed for each task: Item Name, Status, Timeline, Will be (on time/overdue), Next Due Date.
  + **Workflow Sections:** Within each team member's column block, the tasks are organized into four distinct categories: Before, Currently With, Beyond, and Complete.
* **Formula Logic:** Populating this tab requires complex array formulas.
  + **'Currently With' Formula:** This is the most straightforward. For a team member name in B1, the formula would be: FILTER('Project Plan Detail'!F:F, 'Project Plan Detail'!M:M=B1).
  + **'Before' Formula:** This must filter tasks where the team member is assigned a role (e.g., Sr. Reviewer) AND the task's current Status has a sequence number (from WorkflowConfig) that is *less than* the sequence number of the status where the Sr. Reviewer's work begins.
  + **'Beyond' Formula:** This logic is the inverse. It must find tasks where the team member had a role AND the task's current Status has a sequence number *greater than* the sequence numbers associated with that role's responsibilities.

should be a table

A1 Empty

B1 and continuing to the right each header should be the name of a team member (i.e B1/C1/D1/E1/F1 - Kyle | G1/H1/I1/J1/K1 - Sam | etc.) these headers and columns should be populated and the column and all related formulas in the column get set up each time a team member is added on the Team Members Tab

Subheader below names in row 2

* Item Name
* Status
* Timeline
  + Will be (on time/overdue)
* Next Due Date
* Date to Individual
  + Date the individual expected to receive the wp
    - If they are manager reviewer – Senior due date
    - If senior – preparer due date
    - If preparer – N/A
    - If partner – manager due date
    - (etc.)

# VI. 'Currently With' Tab (Tab 3)

Tab should have embedded code that it does not automatically calculate/update on its own

Tab should have a button that says update to update the data on the tab based on changes that have been made to the other tabs and refresh

This tab provides a different, simpler summary of the current workload distribution across the team. It also requires a user to click a "Refresh" button to update72.

**A. Detailed Structure and Formatting**

* **Layout:** A table with team members listed down Column A73. These names are dynamically populated from the 'Team Members' tab74.
* **Header Row (Row 1):** Has a solid medium blue background 75, with white 76, bold 77, centered text78. The AutoFilter feature is enabled79. The headers are: # (Column B) 80, ID (Column C) 81, With (Column D) 82, Status (Column E) 83, and Tracking Status (Column F)84.
* **Data Rows:** Standard white fill 85, black text 86, top-aligned 87 and left-aligned88. Text wrapping is enabled for columns C through G89. A thin black border is on all cells90.
* **Column A (Person):** Uses merged cells to visually group all tasks under a single person91.

**B. Column-by-Column Breakdown and Formulas**

* **Column B: # (Item Count)**
  + **Purpose:** To count the total tasks assigned to the person in Column A92.
  + **Formula for cell B4 (for name in A4):**

Excel

=COUNTIF('Project Plan Detail'!M:M, A4)

* + **Example:** For "Payton", the Currently With column in the sample data contains this name once (Row 8). The formula returns **1**.
* **Column C: ID (Item Names)**
  + **Purpose:** To gather and list all item names assigned to a person in a single cell93.
  + **Formula for cell C4 (for name in A4):**

Excel

=TEXTJOIN(CHAR(10), TRUE, IF('Project Plan Detail'!M:M=A4, 'Project Plan Detail'!F:F, ""))

* + **TEXTJOIN(...):** Joins text from multiple cells into one string94.
  + **CHAR(10):** This is the delimiter which creates a line feed, or new line, within the cell95.
  + **TRUE:** This argument tells TEXTJOIN to ignore any empty values96.
  + **IF(...):** This is the core logic array97. It checks every cell in column M to see if it equals the name in A498. If true, it returns the value from the Item column (F) 99; if false, it returns an empty string ""100. TEXTJOIN then assembles all the returned item names101.

This is a table

A1 Empty

A2 and down – List of individuals by name of team Member (i.e A3 - Kyle | A4 - Sam | A5 - Han etc.) these headers and columns should be populated and the column and all related formulas in the column get set up each time a team member is added on the Team Members Tab

**Overall Table Purpose and Function**

The primary purpose of this Excel sheet is to function as a comprehensive **task and workflow tracking system**. It is designed to monitor various items or controls, assign them to specific individuals, and track their progress through different stages. Key information such as ownership, current status, deadlines, and related identifiers from previous periods are all captured in a structured format. The inclusion of filters and dynamic counting formulas indicates it is built for active management, allowing users to easily sort, find, and summarize information based on various criteria like the person responsible or the task's status.

**Detailed Structure, Formatting, and Cell References**

This description assumes the table begins at cell **A1** on an active worksheet.

**Header Row (Row 1)**

The header row establishes the data categories for the columns below.

* **Location:** The header occupies the range **A1:G1**.
* **Formatting:**
  + **Background Color:** The cells have a solid fill of a standard medium blue (approximating RGB: 79, 129, 189).
  + **Font:** The text is formatted in **white**, **bold**, and is **horizontally centered** within each cell.
  + **Functionality:** The **AutoFilter** feature is enabled for this row. This is visually indicated by the small dropdown filter arrow (▼) on the right side of each header cell. This is the core of the table's interactive functionality.
* **Cell-by-Cell Breakdown:**
  +  **B1:** Contains the text #. This column is for the count of items assigned to the person.
  +  **C1:** Contains the text ID. This serves as the primary identifier for the task.
  +  **D1:** Contains the text With. This indicates who currently has the action item or where it is in the workflow.
  +  **E1:** Contains the text Status. This describes the current state of the task (e.g., "Not Started", "Pending Evidence").
  +  **F1:** Contains the text Tracking Status. This provides deadline-related information, such as how overdue an item is.

The rows below the header contain the specific details for each task, grouped by the assigned person.

* **General Formatting:**
  + **Background Color:** Standard white fill.
  + **Font:** Standard black, non-bold text.
  + **Alignment:** Text is generally **top-aligned** and **left-aligned**, with **text wrapping** enabled for columns C through G to accommodate multiple lines of text within a single cell.
  + **Borders:** A thin black border is applied to all cell edges, creating a clear grid structure.
  + **Row Height:** Row heights are automatically adjusted to be taller than the default to fit the multi-line content.

**Overall Purpose & Setup**

The purpose of this Excel file is to create a **dynamic summary dashboard** on a main tab. This dashboard automatically aggregates tasks or items from a comprehensive master list located on a separate worksheet named **"Project Plan Detail"**. For each person listed on the dashboard, it displays a real-time count of their assigned items, the specific names of those items, their current status, and a timing metric (like how long they are overdue).

The setup consists of two distinct worksheets:

1. **'Project Plan Detail' Sheet (Source):** This sheet serves as the raw data source. It contains the detailed information for every single task or item. Based on your requirements, the key columns on this sheet are:
   * **Column F:** Contains the name or ID of the item (e.g., "NHC.AM.IFRS16 (Sales Return Approval)").
   * **Column L:** Contains the status of the item (e.g., "Spilling Partner Notice").
   * **Column M:** Contains the name of the person the item is "Currently With" (e.g., "Grant").
   * **Column N:** Contains the timing information for the item (e.g., "Overdue by 163 days").
2. **Main Summary Sheet (Dashboard):** This is the formatted table you see in the images. It pulls information from the 'Project Plan Detail' sheet and presents it in a consolidated view.

**Formula Explanations (Revised)**

Below is a breakdown of the formulas for your summary dashboard. We will assume your summary table starts in **Column A** of your main sheet, with the headers in row 3 and the first person, "Kyle," listed in cell A4.

**Column B: # (Item Count)**

This column counts how many items are assigned to the person in that row. It uses the COUNTIF function, referencing the data on the 'Project Plan Detail' sheet.

**Formula for cell B4 (for Kyle):**

Excel

=COUNTIF('Project Plan Detail'!M:M, A4)

* COUNTIF(...): This function counts the number of cells within a range that meet a specific criterion.
* 'Project Plan Detail'!M:M: This is the **range** to check. It looks at the entirety of **Column M** (the "Currently With" column) on the sheet named **'Project Plan Detail'**.
* A4: This is the **criterion**. The formula counts how many times the name "Kyle" (the value in A4 on your summary sheet) appears in column M of the data sheet.

**Columns C, D, and E: ID, Status, and Timing**

These columns use a combination of the TEXTJOIN and IF functions to gather and list all the relevant information for each person. The formula structure is identical for all three; only the column being referenced for data (F:F, L:L, or N:N) changes.

**Cell C4: ID (Item Names)**

This formula collects all item names assigned to Kyle from the 'Project Plan Detail' sheet and lists them, separated by a line break.

**Formula:**

Excel

=TEXTJOIN(CHAR(10), TRUE, IF('Project Plan Detail'!M:M=A4, 'Project Plan Detail'!F:F, ""))

* TEXTJOIN(...): Joins text from multiple cells into one, using a specified delimiter.
* CHAR(10): This is the **delimiter**. CHAR(10) creates a **line feed** (a new line), causing each item to be displayed on its own line within the cell. *Note: "Wrap Text" formatting must be enabled on the cell for this to be visible.*
* TRUE: This argument tells TEXTJOIN to ignore any empty values, which is crucial for the formula to work cleanly.
* IF('Project Plan Detail'!M:M=A4, 'Project Plan Detail'!F:F, ""): This is the core logic that finds the correct items. It is an array formula that processes the entire column.
  + **Logical Test:** 'Project Plan Detail'!M:M=A4 checks every cell in column M on the data sheet to see if it equals "Kyle".
  + **Value if True:** For every row where the test is true, it returns the value from the corresponding cell in **Column F** ('Project Plan Detail'!F:F), which contains the Item Name.
  + **Value if False:** If a cell in column M does not contain "Kyle", it returns an empty string "".
* The TEXTJOIN function then takes all the returned item names and joins them together with line breaks.

**Cell D4: Status**

This formula works exactly like the one for the ID, but it pulls the corresponding data from the "Status" column (L) on the 'Project Plan Detail' sheet.

**Formula:**

Excel

=TEXTJOIN(CHAR(10), TRUE, IF('Project Plan Detail'!M:M=A4, 'Project Plan Detail'!L:L, ""))

**Cell E4: Timing**

This formula follows the same pattern, pulling data from the "Timing" column (N) on the 'Project Plan Detail' sheet.

**Formula:**

Excel

=TEXTJOIN(CHAR(10), TRUE, IF('Project Plan Detail'!M:M=A4, 'Project Plan Detail'!N:N, ""))

# VII. 'Wps by individual' Tab

Tab should have embedded code that it does not automatically calculate/update on its own

Tab should have a button that says update to update the data on the tab based on changes that have been made to the other tabs and refresh

This is a powerful, multi-faceted tool that serves as a detailed, filterable summary report for a single individual selected from a dropdown list.

* **Control Cell:** Cell **E2** contains a dropdown to select a team member's name, which drives the filter102.
* **Core Formula:** The report is generated by a single dynamic array formula that filters the master data and reorders the columns to match a predefined set of headers on the sheet103.
* **Checkboxes for Filtering:** This sheet will contain two checkboxes to provide further filtering capabilities.
  + **'Currently With Filter' Checkbox:** When checked, this will refine the results to show only the tasks that are *currently with* the selected individual (i.e., their name is in the Currently With column).
  + **'Already Passed Toggle' Checkbox:** When checked, this will hide any tasks where the individual's role has already been marked as "Passed" by the milestone formula, allowing them to focus only on active and upcoming items.

This section details the two potential methods for constructing the dynamic, filtered report on the 'Wps by individual' tab. This report's purpose is to aggregate data from the master tracker to give a detailed item-level list of workload assigned to an individual.

**Method 1: The Single-Formula Solution (Recommended)**

This modern approach uses a single, powerful LET formula placed in a single cell (e.g., G5) to generate the entire report.

* **Formula:**

Excel

=LET(Name, $E$2, MasterHeaders, $G$4:$K$4, SourceData, 'Project Plan Detail'!$A$3:$Q$220, SourceHeaders, 'Project Plan Detail'!$A$2:$Q$2, SourceSearchRange, 'Project Plan Detail'!$H$3:$L$220, FilteredRows, FILTER(SourceData, BYROW(SourceSearchRange, LAMBDA(row, SUM(--ISNUMBER(SEARCH(Name, row))) > 0)), "No matching tasks found"), IF(OR(FilteredRows = "No matching tasks found"), FilteredRows, CHOOSECOLS(FilteredRows, MATCH(MasterHeaders, SourceHeaders, 0))))

* **Formula Deep Dive:**
  + **LET(...)**: This function allows us to assign names to ranges and calculations, making the formula cleaner and more efficient. We define Name (the person to filter by from E2) , MasterHeaders (the desired output headers from G4:K4) , SourceData , SourceHeaders , and SourceSearchRange.
  + **FilteredRows, FILTER(...)**: This is the core filtering step.
    - **BYROW(SourceSearchRange, LAMBDA(...))**: This function processes the SourceSearchRange (the role columns H:L) one row at a time.
    - **LAMBDA(row, ...)**: This creates a mini-function that is applied to each row.
    - **SEARCH(Name, row)**: This looks for the Name within the current row. It returns an array of numbers (positions) and errors.
    - **ISNUMBER(...)**: This converts the array to TRUE/FALSE values (TRUE where the name was found).
    - **--**: The double negative coerces TRUE/FALSE to 1/0.
    - **SUM(...) > 0**: This sums the 1s and 0s. If the sum is greater than 0, it means the name was found at least once in that row, and BYROW returns TRUE for that row.
  + **IF(OR(...), ..., CHOOSECOLS(...))**: This is the final output step. If the filter found nothing, it returns the "No matching tasks found" message. Otherwise, it reorders the data.
    - **CHOOSECOLS(FilteredRows, MATCH(MasterHeaders, SourceHeaders, 0))**: MATCH creates an array of numbers representing the column positions of our desired MasterHeaders within the original SourceHeaders. CHOOSECOLS then uses this array to rearrange the FilteredRows into the correct final order.

**Row 1 & 2: Merged Headers**

* **Q1:R1**: Merged with the text "Date the review is expected to be done and pass along to next reviewer". The cell has a purple fill (#B4C6E7).
* **P2:S2**: Merged with the text "Due Dates". The cell has a light blue fill (#DDEBF7).

**Row 3: Column Headers**

column **(Tracking Vs. Schedule):**

* ok: Green fill (#C6EFCE) with dark green text. (Row 5)
* Overdue: Light red fill (#FFC7CE) with dark red text. (Rows 6-24)

column **(Tracking Timeline):**

* Complete: Green fill (#70AD47). (Row 5)
* Text starting with "overdue": Light red fill (#FFC7CE). (Rows 6-24)
* Text starting with "Due in": Dark red fill (#C00000) with white text. (Rows 27-28)
* Text starting with "Due in": Yellow fill (#FFFF00). (Rows 29-30)

Column (Status)

* **Cell G2**: Contains the formula =COUNTA(F5:F28) which resolves to "66.00 Count". This is likely counting the number of tasks listed. The cell has a light grey fill.

**The Single-Formula Solution**

This approach uses the LET function to make the formula clean and easy to read. It will perform the filtering and the column standardization all in one step.

**Step 1: Set Up Your Control Panel**

This remains the same. On your main sheet, you need two things:

1. A cell with the name to filter by (e.g., $E$2).
2. A range with your desired final column headers (e.g., $G$4:$????$4). The order of these headers will control the order of the output.
   1. Unique ID
   2. Level 1
   3. Level 2
   4. Level 3
   5. Workpaper or Leap item
   6. Status
   7. Currently With
   8. Tracking Vs. Schedule
   9. Tracking Timeline
   10. Preparer
   11. Sr. Reviewer
   12. Mgr Reviewr
   13. Partner Review
   14. Partner Review 2
   15. Prep Due Date
   16. Sr Review Due Date
   17. Mgr Review Due Date
   18. Partner Review Due Date
   19. Partner Review 2 Due Date

**Step 2: Place the Formula**

On your main dashboard, in the cell directly below your master headers (e.g., G5), place the following formula:

Excel

=LET(

// --- SETUP: Define your inputs ---

Name, $E$2,

MasterHeaders, $G$4:$K$4,

// --- SOURCE: Point this to your single dataset (EDIT THESE THREE LINES) ---

SourceData, 'Int#2'!$A$3:$Q$220,

SourceHeaders, 'Int#2'!$A$2:$Q$2,

SourceSearchRange, 'Int#2'!$H$3:$L$220,

// --- STEP 1: Filter the rows that match the name ---

FilteredRows, FILTER(SourceData, BYROW(SourceSearchRange, LAMBDA(row, SUM(--ISNUMBER(SEARCH(Name, row))) > 0)), "No matching tasks found"),

// --- STEP 2: If rows were found, reorder the columns. Otherwise, show the message. ---

IF(

OR(FilteredRows = "No matching tasks found"),

FilteredRows,

CHOOSECOLS(FilteredRows, MATCH(MasterHeaders, SourceHeaders, 0))

)

)

**How to Customize This Formula**

You only need to edit the three lines in the --- SOURCE --- section to point to your single data sheet:

* **SourceData**: Change 'Int#2'!$A$3:$Q$220 to the full data range of your table, not including the headers.
* **SourceHeaders**: Change 'Int#2'!$A$2:$Q$2 to the header row of your data table.
* **SourceSearchRange**: Change 'Int#2'!$H$3:$L$220 to the range of columns where the names of assigned people are located.

**How It Works**

1. **LET**: This function lets us name parts of our formula (Name, MasterHeaders, FilteredRows, etc.) to make it clean and efficient.
2. **FilteredRows**: This part is identical to the original logic. It filters your SourceData to find all rows containing the Name you specified. If it finds nothing, it returns the text "No matching tasks found".
3. **IF(...)**: This is the final output. It checks if the FilteredRows variable contains our "not found" message.
   * If it does, the formula simply outputs "No matching tasks found".
   * If data *was* found, it proceeds to the CHOOSECOLS function.
4. **CHOOSECOLS(FilteredRows, MATCH(MasterHeaders, SourceHeaders, 0))**: This reorders the data. The MATCH function figures out what columns to grab based on your MasterHeaders, and CHOOSECOLS arranges the FilteredRows into that new order.

The result is a single, fast, and responsive formula that gives you complete control over the final report's column order.

### Formula #1: Role Identification

Excel

=IF(COUNTIF(K5,"="&$E$2),"Preparer",

IF(COUNTIF(L5,"="&$E$2),"Reviewer",

IF(COUNTIF(M5,"="&$E$2),"Manager Reviewer",

IF(OR(COUNTIF(N5,"="&$E$2),COUNTIF(O5,"="&$E$2)),"Partner Reviewer","Not Found"))))

**Overall Purpose**

This formula identifies a person's **role** on a specific task. It searches for a name, provided in cell $E$2, across the various role columns for the task in row 5 (K5:O5). It returns the title of the first role it finds that matches the name.

**Step-by-Step Breakdown**

The formula checks each role column sequentially for the name in $E$2.

1. **Check Preparer:** COUNTIF(K5,"="&$E$2) checks if the name in $E$2 is in the **"Preparer"** cell (K5). If it is, the formula returns "Preparer" and stops.
2. **Check Sr. Reviewer:** If not found, COUNTIF(L5,"="&$E$2) checks the **"Sr. Reviewer"** cell (L5). If it matches, it returns "Reviewer".
3. **Check Mgr Reviewer:** If not found, COUNTIF(M5,"="&$E$2) checks the **"Mgr Reviewer"** cell (M5). If it matches, it returns "Manager Reviewer".
4. **Check Partner Reviewer:** If still not found, OR(COUNTIF(N5,"="&$E$2),COUNTIF(O5,"="&$E$2)) checks if the name exists in **either** the **"Partner Review"** cell (N5) or the next partner column (O5). If it's in either, it returns "Partner Reviewer".
5. **Default:** If the name is not found in any of these cells, the formula returns "Not Found".

### Formula #2: Milestone Status Check in c5

**Overall Purpose**

This formula determines if a specific work stage has been completed, returning **"Passed"** or **"Not Passed"**. It acts as a "gating" function, understanding that when a task reaches a later status (e.g., "Partner Review"), all earlier stages (like "Senior Review") are implicitly complete. It evaluates the role in column A against the overall task status in column G.

**Step-by-Step Breakdown**

The formula's logic changes based on the role in cell A5.

1. **If Role is "Preparer":** The rule is simple. The task has "Passed" only if the status in G5 is "Complete".
2. **If Role is "Reviewer":** The formula uses MATCH to find the position of the current status (G5) within a hardcoded list of all 20 workflow statuses. If the position is >7, it's "Passed". The 8th status is "Manager Review," so this means the Reviewer's stage is considered passed as soon as the task moves to the Manager Review stage or beyond.
3. **If Role is "Manager Review":** It uses the same MATCH logic but checks if the position is >10. The 11th status is "Partner Review," so the Manager stage is passed once the task reaches Partner Review.
4. **If Role is "Partner Review":** It checks if the position is >13. The 14th status is "2Partner Review," meaning this stage is passed once the second Partner Review begins.
5. **If Role is "Partner Review2":** The task has "Passed" only if the status in G5 is "Complete".
6. **Default:** If the role in A5 doesn't match any of these, it returns "Not Passed".

* **Utilize** helper shet 4c table.

2. Your formula then becomes a simple and robust lookup.

=IF(

XLOOKUP(G5, Workflow[Status], Workflow[Sequence]) >= XLOOKUP(A5, Workflow[Role Milestone], Workflow[Sequence]),

"Passed",

"Not Passed"

)

This formula looks up the sequence number of the current status and compares it to the sequence

**Step 2: Replace the Old Formula**

You can now replace the entire complex, nested IF/MATCH formula with the following single, simple formula. Assuming the role to check is in cell A5 and the current status is in G5, this formula would go in cell B5.

Excel

=LET(

Role, A5,

CurrentStatus, G5,

CurrentSequence, XLOOKUP(CurrentStatus, WorkflowConfig!A:A, WorkflowConfig!B:B, 0),

RequiredSequence, XLOOKUP(Role, WorkflowConfig!C:C, WorkflowConfig!B:B, 0),

IF(Role = "Preparer",

IF(CurrentStatus = "Complete", "Passed", "Not Passed"),

IF(CurrentSequence >= RequiredSequence, "Passed", "Not Passed")

)

)

**How**

**How the New Formula Works**

This formula is much easier to understand:

1. **LET**: We name our inputs (Role, CurrentStatus) to make the formula readable.
2. **CurrentSequence**: It performs an XLOOKUP to find the current status (from G5) in the Status column of our helper table and returns its Sequence number.
3. **RequiredSequence**: It performs a second XLOOKUP to find the current role (from A5) in the RoleMilestone column and returns the Sequence number required for that role to pass.
4. **IF Logic**:
   * First, it handles the special case for the "Preparer," which passes only when the task is "Complete".
   * For all other roles, it simply checks if the CurrentSequence is greater than or equal to the RequiredSequence. If it is, the milestone has "Passed".

VBA Checkboox Buttons:

* 'Wps by individual' refresh
* Updates the sheet calculations
* 'Wps by individual' Currently With Filter Toggle Checkbox
* Hides all rows where the person In E2 is not the value in Currently with
* 'Wps by individual' Already Passed Toggle Checkbox
* Hides all rows with Passed Noted in column C

# 4xIV. Summary Dashboards (Tabs: Level 1, 2, 3, Account)

These tabs provide high-level, visual summaries for leadership or management, designed to transform the granular data from 'Project Plan Detail' into actionable insights regarding project progress and potential bottlenecks67. Each tab should have an "Update" button to refresh its data.

## A. Brainstormed Dashboard Design: 'Level 1 Summary'

This dashboard serves as the main "Control Tower" view of the entire project.

* **Layout:** A clean, modern layout with slicers at the top, key performance indicator (KPI) cards below them, a main summary table on the left, and visual charts on the right.
* **Interactivity:** Slicers for Partner and Phase would be placed at the top of the dashboard, allowing a user to filter the entire sheet's view dynamically.
* **KPI Cards:** A series of large, prominent cards at the top displaying critical, at-a-glance metrics:
  + **Total Items:** Formula: =COUNTA('Project Plan Detail'!F:F)
  + **% Complete:** Formula: =COUNTIF('Project Plan Detail'!L:L, "Complete") / COUNTA('Project Plan Detail'!F:F), formatted as a percentage.
  + **Items Overdue:** Formula: =COUNTIF('Project Plan Detail'!N:N, "Overdue")
  + **On Track:** Formula: =COUNTIF('Project Plan Detail'!N:N, "On Schedule")
* **Main Summary Table (PivotTable):**
  + **Source:** The data range from the 'Project Plan Detail' sheet.
  + **Rows:** Level 1 field.
  + **Columns:** A calculated field or helper column that groups the detailed statuses into broader categories (e.g., "Not Started", "In Progress", "Review Stages", "Complete").
  + **Values:** Count of Item.
  + **Purpose:** This table shows exactly how many workpapers are in each stage of the workflow for each major audit area, highlighting where work is concentrated.
* **Charts & Visuals:**
  + **Overall Health (Donut Chart):** A large donut chart visually representing the "% On Track vs % Delayed"68. It would show three segments: "On Schedule" (Green), "Overdue" (Red), and "Complete" (Blue), based on the Timing column.
  + **Progress by Area (Stacked Bar Chart):** A horizontal stacked bar chart. Each bar represents a Level 1 audit area. The length of the bar is the total number of items, and it is segmented by color based on status (e.g., green for Complete, yellow for In Progress, red for Overdue). This provides an immediate visual comparison of progress and problems across different parts of the project.

## B. 'Level 2/3' and 'Account' Summary Tabs

* **Level 2/3 Summaries:** These would be drill-down dashboards. The 'Level 2 Summary' would be nearly identical to the Level 1 dashboard but would include a slicer for the Level 1 field. When a user selects a Level 1 area (e.g., "Revenue"), the entire dashboard would update to show the breakdown of its Level 2 sub-components. The 'Level 3' tab would follow the same pattern, drilling down from Level 2.
* **Account Summary:** This dashboard would pivot the data differently. Instead of using the Level hierarchy, its primary dimension would be the Related Account if Applicable column69, showing the status and timeliness of all workpapers associated with specific financial statement accounts. The layout of KPI cards, tables, and charts would be similar, providing a view of the audit from an account-based perspective.

# VIII. Helper and Configuration Tabs

# A. Tab: Team Members

* **Purpose:** The single, authoritative source for the team member list 104, referenced by functions throughout the entire workbook105.
* **Setup:** A worksheet named Team Members106. Cell A1 has the header "Team Member Name"107. Names are listed one per row, starting in A2108. It should also include the values: "- Later", "TBD", and "Pending Support".
* **Dynamic Named Range:** A named range TeamMemberList is created with the formula: =OFFSET('Team Members'!$A$2,0,0,COUNTA('Team Members'!$A:$A)-1,1)109. This ensures the list automatically expands when new members are added110.
* **Interactive Buttons:**
  + **"Add Team Member":** Runs a VBA macro that prompts the user with an InputBox for the new name 111, checks that the input isn't blank 112, finds the first empty cell in column A 113, writes the name into that cell 114, and displays a confirmation MsgBox115.
  + **"Remove Team Member":** Runs a VBA macro to delete a selected team member.

VBA

' Macro to remove a selected team member from the list.

' Version: 1.0

' Created: 2025-06-12

Public Sub RemoveTeamMember\_v1\_20250612()

Dim ws As Worksheet

Dim nameToRemove As Range

Dim response As VbMsgBoxResult

On Error Resume Next 'Simple error handling for selection

Set ws = ThisWorkbook.Sheets("Team Members")

Set nameToRemove = Application.Selection

On Error GoTo 0

' --- Validate Selection ---

If nameToRemove Is Nothing Or nameToRemove.Count > 1 Or Intersect(nameToRemove, ws.Columns("A")) Is Nothing Then

MsgBox "Please select a single name from the list in Column A to remove.", vbInformation

Exit Sub

End If

' --- Confirm Deletion ---

response = MsgBox("Are you sure you want to permanently remove '" & nameToRemove.Value & "' from the team list?", \_

vbYesNo + vbExclamation, "Confirm Deletion")

If response = vbYes Then

Application.ScreenUpdating = False

Debug.Print "User confirmed deletion of: " & nameToRemove.Value

nameToRemove.Delete Shift:=xlUp 'Delete and shift cells up

MsgBox "'" & nameToRemove.Value & "' has been removed.", vbInformation

Application.ScreenUpdating = True

Else

Debug.Print "User cancelled deletion."

End If

End Sub

# B. Tab (Hidden): Statuses

* **Purpose:** A hidden configuration table for all workflow statuses and their corresponding visual styles116.
* **Setup:** A new worksheet named Statuses117. Headers in A1:C1 are Status, Font Color (HEX), Fill Color (HEX)118. The table is populated with every status and its color codes119. The sheet is then hidden.
* **Data Table:** The table contains all 23 specified statuses and their associated HEX color codes for font and fill120. For example:
  + Pending Evidence, #FFFFFF, #FF7C80
  + In Progress, #FFFFFF, #FFC000
  + Complete, #FFFFFF, #66FF99
* **Purpose:** This hidden sheet acts as a configuration table for workflow statuses and their corresponding visual styles.
* **Implementation:**
  + Create a new worksheet named Statuses.
  + In A1:C1, enter the headers: Status, Font Color (HEX), Fill Color (HEX).
  + Populate this table exactly as specified in the source document, with each status on a new row.
    - *Example Row 2:* Not Started, #FFFFFF, #FFFFFF
    - *Example Row 3:* Pending Evidence, #FFFFFF, #FF7C80
    - Continue for all 23 specified statuses.
  + Create a **Dynamic Named Range** for the status list. Name it StatusList. In the "Refers to" field, enter: =OFFSET('Statuses'!$A$2,0,0,COUNTA('Statuses'!$A:$A)-1,1).
  + Right-click the Statuses tab and select "Hide".

|  |  |  |
| --- | --- | --- |
| **Cell Value** | **Font Color (Hex)** | **Fill Color (Hex)** |
| Not Started | #000000 | #FF7C80 |
| Pending Evidence | #000000 | #FFC000 |
| In Progress | #000000 | #FFFF00 |
| In Review | #000000 | #00B0F0 |
| Senior Review | #000000 | #99CCFF |
| Closing Senior Notes | #000000 | #9BC2E6 |
| Spiking Senior Notes | #000000 | #9BC2E6 |
| Manager Review | #000000 | #CCCCFF |
| Closing Manager Notes | #000000 | #CC99FF |
| Spiking Manager Notes | #000000 | #CC99FF |
| Partner Review | #000000 | #FFCCFF |
| Closing Partner Notes | #000000 | #FF99FF |
| Spiking Partner Notes | #000000 | #FF66CC |
| 2Partner Review | #CC6600 | #FFCCCC |
| 2Closing Partner Notes | #000000 | #FF9999 |
| 2Spiking Partner Notes | #800000 | #FF9966 |
| 3Partner Review | #CC6600 | #FFCCCC |
| 3Closing Partner Notes | #000000 | #FF9999 |
| 3Spiking Partner Notes | #800000 | #FF9966 |
| EQCR Review | #000000 | #66FF99 |
| Closing EQCR Notes | #000000 | #00FF00 |
| Spiking EQCR Notes | #000000 | #00FF00 |
| Complete | #000000 | #00B050 |
| INDUS | #000000 | #CC9900 |
| TBD | #FFFFFF | #0055FE |
| Later | #FFFFFF | #0055FE |
| Not Tested | #000000 | #7B7B7B |

# C. Tab (Hidden): LookupData

* **Purpose:** A helper sheet that maps every possible task status to a specific role's column letter or a terminal state, making the 'Currently With' formula manageable and scalable121.
* **Setup:** A sheet named LookupData122. Headers are Status and Result123. The Result column contains the corresponding column letter (e.g., 'X') or terminal status name (e.g., 'Complete') for each status124. A named range StatusLookupTable refers to this table125.

This new helper sheet is the engine for the "Currently With" column. It maps every possible task status to a specific person's column or a terminal state, making the main formula far more manageable and scalable.

* **Implementation Steps:**
  1. Create a worksheet named **LookupData**.
  2. In A1:B1, enter the headers: **Status** and **Result**.
  3. Populate this table with every single status from the StatusList. In the Result column, enter the corresponding column letter where that role's name is stored on the Project Plan Detail sheet, or the name of the terminal status itself.

|  |  |
| --- | --- |
| **Status** | **Result** |
| Not Started | P |
| Pending Evidence | P |
| In Progress | P |
| In Review | R |
| Senior Review | R |
| Closing Senior Notes | P |
| Spiking Senior Notes | R |
| Manager Review | S |
| Closing Manager Notes | P |
| Spiking Manager Notes | S |
| Partner Review | T |
| Closing Partner Notes | P |
| Spiking Partner Notes | T |
| 2Partner Review | U |
| 2Closing Partner Notes | P |
| 2Spiking Partner Notes | U |
| 3Partner Review | V |
| 3Closing Partner Notes | P |
| 3Spiking Partner Notes | V |
| EQCR Review | W |
| Closing EQCR Notes | P |
| Spiking EQCR Notes | W |
| Complete | Complete |
| Not Tested | Not Tested |
| INDUS | INDUS |
|  |  |

Export to Sheets

1. Create a **Named Range** called **StatusLookupTable** that refers to this entire table (e.g., LookupData!$A$2:$B$25).

**'LookupData' Tab (Helper Sheet 3b)**

* **Purpose:** This new helper sheet is the engine for the "Currently With" column. Its purpose is to map every possible task status to a specific person's column or a terminal state, making the main formula far more manageable and scalable than a hardcoded nested IF statement.
* **Implementation:**
  1. A new worksheet is created and named LookupData.
  2. In cells A1:B1, the headers Status and Result are entered.
  3. The table is populated with every single status from the StatusList.
  4. In the Result column, the corresponding column letter where that role's name is stored on the 'Project Plan Detail' sheet is entered. For terminal statuses, the name of the status itself is entered.
* **Data Table Example:**

|  |  |
| --- | --- |
| **Status** | **Result** |
| Not Started | P |
| Pending Evidence | P |
| In Progress | P |
| In Review | R |
| Senior Review | R |
| Closing Senior Notes | P |
| Spiking Senior Notes | R |
| Manager Review | S |
| Closing Manager Notes | P |
| Spiking Manager Notes | S |
| Partner Review | T |
| Closing Partner Notes | P |
| Spiking Partner Notes | T |
| 2Partner Review | U |
| 2Closing Partner Notes | P |
| 2Spiking Partner Notes | U |
| 3Partner Review | V |
| 3Closing Partner Notes | P |
| 3Spiking Partner Notes | V |
| EQCR Review | W |
| Closing EQCR Notes | P |
| Spiking EQCR Notes | W |
| Complete | Complete |
| Not Tested | Not Tested |
| INDUS | INDUS |
|  |  |

* **Named Range:** A named range called StatusLookupTable is created that refers to this entire table (e.g., LookupData!$A$2:$B$25).
* **Formula Integration:** With this table, the formula in the Currently With column (M) on the 'Project Plan Detail' sheet becomes a robust INDEX/VLOOKUP combination, for example: =IFERROR(INDEX(P5:AC5, 1, MATCH(VLOOKUP(L5, StatusLookupTable, 2, FALSE), $P$1:$AC$1, 0)), VLOOKUP(L5, StatusLookupTable, 2, FALSE)). This formula looks up the status (L5), gets the corresponding column header from the helper table, finds that header in the main sheet's header row, and returns the name from that column for the current row.

# D. Tab (Hidden): WorkflowConfig

* **Purpose:** The engine for the advanced "Done?" / "Passed" milestone calculation, defining the entire workflow numerically126.
* **Setup:** A sheet named WorkflowConfig127.
* **Data Table:** A three-column table containing Status, Sequence, and RoleMilestone128.
  + **Status:** An exact list of all possible statuses129.
  + **Sequence:** A unique number defining each status's position in the workflow130.
  + **RoleMilestone:** The name of the role whose work is considered "Passed" once the workflow reaches the corresponding status131.

|  |  |  |
| --- | --- | --- |
| **Status** | **Sequence** | **RoleMilestone** |
| Not Started | 1 |  |
| Pending Evidence | 2 |  |
| In Progress | 3 |  |
| In Review | 4 |  |
| Senior Review | 5 |  |
| Closing Senior Notes | 6 |  |
| Spiking Senior Notes | 7 |  |
| **Manager Review** | **8** | **Reviewer** |
| Closing Manager Notes | 9 |  |
| Spiking Manager Notes | 10 |  |
| **Partner Review** | **11** | **Manager Review** |
| Closing Partner Notes | 12 |  |
| Spiking Partner Notes | 13 |  |
| **2Partner Review** | **14** | **Partner Review** |
| 2Closing Partner Notes | 15 |  |
| 2Spiking Partner Notes | 16 |  |
| EQCR Review | 17 |  |
| Closing EQCR Notes | 18 |  |
| Spiking EQCR Notes | 19 |  |
| **Complete** | **20** | **Partner Review2** |