-- Identifies duplicate employee numbers in the Staff Data table to support data cleanup and validation.

EmployeeNo\_Duplicate\_Check =

VAR CurrentValue = 'Staff Data'[STAFF\_ID]

VAR CountOfValue =

CALCULATE(

COUNTROWS('Staff Data'),

FILTER('Staff Data', 'Staff Data'[STAFF\_ID] = CurrentValue)

)

RETURN IF(CountOfValue > 1, TRUE(), FALSE())

-- Identifies duplicate post IDs in the Staff Data table to flag possible data entry issues.

PostID\_Duplicate\_Check =

VAR CurrentValue = 'Staff Data'[POSITION\_ID]

VAR CountOfValue =

CALCULATE(

COUNTROWS('Staff Data'),

FILTER('Staff Data', 'Staff Data'[POSITION\_ID] = CurrentValue)

)

RETURN IF(CountOfValue > 1, TRUE(), FALSE())

-- Generates a unique anonymised code for each employee based on employee number, post ID, and first name.

Anonymised Code =

VAR Emp = 'Staff Data'[STAFF\_ID]

VAR PostA = 'Staff Data'[POSITION\_ID]

VAR FirstName = 'Staff Data'[FIRST\_NAME]

VAR BaseCode = "A" & RIGHT(Emp, 6)

-- Create a summary table with unique combinations of POSITION\_ID and FIRST\_NAME for the employee

VAR UniqueCombos =

SUMMARIZE(

FILTER(

'Staff Data',

'Staff Data'[STAFF\_ID] = Emp

),

'Staff Data'[POSITION\_ID],

'Staff Data'[FIRST\_NAME]

)

-- Count of unique combinations

VAR UniqueComboCount = COUNTROWS(UniqueCombos)

-- Create a table to rank the current combination

VAR CurrentCombo = PostA & "|" & FirstName

VAR ComboWithIndex =

ADDCOLUMNS(

UniqueCombos,

"ComboKey", 'Staff Data'[POSITION\_ID] & "|" & 'Staff Data'[FIRST\_NAME]

)

VAR Index =

RANKX(

ComboWithIndex,

[ComboKey],

CurrentCombo,

ASC,

DENSE

)

RETURN

IF(

UniqueComboCount = 1,

BaseCode,

BaseCode & "\_" & Index

)

-- Flags whether a record should be removed by matching it against the ToRemove table.

Not Ease =

VAR Match =

LOOKUPVALUE(

ToRemove[ID2],

ToRemove[ID1], [ID1\_Name],

ToRemove[ID2], [ID2\_Role\_Name]

)

RETURN

IF(NOT(ISBLANK(Match)), "Remove", "Keep")

-- Identifies employee records marked as LeaverRecords by comparing key identifiers with the LeaverRecords table.

LeaverRecords =

VAR Match =

LOOKUPVALUE(

LeaverRecords[ID2],

LeaverRecords[ID1], [ID1\_Name],

LeaverRecords[ID2], [ID2\_Role\_Name]

)

RETURN

IF(NOT(ISBLANK(Match)), "Remove", "Keep")

-- Categorises termination status based on leaver/removal status and presence of termination-related dates.

TerminationStatus =

SWITCH(TRUE(),

'Staff Data'[LeaverRecords] = "Keep" && ISBLANK('Staff Data'[POST\_END\_DATE]) && ISBLANK('Staff Data'[ACTUAL\_TERMINATION\_DATE]), "No termination",

ISBLANK('Staff Data'[POST\_END\_DATE]) || ISBLANK('Staff Data'[ACTUAL\_TERMINATION\_DATE]), "Missing Date",

'Staff Data'[Not Ease] = "Keep" && 'Staff Data'[LeaverRecords] = "Keep" &&

NOT(ISBLANK('Staff Data'[POST\_END\_DATE])) && NOT(ISBLANK('Staff Data'[ACTUAL\_TERMINATION\_DATE])),

"CHECK - no indication of termination but has both dates",

'Staff Data'[Not Ease] = "Remove" && 'Staff Data'[LeaverRecords] = "Keep" &&

NOT(ISBLANK('Staff Data'[POST\_END\_DATE])) && NOT(ISBLANK('Staff Data'[ACTUAL\_TERMINATION\_DATE])),

"CHECK - non-Ease but has both termination dates",

"Has both dates"

)

-- Returns a clean FTE value based on hierarchy of checks, defaulting to 2 if invalid or duplicate.

Final\_Validated\_FTE 1 =

VAR Emp = 'Staff Data'[STAFF\_ID]

VAR EmpCount =

CALCULATE(

COUNTROWS('Staff Data'),

FILTER('Staff Data', 'Staff Data'[STAFF\_ID] = Emp)

)

RETURN

SWITCH(

TRUE(),

'Staff Data'[Final\_Validated\_FTE] > 0 && 'Staff Data'[Final\_Validated\_FTE] <= 1, 'Staff Data'[Final\_Validated\_FTE],

NOT(ISBLANK('Staff Data'[Final\_Validated\_FTE])) && ('Staff Data'[Final\_Validated\_FTE] > 1 || 'Staff Data'[Final\_Validated\_FTE] = 0), 2,

ISBLANK('Staff Data'[Final\_Validated\_FTE]) && NOT(ISBLANK('Staff Data'[Validated\_FTE])) &&

'Staff Data'[Validated\_FTE] > 0 && 'Staff Data'[Validated\_FTE] <= 1, 'Staff Data'[Validated\_FTE],

ISBLANK('Staff Data'[Final\_Validated\_FTE]) && NOT(ISBLANK('Staff Data'[Validated\_FTE])) &&

('Staff Data'[Validated\_FTE] > 1 || 'Staff Data'[Validated\_FTE] = 0), 2,

ISBLANK('Staff Data'[Final\_Validated\_FTE]) && ISBLANK('Staff Data'[Validated\_FTE]) && NOT(ISBLANK('Staff Data'[FTE])) &&

'Staff Data'[FTE] > 0 && 'Staff Data'[FTE] <= 1, 'Staff Data'[FTE],

ISBLANK('Staff Data'[Final\_Validated\_FTE]) && ISBLANK('Staff Data'[Validated\_FTE]) && NOT(ISBLANK('Staff Data'[FTE])) &&

('Staff Data'[FTE] > 1 || 'Staff Data'[FTE] = 0), 2,

NOT(ISBLANK('Staff Data'[FTE])) && NOT(ISBLANK('Staff Data'[Validated\_FTE])) &&

('Staff Data'[FTE] + 'Staff Data'[Validated\_FTE] > 1), 2,

ISBLANK('Staff Data'[FTE]) && ISBLANK('Staff Data'[Final\_Validated\_FTE]) && ISBLANK('Staff Data'[VALIDATED\_FTE]), 2,

'Staff Data'[VALIDATED\_FTE] > 'Staff Data'[FTE], 2,

EmpCount > 1, 2

)

-- Checks whether an employee from Staff Data also exists in the FACULTY\_A dataset by employee number.

Employee Match Status A & H =

VAR EmpNum = 'Staff Data'[STAFF\_ID]

VAR Employee = LOOKUPVALUE('FACULTY\_A'[STAFF\_ID], 'FACULTY\_A'[STAFF\_ID], EmpNum)

RETURN

IF(

ISBLANK(EmpNum),

BLANK(),

IF(

ISBLANK(Employee),

"No Match",

"Match"

)

)

-- Checks whether an employee from Staff Data also exists in the FACULTY\_B dataset by employee number.

Employee Match Status FACULTY\_B =

VAR EmpNum = 'Staff Data'[STAFF\_ID]

VAR Employee = LOOKUPVALUE('FACULTY\_B'[STAFF\_ID], 'FACULTY\_B'[STAFF\_ID], EmpNum)

RETURN

IF(

ISBLANK(EmpNum),

BLANK(),

IF(

ISBLANK(Employee),

"No Match",

"Match"

)

)

-- Calculates job duration in months based on start date and earliest end or termination date.

Job Duration =

VAR StartDate = 'Staff Data'[POST\_START\_DATE].[Date]

VAR EndDate = 'Staff Data'[POST\_END\_DATE].[Date]

VAR TermDate = 'Staff Data'[ACTUAL\_TERMINATION\_DATE].[Date]

RETURN

IF(

ISBLANK(EndDate) && ISBLANK(TermDate),

BLANK(),

DATEDIFF(

StartDate,

IF(

ISBLANK(EndDate), TermDate,

IF(

ISBLANK(TermDate), EndDate,

MIN(EndDate, TermDate)

)

),

MONTH

)

)

-- Categorises job durations as either greater than or less than 6 months for quick analysis.

Duration Threshold = IF(ISBLANK('Staff Data'[Job Duration]), BLANK(), IF('Staff Data'[Job Duration] > 6, "> 6", "< 6"))

-- Identifies employees scheduled to leave in the future based on termination date.

Future LeaverRecords = IF('Staff Data'[ACTUAL\_TERMINATION\_DATE].[Date] > NOW(), "Future LeaverRecords", "No")

-- Flags employees who joined within the past 6 months to track new starters.

New Starter =

IF(

'Staff Data'[POST\_START\_DATE] >= EDATE(TODAY(), -6),

"New Starter",

"No"

)

-- Compares Staff Data with FACULTY\_A to find matching ID1 and ID2 role combinations.

ID1 & ID2 Match Status A & H =

IF (

NOT (

ISBLANK (

LOOKUPVALUE (

'FACULTY\_A'[ID1\_Name ],

'FACULTY\_A'[ID1\_Name ], 'Staff Data'[ID1\_Name],

'FACULTY\_A'[ID2\_Role\_Name], 'Staff Data'[ID2\_Role\_Name]

)

)

),

"Match",

"Mismatch"

)

-- Compares Staff Data with FACULTY\_A to find matching ID1 and ID2 role combinations.

ID1 & ID2 Match Status A & H =

IF (

NOT (

ISBLANK (

LOOKUPVALUE (

'FACULTY\_A'[ID1\_Name ],

'FACULTY\_A'[ID1\_Name ], 'Staff Data'[ID1\_Name],

'FACULTY\_A'[ID2\_Role\_Name], 'Staff Data'[ID2\_Role\_Name]

)

)

),

"Match",

"Mismatch"

)

-- Validates if ID1 and ID2 combinations in Staff Data exist in the FACULTY\_B dataset.

ID1 & ID2 Match Status FACULTY\_B =

IF (

NOT (

ISBLANK (

LOOKUPVALUE (

'FACULTY\_B'[ID1\_Name],

'FACULTY\_B'[ID1\_Name], 'Staff Data'[ID1\_Name],

'FACULTY\_B'[ID2\_Role\_Name], 'Staff Data'[ID2\_Role\_Name]

)

)

),

"Match",

"Mismatch"

)

-- Flags records as 'Vacant' based on name fields or employee number matching "vacant".

Vacants Check = SWITCH(TRUE(),

'Staff Data'[FIRST\_NAME] = "vacant" && 'Staff Data'[LASTNAME] = "vacant","Vacant",

'Staff Data'[STAFF\_ID] = "vacant","Vacant",

"Non Vacant"

)

-- Identifies duplicate employee numbers in FACULTY\_B for data validation.

EmployeeNo\_Duplicate\_Check =

VAR CurrentValue = 'FACULTY\_B'[STAFF\_ID]

RETURN

IF(

ISBLANK(CurrentValue),

FALSE(),

CALCULATE(

COUNTROWS('FACULTY\_B'),

FILTER('FACULTY\_B', 'FACULTY\_B'[STAFF\_ID] = CurrentValue)

) > 1

)

-- Flags duplicate POSITION\_IDs in FACULTY\_B to detect potential data quality issues.

PostID\_Duplicate\_Check =

VAR CurrentValue = 'FACULTY\_B'[POSITION\_ID]

RETURN

IF(

ISBLANK(CurrentValue),

FALSE(),

CALCULATE(

COUNTROWS('FACULTY\_B'),

FILTER('FACULTY\_B', 'FACULTY\_B'[POSITION\_ID] = CurrentValue)

) > 1

)

-- Generates a unique anonymised code for each employee to pseudonymise identity.

Anonymised Code =

VAR Emp = 'FACULTY\_B'[STAFF\_ID]

VAR PostA = 'FACULTY\_B'[POSITION\_ID]

VAR FirstName = 'FACULTY\_B'[FIRST\_NAME]

RETURN

IF (

ISBLANK(Emp),

BLANK(),

VAR BaseCode = "A" & RIGHT(Emp, 6)

VAR UniqueCombos =

SUMMARIZE(

FILTER(

'FACULTY\_B',

'FACULTY\_B'[STAFF\_ID] = Emp

),

'FACULTY\_B'[POSITION\_ID],

'FACULTY\_B'[FIRST\_NAME]

)

VAR UniqueComboCount = COUNTROWS(UniqueCombos)

VAR CurrentCombo = PostA & "|" & FirstName

VAR ComboWithIndex =

ADDCOLUMNS(

UniqueCombos,

"ComboKey", 'FACULTY\_B'[POSITION\_ID] & "|" & 'FACULTY\_B'[FIRST\_NAME]

)

VAR Index =

RANKX(

ComboWithIndex,

[ComboKey],

CurrentCombo,

ASC,

DENSE

)

RETURN

IF(

UniqueComboCount = 1,

BaseCode,

BaseCode & "\_" & Index

)

)

-- Creates a unique ID by combining department, employee, assignment, and last name.

ID1\_Name = [DEPT] & "|" & [STAFF\_ID] & "|" & [ASSIGNMENT\_ID] & "|" & [LASTNAME]

-- Constructs a composite role-based identifier for detailed record comparison.

ID2\_Role\_Name =

[DEPT] & "|" &

[FIRST\_NAME] & "|" &

[LASTNAME] & "|" &

[STAFF\_ID] & "|" &

[PERSON\_ID] & "|" &

[ASSIGNMENT\_ID] & "|" &

[JOB\_TITLE] & "|" &

[Grade] & "|" &

FORMAT([FTE], "0") & "|" &

[SUPERVISOR\_NAME] & "|" &

[IS\_LINE\_MANAGER]

-- Flags if an employee in FACULTY\_B is matched in Staff Data based on employee number.

Employee Match Status =

VAR EmpNum = 'FACULTY\_B'[STAFF\_ID]

VAR Employee = LOOKUPVALUE('Staff Data'[STAFF\_ID], 'Staff Data'[STAFF\_ID], EmpNum)

RETURN

IF(

ISBLANK(EmpNum),

BLANK(),

IF(

ISBLANK(Employee),

"No Match",

"Match"

)

)

-- Returns a clean and valid FTE value or flags invalid cases with 2.

Final\_Validated\_FTE 1 =

VAR Emp = 'FACULTY\_B'[STAFF\_ID]

VAR EmpCount =

CALCULATE(

COUNTROWS('FACULTY\_B'),

FILTER('FACULTY\_B', 'FACULTY\_B'[STAFF\_ID] = Emp)

)

RETURN

SWITCH(

TRUE(),

EmpCount > 1,2,

'FACULTY\_B'[Final\_Validated\_FTE] > 0 && 'FACULTY\_B'[Final\_Validated\_FTE] <= 1, 'FACULTY\_B'[Final\_Validated\_FTE],

NOT(ISBLANK('FACULTY\_B'[Final\_Validated\_FTE])) && ('FACULTY\_B'[Final\_Validated\_FTE] > 1 || 'FACULTY\_B'[Final\_Validated\_FTE] = 0), 2,

ISBLANK('FACULTY\_B'[Final\_Validated\_FTE]) && NOT(ISBLANK('FACULTY\_B'[Ease FTE])) &&

'FACULTY\_B'[Ease FTE] > 0 && 'FACULTY\_B'[Ease FTE] <= 1, 'FACULTY\_B'[Ease FTE],

ISBLANK('FACULTY\_B'[Final\_Validated\_FTE]) && NOT(ISBLANK('FACULTY\_B'[Ease FTE])) &&

('FACULTY\_B'[Ease FTE] > 1 || 'FACULTY\_B'[Ease FTE] = 0), 2,

ISBLANK('FACULTY\_B'[Final\_Validated\_FTE]) && ISBLANK('FACULTY\_B'[Ease FTE]) && NOT(ISBLANK('FACULTY\_B'[FTE])) &&

'FACULTY\_B'[FTE] > 0 && 'FACULTY\_B'[FTE] <= 1, 'FACULTY\_B'[FTE],

ISBLANK('FACULTY\_B'[Final\_Validated\_FTE]) && ISBLANK('FACULTY\_B'[Ease FTE]) && NOT(ISBLANK('FACULTY\_B'[FTE])) &&

('FACULTY\_B'[FTE] > 1 || 'FACULTY\_B'[FTE] = 0), 2,

NOT(ISBLANK('FACULTY\_B'[FTE])) && NOT(ISBLANK('FACULTY\_B'[Ease FTE])) &&

('FACULTY\_B'[FTE] + 'FACULTY\_B'[Ease FTE] > 1), 2,

ISBLANK(FACULTY\_B[EASE FTE]) && ISBLANK(FACULTY\_B[Final\_Validated\_FTE]) && ISBLANK(FACULTY\_B[FTE]), 2,

FACULTY\_B[EASE FTE] > FACULTY\_B[FTE],2

)

-- Identifies records as 'Vacant' in FACULTY\_B dataset based on name, colour code, or employee number.

Vacants Check =

SWITCH(TRUE(),

FACULTY\_B[FIRST\_NAME] = "vacant" && FACULTY\_B[LASTNAME] = "vacant", "Vacant",

FACULTY\_B[Colour Code] = 2,"Vacant",

FACULTY\_B[STAFF\_ID] = "vacant","Vacant",

"Non Vacant"

)

-- Flags duplicate STAFF\_ID entries in FACULTY\_A dataset.

EmployeeNo\_Duplicate\_Check =

VAR CurrentValue = 'FACULTY\_A'[STAFF\_ID]

RETURN

IF(

ISBLANK(CurrentValue),

FALSE(),

CALCULATE(

COUNTROWS('FACULTY\_A'),

FILTER('FACULTY\_A', 'FACULTY\_A'[STAFF\_ID] = CurrentValue)

) > 1

)

-- Flags duplicate POST ID entries in FACULTY\_A dataset.

PostID\_Duplicate\_Check =

VAR CurrentValue = 'FACULTY\_A'[POST ID ]

RETURN

IF(

ISBLANK(CurrentValue),

FALSE(),

CALCULATE(

COUNTROWS('FACULTY\_A'),

FILTER('FACULTY\_A', 'FACULTY\_A'[POST ID ] = CurrentValue)

) > 1

)

-- Generates anonymised code for each employee in FACULTY\_A to pseudonymise identity.

Anonymised Code =

VAR Emp = 'FACULTY\_A'[STAFF\_ID]

VAR PostA = 'FACULTY\_A'[POST ID ]

VAR FirstName = 'FACULTY\_A'[FIRST\_NAME]

RETURN

IF (

ISBLANK(Emp),

BLANK(),

VAR BaseCode = "A" & RIGHT(Emp, 6)

VAR UniqueCombos =

SUMMARIZE(

FILTER(

'FACULTY\_A',

'FACULTY\_A'[STAFF\_ID] = Emp

),

'FACULTY\_A'[POST ID ],

'FACULTY\_A'[FIRST\_NAME]

)

VAR UniqueComboCount = COUNTROWS(UniqueCombos)

VAR CurrentCombo = PostA & "|" & FirstName

VAR ComboWithIndex =

ADDCOLUMNS(

UniqueCombos,

"ComboKey", 'FACULTY\_A'[POST ID ] & "|" & 'FACULTY\_A'[FIRST\_NAME]

)

VAR Index =

RANKX(

ComboWithIndex,

[ComboKey],

CurrentCombo,

ASC,

DENSE

)

RETURN

IF(

UniqueComboCount = 1,

BaseCode,

BaseCode & "\_" & Index

)

)

-- Builds a concatenated ID1 identifier from key fields in FACULTY\_A dataset.

ID1\_Name = [DEPT] & "|" & [STAFF\_ID] & "|" & [ASSIGNMENT\_ID] & "|" & [LASTNAME]

-- Validates and returns the most appropriate FTE value for an employee or flags with 2.

Final\_Validated\_FTE 1 =

VAR Emp = 'FACULTY\_A'[STAFF\_ID]

VAR EmpCount =

CALCULATE(

COUNTROWS('FACULTY\_A'),

FILTER('FACULTY\_A', 'FACULTY\_A'[STAFF\_ID] = Emp)

)

RETURN

SWITCH(

TRUE(),

EmpCount > 1, 2,

'FACULTY\_A'[Final\_Validated\_FTE] > 0 && 'FACULTY\_A'[Final\_Validated\_FTE] <= 1, 'FACULTY\_A'[Final\_Validated\_FTE],

NOT(ISBLANK('FACULTY\_A'[Final\_Validated\_FTE])) && ('FACULTY\_A'[Final\_Validated\_FTE] > 1 || 'FACULTY\_A'[Final\_Validated\_FTE] = 0), 2,

ISBLANK('FACULTY\_A'[Final\_Validated\_FTE]) && NOT(ISBLANK('FACULTY\_A'[Validated\_FTE])) &&

'FACULTY\_A'[Validated\_FTE] > 0 && 'FACULTY\_A'[Validated\_FTE] <= 1, 'FACULTY\_A'[Validated\_FTE],

ISBLANK('FACULTY\_A'[Final\_Validated\_FTE]) && NOT(ISBLANK('FACULTY\_A'[Validated\_FTE])) &&

('FACULTY\_A'[Validated\_FTE] > 1 || 'FACULTY\_A'[Validated\_FTE] = 0), 2,

ISBLANK('FACULTY\_A'[Final\_Validated\_FTE]) && ISBLANK('FACULTY\_A'[Validated\_FTE]) && NOT(ISBLANK('FACULTY\_A'[FTE])) &&

'FACULTY\_A'[FTE] > 0 && 'FACULTY\_A'[FTE] <= 1, 'FACULTY\_A'[FTE],

ISBLANK('FACULTY\_A'[Final\_Validated\_FTE]) && ISBLANK('FACULTY\_A'[Validated\_FTE]) && NOT(ISBLANK('FACULTY\_A'[FTE])) &&

('FACULTY\_A'[FTE] > 1 || 'FACULTY\_A'[FTE] = 0), 2,

NOT(ISBLANK('FACULTY\_A'[FTE])) && NOT(ISBLANK('FACULTY\_A'[Validated\_FTE])) &&

('FACULTY\_A'[FTE] + 'FACULTY\_A'[Validated\_FTE] > 1), 2,

ISBLANK('FACULTY\_A'[Validated\_FTE]) && ISBLANK('FACULTY\_A'[Final\_Validated\_FTE]) && ISBLANK('FACULTY\_A'[FTE]), 2,

'FACULTY\_A'[Validated\_FTE] > 'FACULTY\_A'[Final\_Validated\_FTE], 2

)

-- Compares STAFF\_ID from FACULTY\_A with Staff Data to confirm match status.

Employee Match Status =

VAR EmpNum = 'FACULTY\_A'[STAFF\_ID]

VAR Employee = LOOKUPVALUE('Staff Data'[STAFF\_ID], 'Staff Data'[STAFF\_ID], EmpNum)

RETURN

IF(

ISBLANK(EmpNum),

BLANK(),

IF(

ISBLANK(Employee),

"No Match",

"Match"

)

)

-- Flags FACULTY\_A rows as 'Vacant' using multiple business logic checks.

Vacants Check =

SWITCH(TRUE(),

'FACULTY\_A'[FIRST\_NAME] = "vacant" && 'FACULTY\_A'[LASTNAME] = "vacant", "Vacant",

'FACULTY\_A'[Colour Code] = 2, "Vacant",

'FACULTY\_A'[STAFF\_ID] = "vacant", "Vacant",

"Non Vacant"

)

### 3. **Field Values**

For anything like:

dax

CopyEdit

'Raw Data'[FIRST\_NAME] = "vacant"

You can:

* Keep "vacant" if it's part of your logic (just document that it's a placeholder used in the simulation).
* Add a note in your GitHub README explaining how "vacant" is a **synthetic flag value** used for logic filtering.