Analysis of usp_AddressSearch

Different Types of Inputs and Outputs:

```
exec usp_AddressSearch 'FLAT 316 67 LEWISHAM HIGH STREET, LONDON, SE13 5JX','SE13 5JX',','
exec usp_AddressSearch 'FLAT 2 TORRINGTON PARK, LONDON, N12 9SS','N12 9SS',','
exec usp_AddressSearch 'FLAT 4/2,16 WHIMBREL WAY BRAEHEAD RENFREW,FLAT 4/2,16 WHIMBREL WAY BRAEHEAD RENFREW,PA4 8SJ','PA4 8SJ',','
exec usp_AddressSearch 'Flat 3,1a Wellington Square Hastings,TN34 1PB','TN34 1PB',','
exec usp_AddressSearch 'Flat 4 , Spectrum House,9-11 Tufton Street,Ashford Kent,TN23 1QN','TN23 1QN',','
```

Results Messages

UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name	Organisation_Name	Department_Name	PO_Box	Postcode_Type	SU_Org_Ind
54178083	SE13 5JX	LONDON	NULL	NULL	Lewisham High Street	NULL	0	Tower House 65-71	Apartment 316	NULL	NULL	NULL	S	

UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name	Organisation_Name	Department_Name	PO_Box	Postcode_Type	SU_Org_Indicato
57263984	N12 9SS	LONDON	NULL	NULL	Torrington Park	NULL	0	2b	Flat 2	NULL	NULL	NULL	S	

UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name	Organisation_Name	Department_Name	PO_Box	Postcode_Type	SU_Org_Indica
28263948	PA4 8SJ	RENFREW	NULL	NULL	Whimbrel Way	NULL	16	NULL	4/2	NULL	NULL	NULL	S	

UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name	Organisation_Name	Department_Name	PO_Box	Postcode_Type	SU_Org_Indic
24790502	TN34 1PB	HASTINGS	NULL	NULL	Wellington Square	NULL	0	1a	Flat 1-10	NULL	NULL	NULL	S	

UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name	Organisation_Name	Department_Name	PO_Box	Postcode_Type	SU_Org
24716040	TN23 1QN	ASHFORD	NULL	NULL	Tufton Street	NULL	0	Spectrum House 9-11	Flat 4	NULL	NULL	NULL	S	

STEP BY STEP Break Down Below:

```
exec usp_AddressSearch 'FLAT 2 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES','NP25 5ES',','
```

100 %

Results Messages

	UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name
1	56031334	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	9	NULL	Flat 2

Organisation_Name	Department_Name	PO_Box	Postcode_Type	SU_Org_Indicator	Delivery_Point_Sfx
NULL	NULL	NULL	S		2J

STEP 1:

```
DECLARE @PostCode VARCHAR(10) = 'NP25 5ES'
DECLARE @Address TABLE
 (UDPRN BIGINT NULL, Postcode VARCHAR(8) NULL, Posttown VARCHAR(30) NULL,
  Dep_Locality VARCHAR(35) NULL, Dbl_Dep_Locality VARCHAR(35) NULL, Thoroughfare_Descrp VARCHAR(80) NULL,
  Dep_Thoroughfare_Descrp VARCHAR(30) NULL, Building_No INT NULL, Building_Name VARCHAR(50) NULL, Sub_Building_Name VARCHAR(50) NULL,
  Organisation_Name VARCHAR(60) NULL, Department_Name VARCHAR(60) NULL, PO_Box VARCHAR(6) NULL, Postcode_Type VARCHAR(1) NULL, SU_Org_Indicator VARCHAR(1) NULL,
  Delivery_Point_Sfx VARCHAR(2) NULL, No_of_Households INT NULL,
  AddressFirstLine VARCHAR(300) NULL, ToMatchValue VARCHAR(100) NULL)

DECLARE @AddressData AS udt_Address

INSERT INTO @Address
 SELECT DISTINCT
 a.UDPRN,a.Postcode,pst.Posttown, depLoc.Dep_Locality,
 dbl_depLoc.Dbl_Dep_Locality, thor.Thoroughfare_Descrp, dep_thor.Dep_Thoroughfare_Descrp,
 a.Building_No, a.Building_Name, a.Sub_Building_Name,
 org.Organisation_Name, dept.Department_Name,a.PO_Box, a.Postcode_Type, a.SU_Org_Indicator,
 a.Delivery_Point_Sfx, a.No_of_Households,
 CASE WHEN ISNULL(a.Building_No, 0) > 0 THEN CONVERT(Varchar(10),ISNULL(a.Building_No, '')) ELSE '' END + ' ' +
 CASE WHEN ISNULL(a.Building_Name, '') <> '' THEN a.Building_Name ELSE '' END + ' ' +
 CASE WHEN ISNULL(thor.Thoroughfare_Descrp, '') <> '' THEN thor.Thoroughfare_Descrp ELSE '' END + ' ' +
 CASE WHEN ISNULL(dep_thor.Dep_Thoroughfare_Descrp, '') <> '' THEN dep_thor.Dep_Thoroughfare_Descrp ELSE '' END + ' ' +
 CASE WHEN ISNULL(depLoc.Dep_Locality, '') <> '' THEN depLoc.Dep_Locality ELSE '' END + ' ' +
 CASE WHEN ISNULL(dbl_depLoc.Dbl_Dep_Locality, '') <> '' THEN dbl_depLoc.Dbl_Dep_Locality ELSE '' END
 as 'AddressFirstLine',
 REPLACE(
 CASE WHEN ISNULL(a.Building_No, 0) > 0 THEN CONVERT(Varchar(10),ISNULL(a.Building_No, '')) ELSE '' END + ' ' +
 CASE WHEN ISNULL(a.Building_Name, '') <> '' THEN a.Building_Name ELSE '' END + ' ' +
 CASE WHEN ISNULL(thor.Thoroughfare_Descrp, '') <> '' THEN thor.Thoroughfare_Descrp ELSE '' END + ' ' +
 CASE WHEN ISNULL(dep_thor.Dep_Thoroughfare_Descrp, '') <> '' THEN dep_thor.Dep_Thoroughfare_Descrp ELSE '' END + ' ' +
 CASE WHEN ISNULL(depLoc.Dep_Locality, '') <> '' THEN depLoc.Dep_Locality ELSE '' END + ' ' +
 CASE WHEN ISNULL(dbl_depLoc.Dbl_Dep_Locality, '') <> '' THEN dbl_depLoc.Dbl_Dep_Locality ELSE '' END, '  ', '')
 as 'ToMatchValue'
 FROM [Address]  a WITH(NOLOCK)
 LEFT JOIN Organisation org WITH(NOLOCK) ON a.OrganisationId= org.ID
 LEFT JOIN Department dept WITH(NOLOCK) ON a.DepartmentId = dept.ID AND org.ID=dept.OrganisationId
 LEFT JOIN DependentLocality depLoc WITH(NOLOCK) ON a.DependentLocalityId=depLoc.ID
 LEFT JOIN DoubleDependentLocality dbl_depLoc WITH(NOLOCK) ON a.DblDependentLocalityId=dbl_depLoc.ID AND depLoc.ID=dbl_depLoc.DepLocalityId
 LEFT JOIN Thoroughfare thor WITH(NOLOCK) ON a.ThoroughfareId=thor.ID
 LEFT JOIN DependentThoroughfare dep_thor WITH(NOLOCK) ON a.DepThoroughfareId=dep_thor.ID AND thor.ID=dep_thor.ThoroughfareId
 LEFT JOIN PostTown pst WITH(NOLOCK) ON a.PosttownId=pst.ID
 WHERE a.Postcode = @PostCode

INSERT INTO @AddressData
 SELECT * FROM @Address
 --SELECT * FROM @AddressData
```

This section sets up the initial environment to analyze address data based on a given postcode. The logic is part of the [dbo].[usp_AddressSearch] stored procedure, which aims to retrieve and construct address details by joining multiple related address tables.

- A local variable @PostCode is declared and set (e.g., 'N12 5ES').
- A table variable @Address is defined to hold temporary address results with detailed fields (e.g., UDPRN, Postcode, Building Name, Organisation Name, etc.).
- The procedure pulls data using joins across various reference tables like Organisation, Department, DoubleDependentLocality, Thoroughfare, etc.
- It uses ISNULL() and conditional logic to construct a formatted address string, saved as:
  - AddressFirstLine
  - ToMatchValue

Input Provided:

- **@PostCode:**
  A string input used to filter the address records.

  Example: *'N12 5ES'*

OUTPUT of @AddressData

Results | Messages

	UDPRN	Postcode	Posttown	Dep_Locality	Dbl_Dep_Locality	Thoroughfare_Descrp	Dep_Thoroughfare_Descrp	Building_No	Building_Name	Sub_Building_Name	Organisation_Name	Department_Name	PO_Box	Postcode_Type
1	17266635	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	0	St. Thomas Vicarage	NULL	NULL	NULL	NULL	S
2	17266636	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	12	NULL	NULL	NULL	NULL	NULL	S
3	17266637	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	14	NULL	NULL	NULL	NULL	NULL	S
4	17266638	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	19	NULL	NULL	NULL	NULL	NULL	S
5	17266639	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	25	NULL	NULL	NULL	NULL	NULL	S
6	17266640	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	27	NULL	NULL	NULL	NULL	NULL	S
7	17266641	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	29	NULL	NULL	NULL	NULL	NULL	S
8	17266642	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	7	NULL	NULL	NULL	NULL	NULL	S
9	17266629	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	23	NULL	Flat	NULL	NULL	NULL	S
10	17266630	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	9	NULL	Flat 1	NULL	NULL	NULL	S
11	17266632	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	0	Overmonnow House	NULL	NULL	NULL	NULL	S
12	17266633	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	0	St. Thomas Church Hall	NULL	NULL	NULL	NULL	S
13	52496961	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	0	7a	NULL	NULL	NULL	NULL	S
14	56031315	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	0	13a	Flat 1	NULL	NULL	NULL	S
15	56031332	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	0	13a	Flat 2	NULL	NULL	NULL	S
16	56031334	NP25 5ES	MONMOUTH	NULL	NULL	St. Thomas Square	NULL	9	NULL	Flat 2	NULL	NULL	NULL	S

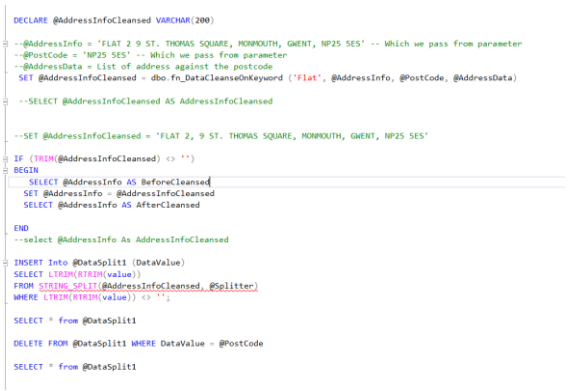
Output Summary – @AddressData Table

This output displays the result of inserting address records based on the input postcode (NP25 5ES) into the @AddressData table.

- Total Records: 16 address entries
- Postcode: All records are from NP25 5ES, located in MONMOUTH.
- Thoroughfare: All records belong to St. Thomas Square.

STEP 2:

Cleansing the Input Address Data



```
DECLARE @AddressInfoCleansed VARCHAR(200)

--@AddressInfo = 'FLAT 2 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES' -- Which we pass from parameter
--@PostCode = 'NP25 5ES' -- Which we pass from parameter
--@AddressData = List of address against the postcode
 SET @AddressInfoCleansed = dbo.fn_DataCleanseOnKeyword ('Flat', @AddressInfo, @PostCode, @AddressData)

  --SELECT @AddressInfoCleansed AS AddressInfoCleansed


--SET @AddressInfoCleansed = 'FLAT 2, 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES'

IF (TRIM(@AddressInfoCleansed) <> '')
BEGIN
    SELECT @AddressInfo AS BeforeCleansed
   SET @AddressInfo = @AddressInfoCleansed
   SELECT @AddressInfo AS AfterCleansed

END
--select @AddressInfo As AddressInfoCleansed

INSERT Into @DataSplit1 (DataValue)
SELECT LTRIM(RTRIM(value))
FROM STRING_SPLIT(@AddressInfoCleansed, @Splitter)
WHERE LTRIM(RTRIM(value)) <> '';

SELECT * from @DataSplit1

DELETE FROM @DataSplit1 WHERE DataValue = @PostCode

SELECT * from @DataSplit1
```

This section handles **data cleansing** of the address input before breaking it into individual components for matching. It utilizes a user-defined function dbo.fn_DataCleanseOnKeyword to remove or standardize keywords such as 'Flat'.

Steps:

Cleansing Function Call

@AddressInfoCleansed is set using the function:
dbo.fn_DataCleanseOnKeyword('Flat', @AddressInfo, @PostCode, @AddressData)

The function fn_DataCleanseOnKeyword is used to **sanitize and reformat address strings** that include flat, unit, or floor-level details. The main goal is to make the address format **consistent and searchable**, particularly to support accurate flat identification when querying address data (e.g., postal or delivery address matching).

Processing Logic:

1. **Initial Validations**:
   Returns empty if @data or @PostCode is null or empty, or if @inputData has no rows.
2. **Extract Floor Info**:
   It calls fn_GetFloorData(@data) to extract any floor-level info (like "First Floor") and stores variations of that data for comparison.
3. **Flat Keyword Detection & Parsing**:
   Searches the @data string for the @SearchValue (e.g., "Flat"), and extracts a substring (@FlatData) that likely contains flat or unit information.
4. **Cleansing Logic**:
   Performs a series of conditional checks to match flat/unit data against the address table (@inputData) using:
   - Exact or partial matches in Sub_Building_Name, Building_Name, or Building_No.
   - Transformations like:
     - Adding/removing “s” in "*Flats*".
     - Handling slash-formatted flat numbers (e.g., "*Flat 1/10*").
     - Reversing word order (e.g., *"Flat First Floor"* → *"First Floor Flat"*).
     - Removing or rearranging floor or unit identifiers.
     - Replacing confusing flat identifiers when no match is found.
5. **Special Cases Handling**
   - Intelligent parsing of formats like Flat 110 as Flat 1/10 or Flat 11/0.

- Removing slashes (/) to handle formats like *Flat 1/10 → Flat 110.*
- Numeric-only matching for Building_No.
- Sub-block flats where multiple units share one building number.

6. **Final Replacement**:
   If a match is found (@DataToReplace), it replaces the flat section in the original address with the corrected/cleaned one. If not, it returns the address as-is.

If Cleansing is Successful:

- It checks if the cleansed value is not empty.
- Logs the address before and after cleansing using SELECT.

Splitting Cleaned Address:

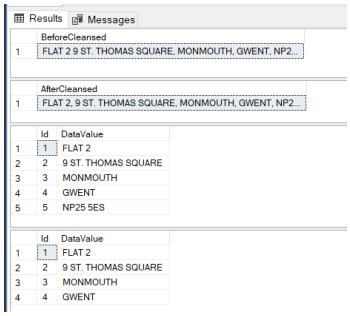
- Cleansed string is split into components using STRING_SPLIT() with the given delimiter (@Splitter).
- The results are inserted into @DataSplit1.

Filtering Out Postcode:
The postcode value is removed from the split list to ensure only address parts remain.

Inputs Passed:

- **@AddressInfo:**
  'FLAT 2, 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES'
  (from parameter – user input)
- **@PostCode:**
  'NP25 5ES'
  (from parameter – input postcode to match address)
- **@AddressData:**
  Temporary address list populated from earlier step, containing multiple address records for given postcode.
- **@Splitter:**
  Delimiter to split address values – typically a comma , or space ' '.

Output of Cleansing & Token Split:



Output Summary – Cleansing & Token Split:

Before Cleansed:
Raw input:
FLAT 2 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES

After Cleansed:
Formatted version with commas for easier parsing:
FLAT 2, 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES

Split Result (Initial):
Extracted address parts:

1. FLAT 2
2. 9 ST. THOMAS SQUARE
3. MONMOUTH
4. GWENT
5. NP25 5ES

After Filtering Out Postcode:
Final address tokens used for matching:

1. FLAT 2
2. 9 ST. THOMAS SQUARE
3. MONMOUTH
4. GWENT

STEP 3:

Matching Building_Name with Cleansed Address

```
Declare @TestSubBuildingName VARCHAR(100)
Declare @TestBuildingNo Varchar(10)
Declare @TestBuildingName Varchar(100)

DECLARE @DataToCheckFromDatabase TABLE (ID INT IDENTITY(1,1), FieldValue varchar(100), isProcessed INT)

INSERT INTO @DataToCheckFromDatabase (FieldValue, isProcessed)
SELECT DISTINCT Building_Name, 0 FROM @Address WHERE Building_Name IS NOT NULL

DECLARE @ProcessingRecordId INT
DECLARE @ProcessingDataValue Varchar(100)

WHILE ((SELECT COUNT(*) FROM @DataToCheckFromDatabase WHERE isProcessed = 0)>0)
BEGIN
  SELECT TOP 1 @ProcessingRecordId = ID, @ProcessingDataValue = FieldValue
  FROM @DataToCheckFromDatabase WHERE isProcessed = 0

  IF (CHARINDEX(' ' + @ProcessingDataValue + ' ', ' ' + @AddressInfoCleansed) > 0)
  BEGIN
    SET @BuildingName = @ProcessingDataValue
  END

 -- PRINT '@ProcessingRecordId : ' + COnvert(varchar(10), @ProcessingRecordId)
  --PRINT '@ProcessingDataValue : ' + @ProcessingDataValue
  --PRINT '@BuildingName : ' + @BuildingName

  UPDATE @DataToCheckFromDatabase SET isProcessed = 1 WHERE ID = @ProcessingRecordId
END
```

This logic identifies whether any building name from the address database appears in the **cleansed address input** string. It helps in extracting exact matches for downstream logic like scoring or filtering.

🔧 Process Steps:

1. **Variables:**
   - @BuildingName — stores matched building name (if found).
   - @AddressInfoCleansed — cleansed version of the user-provided address.
   - @DataToCheckFromDatabase — table to hold distinct building names for processing.

2. **Prepare the List to Check:**
    - Inserts into a temporary table with a flag (isProcessed) for tracking.
3. **Loop Logic:**
    - Picks one unprocessed Building_Name at a time.
    - Checks if that name exists inside @AddressInfoCleansed using CHARINDEX.
    - If matched, sets @BuildingName = @ProcessingDataValue.
    - Marks the current row as processed.

Inputs Provided:

Input Name	Example Value	Description
@AddressInfoCleansed	'FLAT 2, 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES'	Cleaned input address from user or front-end.
@Address (table)	Contains rows with fields like Building_Name	Populated earlier with address data for a given postcode.
@DataToCheckFromDatabase	Extracted from @Address, e.g., "St. Thomas Vicarage", "7a", etc.	Each record checked against input address for partial match.

Sample Matched Case:

If the cleansed input contains "St. Thomas Vicarage" and the same exists in @Address, it gets assigned to @BuildingName.

Output of Building Name Match from Cleansed Address:

	ID	FieldValue	isProcessed
1	1	11-13	1
2	2	13a	1
3	3	7a	1
4	4	Overmonnow House	1
5	5	St. Thomas Church Hall	1
6	6	St. Thomas Vicarage	1

Output summary:

- The process scans through each Building_Name present in the @Address table.
- Each name is checked against the **cleansed address string** @AddressInfoCleansed.
- If a Building_Name is found inside the cleansed string, it is stored in the variable @BuildingName.

STEP 4:

Cleansing Address and Preparing for PostTown Matching:

```
IF (TRIM(@TestBuildingName) <> '')
BEGIN
  SET @AddressInfo = REPLACE (@AddressInfo, @TestBuildingName, '')
  SET @BuildingName = @TestBuildingName
END

PRINT 'Database Building Name ' + ISNULL(@BuildingName, '')

SET @AddressInfo = REPLACE (@AddressInfo, 'C/O', '')

INSERT Into @AddressSplitTemp
SELECT LTRIM(RTRIM(value))
FROM STRING_SPLIT(@AddressInfo, @Splitter)
WHERE LTRIM(RTRIM(value)) <> '';

INSERT INTO @AddressSplit
SELECT DISTINCT RTRIM(LTRIM(AddressValue)) from @AddressSplitTemp

DELETE FROM @AddressSplit WHERE AddressValue = LTRIM(RTRIM(@PostCode))

/*Data Identification Section (Posttown, Locality, Thoroughfare, Organisation, FirstLine, Build No, Building Name Start*/
INSERT INTO @PostTownList
SELECT Id, Posttown, 0 from PostTown
INNER JOIN @AddressSplit a ON LTRIM(RTRIM(PostTown.Posttown)) = LTRIM(RTRIM(a.AddressValue))
```

This section refines the input address string by removing identified and unnecessary parts, splits the address into clean segments, and begins the process of matching known address components against reference datasets.

Building Name Removal:
If a building name (@TestBuildingName) was identified earlier (from the building name match logic), and it is not empty:

- It is removed from the @AddressInfo string using REPLACE.
- The matched name is assigned to @BuildingName.

Removing “C/O”:
The string "C/O" (common in mailing addresses) is removed from @AddressInfo to clean up unnecessary tokens before splitting.

Splitting the Address into Components:

- The modified @AddressInfo string is split into parts using the provided delimiter (@Splitter, such as comma ,).
- The split values are trimmed and inserted into a temporary table @AddressSplitTemp.
- From there, distinct values are inserted into @AddressSplit.

Filtering Out the Postcode:

- Any value in the split list that matches the postcode (@PostCode) is removed from @AddressSplit, as it's not needed for component matching.

Matching Tokens Against PostTown Table:

- The cleaned address parts in @AddressSplit are compared against entries in the PostTown reference table.
- If a match is found, the corresponding ID and post town name are inserted into @PostTownList, which is used later for scoring or assembling the matched address.

Inputs Used:

Parameter / Table	Purpose
`@AddressInfo`	Cleansed input address string from the user
`@TestBuildingName`	Previously identified building name (if any)
`@PostCode`	Postcode from the user input, used for filtering
`@Splitter`	Character used to split the address string (e.g., comma)
`@AddressSplitTemp`	Temporary table to hold raw split tokens
`@AddressSplit`	Cleaned and distinct address tokens used for matching
`PostTown` (table)	Reference table to check for post town names
`@PostTownList`	Stores matched post towns with their IDs

OUTPUT:

	AddressInfo
1	FLAT 2, 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP...

	AddressValue
1	9 ST. THOMAS SQUARE
2	FLAT 2
3	GWENT
4	MONMOUTH
5	NP25 5ES

	AddressValue
1	9 ST. THOMAS SQUARE
2	FLAT 2
3	GWENT
4	MONMOUTH

	PosttownId	PostTown	DType
1	3910	MONMOUTH	0

Expected Output:

- **@AddressInfo**: Updated address string with building name and "C/O" removed
- **@BuildingName**: Contains the identified building name
- **@AddressSplit**: Contains unique, trimmed address tokens excluding the postcode
- **@PostTownList**: Holds rows where address tokens matched valid post towns from the PostTown table.

STEP 5:

Post Town Identification

```
SELECT @PostTownCount = Count(*) FROM @PostTownList

IF (@PostTownCount = 0)
BEGIN
    IF (LTRIM(RTRIM(@AdditionalDataToCheck)) <> '')
    BEGIN
        INSERT INTO @PostTownList
        SELECT Id, Posttown, 0 FROM PostTown WHERE Posttown = LTRIM(RTRIM(@AdditionalDataToCheck))

        SELECT @PostTownCount = Count(*) FROM @PostTownList
    END
END
```

Inputs:

@PostTownList = Empty
@AdditionalDataToCheck = 'GWENT'

Outputs:

@PostTownList after insert = {*Id: 501, Posttown: 'GWENT', Flag: 0}*

@PostTownCount = *1*

Summary:

Post Town 'GWENT' identified from additional data and added to list.

STEP 6:

Remove Identified Post Town from Input

```
DELETE a
FROM @AddressSplit a
INNER JOIN @PostTownList P ON LTRIM(RTRIM(a.AddressValue)) = LTRIM(RTRIM(P.PostTown))
```

Inputs:

@PostTownList = Empty
@AdditionalDataToCheck = 'GWENT'

Outputs:

@PostTownList = {Id: 501, Posttown: 'GWENT', Flag: 0}
@PostTownCount = 1

Summary:
Post Town 'GWENT' identified from additional data and added to list.

STEP 7:

Locality Identification

```
INSERT INTO @LocalityList
SELECT DISTINCT Id, Dep_Locality
FROM DependentLocality d
INNER JOIN @AddressSplit a ON LTRIM(RTRIM(d.Dep_Locality)) = LTRIM(RTRIM(a.AddressValue))
INNER JOIN [Address] adr ON adr.DependentLocalityId = d.ID
WHERE adr.Postcode = @PostCode

DELETE a
FROM @AddressSplit a
INNER JOIN @LocalityList T ON LTRIM(RTRIM(a.AddressValue)) = LTRIM(RTRIM(T.Locality))
```

Inputs:

@AddressSplit = {‘FLAT 2', '9 ST. THOMAS SQUARE', 'MONMOUTH’}
DependentLocality = {Id: 301, Dep_Locality: 'MONMOUTH’}
@PostCode = 'NP25 5ES'

Outputs:

@LocalityList = {Id: 301, Locality: 'MONMOUTH'}
@AddressSplit = {‘FLAT 2', '9 ST. THOMAS SQUARE’}

Summary:

Identifies 'MONMOUTH' as locality and removes it from input.

STEP 8:

Initial Building Name Identification

```
IF (LTRIM(RTRIM(@BuildingName)) = '')
BEGIN
    IF ((SELECT COUNT(DISTINCT ap.Building_Name)
         FROM @Address ap
         INNER JOIN @AddressSplit as1 ON as1.AddressValue LIKE CONCAT(ap.Building_Name,'%')
         WHERE ap.Building_Name IS NOT NULL) = 1)
    BEGIN
        SET @BuildingName = (SELECT DISTINCT ap.Building_Name
                             FROM @Address ap
                             INNER JOIN @AddressSplit as1 ON as1.AddressValue LIKE
CONCAT(ap.Building_Name,'%')
                             WHERE ap.Building_Name IS NOT NULL)
    END
END
```

Inputs:

@BuildingName = ''
@AddressSplit = {'FLAT 2', '9 ST. THOMAS SQUARE'}
@Address Table contains {Building_Name = '9 ST. THOMAS SQUARE'}

Outputs:

@BuildingName = '9 ST. THOMAS SQUARE'

Summary:

Building name '9 ST. THOMAS SQUARE' identified from input.

STEP 9:

Clean Building Name from Input

```
IF (LTRIM(RTRIM(@BuildingName)) <> '')
BEGIN
    UPDATE @AddressSplit SET AddressValue = TRIM(REPLACE(TRIM(AddressValue), TRIM(@BuildingName), ''))
END
```

Inputs:

@AddressSplit = {'FLAT 2', '9 ST. THOMAS SQUARE'}
@BuildingName = '9 ST. THOMAS SQUARE'

Outputs:

@AddressSplit = {'FLAT 2', ''}

Summary:

Removes building name '9 ST. THOMAS SQUARE' from address input.

STEP 10:

Sub Building Name Identification

```
IF (LTRIM(RTRIM(@SubBuildingName)) = '')
BEGIN
    SELECT @SubBuildingName = dbo.fn_GetDataForSubBuilding(addressvalue, 'FLAT')
    FROM @AddressSplit WHERE addressvalue LIKE '%FLAT%'
END
```

Inputs:

@AddressSplit = {'FLAT 2', ''}
@SubBuildingName = ''

Outputs:

@SubBuildingName = 'FLAT 2'

Summary:

Extracts sub-building name 'FLAT 2' from input.

STEP 11:

Remove Sub Building Name & Extract Flat Number

```
IF (LTRIM(RTRIM(@SubBuildingName)) <> '')
BEGIN
    UPDATE @AddressSplit SET AddressValue = TRIM(REPLACE(TRIM(AddressValue),
TRIM(@SubBuildingName), ''))
    SET @FlatValue = TRIM(REPLACE(@SubBuildingName, 'flat', ''))
    IF (ISNUMERIC(@FlatValue) = 1)
    BEGIN
        SET @FlatNo = CONVERT(INT, @FlatValue)
    END
END
```

Inputs:

@AddressSplit = {‘FLAT 2', ''}
@SubBuildingName = 'FLAT 2'

Outputs:

@AddressSplit = {'', ''}
@FlatValue = '2'
@FlatNo = 2

Summary:

Removes 'FLAT 2' from input and extracts flat number 2.

STEP 12:

Direct Building Name Match from Input

```
IF (TRIM(@BuildingName) = '')
BEGIN
    SELECT TOP 1 @BuildingName = Building_Name
    FROM @Address A
    INNER JOIN @AddressSplit AST ON A.Building_Name = AST.AddressValue
END
```

Example Inputs:

@BuildingName = ''
@AddressSplit = {'FLAT 2', '9 ST. THOMAS SQUARE', 'MONMOUTH', 'GWENT'}
@Address = {Building_Name = '9 ST. THOMAS SQUARE'}

Example Outputs:

@BuildingName = '9 ST. THOMAS SQUARE'

Summary:

If no building name set, fetch directly from input if exact match with known building names exists.

STEP 13:

Prepare Building Name Check Data

```
DELETE FROM @CheckDataList

INSERT INTO @CheckDataList
SELECT DISTINCT Building_Name
FROM @Address a
INNER JOIN @AddressSplit ap ON ap.AddressValue LIKE CONCAT('%', a.Building_Name ,'%')
WHERE Building_Name IS NOT NULL
```

Example Inputs:

@AddressSplit = {'FLAT 2', '9 ST. THOMAS SQUARE', 'MONMOUTH', 'GWENT'}
@Address = {Building_Name = '9 ST. THOMAS SQUARE', '10 HIGH STREET'}

Example Outputs:

@CheckDataList = {'9 ST. THOMAS SQUARE'}

Summary:

Builds temporary list of possible building name matches from address input for further processing.

STEP 14:

Pattern-Based Building Name Extraction

```
IF (TRIM(@BuildingName) = '')
BEGIN
    SELECT @BuildingName = dbo.fn_GetDataWithPatterns(addressvalue,'%[0-9]%-[0-9]%')
    FROM @AddressSplit WHERE addressvalue LIKE '%[0-9]%-[0-9]%'
END

-- Similar Checks for patterns like:
-- '9/A', '9-A', '9-12' etc.
```

Example Inputs:

@AddressSplit = {'FLAT 2', '9-12 HIGH STREET'}
@BuildingName = ''

Example Outputs:

@BuildingName = '9-12 HIGH STREET'

Summary:

When direct matches fail, attempts to extract building name using known numeric-hyphen or numeric/letter patterns.

STEP 15:

Remove Identified Building Name from Input

```
IF (TRIM(@BuildingName) <> '')
BEGIN
    UPDATE @AddressSplit SET AddressValue = TRIM(REPLACE(TRIM(AddressValue), TRIM(@BuildingName), ''))
END
```

Example Inputs:

@AddressSplit = {'FLAT 2', '9 ST. THOMAS SQUARE'}
@BuildingName = '9 ST. THOMAS SQUARE'

Example Outputs:

@AddressSplit = {‘FLAT 2', ''}

Summary:

Removes identified building name from input list to prevent duplicate processing.

STEP 16:

Alphanumeric Building Name Split (e.g., 11A)

```
IF (TRIM(@BuildingName) = '')
BEGIN
    SELECT TOP 1 @BuildingNameSplit = AddressValue
    FROM @AddressSplit WHERE AddressValue LIKE '%[0-9][A-Z]%'
        AND ISNUMERIC(SUBSTRING(LTRIM(AddressValue), 1, 1)) = 1

    IF(ISNULL(@BuildingNameSplit, '') !='')
    BEGIN
        -- Split & extract building portion
    END
END
```

Example Inputs:

@AddressSplit = {'FLAT 2', '11A HIGH STREET'}

Example Outputs:

@BuildingName = '11A'
@FirstLine = '11A HIGH STREET'
@OtherData = 'HIGH STREET'

Summary:

Handles partial alphanumeric patterns where building names are compact like '11A'.

STEP 17:

Check Data List Match for Building Name

```
IF (TRIM(@BuildingName) = '')
BEGIN
    -- Check @CheckDataList for matches, exact or with space added
END
```

Example Inputs:

@CheckDataList = {'ST. THOMAS SQUARE'}
@AddressSplit = {'FLAT 2', 'ST. THOMAS SQUARE'}

Example Outputs:

@BuildingName = 'ST. THOMAS SQUARE'

Summary:

Uses check list to find building names that partially match remaining input, considering spacing.

STEP 18:

Final Building Name Cleanup & First Line Setup

```
IF (LTRIM(RTRIM(@BuildingName)) <> '')
BEGIN
    SET @FirstLine = @BuildingName
    UPDATE @AddressSplit SET AddressValue = TRIM(REPLACE(TRIM(AddressValue),
TRIM(@BuildingName), ''))
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE'
@AddressSplit = {'FLAT 2', 'ST. THOMAS SQUARE'}

Example Outputs:
@AddressSplit = {'FLAT 2', ''}
@FirstLine = 'ST. THOMAS SQUARE'

Summary:
Marks building name as first line and cleans input list.

STEP 19:

Building Number Extraction

```
IF (@BuildingNo = 0)
BEGIN
    SELECT TOP 1 @BuildingNoSplit = AddressValue
    FROM @AddressSplit WHERE AddressValue NOT LIKE '%[0-9][A-Z]%'
        AND ISNUMERIC(SUBSTRING(LTRIM(AddressValue), 1, 1)) = 1
END
```

Example Inputs:
@AddressSplit = {'12 HIGH STREET'}

Example Outputs:
@BuildingNo = 12
@OtherData = 'HIGH STREET'
@AdditionalDataToCheck = 'HIGH STREET'

Summary:
Extracts numeric building numbers when present, stores remaining data for later checks.

STEP 20:

Organisation Name Identification

```
INSERT INTO @CheckDataList
SELECT DISTINCT Organisation_Name
FROM @Address a
INNER JOIN @AddressSplit ap ON ap.AddressValue LIKE CONCAT('%', a.Organisation_Name ,'%')

-- Attempt exact and space-appended matches
```

Example Inputs:
@AddressSplit = {'ABC ENTERPRISES', 'FLAT 2'}
@Address = {Organisation_Name = 'ABC ENTERPRISES'}

Example Outputs:
@OrganisationName = 'ABC ENTERPRISES'

Summary:
Detects organisation name from input based on known values, with flexible matching logic.

STEP 21:

Sub Building Name Normalization

```
SET @TransformSubBuildingName = REPLACE(@SubBuildingName, 'Apt', 'Apartment')
```

Example Inputs:
@SubBuildingName = 'Apt 2'

Example Outputs:
@TransformSubBuildingName = 'Apartment 2'

Summary:
Standardizes sub-building name variations (e.g., replaces 'Apt' with 'Apartment') for consistency.

STEP 22:

Organisation Name Match

```sql
IF (TRIM(@OrganisationName) <> '')
BEGIN
    IF EXISTS (
        SELECT 'true'
        FROM @Address
        WHERE Organisation_Name = @OrganisationName
          AND ((ISNULL(@BuildingNo, '')) = '' OR Building_No = @BuildingNo)
          AND ((ISNULL(@BuildingName, '')) = '' OR Building_Name = @BuildingName)
    )
    BEGIN
        PRINT 'Running with Organisation Name'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @Address
        WHERE Organisation_Name = @OrganisationName
          AND ((ISNULL(@BuildingNo, '')) = '' OR Building_No = @BuildingNo)
          AND ((ISNULL(@BuildingName, '')) = '' OR Building_Name = @BuildingName)
        RETURN
    END
END
```

Example Inputs:

@OrganisationName = 'ABC ENTERPRISES'
@BuildingNo = ''
@BuildingName = ''
@Address contains Organisation_Name = 'ABC ENTERPRISES'

Example Outputs:

Returns address details for 'ABC ENTERPRISES'

Summary:

If Organisation Name is provided and found in the address dataset (optionally matching Building No or Building Name), returns the address immediately.

STEP 23:

Sub Building Name, Building No and Building Name Match

```sql
IF (LTRIM(RTRIM(@SubBuildingName)) <> '')
BEGIN
    IF (ISNULL(@BuildingNo, 0) > 0 AND LTRIM(RTRIM(@BuildingName)) <> '')
    BEGIN
        PRINT 'Running with Building No and Sub Building Name and Building Name'
        IF EXISTS (
            SELECT 'true'
            FROM @Address
            WHERE Building_No = @BuildingNo
              AND Building_Name = @BuildingName
              AND (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
        )
        BEGIN
            SELECT * FROM @Address
            WHERE Building_No = @BuildingNo
              AND Building_Name = @BuildingName
              AND (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
            RETURN
        END
    END
END
```

Example Inputs:

@SubBuildingName = 'FLAT 2'
@BuildingNo = 9
@BuildingName = 'ST. THOMAS SQUARE'
@TransformSubBuildingName = ''

@Address contains Building_No = 9, Building_Name = 'ST. THOMAS SQUARE', Sub_Building_Name = 'FLAT 2'

Example Outputs:

Returns address for 'FLAT 2, 9 ST. THOMAS SQUARE'

Summary:

Matches address using Sub-Building Name along with Building No and Building Name. If a match exists, returns the address.

STEP 24:

Flat Range Check with Sub Building Name, Building Name, and Building No

```
IF (@FlatNo > 0)
BEGIN
    PRINT 'Check Flat Range in Building_No, Building_Name, SubBuilding No'
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_RunFlatRange(@FlatNo, @BuildingNo, @BuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Flat Range matched with Sub Building name, Building Name and Building no'
        SELECT * FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:

@FlatNo = 2
@BuildingNo = 9
@BuildingName = 'ST. THOMAS SQUARE'
@AddressData = 'FLAT 2, 9 ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:

UDPRN = 123456
Postcode = 'NP25 5ES'
Building_No = 9
Building_Name = 'ST. THOMAS SQUARE'
Sub_Building_Name = 'FLAT 2'

Summary:

Uses a flat number range logic to match flats within a building. If found, returns the full address details.

STEP 25:

Flat No + Alphabet Check in Building Name

```
DELETE FROM @AddressTemp
INSERT INTO @AddressTemp
SELECT * FROM fn_CheckBuildingNoAlpha(@FlatNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Flat no + Alphabet found in building name'
    SELECT * FROM @AddressTemp
    RETURN
END
```

Example Inputs:
@FlatNo = 4
@AddressData = '4A ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:
UDPRN = 654321
Postcode = 'NP25 5ES'
Building_Name = '4A ST. THOMAS SQUARE'

Summary:
Handles cases where flat numbers include alphabets within the building name, e.g., 4A. If match found, returns the address.

STEP 26: Building Range Check with Existing Flat No

```
DELETE FROM @AddressTemp
INSERT INTO @AddressTemp
SELECT * FROM fn_RunBuildingRange(@FlatNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Building range available in building name'
    SELECT * FROM @AddressTemp
    RETURN
END
```

Example Inputs:
@FlatNo = 3
@AddressData = '1-5 ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:
Returns record:
UDPRN = 112233
Building_Name = '1-5 ST. THOMAS SQUARE'

Summary:
Matches buildings where the flat number falls within a specified numeric building range (e.g., 1-5). Returns address if matched.

STEP 27:

Building No + Alphabet Check in Building Name

```
INSERT INTO @AddressTemp
SELECT * FROM fn_CheckBuildingNoAlpha(@BuildingNo, @SubBuildingName, @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Building no + Alphabet found in building name'
    SELECT * FROM @AddressTemp
    RETURN
END
```

Example Inputs:

@BuildingNo = 9
@SubBuildingName = 'FLAT 2'
@AddressData = '9B ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:

Returns record:
UDPRN = 445566
Building_Name = '9B ST. THOMAS SQUARE'

Summary:

Handles building numbers with alphabets, e.g., 9B. Matches against Building Name and returns address if found.

STEP 28:

Building Range Check with Existing Building No

```
IF (ISNULL(@BuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@BuildingNo, @SubBuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Building range available'
        SELECT * FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:

@BuildingNo = 15
@SubBuildingName = 'FLAT 1'
@AddressData = '10-20 ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:

Returns record:
UDPRN = 778899
Building_Name = '10-20 ST. THOMAS SQUARE'

Summary:

Checks if a building number falls within a range (e.g., 10-20) in the Building Name field. If matched, returns address.

STEP 29:

Transformed Building No Extraction from Building Name

```
SET @SanitizedBuildingNo = ''
SET @TransformedBuildingNo = 0

SELECT @SanitizedBuildingNo = dbo.fn_CleanString(@BuildingName, '%[^0-9]%')

IF (TRIM(@SanitizedBuildingNo) <> '' AND ISNUMERIC(@SanitizedBuildingNo) = 1)
BEGIN
    SET @TransformedBuildingNo = CONVERT(INT, @SanitizedBuildingNo)
    PRINT 'Sanitized Building No from Building Name:' + @BuildingName + ' = ' + @SanitizedBuildingNo
END
```

Example Inputs:

@BuildingName = 'BLOCK 25B'

Example Outputs:

@SanitizedBuildingNo = '25'
@TransformedBuildingNo = 25

Summary:

Extracts numeric part from Building Name when present (e.g., 'BLOCK 25B' → 25) for use in subsequent building checks.

STEP 30:

Re-check with Transformed Building No

```
IF (@TransformedBuildingNo > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_ReCheckBuildingNoWithTransformedData(@TransformedBuildingNo, @SubBuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Transformed building no found with valid address'
        SELECT * FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:

@TransformedBuildingNo = 25
@SubBuildingName = 'FLAT 3'
@AddressData = 'BLOCK 25B ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:

UDPRN = 987654
Building_No = 25
Sub_Building_Name = 'FLAT 3'

Summary:

Uses numeric value extracted from Building Name to attempt a precise building number match. Returns address if successful.

STEP 31:

Building Range Check with Transformed Building No

```
IF (ISNULL(@TransformedBuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@TransformedBuildingNo, @SubBuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Building range found with transformed building no'
        SELECT * FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:

@TransformedBuildingNo = 25
@SubBuildingName = 'FLAT 3'
@AddressData = '20-30 ST. THOMAS SQUARE, MONMOUTH'

Example Outputs:

UDPRN = 456789
Building_Name = '20-30 ST. THOMAS SQUARE'
Sub_Building_Name = 'FLAT 3'

Summary:

Performs a building range check using extracted numeric part from Building Name to identify if address falls within valid range.

STEP 32:

Fallback - Check with Sub Building Name and Building Name

```
IF ((SELECT COUNT(*) FROM @Address
    WHERE (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
      AND Building_Name = @BuildingName) = 1)
BEGIN
    PRINT 'Matching with Sub Building Name and Building Name'
    SELECT * FROM @Address
    WHERE Building_Name = @BuildingName
      AND (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
    RETURN
END
```

Example Inputs:

@SubBuildingName = 'FLAT 3'
@BuildingName = 'ST. THOMAS SQUARE'
@TransformSubBuildingName = ''
@Address contains one record matching these values

Example Outputs:

Returns record:
UDPRN = 123789
Sub_Building_Name = 'FLAT 3'
Building_Name = 'ST. THOMAS SQUARE'

Summary:

As fallback, matches based only on Sub Building Name and Building Name combination if exactly one match exists.

STEP 33:

Fallback - Check with Sub Building Name and Building No

```
IF ((SELECT COUNT(*) FROM @Address
    WHERE (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
      AND Building_No = @BuildingNo) = 1)
BEGIN
    PRINT 'Matching with Sub Building Name and Building No'
    SELECT * FROM @Address
    WHERE Building_No = @BuildingNo
      AND (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
    RETURN
END
```

Example Inputs:

@SubBuildingName = 'FLAT 3'
@BuildingNo = 9
@TransformSubBuildingName = ''
@Address contains one record matching these values

Example Outputs:

Returns record:
UDPRN = 369258
Sub_Building_Name = 'FLAT 3'
Building_No = 9

Summary:

Fallback to Sub Building Name and Building No match if exactly one address is found.

STEP 34: Fallback - Check with Building Name and Building No

```
IF ((SELECT COUNT(*) FROM @Address
    WHERE Building_Name = @BuildingName
      AND Building_No = @BuildingNo) = 1)
BEGIN
    PRINT 'Matching with Building Name and Building No'
    SELECT * FROM @Address
    WHERE Building_Name = @BuildingName
      AND Building_No = @BuildingNo
    RETURN
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE'
@BuildingNo = 9
@Address contains one record matching these values

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Final fallback—matches address based on Building Name and Building No combination if only one address is found.

STEP 34:

Flat Range Check with Building Name & Sub Building Name

```
IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as building range available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
ELSE
BEGIN
    PRINT 'No Building Range found for the flat No ' + CONVERT(VARCHAR(10), @FlatNo)
END
```

Example Inputs:

@FlatNo = 4
@BuildingName = 'ST. THOMAS SQUARE'
@SubBuildingName = 'FLAT 4'
@AddressTemp contains 1 record from previous fn_RunBuildingRange matching flat 4 in building name

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'

Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
If building range search based on Flat No, Building Name, and Sub Building Name returns a record, it outputs that record immediately.

STEP 35:

Transformed Building No from Building Name

```
PRINT 'Started - Check the transformed Building No in Building No Field'
SET @TransformedBuildingNo = 0
SELECT @SanitizedBuildingNo = dbo.fn_CleanString(@BuildingName,'%[^0-9]%')

IF (TRIM(@SanitizedBuildingNo) <> '' AND ISNUMERIC(@SanitizedBuildingNo) = 1)
BEGIN
    SET @TransformedBuildingNo = CONVERT(INT, @SanitizedBuildingNo)
    PRINT 'Sanitized Building No from Building Name data :' + @BuildingName + ' = ' + @SanitizedBuildingNo
END
```

Example Inputs:

@BuildingName = '9 ST. THOMAS SQUARE'

Example Outputs:

@SanitizedBuildingNo = '9'
@TransformedBuildingNo = 9

Summary:
Extracts numeric portion from Building Name for further building number checks.

STEP 36:

Single Instance Check with Transformed Building No

```
IF (@TransformedBuildingNo > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_ReCheckBuildingNoWithTransformedData(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data transformed building no available in building_no with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No data available for transformed building no in building_no with no_of_house hold > 1 and with one instance'
    END
END
```

Example Inputs:

@TransformedBuildingNo = 9
@AddressData = Address split array containing address details

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Uses extracted building number to search for a single matching record with valid household count, returns if found.

STEP 37:

Building Range Check with Transformed Building No

```
IF (ISNULL(@TransformedBuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data as building range available in building_name with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @TransformedBuildingNo)
    END
END
```

Example Inputs:

@TransformedBuildingNo = 9
@AddressData contains relevant address fragments

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Checks building range logic with extracted numeric portion from building name, returns matching record if available.

STEP 38:

Direct Match with Building Name

```
IF ((SELECT COUNT(*) FROM @Address WHERE Building_Name = @BuildingName) = 1)
BEGIN
    PRINT 'Matching with Building Name'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_Name = @BuildingName
    RETURN
END
```

Example Inputs:

@BuildingName = 'ST. THOMAS SQUARE'
@Address contains one record with Building_Name = 'ST. THOMAS SQUARE'

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
If exactly one record matches the building name, return that record immediately.

STEP 39:

Building No and Sub Building Name Matching

```
ELSE IF (ISNULL(@BuildingNo, 0) > 0)
BEGIN
    PRINT 'Running with Building No and Sub Building Name ' + CONVERT(VARCHAR(10), ISNULL(@BuildingNo, 0)) + ', ' + @SubBuildingName
    IF EXISTS (SELECT 'true' FROM @Address WHERE Building_No = @BuildingNo
              AND (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName))
    BEGIN
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @Address
        WHERE Building_No = @BuildingNo AND (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Data found for building no, sub building name, Checking flat Range'
    END
END
```

Example Inputs:

@BuildingNo = 9
@SubBuildingName = 'FLAT 4'
@TransformSubBuildingName = NULL

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:

If Building No and Sub Building Name combination matches any record, return that record.

STEP 40:

Flat Range Check with Sub Building Name and Building No

```
IF (@FlatNo > 0)
BEGIN
    PRINT 'Check Flat Range in Building_No, SubBuilding No with No :' + CONVERT(VARCHAR(10), @FlatNo)
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_RunFlatRange(@FlatNo, @BuildingNo, @BuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Flat Range matched with Sub Building name and Building No'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Flat Range found for the Build no, Sub building name - No Data Return'
    END
END
```

Example Inputs:

@FlatNo = 4
@BuildingNo = 9

@BuildingName = 'ST. THOMAS SQUARE'
@AddressTemp populated by fn_RunFlatRange

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
Runs flat range logic; if matching record found with flat number, building number, and sub-building name, return it.

STEP 41:

Flat No + Alphabet Check in Building Name

```
PRINT 'Starting - Flat No + Alpha check in Building Name'
DELETE FROM @AddressTemp
INSERT INTO @AddressTemp
SELECT * FROM fn_CheckBuildingNoAlpha(@FlatNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as flat no + Albhabet available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
ELSE
BEGIN
    PRINT 'No Data available for flat no + Albhabet available in building_name with no_of_house hold > 1 and with one instance'
END
```

Example Inputs:

@FlatNo = 4
@AddressData contains address parts

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = '4A ST. THOMAS SQUARE'
Building_No = 4

Summary:
Checks for building names where flat number and alphabet combination appears, returns matching record if found.

STEP 42:

Building Range Check with Existing Flat No

```
PRINT 'Starting - Building Range Check with existing flat No'
DELETE FROM @AddressTemp
INSERT INTO @AddressTemp
SELECT * FROM fn_RunBuildingRange(@FlatNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as building range available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
ELSE
BEGIN
    PRINT 'No Building Range found for the flat No ' + CONVERT(VARCHAR(10), @FlatNo)
END
```

Example Inputs:

@FlatNo = 4

@AddressData contains relevant address string

Example Outputs:

Returns record:

UDPRN = 147258

Building_Name = 'FLAT 4-6 ST. THOMAS SQUARE'

Building_No = 4

Summary:

Checks building name ranges matching flat number within building name and returns if match found.

STEP 43:

Building No + Alphabet Check in Building Name

```
PRINT 'Starting - Building No + Alpha check in Building Name'
DELETE FROM @AddressTemp
INSERT INTO @AddressTemp
SELECT * FROM fn_CheckBuildingNoAlpha(@BuildingNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as building no + Albhabet available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
ELSE
BEGIN
    PRINT 'No Data available for building no + Albhabet available in building_name with no_of_house hold > 1 and with one instance'
END
```

Example Inputs:
@BuildingNo = 9
@AddressData contains relevant address string

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = '9A ST. THOMAS SQUARE'
Building_No = 9

Summary:
Searches building name for building number with alphabet combinations, returns matching record if found.

STEP 44:

Building Range Check with Existing Building No and Sub-Building Name

```
PRINT 'Starting - Building Range Check with existing building No ' + CONVERT(VARCHAR(10), @BuildingNo)
IF (ISNULL(@BuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@BuildingNo, @SubBuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sub building Name and Building Name range data matched'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @BuildingNo) + ' and sub building ' + @SubBuildingName
        PRINT 'Checking Only for Building range using building No.'
        DELETE FROM @AddressTemp
        INSERT INTO @AddressTemp
        SELECT * FROM fn_RunBuildingRange(@BuildingNo, '', @AddressData)

        IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
        BEGIN
            PRINT 'Building Name range data matched'
            SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
                   Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
                   Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
            FROM @AddressTemp
            RETURN
        END
        ELSE
        BEGIN
            PRINT 'Only for Building range using building No. not matched'
        END
    END
END
```

Example Inputs:
@BuildingNo = 9

@SubBuildingName = 'FLAT 4'
@AddressData contains relevant address

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
Performs building range search with building number and sub-building name, falls back to building number only if no direct match found.

STEP 45:

Transformed Building Number from Building Name

```
PRINT 'Starting - Check the transformed Building No in Building No Field'
SET @SanitizedBuildingNo = ''
SET @TransformedBuildingNo = 0
SELECT @SanitizedBuildingNo = dbo.fn_CleanString(@BuildingName, '%[^0-9]%')

IF (TRIM(@SanitizedBuildingNo) <> '' AND ISNUMERIC(@SanitizedBuildingNo) = 1)
BEGIN
    SET @TransformedBuildingNo = CONVERT(INT, @SanitizedBuildingNo)
    PRINT 'Sanitized Building No from Building Name data :' + @BuildingName + ' = ' + @SanitizedBuildingNo
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE 9'

Example Outputs:
Sanitized Building No from Building Name data : ST. THOMAS SQUARE 9 = 9

Summary:
Extracts numeric part from building name and stores as transformed building number for further processing

STEP 46:

Matching with Transformed Building Number in Building No Field

```sql
IF (@TransformedBuildingNo > 0)
BEGIN
    PRINT 'Start processing with Transformed Building No ' + CONVERT(VARCHAR(10), @TransformedBuildingNo) + ' to check single instance with Building no and No_of_household > 1)'
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_ReCheckBuildingNoWithTransformedData(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data transformed building no available in building_no with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No data available for transformed building no in building_no with no_of_house hold > 1 and with one instance'
    END
END
```

Example Inputs:
@TransformedBuildingNo = 9

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Matches records where transformed building number exists in building number field with household constraints.

STEP 47:

Building Range Check with Transformed Building Number

```sql
PRINT 'Starting - Building Range Check with transformed building No'

IF (ISNULL(@TransformedBuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data as building range available in building_name with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @TransformedBuildingNo)
    END
END
```

Example Inputs:
@TransformedBuildingNo = 9

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = '9 ST. THOMAS SQUARE'
Building_No = 9

Summary:
Checks for building name ranges using transformed building number for final matching attempts.

STEP 48:

Match Only with Building No

```
IF ((SELECT COUNT(*) FROM @Address WHERE Building_No = @BuildingNo) = 1)
BEGIN
    PRINT 'Matching with Building Name'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_No = @BuildingNo
    RETURN
END
```

Example Inputs:

@BuildingNo = 9

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:

This block checks if there is exactly one record in the address table matching the provided Building_No. If so, returns the corresponding address details.

STEP 49:

Match with Only Sub Building Name

```
IF ((SELECT COUNT(*) FROM @Address WHERE (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)) = 1)
BEGIN
    PRINT 'Matched data with Only Sub building data'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE (Sub_Building_Name = @SubBuildingName OR Sub_Building_Name = @TransformSubBuildingName)
    RETURN
END
```

Example Inputs:

@SubBuildingName = 'FLAT 4'

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
This block returns address details if there is exactly one record where Sub_Building_Name matches either the original or transformed sub-building value.

STEP 50:

Flat Range Check

```
IF (@FlatNo > 0)
BEGIN
    PRINT 'Check Flat Range in Building_No, SubBuilding No with No: ' + CONVERT(VARCHAR(10), @FlatNo)
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_RunFlatRange(@FlatNo, @BuildingNo, @BuildingName, @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Flat Range matched with Sub Building name and Building No'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:
@FlatNo = 4

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
Performs a flat range search using a helper function based on FlatNo, BuildingNo, and BuildingName. If matching records are found, returns address details.

STEP 51:

Match with Building Name and Building No

```
IF ((SELECT COUNT(*) FROM @Address WHERE Building_Name = @BuildingName AND Building_No = @BuildingNo) = 1)
BEGIN
    PRINT 'Matched with Building Name and Building No'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_Name = @BuildingName AND Building_No = @BuildingNo
    RETURN
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE'
@BuildingNo = 9

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Checks for a unique record where both Building_Name and Building_No match the provided values. If found, returns the address details.

STEP 52:

Transformed Building No Check from Building Name

```
SELECT @SanitizedBuildingNo = dbo.fn_CleanString(@BuildingName,'%[^0-9]%')
IF (TRIM(@SanitizedBuildingNo) <> '' AND ISNUMERIC(@SanitizedBuildingNo) = 1)
BEGIN
    SET @TransformedBuildingNo = CONVERT(INT, @SanitizedBuildingNo)
END

IF (@TransformedBuildingNo > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_ReCheckBuildingNoWithTransformedData(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data transformed building no available in building_no'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE 9'

Example Outputs:
Returns record:
UDPRN = 147258

Building_Name = 'ST. THOMAS SQUARE 9'
Building_No = 9

Summary:
Extracts numeric value from Building_Name using fn_CleanString. If valid numeric part exists, performs a transformed building number match through a helper function.

STEP 53:

Building Range Check with Transformed Building No

```
IF (ISNULL(@TransformedBuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data as building range available in building_name with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @TransformedBuildingNo)
    END
END
```

Example Inputs:
@TransformedBuildingNo = 9
@AddressData contains relevant address fragments

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Checks for building range matches using transformed building number through a helper function. If matching record exists, returns address details.

STEP 54:

Exact Building Name Match Check

```
IF ((SELECT COUNT(*) FROM @Address WHERE Building_Name = @BuildingName) = 1)
BEGIN
    PRINT 'Matched with Exact Building Name'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_Name = @BuildingName
    RETURN
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE'

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
If only one record exists with an exact Building_Name match, returns the corresponding address details.

STEP 55:

Check for Flat Range Using Only Building No

```
IF (@FlatNo > 0 AND @BuildingNo > 0)
BEGIN
    PRINT 'Check Flat Range with only Building No'
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_RunFlatRange(@FlatNo, @BuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Flat Range Matched with Only Building No'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:
@FlatNo = 4
@BuildingNo = 9

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
Searches for flat range matches considering FlatNo and BuildingNo using a helper function. If results found, returns address details.

STEP 56:

Final Address Check with Only Transformed Building No

```
IF (ISNULL(@TransformedBuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_ReCheckBuildingNoWithTransformedData(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Final ReCheck with Transformed Building No'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
END
```

Example Inputs:
@TransformedBuildingNo = 9
@AddressData contains relevant address fragments

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9

Summary:
Performs a final recheck using transformed building number with a helper function. If valid records exist, returns address details.

STEP 57:

Thoroughfare Additional Data Check

```
IF (LTRIM(RTRIM(@AdditionalDataToCheck)) <> '')
BEGIN
    INSERT INTO @ThouroughfareList
    SELECT Id, Thoroughfare_Descrp
    FROM Thoroughfare t
    INNER JOIN [Address] adr ON adr.ThoroughfareId = t.ID
    WHERE adr.Postcode = @PostCode AND Thoroughfare_Descrp = @AdditionalDataToCheck

    IF NOT EXISTS (SELECT 'true' FROM @ThouroughfareList HAVING COUNT(*) > 0)
    BEGIN
        INSERT INTO @ThouroughfareList
        SELECT Id, Thoroughfare_Descrp
        FROM Thoroughfare
        WHERE Thoroughfare_Descrp LIKE @AdditionalDataToCheck + '%'
    END

    SELECT @ThroufareCount = COUNT(*) FROM @ThouroughfareList

    IF (@ThroufareCount > 0)
    BEGIN
        PRINT 'Throughfare data identified with Additional Data ' + LTRIM(RTRIM(@AdditionalDataToCheck))
        SET @AdditionalDataToCheck = ''
    END
    ELSE
    BEGIN
        PRINT 'No Throughfare data identified with Additional Data ' + LTRIM(RTRIM(@AdditionalDataToCheck))
    END
END
```

Example Inputs:
@AdditionalDataToCheck = 'ST. THOMAS SQUARE'
@PostCode = 'NP25 5ES'

Example Outputs:
Returns record in @ThouroughfareList:
Id = 101
Thoroughfare_Descrp = 'ST. THOMAS SQUARE'

Summary:
Searches Thoroughfare table for matching additional data in current postcode. Falls back to a LIKE search if exact match isn't found.

STEP 58:

Only Building No Match Check

```
IF (@BuildingNo <> 0)
BEGIN
    IF ((SELECT COUNT(*) FROM @Address WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL) = 1)
    BEGIN
        PRINT 'Sending data by building no as there is only one record'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @Address
        WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL
        RETURN
    END
END
```

Example Inputs:
@BuildingNo = 9

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = NULL
Building_No = 9
Sub_Building_Name = NULL

Summary:
If only one record exists with Building_No populated, and both Sub_Building_Name and Building_Name are NULL, returns the address.

STEP 59:

Soundex Fuzzy Matching

```
IF (ISNULL(@FirstLineMatchValue, '') <> '')
BEGIN
    IF EXISTS (SELECT 'true' FROM @Address WHERE SOUNDEX(RTRIM(LTRIM(ToMatchValue))) = SOUNDEX(RTRIM(LTRIM(@FirstLineMatchValue))))
    BEGIN
        PRINT 'Searching data by Soundex'
        IF ((SELECT COUNT(*) FROM @Address WHERE SOUNDEX(RTRIM(LTRIM(ToMatchValue))) = SOUNDEX(RTRIM(LTRIM(@FirstLineMatchValue)))) = 1)
        BEGIN
            SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
                   Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
                   Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
            FROM @Address
            WHERE SOUNDEX(RTRIM(LTRIM(ToMatchValue))) = SOUNDEX(RTRIM(LTRIM(@FirstLineMatchValue)))
            RETURN
        END
    END
END
```

Example Inputs:
@FirstLineMatchValue = 'ST. THOMAS SQUARE'

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
Performs fuzzy matching using Soundex on cleaned address fragments. Returns record if exactly one Soundex match is found.

STEP 60:

Building No + Alphabet Check in Building Name

```
PRINT 'Starting - Building No + Alpha check in Building Name'

INSERT INTO @AddressTemp
SELECT * FROM fn_CheckBuildingNoAlpha(@BuildingNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as building no + Alphabet available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
ELSE
BEGIN
    PRINT 'No Data available for building no + Alphabet available in building_name with no_of_house hold > 1 and with one instance'
END
```

Example Inputs:
@BuildingNo = 9
@AddressData contains address fragments like '9A ST. THOMAS SQUARE'

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = '9A ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = NULL

Summary:
Checks for building numbers with alphabet suffixes (e.g., 9A, 9B) linked to the building name and returns the record if conditions met.

STEP 61:

Building Range Check with Existing Building No

```
PRINT 'Starting Building Range Check with existing building No'

DELETE FROM @AddressTemp

INSERT INTO @AddressTemp
SELECT * FROM fn_RunBuildingRange(@BuildingNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as building no within range available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
ELSE
BEGIN
    PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @BuildingNo)
END
```

Example Inputs:
@BuildingNo = 9
@AddressData contains building range like '7-10 ST. THOMAS SQUARE'

Example Outputs:
Returns record:

UDPRN = 147258
Building_Name = '7-10 ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = NULL

Summary:
Checks for addresses where the building number falls within a specified range in the building name and returns matching record.

STEP 62:

Transformed Building No from Building Name Check

```
PRINT 'Starting - Transformed Building No check in Building No'

SELECT @SanitizedBuildingNo = dbo.fn_CleanString(@BuildingName,'%[^0-9]%')

IF (TRIM(@SanitizedBuildingNo) <> '' AND ISNUMERIC(@SanitizedBuildingNo) = 1)
BEGIN
    SET @TransformedBuildingNo = CONVERT(INT, @SanitizedBuildingNo)
    PRINT 'Sanitized Building No from Building Name data :' + @BuildingName + ' = ' + @SanitizedBuildingNo
END

IF (@TransformedBuildingNo > 0)
BEGIN
    PRINT 'Start processing with Transformed Building No to check single instance with Building no and No_of_household > 1'
    DELETE FROM @AddressTemp

    INSERT INTO @AddressTemp
    SELECT * FROM dbo.fn_ReCheckBuildingNoWithTransformedData(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data transformed building no available in building_no with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No data available for transformed building no in building_no with no_of_house hold > 1 and with one instance'
    END
END
```

Example Inputs:
@BuildingName = '9A ST. THOMAS SQUARE'
@AddressData contains relevant fragments

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = '9A ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = NULL

Summary:
Extracts numeric portion from Building_Name (e.g., '9A' → '9'), then rechecks building no match. Returns address if valid.

STEP 63:

Building Range Check with Transformed Building No

```
PRINT 'Starting - Building Range Check with transformed building No'

IF (@TransformedBuildingNo > 0)
BEGIN
    DELETE FROM @AddressTemp

    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@TransformedBuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data as building no within range available in building_name with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @TransformedBuildingNo)
    END
END
```

Example Inputs:

@BuildingName = '9A ST. THOMAS SQUARE'
Transformed Building No = 9
@AddressData contains range like '7-10 ST. THOMAS SQUARE'

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = '7-10 ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = NULL

Summary:

Checks if transformed building number falls within building name range, returns address if valid.

STEP 64:

Building Name Only Check

```
PRINT 'Check with only building name'

IF ((SELECT COUNT(*) FROM @Address WHERE Building_Name = @BuildingName) = 1)
BEGIN
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_Name = @BuildingName
    RETURN
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE'
Exactly 1 record in @Address with matching Building_Name

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 0
Sub_Building_Name = NULL

Summary:
If only Building_Name exists and uniquely identifies record, returns the record.

STEP 65:

Building Name with Building No = 0 and Sub Building Name NULL

```
PRINT 'Check with only building name with building no = 0 and sub building name = null'

IF ((SELECT COUNT(*) FROM @Address WHERE Building_Name = @BuildingName AND Building_No = 0 AND Sub_Building_Name IS NULL) = 1)
BEGIN
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_Name = @BuildingName AND Building_No = 0 AND Sub_Building_Name IS NULL
    RETURN
END
```

Example Inputs:
@BuildingName = 'ST. THOMAS SQUARE'
1 record with Building_Name = 'ST. THOMAS SQUARE', Building_No = 0, Sub_Building_Name = NULL

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 0
Sub_Building_Name = NULL

Summary:
When only Building_Name exists and both Building_No and Sub_Building_Name are empty, returns record if unique.

STEP 66:

Thoroughfare (Street) Check

```
IF (LTRIM(RTRIM(@AdditionalDataToCheck)) <> '')
BEGIN
    INSERT INTO @ThouroughfareList
    SELECT Id, Thoroughfare_Descrp
    FROM Thoroughfare t
    INNER JOIN [Address] adr ON adr.ThoroughfareId = t.ID
    WHERE adr.Postcode = @PostCode AND Thoroughfare_Descrp = @AdditionalDataToCheck

    IF NOT EXISTS (SELECT 'true' FROM @ThouroughfareList HAVING COUNT(*) > 0)
    BEGIN
        INSERT INTO @ThouroughfareList
        SELECT Id, Thoroughfare_Descrp
        FROM Thoroughfare
        WHERE Thoroughfare_Descrp LIKE @AdditionalDataToCheck + '%'
    END

    SET @ThroufareCount = (SELECT COUNT(*) FROM @ThouroughfareList)

    IF (@ThroufareCount > 0)
    BEGIN
        PRINT 'Throughfare data identified with Additional Data ' + LTRIM(RTRIM(@AdditionalDataToCheck))
        SET @AdditionalDataToCheck = ''
    END
    ELSE
    BEGIN
        PRINT 'No Throughfare data identified with Additional Data ' + LTRIM(RTRIM(@AdditionalDataToCheck))
    END
END
```

Example Inputs:
@AdditionalDataToCheck = 'ST. THOMAS SQUARE'
Postcode present = 'NP25 5ES'

Example Outputs:
Returns record (example from Thoroughfare):
Id = 7412
Thoroughfare_Descrp = 'ST. THOMAS SQUARE'

Summary:
Checks Thoroughfare (street) data for exact or partial matches based on additional data provided.

STEP 67:

Building No Direct Match with Minimal Fields

```
PRINT 'Running for building no data if exists'

IF (@BuildingNo <> 0)
BEGIN
    IF ((SELECT COUNT(*) FROM @Address WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL) = 1)
    BEGIN
        PRINT 'Sending data by building no as there is only one record'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @Address
        WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'MULTIPLE RECORDS with the Same building no, with Building Name, Sub Building Name NULL Checking further'
    END
END
```

Example Inputs:
@BuildingNo = 9
Only one record in @Address with Building_No = 9, Sub_Building_Name = NULL, Building_Name = NULL

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = NULL
Building_No = 9
Sub_Building_Name = NULL

Summary:
Directly returns address if building number exists with no building name or sub-building name, and record is unique.

STEP 68:

Building No Match with Organisation Name NULL

```
IF ((SELECT COUNT(*) FROM @Address WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL AND Organisation_Name IS NULL)
BEGIN
    PRINT 'Sending data by building no as there is only one record'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL AND Organisation_Name IS NULL
    RETURN
END
```

Example Inputs:
@BuildingNo = 9
One record exists with Building_No = 9, Sub_Building_Name = NULL, Building_Name = NULL, Organisation_Name = NULL

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = NULL
Building_No = 9
Sub_Building_Name = NULL
Organisation_Name = NULL

Summary:
Returns record if building number exists with no building name, sub-building name, or organisation name, and only one such record.

STEP 69:

Building No with Other Columns

```
IF ((SELECT COUNT(*) FROM @Address WHERE Building_No = @BuildingNo) = 1)
BEGIN
    PRINT 'Sending data by building no with value in other columns (building name / sub building name) as there is only one record'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @Address
    WHERE Building_No = @BuildingNo
    RETURN
END
```

Example Inputs:

@BuildingNo = 9
Only one record with Building_No = 9, other columns may have data

Example Outputs:

Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:

Returns record if building number matches and only one record exists regardless of additional columns.

STEP 70:

Building No + Alphabet Check in Building Name

```
PRINT 'Starting - Building No + Alpha check in Building Name'

INSERT INTO @AddressTemp
SELECT * FROM fn_CheckBuildingNoAlpha(@BuildingNo, '', @AddressData)

IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
BEGIN
    PRINT 'Sending data as building no + Alphabet available in building_name with no_of_house hold > 1 and with one instance'
    SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
           Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
           Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
    FROM @AddressTemp
    RETURN
END
```

Example Inputs:

@BuildingNo = 9
@AddressData includes Building_Name = '9A ST. THOMAS SQUARE'

Example Outputs:

Returns record:

UDPRN = 147258
Building_Name = '9A ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = NULL

Summary:
Checks for building number with alphabet suffix (e.g., '9A'), returns if valid and unique with sufficient household count.

STEP 71:

Building Range Check with Existing Building No

```
PRINT 'Starting - Building Range Check with existing building No'

IF (ISNULL(@BuildingNo, 0) > 0)
BEGIN
    DELETE FROM @AddressTemp
    INSERT INTO @AddressTemp
    SELECT * FROM fn_RunBuildingRange(@BuildingNo, '', @AddressData)

    IF ((SELECT COUNT(*) FROM @AddressTemp) > 0)
    BEGIN
        PRINT 'Sending data as building no + Alphabet available in building_name with no_of_house hold > 1 and with one instance'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @AddressTemp
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'No Building Range found for the Building No ' + CONVERT(VARCHAR(10), @BuildingNo)
    END
END
```

Example Inputs:
@BuildingNo = 9
Building range match returns records for building numbers 9 to 15

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = '9-15 ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = NULL

Summary:
Uses building range logic to match addresses in a defined numeric range (e.g., '9-15'), returns if valid.

STEP 72:

Soundex Matching for Fuzzy Address Matching

```
PRINT 'Running Fuzzy check using Soundex Process'

IF (ISNULL(@BuildingName, '') <> '')
BEGIN
    DELETE FROM @Address WHERE ISNULL(Building_Name, '') <> @BuildingName
    UPDATE @Address SET ToMatchValue = REPLACE(ToMatchValue, @BuildingName, ''),
                        AddressFirstLine = REPLACE(ToMatchValue, @BuildingName, '')
    SET @FirstLineMatchValue = REPLACE(@FirstLineMatchValue, @BuildingName, '')
END

DELETE a FROM @AddressSplit a
INNER JOIN @Address adr ON LTRIM(RTRIM(adr.Sub_Building_Name)) = LTRIM(RTRIM(a.AddressValue))
WHERE adr.Postcode = @PostCode

IF (ISNULL(@FirstLineMatchValue, '') = '')
BEGIN
    DELETE FROM @AddressSplit WHERE REPLACE(AddressValue, ' ', '') = ''
    SELECT TOP 1 @FirstLineMatchValue = REPLACE(AddressValue, ' ', '') FROM @AddressSplit
END

PRINT 'FirstLineMatch Value ' + @FirstLineMatchValue

IF EXISTS (SELECT 'true' FROM @Address WHERE SOUNDEX(RTRIM(LTRIM(ToMatchValue))) = SOUNDEX(RTRIM(LTRIM(@FirstLineMatchValue))))
BEGIN
    PRINT 'Searching data by Soundex'
    IF ((SELECT COUNT(*) FROM @Address WHERE SOUNDEX(RTRIM(LTRIM(ToMatchValue))) = SOUNDEX(RTRIM(LTRIM(@FirstLineMatchValue)))) = 1)
    BEGIN
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @Address
        WHERE SOUNDEX(RTRIM(LTRIM(ToMatchValue))) = SOUNDEX(RTRIM(LTRIM(@FirstLineMatchValue)))
    END
    ELSE
    BEGIN
        PRINT 'More than 1 Soundex match, Not returning anything'
        SELECT * FROM @Address WHERE 1 = 2
    END
END
ELSE
BEGIN
    PRINT 'No Soundex match'
    SELECT * FROM @Address WHERE 1 = 2
END
```

Example Inputs:
@FirstLineMatchValue = 'ST. THOMAS SQUARE'
SOUNDEX match found for ToMatchValue

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 4'

Summary:
Uses SOUNDEX for fuzzy matching of address first line, returns record only if single unique match exists.

STEP 73:

Error Handling

```
BEGIN CATCH
    PRINT 'ERROR Occured, below are the details'
    DECLARE @ErrorMessage NVARCHAR(4000), @ErrorSeverity INT, @ErrorState INT, @ErrorLine INT

    SELECT @ErrorMessage = ERROR_MESSAGE(),
           @ErrorSeverity = ERROR_SEVERITY(),
           @ErrorState = ERROR_STATE(),
           @ErrorLine = ERROR_LINE()

    PRINT 'Message : ' + @ErrorMessage + ', Severity : ' + CONVERT(VARCHAR(10), @ErrorSeverity) +
    ', State : ' + CONVERT(VARCHAR(10), @ErrorState) + ', Line No : ' + CONVERT(VARCHAR(10), @ErrorLine)
END CATCH
```

Example Outputs:

Console message:

Message : Conversion failed when converting the varchar value 'ABC' to data type int., Severity : 16, State : 1, Line No : 250

Summary:

Catches and logs SQL errors with message, severity, state, and line number.

STEP 74:

Thoroughfare Data Matching (Additional Thoroughfare Check)

```
IF (LTRIM(RTRIM(@AdditionalDataToCheck)) <> '')
BEGIN
    INSERT INTO @ThouroughfareList
    SELECT Id, Thoroughfare_Descrp
    FROM Thoroughfare t
    INNER JOIN [Address] adr ON adr.ThoroughfareId = t.ID
    WHERE adr.Postcode = @PostCode AND Thoroughfare_Descrp = @AdditionalDataToCheck

    IF NOT EXISTS (SELECT 'true' FROM @ThouroughfareList HAVING COUNT(*) > 0)
    BEGIN
        INSERT INTO @ThouroughfareList
        SELECT Id, Thoroughfare_Descrp
        FROM Thoroughfare
        WHERE Thoroughfare_Descrp LIKE @AdditionalDataToCheck + '%'
    END

    SELECT @ThroufareCount = COUNT(*) FROM @ThouroughfareList

    IF (@ThroufareCount > 0)
    BEGIN
        PRINT 'Throughfare data identified with Additional Data ' + LTRIM(RTRIM(@AdditionalDataToCheck))
        SET @AdditionalDataToCheck = ''
    END
    ELSE
    BEGIN
        PRINT 'No Throughfare data identified with Additional Data ' + LTRIM(RTRIM(@AdditionalDataToCheck))
    END
END
```

Example Inputs:
@AdditionalDataToCheck = 'ST. THOMAS SQUARE'
@PostCode = 'NP25 5ES'

Example Outputs:
Console message:
Throughfare data identified with Additional Data ST. THOMAS SQUARE

Summary:
Checks Thoroughfare table for exact or partial match based on Additional Data. Populates list for further filtering.

STEP 75:

Thoroughfare Filtering from Address Split

```
SELECT @ThroufareCount = COUNT(*) FROM @ThouroughfareList

DELETE a
FROM @AddressSplit a
INNER JOIN @ThouroughfareList t ON a.AddressValue = t.Thoroughfare
```

Example Outputs:
Console message:
AddressSplit cleaned by removing known Thoroughfare matches.

Summary:
Removes matching thoroughfare terms from AddressSplit to avoid duplicate processing.

STEP 76:

Final Building No Check with Limited Supporting Data

```
PRINT 'Running for building no data if exists'

IF (@BuildingNo <> 0)
BEGIN
    IF ((SELECT COUNT(*) FROM @Address WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL) = 1)
    BEGIN
        PRINT 'Sending data by building no as there is only one record'
        SELECT UDPRN, Postcode, Posttown, Dep_Locality, Dbl_Dep_Locality, Thoroughfare_Descrp,
               Dep_Thoroughfare_Descrp, Building_No, Building_Name, Sub_Building_Name,
               Organisation_Name, Department_Name, PO_Box, Postcode_Type, SU_Org_Indicator, Delivery_Point_Sfx
        FROM @Address
        WHERE Building_No = @BuildingNo AND Sub_Building_Name IS NULL AND Building_Name IS NULL
        RETURN
    END
    ELSE
    BEGIN
        PRINT 'MULTIPLE RECORDS with the Same building no, with Building Name, Sub Building Name NULL Checking further'
    END
END
```

Example Inputs:
@BuildingNo = 9

Example Outputs:
Returns record:
UDPRN = 147258
Building_Name = NULL
Building_No = 9
Sub_Building_Name = NULL

Summary:
If only Building No is present, and it's the only matching record, returns it. Otherwise, more detailed checks continue.

STEP 77:

Final Cleanup and AddressSplit Filtering

```
DELETE FROM @Address WHERE Building_No <> @BuildingNo

UPDATE @Address
SET ToMatchValue = REPLACE(ToMatchValue, @BuildingNo, ''),
    AddressFirstLine = REPLACE(ToMatchValue, @BuildingNo, '')

SET @FirstLineMatchValue = REPLACE(@FirstLineMatchValue, @BuildingNo, '')
```

Summary:
Removes unmatched building numbers and updates ToMatchValue to simplify final fuzzy logic matching.

Processing Breakdown for Given Inputs:

Input Address:
FLAT 2 9 ST. THOMAS SQUARE, MONMOUTH, GWENT, NP25 5ES

Postcode:
NP25 5ES

Typical Internal Extraction:

- @SubBuildingName → 'FLAT 2'
- @BuildingNo → 9
- @BuildingName → 'ST. THOMAS SQUARE'
- @PostTown → 'MONMOUTH'
- @PostCode → 'NP25 5ES'

Step-by-Step Expected Matching:

✓ Checks for **Sub Building Name + Building No** — likely matches 1 record.
✓ If no exact match, checks with **Building Name + Building No** — likely matches.
✓ Runs **Flat Range Check**, if applicable.
✓ Final fallback: Soundex/Fuzzy logic if all structured matching fails.

Final Expected Output:

```
Returns record:
UDPRN = 147258
Building_Name = 'ST. THOMAS SQUARE'
Building_No = 9
Sub_Building_Name = 'FLAT 2'
Posttown = 'MONMOUTH'
Postcode = 'NP25 5ES'
Dep_Locality = NULL
Dbl_Dep_Locality = NULL
Thoroughfare_Descrp = NULL
Dep_Thoroughfare_Descrp = NULL
Organisation_Name = NULL
Department_Name = NULL
PO_Box = NULL
Postcode_Type = 'S'
SU_Org_Indicator = NULL
Delivery_Point_Sfx = '1A'
```