**Data Cleaning Documentation: Courses Table**

**1. Initial Data Ingestion & Column Selection**

* **Step**: Table.SelectColumns(Source, {"Name", "Content"})
  + **Purpose**:
    - Isolate the Name (course title) and Content (raw HTML/structured data) columns from the source.
    - Reduces memory usage by discarding irrelevant columns early.
  + **Technical Notes**:
    - Assumes Content contains nested data (e.g., HTML elements, JSON) to be parsed later.
    - The Source is likely a folder or web API response containing multiple files/records.

**2. Filtering Hidden System Files**

* **Step**: Table.SelectRows(#"Removed Other Columns", each [Attributes]?[Hidden]? <> true)
  + **Purpose**:
    - Exclude files marked as "Hidden" in their metadata (e.g., temporary files, system files).
    - Ensures only user-visible courses are processed.
  + **Edge Cases**:
    - Uses ? (optional chaining) to gracefully handle missing Attributes or Hidden properties.
    - Rows without [Attributes] are retained (treated as null ≠ true).

**3. Custom Function to Parse Structured Content**

* **Step**: Table.AddColumn(#"Filtered Hidden Files1", "Transform File", each #"Transform File"([Content]))
  + **Purpose**:
    - Invoke a custom function Transform File to extract structured data from the Content column (e.g., HTML parsing, JSON decoding).
  + **Technical Notes**:
    - The function Transform File is not defined here but likely returns a nested table/record.
    - Example extracted fields: sub\_id, course metadata, pricing, ratings, etc.
  + **Outcome**:
    - Generates a new column Transform File with parsed data as a nested table.

**4. Expanding Nested Data**

* **Steps**: Table.ExpandTableColumn(#"Removed Other Columns1", "Transform File", Table.ColumnNames(#"Transform File"(#"Sample File")))
  + **Purpose**:
    - Expand the nested table in Transform File into flat columns (e.g., sub\_id, ud-heading-md, ud-sr-only).
    - Dynamically retrieves column names from a sample file to ensure schema consistency.
  + **Technical Notes**:
    - Uses Sample File to infer column names, avoiding hardcoding.
    - Risks schema drift if sample file changes structure.

**5. Column Renaming & Removal**

**Substeps**:

1. **Initial Renaming**:
   * course-card-image\_image\_\_iJLJg src → Image
   * ud-heading-md → Name
   * ud-heading-md href → Link
   * ud-sr-only → Description
   * ud-sr-only 6 → Levels
   * Later renamed Levels → Level after cleaning.
2. **Redundant Column Removal**:
   * Columns like ud-sr-only 5, ud-icon href, and others are dropped due to irrelevance or duplication.
3. **Final Schema**:
   * Retained columns: sub\_id, Image, Name, Link, Description, Level, instructor, Rating, Current price, Original Price, etc.

* **Purpose**:
  + Simplify column names for usability.
  + Remove noise (e.g., internal class names like ud-sr-only from scraped HTML).

**6. Data Type Enforcement**

* **Step**: Table.TransformColumnTypes(#"Expanded Table Column1", {{"sub\_id", Int64.Type}, ...})
  + **Key Conversions**:
    - sub\_id → Integer (unique course identifier).
    - ud-heading-sm → Rating out of 5 (decimal → rounded to integer).
    - Current price/Original Price → Currency (after cleaning symbols like EÂ£).
  + **Edge Cases**:
    - Original Price set to 0 if missing to avoid nulls in discount calculations.

**7. Null Handling & Filtering**

* **Steps**:
  1. Remove rows where sub\_id is null: Table.SelectRows(#"Changed Type", each ([sub\_id] <> null))
  2. Remove rows with null Duration: Table.SelectRows(#"Changed Type1", each ([Duration] <> null))
  3. **Purpose**:
     + Ensure critical fields like sub\_id and Duration are populated.
     + Prevents errors in joins/aggregations downstream.

**8. Text Cleaning & Transformation**

* **Steps**:
  1. **Trim Whitespace**: Table.TransformColumns(#"Replaced Value1", {{"Name", Text.Trim, type text}})
  2. **Replace Currency Symbols**: Table.ReplaceValue("EÂ£", "", Replacer.ReplaceText, {"Current price", "Original Price"})
  3. **Clean Reviews**: Table.ReplaceValue("review", "", Replacer.ReplaceText, {"NoSubscribers"})
  4. **Outcome**:
     + Current price: EÂ£249.99 → 249.99 (numeric).
     + "Advanced Python" → Advanced Python (quotes removed).

**9. Derived Columns**

**a. Duration Calculation**

* **Step**: Table.AddColumn(#"Replaced Value9", "Duration", each

if Text.Contains([Duration hours], "min") then

Number.From(Text.BeforeDelimiter([Duration hours], " ")) / 60

else if Text.Contains([Duration hours], "hour") then

Number.From(Text.BeforeDelimiter([Duration hours], " "))

else null)

* + **Logic**:
    - Converts "1 hour 30 mins" → 1.5 hours.
    - Uses Text.BeforeDelimiter to split strings.
  + **Edge Cases**:
    - Handles nulls by excluding them in later filtering.

**b. Language Detection**

* **Step**: Table.AddColumn(#"Filtered Rows2", "Language", each

let

textChars = Text.ToList([Name]),

charCodes = List.Transform(textChars, Character.ToNumber),

engCount = List.Count(List.Select(charCodes, \_ >= 65 and \_ <= 122)), *// Latin*

arabicCount = List.Count(List.Select(charCodes, \_ >= 1569 and \_ <= 1791)), *// Arabic*

... *// Other languages*

language = *// Logic to pick max count*

in

language)

* + **Purpose**:
    - Detect course language based on Unicode ranges in the Name field.
  + **Limitations**:
    - Overlap in Unicode ranges (e.g., French accents vs. Spanish) may cause misclassification.
    - Fails for multilingual course names.

**c. Discount Calculation**

* **Step**: Table.AddColumn(#"Renamed Columns8", "Discount", each

if [Original Price] = 0 then 0

else ([Original Price] - [Current price]) / [Original Price] \* 100)

* + **Logic**:
    - Avoids division by zero by setting Discount to 0 if Original Price is missing.
    - Example: Original Price = 100→CurrentPrice=100→*CurrentPrice*=75 → Discount = 25%.

**d. Status Assignment (Simulated)**

* **Step**: Table.AddColumn(#"Changed Type6", "Status", each

let RandomIndex = Number.RandomBetween(1, 100) in

if RandomIndex <= 60 then "Published"

else if RandomIndex <= 85 then "Draft"

else "Archived")

* + **Purpose**:
    - Simulates real-world status distribution for testing:
      * 60% Published, 25% Draft, 15% Archived.
  + **Limitation**:
    - Randomization is not reproducible (resets on refresh).

**10. Final Schema Adjustments**

* **Renaming**:
  + Name → Title
  + Image → ImageUrl
  + Link → VideoUrl
  + reviews → NoSubscribers
* **Logical Flags**:
  + IsFree: True if Current price = 0.
  + IsApproved: True if Status = Published.

**11. Instructor Filtering**

* **Step**: Table.SelectRows(#"Changed Type10", each ([instructor] <> "A Course You'll Actually Finish..." and ...))
  + **Purpose**:
    - Exclude invalid instructors (e.g., non-Latin names, spammy entries).
  + **Technical Notes**:
    - Hardcoding 100+ names is error-prone. Consider regex or a separate "blocklist" table.

**Key Insights & Recommendations**

1. **Language Detection**:
   * Use ML-based language detection for higher accuracy (e.g., Azure Cognitive Services).
2. **Status Assignment**:
   * Replace randomization with actual data if available (e.g., API field is\_published).
3. **Instructor Filtering**:
   * Use regex to exclude non-ASCII characters: Table.SelectRows(..., each not Text.Contains([instructor], "[\u0080-\uFFFF]"))
4. **Error Handling**:
   * Wrap steps in try...otherwise to log errors (e.g., parsing failures in Transform File).
5. **Performance**:
   * The Language column calculation is computationally intensive. Precompute during extraction if possible.

**Before Data Cleaning:**

A screenshot of a computer

AI-generated content may be incorrect.  
**After Data Cleaning:**A screenshot of a computer

AI-generated content may be incorrect.