**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**A picture containing logo

Description automatically generated**

**BI PROJECT**

**UK Car Accidents – 2011 - 2014**

XÂY DỰNG VÀ KHAI THÁC KDL

Trần Huy Vũ 18127257

Quách Phú Thành 18127215

Trần Ngọc Bảo Trân 18127234  
Trịnh Quang Hà 18127094

**Môn: Hệ thống thông tin phục vụ trí tuệ kinh doanh**

**Thành phố Hồ Chí Minh - 2021**

# Bảng thông tin chi tiết nhóm

|  |  |  |
| --- | --- | --- |
| **Mã nhóm:** | 18HTTT\_11 | |
| **Số lượng:** | 4 | |
| **MSSV** | **Họ tên** | **Email** |
| 18127257 | Trần Huy Vũ | 18127257@student.hcmus.edu.vn |
| 18127215 | Quách Phú Thành | 18127215@student.hcmus.edu.vn |
| 18127234 | Trần Ngọc Bảo Trân | 18127234@student.hcmus.edu.vn |
| 18127094 | Trịnh Quang Hà | 18127094@student.hcmus.edu.vn |

|  |
| --- |
| **Mô tả dữ liệu** |

1. Bảng Accidents1114

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | Accident\_Index | Mã vụ tai nạn | String | Accident1114 |
| 2 | LSOA\_of\_Accident | Khu vực địa lý của vị  trí xảy ra tai nạn | String | Accident1114 |
| 3 | Local\_Authority\_(District) | Tên khu vực địa  phương xảy ra vụ việc | String | Accident1114 |
| 4 | Accident\_Severity | Mức độ nghiêm trọng của tai nạn | String | Accident1114 |
| 5 | Speed\_limit | Tốc độ giới hạn | String | Accident1114 |
| 6 | Road\_Type | Loại đường xảy ra tai nạn | String | Accident1114 |
| 7 | Urban\_or\_Rural\_Area | Khu vực nội hay ngoại thành | String | Accident1114 |
| 8 | Time | Thời gian xảy ra tai nạn | DateTime | Accident1114 |
| 9 | Light\_Conditions | Điều kiện ánh sáng khi xảy ra tai nạn | String | Accident1114 |
| 10 | Number\_of\_Vehicles | Số xe liên quan | String | Accident1114 |
| 11 | Number\_of\_Casualities | Số nạn nhân liên quan | String | Accident1114 |
| 12 | Date | Ngày xảy ra tai nạn | Date | Accident1114 |

1. Bảng Accident\_Severity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã độ nghiêm trọng của tai nạn | String | Accident\_Severity\_NDS |
| 2 | label | Tên độ nghiêm trọng của tai nạn | String | Accident\_Severity\_NDS |

1. Bảng Road\_Type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã loại đường | String | Road\_Type\_NDS |
| 2 | label | Tên loại đường | String | Road\_Type\_NDS |

1. Bảng Light\_Conditions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã điều kiện ánh sáng | String | Light\_Conditions\_NDS |
| 2 | label | Tên điều kiện ánh sáng | String | Light\_Conditions\_NDS |

1. Bảng Urban\_Rural

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã nông thôn hay thành thị | String | Urban\_Rural\_NDS |
| 2 | label | Tên nông thôn hay thành thị | String | Urban\_Rural\_NDS |

1. Bảng Local\_Authority\_District

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã khu vực | String | Local\_Authority\_(District) |
| 2 | label | Tên khu vực | String | Local\_Authority\_District\_NDS |

1. Bảng Casualty\_Type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã loại thương vong | String | Casualty\_Type |
| 2 | label | Tên loại thương vong | String | Casualty\_Type |

1. Bảng Age\_Band\_of\_Casualty

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã độ tuổi thương vong | String | Age\_Band\_of\_Casualty |
| 2 | label | Độ tuổi thương vong | String | Age\_Band\_of\_Casualty |

1. Bảng Sex\_Casualty

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã giới tính thương vong | String | Sex\_Casualty |
| 2 | label | Tên giới tính thương vong | String | Sex\_Casualty |

1. Bảng Casualty\_severity

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã mức độ thương vong | String | Casualty\_severity |
| 2 | label | Tên mức độ thương vong | String | Casualty\_severity |

1. Bảng Casualty1114

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | Casualty\_Reference | Mã thương vong | String | Casualty1114 |
| 2 | Casualty\_severity | Mức độ thương vong | String | Casualty1114 |
| 3 | Age\_of\_Casualty | Độ tuổi thương vong | String | Casualty1114 |
| 4 | Casualty\_Type | Loại thương vong | String | Casualty1114 |
| 5 | Sex\_of\_Casualty | Giới tính của người thương vong | String | Casualty1114 |
| 6 | Accident\_Index | Mã tai nạn | String | Casualty1114 |
| 7 | Vehicle\_Reference | Mã phương tiện | String | Casualty1114 |

1. Bảng Vehicle\_Type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã loại phương tiện | String | Vehicle\_Type |
| 2 | label | Tên loại phương tiện | String | Vehicle\_Type |

1. Bảng Journey\_Purpose

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | code | Mã mục đích chuyến đi | String | Journey\_Purpose |
| 2 | label | Tên mục đích chuyến đi | String | Journey\_Purpose |

1. Bảng Vehicle1114

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | Accident\_Index | Mã vụ tai nạn | String | Vehicle1114 |
| 2 | Vehicle\_Reference | Mã phương tiện | String | Vehicle1114 |
| 3 | Vehicle\_Type | Loại phương tiện | String | Vehicle1114 |
| 4 | Journey\_Purpose\_of\_Driver | Mục đích chuyến đi của tài xế | String | Vehicle1114 |

1. Bảng Postcodes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | postcode | Mã potscode | String | Postcodes |
| 2 | city | Thành phố | String | Postcodes |
| 3 | country\_name | Tên nước | String | Postcodes |
| 4 | region\_name | Tên miền | String | Postcodes |
| 5 | county | tỉnh | String | Postcodes |

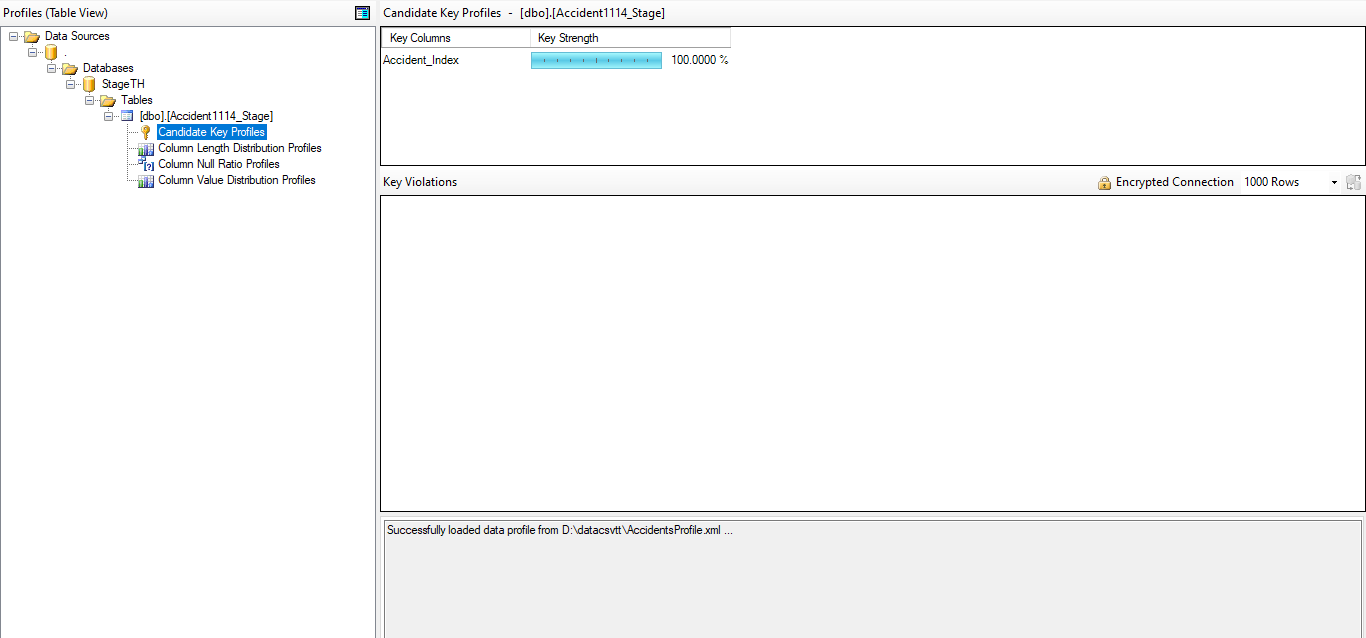
1. Bảng PCD\_OA\_LSOA\_MSOA\_LAD\_AUG21\_UK\_LU

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Trường dữ liệu | Mô tả | Kiểu dữ liệu | Tên file |
| 1 | pcd8 | Mã potscode | String | PCD\_OA\_LSOA\_MSOA\_LAD\_AUG21\_UK\_LU |
| 2 | lsoa11cd | Mã | String | PCD\_OA\_LSOA\_MSOA\_LAD\_AUG21\_UK\_LU |

|  |
| --- |
| **Phân tích profiling data** |

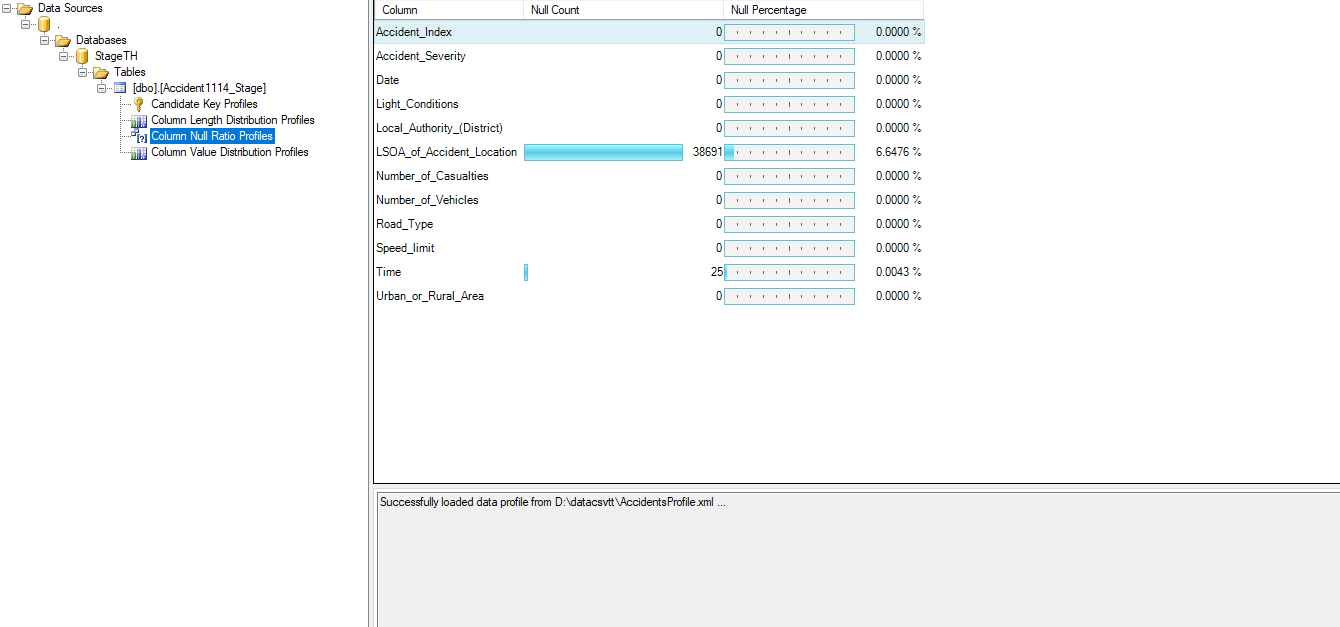
1. Accidents1114

**Candidate Key**



Theo như Data Profiling, bảng Accidents1114 có 1 khoá ứng viên Accident\_Index, do đó ta có thể chọn Accident\_Index làm khoá chính.

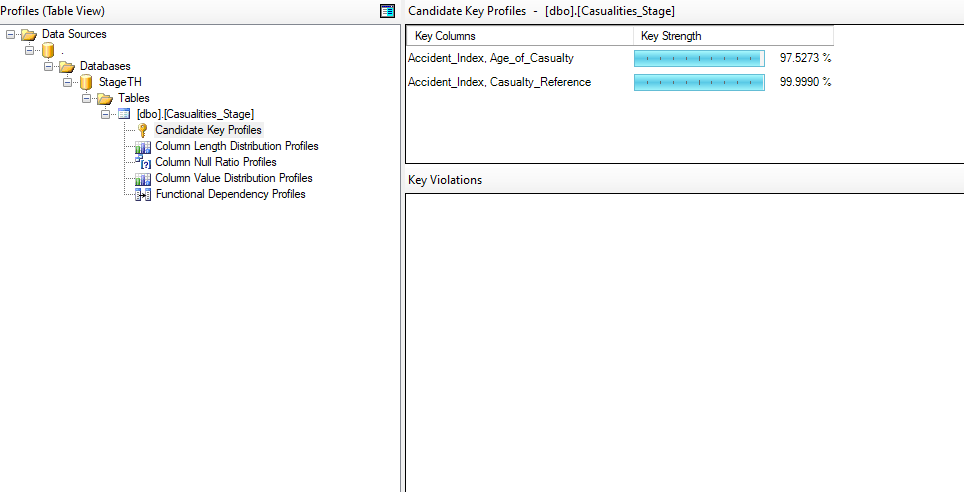
**Column Null Ratio**



Sau khi chạy Data Profiling, ta thấy có 2 trường LSOA\_of\_Accident\_Location và Time có những dòng bị null.

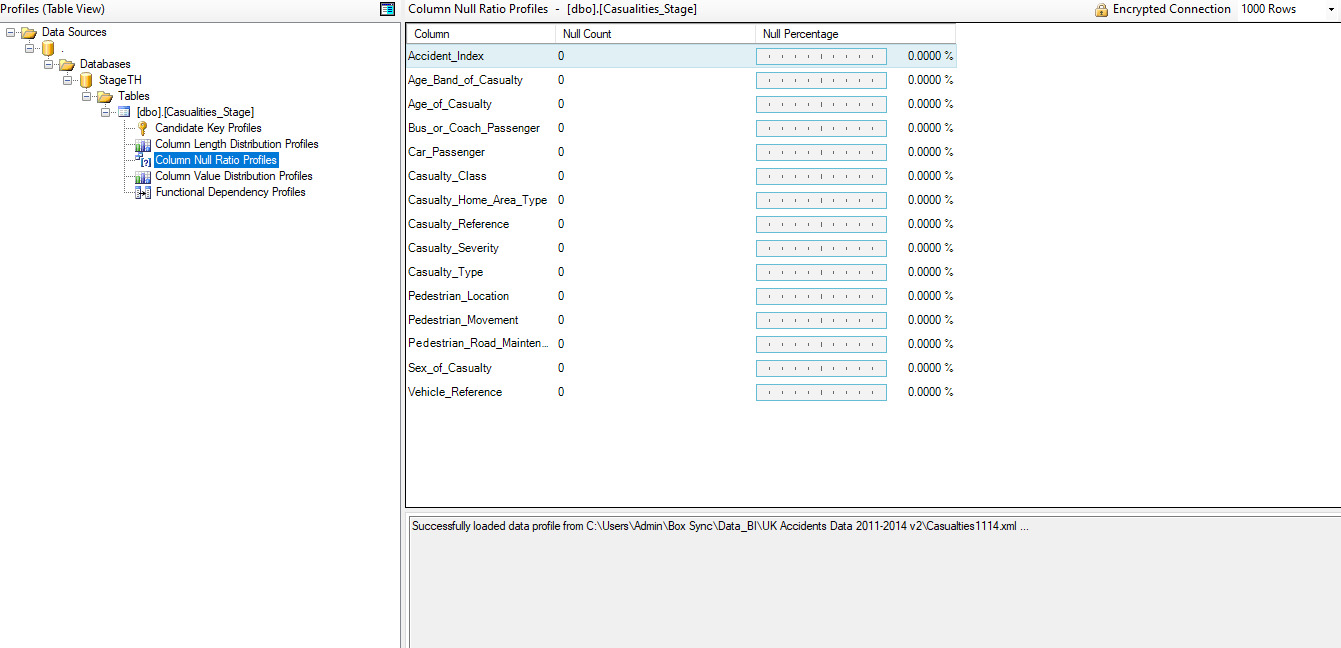
1. Casualities

**Candidate Key**



Theo như Data Profiling, bảng Casualities1114 có khoá ứng viên gồm 2 thuộc tính (Accident\_Index, Casuality\_Reference) đạt 99,999%. Sau đó, nhóm quyết định thêm thuộc tính Vehicle\_Reference vào để làm khoá chính. Như vậy, bảng Casualities1114 sẽ có khoá chính là (Accident\_Index, Vehicle\_Reference, Casuality\_Reference).

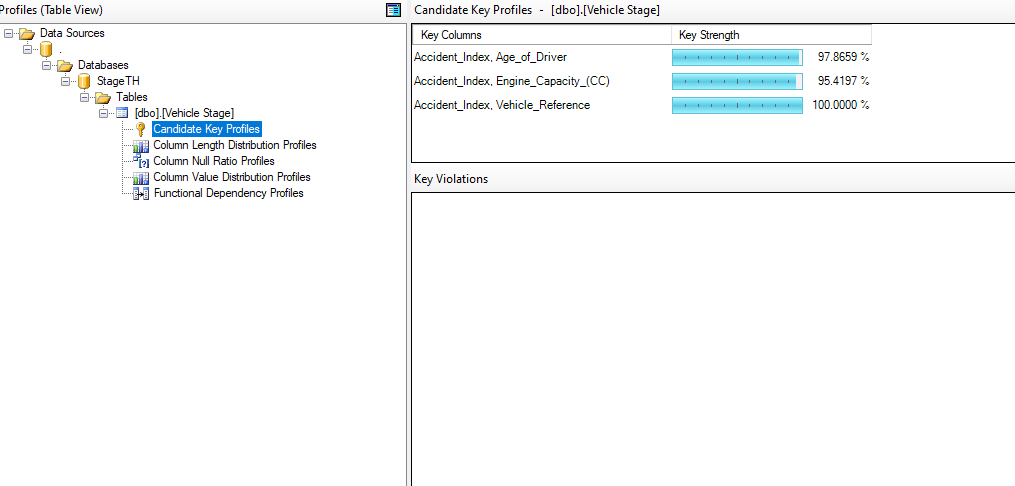
**Column Null Ratio**



Như hình trên, ta thấy các cột dữ liệu đều không mang giá trị null.

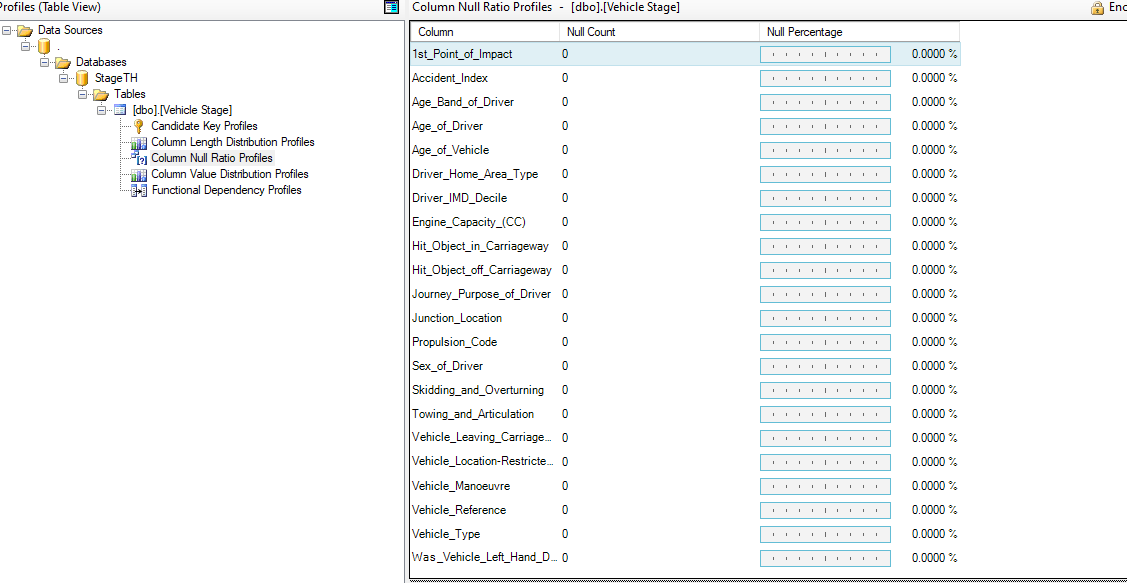
1. Vehicles

**Candidate Key**



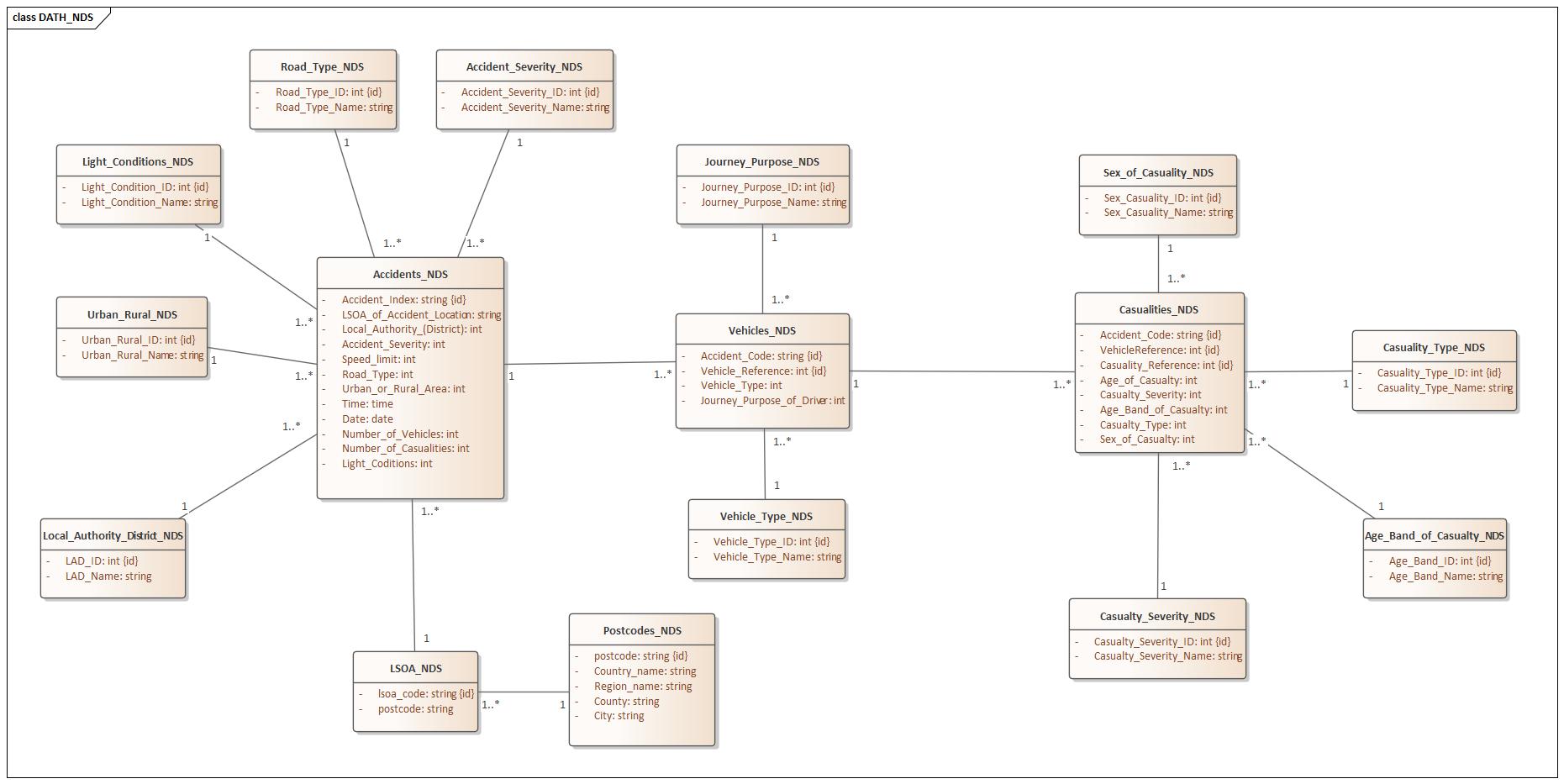
Theo như Data Profiling, bảng Vehicles1114 có khoá ứng viên gồm 2 thuộc tính (Accident\_Index, Casuality\_Reference) đạt 100%. Do đó, bảng Vehicles1114 sẽ có khoá chính là (Accident\_Index, Vehicle\_Reference).

**Column Null Ratio**



Như hình trên, ta thấy các cột dữ liệu đều không mang giá trị null.

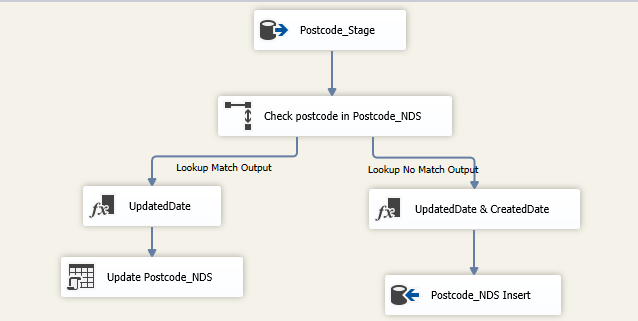
|  |
| --- |
| **Thiết kế NDS** |



|  |
| --- |
| **Mô tả quá trình ETL Stage 🡪 NDS** |

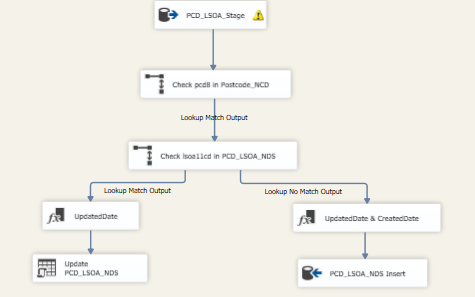
1. Bảng Postcode\_NDS

Đầu tiên sử dụng một OLE DB source để có thể lấy dữ liệu từ bảng Postcode\_Stage. Tiếp theo dùng lookup để so sánh khóa chính là Postcode của bảng Postcode\_NDS với khóa chính postcode của bảng Postcode\_Stage, nếu một dòng ở bảng Stage được tìm thấy ở bảng NDS thì sẽ được cho vào flow match output và từ flow này sẽ được dẫn đến derived column để thêm cột updatedDate có dữ liệu là ngày giờ hiện tại, tiếp sau đó sử dụng OLE DB command để cập nhật lại dữ liệu thông qua query. Ngược lại, ở flow no match sẽ sử dụng derived column để thêm vào hai cột createdDate và UpdatedDate được lấy theo ngày giờ hiện tại. Sau đó dùng OLE DB Destination để đổ dữ liệu.

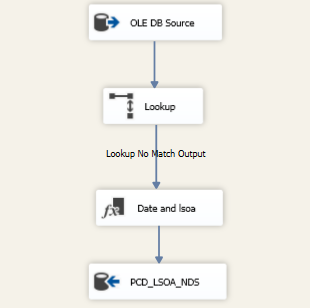


1. Bảng PCD\_LSOA\_NDS

Đầu tiên sử dụng một OLE DB source để có thể lấy dữ liệu từ bảng PCD\_LSOA\_Stage. Tiếp theo dùng lookup để so sánh khóa chính là lsoa11cd của bảng PCD\_LSOA\_NDS với bảng Postcode\_NDS. Nếu dòng trong bảng PCD\_LSOA\_NDS tìm thấy ở Postcode\_NDS thì được cho vào dòng flow match output. Pcd của bảng PCD\_LSOA\_NDS với khóa chính postcode của bảng PCD\_LSOA\_Stage, nếu một dòng ở bảng Stage được tìm thấy ở bảng NDS thì sẽ được cho vào flow match output và từ flow này sẽ được dẫn đến derived column để thêm cột updatedDate có dữ liệu là ngày giờ hiện tại, tiếp sau đó sử dụng OLE DB command để cập nhật lại dữ liệu thông qua query. Ngược lại, ở flow no match sẽ sử dụng derived column để thêm vào hai cột createdDate và UpdatedDate được lấy theo ngày giờ hiện tại. Sau đó dùng OLE DB Destination để đổ dữ liệu.



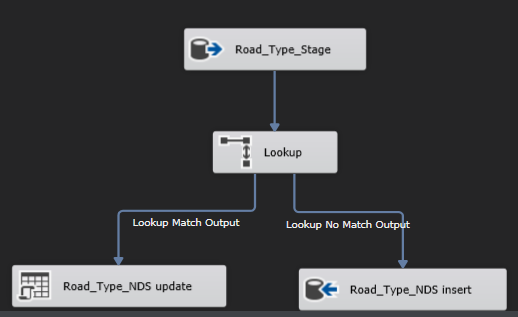
Tuy nhiên ở bảng Postcode\_Stage có một số hàng pcd không có lsoa, nên em đã thực hiện thêm lsoa vào những hàng pcd bị trống lsoa. Đầu tiên em sẽ sử dụng một OLE DB Source để nạp dữ liệu từ Postcode\_NDS. Sau đó sẽ lookup dữ liệu đến PCD\_LSOA\_NDS để so sánh xem mã Postcode của Potscode\_NDS đã tồn tại bên PCD\_LSOA\_NDS chưa, nếu chưa thì sẽ vào Lookup No Match Output, ở đây sẽ thêm ba cột là lsoa11cd, CreatedDate, UpdatedDate. Ở cột lsoa11cd thì sẽ lấy dữ liệu là mã postcode, còn hai cột CreatedDate và UpdatedDate thì sẽ lấy theo ngày và giờ hiện tại.



1. Các bảng cha của Accidents\_NDS

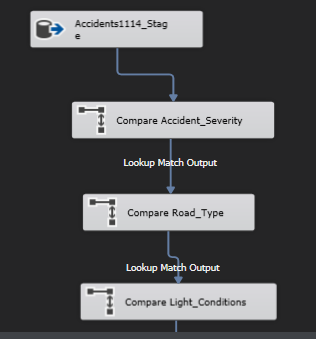
Bao gồm các bảng: Local\_Authority\_District\_NDS, Accident\_Severity\_NDS, Urban\_Rural\_NDS, Light\_Conditions\_NDS, Road\_Type\_NDS

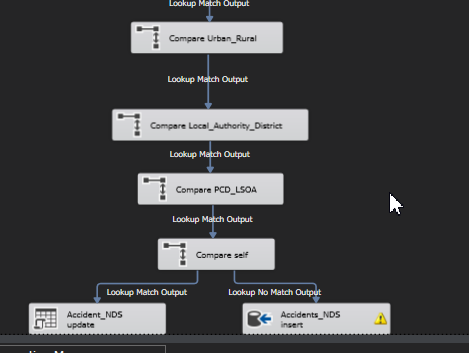
Mô tả: Các bảng này có cùng một dataflow như nhau. Lấy bảng Road\_Type\_NDS làm ví dụ, đầu tiên nhóm sử dụng một OLE DB source để có thể lấy dữ liệu từ db StageTH bảng Road\_Type\_Stage. Tiếp theo nhóm sử dụng lookup để so sánh dữ liệu từ Stage và NDS, nếu một dòng ở Stage được tìm thấy ở NDS thì sẽ vào flow match output và ngược lại. Flow match output sẽ sử dụng OLE DB command để update dữ liệu thông qua query, ngược lại flow no match sẽ sử dụng OLE DB Destination để đổ (insert) dữ liệu. Tương tự cho các bảng còn lại.



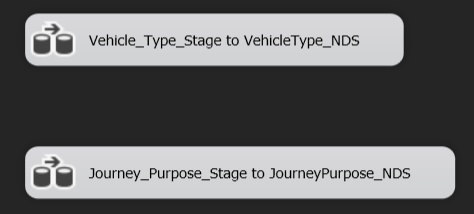
* 1. Bảng Accidents\_NDS

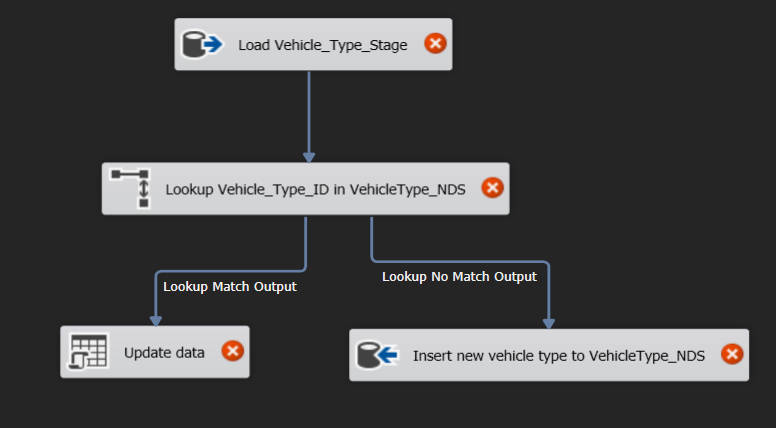
Mô tả: Khởi đầu bằng việc đọc dữ liệu từ bảng Accidents1114\_Stage ở db StageTH bằng OLE DB Source. Sau đó lần lượt dùng lookup để so sánh các khóa ngoại liên quan đến các bảng cha ở trên để xem có tồn tại ở các bảng cha chưa, nếu có thì sẽ theo flow match để tiếp tục và ngược lại thì sẽ bỏ qua dòng đó. Sau khi đã xong các lookup khóa ngoại, nhóm tiếp tục sử dụng lookup một lần nữa để so sánh xem mã tai nạn (Accident\_Index) đã tồn tại ở NDS hay chưa, nếu có thì vào flow match và ngược lại. Cuối cùng flow match output sẽ sử dụng OLE DB command để update dữ liệu thông qua query, ngược lại flow no match sẽ sử dụng OLE DB Destination để đổ (insert) dữ liệu vào NDS.



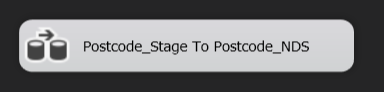


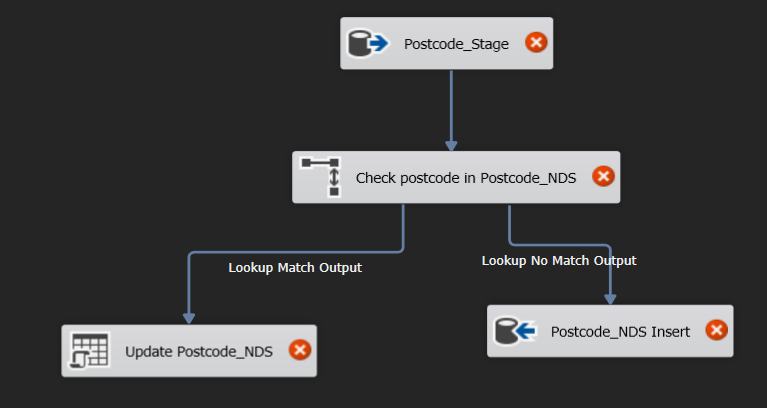
1. Các bảng cha của Vehicles\_NDS







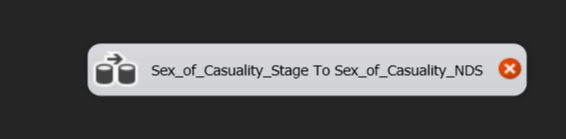


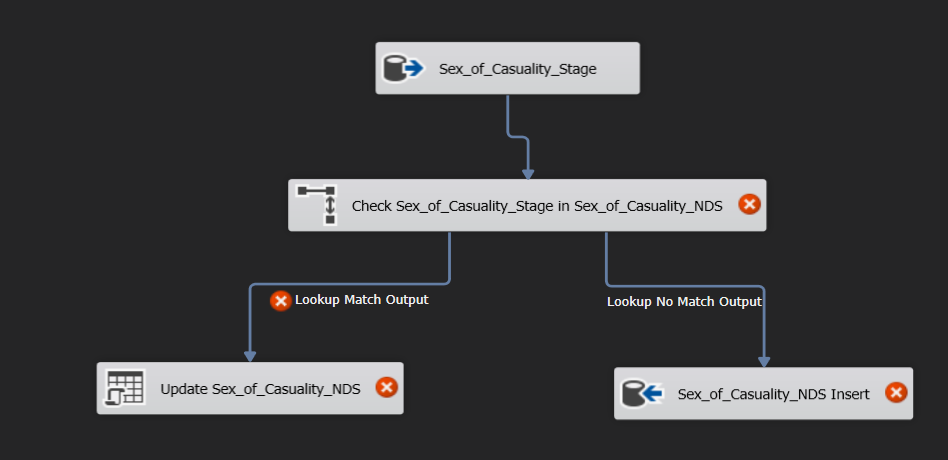


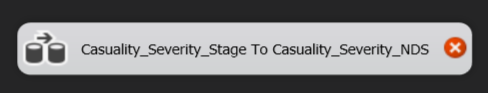
1. Bảng Vehicles\_NDS

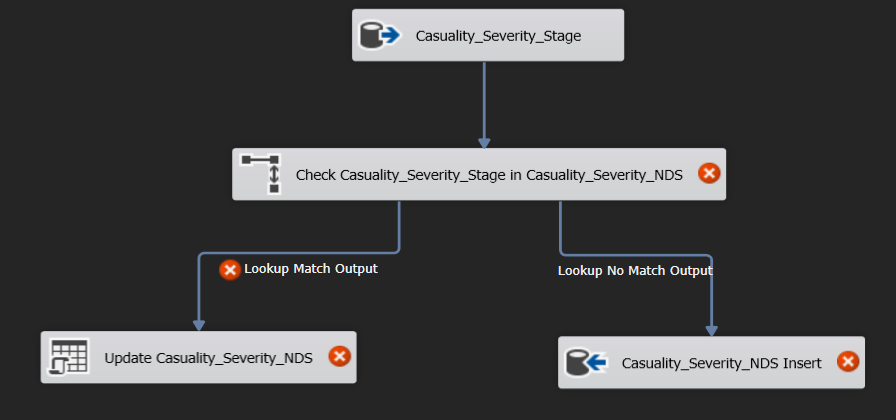
Mô tả: Khởi đầu bằng việc đọc dữ liệu từ bảng Vehicles1114\_Stage ở db StageTH bằng OLE DB Source. Sau đó lần lượt dùng lookup để so sánh các khóa ngoại liên quan đến các bảng cha ở trên để xem có tồn tại ở các bảng cha chưa, nếu có thì sẽ theo flow match để tiếp tục và ngược lại thì sẽ bỏ qua dòng đó. Sau khi đã xong các lookup khóa ngoại, nhóm tiếp tục sử dụng lookup một lần nữa để so sánh xem mã tai nạn (Accident\_Index và Vehicles\_reference) đã tồn tại ở NDS hay chưa, nếu có thì vào flow match và ngược lại. Cuối cùng flow match output sẽ sử dụng OLE DB command để update dữ liệu thông qua query, ngược lại flow no match sẽ sử dụng OLE DB Destination để đổ (insert) dữ liệu vào NDS.

1. Các bảng cha của Casualities\_NDS

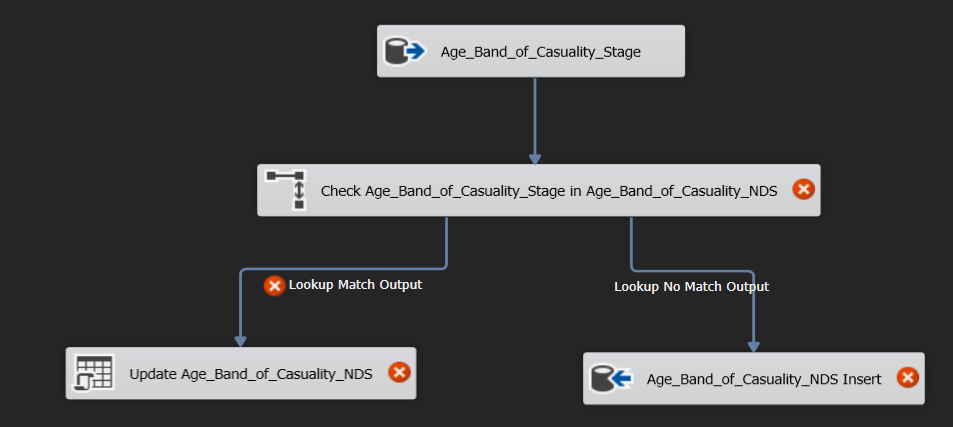


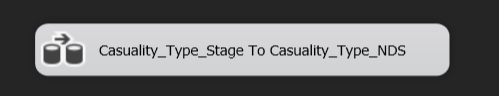


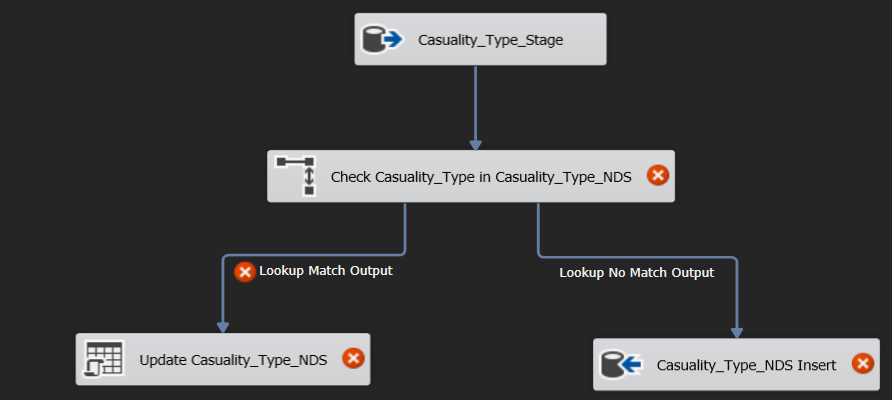






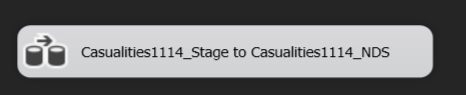


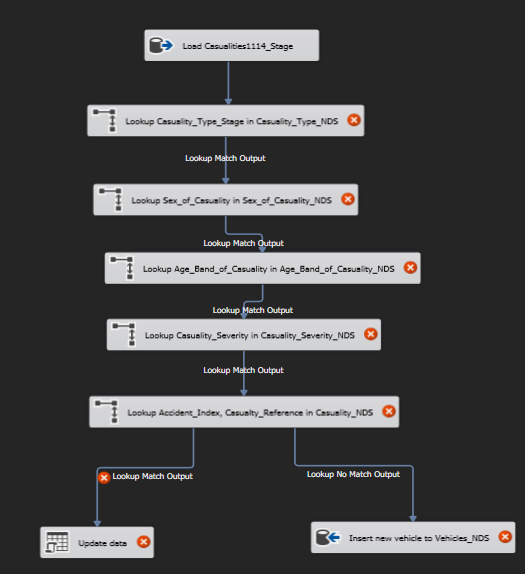




1. Bảng Casualities\_NDS

Mô tả: Khởi đầu bằng việc đọc dữ liệu từ bảng Casualities1114\_Stage ở db StageTH bằng OLE DB Source. Sau đó lần lượt dùng lookup để so sánh các khóa ngoại liên quan đến các bảng cha ở trên để xem có tồn tại ở các bảng cha chưa, nếu có thì sẽ theo flow match để tiếp tục và ngược lại thì sẽ bỏ qua dòng đó. Sau khi đã xong các lookup khóa ngoại, nhóm tiếp tục sử dụng lookup một lần nữa để so sánh xem mã tai nạn (Accident\_Index và Casuilities\_reference) đã tồn tại ở NDS hay chưa, nếu có thì vào flow match và ngược lại. Cuối cùng flow match output sẽ sử dụng OLE DB command để update dữ liệu thông qua query, ngược lại flow no match sẽ sử dụng OLE DB Destination để đổ (insert) dữ liệu vào NDS.

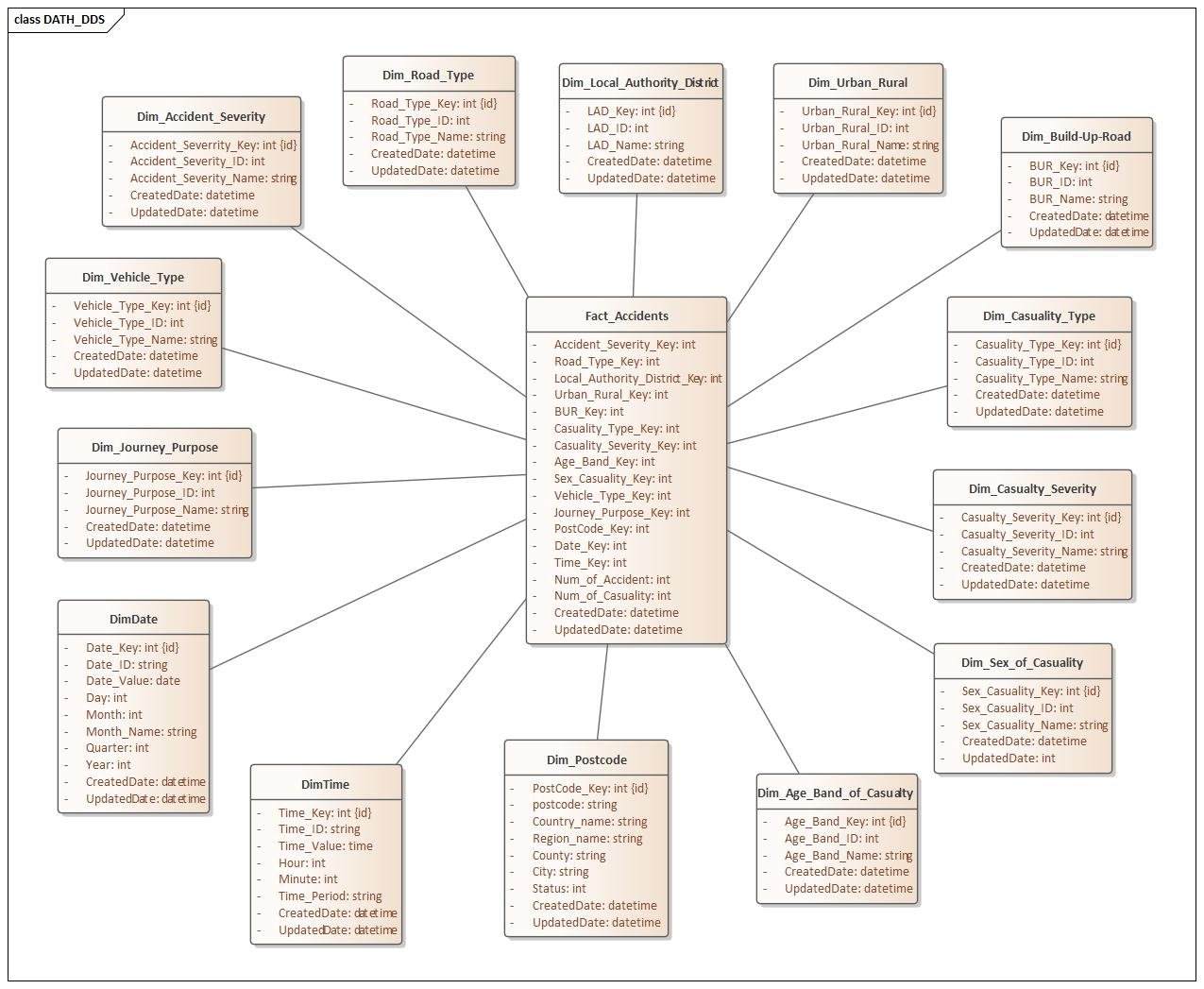




|  |
| --- |
| **Thiết kế DDS** |

Do yêu cầu từ đề bài cần đến các measure như số lượng tai nạn và số lượng người bị thương nên không thể sử dụng bảng Accident\_NDS để tạo thành bảng Fact được. Nhóm đã quyết định mỗi dòng (đơn vị nhỏ nhất) trong bảng Fact sẽ là một loại tai nạn được xác định bằng nhiều khóa ngoại từ các Dimension như loại tai nạn, mã vị trí, thời gian trong ngày và có 2 measure như trên. Như vậy, mỗi dòng trong đó biểu thị số lần một loại tai nạn xảy ra.

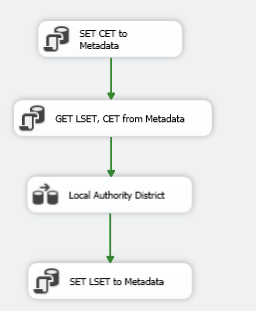
Các bảng dimension được xác định gồm 14 bảng được tạo và ETL từ các bảng cha của Accident, Vehicle và Casuality ở NDS.



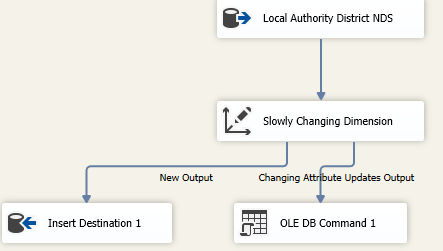
1. Mô tả quá trình ETL từ NDS đến DDS
   1. Các bảng Dimension

i. Dim\_Local\_Authority\_District

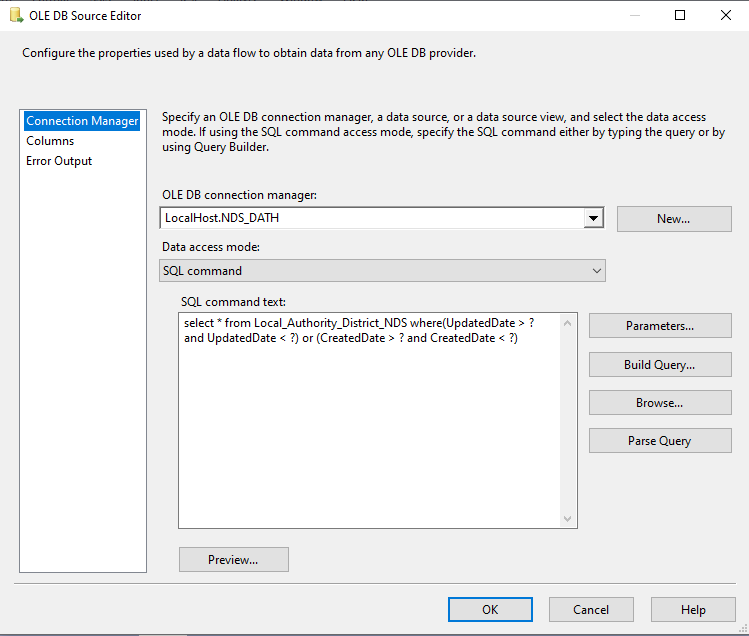
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Local\_Authority\_District mỗi khi thực hiện ETL.



Data flow Dim\_Local\_Authority\_District



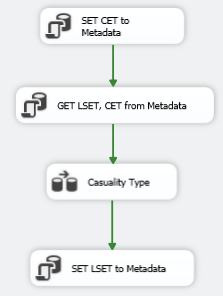
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Local\_Authority\_District\_NDS với điều kiện như sau:



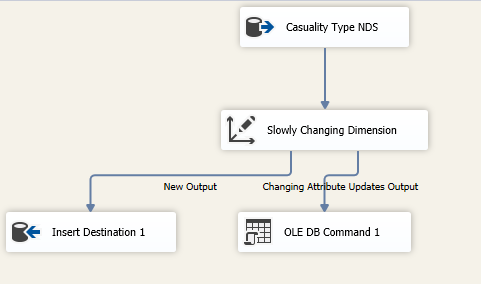
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn LAD\_ID làm business key. Bởi vì bảng Dim\_Local\_Authority\_District là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

ii. Dim\_Casuality\_Type

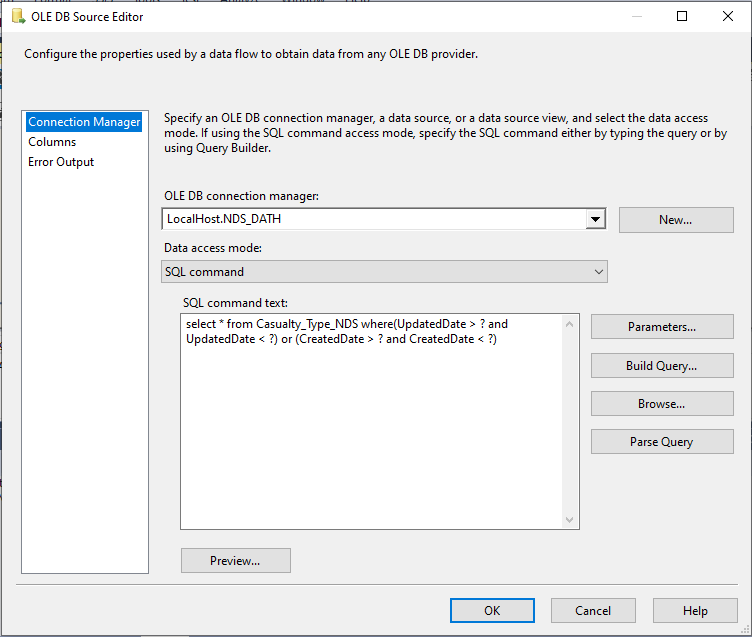
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Casuality\_Type mỗi khi thực hiện ETL.



Data flow Dim\_Casuality\_Type



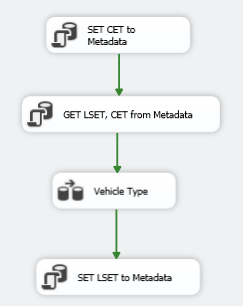
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Casuality\_Type\_NDS với điều kiện như sau:



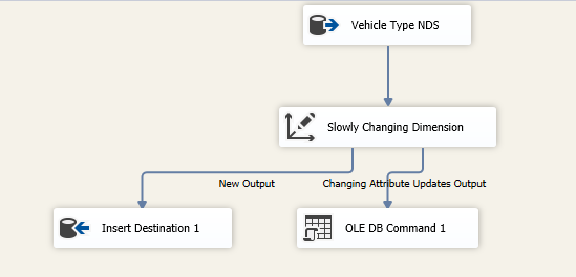
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Casuality\_Type\_ID làm business key. Bởi vì bảng Dim\_Casuality\_Type là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

iii. Dim\_Vehicle\_Type

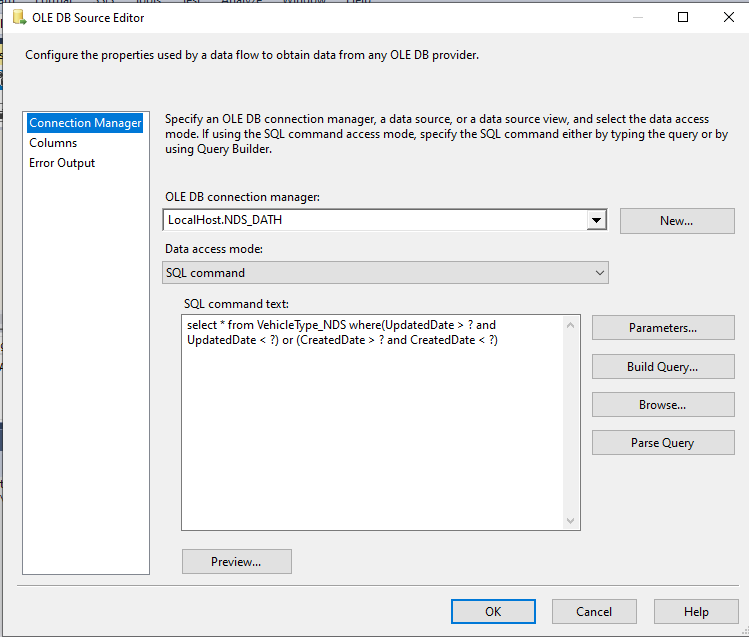
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Vehicle\_Type mỗi khi thực hiện ETL.



Data flow Dim\_Vehicle\_Type



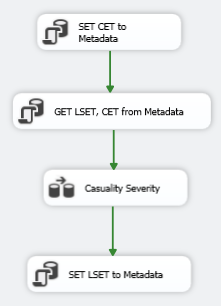
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Vehicle\_Type\_NDS với điều kiện như sau:



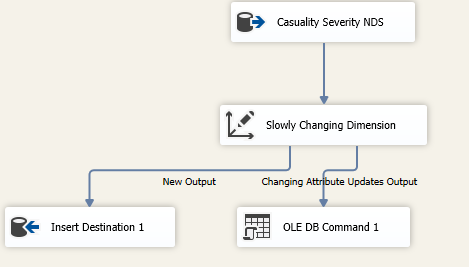
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Vehicle\_Type\_ID làm business key. Bởi vì bảng Dim\_Vehicle\_Type là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

iv. Dim\_Casuality\_Severity

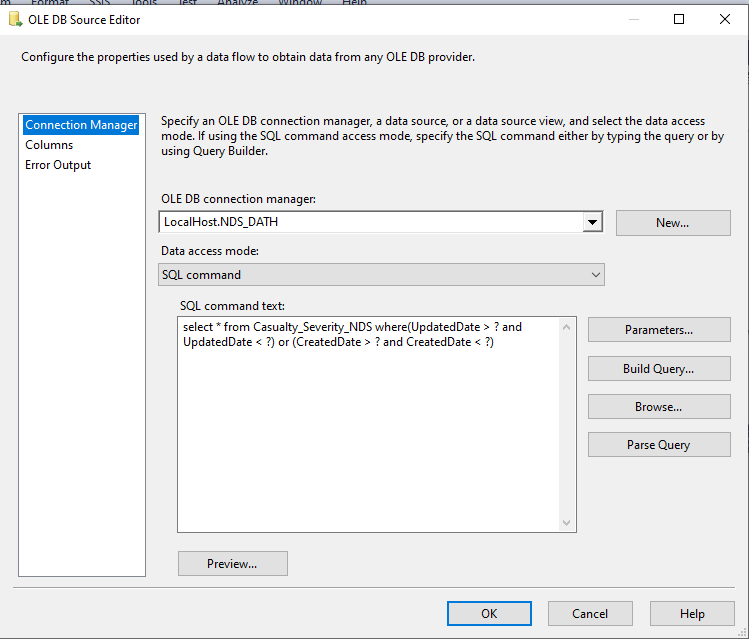
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Casuality\_Severity mỗi khi thực hiện ETL.



Data flow Dim\_Casuality\_Severity



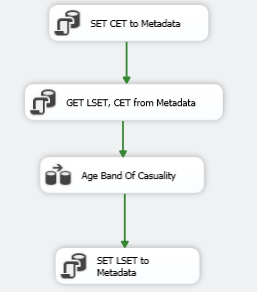
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Casuality\_Severity\_NDS với điều kiện như sau:



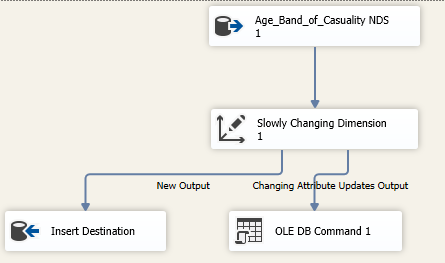
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Casuality\_Severity\_ID làm business key. Bởi vì bảng Dim\_Casuality\_Severity là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

v. Dim\_Age\_Band\_Of\_Casuality

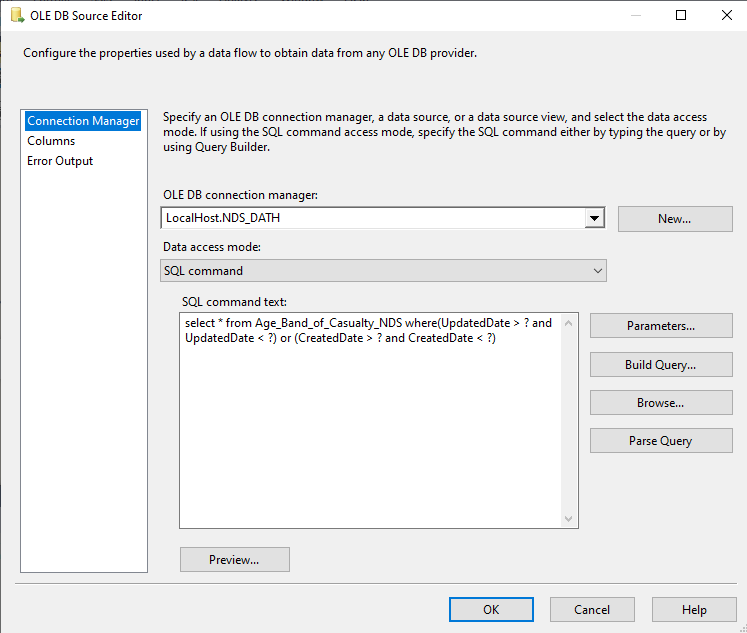
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Age\_Band\_Of\_Casuality mỗi khi thực hiện ETL.



Data flow Dim\_Age\_Band\_Of\_Casuality



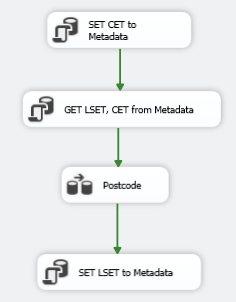
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Age\_Band\_Of\_Casuality\_NDS với điều kiện như sau:



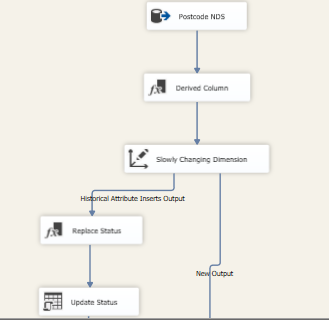
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Age\_Band\_ID làm business key. Bởi vì bảng Dim\_Age\_Band\_Of\_Casuality là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

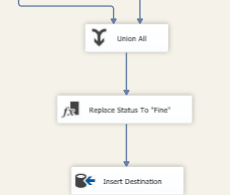
vi. Dim\_Postcode

Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Postcode mỗi khi thực hiện ETL.

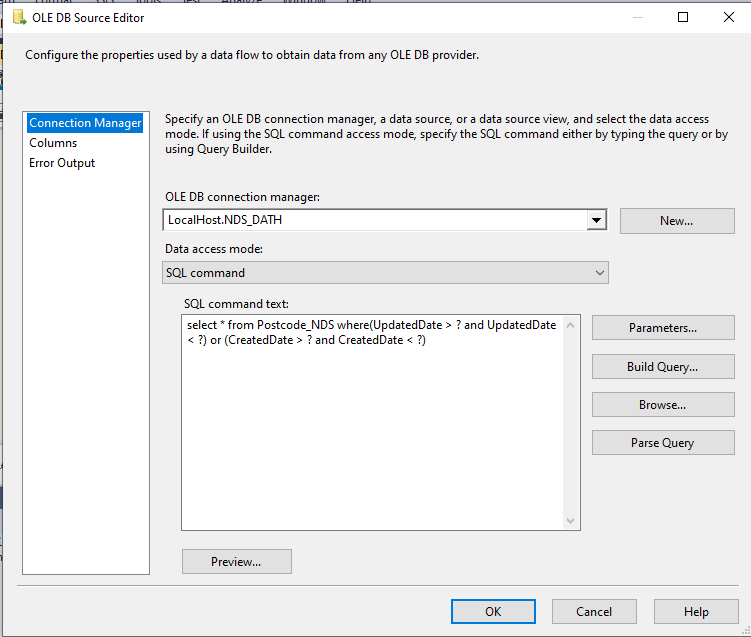


Data flow Dim\_Postcode





Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Postcode\_NDS với điều kiện như sau:

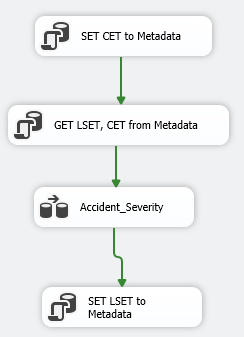


Tiếp theo em dùng Derived column để thêm một cột Status, giá trị trong cột này được đặt mặc định là Fine.

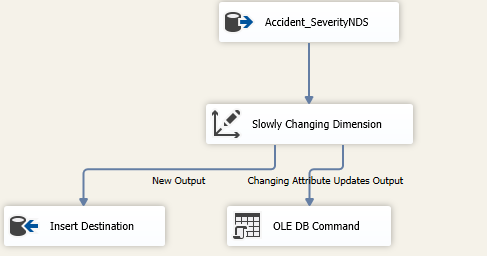
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Postcode\_ID làm business key. Bởi vì bảng Dim\_Postcode là scd loại 2 vì cần phải lưu giữ những giá trị cũ cũng như có nhiều thuộc tính thay đổi. Cho nên những thuộc tính trong Attribute column đều được chỉnh về Historical Attribute. Tiếp theo đó chọn Using single column to show current and expired record. Ở đây em chọn Statu, và giá trị hiện tại em set là Fine, và giá trị kết thúc(Expiration value) sẽ set thành Warm.

vii. Dim\_Accident\_Severity

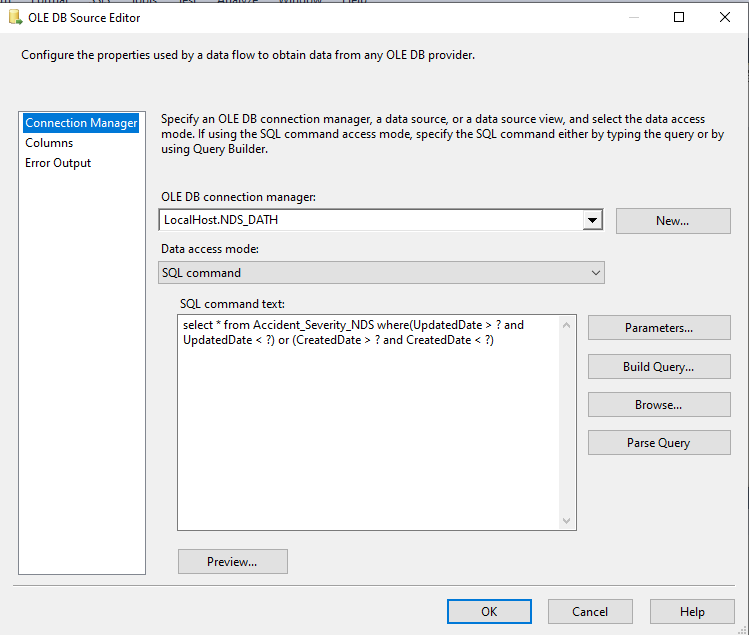
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Accident\_Severity mỗi khi thực hiện ETL.



Data flow Dim\_Accident\_Severity



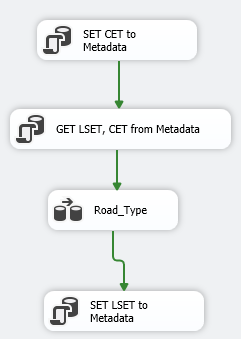
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Accident\_Severity\_NDS với điều kiện như sau:



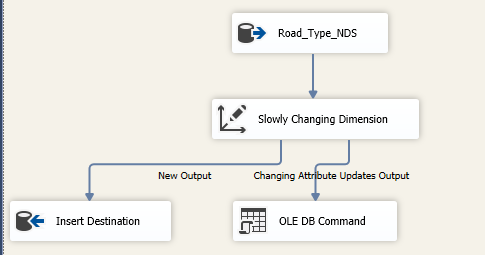
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Accident\_Severity\_ID làm business key. Bởi vì bảng Dim\_Accident\_Severity là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

viii. Dim\_Road\_Type

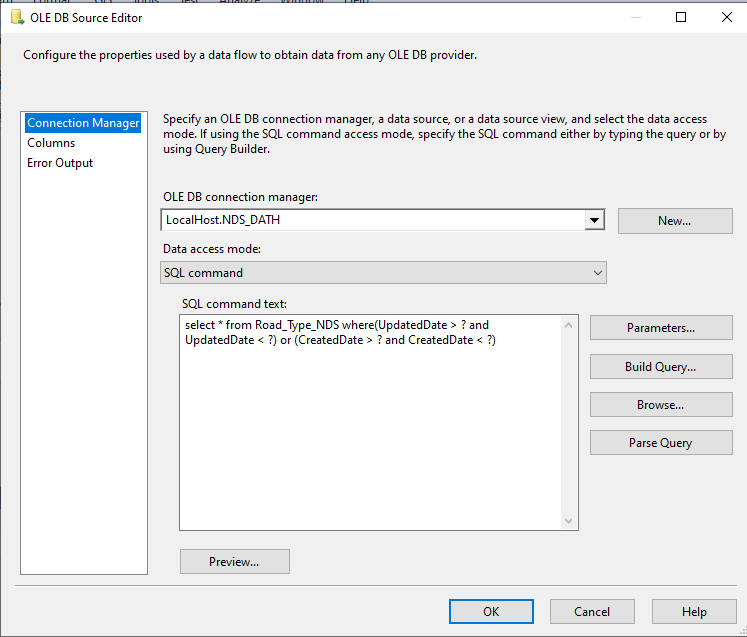
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Road\_Type mỗi khi thực hiện ETL.



Data flow Dim\_Road\_Type



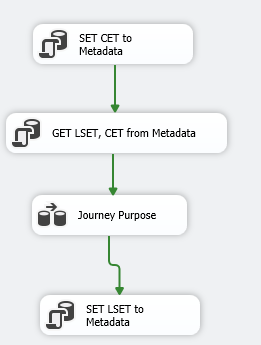
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Road\_Type\_NDS với điều kiện như sau:



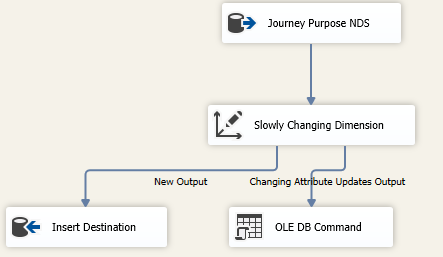
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Road\_Type\_ID làm business key. Bởi vì bảng Dim\_Road\_Type là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

ix. Dim\_Journey\_Purpose

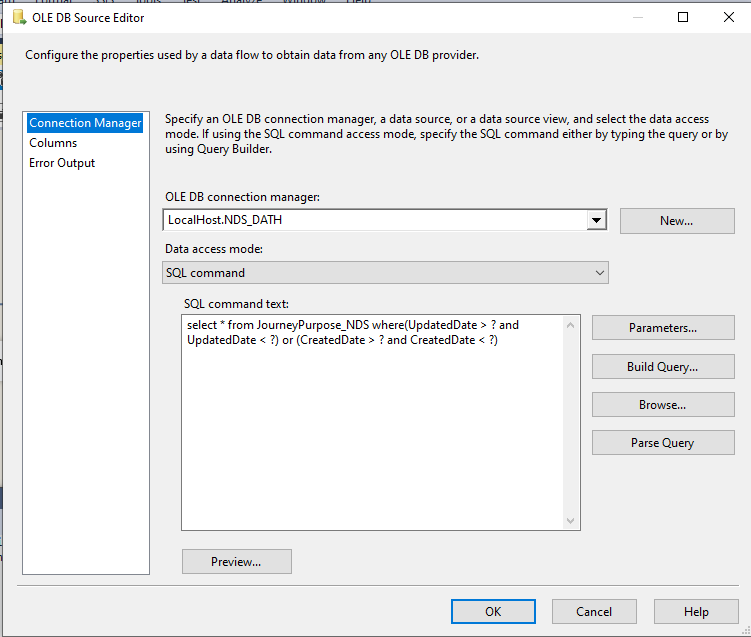
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Journey\_Purpose mỗi khi thực hiện ETL.



Data flow Dim\_Journey\_Purpose



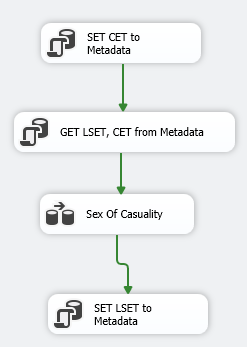
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Journey\_Purpose\_NDS với điều kiện như sau:



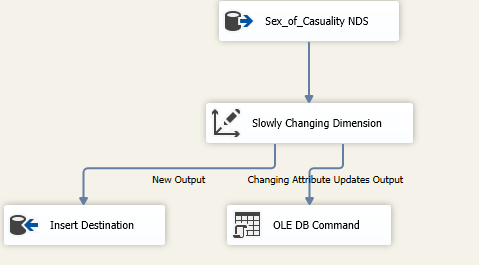
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Journey\_Purpose\_ID làm business key. Bởi vì bảng Dim\_Journey\_Purpose là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

x. Dim\_Sex\_Of\_Casuality

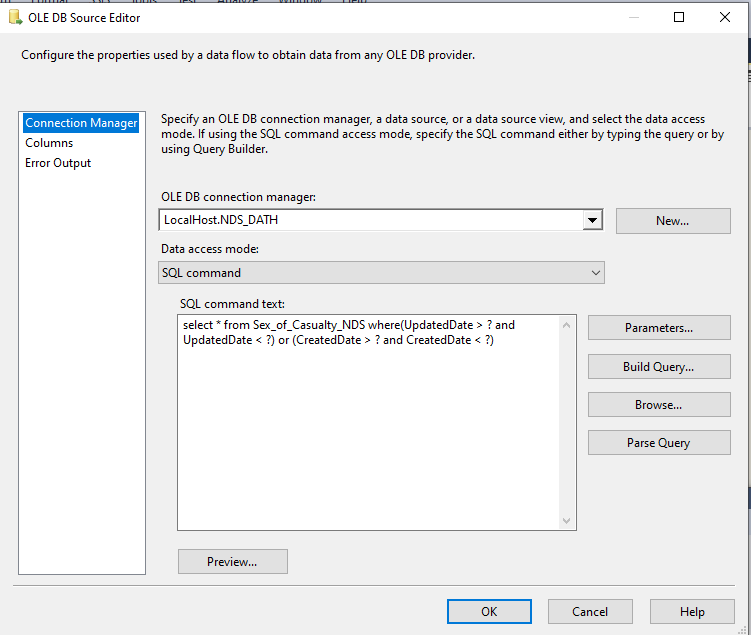
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Sex\_Of\_Casuality mỗi khi thực hiện ETL.



Data flow Dim\_Sex\_Of\_Casuality



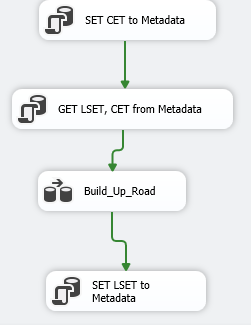
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Sex\_Of\_Casuality\_NDS với điều kiện như sau:



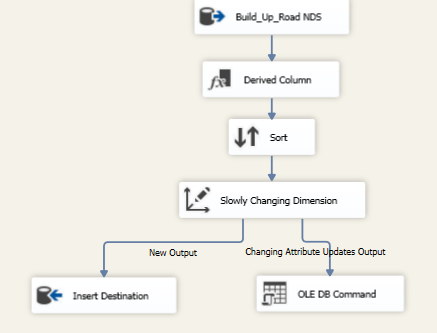
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Sex\_Casuality\_ID làm business key. Bởi vì bảng Dim\_Sex\_Of\_Casuality là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

xi. Dim\_Buil\_Up\_Road

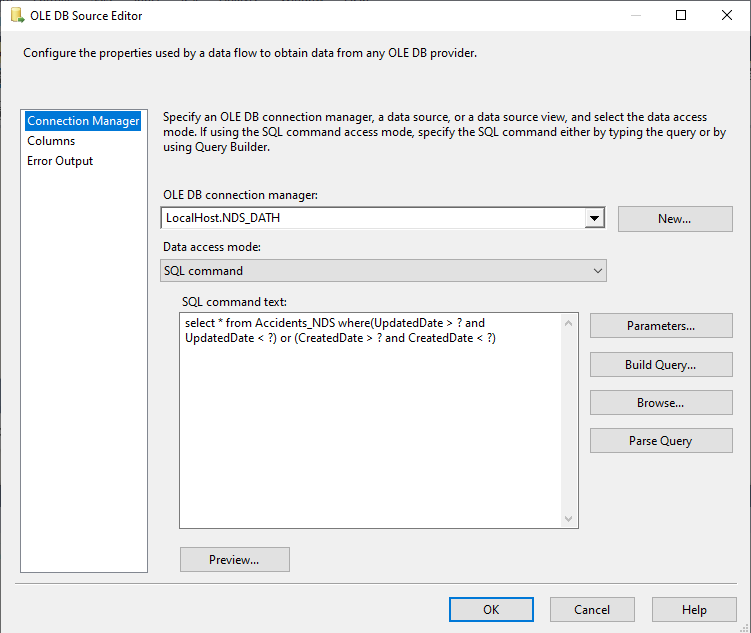
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Buil\_Up\_Road mỗi khi thực hiện ETL.



Data flow Dim\_Buil\_Up\_Road



Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Accident\_NDS với điều kiện như sau:

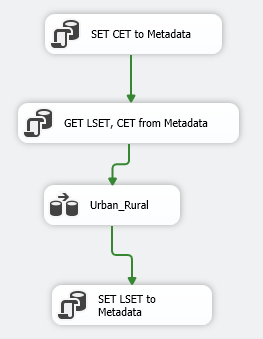


Tiếp theo dùng Derived column để đặt BUR\_ID, nếu là Build\_Up\_Road thì BUR\_ID là 1 còn lại là 2. Sử dụng Sort để loại bỏ đi các dòng có thể bị trùng trong Accident\_NDS.

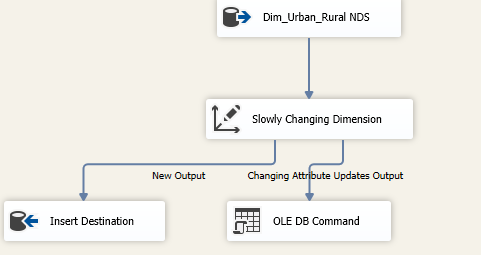
Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn BUR\_ID làm business key. Bởi vì bảng Dim\_Buil\_Up\_Road là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

xii. Dim\_Urban\_Rural

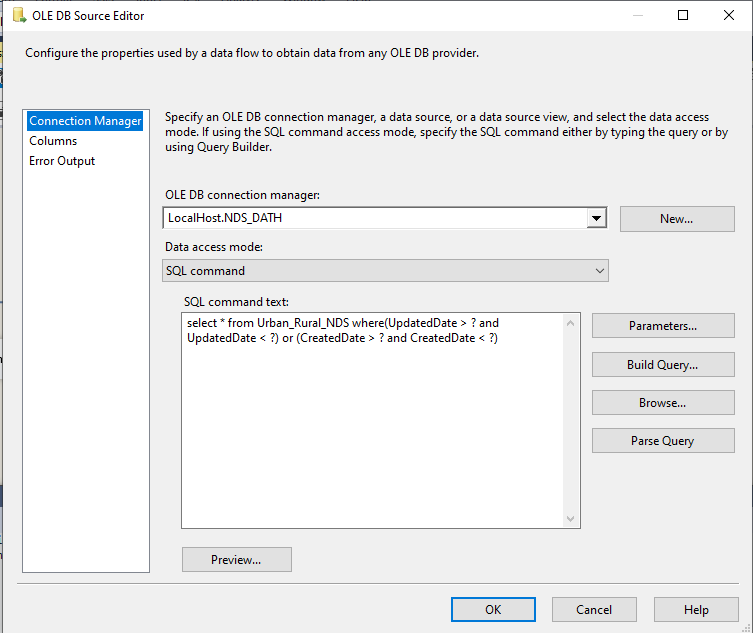
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng Dim\_Urban\_Rural mỗi khi thực hiện ETL.



Data flow Dim\_Urban\_Rural



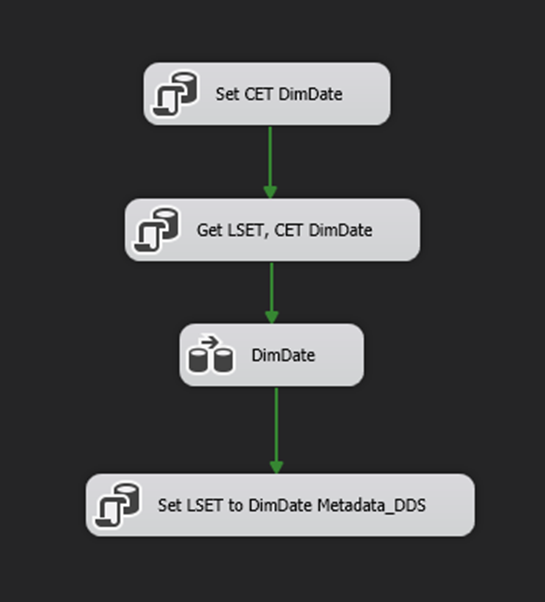
Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Urban\_Rural\_NDS với điều kiện như sau:

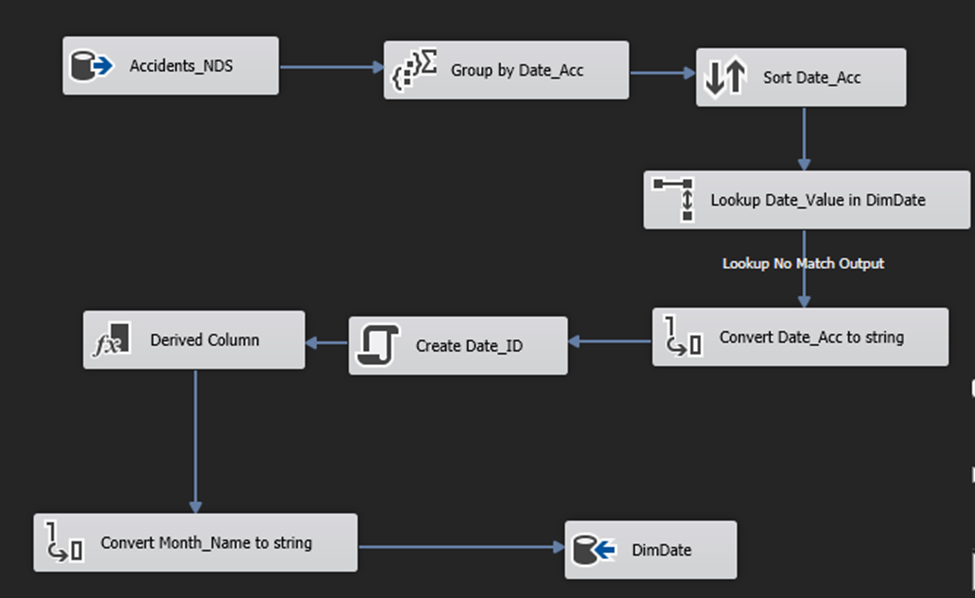


Sau đó sử dụng Slowly changing Dimension để thực hiện ghi nhận thay đổi. Chọn Urban\_Rural\_ID làm business key. Bởi vì bảng Dim\_Urban\_Rural là scd loại 1 chỉ cần ghi đè lên giá trị cũ, nên những Dimension column đều được chỉnh Change Type về Changing Attribute.

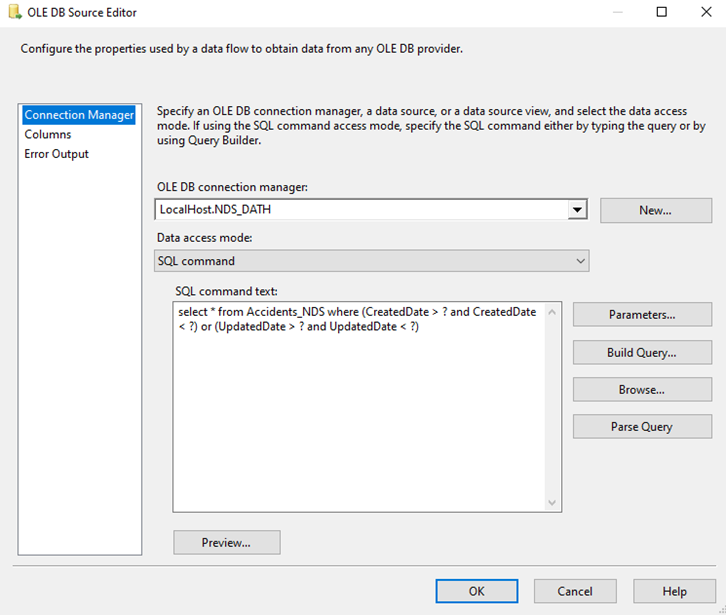
* 1. DimDate

Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng DimDate mỗi khi thực hiện ETL.



Data flow DimDate

Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Accidents\_NDS với điều kiện như sau:



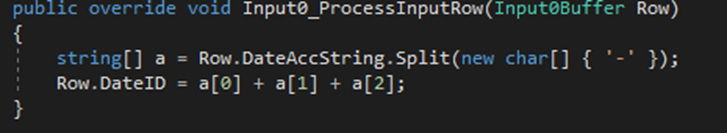
Tiếp theo, sử dụng Aggregate để group by dữ liệu theo Date\_Acc để lấy được những ngày xảy ra tai nạn được ghi nhận lại trong bảng Accidents\_NDS.



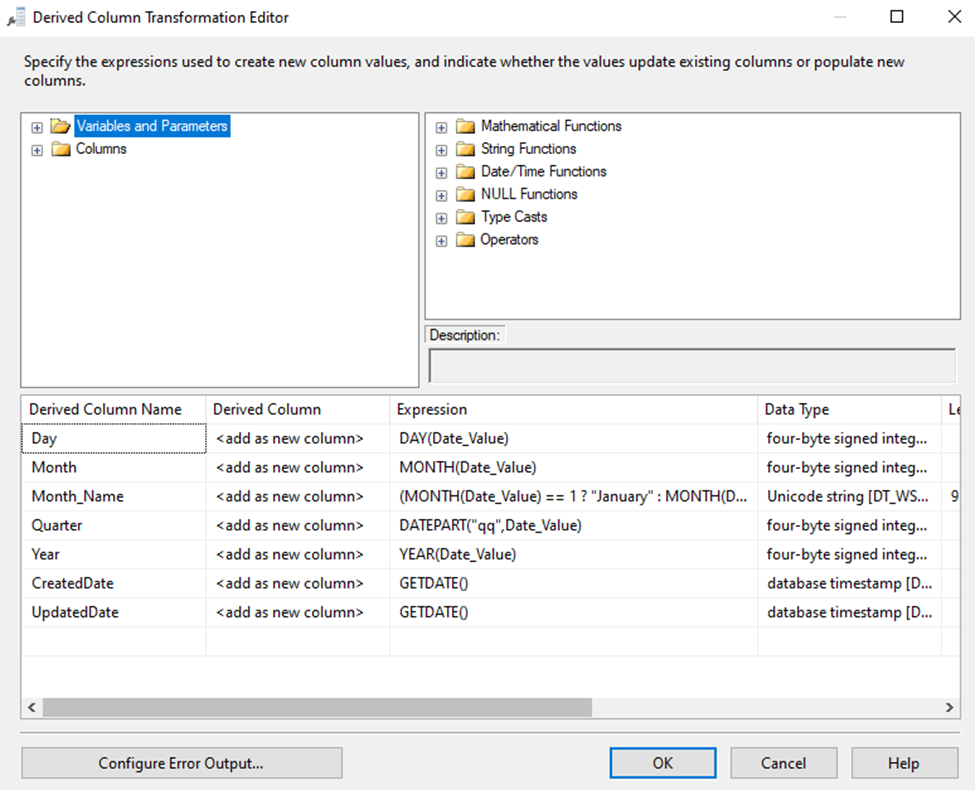
Tiếp theo, tiến hành Sort dữ liệu Date\_Acc và thực hiện Lookup giá trị Date\_Acc có match với Date\_Value trong bảng DimDate chưa, nếu match thì ngày đã tồn tại trong bảng DimDate nên sẽ bỏ qua, trường hợp no match sẽ tiến hành tạo dòng dữ liệu mới trong bảng DimDate.

Tạo dòng dữ liệu mới trong bảng DimDate:

Sử dụng Data Conversion chuyển Date\_Acc sang kiểu string và sử dụng Script Component để tạo Date\_ID có định dạng yyyymmdd.



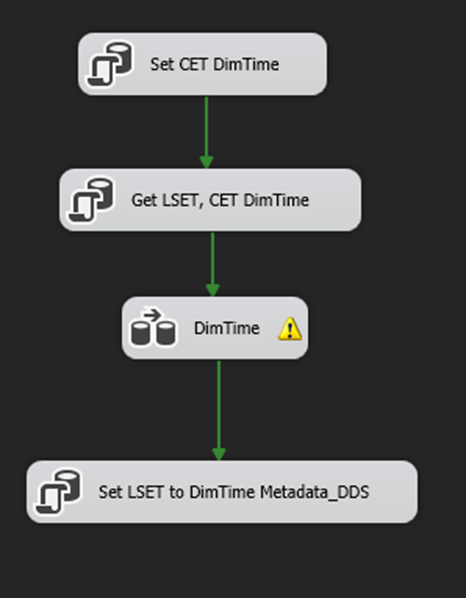
Sử dụng Derived Column và những hàm được cung cấp để lấy những giá trị cần thiết trong bảng DimDate.



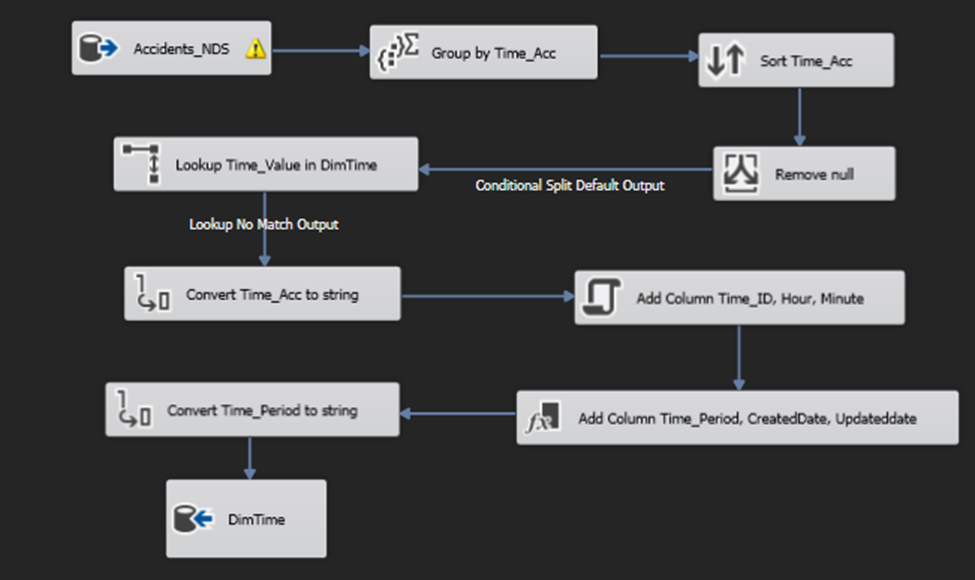
Cuối cùng, thực hiện Insert dữ liệu vào bảng DimDate

* 1. DimTime

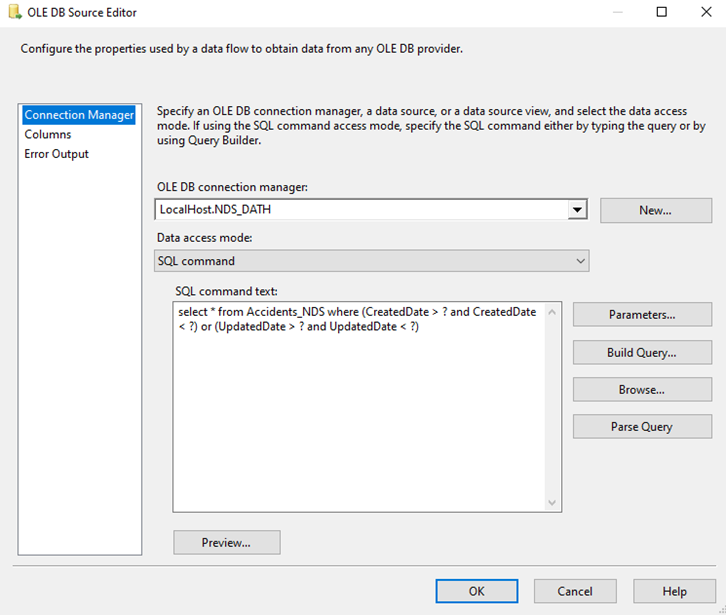
Ở phần control flow, thực hiện incremental ETL sử dụng CET và LSET của bảng DimTime mỗi khi thực hiện ETL.



Data flow DimTime



Đầu tiên ta sử dụng OLE DB Source để load dữ liệu từ bảng Accidents\_NDS với điều kiện như sau:



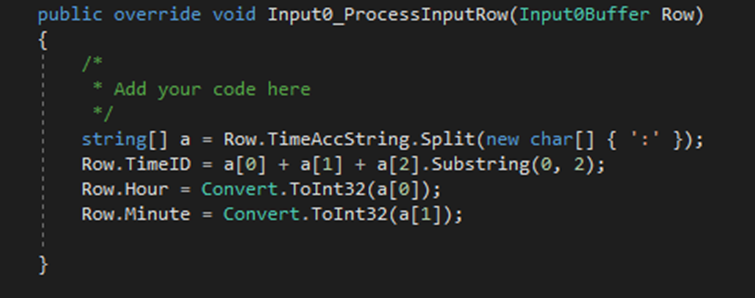
Tiếp theo, sử dụng Aggregate để group by dữ liệu theo Time\_Acc để lấy được những thời điểm xảy ra tai nạn được ghi nhận lại trong bảng Accidents\_NDS.



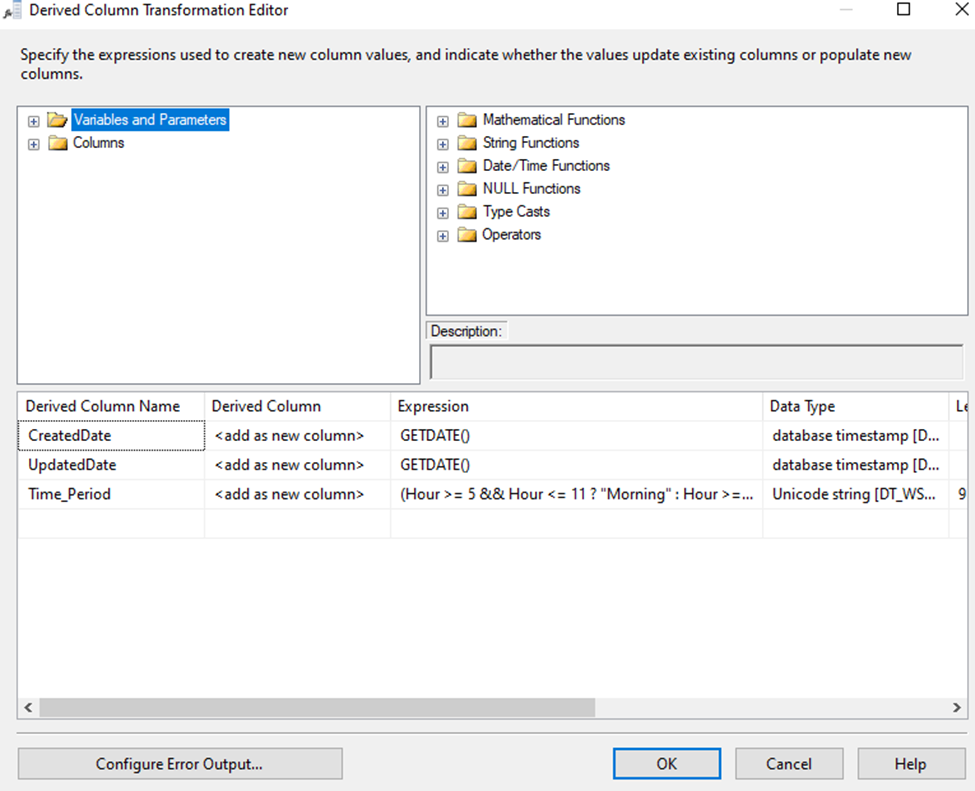
Tiếp theo, tiến hành Sort dữ liệu Time\_Acc, sau đó sử dụng Conditional Split để xoá các dòng dữ liệu mang giá trị null và thực hiện Lookup giá trị Time\_Acc có match với Time\_Value trong bảng DimTime chưa, nếu match thì ngày đã tồn tại trong bảng DimTime nên sẽ bỏ qua, trường hợp no match sẽ tiến hành tạo dòng dữ liệu mới trong bảng DimTime.

Tạo dòng dữ liệu mới trong bảng DimTime:

Sử dụng Data Conversion chuyển Date\_Acc sang kiểu string và sử dụng Script Component để tạo Time\_ID có định dạng hhmmss và các giá trị Hour và Minute.



Sử dụng Derived Column và những hàm được cung cấp để lấy những giá trị cần thiết còn lại trong bảng DimTime, trong đó Time\_Period là khoảng thời gian trong ngày tương ứng với thời điểm đó được quy định theo đề bài.

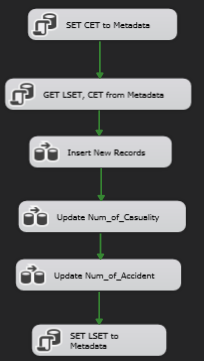


Cuối cùng, thực hiện Insert dữ liệu vào bảng DimTime.

* 1. Bảng Fact\_Accident
     1. Ý tưởng bên ngoài

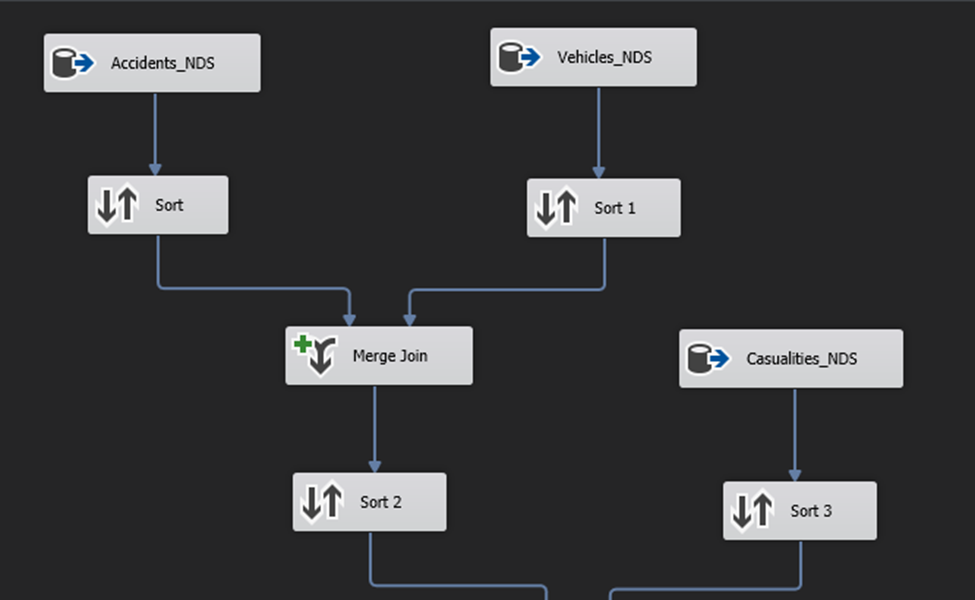
Vì ở NDS và DDS nhóm đều sử dụng incremental ETL nên flow chính vẫn sẽ phải có việc SET và GET LSET CET từ bảng metadata phù hợp để lấy được dữ liệu từ 3 bảng chính của NDS là Accident, Vehicle và Casuality.

Do việc merge join dữ liệu và các trường measure thuộc dạng đếm nên nhóm đã phân flow chính của việc ETL như sau: khởi tạo các giá trị loại tai nạn còn thiếu trong bảng Fact với 2 loại measure bằng 0 → Nạp dữ liệu cho measure num\_of\_casuality bằng việc đếm → Nạp dữ liệu cho measure num\_of\_accidents bằng việc đếm có loại bỏ trùng.

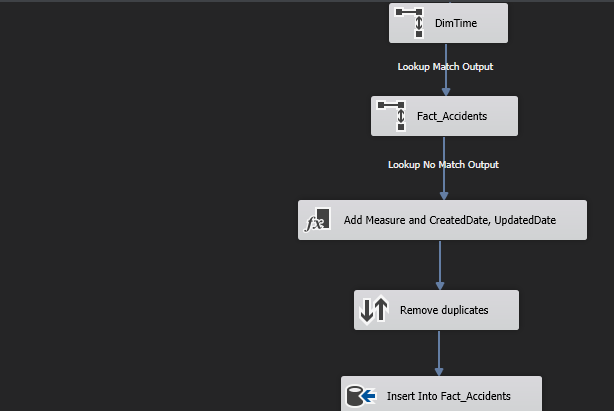


* + 1. Thực hiện bên trong

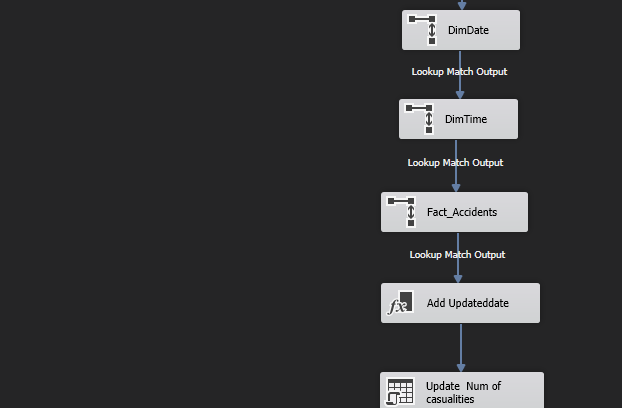
Khởi tạo các giá trị loại tai nạn còn thiếu trong bảng Fact: Đầu tiên là các bước merge join các bảng chính ở NDS và các bảng cha ở NDS nếu cần để lấy data nguồn từ NDS. Sau đó thực hiện lần lượt 14 lần lookup các bảng dimension để so sánh và lấy ra các key cần cho bảng Fact. Lần lookup thứ 15 được dùng để xác định tổ hợp các key kể trên đã xuất hiện trong bảng Fact chưa hay nói đơn giản hơn là loại tai nạn đó đã xuất hiện trong bảng Fact chưa. Vì mục đích của bước này là thêm các loại tai nạn chưa xuất hiện trong Fact nên ta chỉ cần flow no match output của lookup thứ 15. Cuối cùng nhóm sử dụng derived column để thêm 2 loại measure có giá trị bằng 0 vào kết quả lookup và sử dụng sort để xóa các dòng bị trùng tổ hợp key trước khi đổ vào DDS.



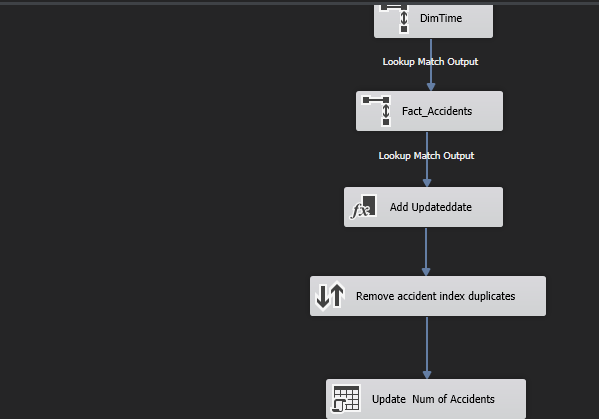




Nạp dữ liệu cho measure num\_of\_casuality: Đầu tiên là các bước merge join cũng như lookup như bước đầu để lọc dữ liệu từ NDS. Điểm khác biệt ở bước này là nhóm sử dụng match output của lookup thứ 15 để bắt đầu update dữ liệu và không cần lọc trùng dữ liệu vì số dòng được kết ở trên đúng chính xác bằng số lượng người bị thương. Cuối cùng nhóm dùng derived column để sửa lại cột updateddate đúng vào thời điểm update và OLE DB command để update lại các cột updated date và num\_of\_casuality = num\_of\_casuality + 1.



Nạp dữ liệu cho measure num\_of\_casuality: Bước này gần như giống hoàn toàn với bước trên từ đầu đến lúc lookup xong. Điểm khác biệt ở bước này là nhóm cần lọc trùng dữ liệu vì số dòng được kết ở trên có thể bị trùng accident\_index và bị đếm 2 lần cho một tai nạn. Vì vậy nhóm sử dụng sort để lọc trùng các trường key của fact và accident\_index để đảm bảo. Các bước cuối cùng tương tự nhóm dùng derived column để sửa lại cột updateddate đúng vào thời điểm update và OLE DB command để update lại các cột updated date và num\_of\_accident = num\_of\_accident + 1.



|  |
| --- |
| **OLAP và MDX** |

1. Câu 1
   1. Phân tích yêu cầu

Tổng hợp số lượng thương vong theo Mức độ nghiêm trọng của thương vong (Casualty Severity) và Các quận địa phương (Local Authority District) trong từng năm.

Sự kiện: Khi có thương vong xảy ra

Bối cảnh sự kiện:

* Ai: Không
* Ở đâu: Các quận địa phương (Local Authority District)
* Cái gì: Mức độ nghiêm trọng của thương vong (Casualty Severity)
* Khi nào: Không

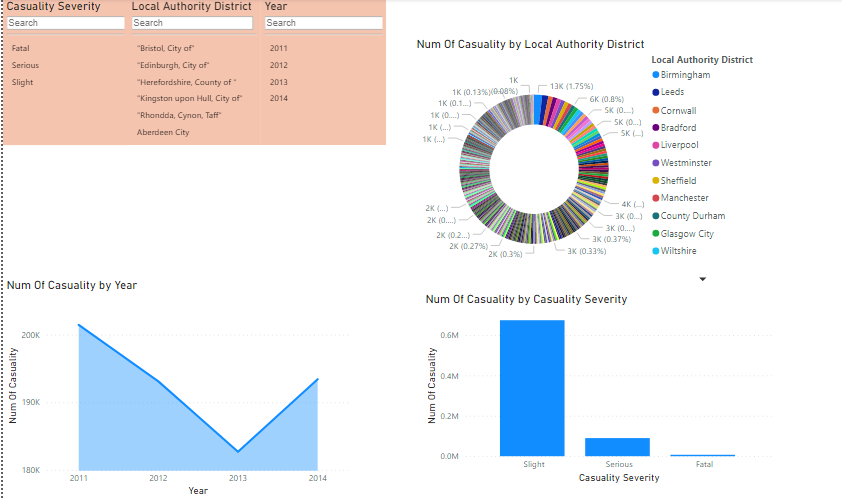
Đo lường: số lượng tai nạn

* 1. Thực hiện MDX

Ý tưởng: Thể hiện trường thời gian ở dạng cột với cột lớn nhất là các năm vì số lượng năm tăng theo thời gian. Cuối cùng là trường các quận địa phương cùng các mức độ nghiêm trọng tương ứng sẽ được thể hiện theo dòng.



* 1. Thực hiện qua Power BI



1. Câu 2
   1. Phân tích yêu cầu

Tổng hợp số lượng thương vong theo Mức độ nghiêm trọng của thương vong (Casualty Severity) và Các quận địa phương (Local Authority District) trong các quý qua từng năm.

Sự kiện: Khi có thương vong xảy ra

Bối cảnh sự kiện:

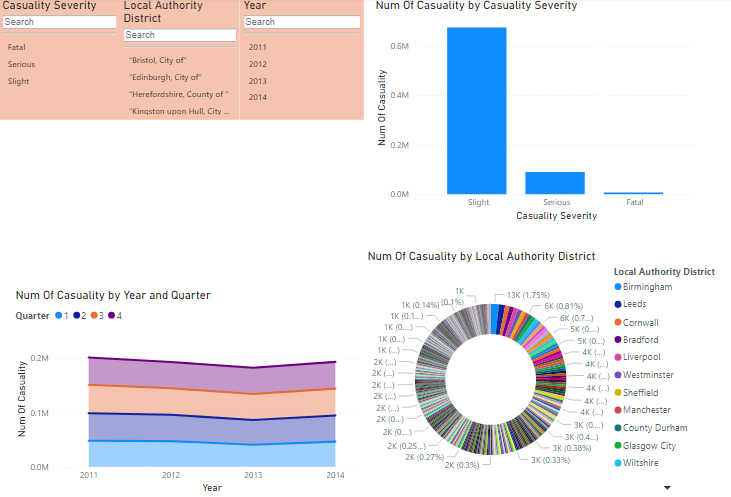
* Ai: Không
* Ở đâu: Các quận địa phương (Local Authority District)
* Cái gì: Mức độ nghiêm trọng của thương vong (Casualty Severity)
* Khi nào: Không

Đo lường: số lượng tai nạn

* 1. Thực hiện MDX

Ý tưởng: Thể hiện trường thời gian ở dạng cột với cột lớn nhất là các quý vì số lượng năm tăng theo thời gian. Cuối cùng là trường các quận địa phương cùng các mức độ nghiêm trọng tương ứng sẽ được thể hiện theo dòng.





1. Câu 3
   1. Phân tích yêu cầu

Tổng hợp số lượng thương vong theo Giới tính(Gender), Loại thương vong(Casualty Type) và Độ tuổi (Age Band) trong từng năm.

Sự kiện: Khi có thương vong xảy ra

Bối cảnh sự kiện:

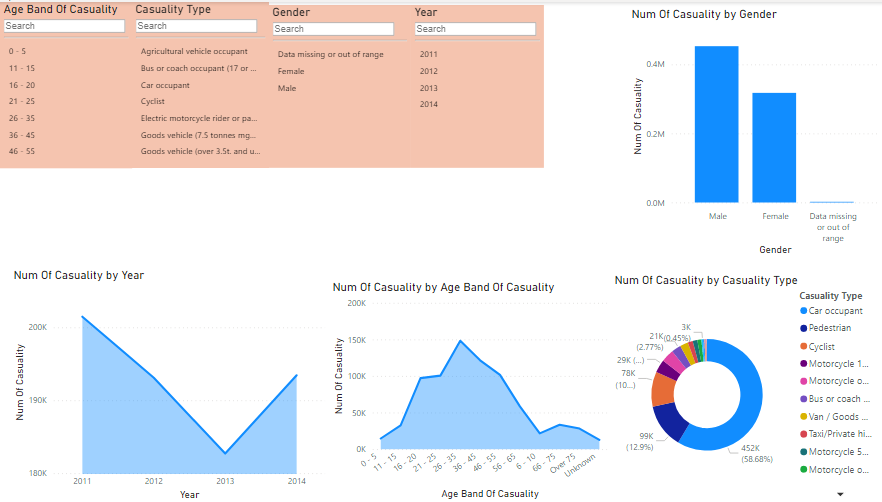
* Ai: Không
* Ở đâu: Các quận địa phương (Local Authority District)
* Cái gì: Mức độ nghiêm trọng của thương vong (Casualty Severity)
* Khi nào: Không

Đo lường: số lượng tai nạn

* 1. Thực hiện MDX

Ý tưởng: Thể hiện trường thời gian ở dạng cột với cột lớn nhất là các năm vì số lượng năm tăng theo thời gian sau đó là độ tuổi. Cuối cùng là trường giới tình cùng các loại nghiêm trọng tương ứng sẽ được thể hiện theo dòng.





1. Câu 4
   1. Phân tích yêu cầu

Thống kê số lượng TNGT theo Mức Độ Nghiêm Trọng và Thời Điểm Trong Ngày (Morning: 5am-12pm, Afternoon: 12pm-5pm, Evening: 5pm-9pm, Night: 9pm-5am) trong các năm.

Sự kiện: Khi có một tai nạn xảy ra

Bối cảnh sự kiện:

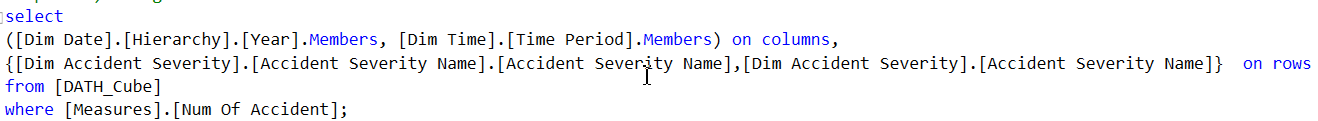
* Ai: Không
* Ở đâu: Không
* Cái gì: Mức độ nghiêm trọng
* Khi nào: Thời điểm trong ngày xảy ra tai nạn

Đo lường: số lượng tai nạn

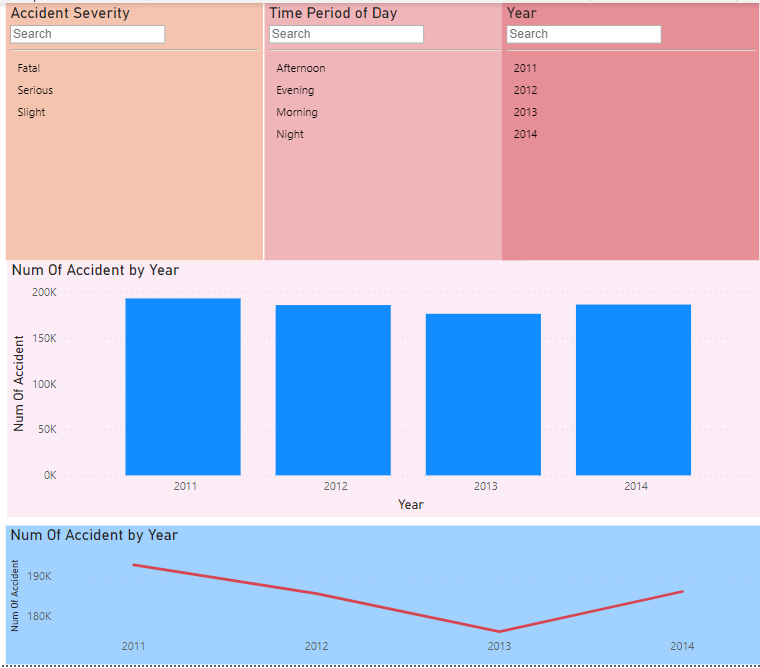
* 1. Thực hiện MDX

Ý tưởng: Thể hiện các trường thời gian ở dạng cột với cột lớn nhất là các năm vì số lượng năm tăng theo thời gian sau đó mới đến các cột thời điểm trong ngày tương ứng từng năm. Cuối cùng là trường mức độ nghiêm trọng sẽ được thể hiện theo dòng có kết hợp thêm để xuất hiện dòng tổng tất cả các mức độ.

Code MDX và kết quả sau khi chạy:



* 1. Thể hiện qua PowerBI



1. Câu 5
   1. Phân tích yêu cầu

Thống kê số lượng TNGT theo Mức Độ Nghiêm Trọng, Vùng (Urban\_or\_Rural\_Area), và Kiểu Đường (Road Type) trong các năm.

Sự kiện: Khi có một tai nạn xảy ra

Bối cảnh sự kiện:

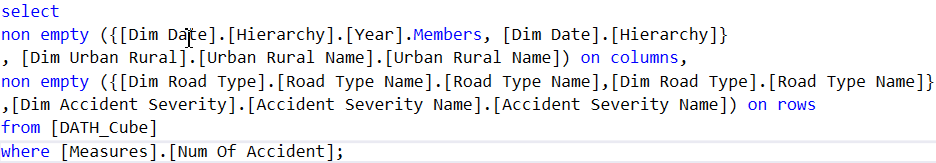
* Ai: Không
* Ở đâu: Vùng (Urban\_or\_Rural\_Area)
* Cái gì: Mức độ nghiêm trọng, kiểu đường (Road Type)
* Khi nào: Năm xảy ra tai nạn

Đo lường: số lượng tai nạn

* 1. Thực hiện MDX

Ý tưởng: Thể hiện trường thời gian ở dạng cột với cột lớn nhất là các năm vì số lượng năm tăng theo thời gian sau đó là các cột thể hiện vùng. Cuối cùng là trường kiểu đường cùng các mức độ nghiêm trọng tương ứng sẽ được thể hiện theo dòng.

Code MDX và kết quả sau khi chạy:



* 1. Thể hiện qua PowerBI



1. Câu 7
   1. Phân tích yêu cầu

Tổng hợp số lượng tai nạn theo Mục Đích Hành Trình (Journey Purpose) và Loại Phương Tiện (Vehicle\_Type).

Sự kiện: Khi có một tai nạn xảy ra

Bối cảnh sự kiện:

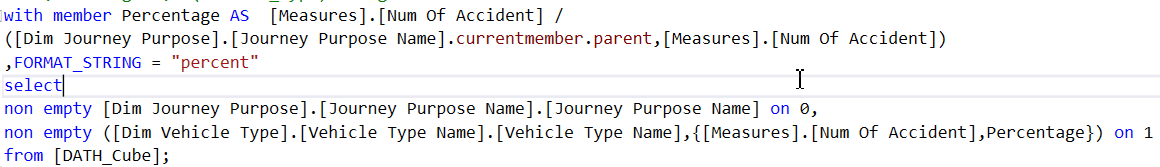
* Ai: Không
* Ở đâu: Không
* Cái gì: Mục đích hành trình (Journey Purpose) và loại phương tiện (Vehicle\_Type)
* Khi nào: Không

Đo lường: số lượng tai nạn

* 1. Thực hiện MDX

Ý tưởng: Định nghĩa một member mới là Percentage để thể hiện % số lượng tai nạn theo mỗi loại Mục đích hành trình. Sử dụng loại phương tiện để thể hiện cho dòng và các dòng con tương ứng sẽ là 2 loại measure gồm số lượng tai nạn, % tai nạn trên tổng các mục đích hành trình → Mục đích hành trình sẽ được thể hiện ở dạng cột.

Code MDX và kết quả sau khi chạy:



* 1. Thể hiện qua PowerBI



1. Câu 9
   1. Phân tích yêu cầu

Thống kê số lượng tai nạn theo Mức Độ Nghiêm Trọng, Loại Phương Tiện (Vehicle Type), Built-up Road trong các năm.

Sự kiện: Khi có một tai nạn xảy ra

Bối cảnh sự kiện:

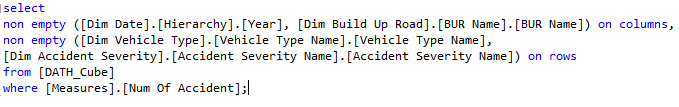
* Ai: Không
* Ở đâu: Không
* Cái gì: Mức độ nghiêm trọng, Loại phương tiện và Built-up Road
* Khi nào: Năm xảy ra tai nạn

Đo lường: số lượng tai nạn

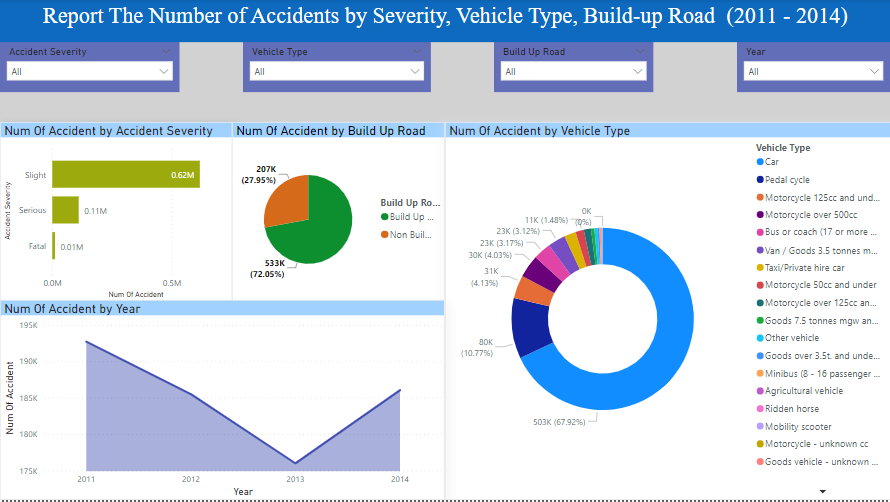
* 1. Thực hiện MDX

Ý tưởng: Thể hiện trường thời gian ở dạng cột với cột lớn nhất là các năm vì số lượng năm tăng theo thời gian sau đó là các cột thể hiện Build Up Road để phân biệt các trường hợp Build Up Road và Non Build Up Road. Các trường loại phương tiện cùng với các mức độ nghiêm trọng tương ứng sẽ được thể hiện theo dòng.

Code MDX và kết quả sau khi chạy:



* 1. Thể hiện qua PowerBI



1. Calculated measure

Tạo thêm 1 calculated measure để tính tỉ lệ mức độ tăng giảm số lượng tai nạn giao thông so với năm trước đó để biết được tình hình tai nạn giao thông qua các năm có cải thiện so với năm trước hay không (tính theo đơn vị phần trăm, giá trị âm nghĩa là số lượng TNGT giảm so với năm trước, số dương nghĩa là số lượng TNGT tăng so với năm trước).

Tiến hành tạo calculated member với Name là [Accident Rate] và chọn Parent hierarchy là Measures vì ta muốn tạo 1 calculated measure.

Sau khi chọn hierarchy, tiến hành điền biểu thức MDX để tính toán, ở đây ta sẽ chia ra 2 trường hợp:

- Trường hợp năm trước đó không có số liệu thì mặc định sẽ là 0.

- Ngược lại sẽ tính theo công thức lấy số lượng TNGT năm muốn tính toán trừ đi số lượng TNGT năm liền trước nó, sau đó chia kết quả cho số lượng TNGT năm trước đó.

Case

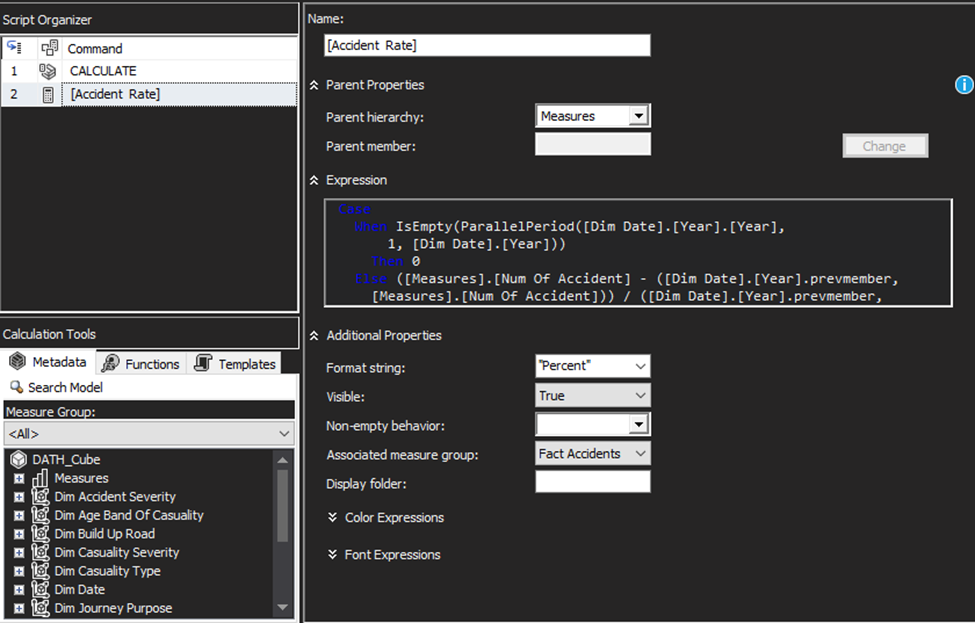
When IsEmpty(ParallelPeriod([Dim Date].[Year].[Year], 1, [Dim Date].[Year]))

Then 0

Else ([Measures].[Num Of Accident] - ([Dim Date].[Year].prevmember,[Measures].[Num Of Accident])) / ([Dim Date].[Year].prevmember,[Measures].[Num Of Accident])

End

Sau khi viết biểu thức MDX xong, ta sẽ tiến hành chọn thêm một số thuộc tính cho biểu thức. Ở đây, ta sẽ chọn Format string là Percent để hiển thị theo phần trăm và chọn Associated measure group là Fact Accidents.



1. Câu 13

Dùng region map để biểu diễn số lượng TNGT ở các vùng trong năm.

Nhóm sử dụng công cụ powerBI để vẽ region map.

Viết câu query để lấy ra những thông tin cần thiết trong 3 bảng sau

Postcode, Fact\_accidents và DimDate.

Postcode để lấy thông tin vùng

Fact\_accidents để lấy số lượng ca tai nạn ứng với từng vùng và DimDate để lấy năm tương ứng

Text

Description automatically generated

Sau đó lấy data ra file csv để thêm vào powerBI và dùng công cụ filter map để thực hiện vẽ. Ta được những bản đồ sau:

Map

Description automatically generated

Map

Description automatically generated

Có 3 màu tương ứng với 3 vùng là England, Scotland và Wales.

Ta có một số nhận xét sau:

* Những vùng màu càng đậm thì càng có nhiều tai nạn và ta thấy đa số tập trung ở những thành phố lớn
* England là khu vực có mật độ tai nạn cao nhất

Map

Description automatically generated

* Khi đưa chuột vào thì ta thấy hiện ra các chỉ số về thành phố, khu vực và số lượng tai nạn

|  |
| --- |
| **KPI và Action** |

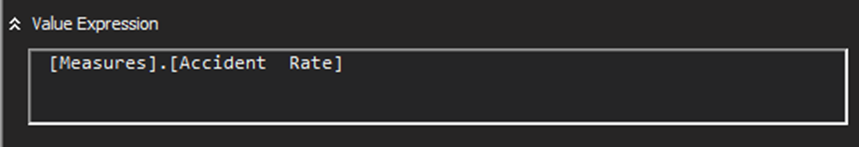
Mục tiêu: Tạo KPI để đánh giá mức độ tăng giảm số lượng TNGT so với năm trước đó qua các năm.

Thực hiện:

Đầu tiên ta cấu hình KPI với Name là Accident Change Rate và **Associated measure group** là Fact Accidents. Sau đó ta sẽ thêm các biểu thức cần thiết cho KPI.

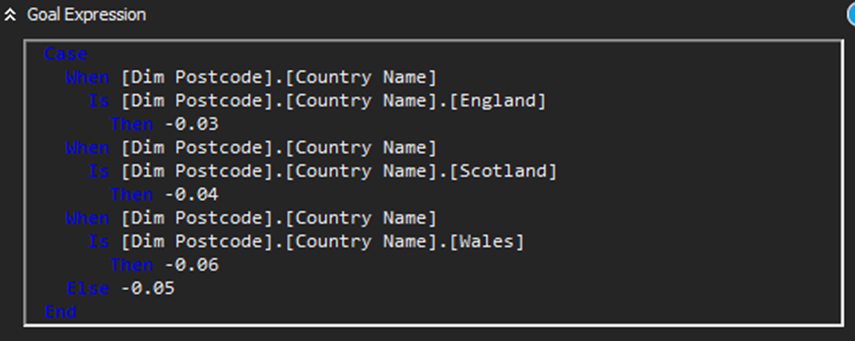
**Value Expression**

Ở đây ta sử dụng calculated measure [Accident Rate] vừa tạo ở trên để làm giá trị cần đánh giá.



**Goal Expression**

Mục tiêu sẽ cần giảm tỉ lệ gia tăng số lượng TNGT qua các năm. Ở đây, nhóm sẽ chia từng nước với các goal khác nhau. Ví dụ ở England sẽ đặt mục tiêu giảm 3% số lượng TNGT so với năm trước, Scotland đặt mục tiêu sẽ giảm 4% , Wales là 6% và các nước còn lại là 5%.



**Status Expression**

Đánh giá trạng thái hiện tại của KPI so với goal đã đặt ra. Ở đây, ta lấy KPI value chia cho KPI Goal, nếu đạt từ 0.9 trở lên thì sẽ trả về giá trị 1 (trạng thái tốt). Trường hợp KPI Value chia cho KPI Goal nằm trong khoảng 0.7 đến 0.9 hoặc KPI Value bằng 0 thì sẽ trả về giá trị 0 (trạng thái trung bình). Ngược lại sẽ trả về giá trị -1 (trạng thái chưa đạt).



**Trend Expression**

So sánh KPI so với các năm trước, ở đây nhóm sẽ so sánh chỉ số KPI so với 1 năm trước đó. Nếu trước đó không có năm nào thì sẽ trả về 0. Trường hợp [Accident Rate] lớn hơn năm trước thì sẽ trả về 1 (xu hướng tăng). Trường hợp [Accident Rate] bằng năm trước đó thì sẽ trả về 0 (không tăng không giảm). Ngược lại, sẽ trả về -1 (xu hướng giảm).



Kết quả khi chạy KPI Accident Change Rate (nước England)



|  |
| --- |
| **Data mining** |

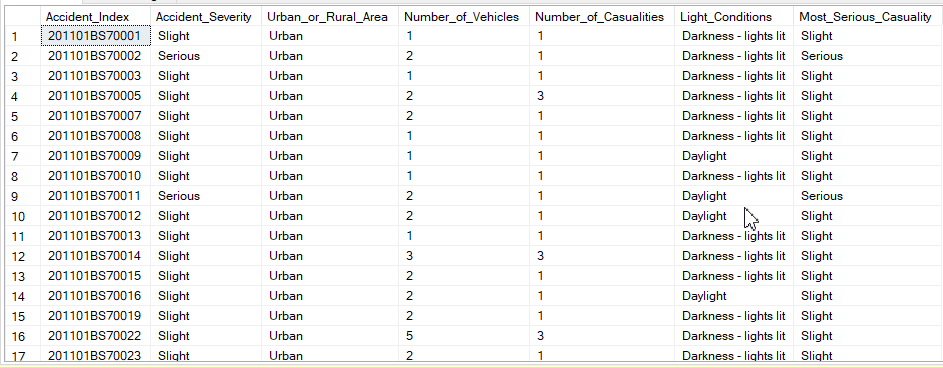
1. Phát biểu bài toán và giải thuật

Nhóm sẽ thực hiện dự đoán mức độ nghiêm trọng của tai nạn dựa trên một số trường cần thiết như Light\_Conditions (tình trạng đèn đường), Urban\_Rural (nơi xảy ra là thành thị hay nông thôn), Num\_of\_vehicles (số lượng phương tiện liên quan), Num\_of\_casualities (Số nạn nhân) và Accident\_Severity (mức độ nghiêm trọng).

Nhóm sẽ sử dụng thuật toán Cây quyết định (Decision Tree) để thực hiện bài toán này.

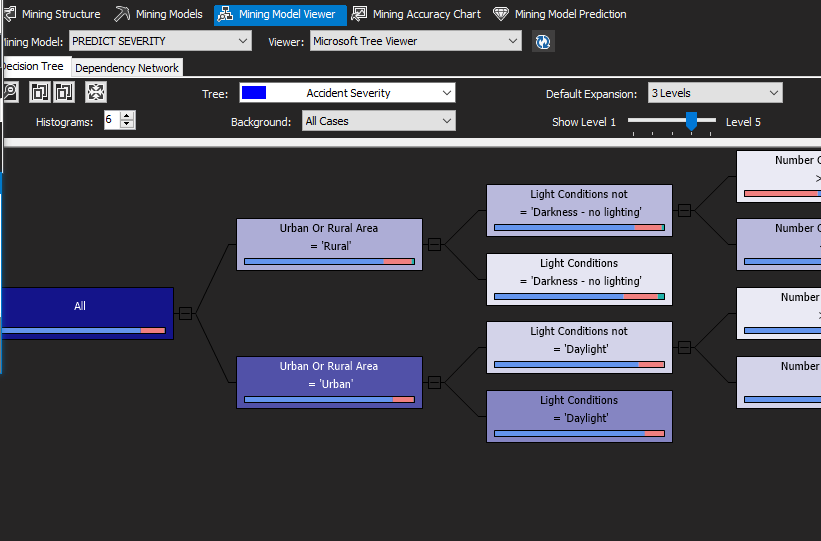
1. Thực hiện

Đầu tiên, nhóm sử dụng dữ liệu từ NDS để trích xuất ra các thuộc tính cần thiết.



Tiếp theo nhóm sử dụng VisualStudio để lấy dữ liệu từ bảng trích xuất trên và đưa vào model DecisionTree với các cột input bao gồm Light\_Conditions, Urban\_Rural, Num\_of\_vehicles, Num\_of\_casualities và cột cần predict là Accident\_Severity. Ngoài ra, nhóm sử dụng tỉ lệ tập train và test là 80:20.

Kết quả đạt được là một cây quyết định gồm 5 level:



Thông qua biểu đồ LiftChart ta có thể thấy đường biểu thị cho thuật toán cây quyết định (màu cam) khá gần với đường thuật toán tốt nhất (màu xanh). Ngoài ra, độ chính xác của thuật toán lên đến 87.5%.

Bảng thể hiện độ quan trọng và liên hệ giữa các thuộc tính độc lập vào biến phụ thuộc:

