Using PROV-O to Represent Lineage in Statistical Processes: A Record Linkage Example*

Franck Cotton[1], Guillaume Duffes[2] and Flavio Rizzolo[3]

[1] Institut national de la statistique et des études économiques, France
franck.cotton@insee.fr

[2] Institut national de la statistique et des études économiques, France
guillaume.duffes@insee.fr

[3] Statistics Canada, Canada
flavio.rizzolo@statcan.gc.ca

Abstract. Provenance and lineage, i.e. information about the data origins and its evolution throughout its life-cycle, are becoming more critical in a world of official statistics, which is under growing pressure to be more open and transparent, and whose statistical outputs are increasingly based on admin data and other sources of varying quality. We focus here on provenance and lineage requirements in the context of record linkage, i.e. the activity of finding and linking records from different sources that refer to the same entity, which provides one of the most complex lineage use cases in statistical production. We define a generic lineage model with different levels of granularity and explore the use of PROV-O to capture its metadata in a machine-actionable standard.

Keywords: Provenance, Lineage, Record Linkage, PROV.

1 Introduction

Statistical offices need to find, acquire and integrate data from both traditional and new data sources at an ever-increasing speed. In a world of fake news and alternate facts, the demand for trusted data has never been higher. To be able to assess the quality of statistical outputs, it is essential to understand the data lifecycle across the entire statistical production (best described by the Generic Statistical Business Process Model [1]).

Provenance and lineage can be information on processes, methods and data sources and their relationships to data outputs. They provide the complete traceability of where data has resided and what actions have been performed on the data over the course of its life. With the advent of Big Data, cloud platforms and IoT, statistical offices have started to use more distributed data processing approaches, which makes provenance and lineage more critical than ever before.

The PROV data model [2] is a well-established model for provenance metadata and is mapped to RDF through the PROV ontology [3]. Since more and more statistical offices use RDF for metadata, it is valuable to investigate the use of PROV-O for capturing lineage and provenance metadata about the statistical processes. To this end we apply PROV to describe metadata lineage about an activity at the core of statistical production: record linkage. Record linkage depends on several capabilities with complex lineage requirements, e.g. data cleansing, standardization, coding, integration, entity matching, etc. which need to record traceability at multiple levels of granularity, e.g. data set, variable, record, etc. In this paper, we explore the use of PROV-O for the representation of lineage metadata needed to document a record linkage process.

2 Data Matching and Record Linkage

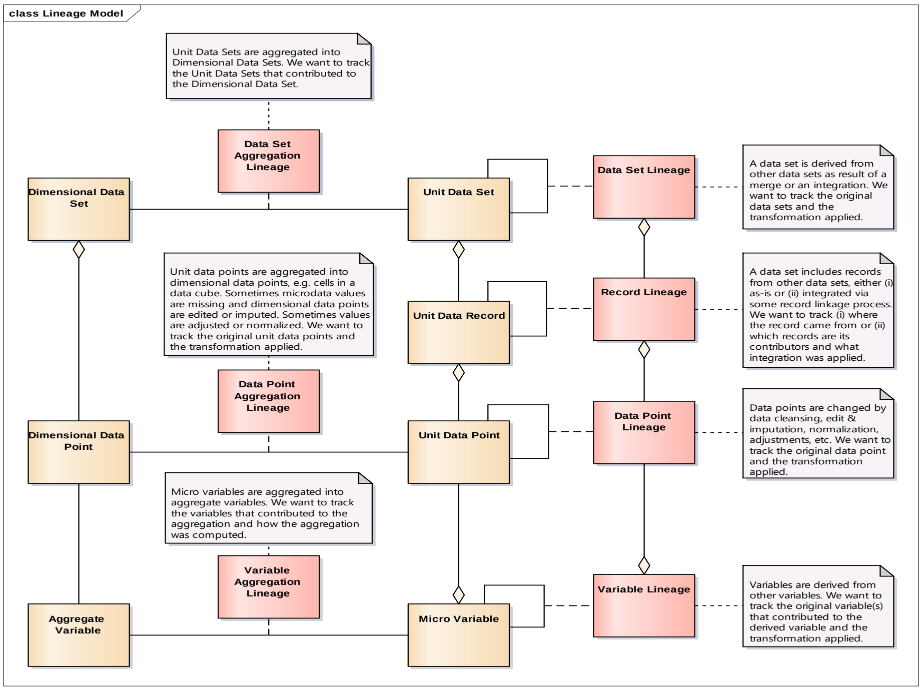
Data matching, or entity resolution, is the identification, matching and linking of different instances/representations of the same real-world entity [4]. Entities of interest might be people, families, businesses, tax payers, products, countries, etc. In any complex environment in which information integration of heterogeneous sources plays a crucial role, entities could multiply very rapidly as a result of differences in representations, e.g. same business with different spelling in name, address, etc., entirely different ids, or differences in states/versions/variants, e.g. preliminary, final, raw, imputed, v2010, v2012, etc. Entity resolution applies to all types of data integration scenarios and has received special attention in the statistical production domain for resolving statistical units and linking their records, which is known as record linkage [5].

Deciding whether records match or not is often computationally expensive and domain specific. One way of dealing with this problem is to have entirely different, specialized approaches for each entity domain, i.e. people, businesses, data sets, metadata, etc. This is the approach traditionally followed in record linkage. Another approach is to divide the problem into a generic part (for iterating over the data set, deal with constraint and merge/link records) and a set of domain-specific similarity functions (to determine whether two records refer to the same entity) [6]. The latter approach allows for a separation of concerns between the data management algorithms and the actual pair-wise domain-specific resolution functions. Both approaches are used extensively depending on the application.

2.1 Lineage requirements for Record Linkage

Linking two records about the same entity brings about a number of provenance and lineage requirements at different levels, e.g. between the record themselves, the variables from different dataset that are used for linking, the data sets produced at different stages of the record linkage process, etc. These different types of lineage are to be maintained for audit and reproducibility purposes. The description of the linkage process should include lineage metadata as specified by the model in Figure 1.

We will use here the Record Linkage Project Process Model [7] as a reference to describe the types of lineage required for record linkage. For an in-depth presentation of all aspects of record linkage, please refer to [4] and [5].



Unit Data Sets are aggregated into Dimensional Data Sets. We want to track the Unit Data Sets that contributed to the Dimensional Data Set.

Unit data points are aggregated into dimensional data points, e.g. cells in a data cube. Sometimes microdata values are missing and dimensional data points are edited or imputed. Sometimes values are adjusted or normalized. We want to track the original unit data points and the transformation applied.

Micro variables are aggregated into aggregate variables. We want to track the variables that contributed to the aggregation and how the aggregation was computed.

A data set is derived from other data sets as result of a merge or an integration. We want to track the original data sets and the transformation applied.

A data set includes records from other data sets, either (i) as-is or (ii) integrated via some record linkage process. We want to track (i) where the record came from or (ii) which records are its contributors and what integration was applied.

Data points are changed by data cleansing, edit & imputation, normalization, adjustments, etc. We want to track the original data point and the transformation applied.

Variables are derived from other variables. We want to track the original variable(s) that contributed to the derived variable and the transformation applied.

Fig. 1. Lineage Model (Statistics Canada)

The first preliminary step involving data manipulation with lineage requirements is *4.1 Standardize linkage variables*. In this step, the source data sets are subset for privacy protection to contain only the linkage variables. Variables commonly kept include names, demographic variables (e.g. sex, date of birth), geographic variables (e.g. postal codes) and unique identifiers (e.g. social or health insurance numbers, business numbers). Values contained with the linkage variables may be combined or concatenated to create linkage keys. Structure, format (e.g. data types) and codesets are standardized to facilitate comparability and integration. The resulting data sets are the standardized data sets.

This step requires *data set lineage* between the source data sets and their respective standardized data sets, as well as *variable lineage* between the linkage variables. The latter will include information about the changes in formats and codesets in the cases where the variables were transformed in the process of creating the standardized data sets. The step is illustrated in Figure 2.

Metadata required: standardized linkage variables, standard formats and codesets.

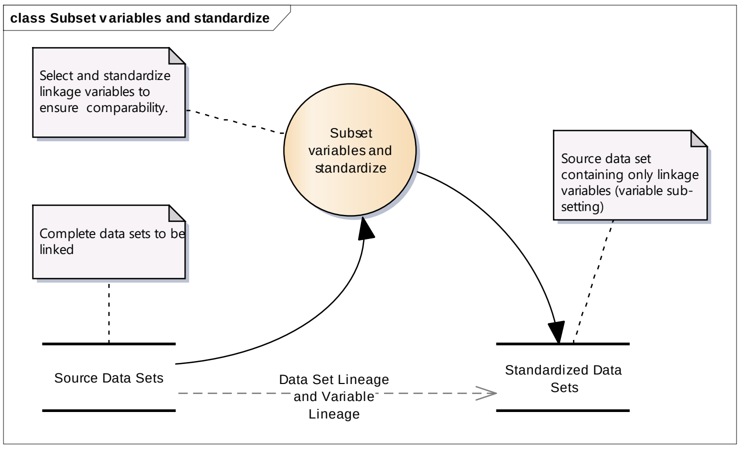

class Subset variables and standardize
Select and standardize linkage variables to ensure comparability.
Subset variables and standardize
Source data set containing only linkage variables (variable sub-setting)
Complete data sets to be linked
Source Data Sets
Data Set Lineage and Variable Lineage
Standardized Data Sets

Fig. 2. Subset source data sets to standardized linkage variables

The next step is *4.3 Identify in-scope records for linkage*. This step defines inclusion and exclusion criteria to identify records from the standardized data sets that are eligible for record linkage – records with incomplete or missing values may be considered ineligible for linkage. In all cases, counts of eligible and ineligible records are produced for each standardized data set. In some cases, a new collection of linkage-ready data sets containing only eligible records is produced, which requires *data set lineage* between them and the standardized data sets with all records. The step is illustrated in Figure 3.

Metadata required: inclusion and exclusion criteria, counts of eligible and ineligible records.

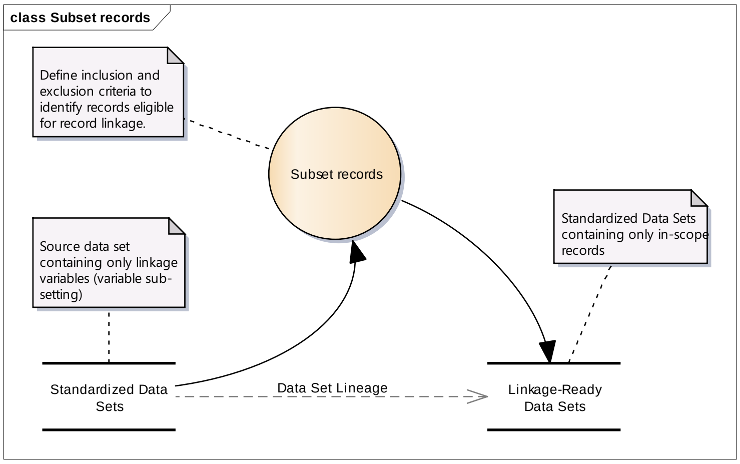

class Subset records
Define inclusion and exclusion criteria to identify records eligible for record linkage.
Subset records
Standardized Data Sets containing only in-scope records
Source data set containing only linkage variables (variable sub-setting)
Standardized Data Sets
Data Set Lineage
Linkage-Ready Data Sets

Fig. 3. Subset standardized data sets to in-scope records

From the linkage-ready data sets containing either all or only eligible records, the actual record linkage process starts. The first linkage step is *5.1 Identify potential pairs*. This could be done by generating the cross product of the linkage-ready data sets, i.e. compare all possible pairs of records, or by blocking, i.e. partitioning the linkage-ready data sets into mutually exclusive and exhaustive blocks of records to reduce the number of pairs to be compared, since comparison is restricted to pairs of records only within a block.

The goal of blocking is to reduce the number of pairs to compare by removing pairs that are unlikely to produce true matches. Blocking keys are used to define the blocks of similar pairs. For instance, a common blocking key for person units is postal code, which is based on the assumption that two entities that have similar postal code are more likely to actually be the same person. This comparison is based on some similarity function to be applied to the blocking keys.

Computing the full cross-product, let alone creating a cross-product data set, is often unfeasible, but computing the cross-product within blocks is not. This requires both *data set lineage* from the linkage-ready data sets to each block, and also *record lineage* from each pair of linkage-ready data set records to each resulting block record. The step is illustrated in Figure 4.

Metadata required: blocking keys, similarity function.

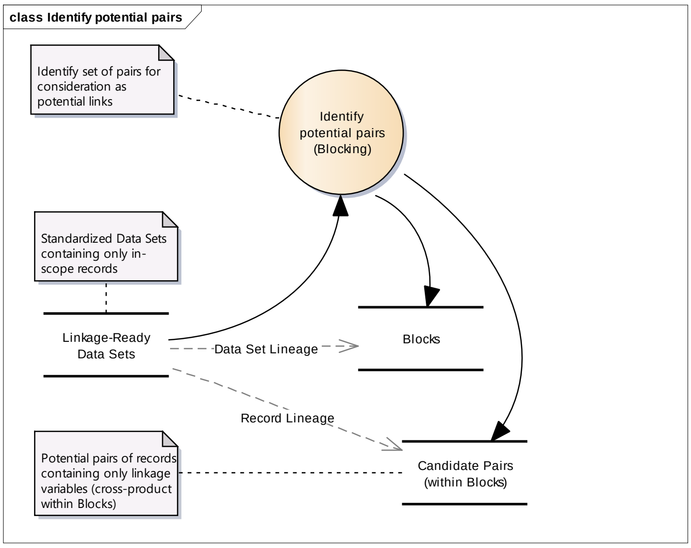

class Identify potential pairs
Identify set of pairs for consideration as potential links
Identify potential pairs (Blocking)
Standardized Data Sets containing only in-scope records
Linkage-Ready Data Sets
Data Set Lineage
Blocks
Record Lineage
Potential pairs of records containing only linkage variables (cross-product within Blocks)
Candidate Pairs (within Blocks)

Fig. 4. Identify potential pairs of eligible records

Once the pairs of records are established, the candidate record pairs in the blocks need to be fully compared to determine their overall similarity. This similarity is calculated by comparing several variables beyond the blocking variables used to define the blocks. For instance, if the blocking was based on postal code, then street name and last name could be used now to compare pairs of records within the blocks.

Record comparison could be done deterministically or probabilistically. In deterministic record linkage, a pair of records is said to be a match if they agree on each element within a collection of variables called the *match key*. For example, if the match key consists of last name, street number and name, and year of birth, a comparison function could define a match to occur only when names agree on all characters, the years of birth are the same, and the street numbers are identical.

Under the Fellegi–Sunter model for probabilistic linkage, pairs of records are classified as *match*, *possible match*, or *no match*. This is determined by comparing the likelihood ratio of having a match, namely R, against two thresholds called Upper and Lower. If R is greater than or equal to Upper, then the pair is considered a match. If R is smaller than or equal to Lower, then the pair is considered a no match. If R is somewhere between Lower and Upper then the pair is considered a possible match and is submitted to manual review for a final decision.

Deterministic and probabilistic comparisons and validation are done *in 5.2 Field and record comparison*, *5.3 Linkage rules*, and *6 Assess quality*. Record comparison relies on field comparison, which is done using comparison functions or linkage rules.

The result of this iterative process is a set of *linkage keys*. A linkage key is a code created to identify the records related to the same entity in the source data sets. As such, it does not contain any identifying information that may have been used to create the links. Lineage involved in these steps include a combination of *record lineage* and *variable lineage*, since the block records that match need to be traced back to their respective blocks and are assigned a linkage key, which functions as a new variable. This step is illustrated in Figure 5.

Metadata required:

- *Deterministic record linkage*: comparison function, match key.
- *Probabilistic record linkage*: likelihood ratio R, Lower, Upper, field agreement weights (per linkage variable), agreement level weights (per record pair), record similarity, agreement level (match, possible match, no match).
- *Both*: linkage keys.

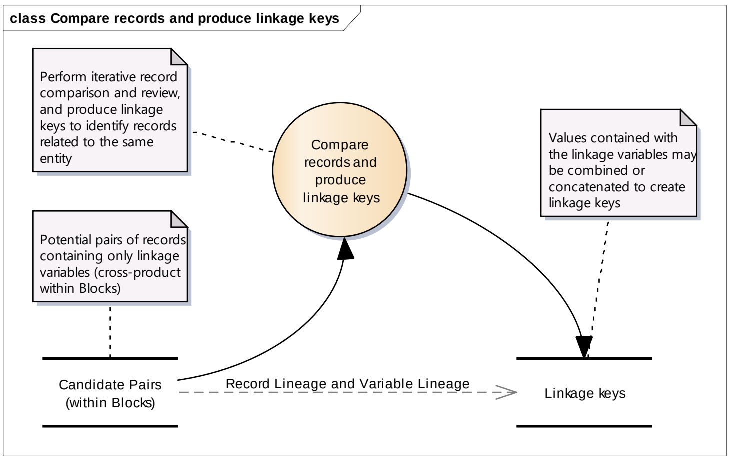

class Compare records and produce linkage keys
Perform iterative record comparison and review, and produce linkage keys to identify records related to the same entity
Compare records and produce linkage keys
Values contained with the linkage variables may be combined or concatenated to create linkage keys
Potential pairs of records containing only linkage variables (cross-product within Blocks)
Candidate Pairs (within Blocks)
Record Lineage and Variable Lineage
Linkage keys

Fig. 5. Compare records to find matches and produce linkage keys

3 Using PROV-O

3.1 Vocabularies used

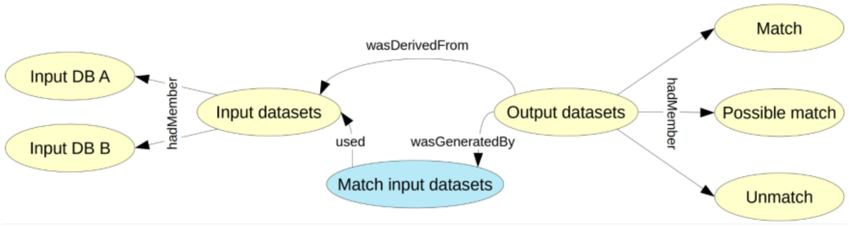
As the title of the paper suggests, we will be using the PROV vocabulary [3] to represent the different entities and activities that we identified in the record linkage process. PROV is expected to have a great utility for the representation of statistical processes at various levels of details, and it is actually one of the pillars of an ongoing work recently launched by the standardization activity of the UNECE ModernStats initiative[1], which aims at building a Core Ontology for the Official Statistics community[2].

Apart from PROV, we use in the following examples a few well-known vocabularies with their usual namespaces and prefixes, as well as elements from the Data Quality Management Vocabulary [8] in the detailed models.

3.2 A very simple case

We start in this section from a simplified model of record linkage. Basically, we match two datasets A and B and the record linkage produces three outputs: the set of records that are matched, the set of records that are not matched, and the set of records that could be matches but need additional verification (through human review for example). Modelling this simplified view with PROV will allow us to grasp the principles of the modelization and how to attach lineage metadata to the process.

In PROV-O terms, we can represent the inputs and outputs of the record linkage operation as instances of `prov:Collection`, composed of instances of `coos:StatisticalDataset`. In a simple PROV representation, a “Match input datasets” activity uses the input collection and generates the output collection. This output collection is linked to the input collection by a `prov:wasDerivedFrom` property. This simple model is represented in the following figure.



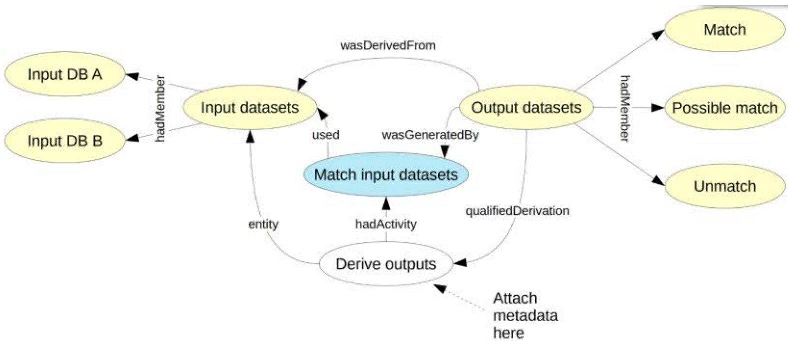
In the Statistics Canada model (Figure 1), the link between input and output data sets should be described at high-level with rich *data set lineage* metadata. For now, we only have a `prov:wasDerivedFrom` RDF predicate, which is clearly not fit for purpose since RDF predicates cannot bear additional information. We thus have to use the qualification mechanism described in the PROV-O specification. According to this mechanism, the `prov:wasDerivedFrom` property can be qualified as a

[1] https://statswiki.unece.org/display/hlgbas

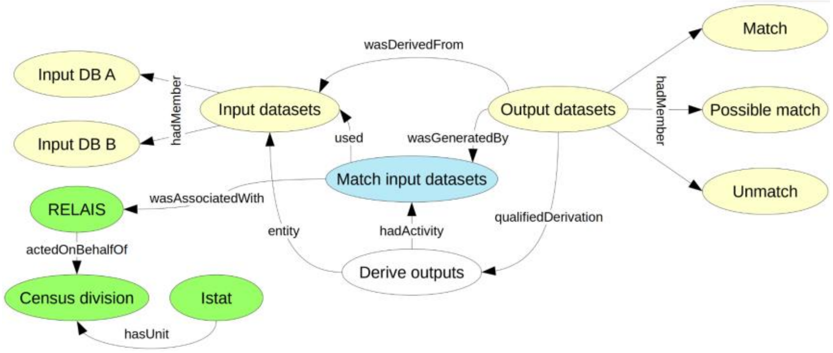
[2] https://github.com/linked-statistics/COOS/

prov:qualifiedDerivation property pointing from the output entity to a prov:Derivation instance.

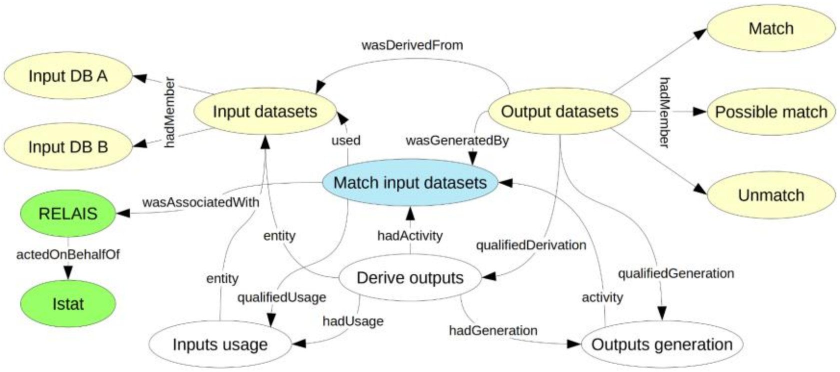
It is recommended in PROV to define sub-classes of prov:Derivation for specific use cases. Here, referring to the Statistics Canada model, we can define the coos:DataSetDerivation class (note that the placement in the COOS namespace is provisional). An instance of this class, “Derive outputs” is then created, connecting the output dataset to the inputs and to the activity. More lineage metadata can also be attached to the “Derive outputs” derivation. The corresponding model is given in the following figure.



We can enrich the model by adding information on the agents involved in the matching operation. We will keep the model simple and just suppose that the matching is made by Istat, the Italian statistical organization, using the RELAIS software [9]. The software will be represented as an instance of the prov:SoftwareAgent class, and Istat by an instance of the coos:Organization (which is a sub-class of prov:Organization). The matching activity is then connected to the RELAIS resource by a prov:wasAssociatedWith property.



It should be noted that the qualification mechanism can be applied to all the properties of the basic model, which allows to add information on how the inputs were used by the activity or how the outputs were generated. The fully-qualified model is given in the following figure.



4 Digging deeper

For some use cases, the high-level view described in the previous section is sufficient, but if we want to describe more precisely the statistical process it is necessary to break it down into more fine-grained steps. In this section, we will use PROV again to describe the four diagrams introduced in the first part of this paper. In addition, we will include statements conforming to the Data Quality Management Vocabulary [8]. We will call the four steps “Class 1” (Figure 2) to “Class 4” (Figure 5) in order to make unambiguous references to the descriptions made above.

Each step has been modelled as a RDF/Turtle fragment, and the four fragment files can be found on GitHub[3]. For the sake of brevity, we will only deal with fragments 2 (Figure 3) and 4 (Figure 5) below.

4.1 From standardized data sets to linkage-ready data sets

The following Turtle fragment and the picture below correspond to the “Class 2” (Figure 3) step.

```
@prefix reclink: <http://example.org/recordlinkage/> .
@prefix prov: <http://www.w3.org/ns/prov#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dqm: <http://purl.org/dqm-vocabulary/v1/dqm#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

reclink:standardizedDatasets a prov:Entity ;
        rdfs:label "Standardized datasets"@en .

reclink:linkageReadyDatasets a prov:Entity ;
        rdfs:label "Linkage-ready datasets"@en ;
        prov:wasDerivedFrom reclink:standardizedDatasets ;
        prov:qualifiedDerivation [
             a prov:Derivation ;
             prov:entity reclink:standardizedDatasets ;
             prov:hadActivity reclink:subsetInScopeRecords ;
        ] .
```

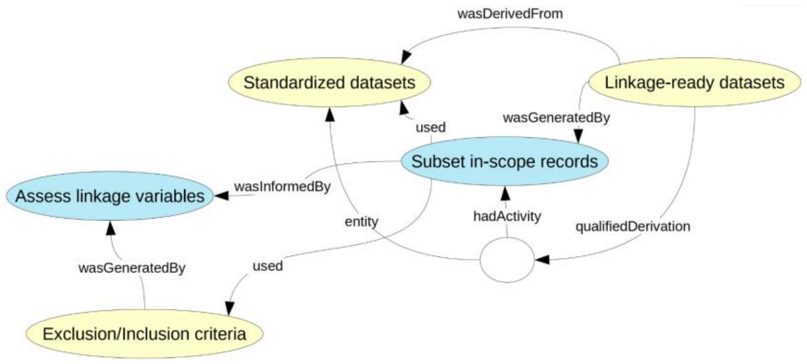
[3] https://github.com/FranckCo/PROV-Process/blob/master/src/main/resources/data

```
reclink:subsetInScopeRecords a prov:Activity ;
        rdfs:label "Identify in-scope records"@en ;
        prov:wasInformedBy reclink:assessLinkageVariable .

reclink:assessLinkageVariable a prov:Activity ;
        rdfs:label "Assess linkage variables"@en .

reclink:inclusionExclusionCriteria a prov:Entity , dqm:MatchingValueRule ;
        dqm:assessment "true"^^xsd:boolean.
```

The `prov:Entity` named `reclink:standardizedDatasets` is a (collection of) subsets from the source datasets that only include the linkage variables, referred to as "Standardized Data Sets" in the "Class 1" (Figure 2) diagram. The `reclink:assessLinkageVariables` activity evaluates the quality and discriminatory power of the linkage variables expected to impact both the quality of the linkage and ultimately the use of the matched data. This activity also generates the `reclink:inclusionExclusionCriteria` entity which establishes inclusion and exclusion criteria to identify records from the source data sets that are eligible for record linkage. These criteria are a specific form of multi property requirements in which external data are used to identify data requirements violations in an instance, which makes them instances of `dqm:MatchingValueRule`.



4.2 From candidate pairs to linked keys

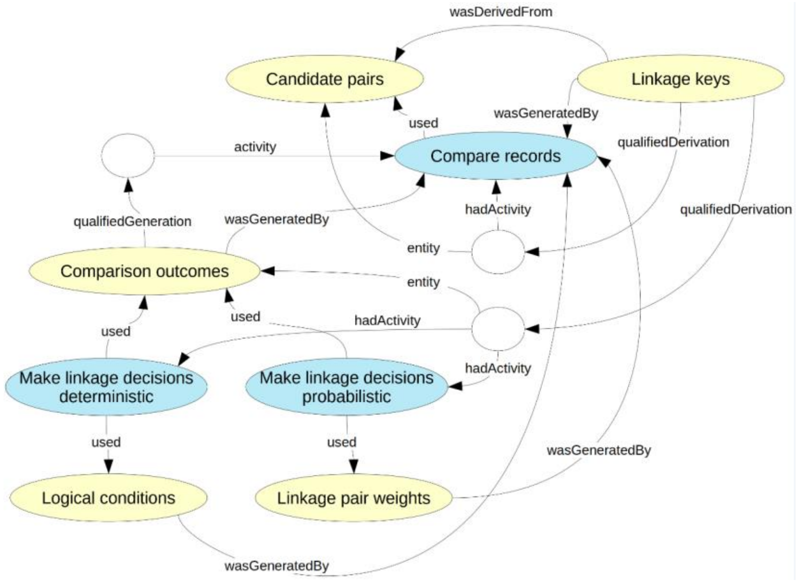
This last part of the record linkage operation corresponds to the "Class 4" (Figure 5) step above. The Turtle code of this fragment is too long to be included below but can be found in Github[4]. We see in this fragment that the `reclink:blocks` PROV entity, i.e. Blocks in the "Class 3" (Figure 4) diagram, is derived from `reclink:linkageReadyDatasets`, i.e. Linkage-Ready Data Sets in the "Class 2" diagram (Figure 3), through the activity `reclink:identifyPotentialPairs`, i.e. Identify potential pairs in the "Class 3" diagram (Figure 4). The `reclink:compareRecords`, i.e. Compare records in the "Class 4" diagram (Figure 5), involves the comparison of strings or numeric representation of the paired records. The entity

[4] https://github.com/FranckCo/PROV-Process/blob/master/src/main/resources/data/fragment-class-4.ttl

`reclink:comparisonOutcomes` generated by the `reclink:compareRecords` activity, is used to make a linkage decision to determine whether each record pair is a match or a non-match: either through a probabilistic method with the `reclink:makeLinkageDecisionsProbabilistic` activity or a deterministic one with the `reclink:makeLinkageDecisionsDeterministic` activity, i.e. Produce linkage keys in “Class 4” diagram (Figure 5).

In a deterministic linkage, this decision is based on a sequence of logical conditions (i.e. the entities `reclink:logicalConditions`) following a rule-based approach. The logical conditions state roughly that values of a given tested property must always obtain a certain state under the condition that values of another property obtain a certain state (condition). Therefore they can also be assimilated to instances of `dqm:ConditionRule`. In a probabilistic linkage, the entity `reclink:linkagePairsWeight` is calculated for each record pair and compared to two thresholds in order to make a decision. These weights are computed and somehow used as data quality scores, which makes them instances of `dqm:DataQualityScore`.

The final output is a linkage key file, i.e. Linkage keys in the “Class 4” diagram (Figure 5): the `reclink:linkageKeys` is an anonymized file that contains only the unique identifiers necessary for identifying the records related to the same entity in the source data sets (`reclink:sourceDatasets`).



5 Conclusion

We showed in this paper that PROV-O can be used to represent lineage of statistical processes at different levels of detail, from the high-level view where a complex operation like record linkage is considered as a single activity, to much more fine-grained representations. An aspect to be addressed in future work (but which is quite easy to grasp conceptually) is how a dataset can be viewed as a collection of records and variables that can be modelled as PROV entities, so the operations described in this paper at the dataset level can be directly extended to the record or variable (or block) levels.

The precise nature and content of the lineage metadata itself, e.g. similarity functions, codesets, likelihood ratios, weights, in the record linkage example, can also be represented in RDF to be attached to PROV constructs. One solution to do that could be to use a flexible mechanism, like the one described in the SDMX information model [10]. An ongoing initiative [11] is currently attempting to express this part of the SDMX model in OWL, which would allow to express in RDF any SDMX Metadata Structure Definition and Metadata Set.

Metadata is always more powerful when used in an active way to actually implement (and not only document) the statistical process: this is the paradigm of "active metadata" which has been developed in the statistical community in recent years. We can actually change perspectives and use provenance metadata to specify the process, with automatic process generation from this formal specification: that is where machine-actionability becomes really interesting.

References

1. The Generic Statistical Business Process Model (GSBPM), https://statswiki.unece.org/display/GSBPM/GSBPM+v5.1, Version 5.1, January 2019.
2. PROV-DM: The PROV Data Model, https://www.w3.org/TR/prov-dm/,W3C Recommendation 30 April 2013.
3. PROV-O: The PROV Ontology, https://www.w3.org/TR/prov-o/, W3C Recommendation 30 April 2013.
4. Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer 2012. ISBN 978-3-642-31163-5.
5. Herzog, T. N., Scheuren, F. J., Winkler, W. E.: Data quality and record linkage techniques. Springer 2007, ISBN 978-0-387-69502-0, pp. I-XIII, 1-227
6. Whang, S. E., Garcia-Molina, H.: Developments in Generic Entity Resolution, IEEE Data Engineering Bulletin, vol. 34, no. 3, pp. 51-59, Sept. 2011.
7. Sanmartin, C., Trudeau, R.: The Record Linkage Project Process Model. In: UNECE Workshop on Implementing Standards for Statistical Modernisation, Sept 2016.
8. The Data Quality Management Vocabulary Specification, http://semwebquality.org/dqm-vocabulary/v1/dqm, V 1.00, Release 09-10-2011.
9. RELAIS (REcord Linkage At IStat), https://www.istat.it/en/methods-and-tools/methods-and-it-tools/process/processing-tools/relais, Last edit: 20 March 2018.
10. SDMX Information Model: UML Conceptual Design, version 2.1, Chapter 7: Metadata Structure Definition and Metadata Set, p.75-91, July 2011.
11. SDMX Metadata: An RDF vocabulary for representing the SDMX metadata model, https://linked-statistics.github.io/SDMX-Metadata/sdmx-metadata.html, W3C Draft Community Group Report 09 April 2017.