Prodotti tradizionali italiani

RACCOLTA DATI

Questo elaborato è stato realizzato per aggregare e fornire informazioni, attraverso elaborazione di dati aperti, su quei prodotti italiani che sono riconosciuti come ‘prodotti tradizionali italiani’, ovvero prodotti che fanno parte della tradizione di una regione, e ne fanno parte poichè le variabili climatiche, territoriali, ambientali, storiche e culturali si sono combinate in modo che potessero prevalere questi prodotti rispetto ad altri. I dati sono stati ricercati inizialmente nel sito http://dati.gov.it/ che è un portale promosso dal governo italiano che mette a disposizione dati rilasciati in formato aperto dalle pubbliche amministrazioni. I dataset trovati in questo sito e compatibili col nostro scopo sono i seguenti:

- https://www.dati.gov.it/dataset/prodotti-agroalimentari-tradizionali (questo dataset contiene informazioni sui prodotti tradizionali umbri ed è stato predisposto dalla regione umbria sulla base del D.M. 8 settembre 1999, n. 350. Esso è rilasciato con la licenza **Creative Commons Attribuzione 4.0 International (CC-BY 4.0)** che è una licenza della famiglia Creative commons che permette agli altri di distribuire, modificare e sviluppare anche commercialmente l'opera, con il vincolo di riconoscere sempre l'autore originale. Date queste caratteristiche possiamo affermare che questa licenza è compatibile con i dati aperti. Questo dataset è rilasciato in formato csv che è un formato di terzo livello, cioè presenta dati strutturati rilasciati con un formato aperto e non proprietario)

- https://www.dati.gov.it/dataset/prodotti-tradizionali-trentini (questo dataset contiene informazioni sui prodotti tradizionali trentini ed è stato rilasciato dalla provincia autonoma di Trento con la licenza **Creative Commons CC0 1.0** che è una licenza della famiglia Creative commons che permette di copiare, modificare, distribuire ed utilizzare l'opera, anche per fini commerciali, senza chiedere alcun permesso e senza alcun vincolo neanche il riconoscimento dell’autore dell’opera. Date queste caratteristiche possiamo affermare che questa licenza è compatibile con i dati aperti. Questo dataset è rilasciato in formato csv che è un formato di terzo livello, cioè presenta dati strutturati rilasciati con un formato aperto e non proprietario)

- https://www.dati.gov.it/dataset/ricette-tipiche-trentine (questo dataset contiene informazioni su alcune ricette di prodotti tradizionali trentini ed è stato rilasciato dalla provincia autonoma di Trento con la licenza **Creative Commons CC0 1.0** che è una licenza della famiglia Creative commons caratterizzata dal fatto che la persona che ha associato un'opera con questa licenza ha dedicato l'opera al pubblico dominio attraverso la rinuncia a tutti i suoi diritti sull'opera in tutto il mondo come previsti dalle leggi sul diritto d'autore, inclusi tutti i diritti connessi al diritto d'autore o affini, nella misura consentita dalla legge. Date queste caratteristiche possiamo affermare che questa licenza è compatibile con i dati aperti. Questo dataset è rilasciato in formato csv e xml che sono formati di terzo livello, cioè presentano dati strutturati rilasciati con un formato aperto e non proprietario)

La ricerca è stata estesa ad altri siti che rilasciano dati con licenza open senza risultati finchè è stato trovato questo dataset sul sito del ministero delle politiche agricole, alimentari, forestali e del turismo (https://www.politicheagricole.it/flex/cm/pages/ServeBLOB.php/L/IT/IDPagina/13762).
Esso è la diciannovesima revisione, ovvero è un aggiornamento al 07/02/2019 dell’elenco nazionale dei prodotti agroalimentari tradizionali ai sensi dell’articolo 12, comma 1 della legge 12 dicembre 2016, n.238. È rilasciato con la licenza **Creative Commons Attribuzione 4.0 International (CC-BY 4.0)**

che è una licenza della famiglia Creative commons. Si può affermarlo poichè sono dati pubblicati da una pubblica amministrazione che non ha specificato la licenza di rilascio e se ciò avviene la licenza di rilascio è proprio quella citata prima. Questo dataset è in formato xlsx che è un formato di secondo livello, ovvero presenta dati strutturati ma rilasciati in un formato proprietario, in questo caso il proprietario di quel formato è Microsoft.
Ricapitolando le licenze in gioco sono CC-BY e CC0, esse sono compatibili a patto che la licenza sotto cui si rilasci l'opera derivata dall'utilizzo di questi dataset sia la CC-BY.

ELABORAZIONE DATI

A questo punto avendo raccolto dei dati si è passati alla fase di elaborazione degli stessi. Il file (https://www.politicheagricole.it/flex/cm/pages/ServeBLOB.php/L/IT/IDPagina/13762) si è presentato particolarmente ostico a causa del fatto che:

1. è formato da 21 fogli (uno per ogni regione),
2. il foglio denominato "MOLISE" presenta una tabella doppia subdolamente nascosta dal creatore del dataset facendo partire il foglio dalla colonna J. Infatti facendo scorrere la barra si nota un'altra tabella identica in corrispondenza delle colonne A,B e C.
3. La colonna tipologia presenta delle celle unite in cui per un tot numero di celle la tipologia viene immessa soltanto una volta (nella prima riga a partire dall'alto delle celle unite ES. Foglio ABRUZZO celle dalla A3 alla A9 sono unite ma il nome 'bevande analcoliche, distillati e liquori' è effettivamente presente solo nella cella A3 anche se graficamente non sembra così)
4. La prima riga di ogni foglio è completamente inutile (riporta la regione dei prodotti, si è ovviato creando una colonna regione che desse il nome della regione di quel prodotto)
5. è presente una nota finale in alcuni fogli (campania, emilia-romagna, friuli, lazio, marche, veneto, prov. Bolzano), ovvero la stringa "*deroga alle norme igienico sanitarie" che va eliminata.

Per ovviare a tutti questi problemi sono state effettuate varie elaborazioni che partono dalla riga di codice 7 e finiscono alla riga di codice 78. Per tutte le elaborazioni ci siamo serviti delle librerie pandas e openpyxl. Innanzitutto ci siamo concentrati sul dividere i 21 fogli in differenti file a questo scopo abbiamo tentato di costruire un algoritmo che togliesse tutti gli altri fogli facendone restare soltanto uno e ripetendo questo metodo per tutti i 21 fogli ma nonostante ci siamo riusciti l'algoritmo era troppo lento, lo abbiamo sostituito con quello che si trova scritto nel file python denominato 'codice' e che va dalla riga 13 alla riga 53. è più veloce di quello precedente ma risulta essere anche questo abbastanza lento ma ci siamo accontentati del risultato ottenuto. L'elaborazione comincia dalla riga 32 dove si richiama la funzione reader da noi definita che apre il dataset, apre il foglio relativo ad una regione (attraverso il nome del foglio), e per ogni riga di questo foglio, (a partire dalla seconda riga per risolvere il problema numero 4 e limitandosi alla terza colonna (la C) per risolvere il problema numero 2) accede ad ogni cella di una riga e copia le celle in un array e poi copia questo array in un altro array creando un array di array dove gli elementi sono le righe del foglio (corrisponde alle righe da 13 a 28). In seguito apre un nuovo file, prende il foglio attivo e copia riga per riga nel nuovo file servendosi dell'array precedentemente creato, in questo modo otteniamo un foglio singolo del file originale (righe da 31 a 36). A questo punto risolviamo i problemi numero 3 e 5 con le righe di codice da 39 a 53. Quello che facciamo è partire dal valore della cella B2 e ciclare spostandoci verso sotto nella colonna B incrementando il numero alla destra di B (salvato nella variabile N), in questo modo il ciclo si arresterà nel momento in cui non troverà più un numero all'interno della cella, ovvero quando non necessiteremo di inserire stringhe nelle celle della colonna A. In ogni fase del ciclo se troviamo una stringa visitando

la cella nella colonna A, che si trova nella stessa riga della cella della colonna B precedentemente visitata, la conserviamo nella variabile string altrimenti scriviamo la cella nella colonna A con la stringa che abbiamo conservato precedentemente nella variabile. Il ciclo funziona poichè le stringhe nella colonna A sono presenti solo nella prima cella a partire dall'alto delle celle unite e in quelle sotto metteremo la stringa precedentemente memorizzata, quando passiamo alla prima cella di un nuovo raggruppamento di celle salviamo la nuova stringa e così via. Alla fine del ciclo ci ritroveremo con la variabile N che avrà come valore l'intero successivo all'ultimo numero di riga della colonna B che contiene un intero(ES. Foglio emilia romagna N sarà uguale a 399 ovvero l'intero successivo al numero 398 che è l'ultimo numero di riga della colonna B che contiene un intero ovvero il 396). In questo modo incrementando di 1 la variabile N a prendendo il valore della colonna A in corrispodenza del valore N ci ritroviamo con la cella che contiene la stringa da eliminare nel problema numero 5 e settandola a None abbiamo risolto.
Fatto questo utilizziamo la libreria pandas per:

- aggiungere la colonna regione e dare I dati corretti per ogni riga in base a quale foglio ci troviamo(riga codice 57)
- rinominare la colonna denominata tipologia in categoria poichè ci sembra un nome più consono (riga codice 59)
- eliminare la colonna N poichè identificheremo I prodotti con il loro nome e non con un numero (riga codice 61)
- rimpiazzare gli spazi con underscore ed eliminare tutti I caratteri che non siamo lettere numeri o underscore per evitare problemi con l'encoding (righe codice 62-63)
- salvare il foglio in formato csv

E tutto quello che abbiamo descritto viene fatto per ogni foglio del file iniziale in modo tale da ritrovarci 21 file in csv, uno per ogni regione. In seguito li abbiamo uniti tutti in un unico file csv denominato regioni_finale e abbiamo eliminato I file iniziali(righe codice da 70 a 78)

Nel file (https://www.dati.gov.it/dataset/prodotti-tradizionali-trentini)
abbiamo effettuato le seguenti elaborazioni (utilizzando la libreria pandas):

- abbiamo rinominato il file in TRENTINO.csv per essere riconoscibile
- abbiamo eliminato la colonna denominata url poiché l'url di ogni prodotto presentava le stesse informazioni presenti nel dataset in forma di pagina web, quindi nulla di nuovo (r.c. 87)
- Abbiamo rinominato le colonne per avere coerenza con altri file (r.c. 89-94)
- abbiamo sostituito gli spazi multipli con un carattere underscore ed eliminato i caratteri che non sono lettere, numeri o underscore per non avere problemi con l'encoding (r.c. 96-97)
- abbiamo aggiunto la colonna regione dando ad ogni prodotto la sua regione ovvero la stringa 'trentino' (r.c. 99)

Nel file (https://www.dati.gov.it/dataset/prodotti-agroalimentari-tradizionali)
abbiamo effettuato le seguenti elaborazioni (utilizzando la libreria pandas):

- abbiamo rinominato il file in UMBRIA.csv per essere riconoscibile
- Abbiamo rinominato le colonne per avere coerenza con altri file (r.c. 106-110)
- abbiamo sostituito gli spazi multipli con un carattere underscore ed eliminato i caratteri che non sono lettere, numeri o underscore per non avere problemi con l'encoding (r.c. 112-113)
- abbiamo aggiunto la colonna regione dando ad ogni prodotto la sua regione ovvero la stringa 'umbria' (r.c. 115)

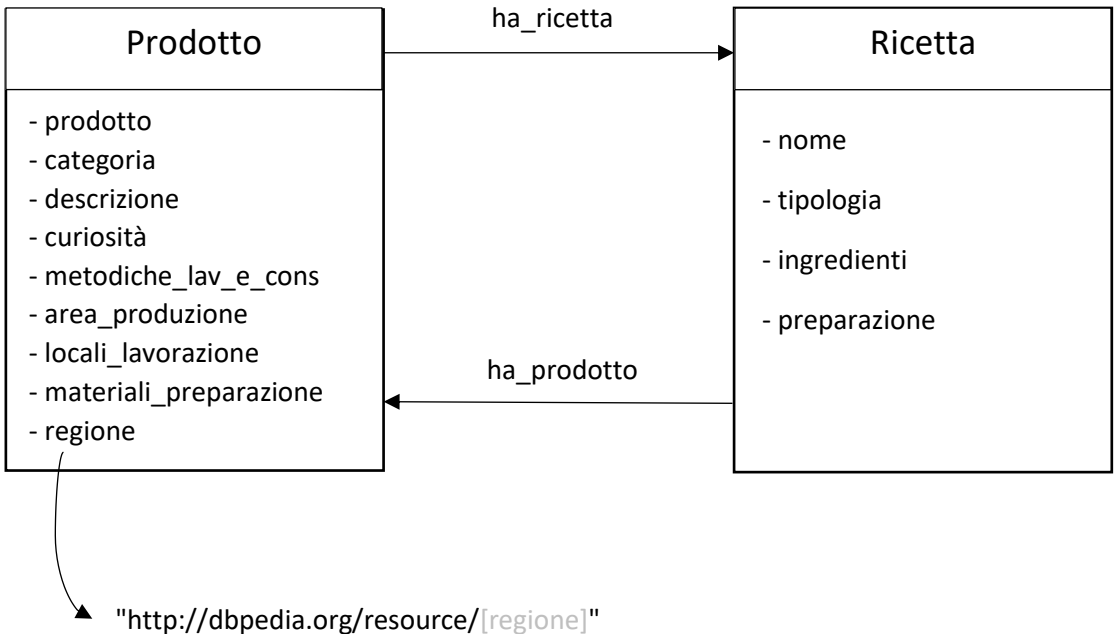
Nel file (https://www.dati.gov.it/dataset/ricette-tipiche-trentine)
abbiamo effettuato le seguenti elaborazioni (utilizzando la libreria pandas):

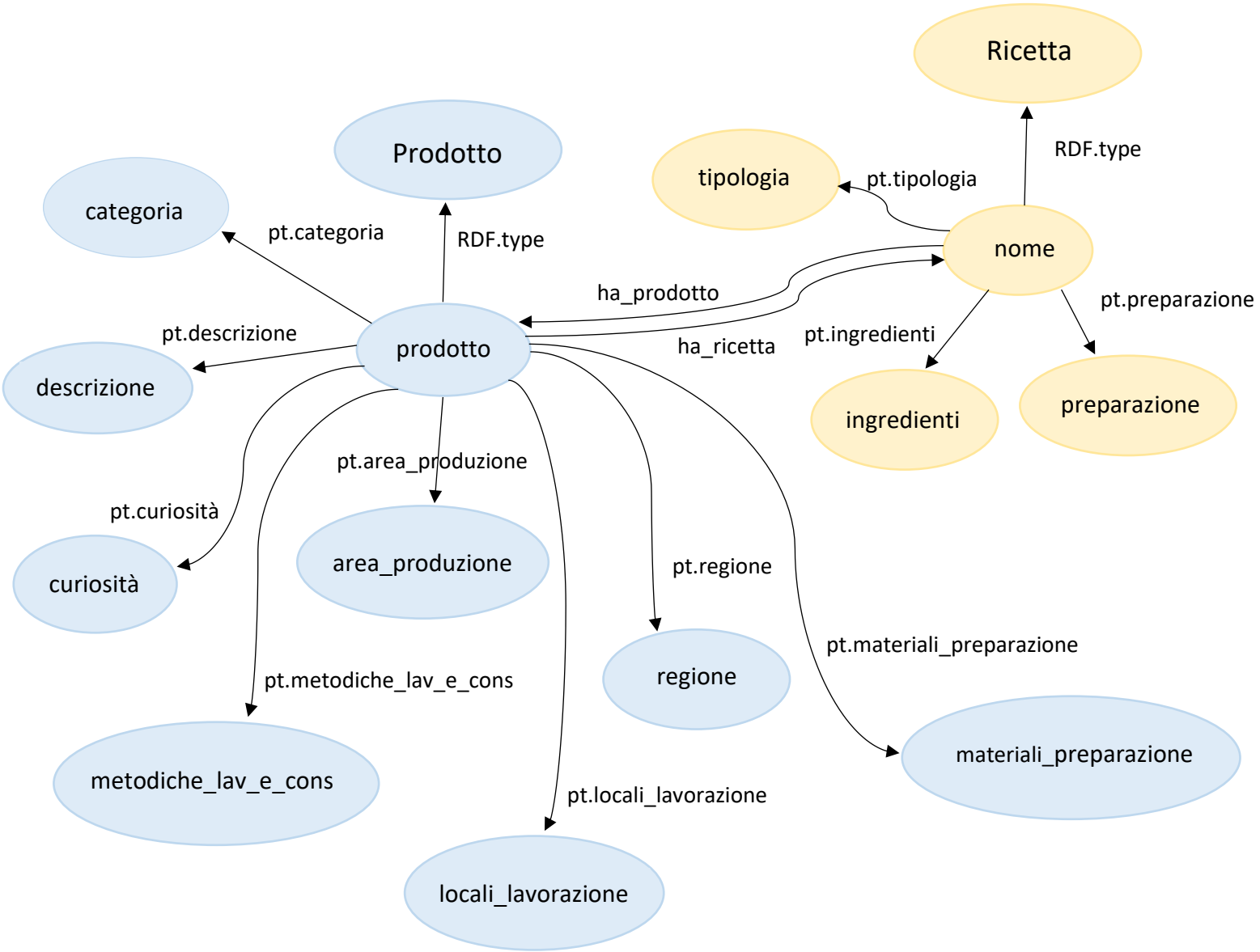
- abbiamo eliminato la colonna denominata RecipID poiché identifichiamo le ricette con il loro nome e non abbiamo bisogno di un numero identificativo (r.c. 122)
- Abbiamo rinominato le colonne per avere coerenza con altri file (r.c. 124-127)
- abbiamo sostituito gli spazi multipli con un carattere underscore ed eliminato i caratteri che non sono lettere, numeri o underscore per non avere problemi con l'encoding (r.c. 130-131)

Fatto questo ci siamo ritrovati con 4 file che non abbiamo unito poichè essi presentavano delle informazioni in più o in meno, ad esempio il file regioni_finale presenta solo la categoria e la regione e unirlo con gli altri due file che descrivono prodotti (I file UMBRIA.csv e TRENTINO.csv) significava inserire una vasta quantità di valori non definiti e abbiamo deciso per questo motivo di non farlo, e inserire le triple nel nostro grafo per ogni file come si evince dal codice. Invece I file 'TRENTINO.csv' e 'ricette.csv' sono stati uniti nelle righe che presentavano lo stesso nome e salvati in un altro file denominato 'ricette-prodotto.csv' in modo tale da trovare quei prodotti di cui si conosceva la ricetta. Putroppo con questo metodo sono state trovate poche unioni, dovute al fatto che in alcuni casi I due nomi confrontati erano l'uno una sottostringa dell'altro e non combaciavano perfettamente. Data la natura esigua delle ricette che abbiamo elaborato ci siamo accontentati dei risultati raggiunti.

ONTOLOGIA

L'ontologia l'abbiamo costruita servendoci dell'applicazione Protege, innanzitutto la base della nostra ontologia è l'url: http://www.prodotti_tipici.org/resource/ mentre il prefisso è:
http://www.prodotti_tipici.org/ontology/
Abbiamo definito due classi nella nostra ontologia: la classe Prodotto e la classe Ricetta.
Ogni istanza della classe Prodotto viene identificato dalla base dell'uri seguito dal nome del prodotto, ogni istanza della classe Ricetta viene identificato dalla base dell'uri seguito dal nome della ricetta con un underscore finale per evitare casi di omonimia con i prodotti (r.c. 182).
La classe Prodotto ha come proprietà: categoria, descrizione, curiosità, metodiche_lav_e_cons, area_produzione, locali_lavorazione, materiali_preparazione e regione. Tuttavia la maggior parte dei prodotti hanno soltanto la categoria e la regione che è una proprietà che 'collega' il Prodotto con una risorsa dbpedia(un url), effettuando quindi un interlinking, che descrive la regione. Inoltre è presente la proprietà ha_ricetta che 'collega' una istanza della classe Prodotto con una istanza della classe Ricetta(un url) nel caso in cui quel prodotto abbia una ricetta. A parte queste ultime due tutte le altre proprietà del prodotto associano una stringa.
La classe Ricetta ha come proprietà: tipologia, ingredienti e preparazione che associano una stringa. Inoltre è presente la proprietà ha_prodotto che è l'opposto della proprietà ha_ricetta e associa una istanza della classe Ricetta(un url) con una istanza della classe Prodotto(un url) nel caso in cui quella ricetta abbia un prodotto.


Prodotto
- prodotto
- categoria
- descrizione
- curiosità
- metodiche_lav_e_cons
- area_produzione
- locali_lavorazione
- materiali_preparazione
- regione
ha_ricetta
ha_prodotto
Ricetta
- nome
- tipologia
- ingredienti
- preparazione
"http://dbpedia.org/resource/[regione]"


Ricetta
Prodotto
tipologia
pt.tipologia
RDF.type
categoria
pt.categoria
RDF.type
nome
ha_prodotto
pt.preparazione
pt.descrizione
prodotto
ha_ricetta
pt.ingredienti
descrizione
ingredienti
preparazione
pt.area_produzione
pt.curiosità
curiosità
area_produzione
pt.regione
pt.materiali_preparazione
regione
pt.metodiche_lav_e_cons
materiali_preparazione
metodiche_lav_e_cons
pt.locali_lavorazione
locali_lavorazione