POLITECNICO DI MILANO

Facoltà di Ingegneria Industriale e dell'Informazione

Corso di Laurea in Ingegneria Informatica
Dipartimento di Elettronica, Informazione e Bioingegneria

UN SISTEMA A SUPPORTO DELLA RACCOLTA E ANALISI DEI DATI DESTRUTTURATI PER CONTESTI DATA-INTENSIVE

Relatore: Prof. Chiara Francalanci

Correlatori: Ing. Francesco Merlo

Marco Vettorello

Tesi di Laurea di:

Marco Chiappone Matr. 755450

Anno Accademico 2013 – 2014

INDICE

INDICE DELLE FIGURE

ABSTRACT

Lo sviluppo dell'informatica e la nascita di Internet offre interessanti scenari per un nuovo approccio di vendita delle proprie conoscenze da parte delle aziende, che possono fornire i propri contenuti e servizi tramite il web.
È necessario però essere certi dell'affidabilità dei servizi proposti e gestire correttamente le informazioni possedute: da qui nasce il bisogno di utilizzare sistemi di gestione delle basi di dati, gli RDBMS e i NoSQL data.
Dalla volontà di Accurat, un'agenzia di ricerca data-driven, design e innovazione, di migliorare la propria pianificazione dell'organizzazione dei dati e di riutilizzarli per poter ottenere nuove informazioni, nascono gli obbiettivi di questo lavoro di Tesi.
Gli scopi principali sono la progettazione e la realizzazione degli strumenti e dei metodi che consentano, con un approccio simile a un data warehouse, il recupero di dati da fonti eterogenee, strutturate, semistrutturate e non strutturate, allo scopo di memorizzarli su un data store e in seguito analizzarli per creare nuova conoscenza.
In particolare, come ausilio per l'analisi, è stata anche creata un'applicazione web che consente di eseguire facilmente, tramite un menu, alcune interrogazioni a Elasticsearch, un *search server* basato su Lucene utilizzato in questo progetto come database NoSQL per contenere i dati.

1. INTRODUZIONE

Negli ultimi decenni si è assistito a un rapido sviluppo delle tecnologie informatiche e inevitabilmente le aziende hanno approfittato dell'avvento di Internet per aumentare il proprio numero di utenti, fornendo attraverso il web i propri contenuti e servizi. Occorreva però essere certi dell'affidabilità dei servizi proposti e gestire correttamente le informazioni possedute, e questo ha portato all'utilizzo dei sistemi di gestione di basi di dati relazionali, gli RDBMS. Nel tempo si è poi sviluppata un'alternativa che sempre più aziende hanno cominciato a prendere in considerazione, ovvero i NoSQL data stores che, come spiegato in seguito, consentono di gestire i dati in maniera diversa e, in certi contesti, più efficace.

Questo lavoro di Tesi nasce dalla necessità di Accurat, agenzia di ricerca data-driven e design presso cui è stato svolto lo stage, di avere un'architettura che, tramite un approccio analogo a un data warehouse, recuperasse i dati da fonti eterogenee presenti sul Web, con lo scopo di effettuarne la pulizia, memorizzarli su un data store e infine di analizzarli per ottenere nuova conoscenza. Grazie alla struttura modulare del progetto, le fasi si salvataggio e di analisi sono possibili anche per quei dati che l'azienda già possiede.

Per rendere agevole quest'ultima operazione, sono state realizzate alcune query allo scopo di mostrare come interagire con il sistema di memorizzazione per ottenere informazioni indipendentemente dal fatto che la sorgente da cui sono state ottenute fosse strutturata, semistrutturata o non strutturata. È stata quindi creata una semplice applicazione web che, sfruttando queste query, consente a qualsiasi utente di interrogare Elasticsearch.

La struttura della Tesi è la seguente:

- il Capitolo 2 mostra una panoramica della letteratura riguardo l'argomento trattato, introduce i sistemi SQL e NoSQL, soffermandosi a descrivere in particolare Elasticsearch, prosegue trattando dei data warehouse e delle fasi che li costituiscono, e termina presentando Accurat.
- Il Capitolo 3 espone le necessità di Accurat in merito alla interazione tra le fonti eterogenee di dati e l'indipendenza da queste delle informazioni una volta recuperate; sulla base di ciò sono stati elaborati gli obiettivi e i requisiti del sistema creato per questo lavoro di Tesi.
- Nel Capitolo 4 viene descritto il progetto, con le sue problematiche e le procedure che hanno condotto alla realizzazione dell'architettura. Vengono introdotte le sorgenti da cui vengono recuperati i dati e mostrati i processi per la loro estrazione, trasformazione, caricamento e organizzazione su Elasticsearch. Il Capitolo continua con la spiegazione delle query utilizzate, del DAL per interagire coi dati e termina presentando le visualizzazioni prodotte da Accurat mediante le informazioni recuperate.
- Il Capitolo 5 mostra una panoramica conclusiva, analizzando i risultati ottenuti e descrivendo alcuni possibili sviluppi futuri.

2. STATO DELL'ARTE

A differenza di altre tecnologie, l'informatica supporta il ciclo di vita di una particolare risorsa aziendale, l'informazione, che rappresenta l'oggetto sia dei processi produttivi o operativi, sia delle attività gestionali. Da questo deriva che l'informazione ha un ruolo organizzativo come risorsa delle operazioni di coordinamento e controllo che presiedono ai compiti operativi, gestendone le interazioni e monitorandone le prestazioni.
Questo Capitolo si pone gli obiettivi di discutere nel Paragrafo 2.1 le caratteristiche generali dei Sistemi NoSql e le principali differenze dai Sistemi Sql: in particolare nel Sottoparagrafo 2.1.1 è introdotto il motore di ricerca Elasticsearch. Nel Paragrafo 2.2 si analizzano i Data Warehouse, indicandone l'architettura, la rappresentazione dei dati e come questi vengono realizzati, mentre nel Paragrafo 2.3 viene presentata Accurat, l'azienda presso cui è stato svolto questo lavoro di Tesi.

2.1. Sistemi Sql e NoSql

La nascita e lo sviluppo di Internet a partire dagli Anni '90 ha permesso a molte aziende di ampliare il proprio numero di utenti, fornendo online una grande quantità di servizi e contenuti. Uno dei casi più eclatanti è l'azienda di commercio elettronico Amazon, nata nel 1994 e che nel 2014 ha fatturato 88,99 miliardi di dollari, con un incremento del 20% rispetto all'anno precedente, quando il fatturato era stato di 74,45 miliardi di dollari. [1]
Questo ha avuto come conseguenza un notevole aumento dei dati presenti sulla rete, mentre ai sistemi informatici si chiedeva di elaborare molte più informazioni, con la necessità di una maggiore richiesta di potenza di calcolo e una migliore organizzazione dei dati.

All’evoluzione del World Wide Web, il Web 2.0, tramite tutte quelle applicazioni online che permettono un elevato livello di interazione tra il sito web e l'utente (ad esempio blog, forum, chat, piattaforme di condivisione di media e social network), è seguito l’aumento di aziende che operano solamente online e che forniscono numerosi contenuti strutturati; conseguentemente è diventato fondamentale che i servizi siano affidabili e che tali contenuti siano gestiti in maniera adeguata. [2]

Gli RDBMS, ovvero i sistemi di gestione di basi di dati relazionali, costituiscono il principale metodo di archiviazione di dati strutturati per le tradizionali applicazioni web e aziendali. I database relazionali si basano su relazioni che consentono l’organizzazione dei dati in insiemi di record: in pratica, tutti i dati da trattare sono memorizzati in strutture fisse, dette tabelle, le cui righe rappresentano specifici record (o tuple) e le cui colonne corrispondono ai campi del record. Ciascuna di queste colonne ha una intestazione e solo associando questa a uno dei dati presenti nella tabella è possibile estrapolare dell’informazione.

In questo contesto, ha rilevante importanza il concetto di *schema* di una base di dati, costituito dalle caratteristiche dei dati, ovvero dal nome dell’entità seguito dai nomi dei suoi attributi, ad esempio:

Videogioco (Titolo, Brand, DataRilascio).

Il concetto di *istanza* si lega invece ai valori memorizzati nella tabella, i quali variano nel tempo.

Le informazioni possedute vengono quindi suddivise in diverse tabelle su cui è possibile eseguire delle interrogazioni, ovvero delle funzioni che producono una relazione su un dato schema. Il concetto di relazione è legato alla teoria degli insiemi, dove rappresenta un sottoinsieme del prodotto cartesiano di *n* domini o componenti e quindi non c’è ordinamento fra le sue n-uple, tutte distinte tra loro. Nel modello relazionale invece le componenti sono dette attributi, ognuno caratterizzato da un nome e un insieme di valori;

un generico elemento di una relazione con attributi viene definito tupla. Una relazione può essere utilizzata per organizzare dei dati importanti nel contesto di una particolare applicazione e generalmente una base di dati è costituita da più relazioni.

Allo scopo di formulare una o più interrogazioni, anche complesse, è possibile costruire espressioni che coinvolgano delle operazioni logiche dette *operatori*, come ad esempio quello di unione, intersezione, differenza, selezione e join. Quest'ultimo in particolare è il più caratteristico dell'algebra relazionale poiché consente di correlare dati contenuti in relazioni diverse utilizzando le chiavi esterne che collegano una tabella a un'altra. Di questo operatore esistono due varianti principali: il join naturale che combina le tuple di due relazioni sulla base dell'uguaglianza dei valori degli attributi comuni alle due relazioni, e il theta-join, ovvero un prodotto cartesiano tra relazioni seguito da una selezione

Grazie a questi operatori è quindi possibile distribuire tutte le informazioni su tabelle SQL, così da rendere più facile la gestione delle stesse senza la necessità di ricorrere a sistemi di calcolo dall'elevata potenza.

Come conseguenza di questa ripartizione della conoscenza, è necessario preservare l'integrità del database, controllando sia le relazioni che la validità dei dati memorizzati nelle tabelle: è questo uno dei motivi per cui i database relazionali si basano su schemi entità-relazione che forniscono una serie di strutture, dette *costrutti*, utilizzate per definire schemi che descrivono l'organizzazione e la struttura dei valori assunti dai dati al variare del tempo. [3]

Per molti anni è stato questo il metodo di memorizzazione utilizzato poiché ritenuto la migliore soluzione per l'immagazzinamento dei dati; tuttavia negli ultimi anni la ricerca di una via alternativa che consentisse di diminuire i costi senza intaccare le prestazioni e anzi, ove possibile, di migliorarle, ha condotto alla progettazione e a un uso sempre maggiore di nuovi sistemi,

detti NoSQL data stores, che grazie alla scalabilità orizzontale consentono di raggiungere ottime prestazioni nelle operazioni di scrittura/lettura su database distribuiti su più server, e che utilizzano nodi a basso costi i quali consentono di ridurre i costi.

Mentre infatti nei tradizionali RDBMS i dati risiedono in un unico nodo e la scalabilità si ottiene incrementandone la capacità di elaborazione (scalabilità verticale), nei sistemi NoSQL i dati vengono partizionati su più macchine (scalabilità orizzontale) in modo tale che ogni nodo contenga solo parte dei dati.

Molteplici sono le caratteristiche che consentono di distinguere questi due tipologie di database. Anzitutto, mentre in quelli SQL, come detto, i dati sono memorizzati in tabelle e le informazioni sono quindi distribuite in differenti strutture logiche, in quelli NoSQL le informazioni sono conservate in documenti la cui natura può essere di tipo key-value o Document Store basati su semantica JSON.

Il primo caso è la forma primitiva di database NoSQL e fa uso di un array associativo come modello di dati principale, rappresentando i dati come un insieme di coppie chiave-valore in modo tale che ogni chiave appaia al massimo una sola volta nella collezione. [4]

Il secondo caso è invece è un sottoinsieme del primo e si basa sul concetto di documento, il quale contiene una struttura interna con metadati che il motore del database può interpretare e utilizzare come ulteriore ottimizzazione.

Ognuno di questi documenti aggregati raccoglie i dati associati a una entità che può quindi essere trattata da un'applicazione come oggetto: in questo modo si valutano in una sola volta tutte le informazioni che a tale entità fanno riferimento, evitando i pesanti calcoli computazionali richiesti in fase di aggregazione in quanto tutti i dati necessari e corrispondenti a uno stesso oggetto sono già disponibili in un unico documento. La conseguenza più immediata è una maggiore velocità di risposta nelle query di ricerca di uno o

più termini all'interno di un documento, caratteristica ricercata in questo lavoro di Tesi che ha previsto anche la realizzazione dell'interfaccia per l'interagire con i dati salvati su Elasticsearch, uno dei più celebri motori di ricerca, definiti come sistemi di gestione di un database NoSQL e dedicati alla ricerca del contenuto dei dati fornendo un supporto per complesse espressioni di ricerca e ricerca full text. [5]

Un'altra caratteristica che distingue i database NoSQL da quelli relazionali è l'assenza di tabelle e questo consente loro di essere schema-less, ossia privi di uno schema predefinito, caratteristica che conferisce a questo tipo di base di dati un vantaggio non trascurabile.

In un database SQL infatti, tutti i campi di una tupla devono avere un valore, cui al limite viene assegnato il marker null quando questo è mancante e quindi sconosciuto. L'alternativa NoSQL, con la sua mancanza di schemi, consente di evitare il problema di tabelle con troppi valori `null`, eliminando la necessità di memorizzare un valore per ciascuno dei valori delle righe dei documenti ove questi siano mancanti, diminuendo la quantità di dati da memorizzare e rendendo più fluida la lettura delle tabelle stesse.

Inoltre tali sistemi sono altamente scalabili e l'errore di un nodo non pregiudica totalmente il funzionamento dell'applicazione, con la conseguenza di una maggiore tolleranza ai guasti.

A fronte di tali vantaggi, gli svantaggi più rilevanti sono la necessità di una facile amministrazione e la richiesta di grandi sforzi progettuali affinché l'applicazione possa supportare la scalabilità orizzontale tipica dei sistemi NoSQL. [6]

Google e Amazon hanno creato i propri sistemi proprietari di data storage NoSQL, BigTable e DynamoDB, ma nel tempo sono stati sviluppati altri importanti progetti, quasi tutti open source, con lo scopo di creare nuovi strumenti di gestione dei dati che fossero a disposizione di tutti gli sviluppatori. In pochi anni si è aperto un nuovo mercato nel campo dei

database nato per affrontare e trovare soluzioni a problemi introdotti dallo sviluppo di nuove applicazioni, progetti e tecnologie che non potevano più essere affrontate utilizzando un singolo strumento come i database relazionali.

2.1.1. Elasticsearch

Elasticsearch è un motore di ricerca Open Source per la ricerca e l'analisi real-time dei dati, progettato per la scalabilità orizzontale, affidabilità e facilità di gestione. [7]
Il suo sviluppatore, Shay Banon, iniziò ad utilizzare una prima versione di Lucene, con l'intento di costruire un motore di ricerca di ricette per la moglie. Lavorare direttamente con Lucene può essere difficile, quindi decise di cominciare a lavorare su un livello di astrazione per rendere più facile per i programmatori Java l'aggiunta di ricerca alle loro applicazioni. Questo suo primo progetto open source prese il nome di Compass.
Successivamente, le necessità lavorative portarono Banon alla riscrittura delle librerie Compass come server autonomo denominato Elasticsearch, il cui primo rilascio pubblico fu nel febbraio del 2010. Da allora, Elasticsearch è diventato uno dei progetti più popolari su GitHub [1]con commit da oltre 300 collaboratori e attorno ad esso si è costituita una società per fornire supporto commerciale e per sviluppare nuove funzionalità, ma Elasticsearch è, e sempre sarà, open source e disponibile a tutti.
Elasticsearch è scritto anche in Java e, come detto, internamente utilizza Lucene per tutta la sua indicizzazione e la ricerca, ma mira a rendere la ricerca full-text facile nascondendo la complessità di Lucene dietro le API RESTful.

[1] https://github.com/

Per comunicare con ElasticSearch infatti ci sono due metodi: le Java API sulla porta 9300 o le succitate API RESTful sulla porta 9200, i cui metodi sono stati utilizzati in questo lavoro di Tesi.
Ma questo è più che un semplice motore di ricerca, è anche un documento distribuito real-time dove ogni campo è indicizzato e ricercabile, capace di scalare centinaia di server e petabyte di dati, strutturati e non.
E' disponibile sotto la licenza Apache 2, e quindi può essere scaricato, usato e modificato gratuitamente. Permette di ridurre fortemente i tempi di indicizzazione, migliora le prestazioni, l'utilizzo delle risorse e di memorizzare dei dati su larga scala per poi renderli facilmente consultabili ed accessibili. Grazie al suo set di API ed ad una serie di client per i più diffusi linguaggi di programmazione, Elasticsearch mette a disposizione una delle soluzioni di ricerca più avanzate. Caratteristica di questo motore di ricerca è quella di essere senza schema, ovvero basterà immettere un documento scritto in JSON e automaticamente questo sarà indicizzato dal sistema. Infatti i numeri e le date vengono automaticamente rilevate e trattate come tali. L'indicizzazione dei dati avviene tramite un identificatore univoco. Questo approccio semplice permette di ottenere i dati attraverso un indice, equivalente a un database nelle tradizionali basi di dati relazionali, un tipo, che equivale al concetto di tabella nei database relazionali, e un ID. Inoltre Elasticsearch consente di controllare come un documento JSON venga mappato nel motore di ricerca. Per mapping (mappatura) si intende il processo di definizione di come un documento dovrebbe essere associato al motore di ricerca, ivi comprese le caratteristiche di cercabilità come i campi che è possibile cercare e se o come sono simboleggiati. In Elasticsearch, un indice può memorizzare documenti di diversi "tipi di mappatura" e permette di associare più definizioni di mappatura per ogni tipo di mappatura. Il mapping esplicito è definito a livello di indice o tipo. Di default, non occorre specificare una mappatura esplicita poiché questa viene creata e registrata

automaticamente quando viene introdotto un nuovo tipo o un nuovo campo solo quando le impostazioni di default devono essere sovrascritte deve essere fornita una definizione di mapping.

Inoltre Elasticsearch fornisce una Query DSL completa basata su JSON per la definizione delle query che possono anche avere dei filtri ad esse associate. Alcune query possono contenere altre interrogazioni (come la query bool), altre possono contenere filtri, e alcune possono contenere sia una query e un filtro. Ciascuna di queste può contenere qualsiasi richiesta dell'elenco delle query o qualsiasi filtro dall'elenco dei filtri, con conseguente possibilità di creare query parecchio complesse.

Sempre più sono gli utenti che stanno migrando ad Elasticsearch e tra questi ve ne sono alcuni celebri come Wikipedia[2], che lo utilizza per fornire ricerca full-text, per la *Search-as-you-Type* (SayT - per la ricerca dinamica e real-time, il box di ricerca presenta dinamicamente i suggerimenti e completa autonomamente le richieste) o The Guardian[3], che lo utilizza per combinare i log con i dati dei social-network al fine di fornire dei feedback real-time ai suoi redattori riguardo la reazione del pubblico ai nuovi articoli. Stack Overflow[4] combina la ricerca full-text con le query di geolocalizzazione e ne fa uso per trovare domande e risposte relative, mentre GitHub per soddisfare le esigenze di ricerca dei 4 milioni di suoi utenti, lo utilizza per indicizzare oltre 8 milioni di repository di codice e indicizzando i dati degli eventi critici. Due invece le applicazioni musicali che ne fanno uso per incrementare la velocità di ricerca, l'aggiornamento automatico, la scalabilità, la ricerca intelligente per fornire contenuti a decine di milioni di utenti: Deezer[5] e SoundCloud[6].

[2] http://en.wikipedia.org
[3] http://www.theguardian.com/
[4] http://stackoverflow.com/
[5] http://www.deezer.com/
[6] https://soundcloud.com/

Tuttavia Elasticsearch non è solo per le grandi aziende, ma ha consentito a molte startup come Klout[7] di sviluppare le proprie idee trasformandole in soluzioni scalabili. [8]

2.2. Data warehouse

Basi di dati distinte sono spesso integrate a posteriori, costituendo un *data warehouse* che consente l'accesso a più fonti di dati eterogenee con un linguaggio comune e secondo uno schema dei dati integrato e coerente. [9]
Il data warehouse e la conseguente integrazione dell'informazione operativa è fondamentale per fornire una fonte di informazioni unica e facilmente accessibile ai processi direzionali di pianificazione e controllo e ai corrispondenti sistemi informatici di supporto.
La tradizionale architettura di un data warehouse descritta nella ricerca è quella riportata nella Figura 1:

[7] https://klout.com/home

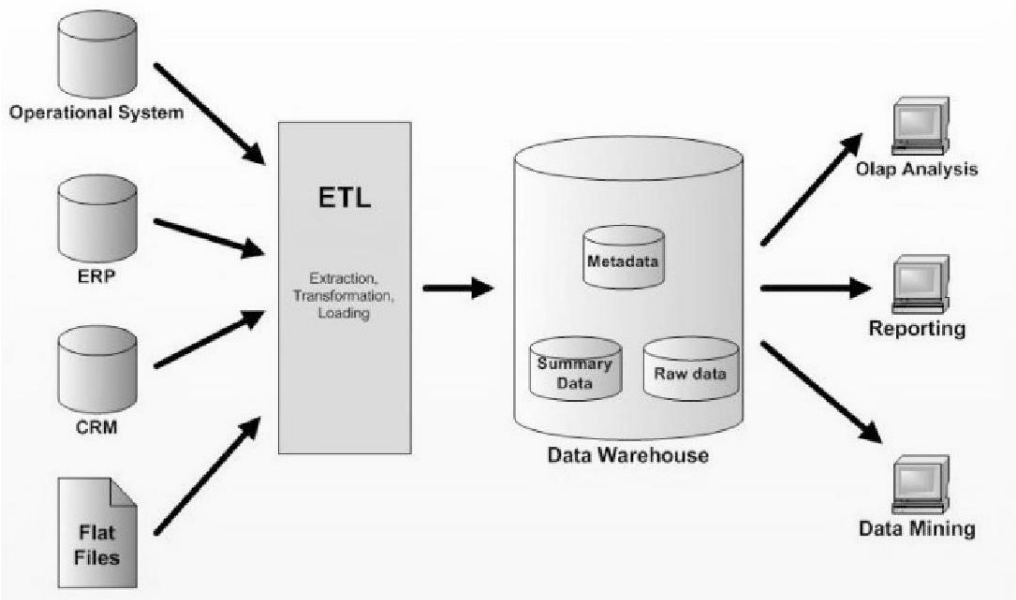

Figura 1: Architettura di una Data Warehouse

I tradizionali sistemi di processi transazionali on-line (denominati OLTP, *Online Transaction Processing*) consentono alle imprese di accumulare grandi quantitativi di dati utili non solo per la gestione dell'impresa, ma anche per pianificare e supportare le decisioni proprio grazie ad opportune manipolazioni dei dati stessi tramite data warehouse. I dati del passato e del presente possono infatti essere utilizzati per la progettazione e la programmazione delle attività future dell'impresa. Tuttavia gli OLTP risultano non del tutto adeguati al supporto decisionale e rimangono evidenti i problemi legati all'accessibilità delle informazioni nonostante l'incremento della velocità di trasmissione dei dati. Al fine di integrare informazioni eterogenee provenienti da diverse fonti, nel tempo i data warehouse hanno assunto un ruolo sempre più importante grazie alla possibilità di poter realizzare dei processi analitici direttamente on-line (detti OLAP, *On Line Analytical Processing*). Mentre i primi hanno come obbiettivo la garanzia di integrità e sicurezza delle transazioni, i secondi cercano la performance nella

ricerca e il raggiungimento di un'ampiezza di interrogazione quanto più grande possibile. [10]

Estendendo la tecnologia delle basi di dati relazionali, i data warehouse forniscono lo strumento pratico per concentrare masse grandissime di informazioni e per strutturarle secondo le esigenze di consultazione e di analisi del management.

Il concetto di data warehouse fu presentato nel 1992 da Inmon, che lo definì come "*una collezione di dati subject-oriented, integrata, non volatile, e variabile nel tempo di supporto ai processi decisionali*". [11]

- Subject-oriented (orientata al soggetto): un data warehouse può essere utilizzato per analizzare una particolare area di attività che è di interesse per i soggetti dell'organizzazione, come ad esempio le "Vendite".
- Integrata: in questo particolare archivio vengono integrati i dati provenienti da diverse risorse. Due sorgenti di dati distinte potrebbero però avere un differente modo di identificare un oggetto, ma in un data warehouse il criterio sarà univoco.
- Non volatile: una volta che i dati sono archiviati, essi non cambiano e l'accesso è consentito in sola lettura. Ciò comporta una maggiore semplicità di progettazione del database rispetto a quella richiesta da un'applicazione transazionale poiché non si considerano le possibili anomalie dovute agli aggiornamenti e non si ricorre a strumenti complessi per gestire l'integrità referenziale o per bloccare record a cui possono accedere altri utenti in fase di aggiornamento.
- Variabile nel tempo: i dati memorizzati in un data warehouse coprono un orizzonte temporale molto più esteso rispetto a quelli archiviati in un sistema operazionale. Nel DW sono contenute una serie di informazioni relative alle aree di interesse che colgono la situazione

relativa ad un determinato fenomeno in un determinato intervallo temporale piuttosto esteso. Questo è in contrasto con ciò che si verifica in un sistema transazionale, dove i dati corrispondono sempre ad una situazione aggiornata, solitamente incapace di fornire un quadro storico del fenomeno analizzato. Ad esempio un sistema transazionale può contenere l'indirizzo più recente di un cliente, mentre una data warehouse può contenere tutti gli indirizzi ad esso associato.

Un data warehouse, come mostrato nella Figura 1, è organizzato su quattro livelli architetturali: il primo si occupa dell'acquisizione dei dati dalle sorgenti e della loro validazione; il secondo si occupa di estrarre, trasformare e caricare i dati dai sistemi transazionali che alimentano il data warehouse; nel terzo livello vengono immagazzinati i dati che hanno superato il livello di trasformazione, mentre nell'ultimo livello sono presenti strumenti per la creazione di query ad hoc, reporting, applicazioni end-user e analisi avanzate dei dati.

2.2.1. Sorgenti

Le sorgenti di dati possono essere *interne*, se risiedono nella competenza dell'azienda, o *esterne* se i dati sono forniti da terze parti.

Esse inoltre, in base al tipo di dati che contengono o a cui forniscono accesso, possono essere distinte in:

- *Strutturate* se i dati sono conservati in database organizzati secondo schemi e tabelle rigide.
- *Semistrutturate* se presentano alcune delle caratteristiche dei dati strutturati e alcune di quelli non strutturati; esempi di questa tipologia di organizzazione di informazioni sono i documenti JSON o XML.

- *Non strutturati* se i dati sono conservati senza alcuno schema, ovvero sotto forma di testo libero; un esempio possono essere i file contenenti testi a carattere narrativo prodotto per mezzo di software di editing testuale, pagine web o un file multimediale.

2.2.2. ETL

Questo livello è formato dai processi di estrazione (Extraction), trasformazione (Transformation) e caricamento (Loading).
I dati vengono estratti da sistemi sorgenti quali database transazionali (OLTP), sistemi informatici come ERP o CRM, o da file di testo. L'estrazione viene definita *statica* se effettuata quando occorre popolare il Data Warehouse per la prima volta, mentre è detta *incrementale* se utilizzata per aggiornare periodicamente l'archivio informatico, estraendo solo i cambiamenti avvenuti nelle sorgenti dall'ultima estrazione.
Successivamente, al fine di rendere omogenei i dati derivanti da sorgenti distinte, su di essi si esegue un processo di trasformazione, ad esempio la normalizzazione dei dati, la selezione dei soli dati di interesse, nella derivazione di nuovi dati calcolati, nel raggruppamento dei dati o nell'esecuzione di join (accoppiamenti) tra dati recuperati da tabelle distinte. Questa trasformazione è anche utile per fare in modo che i dati siano più coerenti al sistema di analisi per il quale vengono sviluppati.
L'ultimo passo consiste nella memorizzazione dei dati in un sistema di sintesi come i data warehouse, ovvero nell'aggiungere o aggiornare un insieme di record per ciascuna tabella.
L'area che racchiude questi processi e che prende il nome di Data Staging, include anche le aree di memorizzazione dei dati estratti pronti per essere caricati e dei tool per preparare i dati al caricamento, ovvero di pulizia (per risolvere conflitti, errori, incompletezze e standardizzare i dati), rimozione di

duplicati o di campi non significativi, trasformazione, combinazione, creazione di chiavi da usare nel data warehouse che saranno diverse da quelle usate nelle sorgenti informative, e archiviazione. [12]

2.2.3. Memorizzazione dei dati

Una volta superato il livello di ETL, i dati vengono memorizzati in questo livello architetturale al fine di creare sintesi informative per gli utenti (aggregazioni e data mart, un sottoinsieme di un data warehouse, specializzato in un particolare soggetto e che consente la formulazione di strategie sulla base di chiari obbiettivi di analisi) tramite operazioni avviate solitamente al completamento dei processi di estrazione, trasformazione e caricamento.

La memorizzazione può avvenire in due modi: in maniera centralizzata, tramite l'ausilio di un data warehouse, oppure in maniera distribuita, tramite un data warehouse seguito da data mart o soltanto data mart.

La rappresentazione dei dati è in forma multidimensionale e questo offre un duplice vantaggio: dal punto di vista funzionale, risulta efficace per garantire tempi di risposta rapidi a fronte di interrogazioni complesse, mentre sul piano logico le dimensioni corrispondono in modo naturale ai criteri di analisi utilizzati dai knowledge worker. [13]

Il modello multidimensionale si basa sul fatto che gli oggetti che influenzano il processo decisionale sono *fatti* del mondo aziendale, ovvero concetti sui quali basare le analisi (ad esempio le spedizioni o le vendite). Le occorrenze di un fatto vengono denominate eventi: ciascuna spedizione o vendita effettuata è un evento. Per ogni fatto vengono in particolare presi in considerazione i valori di un insieme di misure che descrivono quantitativamente gli eventi.

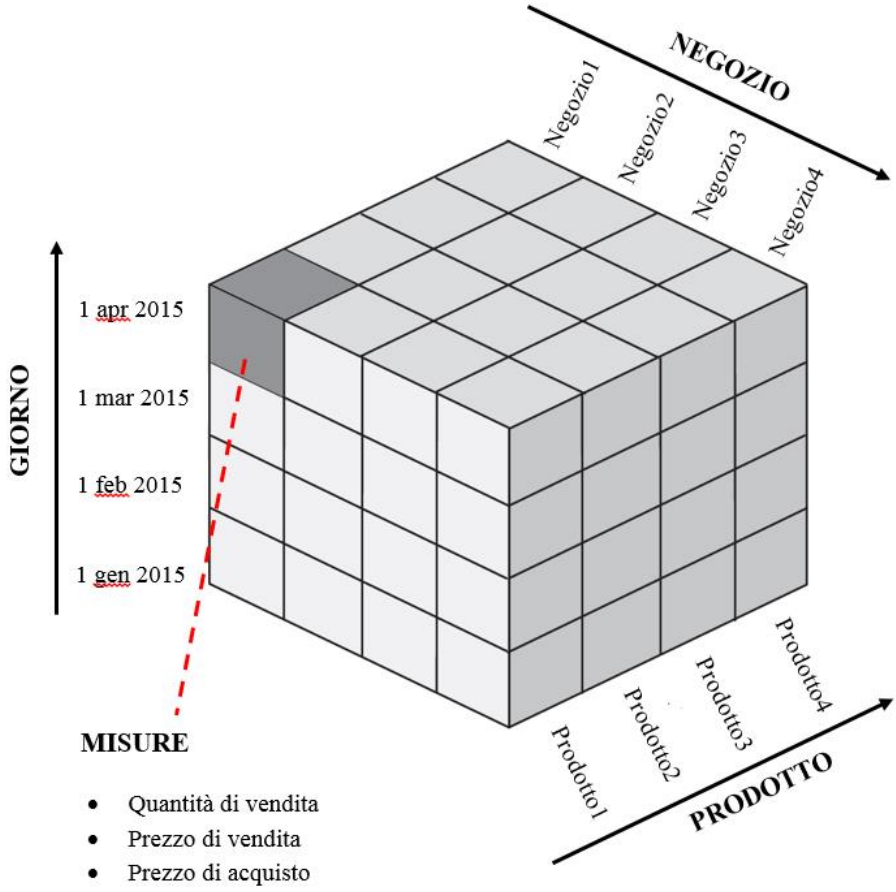

Figura 2: Cubo multidimensionale, esempio vendite in una catena di negozi

A causa dell'elevato numero di eventi presenti all'interno di un'azienda, risulta sconveniente analizzare singolarmente ogni evento: per questo motivo essi vengono idealmente collocati in uno spazio n-dimensionale, al fine di poterli selezionare e raggruppare agevolmente. Gli assi di questo spazio prendono il nome di *dimensioni* di analisi e, ad esempio considerando come fatto le vendite, potrebbero essere i prodotti, il tempo, i negozi o le promozioni.

Il concetto di dimensione genera la metafora del cubo.

2.2.4. Accesso ai dati

In questo livello sono contenuti i sistemi di presentazione delle informazioni agli utenti, raggruppabili in tre grandi categorie.
Nella prima troviamo strumenti per costruire query, strumenti di navigazione OLAP (OLAP viewer) e i Web browser, che stanno diventando l'interfaccia comune per diverse applicazioni.
Nella seconda categoria rientrano le suite per ufficio, ovvero un insieme di applicazioni che permettono all'utente di un computer di creare dei contenuti come documenti di testo, grafici o presentazioni, generalmente per uso personale o lavoro d'ufficio. Spesso, come soluzioni di front end per le informazioni contenute nel livello di memorizzazione, le aziende utilizzano gli strumenti ordinari del lavoro quotidiano, come fogli elettronici e programmi di videoscrittura. Questa soluzione può risultare sia conveniente dal punto di vista dell'efficienza e della produttività, sia rassicurante per gli utenti che si avvicinano per la prima volta al data warehouse, non costringendoli ad imparare nuovi e complessi strumenti.
Nell'ultima categoria sono presenti gli strumenti di grafica e GUI, un tipo di interfaccia utente che consente all'utente di interagire con in data warehouse tramite oggetti grafici convenzionali.

2.3. Accurat

Accurat è un'agenzia di ricerca data-driven, design e innovazione fondata nel 2011 da Giorgia Lupi, Simone Quadri e Gabriele Rossi.
L'azienda ha sede a Milano e opera sul mercato italiano e internazionale. Tra gli obbiettivi preposti vi sono l'analisi di dati, la progettazione di strumenti analitici e visivi atti a creare nell'utente sia comprensione che

consapevolezza e coinvolgimento, ma anche la ricerca di nuovi metodi per strutturare le informazioni in base a necessità e opportunità.

Così come variegati sono gli obbiettivi, altrettanti sono i settori nei quali l'azienda si propone come competitore e tra questi individuiamo l'editoria, i servizi alla cittadinanza (salute, pianificazione urbana), la strategia aziendale (valutazione, monitoraggio e supporto decisionale), la comunicazione, gli eventi.

Accurat, oltre a fornire ai propri clienti la consulenza di cui essi necessitano, offre i propri prodotti e servizi, ideando, progettando e sviluppando:

- Visualizzazioni dati statiche e/o interattive
- Raccolta, analisi e visualizzazione dati di Social Media
- User Experience per prodotti e servizi
- Interfacce web, tablet e mobile
- Strumenti analitici e dashboard
- Piattaforme di information management
- Mappe interattive e strumenti di valutazione urbana

[14]

3. UN SISTEMA PER IL RECUPERO, LA MEMORIZZAZIONE E L'ANALISI DEI DATI PER ACCURAT

I dati generati dalle aziende incrementano in maniera esponenziale ed è per questo motivo che per esse è un imperativo strategico quello di ottenere strumenti capaci di trasformare le informazioni in vantaggi, sia che stiano cominciando nuovi progetti incentrati sui dati posseduti, sia che si vogliano sfruttare al massimo gli investimenti in ambito di Big Data.
Questo lavoro di Tesi nasce dalla necessità di Accurat di avere un sistema per memorizzare e analizzare, secondo le proprie necessità, i dati di cui vuole entrare in possesso o che già possiede. In questo Capitolo viene descritto come l'azienda faccia uso di tale sistema, in particolare nel Paragrafo 3.1 vengono definiti gli obbiettivi di questo lavoro di Tesi, nel Paragrafo 3.2 è descritto a grandi linee il processo di recupero e di pulizia dei dati, nel Paragrafo 3.3 è trattata la trasformazione dei dati puliti in informazione grazie anche alla profondità storica, nel Paragrafo 3.4 si analizza l'integrazione tra fonti di dati eterogenee, mentre il Paragrafo 3.5 fornisce una descrizione dell'indipendenza dai sistemi sorgente.

3.1. Obiettivi

L'incremento del numero di progetti che Accurat ha portato avanti negli anni ha originato inevitabilmente una quantità di dati crescente nel tempo.

Per via dell'ambiente in cui essa opera, spesso i risultati ottenuti nei lavori sono mostrati tramite elementi grafici dove l'informazione è rappresentata più in forma grafica che testuale.

Tuttavia, la volontà dell'azienda di inserirsi in un segmento di mercato più ampio ha portato a chiedersi come poter migliorare la propria pianificazione dell'organizzazione dei dati e come poter utilizzare i dati ottenuti da un progetto al fine di ottenere nuove informazioni che possano essere adoperate sia per formulare e validare ipotesi per il lavoro corrente, ma anche per quelli futuri. Più volte infatti i dati sono stati memorizzati semplicemente su fogli elettronici e programmi di videoscrittura utilizzati come supporto per le grafiche finali, ma esisteva la necessità di avere questi dati in un sistema per creare e manipolare banche di dati, così da poter eseguire query complesse e ottenere risultati costituiti da informazioni derivanti da più sorgenti. Integrare i dati provenienti da fonti eterogenee è quindi diventato molto importante per Accurat, sia per aiutare i dipendenti nella interpretazione dei dati esistenti, sia per consentirgli di ottenere nuova conoscenza, sia per ridurre i tempi e quindi i costi.

Per questo motivo l'azienda ha deciso di realizzare questo progetto, con lo scopo di integrare, in maniera efficiente, i dati recuperati da fonti diverse tra loro. In particolare, l'obbiettivo di questo lavoro di Tesi è quello di creare un'architettura modulare capace di supportare i processi di estrazione, pulizia e caricamento dei dati di differenti sorgenti al fine di consentirne la manipolazione per l'ottenimento di nuova informazione. Quest'ultima fase avviene mediante un'applicazione web che permette una facile ed efficiente consultazione dei dati tramite un browser.

Per ottenere questo proposito sono state create delle procedure di acquisizione automatica di dati con l'intento di importarli e di interagire con le basi di dati che li contengono; questo è avvenuto tramite:

- la modellizzazione dei dati da acquisire dalle sorgenti;

- l’implementazione dei processi di recupero di dati mediante software creati ad hoc in base al tipo di sorgente;
- le operazioni di salvataggio dei dati su file, la segnalazione di anomalie e l’importazione su Elasticsearch;
- la pianificazione di alcune query per interrogare le basi di dati;
- la creazione di una applicazione per interagire con i dati raccolti.

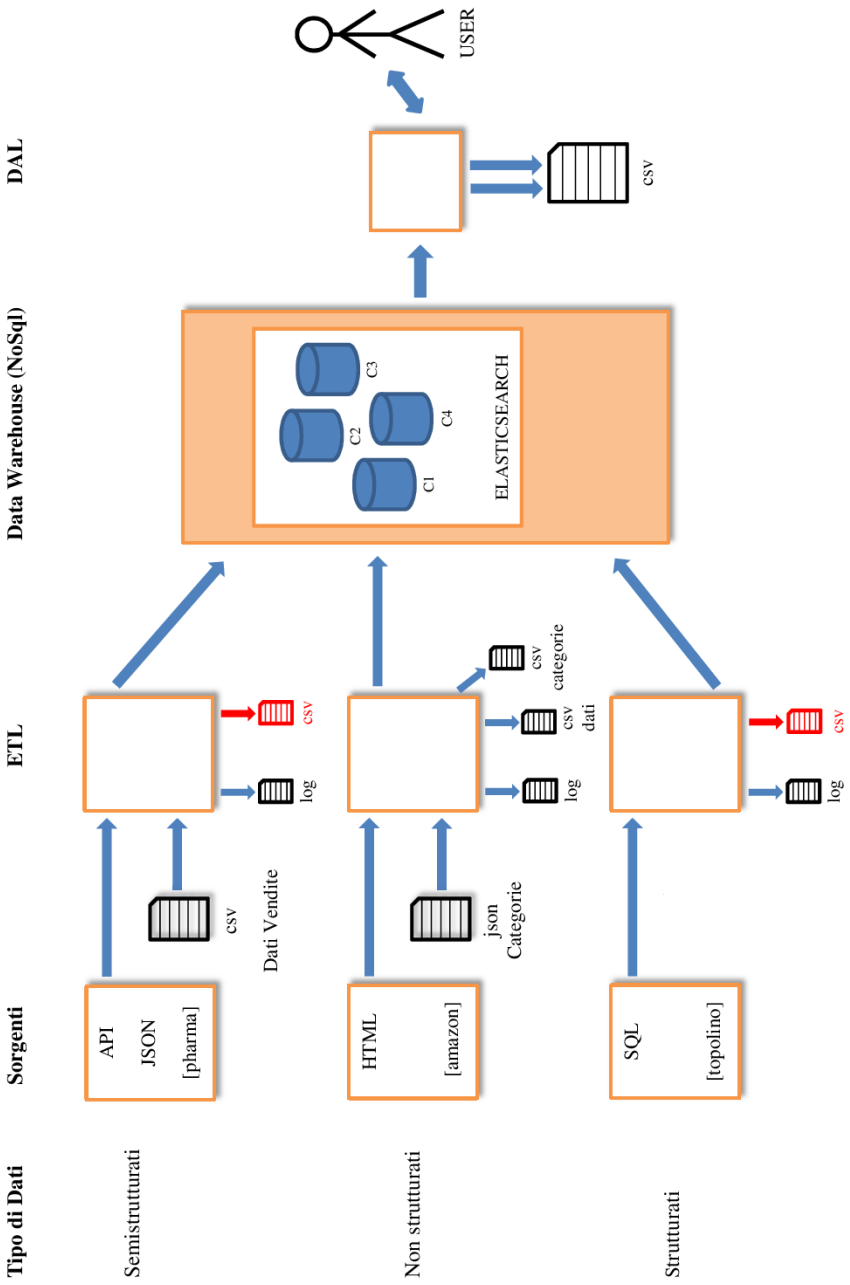

Figura 3: I processi di recupero dati, memorizzazione e interazione per questo lavoro di Tesi

3.2. Recupero e pulizia dei dati

In questo lavoro di tesi, volendo creare un sistema che potesse trattare i dati provenienti da tutte e tre le tipologie di sorgenti descritte nel Paragrafo 2.2.1, ovvero dati strutturati, semistrutturati e non strutturati, sono state prese in esame tre distinte fonti, ovvero rispettivamente:

- COA I.N.D.U.C.K.S.[8], un database che indicizza i contenuti dei fumetti Disney
- Il portale dell'Agenzia Italiana del Farmaco (AIFA)[9]
- Il sito web italiano di Amazon.com, Inc.[10], azienda di commercio elettronico.

Il recupero dei dati, a seconda della tipologia delle sorgenti, è avvenuta in maniera distinta: nel primo caso i dati, già in possesso di Accurat, erano stati recuperati tramite il dump della base di dati, importandola in un calcolatore così da eseguire in locale delle query SQL; quelli contenuti nel sito AIFA sono stati estrapolati tramite API JSON, mentre nell'ultimo caso l'estrazione è avvenuta tramite web scraping dell'HTML, una tecnica informatica di estrazione di dati da un sito web, mediante un software, detto crawler, che simula la navigazione umana nelle pagine del portale.

3.3. Trasformazione dei dati

Ai processi di acquisizione dei risultati e della loro pulizia, sono seguiti quelli di trasformazione.

[8] http://coa.inducks.org/
[9] http://www.agenziafarmaco.gov.it/
[10] http://www.amazon.it/

Poiché nel sistema di memorizzazione confluiscono dati provenienti da più fonti e spesso anche quelli derivanti dalla stessa sorgente possono risultare disomogenei, un requisito fondamentale è stata l'integrazione raggiunta ad esempio perseguendo una omogeneità sistematica di tutte le variabili, mediante l'utilizzo delle stesse grandezze fisiche, o anche un ordine prestabilito dei dati recuperati.

Di grande interesse per Accurat era ottenere tra i dati di ciascun prodotto anche, quando presente, l'URL delle relative immagini: in alcuni lavori precedenti infatti, tramite un tool creato dall'azienda, erano state ricavate le tavolozze dei colori di alcune immagini con lo scopo di arricchire i propri progetti con queste informazioni. La volontà è stata quella di riproporre questo approccio anche nei nuovi lavori, al fine di ottenere anche per questi delle informazioni interessanti che potessero essere di ausilio per individuare scelte e preferenze cromatiche delle case produttrici e dei fruitori relativamente a specifici prodotti o alle loro confezioni.

Non di secondaria importanza era il recupero o l'aggiunta di dettagli temporali al fine di poter eseguire delle analisi sul tempo non solo sui dati della stessa sorgente, ma anche confrontando quelli ottenuti da fonti eterogenee.

Anche l'inserimento di queste informazioni ha lo scopo di capire come si evolvano nel tempo sia i dati immediatamente visualizzabili come ad esempio prezzi e dimensioni di uno stesso articolo o uno concorrente, ma anche quelli ottenibili solo a seguito di un'analisi temporale come può essere il trend, anche in base ai gusti dei clienti.

Occorreva però individuare una idonea granularità temporale in base alle esigenze e al contesto, in modo tale che questa andasse da quella grossolana relativamente alla semplice indicazione dell'anno ed eventualmente semestre di riferimento, sino ad arrivare ad una fine come l'indicazione dell'orario di recupero.

I dati archiviati devono quindi poi essere facilmente letti ed elaborati dagli utenti in modo da favorire la produzione di informazioni. Tuttavia alcuni dati presenti in certi prodotti possono essere assenti in altri, quindi, per non appesantire il data store con eccessivi campi `null`, era preferibile utilizzarne uno privo di schema predefinito, potendo così memorizzare all'interno di uno stesso database anche prodotti con un diverso numero di dati recuperati.

3.4. Integrazione tra fonti di dati eterogenee

L'integrazione dei dati coinvolge la combinazione di dati che risiedono in fonti diverse e il fornisce agli utenti una vista unificata di questi dati. Questo processo risulta importante se si desidera che i risultati coinvolgano più indici Elasticsearch.

I problemi relativi alla combinazione tra fonti di dati eterogenee in un'unica interfaccia di interrogazione esistono da qualche tempo; all'inizio degli Anni '80 infatti, gli informatici hanno iniziato la progettazione di sistemi per l'interoperabilità di database eterogenei. [15]

In questo lavoro di Tesi è stato utilizzato un approccio simile al data warehouse, estraendo i dati da fonti eterogenee, trasformandoli e caricandoli in più database NoSQL interrogabili tramite un unico schema di visualizzazione in modo che le informazioni provenienti da fonti diverse diventassero compatibili. Questo approccio offre un'architettura strettamente accoppiata perché i dati si trovano in un unico repository interrogabile, così da ridurre i tempi di risoluzione delle query.

Una parte del lavoro attinente l'integrazione dei dati ha riguardato anche il problema dell'integrazione semantica, ovvero il metodo di risoluzione dei conflitti semantici tra le sorgenti di dati eterogenee. Per esempio, le query che interessano indici Elasticsearch provenienti da fonti con schemi

differenti potrebbero avere concetti e definizioni i cui significati sono diversi. In un indice infatti la voce *prezzo* potrebbe sottintendere il tipo di valuta (Euro) poiché essa è uguale per tutti i prodotti essendo venduti nel mercato italiano. In altri contesti invece può essere più sensato definire un campo *prezzo* che sia sempre un numero, intero o a virgola mobile, e uno relativo alla *valuta* nel caso in cui siano presenti oggetti con valute differenti tra loro (come nel caso di Lira ed Euro).

Così facendo si evitano i problemi di mapping esplicito di cui si è discusso nel Paragrafo 2.1.1, e il processo di definizione di come un documento dovrebbe essere associato al motore di ricerca Elasticsearch, comprese le caratteristiche di cercabilità, come quali campi è possibile cercare, nonché se e quali sono analizzabili lessicalmente, avviene automaticamente ed implicitamente.

Questo tipo di mappatura è definita a livello di indice o tipo e, di default, non è necessaria, dal momento che viene automaticamente creata e registrata quando viene introdotto un nuovo tipo o nuovo campo e vi sono dei valori predefiniti. Solo quando le impostazioni predefinite devono essere sovrascritte, occorre fornire una chiara definizione di mapping.

3.5. Indipendenza dai sistemi sorgente

L’obbiettivo finale di questo lavoro di Tesi è stato la creazione di un’applicazione che operasse sui dati raccolti, manipolandoli e restituendo eventualmente nuova informazione. L'indipendenza dei dati dalla sorgente permette di creare dei servizi che operino su dati provenienti da sorgenti differenti, consentendo all’utente di interagire con gli indici Elasticsearch ad un elevato livello di astrazione, di trascurare i dettagli realizzativi ed

eventualmente aggiungere uno schema esterno in base ad esigenze future dell'utente.

Mentre nei DBMS questa indipendenza è garantita da un'architettura a livelli (livello logico, interno ed esterno), Elasticsearch è organizzato in maniera diversa, a cominciare dalla struttura base detta *"inverted index"* (indice invertito), progettata per consentire ricerche full-text molto veloci. Questo tipo di indice, che consiste in un elenco di tutte le parole uniche che appaiono in qualsiasi documento e, per ogni parola, la lista dei documenti in cui essa viene visualizzata, è risultato particolarmente utile per le query che hanno richiesto il numero di occorrenze di determinate parole all'interno di uno o più indici.

Tuttavia ci sono alcuni problemi con questo tipo di indici, come il fatto di valutare diversa una parola scritta con caratteri minuscoli da una uguale ma scritta con uno o più caratteri maiuscoli, oppure il considerare distinti, ai fine della ricerca, un termine singolare dal suo plurale quando invece l'utente vorrebbe includere anche questo nel risultato. Quindi, se ad esempio in uno degli indici Elasticsearch vi fossero le due istanze "lettore mp3" e "Lettore mp3" e l'utente cercasse il numero di occorrenze della parola "lettore", il risultato prodotto dal conteggio sarebbe di 1. L'unico modo di risolvere questo problema è quello di normalizzare i termini in un formato standard, memorizzando ad esempio tutte le parole con lettere minuscole o tutti i termini al singolare (quest'ultimo caso però richiederebbe un'attenta analisi semantica) [16]. Questo processo di tokenizzazione e di normalizzazione, eseguita da *analizzatori,* è denominata *analisi* e consiste appunto nell'analisi lessicale di un blocco di testo in modo che possa essere utilizzato da un indice inverso. Essa è poi seguita da una normalizzazione dei termini in un formato standard così da migliorarne la "ricercabilità". [17]

4. SOLUZIONE E PROGETTAZIONE DEL SISTEMA

In questo Capitolo verrà descritto il progetto, con le sue problematiche e i procedimenti che hanno portato alla progettazione del sistema.
L'obbiettivo è quello di mostrare come sia possibile recuperare dati da sorgenti differenti, pulirli e memorizzarli in un unico ambiente, con l'intento di manipolarli per ottenere nuova conoscenza.
La trattazione si suddivide nei seguenti paragrafi:

- La sezione 4.1 descrive specificatamente le tre sorgenti di dati utilizzate per il sistema.
- La sezione 4.2 espone del modello utilizzato per i dati recuperati dalle fonti di dati
- La sezione 4.3 e le sottosezioni presentano i processi di estrazione, trasformazione e caricamento in ciascuno dei casi di studio.
- La sezione 4.4 esegue un'analisi sulle specifiche da soddisfare per organizzare e indicizzare i dati.
- La sezione 4.5 descrive il livello che fornisce un accesso semplificato ai dati memorizzati, il DAL, alcuni esempi di query e i risultati prodotti.

4.1. Sorgenti del progetto

Uno dei requisiti del sistema è quello di essere in grado di gestire i dati indipendentemente dalla tipologia della relativa sorgente.
Per ognuna delle sorgenti di dati elencate nel Paragrafo 2.2.1, ovvero strutturate, semistrutturate e non strutturate, sono stati quindi presi in esame

dei casi esemplificativi con lo scopo di mostrare come raggiungere obiettivo preposto esplorando ognuno dei processi.

4.1.1. La sorgente per Pharma

La banca dati ufficiale del sito dell'Agenzia Italiana del Farmaco dà la possibilità di cercare i farmaci per denominazione o nome commerciale. Ogni confezione farmaceutica in commercio in Italia è identificata in modo univoco da un codice detto AIC. L'autorizzazione all'immissione in commercio, l'AIC appunto, è il provvedimento autorizzativo con il quale l'AIFA certifica che un dato medicinale può essere commercializzato in Italia ed è concessa dopo che un gruppo di esperti ha valutato la qualità, la sicurezza e l'efficacia del medicinale. Esso consiste di un codice numerico a 6 cifre che identifica in modo univoco un determinato farmaco. Le singole confezioni di un farmaco sono identificate da ulteriori 3 cifre che seguono il numero AIC l'intero codice a 9 cifre consente di identificare la confezione distinguendola anche in base al numero di compresse/unità, alla percentuale di principio attivo, alla via di somministrazione, ecc. [18].
É inoltre possibile utilizzare le API JSON per recuperare informazioni più o meno generiche come le case farmaceutiche che producono farmaci o il numero di farmaci contenenti lo stesso principio attivo. I dati di questa tipologia sono semi strutturati.

4.1.2. La sorgente per Amazon

Nell'ambito del progetto Amazon era invece necessario recuperare i dati estraendoli direttamente dalle pagine web del celebre sito di e-commerce Amazon.it, nel quale sono disponibili oggetti di praticamente ogni genere e

raggruppati per questo in categorie. Alcune di esse sono caratterizzate da un elenco, aggiornato a cadenza oraria e detto *Bestseller*, dei 100 prodotti più popolari, in base alle vendite. Una delle peculiarità del sito è la possibilità di recensire i prodotti acquistati, utilizzando la scala Likert esprimendo una preferenza che va da uno a cinque. Data l'enorme mole di dati non strutturati da recuperare giornalmente dai prodotti più venduti per ogni categoria, è stata necessaria la creazione di una procedura automatica che ne consentisse la estrazione: per questo motivo è stato creato uno script che eseguisse, autonomamente, il web scraping delle pagine HTML dei prodotti da prendere in considerazione.

4.1.3. La sorgente per Topolino

Accurat era in possesso dei dati estrapolati dal database presente su I.N.D.U.C.K.S., un portale contenente un elenco dettagliato di oltre 160.000 storie dei fumetti Disney e più di 50.000 pubblicazioni, ma che include anche la descrizione di storie, personaggi, creatori, il numero di pagine e molto altro. Il tutto collegato tramite riferimenti incrociati. A ciascuna storia è assegnato un codice univoco, così da rendere più agevole la ricerca e la eventuale ristampa.

Il portale è gestito da un gruppo di circa trenta persone, distribuite in quattro continenti, il cui intento è quello di indicizzare ogni fumetto o rivista Disney presente nel mondo.

È possibile accedere a I.N.D.U.C.K.S. in diversi modi anche se l'interfaccia principale è un motore di ricerca, COA, quotidianamente aggiornato e disponibile in quattordici lingue. Esso però non fa parte del progetto di indicizzazione, ma consente agli utenti di navigare e cercare dati tramite formati testuali difficilmente gestibili dall'utente. [19]

Il dump di questa base di dati e le successive interrogazioni hanno consentito di produrre nuova informazione e questo tipo di dati, derivanti da un database relazionale, sono stati utilizzati come esempio di sorgente strutturata.

4.2. Il modello dei dati

Tra i requisiti del sistema vi è la necessità di poter interagire con sorgenti dati differenti e di gestire correttamente le informazioni che in alcuni casi possono essere tra loro disomogenee anche quando provenienti dalle stesse fonti. Per questo motivo, in fase di progettazione, è stato individuato un modello dei dati comune per tutti i dati, indipendentemente dal metodo di recupero degli stessi.

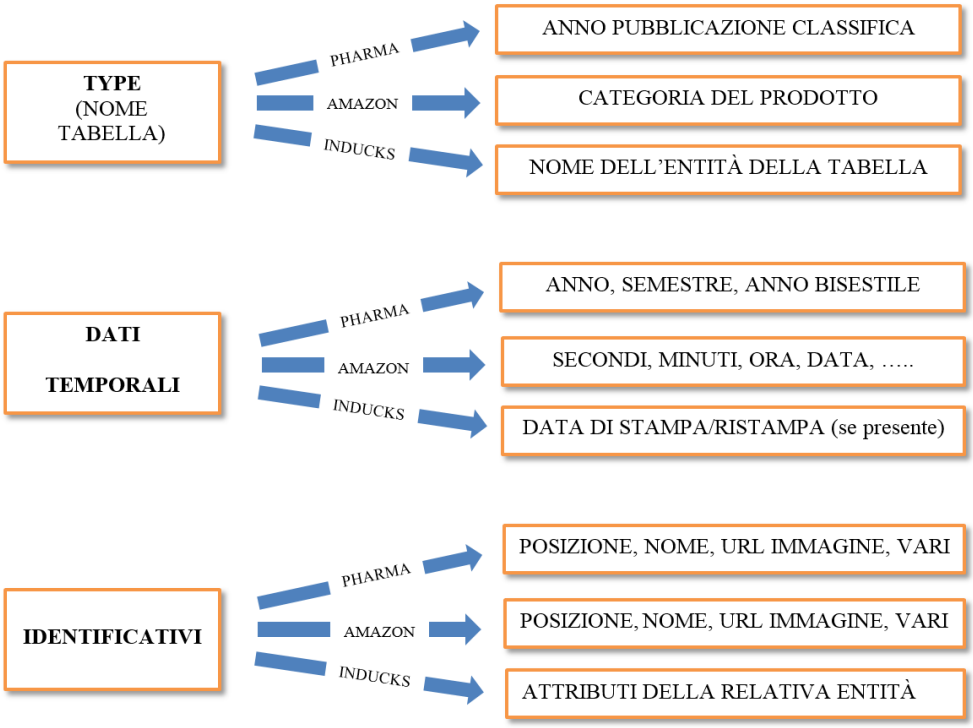

Figura 4: Modello dei dati

Il modello dei dati, la cui rappresentazione è riportata nella Figura 4, è costituito da tre raggruppamenti, a secondo delle informazioni che ognuno di esso rappresenta. Per ogni raggruppamento, in base alle informazioni ricavabili dagli specifici portali web, sono quindi stati indicati quali dati rientrino in esso.

Poiché tutte le informazioni acquisite devono essere memorizzate anche su Elasticsearch, è stato necessario includere il nome del type come prima colonna: questa posizione è stata una scelta progettuale dettata dalla volontà di avere subito presente il nome di riferimento per questa associazione.

Anche la scelta dei nomi è stata ponderata in base a quali dati si stessero recuperando. Le liste dei farmaci da banco più venduti escono a cadenza semestrale quindi ha avuto senso scegliere l'anno come nome della tabella e di avere l'informazione sul semestre all'interno così da poter discriminare le due liste dello stesso anno, anche se al momento la classifica recuperata è solamente una e relativa alla prima metà del 2014.

Per le informazioni dei prodotti presenti su Amazon.it si è scelto di indicare come type il nome della relativa categoria di appartenenza in modo da rispecchiare i raggruppamenti del portale web.

Invece per le tabelle generate dalle query sul database di I.N.D.U.C.K.S., volendo aver sempre presente il tipo di relazione specificata in queste, essa è stata il nome specificato come type.

Essendo l'esecuzione di analisi "intra-sorgente" e "inter-sorgenti" su base temporale una delle necessità di Accurat, sono stati recuperate o aggiunte, dove possibile, delle informazioni per raggiungere questo scopo.

In particolare, tra i dati recuperati per il progetto Pharma non erano presenti quelli relativi al tempo e per questo motivo sono stati aggiunti come arricchimento. In fase di progettazione si è ritenuto sensato aggiungere semplicemente il semestre, l'anno e se questo sia bisestile o meno, una granularità differente sarebbe stata poco utile agli scopi.

Dal momento invece che l'acquisizione delle informazioni da Amazon.it è avvenuta a cadenza giornaliera, si è scelto in questo caso una granularità decisamente più fine. Nello specifico sono stati aggiunti, relativamente ad ogni recupero:

- il secondo
- il minuto
- l'ora
- giorno
- mese
- anno
- giorno della settimana
- giorno dell'anno
- settimana dell'anno
- trimestre
- quadrimestre
- semestre
- l'indicazione di anno bisestile
- l'indicazione se sia un giorno del fine settimana o meno
- la stagione

Per alcuni file relativi alle storie Disney, a causa della relazione in essi descritta, non è stato sempre possibile avere dei dati sul tempo. Tuttavia in una tabella in particolare, quella sui numeri dei fumetti, è presente la loro data di stampa o ristampa che può quindi essere considerata come riferimento temporale.

Infine vi sono le informazioni vere e proprie recuperate dai portali. Un requisito fondamentale per il sistema è stato l'integrazione, da raggiungere ad esempio perseguendo una omogeneità sistematica di tutte le variabili,

mediante l'utilizzo delle stesse grandezze fisiche, o anche in alcuni casi di un ordine prestabilito dei dati acquisiti.
Recuperando delle classifiche sia nel progetto Pharma che in quello Amazon, vi sono degli identificativi comuni come la posizione e il nome, oltre alla URL dell'immagine del prodotto o della sua confezione; gli altri sono legati specificatamente alla sorgente di dati e saranno approfonditi nel Paragrafo 4.3.
Gli identificativi delle tabelle del progetto I.N.D.U.C.K.S. sono gli attributi della relativa entità e possono essere ad esempio i nomi dei personaggi, nomi e codici delle storie, o le occorrenze di determinate parole.

4.3. ETL (Extraction, Transformation, Loading)

È stato necessario caricare i dati su Elasticsearch in modo che possa servire al suo scopo di facilitare l'analisi di business, identificando esigenze e soluzioni ai problemi. I dati presenti nelle tre sorgenti prese in considerazione sono stati quindi estratti e copiati in documenti allo scopo di integrare, riorganizzare e consolidare grandi volumi di dati, fornendo in tal modo un nuovo sistema di organizzazione assimilabile a un data warehouse.
Il processo di estrazione dei dati dai sistemi sorgente e di caricamento in un data warehouse è comunemente chiamato ETL (estrazione, trasformazione, caricamento) e si riferisce ad un ampio procedimento, non a tre passi ben definiti, anche se l'acronimo non evidenzia la fase di trasporto e il legame fra i tre livelli.

4.3.1. Il processo ETL per Pharma

Periodicamente l'Agenzia Italiana del Farmaco mette a disposizione nel proprio portale la lista dei 50 farmaci da banco più venduti nel semestre precedente.

Utilizzando il codice AIC e interrogando correttamente la banca dati farmaci ufficiale dell'AIFA tramite le API JSON, sono state recuperate alcune caratteristiche che appartengono proprio a quello specifico farmaco, ovvero:

- il codice della famiglia del farmaco;
- la famiglia del farmaco;
- lo stato;
- il principio attivo;
- il nome e il codice della ditta che produce il medicinale;
- il numero di confezioni in commercio;
- il numero dei farmaci attivi;
- il tipo di procedura;
- l'ATC (il sistema di classificazione Anatomico Terapeutico e Chimico);
- il numero di farmaci attivi con stesso atc e prodotti dalla stessa azienda;
- il numero di farmaci attivi aventi lo stesso atc;
- il numero di farmaci attivi prodotti dalla stessa ditta;
- il numero di farmaci attivi avente lo stesso principio attivo.

In aggiunta a questi dati, per distinguere questa classifica da quelle che possono essere recuperate in futuro, sono state aggiunte anche le indicazioni del semestre e dell'anno cui si fa riferimento, e se si tratta di un anno bisestile. Questo può consentire anche di verificare ad esempio l'andamento complessivo nel tempo delle vendite di certi medicinali o di eseguire confronti con prodotti venduti nello stesso periodo ma presenti in una fonte di dati diversa.

Inizialmente sono state necessarie alcune prove manuali per capire come utilizzare correttamente le API, in modo da avere la certezza che i risultati dalle query fossero realmente esatti e coerenti.
Questi test sono stati effettuati su un sottoinsieme di 20 sui 50 farmaci da banco più venduti nel secondo semestre del 2013, creando un foglio di calcolo avente, per ogni codice del medicinale, i risultati delle interrogazioni inseriti manualmente in celle distinte.
Avendo quindi verificato la corretta restituzione dei dati coerentemente alla query creata per ottenerli, il passo successivo è stato l'individuare la procedura che consentisse di ottenere queste informazioni relativamente al periodo richiesto e meccanicamente tramite uno script.
L'input che ha consentito l'analisi per l'acquisizione è stato un file JSON contenente la sequenza ordinata degli AIC dei cinquanta più venduti farmaci da banco o di automedicazione e non soggetti a prescrizione medica, separati da virgole e racchiusi in parentesi quadre.
Il processo di estrazione è stato agevolato dalla presenza di API per l'interrogazione del database del portale dell'AIFA mentre la memorizzazione dei dati è avvenuta inizialmente solo su un file CSV.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	12745093	12745	TACHIPIRINA	A	Paracetamolo	AZIENDE CHIMICI	219	340	21	N	N02BE01	56	356	56	356
2	590051	590	RINAZINA	A	Nafazolina	GLAXOSMITHKLI	1136	73	2	N	R01AA08	2	4	2	16
3	13046040	13046	ENTEROGERMINA	A	Microorganismi an	SANOFI S.P.A.	8055	562	10	N	A07FA	10	28	10	28
4	42028011	42028	OKITASK	A	Ketoprofene, asso	DOMPE' FARMAC	28	32	2	N	M01AE53	2	13	2	13
5	34548040	34548	VOLTAREN EMULGEL	A	Diclofenac	NOVARTIS FARM	114	177	8	N	M02AA15	8	61	17	228
6	12745016	12745	TACHIPIRINA	A	Paracetamolo	AZIENDE CHIMICI	219	340	21	N	N02BE01	56	356	56	356
7	29651066	29651	GLICEROLO CARLO EI	A	Glicerolo	CARLO ERBA O.T.	7095	22	5	N	A06AG04	2	28	5	99
8	29651039	29651	GLICEROLO CARLO EI	A	Glicerolo	CARLO ERBA O.T.	7095	22	5	N	A06AX01	3	71	5	99
9	25669019	25669	MOMENT	A	Ibuprofene	AZIENDE CHIMICI	219	340	18	N	M01AE01	30	294	30	397
10	13046038	13046	ENTEROGERMINA	A	Microorganismi an	SANOFI S.P.A.	8055	562	10	N	A07FA	10	28	10	28

Figura 5: Selezione dei primi 10 farmaci e relative informazioni

L'output così prodotto, utile per il successivo lavoro di visualizzazione di Accurat, era però troppo semplice per la memorizzazione in una tabella (type) di Elasticsearch poiché mancante della relativa intestazione. In un

secondo momento è stato infatti deciso di fare in modo che la compilazione del documento Elasticsearch non avvenisse manualmente ma automaticamente eseguendo lo script che è stato quindi opportunamente modificato includendo i campi e nuove colonne indicanti l'anno di riferimento, il semestre e la posizione in classifica di ciascun farmaco, la quale era prima indicata dalla riga del file CSV.
Con la resa automatica del processo di caricamento delle informazioni sul database NoSQL, si è inoltre ritenuta necessaria la contemporanea creazione di un file di log che registrasse eventuali segnalazioni di errore.

4.3.2. Il processo ETL per Amazon

Il sito italiano di e-commerce Amazon.it consente l'acquisto on-line di migliaia di prodotti. Il portale suddivide i prodotti in diverse categorie, alcune delle quali presentano la sezione *Bestseller*, ovvero la classifica dei cento prodotti più popolari, in base alle vendite, che appartengono alla categoria in esame.
Questo elenco, a sua volta, si distribuisce su cinque pagine, ognuna delle quali presenta venti prodotti caratterizzati da alcune informazioni come la posizione in classifica, i primi quaranta caratteri del nome dell'articolo (spazi compresi), una piccola immagine raffigurante l'oggetto, le valutazioni ricevute e il prezzo.
A loro volta, ciascuno di questi raggruppamenti risulta essere la radice di un albero di sottocategorie di *Bestseller*.

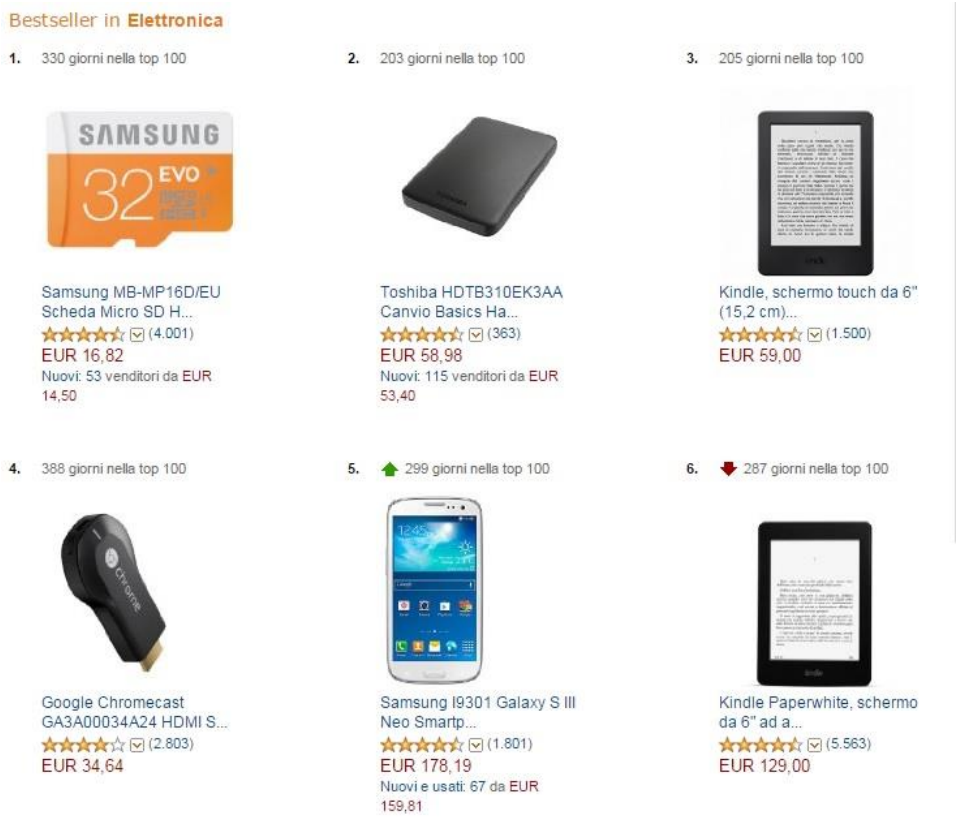

Figura 6: Primo particolare della pagina web dei Bestseller per la categoria Elettronica

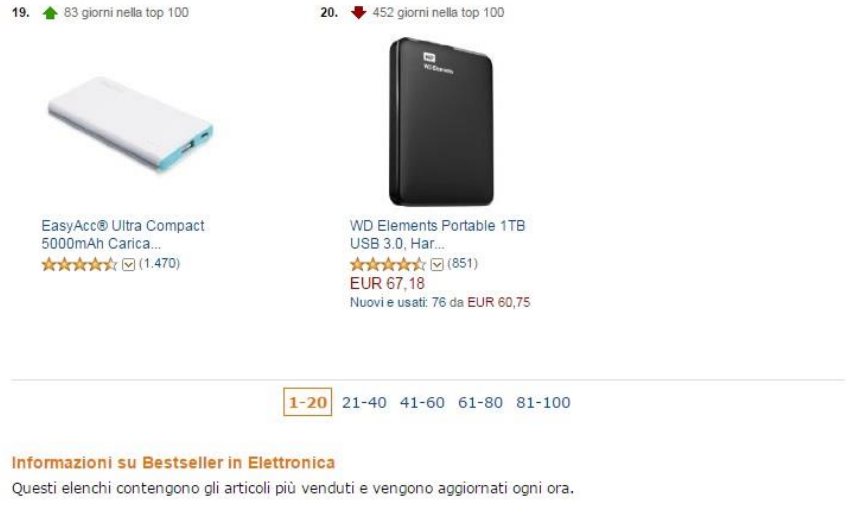

Figura 7: Secondo particolare della pagina web dei Bestseller per la categoria Elettronica

Figura 8: Esempio di albero di sottocategorie di Bestseller

La prima fase è stata quella relativa alla granularità temporale da scegliere: gli aggiornamenti dei *Bestseller* avvengono a cadenza oraria, ma recuperare i dati con quella frequenza sarebbe stato valutato poco utile per i propri scopi poiché, più che le variazioni in classifica nell'arco di un'ora, sarebbero state significative quelle relative a una intera giornata. Quindi, partendo dall'orario di recupero, si va via via aumentando la granularità sino a specificare anche la stagione di recupero, secondo il modello dei dati.

La fase successiva è stata quella di studio del sito: per ognuna delle categorie infatti, è stata posta attenzione su quali fossero le informazioni specifiche di ciascun prodotto in classifica. Per questo motivo, è stato creato un foglio di calcolo avente una colonna indicante tutti i raggruppamenti esistenti; l'intestazione delle altre colonne è derivata da ciascuna delle caratteristiche

presentatesi durante l'analisi dei prodotti e, in corrispondenza di una categoria, si è posta una "x" nel campo quando questa informazione è presente. Questo procedimento ha mostrato come alcune di queste informazioni fossero sempre presenti indipendentemente dalla categoria, altre invece erano quasi sempre presenti, mentre altre ancora erano caratterizzanti di uno specifico raggruppamento. Per questo motivo è stato ritenuto opportuno, ai fini comprensivi ma anche per avere un recupero dei dati che fosse modulare, suddividere i dati dei prodotti in tre tipi:

- *Gialli*, comuni a tutte le categorie.

  Inizialmente in questo raggruppamento rientravano, in ordine:

  - i dati temporali relativi al giorno di recupero, come l'ora, la data, il giorno della settimana e dell'anno;
  - l'URL della pagina del prodotto;
  - il nome dell'articolo nella pagina dei *Bestseller*;
  - la posizione in classifica;
  - l'asin (acronimo per Amazon Standard Identification Number, ovvero il numero unico identificativo usato dall'azienda produttrice e da Amazon per riferirsi a un prodotto);
  - il nome completo dell'articolo;
  - l'URL dell'immagine presente nella pagina del prodotto;
  - il numero totale di stelline, cioè di commenti lasciati nella pagina dell'oggetto in vendita);
  - il numero di preferenze che ha ricevuto ciascuno dei possibili voti attribuibili, da 1 a 5;
  - il prezzo di vendita da parte di Amazon e quello di partenza;
  - lo sconto in Euro e quello in percentuale.

  Successivamente, è stato valutato opportuno arricchire questi dati aggiungendo:

- il numero di rivenditori dell'oggetto nuovo, usato, ricondizionato e i relativi prezzi;
- il venditore principale;
- il massimo, il minimo e il medio livello di profondità dell'albero delle categorie del prodotto dato che un prodotto può appartenere a diverse categorie ed essere quindi raggiungibile tramite percorsi diversi interni al sito.

nome	URL PAGINA	TITOLO LISTA	POSIZIONE	ASIN	TITOLO	URL IMMAG
CASA E CUCINA	x	x	x	x	x	x
ELETTRONICA	x	x	x	x	x	x
GIOCHI E GIOCATTOLI	x	x	x	x	x	x
LIBRI	x	x	x	x	x	x
PRIMA INFANZIA	x	x	x	x	x	x
SPORT E TEMPO LIBERO	x	x	x	x	x	x
VIDEOGIOCHI	x	x	x	x	x	x
INFORMATICA	x	x	x	x	x	x
AUTO E MOTO	x	x	x	x	x	x
CANCELLERIA E PRODOTTI PER UFFICI	x	x	x	x	x	x
CURA DELLA PERSONA	x	x	x	x	x	x
FAI DA TE	x	x	x	x	x	x
FILM E TV	x	x	x	x	x	x
GIARDINO E GIARDINAGGIO	x	x	x	x	x	x
GIOIELLI	x	x	x	x	x	x
OROLOGI	x	x	x	x	x	x
SOFTWARE	x	x	x	x	x	x

Figura 9: Vista di alcuni dei dati "Gialli" per tutte le categorie aventi i Bestseller

- *Verdi*, presenti in quasi tutte le categorie, sono informazioni concettualmente simili tra di loro ma in alcuni casi rappresentate utilizzando identificativi diversi. L'aggiunta di queste informazioni è avvenuta durante una successiva fase di studio.

  Ad esse appartengono, in ordine:
  - la marca, coincidente con l'editore nel caso dei libri;
  - l'andamento in classifica rispetto l'aggiornamento precedente, ovvero se la posizione è in discesa, in ascesa o costante;
  - il numero di giorni presenti nella *top 100*.

La Figura 9, mostra ad esempio come il trend e il codice del produttore siano indicati solo nelle categorie Elettronica e Libri.

nome	MARCA	TREND	NUMERO GIORNI TOP 100
CASA E CUCINA	x		
ELETTRONICA	x	x	x
GIOCHI E GIOCATTOLI	x		
LIBRI	editore	x	x
PRIMA INFANZIA	x		
SPORT E TEMPO LIBERO	x		
VIDEOGIOCHI	x		
INFORMATICA	x		
AUTO E MOTO	x		
CANCELLERIA E PRODOTTI PER UFFICI	x		
CURA DELLA PERSONA	x		
FAI DA TE	x		
FILM E TV			
GIARDINO E GIARDINAGGIO	x		
GIOIELLI	x		
OROLOGI	x		
SOFTWARE	x		
VALIGERIA	x		

Figura 10: Dati "Verdi" per tutte le categorie aventi i Bestseller

- *Viola*, ovvero le informazioni specifiche per ciascuna categoria.
  In questo caso ovviamente l'elenco è molto più lungo e verrà proposto specificatamente per ogni categoria sempre all'interno di questo Paragrafo. Qui viene comunque riportata un'immagine che funge da vista della tabella.

nome	DIMENSIONI	PESO	PESO DI SPED	NUMERO PEZZI	DIM MEMO FISICA	DIM RAM
CASA E CUCINA	x	x	x			
ELETTRONICA	x	x	x		x	x
GIOCHI E GIOCATTOLI	x	x	x	x		
LIBRI			x			
PRIMA INFANZIA	x	x	x			
SPORT E TEMPO LIBERO	x	x	x			
VIDEOGIOCHI	x	x				
INFORMATICA	x	x	x		x	x
AUTO E MOTO	x	x	x			
CANCELLERIA E PRODOTTI PER UFFICI	x			x		
CURA DELLA PERSONA	x	x	x			
FAI DA TE	x	x	x	x		
FILM E TV				x		
GIARDINO E GIARDINAGGIO			x			
GIOIELLI		x	x			
OROLOGI	x	x	x			
SOFTWARE				x		
VALIGERIA	x	x	x			

Figura 11: Vista di alcuni dei dati "Viola" per tutte le categorie aventi i Bestseller

Ai fini di questo lavoro di Tesi, tra tutte le categorie aventi i *Bestseller*, è stato deciso di prendere in considerazione solo un sottoinsieme di sette, ovvero quelle ritenute più interessanti: Casa e cucina, Elettronica, Giochi e giocattoli, Libri, Prima infanzia, Sport e tempo libero, Videogiochi.

Successivamente è stato realizzato un software che analizza, in maniera automatizzata e metodica, una o più pagine web dei prodotti, così da recuperare le informazioni di specifici articoli. La sua progettazione si è basata molto su quella del primo crawler di raccolta dati e segue i ragionamenti fatti per esso, ma invece di un recupero giornaliero dei dati avendo come punto di partenza l'URL dei *Bestseller*, vengono memorizzati i dati dei prodotti di cui si specifica il relativo l'indirizzo web della pagina del portale Amazon.it

L'input di questo crawler è stato un file JSON contenente un array di oggetti, ovvero si sequenze di coppie chiave-valore separate da virgole e racchiuse tra parentesi graffe. Per ciascuna categoria, le chiavi inserite nel documento sono state:

- *id*: il valore ad esso relativo ha lo scopo di identificare la categoria specifica;

- *type*: specifica se si tratta di una categoria o di una sottocategoria;
- *analyze*: flag di tipo true/false per eseguire o meno l'analisi di una categoria;
- *name*: nome della categoria;
- *normalizedname*: è la parola che rappresenta la normalizzazione del nome della categoria;
- *urlPath*: URL della categoria
- *pages*: numero di pagine in cui è suddivisa la lista dei *Bestseller*.

Partendo dalle URL inserite nel file JSON, il compito del crawler è stato quello di eseguire lo scraping del codice HTML di una categoria, memorizzando le informazioni già qui presenti dei prodotti, tra cui la relativa URL utilizzata per un secondo scraping che permettesse di arricchire queste informazioni recuperando i restanti dati in base alla categoria cui l'oggetto appartiene. Queste operazioni avvengono in maniera ciclica per ognuna delle cinque pagine di una categoria, per ogni prodotto classificato in questa *top 100* e per tutti i raggruppamenti indicati nel file di input aventi il flag analyze impostato a "true".

I file di output creati dal software sono stati invece molteplici. È stato ritenuto utile avere la possibilità di memorizzare le pagine HTML dei *Bestseller* e ognuna delle pagine prodotto così da poter eventualmente in futuro recuperare anche manualmente i dati degli articoli più venduti.

Ma la parte più importante e significativa è stata il contestuale salvataggio delle informazioni recuperate su due tipologie di file CSV. Dopo aver analizzato completamente tutti i prodotti di una categoria, viene creato un primo file contenente i dati "gialli", "verdi" e "viola" di ogni articolo classificatosi tra i cento più venduti. Nel secondo invece, le informazioni di ogni prodotto presente tra i *Bestseller* sono salvate in tante righe quanti sono i percorsi di categorizzazione tramite cui è possibile raggiungere l'oggetto in

esame. In questo caso le informazioni memorizzate sono diverse e sono quelle di tipo temporale, la posizione, l'URL e il nome dell'oggetto, il contatore dei percorsi di categorizzazione, la posizione in classifica rispetto lo specifico percorso che viene memorizzato inserendo le categorie che lo compongono in celle distinte.

posizione	url-pagina	titolo	codice-asi	cont	posiz	Liv0	Liv1	Liv2
1	http://ww	Sottomissione	8,85E+09	2	1	Libri		
1	http://ww	Sottomissione	8,85E+09	2	3	Libri	Narrativa	Narrativa contemporanea
2	http://ww	Numero zero	8,85E+09	2	2	Libri		
2	http://ww	Numero zero	8,85E+09	2	5	Libri	Narrativa	Narrativa contemporanea
3	http://ww	American sniper	8,8E+09	4	3	Libri		
3	http://ww	American sniper	8,8E+09	4	1	Libri	Politica	Scienze politiche
3	http://ww	American sniper	8,8E+09	4	1	Libri	Biografie	Biografie e autobiografie
3	http://ww	American sniper	8,8E+09	4	3	Libri	Società e sc	

Figura 12: Vista del file CSV contenente l'albero di categorizzazione dei Libri più venduti

La Figura 11 mostra graficamente parte della struttura del CSV appena descritto, con l'intento di far vedere come ad esempio il libro "American sniper" avente contatore uguale a 4 sia riportato su quattro righe distinte; in ognuna di esse viene poi mostrato, come detto, uno dei livelli dell'albero di categorizzazione e, per ciascun percorso, la posizione in classifica in cui il libro si è attestato.

Almeno inizialmente però, l'idea era stata quella di non includere il percorso della *top 100*, che era quindi da considerarsi implicito. Questo comportava l'assenza del prodotto nel file qualora quella fosse l'unica serie di categorie che consentisse di raggiungere l'oggetto. Nel corso delle varie esecuzioni del software si è poi ritenuto utile esplicitare la presenza di questo percorso, così da avere ben visibili tutti i rami dell'albero delle categorizzazioni.

Per quanto riguarda la parte del crawler che ha consentito di creare i file principali, quelli avente la classifica dei prodotti più venduti per ogni categoria e le relative informazioni, è stato opportuno eseguire un'attenta

fase di studio delle pagine web e del relativo codice HTML, individuando i primi attributi a cui fanno riferimento le informazioni da recuperare. Tuttavia ad esempio, i dati "gialli" come l'URL dell'immagine o il prezzo di vendita di un prodotto non sempre utilizzano lo stesso attributo variando la categoria; oppure l'attributo associato a un dato "viola" di un prodotto potrebbe essere diverso da un altro articolo appartenente alla stessa categoria.

Tuttavia queste, benché non rare, sono delle eccezioni e quindi in certi casi il rilevamento del giusto attributo è avvenuto a posteriori, dopo aver verificato la presenza del problema specifico. Anche perché in certi casi, nonostante l'analisi iniziale, è stata valutata successivamente la possibilità o necessità di aggiungere uno o più campi alla tabella contenente i dati da recuperare. Inoltre, l'aggiornamento orario delle classifiche comporta il possibile inserimento in graduatoria di prodotti in pagine HTML non ancora analizzate che quindi potrebbero avere anche una struttura leggermente diversa, o molto diversa come nel caso dei kindle, cosa che chiaramente ha portato a diverse modifiche del crawler.

Tuttavia, la struttura modulare di recupero dati del software ha consentito di avere almeno come risultato iniziale i file CSV con le informazioni "gialle" dei cento prodotti di ogni categoria, anche se i documenti dei primi giorni di recupero presentavano alcune mancanze o scorrettezze, causate proprio dal non aver preso in considerazione ciascuno degli attributi una determinata caratteristica comune a tutti i prodotti o tutte le diverse strutture delle pagine HTML.

Con la ulteriore decisione di recuperare anche la marca (o l'editore nel caso dei libri), il trend e il numero di giorni in cui l'articolo è presente nella *top 100*, anche questi dati, ossia i "verdi", sono stati acquisiti nelle successive esecuzioni del crawler contestualmente alle informazioni generiche del relativo prodotto. Anche in questo caso l'analisi iniziale relativa agli attributi

a cui riferirsi per recuperare le informazioni non è stata sufficiente ma è stata perfezionata di volta in volta tramite successive esecuzioni del software.
A questo punto, una categoria per volta, sono stati individuati gli attributi utili per recuperare le informazioni "viola".
Si è quindi proceduto con il recupero dell'autore, dell'isbn-13, delle pagine, della collana, dell'edizione, della data di pubblicazione, del peso di spedizione e della relativa unità di misura (in celle separate), e del numero di versioni in cui viene venduto il libro.
L'ordine restante di completamento delle informazioni è stato invece quasi casuale, cercando di sfruttare il più possibile la metodologia di recupero di uno o più dati "viola" quando questi si sono presentati in categorie diverse.

nome	DIM X	DIM Y	DIM Z	UNIT DIM	PESO	UNIT PESO	PESO SPED	UNIT PESO S.
CASA E CUCINA	x	x	x	x	x	x	x	x
ELETTRONICA	x	x	x	x			x	x
GIOCHI E GIOCATTOLI	x	x	x	x	x	x	x	x
LIBRI							x	x
PRIMA INFANZIA	x	x	x	x	x	x	x	x
SPORT E TEMPO LIBERO	x	x	x	x	x	x	x	x
VIDEOGIOCHI	x	x	x	x	x	x		
INFORMATICA	x	x	x	x	x	x	x	x
AUTO E MOTO	x	x	x	x	x	x	x	x
CANCELLERIA E PRODOTTI PER UFFICIO	x	x	x	x				
CURA DELLA PERSONA	x	x	x	x	x	x	x	x
FAI DA TE	x	x	x	x	x	x	x	x
FILM E TV								
GIARDINO E GIARDINAGGIO							x	x
GIOIELLI	x	x			x	x	x	x
OROLOGI	x	x	x	x	x	x	x	x
SOFTWARE								
VALIGERIA	x	x	x	x	x	x	x	x

Figura 13: Prima vista dei dati "viola" di tutti i Bestseller. Sono evidenziati solo quelli relativi alle categorie analizzate

La Figura 12 mostra ad esempio come le dimensioni dell'oggetto in vendita siano caratteristiche presenti, eccezion fatta per i Libri, in tutti i raggruppamenti selezionati, sebbene anche in questo caso non sempre l'attributo HTML di riferimento sia stato comune a tutti i prodotti.

Inoltre, ognuno dei campi “Dimensione memoria fisica”, “Dimensione RAM” e “Dimensione schermo” è stato successivamente suddiviso in due, uno indicante il valore numerico relativo e l’altro indicante la dimensione fisica ad esso associato.

nome	NUMERO PEZZI	DIM MEMO FISICA	DIM RAM	DIM SCHERMO	LINGUA
CASA E CUCINA					
ELETTRONICA		x	x	x	
GIOCHI E GIOCATTOLI	x				x
LIBRI					
PRIMA INFANZIA					
SPORT E TEMPO LIBERO					
VIDEOGIOCHI					x

Figura 14: Seconda vista dei dati "viola" per i Bestseller analizzati

Nelle Figure 13, 14, 15 e 16 si vede come i campi, ad eccezione di “lingua”, “batterie necessarie”, “tipo materiale” e “anno/data”, siano totalmente contraddistintivi di una categoria: il CSV prodotto per ogni categoria avrà quindi come intestazione di ogni colonna: tutti i dati “gialli”, i cui rispettivi valori sono sempre presenti a meno di mancanze in determinati prodotti; tutti i dati “verdi”, i cui rispettivi valori saranno presenti solo nelle categorie specifiche; tutti i dati “viola” caratterizzanti una determinata categoria, anche se in questo caso non sono inusuali i campi bianchi dovuti al fatto che nella pagina del prodotto non sia riportata quella particolare caratteristica.

nome	SOTTOTITOLI	ETA'	NUM. GIOCATORI	BATTERIE NECES	BATTERIE INCL
CASA E CUCINA					
ELETTRONICA					
GIOCHI E GIOCATTOLI		x	x	x	x
LIBRI					
PRIMA INFANZIA				x	
SPORT E TEMPO LIBERO					
VIDEOGIOCHI	x				

Figura 15: Terza vista dei dati "viola" per i Bestseller analizzati

nome	TIPO MATERIALE	ANNO/DATA	USO IN AEREO	LAVABILE LAVASTOVIGLIE
CASA E CUCINA				
ELETTRONICA		x		
GIOCHI E GIOCATTOLI	x			
LIBRI				
PRIMA INFANZIA	x		x	x
SPORT E TEMPO LIBERO	x			
VIDEOGIOCHI		x		

Figura 16: Quarta vista dei dati "viola" per i Bestseller analizzati

nome	GENERE	PIATTAFORMA	EDIZIONE	MANUALE	COLORE
CASA E CUCINA					
ELETTRONICA					x
GIOCHI E GIOCATTOLI					
LIBRI			x		
PRIMA INFANZIA	x				
SPORT E TEMPO LIBERO					x
VIDEOGIOCHI		x		x	

Figura 17: Quinta vista dei dati "viola" per i Bestseller analizzati

L'ultima modifica riguardante i dati "viola" rispetto le specifiche iniziali è stata quella relativa al campo "Colore". Poiché infatti alcuni oggetti elettronici presentano una doppia colorazione, in questa categoria il campo unico è stato sostituito dai campi "Colore primario" e "Colore secondario".
In aggiunta, prima di essere memorizzati nei rispettivi file CSV, i dati recuperati, hanno subito in alcuni casi delle normalizzazioni e delle trasformazioni.
Prima di tutto, poiché il separatore utilizzato nel CSV è stato il segno di interpunzione punto e virgola, è stato necessario togliere questo simbolo da qualunque informazione recuperata. Esso, ove presente, è stato sostituito da una virgola.
Per omogeneità è stato poi deciso di trasformare tutte le parole scritte con caratteri maiuscoli con parole aventi caratteri minuscoli e per lo stesso motivo sono stati "puliti" alcuni dati come i colori, sottotitoli o le lingue. Tuttavia, nonostante questa normalizzazione, alcuni campi di testo

contengono parole non completamente attinenti e non predeterminabili: ad esempio tramite l'attributo HTML di riferimento per il colore sono state estrapolate delle stringhe contenenti, oltre al colore stesso, anche parole e numeri non pertinenti come 100%, 1400mah, oppure 15.5w. Oppure alcuni valori di una delle caratteristiche erano indicate in lingua italiana, inglese o entrambe, a volte abbreviate o con errori di scrittura (ad esempio: sliver invece di silver). In questi casi sarebbe stata necessaria un'analisi semantica, che esula dall'obbiettivo di questo lavoro di Tesi ed è stato quindi deciso di lasciare in questi casi tali informazioni.
Non banale è stata anche la normalizzazione delle informazioni relativa al "numero di giocatori" e all' "età consigliata dal produttore" a causa della forte disomogeneità dei dati riportati nelle pagine dei prodotti. Tra i vari casi ad esempio, la stringa relativa all'età poteva contenere, oltre al numero, anche le parole mese/mesi o anno/anni (o entrambi): in questo caso, tramite una normalizzazione, si è fatto in modo che i numeri aventi associati alla parola mese/mesi fossero convertiti nell'equivalente annuo, lasciando ovviamente inalterato il numero associato alla parola anno/anni.

Di pari passo al percorso di miglioramento del crawler e alla sua esecuzione manuale con il salvataggio in locale dei CSV, è stata automatizzata proprio quest'ultima procedura per memorizzare su un server i file creati dal software. Per fare questo è stato fatto uso di FileZilla, una cross-piattaforma FTP, FTPS e SFTP veloce, affidabile e facilmente utilizzabile grazie a una intuitiva interfaccia utente grafica, e di un semplice script di shell per eseguire automaticamente il software a un determinato orario. Benché infatti gli aggiornamenti delle *top 100* avvengano ogni ora e che quindi vi siano ventiquattro potenziali cambiamenti per categoria nell'arco di una giornata, non sempre le classifiche subiscono delle modifiche rilevanti che riguardino la posizione dei prodotti oppure il loro ingresso nella (o uscita dalla) lista dei

Bestseller. È per questo motivo che è stato deciso un unico orario di raccolta automatica dei dati, ossia le 23.30 italiane, quando probabilmente il traffico internet all'interno del sito è più basso che in altri momenti della giornata.
Terminata la parte del software per recuperare e memorizzare sia le pagine HTML sia i dati su file CSV, e verificatane la correttezza del funzionamento, prima di lasciare che la sua esecuzione non avvenga più manualmente ma automaticamente, è stato fatto in modo che, in caso di uno o più errori, il software memorizzi le informazioni relative ai problemi intercorsi in un file di log consultabile in caso di necessità.
L'ultima modifica relativa al crawler è stata quella di fare in modo che, anche in questo caso, l'output presente nei CSV venisse memorizzato su Elasticsearch. Due sono stati gli indici creati, uno per ogni tipologia di file: *amazon_ranks*, dove sono presenti, per ogni giornata di raccolta dati, tutte le informazioni relative alle classifiche, e *amazon_tree*, nel quale troviamo gli alberi di categorizzazione dei prodotti.
Una volta resa autonoma l'estrazione e il salvataggio dei dati, sia in formato comme-separated values che sul database NoSQL, è stata oppurtuna la modifica dei precedenti file CSV che, a causa del completamento progressivo del crawler, contenevano sempre almeno i dati "gialli" ma erano mancanti di alcuni o tutti i restanti.
Sfruttando l'URL di ciascuno dei prodotti nei file è stato possibile completarli, andando a cercare i valori dei campi mancanti: ovviamente anche in questo caso la ricerca dei dati è avvenuta in maniera automatica mediante un altro crawler, avente un funzionamento analogo al primo.
L'input di questo secondo software è stato un file JSON contenente un array di oggetti, uno per ogni prodotto di cui si è voluto recuperare i dati. Le chiavi inserite nel documento sono state:

- *categorynormalizedname*: è la parola che rappresenta la normalizzazione del nome della categoria;

- *analyze*: flag di tipo true/false per eseguire o meno l'analisi di una categoria;
- *categoryname*: nome della categoria;
- *url*: URL del prodotto.

Diversamente dal caso di prima, la ricerca non avviene partendo dalle pagine dei *Bestseller* ma direttamente dalle pagine di cui sono stati indicati gli indirizzi web.

La procedura di raccolta è invece analoga e per ogni articolo presente nel file JSON.

L'output consiste di un unico file CSV che riporta come intestazione delle sue colonne prima tutti i "gialli", poi tutti i "verdi" e i "viola", specificando prima di ogni proprietà il nome della categoria cui essa appartiene; i dati dei prodotti sono quindi memorizzati nella cella corrispondente in base al raggruppamento cui l'articolo fa parte.

Una successiva manipolazione manuale dei dati così raccolti e salvati ha permesso di completare gli iniziali e incompleti file creati dal primo crawler. Anche questi documenti sono stati inseriti nel server così che le informazioni in possesso potessero estendersi a un intervallo di tempo maggiore; per lo stesso motivo era necessario importare questi file pure su Elasticsearch.

È stato quindi sviluppato un software, che opportunamente modificato può essere riutilizzato per qualunque file di tipo comma-separated values, il quale analizza il file specificato dall'utente, memorizzandone tutte le righe nell'indice specificato. I dati vengono salvati di default come stringhe, quindi quando questi erano numeri da utilizzare per successive aggregazioni è stato necessario esplicitarli come tali. È questo il motivo per cui eventuali riutilizzi del programma per altri file necessitano di un controllo analogo.

4.3.3. Il processo ETL per I.N.D.U.C.K.S.

Sui dati raccolti dal database I.N.D.U.C.K.S., è stata effettuata una serie di query così da ottenere le pubblicazioni, i numeri, le date di stampa, i nomi dei personaggi, gli autori, le storie, le apparizioni, gli eroi, i codici delle differenti versioni e le occorrenze. Questi relativamente alle pubblicazioni italiane. I risultati ottenuti sono stati memorizzati in quattro file, ognuno contenente molteplici fogli di calcolo. Il primo di questo documento, *Titoli_personaggi*, contiene un foglio per ognuno dei personaggi e all'interno sono presenti le occorrenze di alcune parole chiave come "mistero", "caso" e "oro". Il secondo archivio, *Co-occorrenze*, contiene alcuni fogli relativi alle co-occorrenze tra due personaggi Disney e alcune informazioni correlate come il codice della storia o dei personaggi. Il file *Titoli_completo* mostra le occorrenze di alcune parole chiave utilizzando dei filtri come ad esempio l'esclusione dei nomi dei personaggi dalle keyword e delle parole con ricorrenza minore di 10. L'ultimo documento, Inducks, contiene invece le risposte alle query succitate, specificando informazioni come date e prezzi dei numeri.

Ulteriore scopo di questo lavoro di Tesi è stato quello di sfruttare queste informazioni per aggregazioni successive, confrontandole anche con i dati provenienti dalle altre sorgenti. Per questo motivo anche questi documenti sono stati importati su Elasticsearch, non prima di aver eseguito su di essi alcune modifiche. Innanzitutto è stato necessario verificare che i file rispettassero le caratteristiche di un file CSV, ponendo attenzione a che il numero di celle di tutte le righe di un documento fosse sempre uguale. Poi i file nei quali si faceva uso dei filtri sono stati modificati in modo da contenere solo le informazioni ricavate dai filtraggi ed eliminando le altre. Inoltre uno dei fogli finali presentava una colonna indicante il costo, sia la cifra che la valuta (Lira o Euro), del fumetto: essa è stata sdoppiata nei due campi

"prezzo", caratterizzato dalle sole cifre del costo, e “valuta”, ovvero Lira o Euro. In questo modo si evitano i problemi di mapping esplicito, rendendo automatico e implicito il processo di definizione di come un documento dovrebbe essere associato al motore di ricerca Elasticsearch.
Dopo queste correzioni, per i file ottenuti è stata eseguita la stessa procedura di importazione sul database NoSQL svolta per i file relativi al sito Amazon. Anche in questo caso, infatti, è stato sfruttato il software di import precedentemente creato e leggermente modificato al fine di memorizzare esplicitamente come numeri quei campi contenenti solo cifre e sui quali sono state eseguiti in seguito delle operazioni come ad esempio la media.

4.4. Organizzazione e indicizzazione dei dati

L’analisi iniziale e generale delle sorgenti di riferimento, anche in base al tipo di risultato che era stato previsto per ciascun progetto, ha consentito di fare delle valutazioni inerenti al sistema di memorizzazione e analisi di dati, assibilabile a un data warehouse.
Prima di tutto è stato necessario capire come strutturare i contenitori dei dati e come indicizzare i dati da memorizzare su Elasticsearch.
Una delle particolarità che rendono questo motore di ricerca schema-less particolarmente apprezzato è la possibilità di indicizzare automaticamente un documento JSON semplicemente caricandolo e specificando:

- *index*: è come un database in un database relazionale e ha un mapping che definisce più type. È un namespace logico che mappa a uno o più shard primari e può avere zero o più shard replica.
  Uno shard è una singola istanza Lucene; si tratta di un'unità di basso livello automaticamente gestita da elasticsearch e che punta agli shard primari (ovvero dove il nuovo documento viene immediatamente

indicizzato) e shard replica (cioè di copia dei rispettivi primari, utili per la persistenza, nonché per miglioramento delle prestazioni dato che le richieste di estrazione e di ricerca possono essere gestite sia dagli shard primari che replica).

- *type*: un tipo è come una tabella in un database relazionale. Ogni tipo ha un elenco di campi che possono essere specificati per i documenti di quel tipo. Il mapping definisce come ogni campo nel documento viene analizzato.
- *id*: l'ID di un documento identifica il documento stesso.
  La terna index / type / id di un documento deve essere univoca. Se nessun ID è disponibile, allora sarà generato automaticamente in maniera casuale da Elasticsearch.

Per ognuno dei documenti ottenuti dalle raccolte dati, sono stati individuati quindi i relativi index e type mentre la creazione dell'id è stata lasciata al motore di ricerca.

Laddove i documenti sono stati direttamente indicizzati contestualmente all'esecuzione del software di recupero dati (progetto Pharma e parte del progetto Amazon), questi due elementi sono stati specificati all'interno del programma; nel caso invece dei documenti relativi al progetto I.N.D.U.C.K.S.e quelli della prima parte del lavoro su Amazon, essi sono stati specificati tramite riga di comando quando il software di import è stato eseguito.

Prima di sfruttare il motore di ricerca Elasticsearch come database NoSQL dove memorizzare le informazioni recuperate da Accurat, è stata necessaria un'analisi per capire come strutturare i contenitori dei dati.

In particolare, per quanto riguarda il progetto Pharma, il nome attribuito al relativo index è stato il nome del progetto stesso (minuscolo, come da specifica Elasticsearch); il nome del type è stato invece l'anno di riferimento (2014).

Gli index utilizzati per i dati Amazon sono invece due, come specificato nel Paragrafo 4.4.2: uno denominato *amazon_ranks* che contiene le classifiche e le informazioni di ogni prodotto, l'altro chiamato *amazon_trees* nel quale sono presenti gli alberi di categorizzazione dei prodotti delle *top 100*. I type sono stati invece, rispettivamente, il nome della categoria cui appartiene il prodotto e, nel secondo caso, lo stesso nome del raggruppamento ma seguito dal suffisso *_trees*.
Data invece la non omogeneità di molti dei file relativi ai dati recuperati dal database I.N.D.U.C.K.S., è stato necessario generare diversi indici, cercando di importare sotto lo stesso index quanti più documenti possibili, ovvero quelli aventi la stessa struttura e analoghi per il tipo di informazioni in essi contenuti. Il type è stato invece specificato in fase di importazione dei documenti stessi, specificando nella riga di comando di esecuzione del software quale fosse il numero del campo da utilizzare (di default il primo).

4.5. DAL (Data Access Layer)

In questo Paragrafo e nei Sottoparagrafi in cui esso si suddivide vengono descritte le interrogazioni e l'applicazione che consentono anche all'utente finale di poter interagire con le informazioni derivate dall'estrazioni descritte in questo Capitolo e memorizzate su Elasticsearch.
Inizialmente è stata fatta un'analisi per valutare quali tipi di interrogazioni potessero essere interessanti, quali operazioni definire e decidere quale struttura grafica dovesse avere l'applicazione stessa.
Successivamente sono state progettate le query, eseguendo dei test tramite l'estensione *Sense*[11] per Chrome e mediante il plugin *head*[12], che permette di

[11] https://chrome.google.com/webstore/detail/sense-beta/lhjgkmllcaadmopgmanpapmpjgmfcfig
[12] http://mobz.github.io/elasticsearch-head/

interagire graficamente con Elasticsearch, visualizzando gli indici, i suoi documenti e fornendo un ambiente di sviluppo per le interrogazioni.
Infine è stato deciso il tipo di output, ovvero la visualizzazione a schermo dei risultati cercati così da averli immediatamente visibili.

4.5.1. Esempi di query

Le possibili interrogazioni da eseguire, combinando tra loro query e filtri, sono potenzialmente centinaia e in questo lavoro di Tesi sono presentati alcuni esempi quanto più variegati possibili così da soddisfare le esigenze presenti di Accurat, ma cercando anche di prevedere alcune di quelle future.
L'idea di alcune query è nata dal fatto che precedentemente alcune operazioni sui file aggregati, come medie e somme, fossero state eseguite manualmente dall'utente utilizzando i file CSV. In questa fase è stato semplicemente necessario capire come interrogare Elasticsearch per ottenere gli stessi risultati e quindi visualizzarli.
In altri casi sono state individuate delle query utili per Accurat, ma che hanno anche titolo di esempio per successivi sviluppi. La loro progettazione infatti è stata eseguita per i casi specifici, ma esse sono facilmente riutilizzabili indipendentemente dal tipo di sorgente di dati, strutturate, semistrutturate o non strutturate.

4.5.1.1. Le query per Amazon

Tramite le informazioni memorizzate nell'indice *amazon_ranks*, Accurat desiderava sapere quali fossero mensilmente, per ciascuna categoria, il numero di prodotti acquisiti, quello dei prodotti distinti, quello dei venditori distinti, il prezzo medio degli articoli, la media dei commenti e la media delle stelline.

Per ottenere questo tipo di informazioni sono state utilizzate alcune Query DSL, ovvero le Match Query, le Bool Query e le Filtered Query, singolarmente o opportunamente combinate.

Le prime accettano testo o numeri o date, le analizzano e danno il risultato che corrisponde al *match*: in questo caso, memorizzando in variabili l'anno e il mese da considerare, è possibile specificare quale periodo prendere in considerazione per le aggregazioni. Affinché però si possa valutare la contemporanea presenza dei due parametri per ottenere i risultati richiesti, occorre utilizzare le Bool Query. Esse si costruiscono utilizzando una o più clausole booleane, ognuna con un evento specifico a seconda che la clausola debba essere necessariamente presente nel documento come in questo caso (must), che possa essere eventualmente presente (should) o che non debba essere presente (must_not). Le Filtered Query sono invece utilizzate per combinare tra loro le query. [20]

Dopo aver individuato quali prodotti rientrino nella categoria e nel periodo richiesti, si calcolano le varie aggregazioni: tramite l'operatore *cardinality* vengono contati i codici asin distinti così da ottenere la cardinalità degli articoli distinti e lo stesso ragionamento viene applicato per contare i venditori distinti su Amazon nel mese e per il raggruppamento selezionato. Per le medie relative al prezzo, alle stelline e ai commenti viene invece utilizzato l'operatore *avg*. Anche in questi casi, per assegnare un parametro alla variabile tramite cui si eseguono le aggregazioni, viene fatto uso delle Match Query.

Occorre inoltre impostare l'URL di ricerca per comunicare con Elasticsearch tramite le API RESTful sulla porta 9200, specificando anche l'indice, *amazon_ranks*, e il type, ovvero la categoria per la quale si intende eseguire l'interrogazione.

4.5.1.2. Le query per Pharma

La query per visualizzare l'intero contenuto dell'indice *pharma*, ovvero la tabella dei farmaci, fa uso della Match All Query.

Tuttavia, se non diversamente specificato, il numero di righe restituite è di massimo 10: tramite il parametro *size* si è modificato questo limite fissandolo a 50, ovvero la reale dimensione della tabella. Infine viene impostata la posizione come elemento rispetto cui eseguire l'ordinamento crescente. Questo non è un particolare secondario poiché, per impostazione predefinita, i risultati vengono memorizzati e restituiti in ordine decrescente di rilevanza. In Elasticsearch il valore di riferimento per la rilevanza è rappresentato da un numero a virgola mobile restituito nei risultati di ricerca come *_score*, quindi l'ordine predefinito è quello discendente riferito ad esso. Quando si vuole, come per questa query, un ordine differente da quello di default, è necessario specificare l'attributo e il tipo di ordinamento.

Tramite le altre richieste si vogliono ottenere delle viste della tabella dei medicinali, ovvero impostando rispettivamente o il principio attivo, o la famiglia del farmaco o il nome della ditta, si desidera visualizzare solo i farmaci che soddisfano l'interrogazione e le relative informazioni ad essi associate.

La struttura utilizzata per queste query è simile a quella precedente ma, invece della Match All Query, si utilizzano le Match Query per specificare il parametro di ingresso tramite cui effettuare la ricerca in base alla vista che si desidera.

In alcuni casi l'input è stato anche di due o più parole e questo ha comportato la necessità di considerare gli argomenti di ingresso come unico valore tramite l'uso dell'operatore *and* che permette di considerare più parole prese in input come unica stringa. L'uso di questo flag è stato basilare per distinguere due nomi aventi una o più parole in comune, come nei casi in cui

si sono cercati i medicinali aventi come principio attivo il "Paracetamolo" piuttosto che "Paracetamolo, associazioni escl. psicolettici", o quelli prodotti dall'azienda "NOVARTIS CONSUMER HEALTH S.P.A." invece della "NOVARTIS FARMA S.P.A.".
Nell'URL di ricerca per accedere ad Elasticsearch, occorre aggiungere solo il nome dell'indice, *pharma.*

4.5.1.3. Le query per I.N.D.U.C.K.S.

Le interrogazioni sugli indici con i dati recuperati dal portale I.N.D.U.C.K.S. sono tra loro simili e anche queste, con le opportune semplici modifiche, riutilizzabili per gli indici contenenti dati derivanti da sorgenti di dati differenti. In effetti la struttura delle query qui presentate è quasi la medesima delle query sui farmaci, opportunamente modificate cambiando variabili e parametri di riferimento.
La prima interrogazione riguarda l'indice *topolino-cooccorrenze-main* e in esso si cercano i personaggi che appaiono nella stessa storia del personaggio specificato come input, cercando il *match* tramite una Match Query; la query è poi completata specificando il numero massimo di risultati che ci si aspetta e l'ordinamento crescente secondo il nome del personaggio che appare insieme a quello cercato.
Per la seconda invece, relativa a *topolino-inducks-storie-personaggi*, il *match* cercato è tra il nome del personaggio scelto dal menu a tendina e una delle parole che compongono i titoli delle storie di tutti i fumetti del Topolino. Viene quindi indicato il numero massimo di righe visualizzabili e l'ordinamento secondo il codice della storia.
Anche in questo caso l'URL di ricerca per accedere ad Elasticsearch contiene il nome dell'indice, ma questa volta è una variabile che dipende da quale l'utente ha scelto per l'interrogazione.

4.5.1.4. Le query su tutti gli indici

Come detto, le query presentate sono state progettate in base alle necessità e per specifici indici, tuttavia modificando variabili e parametri è possibile il loro riutilizzo anche per indici diversi e con dati derivanti da fonti eterogenee. Ad ulteriore dimostrazione di questo, vengono presentate altre due query che interrogano tutti gli indici di Elasticsearch.

Per fare ciò si utilizzano le Search API, che permettono di eseguire una query e di restituire i risultati della ricerca (hits) che la soddisfano. Esse possono essere applicate su più type di uno stesso indice o su più indici semplicemente indicando nella URL di accesso a Elasticsearch quali indici e quali type prendere in esame.

Tramite la prima interrogazione si cercano i prodotti che soddisfano una certa uguaglianza o disuguaglianza rispetto a un prezzo. Per far questo, alle Query DSL fin qui utilizzate occorre combinare le Range Query, che restituiscono i documenti con i campi i cui termini si trovano all'interno di un certo intervallo. I parametri utilizzati per queste interrogazioni sono stati “lt”, “lte”, “gt” e “gte” per valutare, rispettivamente, se un valore fosse minore, minore o uguale, maggiore, maggiore o uguale del numero inserito in input dall’utente. Occorre specificare una Bool Query per ogni parametro utilizzato e in ognuna di esse bisogna indicare tante Range Query quanti sono i campi su cui si vuole verificare la disuguaglianza.

Nel caso dell’uguaglianza, invece delle Range Query, si utilizzano le Filtered Query, ma lo schema dell’interrogazione è il medesimo.

Nella URL di accesso è indifferentemente possibile utilizzare i nomi degli indici su cui eseguire la ricerca, o la parola chiave *_all* per riferirsi a tutti.

Per contare invece le istanze di una determinata parola tra tutti i documenti memorizzati si utilizzano le URI Search di Elasticsearch, che permettono di

eseguire una query di ricerca usando una URI dove si forniscono i parametri necessari, ovvero l'indirizzo tramite cui comunicare con Elasticsearch, seguito dalla parola che si intende cercare, che in questo caso viene acquisita tramite il DAL. Nel corpo della query vengono poi settati il numero di righe da visualizzare e l'ordinamento secondo il nome del type dove ciascun risultato visualizzato è contenuto.

4.5.2. Descrizione del DAL

L'applicazione consiste di una semplice pagina HTML che, sfruttando la libreria jQuery e le Query DSL basate su JSON fornite da Elasticsearch per definire le query di interrogazione per i propri indici, permette di ottenere dei risultati che altrimenti richiederebbero lunghe ricerche manuali.
Così come tutti gli indici Elasticsearch, anche questa applicazione per la loro interrogazione è stata eseguita in locale e i risultati prodotti sono visualizzati direttamente sullo schermo. Essa, nella sua schermata iniziale che è mostrata nella Figura 17, presenta un menu con *radio button* che consentono di selezionare o uno specifico indice appartenente a uno dei tre raggruppamenti (Pharma, Amazon o I.N.D.U.C.K.S.), o tutti gli indici. Sebbene questi non siano interamente elencati nella schermata a causa del considerevole numero degli stessi, l'opzione "Seleziona tutti" consente di interrogare tutti gli indici presenti su Elasticsearch secondo le query di esempio predisposte.

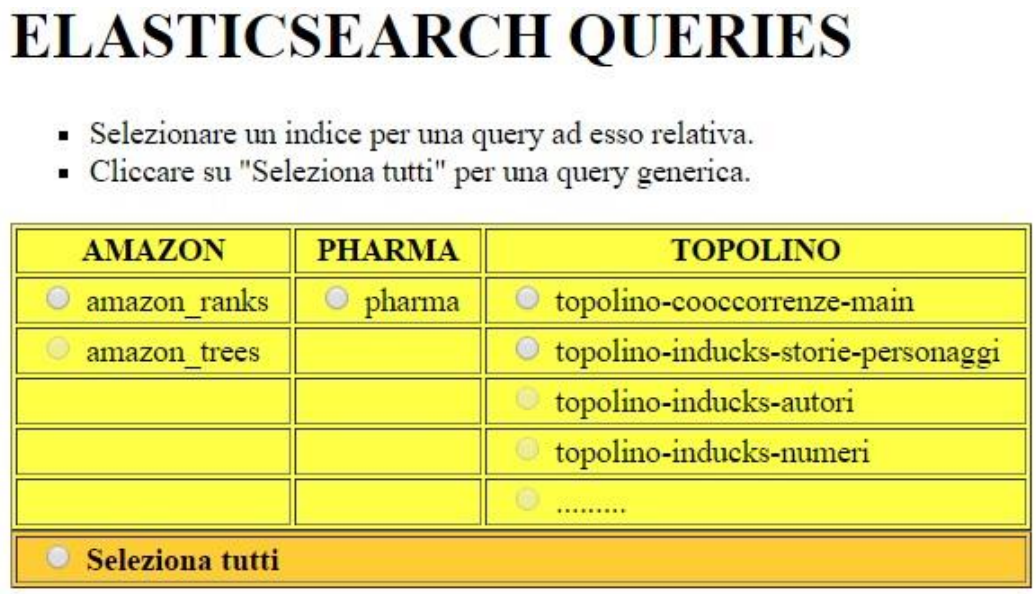

Figura 18: Menu iniziale dell'applicazione per le query su Elasticsearch

La selezione di una delle opzioni mostrerà un sottoelenco di opzioni che consentirà di eseguire una delle query indicate.

In particolare, dopo aver scelto l'indice *amazon_ranks*, occorre impostare le informazioni di input tramite dei menu a tendina, ovvero il type, cioè una delle categorie dei *Bestseller*, il mese e l'anno di riferimento.

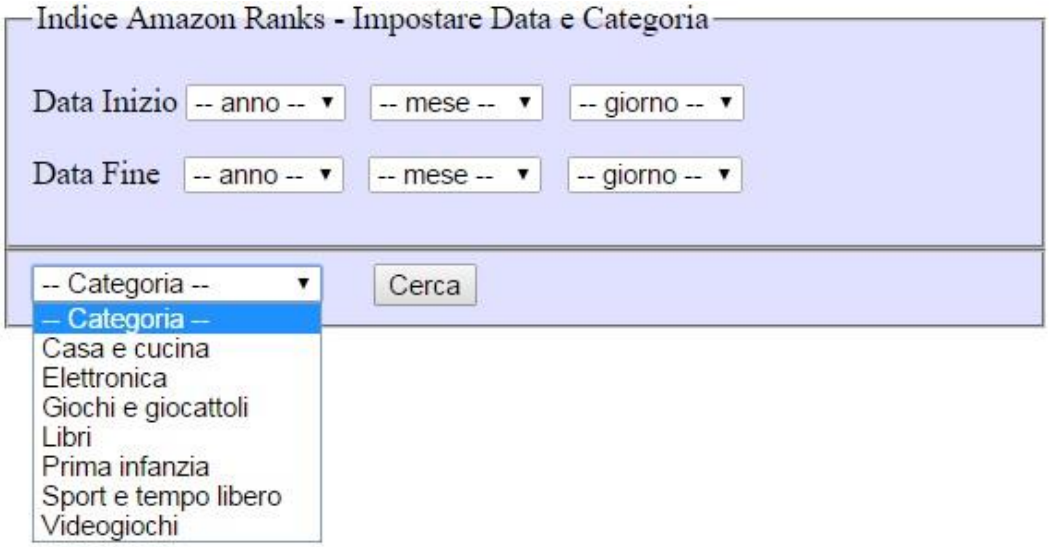

Figura 19: Sottomenu per interrogare l'indice amazon_ranks

Poiché le aggregazioni sono sempre richieste a cadenza mensile, l'inserimento della data di inizio è sufficiente per ottenere il risultato cercato; tuttavia, per maggiore completezza e al fine di avere un form già pronto per eventuali modifiche, è stato incluso anche un menu per impostare in futuro una data di fine diversa.

Il risultato della interrogazione è una tabella che mostra il nome dell'indice e della categoria, il mese e l'anno, il numero di prodotti acquisiti, quello dei prodotti distinti, quello dei venditori distinti, il prezzo medio degli articoli, la media dei commenti e la media delle stelline.

Queste associazioni erano state inizialmente determinate manualmente, anche tramite l'uso delle funzioni dei fogli di calcolo. A causa però della la ricorrente necessità di ottenere queste informazioni, sono state create delle query ad hoc all'interno dell'applicazione così da consentire a chiunque e in breve tempo di ottenerle.

La seconda colonna del menu principale si riferisce al progetto sui farmaci e cliccando sul bottone relativo all'indice *pharma* si ha accesso a un sottomenu con quattro possibili interrogazioni.

La prima voce consente di visualizzare l'intera tabella dei farmaci di cui si è discusso nel Paragrafo 4.2.1, ordinata secondo l'ordine crescente della posizione in classifica.

La seconda opzione permette di ottenere l'elenco dei medicinali, con le relative informazioni, che contengono uno dei principi attivi specificati in un ulteriore menu a tendina, il quale appare quando viene selezionata questa voce. L'output è consiste nell'indicazione del numero di risultati trovati e di una tabella con almeno una riga poiché i principi attivi indicati sono tutti quelli presenti nella tabella di partenza.

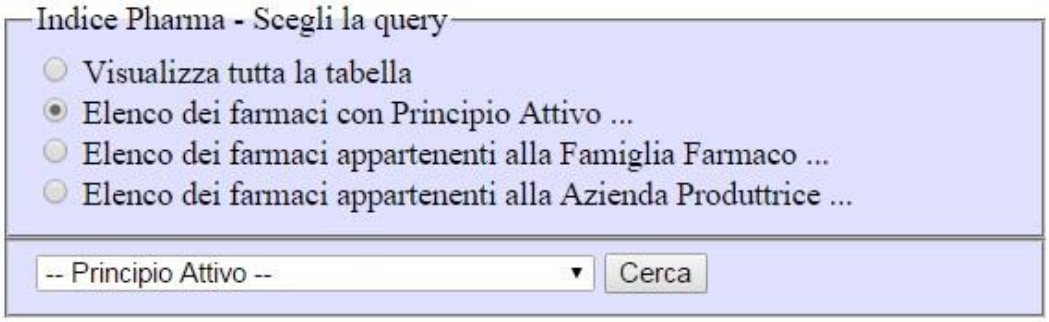

Figura 20: Sottomenu per interrogare l'indice pharma cercando un principio attivo

Similmente a questa scelta, anche la terza e la quarta opzione del sottomenu *pharma* danno come risultato una vista della tabella di partenza e il conteggio dei risultati restituiti. In particolare è possibile filtrare secondo la famiglia del farmaco o l'azienda produttrice, scegliendo tra le voci del menu corrispondente che appare una volta selezionata l'opzione. Anche in questi ulteriori tre casi l'ordinamento è stato mostrato in ordine crescente di posizione in classifica.

Per quanto riguarda invece gli indici contenenti le informazioni sui fumetti Disney, al fine di evitare un lungo elenco degli stessi, ne vengono mostrati a titolo di esempio quattro di cui solo due selezionabili per essere interrogati: quello principale sulle co-occorrenze tra personaggi e quello che contiene i titoli delle storie. In entrambi i casi il tipo di query è la medesima e consente di ottenere le istanze di uno specifico personaggio selezionabile tramite il sottomenu che appare dopo aver scelto l'indice.

Figura 21: Sottomenu per interrogare gli indici Inducks

L'output mostrato è quindi il conteggio delle istanze ottenute come risposta alla interrogazione e una vista della tabella di partenza.

Infine, per mostrare ancora una volta come l'applicazione riesca ad interagire con i dati memorizzati indipendentemente dal tipo di sorgente, sono state implementate delle query di esempio per interrogare tutti gli indici presenti su Elasticsearch. Poiché gli indici raccolgono informazioni molto diverse tra loro, per individuare in fase di analisi delle query che potessero interessare indici di raggruppamenti diversi è stato necessario un accurato studio dei file in possesso così da individuare dei punti in comune.

È stato notato che, così come sui documenti importati su *amazon_ranks*, anche su quelli presenti in *topolino-inducks-numeri* è stato memorizzato il prezzo dell'articolo. Mentre però la relativa colonna è costituita, nei file Amazon, dalle sole cifre del costo, quella che indica il valore di acquisto di un determinato numero del Topolino presenta sia le cifre che la valuta, ossia Lira o Euro.

Il suddetto file è stato quindi duplicato e la colonna relativa al prezzo è stata divisa in due: una i cui campi contengono il le cifre del costo, l'altra con la specifica del tipo di moneta. Così facendo è stato possibile importare il nuovo file facendo un mapping implicito dei costi, definiti dal motore di ricerca come numeri: su di essi sono stati quindi eseguite delle operazioni come la valutazione della correttezza di una disuguaglianza o uguaglianza.
Grazie a questo campo comune e a questa modifica, la prima possibile interrogazione definita per tutti gli indici, cliccando su "Seleziona tutti", è stata quella di ricerca di un articolo il cui prezzo soddisfi una certa equazione o disequazione.

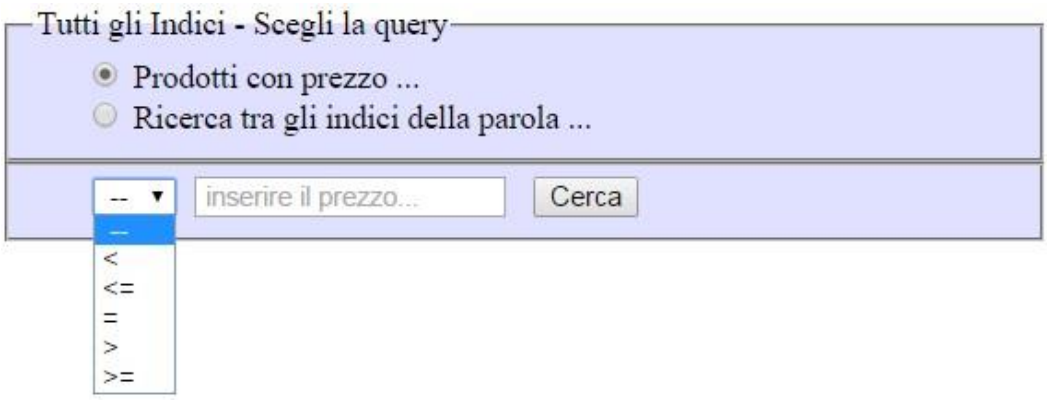

Figura 22: Sottomenu di ricerca tra tutti gli indici

Chiaramente non è stato necessario rinominare i due campi relativi ai prezzi affinché avessero lo stesso nome ma è bastata una corretta progettazione della interrogazione.
L'output fornito consiste di un conteggio delle istanze che hanno fornito un "match" e le viste delle righe delle tabelle dove si ha un prezzo appartenente all'intervallo cercato.
La seconda query valida per tutti i documenti, richiesta da Accurat, ha permesso il conteggio delle istanze di una determinata parola tra tutte le

informazioni in possesso. Certamente un'operazione del genere richiederebbe in certi casi, data la mole di dati, un incredibile ammontare di tempo se eseguita manualmente; al contrario, sfruttando questa applicazione tale ricerca avviene in pochi secondi.
Anche in questo caso, il risultato è il numero di ripetizioni della parola cercata all'interno dei documenti e le righe delle tabelle che la contengono.

4.6. Esempi di utilizzo del sistema

Attraverso i dati raccolti dalle fonti di dati, Accurat ha prodotto due visualizzazioni relativamente ai progetti Pharma e I.N.D.U.C.K.S..
Esse vengono di seguito mostrate come esempio di reale e concreto utilizzo delle informazioni da parte dell'azienda.

4.6.1. L'infografica finale per Pharma

La visualizzazione finale creata da Accurat per il progetto Pharma esplora i 50 farmaci senza obbligo di prescrizione più venduti nel primo semestre 2014. Per ciascun farmaco si restituiscono:

- la posizione di vendita nel secondo semestre 2013;
- le proprietà terapeutiche;
- il principio attivo;
- il numero di varianti;
- i colori della confezione;
- il numero di confezioni vendute;
- le unità per confezione;
- gli effetti indesiderati e le controindicazioni;

- la casa farmaceutica che lo produce (con indicazioni di provenienza e longevità).

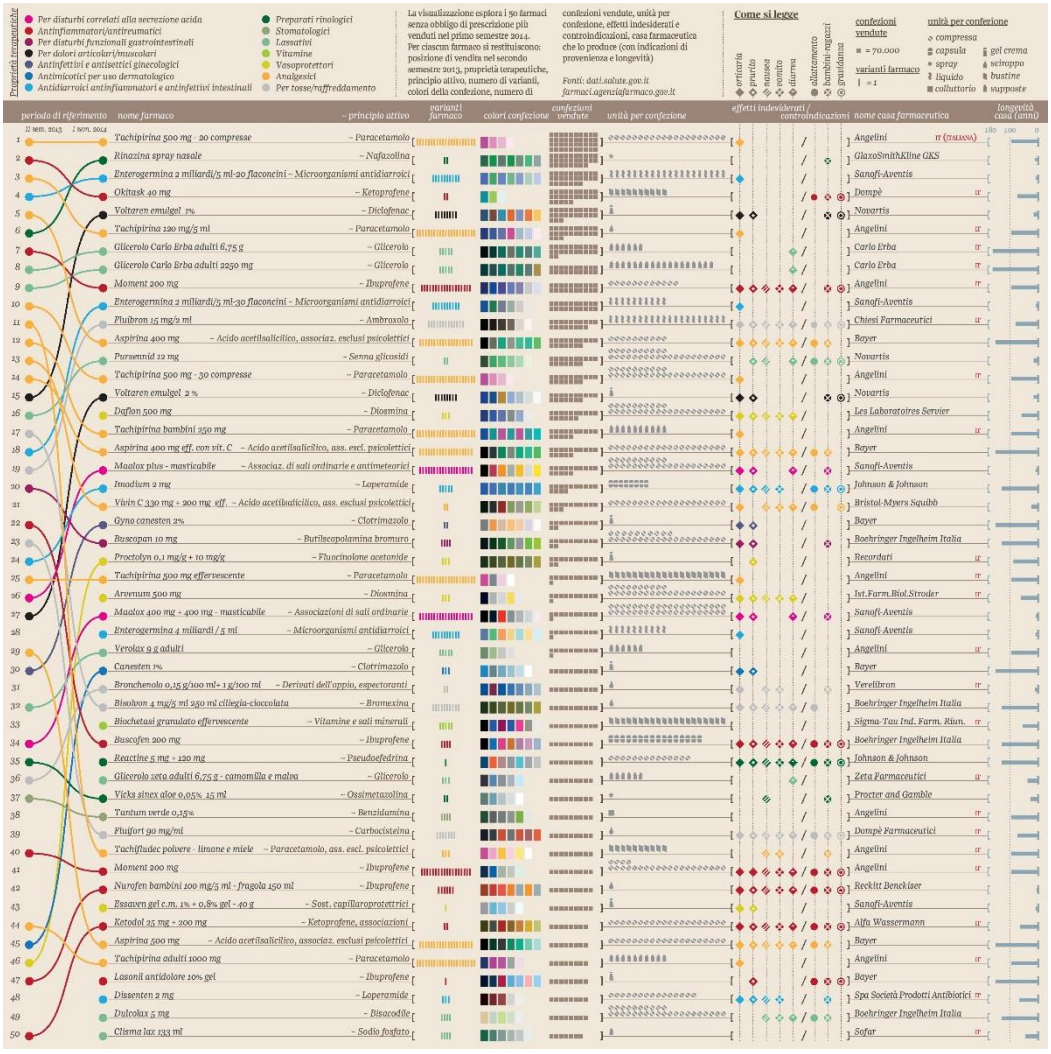

Figura 23: Visualizzazione creata da Accurat sui 50 farmaci da banco più venduti nel primo semestre del 2014

Nelle Figure 23, 24, 25, 26 e 27 vengono mostrati dei particolari che spiegano come leggere la visualizzazione mostrata nella Figura 22: la prima mostra come interpretare i simboli grafici; la seconda mostra i colori associati a ciascun effetto indesiderato e controindicazione; le ultime tre mostrano

invece un esempio di farmaco e del relativo principio attivo, le sue varianti, il colore confezione, il numero di confezioni vendute e di unità, gli effetti indesiderati, le controindicazioni e il nome della casa farmaceutica.

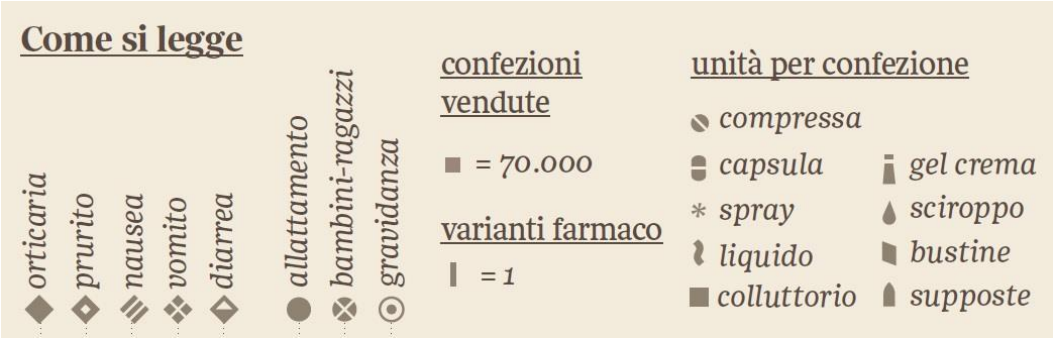

Figura 24: Legenda dei simboli utilizzati da Accurat per la propria visualizzazione sui farmaci.

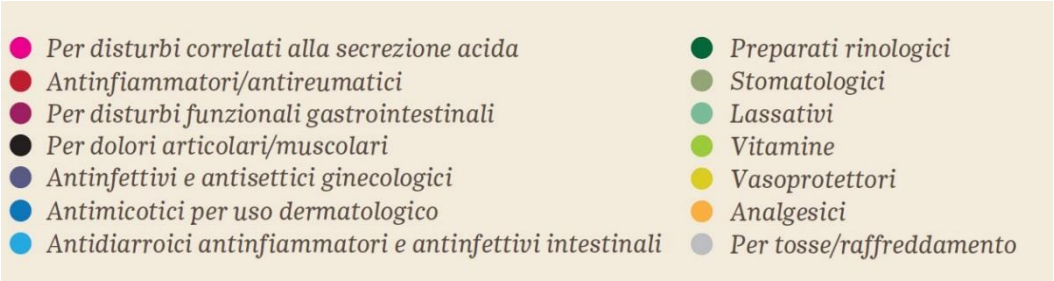

Figura 25: Proprietà terapeutiche dei farmaci

Figura 26: Esempio di farmaco e principio attivo - Tachipirina

Figura 27: Esempio di varianti, colore confezione, numero di confezioni vendute e di unità - Tachipirina

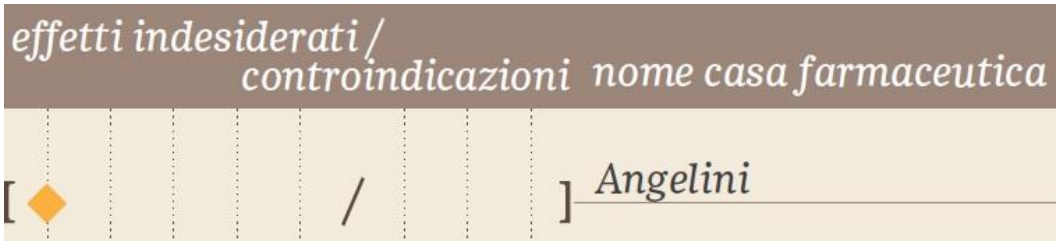

Figura 28: Esempio di effetti indesiderati, controindicazioni e nome della casa farmaceutica - Tachipirina

4.6.2. L'infografica per I.N.D.U.C.K.S.

La visualizzazione finale creata da Accurat sulla base dei dati recuperati dal portale I.N.D.U.C.K.S. esplora i quindici principali personaggi apparsi all'interno degli albi di «Topolino» nel periodo 1949 – 2014 (fino al numero 3.060). Per ciascun personaggio si restituisce:

- il numero di apparizioni nelle storie con indicazione del numero di ruoli da protagonista;
- il numero di apparizioni in copertina;
- i quattro sostantivi maggiormente utilizzati nei titoli delle storie in cui il personaggio è presente;
- le dieci principali relazioni con altri personaggi.

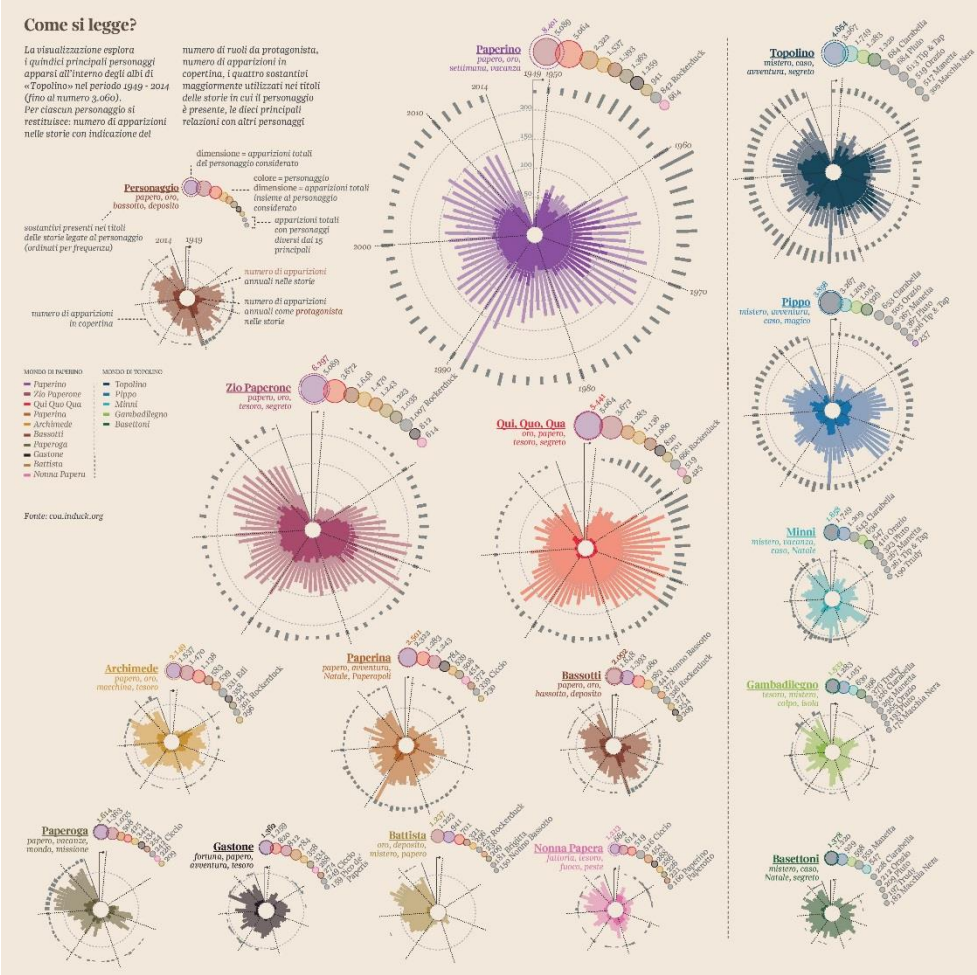

Figura 29: Visualizzazione creata da Accurat sui dati recuperati dal portale I.N.D.U.C.K.S.

Nelle figure 29 e 30 viene mostrato come interpretare l'infografica creata da Accurat e la lista dei personaggi Disney analizzati.

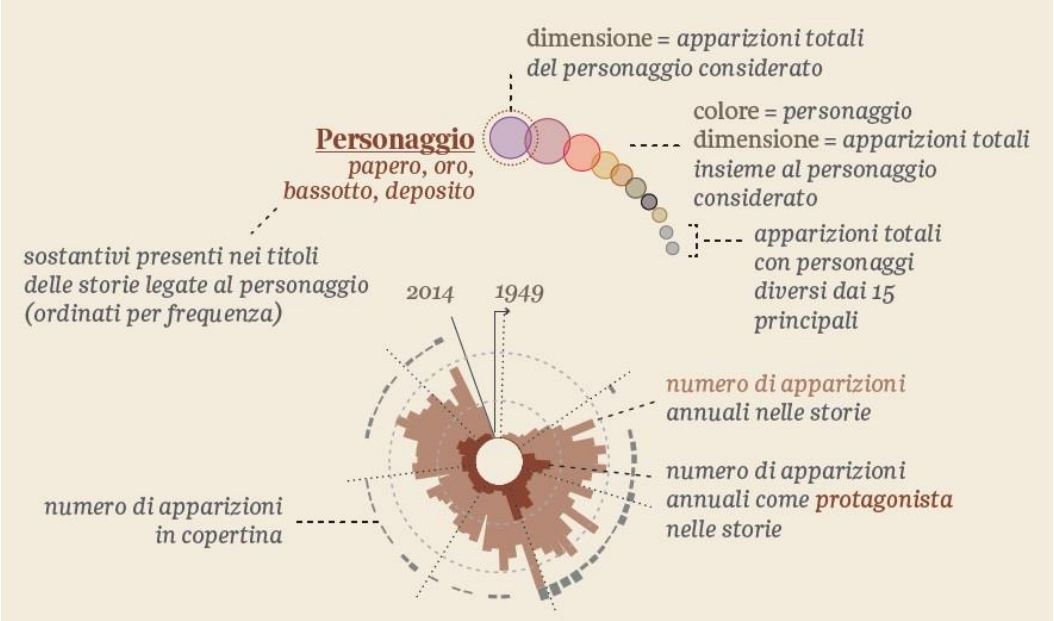

Figura 30: Come leggere le visualizzazioni sui personaggi Disney

Figura 31: Legenda personaggi Disney

Nelle figure 31 e 32 vengono invece mostrati due esempi della visualizzazione relativamente a due tra i personaggi considerati: Paperino e Topolino.

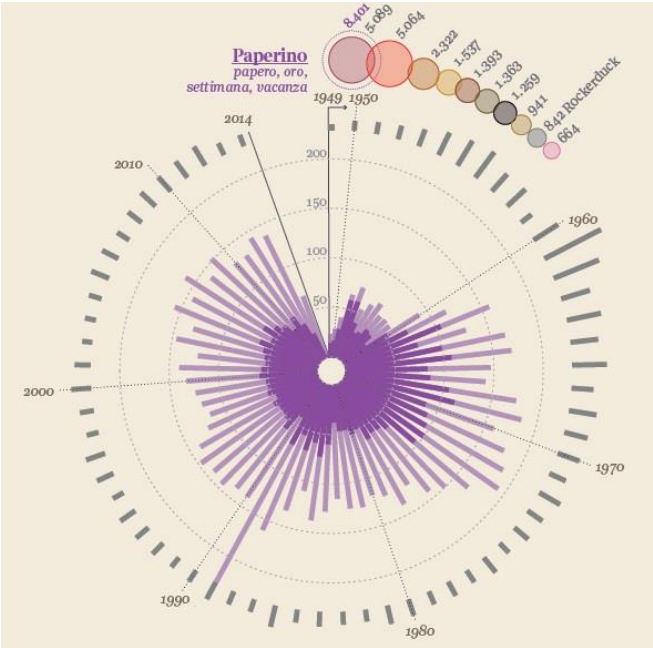

Figura 32: Esempio di visualizzazione - Paperino

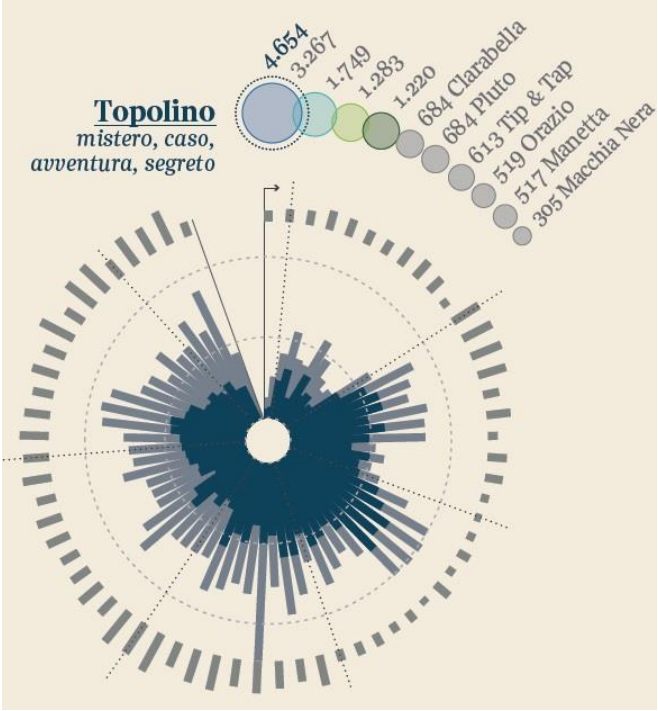

Figura 33: Esempio di visualizzazione - Topolino

5. CONCLUSIONI

L'esigenza di un supporto alle decisioni è diventata oggi un'attività che le aziende spesso richiedono poiché riconoscono il ritorno dell'investimento.
L'architettura descritta in questo lavoro di tesi e i processi che la costituiscono hanno voluto offrire gli strumenti e dei metodi per il recupero di dati da fonti strutturate, semistrutturate e non strutturate, con l'intento di memorizzarli e analizzarli per creare nuova conoscenza.
Come supporto a questa analisi, è stata realizzata un'applicazione web che consente di eseguire facilmente e in maniera grafica alcune interrogazioni a Elasticsearch, utilizzato come database NoSQL, e questo indipendentemente dal tipo di sorgente dei dati.
Per raggiungere questo obiettivo, prima di progettare e implementare l'intero sistema, è stato necessario studiare i portali web di riferimento per capirne la struttura e identificare il metodo più efficiente per recuperare i dati, ad esempio lo scraping piuttosto che l'utilizzo di API specifiche.
Il lavoro si è poi sviluppato nella progettazione e nella realizzazione dei software per recuperare e memorizzare i dati dal sito italiano di Amazon e da quello dell'Agenzia del farmaco, così da arricchire il pacchetto di dati già in possesso precedentemente recuperati dal portale I.N.D.U.C.K.S., un database per i fumetti Disney.
Le informazioni ottenute sono state salvate in fogli elettronici e su Elasticsearch così da averle in un ambiente unico.
Il passo successivo è stato l'ideazione e la creazioni sia delle query, sia del DAL che le utilizza al fine di fornire un accesso semplificato ai dati memorizzati nel data store.
Le interrogazioni, non legate al tipo di sorgente, sono state realizzate in alcuni casi per interagire gli specifici indici per cui sono state create, in altri

per interagire indistintamente con tutti gli indici, proprio per dimostrare l'assenza di legami col tipo di fonte da cui sono stati recuperati i dati.

Per testare infine l'effettiva correttezza dei risultati forniti dall'applicazione, sono state eseguite alcune operazioni di verifica sulle query relative alle medie, ai conteggi e alle viste tramite l'ausilio delle funzionalità dei fogli elettronici.

Questo lavoro di Tesi ha quindi consentito di ridurre i tempi di recupero delle informazioni memorizzate, fornendo all'utente la possibilità di usufruire della comodità e dei celeri tempi di risposta delle query senza la necessità di avere la conoscenza per gestirle, grazie a una semplice interfaccia con menu. Si ritengono pertanto raggiunti gli obiettivi prefissi per questo lavoro di Tesi.

Per quanto riguarda i possibili sviluppi futuri, sicuramente potrebbe essere interessante allargare il set di query partendo da quelle proposte, modificandole secondo nuove necessità dettate anche da successivi progetti. Ad esempio, recuperando i dati relativi alle future classifiche dei 50 farmaci più venduti, si potrebbero eseguire dei confronti per valutare come siano variate nel tempo le necessità delle persone che acquistano i farmaci e delle aziende che li producono.

6. BIBLIOGRAFIA

[1] AMZN-2014.12.31-EX99.1. 29 January 2015.
http://www.sec.gov/Archives/edgar/data/1018724/000101872415000004/amzn-20141231xex991.htm
(ultima consultazione: 30 Marzo 2015)

[2] Massimo Carro. NoSQL Databases.
http://arxiv.org/ftp/arxiv/papers/1401/1401.2101.pdf
(ultima consultazione: 30 Marzo 2015)

[3] Paolo Atzeni, Stefano Ceri, Piero Fraternali, Stefano Paraboschi, Riccardo Torlone. Basi di dati - Modelli e linguaggi di interrogazione. McGraw-Hill, 2002.

[4] Marc Seeger. Key-Value Stores: a practical overview. 21 September 2009.
http://blog.marc-seeger.de/2009/09/21/key-value-stores-a-pratical-overview

[5] DB-ENGINES.
http://db-engines.com/en/article/Search+Engines

[6] G. Barbieri. SQL e NoSQL: cosa sapere sui database non relazionali. 24 Giugno 2014.
http://blog.artera.it/programmazione/sql-nosql-database-non-relazionali
(ultima consultazione: 30 Marzo 2015)

[7] Elastic.co Products.
https://www.elastic.co/products
(ultima consultazione: 10 aprile 2015)

[8] Elasticsearch Guide.
http://www.elastic.co/guide/en/elasticsearch/guide/master/index.html
(ultima consultazione: 10 aprile 2015)

[9] John Miles Smith et al. Multibase: integrating heterogeneous distributed database systems, AFIPS '81 Proceedings of the May 4-7, 1981, national computer conference. pp. 487–499.

[10] Paolo Atzeni, Stefano Ceri, Piero Fraternali, Stefano Paraboschi, Riccardo Torlone. Architetture per l'analisi dei dati. McGraw-Hill, 2007.

[11] W. H. Inmon. Building the Data Warehouse. Wiley Publishing, Inc., 2005.

[12] Francesco Merlo. Data Warehousing. Corso di Sistemi Informativi per l'Impresa - Politecnico di Milano.

[13] C. Vercellis. Business Intelligence - modelli matematici e sistemi per le decisioni. McGraw-Hill, 2006.

[14] ACCURAT - Information design.
http://www.accurat.it/

[15] John Miles Smith et al. Multibase: integrating heterogeneous distributed database systems, AFIPS '81 Proceedings of the May 4-7, 1981, national computer conference. pp. 487–499.

[16] Elasticsearch - Inverted Index.
http://www.elastic.co/guide/en/elasticsearch/guide/current/inverted-index.html
(ultima consultazione: 10 aprile 2015)

[17] Elasticsearch – Analylis.
http://www.elastic.co/guide/en/elasticsearch/guide/current/analysis-intro.html
(ultima consultazione: 10 aprile 2015)

[18] Agenzia Italiana del Farmaco.
https://farmaci.agenziafarmaco.gov.it/bancadatifarmaci/cerca-farmaco

[19] I.N.D.U.C.K.S..
http://www.inducks.org/

[20] Elasticsearch Queries.
http://www.elastic.co/guide/en/elasticsearch/reference/1.x/query-dsl-queries.html
(ultima consultazione: 10 aprile 2015)