

COMPETITIVE POKÉMON GRAPH DATABASE

Giorgio Carbone (matr. 811974)¹, Gianluca Cavallaro (matr. 826049)², Remo Marconzini (matr. 883256)³

Sommario

Nel 2021 è stato festeggiato il 25° anniversario del franchise Pokémon, nato in Giappone il 27 febbraio 1996 con la pubblicazione ufficiale dei primi due giochi della serie, **Pokémon Rosso** e **Pokémon Blu**. Da allora ha avuto inizio un'ascesa vertiginosa e costante in termini di popolarità, che ha reso Pokémon il *media brand* più redditizio di sempre. Videogiochi, cartoni animati, film, carte collezionabili e giocattoli: l'universo Pokémon è in continua espansione e, nel tempo, ha saputo catturare l'interesse di bambini e adulti. Contemporaneamente, il lato videoludico della compagnia ha iniziato a farsi strada nel mondo del gioco competitivo, introducendo dinamiche e meccaniche di gioco sempre più mature. Nel 2009 si è svolto il primo Campionato Mondiale di lotte Pokémon (**Pokémon World Championship**), che, annualmente, porta all'incoronazione del miglior allenatore Pokémon al mondo. Nasce così l'idea di creare una base di dati che racchiuda tutte le informazioni utili ad addentrarsi nel mondo delle lotte competitive: statistiche e descrizioni relative a Pokémon, mosse, strumenti, abilità e altre meccaniche e entità di gioco; relazioni fra gli elementi di *gameplay* e dati di utilizzo relativi ai *match* competitivi, in termini di compagni di squadra, oggetti, mosse, abilità e statistiche più popolari per ogni Pokémon. Per la realizzazione di questo database sono stati utilizzati dati acquisiti mediante Data Scraping da fonti plurime e API. Data la natura del progetto, fortemente incentrata sulle relazioni presenti nei dati, è stato realizzato un database a grafo, creato tramite l'utilizzo di Neo4J.

Keywords

Data Management – Pokémon – API – Scraping – Neo4J

^{1,2,3}Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli studi di Milano-Bicocca, Milano, Italia

Indice

1	Introduzione	2
2	Data Acquisition	2
2.1	Web API	2
	Librerie e processi implementati • Dati ottenuti da <i>PokéAPI</i>	
2.2	Web Scraping	5
	Librerie e processi implementati • Scraping da <i>Pokémon Database</i> • Scraping da <i>Pikalytics</i> • Scraping da <i>VictoryRoad</i> • Scraping da <i>serebii.net</i> • Scraping da <i>Bulbapedia</i>	
3	Data Cleansing, Integration ed Enrichment	14
3.1	Librerie utilizzate	14
3.2	Dataset da API e da scraping di <i>Pokémon Database</i> : Integrazione	15
	Data Cleaning • Relazioni • Pokémon • Items • Abilities e Moves	
3.3	G-Max Moves e Gigantamax Pokémon	19
	Pokémon: integrazione Gigantamax Pokémon • Mosse: integrazione G-Max Moves / Tabella ponte GMAX_MOVE • Dataset Pokémon: nuova chiave primaria	
3.4	I Types: dataset e tabelle ponte relative	20
	Dataset Types • Tabella ponte Type-Type "MOVE_EFFECTIVENESS_ON_POKEMON" • Tabelle ponte Pokémon-Type "IS_OF_TYPE" e Moves-Type "MOVES_IS_TYPE"	
3.5	Altre relazioni che interessano i Pokémon: tabelle ponte	22
	Pokémon-Pokémon "EVOLVES_FROM" e "HAS_VARIANT" • Pokémon-Abilities "MAY_HAS" e Pokémon-Moves "MAY_LEARN"	

3.6	Integrazione regolamento <i>VGC 2022 Series 12</i> a Pokémon	23
	Regolamento <i>Series 12 VGC 2022</i> • Integrazione regolamento al dataset 'Pokémon'	
3.7	Lotte competitive su <i>Showdown!</i> : integrazione e tabelle ponte	25
	Fuzzy matching • Integrazione statistiche utilizzo dei Pokémon • Correzione nomi di Pokémon e mosse nelle tabelle ponte	
4	Data Quality	26
4.1	Currency	26
4.2	Completeness	27
4.3	Consistency	28
5	Data Model and Storage	29
5.1	Definizione dei nodi	30
5.2	Definizione delle relazioni	31
5.3	Model Schema	32
5.4	Data Storage	33
	Import nodi • Import relazioni	
6	Graph Exploration	34
6.1	Query	34
6.2	Competitive Data Graph Visualization	37
	Compagni di squadra più utilizzati • Mosse maggiormente utilizzate	

1. Introduzione

L'idea alla base del progetto è quella di realizzare un database contenente tutte le informazioni relative al gioco Pokémon competitivo, con particolare riferimento al regolamento *Video Game Championship Series 12*, formato ufficiale in vigore per i tornei e gli eventi ufficiali nel periodo Febbraio – Agosto 2022 e valido per il Pokémon World Championship di Londra di agosto 2022. L'obiettivo è quello di ottenere uno strumento utile come supporto al gioco competitivo, sia per giocatori alle prime armi che per quelli esperti. I diversi Pokémon sono posti in relazione con i compagni di squadra, le mosse, gli strumenti, le abilità le statistiche di base con cui risultano abbinati più frequentemente all'interno dei team nei match competitivi. Per questo motivo, la scelta sul tipo di database da realizzare è ricaduta su un database a grafo, realizzato tramite Neo4J. La scelta del database a grafo ha permesso di sfruttare la sua caratteristica di essere *schema less*, che consente di creare nodi, per modellare le diverse entità, e archi, per modellare le varie relazioni, senza seguire uno schema predefinito. Il database è stato popolato attraverso dati ottenuti mediante API e Web Scraping, opportunamente integrati e processati.

2. Data Acquisition

I dati sono stati ottenuti attraverso due metodi diversi: Web API e Web Scraping.

Le API, acronimo di Application Programming Interface, sono particolari interfacce che consentono la comunicazione fra componenti software. In particolare, sono progettate in modo da permettere al programmatore di accedere a dati di terze parti. Tipicamente, le API rispondono alle richieste fornendo dati in formato JSON.

Con il termine Web Scraping, invece, si indica una tecnica di estrazione di dati da pagine web mediante l'utilizzo di appositi software. Tali programmi, tipicamente, simulano la navigazione umana nel World Wide Web, sfruttando il protocollo HTTP per reperire le informazioni.

2.1 Web API

La Web API utilizzata per la raccolta di dati sui Pokémon è *PokéAPI*[1], API pubblica tra le più note in rete, che raccoglie dati relativi all'intero mondo Pokémon e, in particolare, a tutte le meccaniche dei videogiochi finora pubblicati.

2.1.1 Librerie e processi implementati

La Web API è stata utilizzata tramite Python. Inizialmente, sono stati individuati gli url in cui reperire le informazioni utili, ottenuti poi tramite la libreria **Python Requests**[2] attraverso l'utilizzo del metodo `get()`. Gli url, che erano della forma `https://pokeapi.co/api/v2/ability/{id or name}`, dovevano essere completati attraverso un identificativo che indicasse il percorso verso la specifica entità. Pertanto, è stata realizzata una funzione che andasse ad aggiornare tale identificativo, acquisendo le informazioni in maniera automatica. I dati, ottenuti in formato JSON, sono stati salvati all'interno di una lista successivamente convertita in file JSON attraverso il metodo `json.dump()`.

2.1.2 Dati ottenuti da PokéAPI

I dati ottenuti da PokéAPI contengono le principali informazioni riguardanti le entità che costituiscono i fondamenti del gioco Pokémon competitivo, quali Pokémon, mosse, strumenti, abilità e tipi. Sono stati ottenuti un totale di 5 dataset:

- **Species API** (file species.json 49MB, 898 righe × 9 colonne): Lista completa di tutte le specie Pokémon introdotte fino ai giochi *Pokémon Spada e Scudo*, accompagnate con le loro principali caratteristiche. Le informazioni sono state acquisite dalla pagina <https://pokeapi.co/api/v2/pokemon-species/{id}>, inserendo l'id opportuno.

ID	Name	Varieties	Generation	Evolves_from	Has_gender_diff	Is_baby	Is_legendary	Is_mythical
0 1	Bulbasaur	[{"is_default": true, "pokemon": {"name": "bul..."}]	{"name": "generation-i", "url": "https://pokea..."}	None	False	False	False	False
1 2	Ivysaur	[{"is_default": true, "pokemon": {"name": "ivy..."}]	{"name": "generation-i", "url": "https://pokea..."}	{"name": "bulbasaur", "url": "https://pokeapi..."}	False	False	False	False
2 3	Venusaur	[{"is_default": true, "pokemon": {"name": "ven..."}]	{"name": "generation-i", "url": "https://pokea..."}	{"name": "ivysaur", "url": "https://pokeapi.co..."}	True	False	False	False
3 4	Charmander	[{"is_default": true, "pokemon": {"name": "cha..."}]	{"name": "generation-i", "url": "https://pokea..."}	None	False	False	False	False
4 5	Charmeleon	[{"is_default": true, "pokemon": {"name": "cha..."}]	{"name": "generation-i", "url": "https://pokea..."}	{"name": "charmander", "url": "https://pokeapi..."}	False	False	False	False

Figura 1. Species API Dataset

- *ID*: Intero. ID del Pokémon all'interno del Pokédex Nazionale dei videogiochi di ottava generazione. L'ID è univoco per ciascuna specie.
- *Name*: Stringa. Nome della specie del Pokémon.
- *Varieties*: Lista. Contiene tutte le possibili forme alternative di una certa specie.
- *Generation*: Dizionario. Contiene le informazioni della generazione in cui è stata introdotta la specie.
- *Evolves_from*: Dizionario. Contiene le informazioni della specie da cui evolve la specie considerata.
- *Has_gender_diff*: Booleano. Indica se esiste variazione di aspetto fra Pokémon di sesso maschile e femminile.
- *Is_baby*: Booleano. Indica se è un baby Pokémon.
- *Is_legendary*: Booleano. Indica se è un Pokémon leggendario.
- *Is_mythical*: Booleano. Indica se è un Pokémon mitico.
- **Items API** (file item.json 44MB, 1606 righe × 4 colonne): Lista completa degli strumenti introdotti fino ai giochi *Pokémon Spada e Scudo*, accompagnati con le loro principali caratteristiche. Gli strumenti sono elementi fondamentali del gioco competitivo, che tramite i loro effetti possono contribuire al potenziamento di Pokémon e mosse. Le informazioni sono state acquisite dalla pagina <https://pokeapi.co/api/v2/item/{id}>, inserendo l'id opportuno.

Name	Attributes	Category	Effect
0 Master-ball	[{"name": "countable", "url": "https://pokeapi..."}, {"name": "standard-balls", "url": "https://pok..."}]	[{"effect": "Used in battle : Catches a wild..."}]	
1 Ultra-ball	[{"name": "countable", "url": "https://pokeapi..."}, {"name": "standard-balls", "url": "https://pok..."}]	[{"effect": "Used in battle : Attempts to ca..."}]	
2 Great-ball	[{"name": "countable", "url": "https://pokeapi..."}, {"name": "standard-balls", "url": "https://pok..."}]	[{"effect": "Used in battle : Attempts to ca..."}]	
3 Poke-ball	[{"name": "countable", "url": "https://pokeapi..."}, {"name": "standard-balls", "url": "https://pok..."}]	[{"effect": "Used in battle : Attempts to ca..."}]	
4 Safari-ball	[{"name": "countable", "url": "https://pokeapi..."}, {"name": "standard-balls", "url": "https://pok..."}]	[{"effect": "Used in battle : Attempts to ca..."}]	

Figura 2. Items API Dataset

- *Name*: Stringa. Nome dello strumento.
- *Attributes*: Lista. Contiene gli attributi specifici dello strumento.
- *Category*: Dizionario. Contiene le informazioni della categoria a cui appartiene lo strumento.
- *Effect*: Lista. Contiene l'effetto dello strumento, tradotto in diverse lingue.
- **Abilities API** (file abilities.json 10MB, 327 righe × 5 colonne): Lista completa delle abilità introdotte fino ai giochi *Pokémon Spada e Scudo*, accompagnate con le loro principali caratteristiche. Le abilità sono attributi speciali di ogni Pokémon. Le abilità possono agire come potenziamento, aumentando il danno di una mossa o una statistica base, possono introdurre effetti particolari come una condizione meteorologica e, in generale, possono risultare determinanti per l'esito di una battaglia. Ogni Pokémon possiede una sola abilità, determinata casualmente tra 2 o 3 disponibili, di cui una nascosta ottenibile solo tramite specifiche procedure. Le informazioni sono state acquisite dalla pagina <https://pokeapi.co/api/v2/ability/{id}>, inserendo l'id opportuno.

	Name	Generation	Is_Main_Series	Effect_entries		Pokemon
0	Stench	3	True	This Pokémon's damaging moves have a 10% chance to ...	[{"is_hidden": True, "pokemon": {"name": "gloom", "id": 100}, {"is_hidden": False, "pokemon": {"name": "vileplume", "id": 101}}]	
1	Drizzle	3	True	The weather changes to rain when this Pokémon ...	[{"is_hidden": True, "pokemon": {"name": "poliwrath", "id": 102}, {"is_hidden": False, "pokemon": {"name": "poliwrath", "id": 103}}]	
2	Speed-boost	3	True	This Pokémon's Speed rises one stage after each ...	[{"is_hidden": False, "pokemon": {"name": "yanma", "id": 104}}]	
3	Battle-armor	3	True	Moves cannot score critical hits against this ...	[{"is_hidden": True, "pokemon": {"name": "cubone", "id": 105}, {"is_hidden": False, "pokemon": {"name": "marowak", "id": 106}}]	
4	Sturdy	3	True	When this Pokémon is at full HP, any hit that ...	[{"is_hidden": False, "pokemon": {"name": "geodude", "id": 107}, {"is_hidden": True, "pokemon": {"name": "geodude", "id": 108}}]	

Figura 3. Abilities API Dataset

- *Name*: Stringa. Nome dell'abilità.
 - *Generation*: Dizionario. Contiene le informazioni della generazione in cui è stata introdotta l'abilità.
 - *Is_Main_Series*: Booleano. Indica se l'abilità è appartenente ai giochi della serie principale.

• **Moves API** (file moves.json 47MB, 327 righe × 8 colonne): Lista completa delle mosse introdotte fino ai giochi *Pokémon Spada e Scudo*, accompagnate con le loro principali caratteristiche e statistiche. Una mossa è una *skill* che un Pokémon può usare, una volta per turno, in battaglia. Un Pokémon può apprendere un massimo di quattro mosse contemporaneamente. Le informazioni sono state acquisite dalla pagina <https://pokeapi.co/api/v2/move/{id}>, inserendo l'id opportuno.

 - *Effect_entries*: Lista. Contiene l'effetto dell'abilità tradotto in diverse lingue.
 - *Pokémon*: Lista. Contiene la lista dei Pokémon che possiedono quell'abilità.

Name	Type	Power	Accuracy	PP	Damage_class	Introduced_in	Learned_by
0 pound	{"name": "normal", "url": "https://pokeapi.co...}	40.0	100.0	35	{"name": "physical", "url": "https://pokeapi.co...}	{"name": "generation-1", "url": "https://pokeapi.co...}	["name": "defairy", "url": "https://pokeapi.co..."]
1 karate-chop	{"name": "fighting", "url": "https://pokeapi.co...}	50.0	100.0	25	{"name": "physical", "url": "https://pokeapi.co...}	{"name": "generation-1", "url": "https://pokeapi.co...}	["name": "monkey", "url": "https://pokeapi.co..."]
2 double-slap	{"name": "normal", "url": "https://pokeapi.co...}	15.0	85.0	10	{"name": "physical", "url": "https://pokeapi.co...}	{"name": "generation-1", "url": "https://pokeapi.co...}	["name": "defairy", "url": "https://pokeapi.co..."]
3 comet-punch	{"name": "normal", "url": "https://pokeapi.co...}	18.0	85.0	15	{"name": "physical", "url": "https://pokeapi.co...}	{"name": "generation-1", "url": "https://pokeapi.co...}	["name": "hitmonchan", "url": "https://pokeapi.co..."]
4 mega-punch	{"name": "normal", "url": "https://pokeapi.co...}	80.0	85.0	20	{"name": "physical", "url": "https://pokeapi.co...}	{"name": "generation-1", "url": "https://pokeapi.co...}	["name": "charmander", "url": "https://pokeapi.co..."]

Figura 4. Moves API Dataset

- *Name*: Stringa. Nome della mossa.
 - *Type*: Dizionario. Contiene le informazioni del tipo della mossa.
 - *Power*: Intero. Indica la potenza della mossa.
 - *Accuracy*: Intero. Indica la precisione della mossa.
 - *PP*: Intero. Indica il numero di utilizzi della mossa.
 - *Damage_class*: Dizionario. Contiene le informazioni circa la classe di danno della mossa.
 - *Introduced_in*: Dizionario. Contiene le informazioni della generazione in cui è stata introdotta la mossa.
 - *Learned_by*: Lista. Contiene la lista dei Pokémon che possono imparare quella mossa.

• **Types API** (file types.json 789kB, 20 righe × 3 colonne): Lista completa dei tipi Pokémon introdotti fino ai giochi *Pokémon Spada e Scudo*. Il tipo è una caratteristica peculiare di Pokémon e mosse che influenza il calcolo dei danni inflitti da una mossa a un Pokémon. Le informazioni sono state acquisite dalla pagina <https://pokeapi.co/api/v2/type/{id}>, inserendo l'id opportuno.

	Name	Introduced_in	Damage_relations
15	Dragon	{'name': 'generation-i', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/15/'} {'name': 'generation-ii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/15/'} {'name': 'generation-iii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/15/'}	{'double_damage_from': [{"name": "ice", "url": "https://pokeapi.co/api/v2/pokemon/directives/16/"}]}
16	Dark	{'name': 'generation-ii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/16/'} {'name': 'generation-iii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/16/'}	{'double_damage_from': [{"name": "fighting", "url": "https://pokeapi.co/api/v2/pokemon/directives/17/"}]}
17	Fairy	{'name': 'generation-vi', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/17/'} {'name': 'generation-vii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/17/'}	{'double_damage_from': [{"name": "poison", "url": "https://pokeapi.co/api/v2/pokemon/directives/18/"}]}
18	Unknown	{'name': 'generation-ii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/18/'} {'name': 'generation-iii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/18/'}	{'double_damage_from': [], 'double_damage_to': []}
19	Shadow	{'name': 'generation-iii', 'url': 'https://pokeapi.co/api/v2/pokemon/directives/19/'}	{'double_damage_from': [], 'double_damage_to': []}

Figura 5. Types API Dataset

- *Name*: Stringa. Nome del tipo.
 - *Introduced_in*: Dizionario. Contiene le informazioni della generazione in cui è stato introdotto il tipo.
 - *Damage_relations*: Dizionario. Contiene le relazioni di danno dello specifico tipo rispetto a tutti gli altri.

2.2 Web Scraping

La tecnica di Web Scraping è stata utilizzata per acquisire dati da sorgenti plurime:

- *Pokémon Database*[3]: uno dei *fansite* più popolari a tema videogiochi Pokémon. Il sito raccoglie tutte le informazioni più aggiornate relative alle statistiche e alle *game mechanics* dei videogiochi della serie principale.
- *Pikalytics*[4]: *fansite* che fornisce analisi e statistiche di utilizzo relative ai match competitivi disputati sulle diverse piattaforme e nei diversi formati lotta disponibili. Il sito permette ai giocatori e agli appassionati del VGC, il formato di lotta ufficiale dei campionati mondiali, di accedere alla classifica dei Pokémon più utilizzati dagli altri giocatori e a informazioni tecniche di dettaglio relative allo stato corrente del *metagame*. Per ogni Pokémon sono generati *spreadsheet* contenenti informazioni riguardo i compagni di squadra, gli oggetti, le mosse, le abilità e le statistiche più popolari. Il sito si aggiorna ogni primo del mese, con i dati mostrati che fanno riferimento alla totalità dei match registrati durante il mese precedente.
- *Serebii.net*[5]: *fansite* contenente notizie, dataset e informazioni aggiornate relative a tutti i videogiochi, al gioco di carte collezionabili e all'*anime* Pokémon.
- *VictoryRoad*[6]: lo staff di VictoryRoad è composto da giocatori di lunga data, giudici e organizzatori del Pokémon Video Game Championships. Il sito fornisce informazioni e *know-how* relativi alla sfera competitiva dei videogiochi Pokémon.
- *Bulbapedia*[7]: enciclopedia *community-driven* a tema Pokémon che costituisce una delle risorse sul tema più complete e note del web.

2.2.1 Librerie e processi implementati

Il processo di Web Scraping è stato effettuato in Python, con metodi e librerie utilizzate che variano in funzione della struttura delle pagine web nei diversi siti menzionati. Tipicamente, il processo ha visto una fase iniziale di esplorazione del sito target, al fine di verificare la presenza di informazioni utili, a cui è seguita l'ispezione della struttura del documento HTML (del *Document Object Model* del sito) mediante l'utilizzo del *developer tool* apposito del browser, con lo scopo di individuare la posizione dei dati da acquisire all'interno della struttura del documento HTML. La fase successiva ha visto l'utilizzo della libreria **Python Requests** [2], e in particolare del metodo `get()`, al fine di inviare le *GET request* agli url delle pagine target e ottenere gli specifici *Response Object*. Per il parsing e lo scraping delle pagine web, salvate come documenti HTML, è stata utilizzata la libreria **Beautiful Soup** [8]. Viene ottenuto il *BeautifulSoup object*, ovvero il documento HTML *parsed* navigabile e ricercabile utilizzando i metodi forniti dalla libreria, passando come argomento al costruttore `BeautifulSoup()` il contenuto del *Response Object* in bytes (`response.content`), e utilizzando come parser `html.parser`[9]. L'ultima fase del processo di acquisizione ha visto lo sviluppo di *script* che permettessero, in modo automatizzato, la navigazione del BeautifulSoup object, la ricerca e l'acquisizione dei dati di interesse, l'organizzazione degli stessi all'interno di *dataframe Pandas* [10] e il loro salvataggio come file .csv e .JSON. In alcuni dei siti citati (*Pokémon Database*, *Serebii.net* e *Bulbapedia*), parte dei dati sono organizzati in formato tabellare all'interno delle *HTML Tables*, elementi identificabili all'interno della struttura del documento HTML dal tag `<table>`. In questo caso le tabelle sono state lette facilmente, e salvate in *dataframe*, utilizzando la funzione `pandas.read_html()`. Invece, l'acquisizione dei dati relativi al videogioco competitivo, dal sito *Pikalytics*, ha richiesto, in fase iniziale, l'utilizzo della libreria **Selenium**[11], con *Microsoft Edge WebDriver*[12], con lo scopo di simulare l'azione di *scrolling* fino al fondo della pagina principale e permettere il caricamento completo di tutti i dati da acquisire. Sono di seguito elencati e descritti i dataset ottenuti nelle diverse fasi del processo di scraping, con eventuali commenti relativi ai procedimenti specifici.

2.2.2 Scraping da *Pokémon Database*

I dati ottenuti da *Pokémon Database* coprono tutti gli aspetti tecnici e le statistiche relative agli elementi e alle meccaniche di gioco relative ai videogiochi della serie principale e rilevanti per il gioco competitivo, nonché una serie di informazioni descrittive ausiliarie, utili a una completa descrizione degli elementi di gioco e delle relazioni tra gli stessi. L'acquisizione dei dati da questa fonte ha avuto lo scopo di integrare quanto ottenuto tramite l'API, ottenendo attributi e record aggiuntivi (i dati dell'API non includono le informazioni relative all'ultimo videogioco della serie: *Leggende Pokémon: Arceus*) e valori più affidabili per le statistiche tecniche (il sito, a differenza dell'API, è aggiornato e corretto frequentemente). Da *Pokémon Database* sono stati ottenuti un totale di 7 dataset:

- **Complete Pokémon Pokédex** (file `pokedex.csv` 128KB, 1075 righe × 12 colonne): Lista completa di tutti i Pokémon delle 8 generazioni uscite finora, accompagnati dalle loro statistiche principali. Il processo di scraping è stato applicato alla pagina <https://pokemondb.net/pokedex/all>.

#	Name	Type1	Type2	Variant	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	sprite_url
0	1	Bulbasaur	Grass	Poison	None	318	45	49	49	65	65	45 https://img.pokemondb.net/sprites/sword-shield...
1	2	Ivysaur	Grass	Poison	None	405	60	62	63	80	80	60 https://img.pokemondb.net/sprites/sword-shield...
2	3	Venusaur	Grass	Poison	None	525	80	82	83	100	100	80 https://img.pokemondb.net/sprites/sword-shield...
3	3	Venusaur	Grass	Poison	Mega Venusaur	625	80	100	123	122	120	80 https://img.pokemondb.net/sprites/sword-shield...
4	4	Charmander	Fire	None	None	309	39	52	43	60	50	65 https://img.pokemondb.net/sprites/sword-shield...

Figura 6. Complete Pokémon Pokédex Dataset

- #: Intero. ID del Pokémon all'interno del Pokédex Nazionale dei videogiochi di ottava generazione, che racchiude tutti le 905 specie di Pokémon mostrate finora. L'ID non è univoco, essendo presenti varianti della stessa specie.
- Name: Stringa. Nome della specie del Pokémon.
- Type1: Stringa. Tipo del Pokémon.
- Type2: Stringa. Secondo tipo del Pokémon (opzionale, non tutti i Pokémon lo possiedono)
- Variant: Stringa. Se non nulla, il Pokémon è una versione alternativa di una certa specie. Indica il nome della variante specifica.
- Total/HP/Attack/Defense/Sp.Atk/Sp.Def/Speed: Interi. Statistiche di base.
- sprite_url: Stringa. URL relativo allo sprite del Pokémon, ovvero all'immagine in pixel art che ne mostra le fattezze in gioco.

La maggior parte degli attributi del dataset sono stati ottenuti usando la funzione `pandas.read_html()`, estraendo i dati di interesse contenuti nella specifica tabella HTML. L'ottenimento della variabile `sprite_url`, invece, ha richiesto la navigazione del *BeautifulSoup object* e l'estrazione del valore dell'attributo dagli elementi corrispondenti. Considerazioni analoghe sono valide anche per il dataset *Items*, mentre per tutti gli altri dataset ottenuti da questo sito è stato sufficiente l'utilizzo di `pandas.read_html()`.

This is a full list of every Pokémon from all 8 generations of the Pokémon series, along with their main stats.
The table is sortable by clicking a column header, and searchable by using the controls above it.

#	Name	Type	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed
001	Bulbasaur	GRASS POISON	318	45	49	49	65	65	45
002	Ivysaur	GRASS POISON	405	60	62	63	80	80	60
003	Venusaur	GRASS POISON	525	80	82	83	100	100	80

```
<div class="rpip-scroll">
  <table id="pokedex" class="data-table sticky-header block-wide" style="opacity: 1;">
    <thead>
      <tr>
        <th>#</th>
        <th>Name</th>
        <th>Type</th>
        <th>Total</th>
        <th>HP</th>
        <th>Attack</th>
        <th>Defense</th>
        <th>Sp. Atk</th>
        <th>Sp. Def</th>
        <th>Speed</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>001</td>
        <td>Bulbasaur</td>
        <td>GRASS  
POISON</td>
        <td>318</td>
        <td>45</td>
        <td>49</td>
        <td>49</td>
        <td>65</td>
        <td>65</td>
        <td>45</td>
      </tr>
      <tr>
        <td>002</td>
        <td>Ivysaur</td>
        <td>GRASS  
POISON</td>
        <td>405</td>
        <td>60</td>
        <td>62</td>
        <td>63</td>
        <td>80</td>
        <td>80</td>
        <td>60</td>
      </tr>
      <tr>
        <td>003</td>
        <td>Venusaur</td>
        <td>GRASS  
POISON</td>
        <td>525</td>
        <td>80</td>
        <td>82</td>
        <td>83</td>
        <td>100</td>
        <td>100</td>
        <td>80</td>
      </tr>
    </tbody>
  </table>
</div>
```

Figura 7. Pagina Web target del processo di scraping utile ad acquisire il dataset Complete Pokémon Pokédex

- **Items** (file `items.csv` 125KB, 960 righe × 4 colonne): Lista completa di tutti gli oggetti dalle 8 generazioni uscite finora. Il processo di scraping è stato applicato alla pagina <https://pokemondb.net/item/all>.

	Name	Category	Effect	sprite_url
0	Ability Capsule	Hold items	A capsule that allows a Pokémon with two Abilities to have three abilities.	https://img.pokemondb.net/sprites/items/abilit...
1	Ability Patch	General items	Changes a Pokémon's ability to its Hidden Ability.	https://img.pokemondb.net/s.png
2	Ability Urge	Battle items	When used, it activates the Ability of an ally.	https://img.pokemondb.net/sprites/items/abilit...
3	Abomasite	Hold items	Enables Abomasnow to Mega Evolve during battle.	https://img.pokemondb.net/sprites/items/abomas...
4	Absolite	Hold items	Enables Absol to Mega Evolve during battle.	https://img.pokemondb.net/sprites/items/absoli...
...

Figura 8. Items Dataset

- *Name*: Stringa. Nome dell’oggetto.
- *Category*: Stringa. CATEGORIA generale dell’oggetto.
- *Effect*: Stringa. Descrizione dell’effetto dell’oggetto.
- *sprite_url*: Stringa. URL relativo allo *sprite* dell’oggetto, ovvero all’immagine in *pixel art* che ne mostra le fattezze in gioco.

- **Abilities** (file abilities.csv 17KB, 266 righe × 4 colonne): Lista delle abilità dall’ultima generazione dei giochi Pokémon, con una breve descrizione. Il processo di scraping è stato applicato alla pagina <https://pokemondb.net/ability>.

:	Name	Pokémon	Description	Gen.
0	Adaptability	16	Powers up moves of the same type.	4
1	Aerilate	2	Turns Normal-type moves into Flying-type moves.	6
2	Aftermath	10	Damages the attacker landing the finishing hit.	4
3	Air Lock	1	Eliminates the effects of weather.	3
4	Analytic	12	Boosts move power when the Pokémon moves last.	5

Figura 9. Abilities Dataset

- *Name*: Stringa. Nome dell’abilità.
- *Pokémon*: Inter. Numero di Pokémon che possono avere l’abilità specifica.
- *Description*: Stringa. Descrizione dell’effetto dell’abilità.
- *Gen.*: Inter. Generazione in cui è stata introdotta.

- **Moves** (file moves.csv 71KB, 865 righe × 9 colonne): Lista completa delle mosse da tutte le 8 generazioni dei giochi Pokémon, con una breve descrizione. La potenza, la precisione e il PP sono elencati insieme all’effetto aggiuntivo. Il processo di scraping è stato applicato alla pagina <https://pokemondb.net/move/all>.

	Name	Type	Cat.	Power	Acc.	PP	TM	Effect	Prob. (%)
0	10,000,000 Volt Thunderbolt	Electric	NaN	195	—	1	NaN	Pikachu-exclusive Z-Move. High critical hit ra...	—
1	Absorb	Grass	NaN	20	100	25	NaN	User recovers half the HP inflicted on opponent.	—
2	Accelerock	Rock	NaN	40	100	20	NaN	User attacks first.	—
3	Acid	Poison	NaN	40	100	30	NaN	May lower opponent’s Special Defense.	10
4	Acid Armor	Poison	NaN	—	—	20	NaN	Sharply raises user’s Defense.	—

Figura 10. Moves Dataset

- *Name*: Stringa. Nome della mossa.
- *Type*: Stringa. Tipo della mossa.
- *Cat.*: Stringa. CATEGORIA di danno della mossa.
- *Power*: Inter. Potenza di base della mossa.
- *Acc.*: Inter. Precisione di base della mossa.
- *PP*: Inter. Indica il numero di utilizzi della mossa.
- *TM*: Booleano. Indica se la mossa può o meno essere imparata con un oggetto di tipo *Technical Machine*.
- *Effect*: Stringa. Descrizione dell’effetto secondario correlato all’utilizzo della mossa.
- *Prob. (%)*: Inter. Probabilità che si attivi l’effetto secondario della mossa, quando utilizzata. Un valore nullo indica che l’effetto viene sempre attivato.

- **Natures** (file natures.csv 1KB, 23 righe × 5 colonne): Le nature sono una *feature* determinante nei giochi Pokémon. Le nature aumentano una statistica del 10% mentre ne riducono un’altra del 10%, ad eccezione della statistica HP. Esistono 25 nature, di cui 5 neutre (non modificano alcuna statistica). Il processo di scraping è stato applicato alla pagina <https://pokemondb.net/mechanics/natures>.

	Nature	Increases	Decreases	Likes_berrie	Dislikes_berrie
0	Adamant	Attack	Sp. Atk	Spicy	Dry
1	Bashful	Sp. Atk	Sp. Atk	Dry	Dry
2	Bold	Defense	Attack	Sour	Spicy
3	Brave	Attack	Speed	Spicy	Sweet
4	Calm	Sp. Def	Attack	Bitter	Spicy
5	Careful	Sp. Def	Sp. Atk	Bitter	Dry
6	Docile	Defense	Defense	Sour	Sour

Figura 11. Natures Dataset

- *Nature*: Stringa. Nome della natura.
- *Increases*: Stringa. Statistica incrementata.
- *Decreases*: Stringa. Statistica ridotta.
- *Likes_berrie*: Stringa. Bacca (tipo di oggetto) apprezzata dal Pokémon che possiede la specifica natura.
- *Dislikes_berrie*: Stringa. Bacca (tipo di oggetto) disprezzata dal Pokémon che possiede la specifica natura.

- **Type Chart** (file typechart.csv 2KB, 18 righe × 19 colonne). Una tabella ponte che pone in relazione il tipo della mossa usata in battaglia da un Pokémon attaccante con il tipo del Pokémon che subisce la mossa. Tale relazione definisce l’efficacia della mossa in battaglia, determinando un eventuale incremento o riduzione del danno inflitto (es. una mossa di tipo fuoco è poco efficace su un Pokémon di tipo acqua). Il processo di scraping è stato applicato alla pagina <https://pokemondb.net/type>.

Atk_Type/Pokemon_Type	Normal	Fire	Water	Electric	Grass	Ice	Fighting	Poison	Ground	Flying	Psychic	Bug	Rock	Ghost	Dragon	Dark	Steel	
0	Normal	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	½	0	NaN	NaN	½
1	Fire	NaN	½	½	NaN	2	2	NaN	NaN	NaN	NaN	2	½	NaN	½	NaN	2	
2	Water	NaN	2	½	NaN	½	NaN	NaN	NaN	2	NaN	NaN	NaN	2	NaN	½	NaN	NaN
3	Electric	NaN	NaN	2	½	½	NaN	NaN	NaN	0	2	NaN	NaN	NaN	NaN	½	NaN	NaN
4	Grass	NaN	½	2	NaN	½	NaN	NaN	½	2	½	NaN	½	2	NaN	½	NaN	½
5	Ice	NaN	½	½	NaN	2	½	NaN	NaN	2	2	NaN	NaN	NaN	NaN	2	NaN	½
6	Fighting	2.0	NaN	NaN	NaN	NaN	2	NaN	½	NaN	½	½	½	2	0	NaN	2	2
7	Poison	NaN	NaN	NaN	NaN	2	NaN	NaN	½	½	NaN	NaN	NaN	½	½	NaN	NaN	0
8	Ground	NaN	2	NaN	2	½	NaN	NaN	2	NaN	0	NaN	½	2	NaN	NaN	NaN	2
9	Flying	NaN	NaN	NaN	½	2	NaN	2	NaN	NaN	NaN	2	½	NaN	NaN	NaN	½	
10	Psychic	NaN	NaN	NaN	NaN	NaN	NaN	2	2	NaN	NaN	½	NaN	NaN	NaN	NaN	0	½
11	Bug	NaN	½	NaN	NaN	2	NaN	½	½	NaN	½	2	NaN	NaN	½	NaN	2	½
12	Rock	NaN	2	NaN	NaN	NaN	2	½	NaN	½	2	NaN	2	NaN	NaN	NaN	NaN	½
13	Ghost	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2	NaN	NaN	2	NaN	½	NaN
14	Dragon	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2	NaN	½	NaN
15	Dark	NaN	NaN	NaN	NaN	NaN	NaN	½	NaN	NaN	NaN	2	NaN	NaN	2	NaN	½	NaN
16	Steel	NaN	½	½	½	½	NaN	2	NaN	NaN	NaN	NaN	NaN	2	NaN	NaN	NaN	½

Figura 12. Type Chart - Tabella Ponte Types-Types

- *Righe*: Ogni riga della tabella è associata a un particolare tipo di mossa.
- *Colonne*: Ogni colonna della tabella (a parte la prima, indice esplicito del dataset) è associata ad un particolare tipo di Pokémon.
- *Celle*: I valori presenti nelle celle indicano il moltiplicatore applicato al danno inflitto da una mossa di tipo x quando utilizzata su un Pokémon di tipo y. Un valore nullo indica l’assenza di incremento/decremento del danno base.

2.2.3 Scraping da *Pikalytics*

Il processo di Web Scraping da *Pikalytics* ha permesso l’acquisizione dei dati relativi ai match competitivi svolti a **Marzo 2022**, con particolare riferimento a quelli disputati sulla piattaforma **Pokémon Showdown!**[13], seguendo il regolamento ufficiale di **VGC Series 12**. In questo caso, lo scopo del processo di scraping è quello di individuare, per ogni Pokémon utilizzato a Marzo 2022 durante almeno un match competitivo, il *ranking* (in termini di percentuale di utilizzo) e le informazioni relative ai Pokémon compagni di squadra, agli oggetti, alle mosse, alle abilità e alle statistiche più popolari. La struttura della pagina principale di *Pikalytics* prevede la presenza di una sezione laterale contenente una lista dei Pokémon, ordinati in termini di

utilizzo mensile, per cui cliccando sugli elementi della stessa è possibile accedere alla pagina relativa alle statistiche peculiari del singolo Pokémon. La presenza di una barra di *scrolling* per la navigazione della lista laterale impedisce il caricamento della lista completa al momento di apertura della pagina web, ottenendo, con il solo utilizzo di *Python Request* e *Beautiful Soup*, un documento HTML incompleto. È stata quindi utilizzata la libreria **Selenium**, e in particolare il modulo *Action Chains*, al fine di simulare, all'apertura della pagina web, il movimento del cursore nella porzione di pagina della lista e la pressione ripetuta del tasto TAB, al fine di eseguire lo *scrolling*, e quindi il caricamento completo della lista. Una volta ottenuto il documento HTML relativo alla pagina web completamente caricata è stato possibile sfruttare i metodi di **Beautiful Soup** per accedere ai link associati ai singoli elementi della lista e quindi alle pagine relative ai diversi Pokémon. È stato poi sviluppato uno *script* che permettesse, in modo automatizzato, l'acquisizione dei dati di interesse dai diversi elementi che caratterizzano la pagina di un singolo Pokémon, iterando il processo su tutti i link ottenuti.

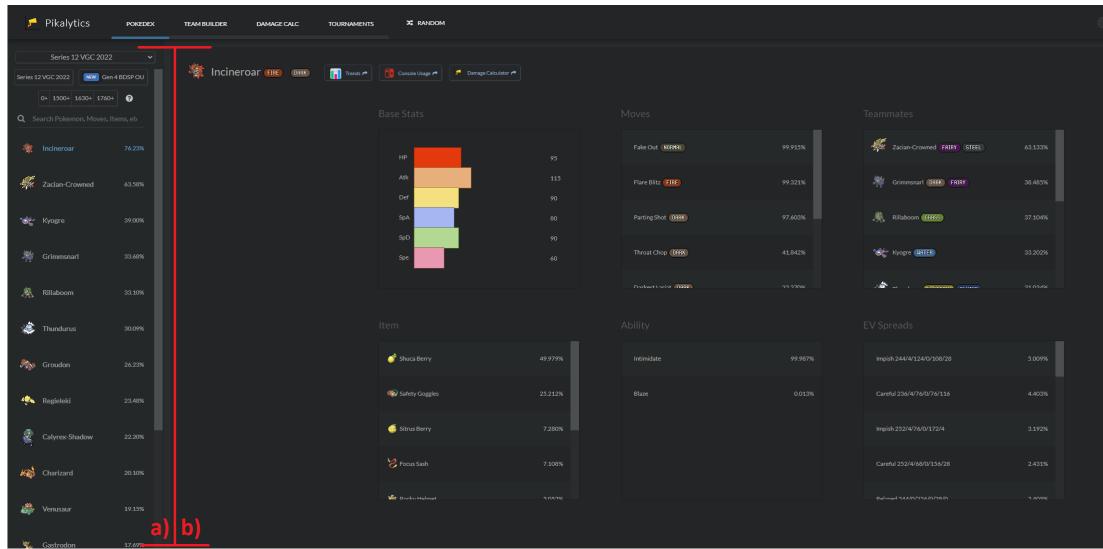


Figura 13. Pagina principale di *Pikalytics*: a) Lista di tutti i Pokémon b) Sezione relativa alle statistiche del singolo Pokémon

Da *Pikalytics* sono stati ottenuti un totale di 7 dataset, principalmente tabelle ponte:

- **Pokémon Usage** (file usage_S12VGC.csv 8KB, 342 righe × 4 colonne): Statistiche di utilizzo dei Pokémon durante Marzo 2022.

	Pokemon	Monthly Usage (k)	Usage Percent (%)	Monthly Rank
0	Zacian-Crowned	1088	65	1
1	Incineroar	1124	59	2
2	Kyogre	744	35	3
3	Grimmsnarl	577	30	4
4	Regieleki	603	29	5

Figura 14. Pokémon Usage Dataset

- **Pokémon:** Stringa. Nome del Pokémon.
- **Monthly Usage (k):** Intero. Numero di migliaia team che hanno visto schierato il Pokémon specifico.
- **Usage Percent (%):** Intero. Percentuale di team (sul

totale dei team schierati) che hanno visto l'utilizzo del Pokémon specifico.

- **Monthly Rank:** Intero. Posizionamento nella classifica dei Pokémon più utilizzati.

- **Moves Usage** (file moves_S12VGC.csv 140KB, 3618 righe × 4 colonne): Mosse più utilizzate, per ogni Pokémon, nel mese di Marzo 2022. È disponibile un massimo di 8 mosse fra quelle più utilizzate per ciascun Pokémon (quelle meno utilizzate sono accorpate e classificate come *Other*).

Pokemon	Move	Type	Use_Percentage (%)
0 Zacian-Crowned	Behemoth Blade	steel	99.996%
1 Zacian-Crowned	Protect	normal	98.672%
2 Zacian-Crowned	Sacred Sword	fighting	84.154%
3 Zacian-Crowned	Play Rough	fairy	64.487%
4 Zacian-Crowned	Substitute	normal	30.274%

Figura 15. Moves Usage - Tabella Ponte Pokémon/Moves

- *Pokémon*: Stringa. Nome del Pokémon.
- *Move*: Stringa. Nome della mossa.
- *Type*: Stringa. Tipo della mossa
- *Use_Percentage (%)*: Intero. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con la mossa specifica sul totale delle volte in cui è stato schierato.

- **Teammates Usage** (file teammates_S12VGC.csv 112KB, 3422 righe × 3 colonne): Compagni schierati più frequentemente, al fianco di ogni Pokémon, nel mese di Marzo 2022.

Pokemon	Teammate	Use_Percentage (%)
0 Zacian-Crowned	Incineroar	54.189%
1 Zacian-Crowned	Kyogre	31.467%
2 Zacian-Crowned	Grimmsnarl	31.148%
3 Zacian-Crowned	Rillaboom	29.083%
4 Zacian-Crowned	Gastrodon	24.347%

Figura 16. Teammates Usage - Tabella Ponte Pokémon/Pokémon

- *Pokémon*: Stringa. Nome del Pokémon.
- *Teammate*: Stringa. Nome del Pokémon compagno di squadra.
- *Use_Percentage (%)*: Intero. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con lo specifico compagno sul totale delle volte in cui è stato schierato.

- **Items Usage** (file items_S12VGC.csv 62KB, 1892 righe × 3 colonne): Oggetti più equipaggiati, per ogni Pokémon, nel mese di Marzo 2022.

Pokemon	Item	Use_Percentage (%)
0 Zacian-Crowned	Rusted Sword	100.000%
1 Incineroar	Shuca Berry	36.082%
2 Incineroar	Safety Goggles	23.390%
3 Incineroar	Sitrus Berry	14.031%
4 Incineroar	Focus Sash	8.994%
...

Figura 17. Items Usage - Tabella Ponte Pokémon/Items

- *Pokémon*: Stringa. Nome del Pokémon.
- *Items*: Stringa. Nome dell'oggetto.
- *Use_Percentage (%)*: Intero. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con lo specifico oggetto equipaggiato sul totale delle volte in cui è stato schierato.

- **Abilities Usage** (file abilities_S12VGC.csv 25KB, 773 righe × 3 colonne): Abilità assegnate più frequentemente ad ogni Pokémon schierato nel mese di Marzo 2022.

	Pokemon	Ability	Use_Percentage (%)
0	Zacian-Crowned	Intrepid Sword	100.000%
1	Incineroar	Intimidate	99.927%
2	Incineroar	Blaze	0.073%
3	Kyogre	Drizzle	100.000%
4	Grimmsnarl	Prankster	99.959%

Figura 18. Abilities Usage - Tabella Ponte Pokémon/Abilities

- *Pokémon*: Stringa. Nome del Pokémon.
- *Ability*: Stringa. Nome dell'abilità.
- *Use_Percentage (%)*: Intero. Percentuale ottenuta dal

rapporto tra il numero di volte in cui il Pokémon è stato schierato con assegnata la specifica abilità sul totale delle volte in cui è stato schierato.

- **Natures e EVs_spread Usage** (file EV_spread_S12VGC.csv 307KB, 6819 righe × 4 colonne): Nature assegnate più frequentemente ad ogni Pokémon schierato nel mese di Marzo 2022. Per ognuna delle nature più comuni, sono acquisite anche le distribuzioni dei punti EVs (nelle diverse statistiche) più frequenti tra i giocatori competitivi. Gli EVs (*Effort Values*) sono punti aggiuntivi accumulabili nelle diverse statistiche di base, utili al potenziamento delle statistiche stesse. Ogni Pokémon può ricevere un investimento massimo di 510 EVs, di cui massimo 255 EVs sulla singola statistica. Con *EV spread* si intende la distribuzione di questi punti aggiuntivi tra le diverse statistiche di base.

	Pokemon	Nature	HP/Atk/Def/SpA/SpD/Spe	Use_Percentage (%)
0	Zacian-Crowned	Jolly	0/252/0/0/4/252	10.388%
1	Zacian-Crowned	Adamant	252/28/4/0/12/212	4.504%
2	Zacian-Crowned	Jolly	4/252/0/0/0/252	4.419%
3	Zacian-Crowned	Adamant	188/164/4/0/4/148	2.891%
4	Zacian-Crowned	Jolly	0/252/4/0/0/252	2.456%

Figura 19. Natures Usage - Tabella Ponte Pokémon/Natures

- *Pokémon*: Stringa. Nome del Pokémon.
- *Nature*: Stringa. Nome della natura.
- *HP/Atk/Def/SpA/SpD/Spe*: Stringa. *EV Spread*: distribuzione dei punti EVs nelle diverse statistiche.

– *Use_Percentage (%)*: Intero. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con assegnata la specifica coppia *Nature/EVs Spread* sul totale delle volte in cui è stato schierato.

2.2.4 Scraping da *VictoryRoad*

Il processo di Web Scraping da *VictoryRoad* ha avuto come target specifico la pagina <https://victoryroadvgc.com/2020-season/>, contenente il regolamento ufficiale e le informazioni generali riguardanti la stagione 2020-2022 del gioco competitivo Pokémon (**2020-22 Play! Pokémon Season**). Le stagioni (che tipicamente durano un anno, a differenza della corrente che ha una durata di tre anni a causa della pandemia da COVID-19) identificano un periodo di tempo durante il quale i giocatori competitivi possono partecipare a eventi online e dal vivo accumulando, posizionandosi nei primi posti, i CP (*Championship Points*). Ogni stagione termina con il mondiale (Pokémon World Championship, WCS), torneo su invito a cui possono accedere solamente i giocatori che hanno raggiunto un sufficiente numero di CP. Le stagioni sono suddivise temporalmente in *Series*, ognuna delle quali prevede uno specifico *ruleset*. Attualmente la serie in corso è la Series 12 (1 Febbraio – 31 Agosto, 2022) e quindi il formato ufficiale (le cui regole sono il target di questa fase del processo di scraping) prende il nome di **Series 12 VGC 2022**. Il processo ha visto sia l'utilizzo delle librerie **Python Request** e **Beautiful Soup**, che una fase di copia-incolla manuale, e ha portato a ottenere un totale di 7 liste:

Name	Size	Value
gigantamax_allowed_list	32	['Venusaur', 'Charizard', 'Blastoise', 'Butterfree', 'Pikachu', 'Meowth', 'Machamp', 'Gengar', 'K
banned	14	['Mew', 'Celebi', 'Jirachi', 'Victini', 'Keldeo (both forms)', 'Genesect (all forms)', 'Diancie', '<...>
crown_tundra_permitted	209	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2 <...> 0,
galar_permitted	397	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2 <...> 8,
isle_of_armor_permitted	210	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2 <...> 1,
national_permitted	48	[243, 244, 245, 252, 253, 254, 255, 256, 257, 258, 259, 260, 380, 381, 480, 481, 482, 485, 488,
national_restricted	23	[150, 249, 250, 382, 383, 384, 483, 484, 487, 643, 644, 646, 716, 717, 718, 789, 790, 791, 792,

Figura 20. Liste Pokémon regolamento 2020-22 Play! Pokémon Season

- *Banned Pokemon:* Pokémon non ammessi dal regolamento. Lista di stringhe. I Pokémon proibiti sono identificati con il nome della loro specie.
- *Permitted Pokemon:* Pokémon ammessi dal regolamento.
 - *Galar Permitted:* Lista di interi. ID dei Pokémon ammessi con riferimento all'ID dei Pokémon nel Pokédex regionale della regione di *Galar*, regione principale del gioco *Pokémon Spada e Scudo*.
 - *Isle of Armor Permitted:* Lista di interi. ID dei Pokémon ammessi con riferimento all'ID dei Pokémon nel Pokédex regionale della regione di *Isle of Armor*, introdotta con l'omonima espansione del gioco *Pokémon Spada e Scudo*.
 - *The Crown Tundra Permitted:* Lista di interi. ID dei Pokémon ammessi con riferimento all'ID dei Pokémon nel Pokédex regionale della regione di *The Crown Tundra*, introdotta con l'omonima espansione del gioco *Pokémon Spada e Scudo*.
 - *National Permitted:* Lista di interi. ID dei Pokémon ammessi con riferimento all'ID del Pokémon all'interno del Pokédex Nazionale dei videogiochi di ottava generazione, che racchiude tutti le 905 specie di Pokémon mostrate finora.
- *Restricted Pokemon:* È permessa al più la presenza di un solo Pokémon di questa lista nella propria squadra. Lista di interi. ID dei Pokémon con riferimento all'ID del Pokémon all'interno del Pokédex Nazionale.
- *Allowed Gigantamax Pokemon:* la meccanica di gioco peculiare dell'attuale stagione di gioco competitivo è la *Dynamax*, una temporanea trasformazione dei Pokémon durante una battaglia che ne modifica le dimensioni e gli HP. La *Gigantamax* è una tipologia speciale di trasformazione *Dynamax*, caratteristica di 32 specie di Pokémon. La lista contiene i nomi dei Pokémon la cui forma *Gigantamax* è ammessa dal regolamento. Lista di stringhe, i Pokémon sono identificati con il nome della loro specie.

2.2.5 Scraping da serebii.net

Nel regolamento del **eries 12 VGC 2022**, in particolare per quanto riguarda le liste dei *Permitted Pokemon*, si fa riferimento a parte dei Pokémon permessi attraverso il loro ID nel Pokédex Nazionale (ottenuto precedentemente dallo scraping di *Pokémon Database* e salvato come **Complete Pokémon Pokédex**), per cui è nota una corrispondenza tra ID e nome del Pokémon, mentre la restante parte attraverso il loro ID nei tre Pokédex regionali, che caratterizzano le diverse regioni del videogioco *Pokémon Spada e Scudo*. Obiettivo di questa fase del processo di scraping è quello di ottenere i tre Pokédex regionali (*Galar*, *Isle of Armor* e *The Crown Tundra*) al fine di permettere, in seguito, di associare ad ogni ID della lista il nome completo di un Pokémon, anche per quei Pokémon non citati con il loro ID nel Pokédex Nazionale, e eventualmente, integrare **Complete Pokémon Pokédex** con i Pokémon in esso non presenti. La maggior parte degli attributi dei tre dataset sono stati ottenuti usando la funzione `pandas.read_html()`, estraendo e manipolando i dati di interesse contenuti nella specifica tabella HTML. L'ottenimento della variabile `sprite_url` ha invece richiesto la navigazione del *BeautifulSoup object* e l'estrazione del valore dell'attributo dagli elementi corrispondenti.

- **Galar Pokédex** (file `swordandshield_pokedex.csv` 54KB, 400 righe × 10 colonne): Lista completa di tutti i Pokémon della regione principale di *Pokémon Spada e Scudo*, accompagnati dalle loro statistiche principali. Il processo di scraping è stato applicato alla pagina https://www.serebii.net/swordshield/galar_pokedex.shtml.

No.	Name	Abilities	HP	Att	Def	S.Att	S.Def	Spd	sprite_url
0 #001	Grookey サルノリ	Overgrow Grassy Surge	50	65	50	40	40	65	https://www.serebii.net/swordshield/pokemon/s...
1 #002	Thwackey パチンキー	Overgrow Grassy Surge	70	85	70	55	60	80	https://www.serebii.net/swordshield/pokemon/s...
2 #003	Rillaboom ゴリランダ	Overgrow Grassy Surge	100	125	90	60	70	85	https://www.serebii.net/swordshield/pokemon/s...
3 #004	Scorbunny ヒバニー	Blaze Libero	50	71	40	40	40	69	https://www.serebii.net/swordshield/pokemon/s...
4 #005	Raboot ラビット	Blaze Libero	65	86	60	55	60	94	https://www.serebii.net/swordshield/pokemon/s...

Figura 21. Galar Pokédex Dataset

- #: Stringa. ID univoco del Pokémon all'interno del Pokédex di *Galar*.
 - Name: Stringa. Nome della specie del Pokémon.
 - Abilities: Stringa. Abilità peculiari della specie.
 - HP/Att/Def/S.Atk/S.Def/Spd: Interi. Statistiche di base.
 - sprite_url: Stringa. URL relativo allo *sprite* del Pokémon, ovvero all'immagine in *pixel art* che ne mostra le fattezze in gioco.
- **Isle of Armor Pokédex** (file isleofarmor_pokedex.csv 29KB, 211 righe × 10 colonne): Lista completa di tutti i Pokémon della regione *Isle of Armor* dell'espansione di *Pokémon Spada e Scudo*, accompagnati dalle loro statistiche principali. Il processo di scraping è stato applicato alla pagina <https://www.serebii.net/swordshield/isleofarmordex.shtml>.

No.	Name	Abilities	HP	Att	Def	S.Att	S.Def	Spd	sprite_url
0 #001	Slowpoke ヤドン	Gluttony Own Tempo Regenerator	90	65	65	40	40	15	https://www.serebii.net/swordshield/pokemon/s...
1 #002	Slowbro ヤドラン	Quick Draw Own Tempo Regenerator	95	100	95	100	70	30	https://www.serebii.net/swordshield/pokemon/s...
2 #003	Slowking ヤドキンダ	Curious Medicine Own Tempo Regenerator	95	65	80	110	110	30	https://www.serebii.net/swordshield/pokemon/s...
3 #004	Buneary ミミロル	Run Away Klutz Limber	55	66	44	44	56	85	https://www.serebii.net/swordshield/pokemon/s...
4 #005	Lopunny ミミロップ	Cute Charm Klutz Limber	65	76	84	54	96	105	https://www.serebii.net/swordshield/pokemon/s...

Figura 22. Isle of Armor Pokédex Dataset

- #: Stringa. ID univoco del Pokémon all'interno del Pokédex di *Isle of Armor*.
 - Name: Stringa. Nome della specie del Pokémon.
 - Abilities: Stringa. Abilità peculiari della specie.
 - HP/Att/Def/S.Atk/S.Def/Spd: Interi. Statistiche di base.
 - sprite_url: Stringa. URL relativo allo *sprite* del Pokémon, ovvero all'immagine in *pixel art* che ne mostra le fattezze in gioco.
- **The Crown Tundra Pokédex** (file thecrowntundra_pokedex.csv 28KB, 210 righe × 10 colonne): Lista completa di tutti i Pokémon della regione *The Crown Tundra* dell'espansione di *Pokémon Spada e Scudo*, con le loro statistiche principali. Il processo di scraping è stato applicato alla pagina <https://www.serebii.net/swordshield/thecrowntundradex.shtml>.

No.	Name	Abilities	HP	Att	Def	S.Att	S.Def	Spd	sprite_url
0 #001	Snom エキハミ	Shield Dust Ice Scales	30	25	35	45	30	20	https://www.serebii.net/swordshield/pokemon/s...
1 #002	Frosmooth モスノウ	Shield Dust Ice Scales	70	65	60	125	90	65	https://www.serebii.net/swordshield/pokemon/s...
2 #003	Wooloo ウールー	Fluffy Run Away Bulletproof	42	40	55	40	45	48	https://www.serebii.net/swordshield/pokemon/s...
3 #004	Dubwool バイウールー	Fluffy Steadfast Bulletproof	72	80	100	60	90	88	https://www.serebii.net/swordshield/pokemon/s...
4 #005	Skwovet ホシガリス	Cheek Pouch Gluttony	70	55	55	35	35	25	https://www.serebii.net/swordshield/pokemon/s...

Figura 23. The Crown Tundra Pokédex Dataset

- #: Stringa. ID univoco del Pokémon all'interno del Pokédex di *The Crown Tundra*.
- Name: Stringa. Nome della specie del Pokémon.
- Abilities: Stringa. Abilità peculiari della specie.
- HP/Att/Def/S.Atk/S.Def/Spd: Interi. Statistiche di base.
- sprite_url: Stringa. URL relativo allo *sprite* del Pokémon, ovvero all'immagine in *pixel art* che ne mostra le fattezze in gioco.

2.2.6 Scraping da Bulbapedia

Come illustrato nella sezione 2.2.4, 32 specie di Pokémon sono in grado di trasformarsi, temporaneamente in battaglia, nella propria variante *Gigantamax*. Ogni Pokémon in forma *Gigantamax* possiede una mossa esclusiva che prende il nome di *G-Max Move*. Obiettivo di questa ultima fase di scraping è stato ottenere una lista dei Pokémon in grado di avere forma *Gigantamax* e delle *G-Max Move* a essi associate. La maggior parte degli attributi dei due dataset sono stati ottenuti usando la funzione `pandas.read_html()`, estraendo e manipolando i dati di interesse contenuti nella specifica tabella HTML. L'ottenimento della variabile `image_url` del dataset **G-Max Move** ha invece richiesto la navigazione del *BeautifulSoup object* e l'estrazione del valore dell'attributo dagli elementi corrispondenti.

- **G-Max Moves** (file `gmax_moves.csv` 6KB, 32 righe × 5 colonne): Lista completa di tutte le *G-Max Move*. Il processo di scraping è stato applicato alla pagina https://bulbapedia.bulbagarden.net/wiki/G-Max_Move.

	Max Move	Gigantamax Pokémon	Type	Additional effect	image_url
0	G-Max Vine Lash	Venusaur	Grass	Inflicts damage for four turns on non-Grass-type...	https://archives.bulbagarden.net/media/upload/...
1	G-Max Wildfire	Charizard	Fire	Inflicts damage for four turns on non-Fire-type...	https://archives.bulbagarden.net/media/upload/...
2	G-Max Cannonade	Blastoise	Water	Inflicts damage for four turns on non-Water-type...	https://archives.bulbagarden.net/media/upload/...
3	G-Max Befuddle	Butterfree	Bug	Inflicts poison, paralysis, or sleep on all opp...	https://archives.bulbagarden.net/media/upload/...
4	G-Max Volt Crash	Pikachu	Electric	Paralyzes all opponents	https://archives.bulbagarden.net/media/upload/...
5	G-Max Gold Rush	Meowth	Normal	Scatters coins on the ground that are picked up...	https://archives.bulbagarden.net/media/upload/...
6	G-Max Chi Strike	Machamp	Fighting	Pumps up the user and its allies, raising thei...	https://archives.bulbagarden.net/media/upload/...
7	G-Max Terror	Gengar	Ghost	Prevents the opponent from escaping or being re...	https://archives.bulbagarden.net/media/upload/...

Figura 24. G-Max Moves Dataset

- *Max Move*: Stringa. Nome della *G-Max Move*.
- *Gigantamax Pokémon*: Stringa. Nome della specie del Pokémon che può apprenderla.
- *Type*: Stringa. Tipo della *G-Max Move*.
- *Additional effect*: Stringa. Breve descrizione dell'effetto aggiuntivo al danno.
- *image.url*: Stringa. URL relativo allo *sprite* del Pokémon *Gigantamax*, ovvero all'immagine in *pixel art* che ne mostra le fattezze in gioco.

- **Gigantamax Pokémon** (file `gigantamax_pkmn.csv` 3KB, 32 righe × 5 colonne): Lista completa di tutti i Pokémon *Gigantamax*. Il processo di scraping è stato applicato alla pagina <https://bulbapedia.bulbagarden.net/wiki/Gigantamax>.

	Pokémon	Type	Height	G-Max Move	G-Max Move Type
0	Charizard	Fire Flying	9'10"+(28.0+ m)	G-Max Wildfire	Fire
1	Butterfree	Bug Flying	5'09"+(17.0+ m)	G-Max Befuddle	Bug
2	Pikachu	Electric	6'11"+(21.0+ m)	G-Max Volt Crash	Electric
3	Meowth	Normal	10'03"+(33.0+ m)	G-Max Gold Rush	Normal
4	Machamp	Fighting	8'00"+(25.0+ m)	G-Max Chi Strike	Fighting
5	Gengar	Ghost Poison	6'07"+(20.0+ m)	G-Max Terror	Ghost

Figura 25. Gigantamax Pokémon Dataset

- *Pokémon*: Stringa. Nome della specie del Pokémon che può trasformarsi in *Gigantamax*.
- *Type*: Stringa. Tipo/i del Pokémon non trasformato.
- *Height*: Stringa. Altezza Pokémon dopo la trasformazione.
- *G-Max Move*: Stringa. Nome della *G-Max Move* peculiare.
- *Type*: Stringa. Tipo della *G-Max Move*.

3. Data Cleansing, Integration ed Enrichment

Il processo di data cleansing, integration ed enrichment ha permesso, partendo da un totale di 23 dataset ottenuti nelle diverse fasi di acquisizione, di ottenere i 6 dataset e le 13 tabelle ponte finali, pronti per essere importati in Neo4J.

3.1 Librerie utilizzate

Anche questa parte del progetto è stata svolta utilizzando Python. Le operazioni di manipolazione, pulizia e unione tra dataframe sono state effettuate mediante l'ausilio delle funzioni e dei metodi della libreria Pandas. Qualora l'unione tra dataset diversi

non fosse possibile mediante match esatto tra i valori di una variabile comune, al fine di eseguire il *record linkage* tra i dataset, è stata utilizzata la libreria **Python Record Linkage Toolkit** [14]. Essendo i record di buona parte dei dataset identificati univocamente dal loro nome (es. nome del Pokémon, della mossa, ...), il processo di record linkage è stato principalmente di *fuzzy matching* e sono stati utilizzati diversi algoritmi per il calcolo della similarità tra stringhe in funzione delle esigenze specifiche.

3.2 Dataset da API e da scraping di *Pokémon Database*: Integrazione

3.2.1 Data Cleaning

I dati ottenuti tramite API nella sezione 2.1 hanno richiesto una serie di modifiche per prepararli alle successive operazioni di integrazione.

L'aspetto che ha richiesto maggiore attenzione è stata la gestione degli attributi di tipo lista o dizionario. Ciascuno di questi oggetti contiene al suo interno il nome o la descrizione dell'attributo in questione più una serie di informazioni aggiuntive ad esso correlate, quali riferimenti a url dell'API o traduzioni della stessa informazione in altre lingue. Tuttavia, essendo interessati esclusivamente al valore della prima chiave, ossia il nome o la descrizione, sono state estratte solo le informazioni necessarie, normalizzando tali attributi ed eliminando le informazioni superflue. Tale operazione è stata effettuata per tutti gli attributi di questo tipo presenti nelle varie tabelle, ad eccezione di quelle che rappresentano una relazione, che sono state trattate in un secondo momento. Le tabelle, in seguito alla normalizzazione, assumono la seguente forma:

ID	Name	Varieties	Generation	Evolves_from	Has_gender_diff	Rarity
0	1 Bulbasaur	[{"is_default": True, "pokemon": {"name": "bul...", "id": 1, "species": "bulbasaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 45, "attack": 45, "defense": 45, "special_attack": 60, "special_defense": 60, "speed": 45}, "height": 0.7, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "ivysaur", "id": 2, "species": "ivysaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "venusaur", "id": 3, "species": "venusaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}], "id": 1, "species": "bulbasaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 45, "attack": 45, "defense": 45, "special_attack": 60, "special_defense": 60, "speed": 45}, "height": 0.7, "weight": 6.9, "order": 1, "is_main_series": true}, "name": "Bulbasaur", "rarity": "Common", "generation": 1, "evolves_from": null, "has_gender_diff": false}, {"1": {"is_default": True, "pokemon": {"name": "bulbasaur", "id": 1, "species": "bulbasaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 45, "attack": 45, "defense": 45, "special_attack": 60, "special_defense": 60, "speed": 45}, "height": 0.7, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "ivysaur", "id": 2, "species": "ivysaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "venusaur", "id": 3, "species": "venusaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}], "id": 2, "species": "ivysaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, "name": "Ivysaur", "rarity": "Common", "generation": 1, "evolves_from": "Bulbasaur", "has_gender_diff": false}, {"2": {"is_default": True, "pokemon": {"name": "ivysaur", "id": 2, "species": "ivysaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "venusaur", "id": 3, "species": "venusaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}], "id": 3, "species": "venusaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}, "name": "Venusaur", "rarity": "Common", "generation": 1, "evolves_from": "Ivysaur", "has_gender_diff": true}, {"3": {"is_default": True, "pokemon": {"name": "venusaur", "id": 3, "species": "venusaur", "type": ["grass", "poison"], "abilities": [{"name": "overgrow", "hidden": false}, {"name": "chlorophyll", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "charmander", "id": 4, "species": "charmander", "type": ["fire"], "abilities": [{"name": "flame-thrown", "hidden": false}, {"name": "heat-up", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "charmeleon", "id": 5, "species": "charmleon", "type": ["fire"], "abilities": [{"name": "flame-thrown", "hidden": false}, {"name": "heat-up", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}], "id": 4, "species": "charmander", "type": ["fire"], "abilities": [{"name": "flame-thrown", "hidden": false}, {"name": "heat-up", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, "name": "Charmander", "rarity": "Common", "generation": 1, "evolves_from": null, "has_gender_diff": false}, {"4": {"is_default": True, "pokemon": {"name": "charmander", "id": 4, "species": "charmander", "type": ["fire"], "abilities": [{"name": "flame-thrown", "hidden": false}, {"name": "heat-up", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_default": False, "pokemon": {"name": "charmleon", "id": 5, "species": "charmleon", "type": ["fire"], "abilities": [{"name": "flame-thrown", "hidden": false}, {"name": "heat-up", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}], "id": 5, "species": "charmleon", "type": ["fire"], "abilities": [{"name": "flame-thrown", "hidden": false}, {"name": "heat-up", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, "name": "Charmeleon", "rarity": "Common", "generation": 1, "evolves_from": "Charmander", "has_gender_diff": false}}]				

Figura 26. Dataset Species API dopo la manipolazione

Rispetto al dataset **Species API**, gli attributi *Is_baby*, *Is_legendary* e *Is_mythical* sono stati riassunti nell'attributo categorico *Rarity*, che può assumere come valori *Common*, *Baby*, *Legendary* e *Mythical*.

Una ulteriore operazione eseguita sul dataset **Abilities API** è stata quella di mantenere solo le abilità disponibili nei giochi della serie principale, cioè le sole utilizzabili anche nel gioco competitivo. Pertanto, sono state eliminate le righe con valore *False* rispetto all'attributo *Is_main_series*. Tale attributo è stato poi rimosso.

ID	Name	Generation	Effect_entries	Pokemon
0	Stench	3	This Pokémon's damaging moves have a 10% chance...	[{"is_hidden": true, "pokemon": {"name": "gloom", "id": 1, "species": "gloom", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 45, "attack": 45, "defense": 45, "special_attack": 60, "special_defense": 60, "speed": 45}, "height": 0.7, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "vileplume", "id": 2, "species": "vileplume", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "misdreavus", "id": 3, "species": "misdreavus", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}], "id": 1, "species": "gloom", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 45, "attack": 45, "defense": 45, "special_attack": 60, "special_defense": 60, "speed": 45}, "height": 0.7, "weight": 6.9, "order": 1, "is_main_series": true}, "name": "Stench", "rarity": "Common", "generation": 3, "evolves_from": null, "has_gender_diff": false}, {"1": {"is_hidden": true, "pokemon": {"name": "gloom", "id": 1, "species": "gloom", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 45, "attack": 45, "defense": 45, "special_attack": 60, "special_defense": 60, "speed": 45}, "height": 0.7, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "vileplume", "id": 2, "species": "vileplume", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "misdreavus", "id": 3, "species": "misdreavus", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}], "id": 2, "species": "vileplume", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 60, "attack": 60, "defense": 60, "special_attack": 80, "special_defense": 80, "speed": 60}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, "name": "Drizzle", "rarity": "Common", "generation": 3, "evolves_from": null, "has_gender_diff": false}, {"2": {"is_hidden": false, "pokemon": {"name": "misdreavus", "id": 3, "species": "misdreavus", "type": ["dark", "ghost"], "abilities": [{"name": "dark-pulse", "hidden": false}, {"name": "dark-pulse", "hidden": true}], "stats": {"hp": 85, "attack": 85, "defense": 85, "special_attack": 100, "special_defense": 100, "speed": 85}, "height": 1.5, "weight": 30.0, "order": 3, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "speed-boost", "id": 4, "species": "speed-boost", "type": ["normal"], "abilities": [{"name": "speed-boost", "hidden": false}, {"name": "speed-boost", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "battle-armor", "id": 5, "species": "battle-armor", "type": ["normal"], "abilities": [{"name": "speed-boost", "hidden": false}, {"name": "speed-boost", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}], "id": 3, "species": "speed-boost", "type": ["normal"], "abilities": [{"name": "speed-boost", "hidden": false}, {"name": "speed-boost", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, "name": "Speed-boost", "rarity": "Common", "generation": 3, "evolves_from": null, "has_gender_diff": false}, {"3": {"is_hidden": false, "pokemon": {"name": "battle-armor", "id": 5, "species": "battle-armor", "type": ["normal"], "abilities": [{"name": "speed-boost", "hidden": false}, {"name": "speed-boost", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "sturdy", "id": 6, "species": "sturdy", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "battle-armor", "id": 5, "species": "battle-armor", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}], "id": 4, "species": "sturdy", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, "name": "Sturdy", "rarity": "Common", "generation": 3, "evolves_from": null, "has_gender_diff": false}, {"4": {"is_hidden": false, "pokemon": {"name": "battle-armor", "id": 5, "species": "battle-armor", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 58, "attack": 70, "defense": 65, "special_attack": 50, "special_defense": 40, "speed": 32}, "height": 1.0, "weight": 13.0, "order": 2, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "sturdy", "id": 6, "species": "sturdy", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, {"is_hidden": false, "pokemon": {"name": "sturdy", "id": 6, "species": "sturdy", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}], "id": 5, "species": "sturdy", "type": ["normal"], "abilities": [{"name": "sturdy", "hidden": false}, {"name": "sturdy", "hidden": true}], "stats": {"hp": 39, "attack": 52, "defense": 44, "special_attack": 40, "special_defense": 30, "speed": 25}, "height": 0.5, "weight": 6.9, "order": 1, "is_main_series": true}, "name": "Sturdy", "rarity": "Common", "generation": 3, "evolves_from": null, "has_gender_diff": false}}]

Figura 27. Dataset Abilities API dopo la manipolazione

Le altre 3 tabelle non hanno richiesto operazioni aggiuntive:

ID	Name	Attributes	Category	Effect
0	Master-ball	[countable, consumable, usable-in-battle, hold...]	Standard-balls	Catches a wild Pokémon every time.
1	Ultra-ball	[countable, consumable, usable-in-battle, hold...]	Standard-balls	Tries to catch a wild Pokémon. Success rate i...
2	Great-ball	[countable, consumable, usable-in-battle, hold...]	Standard-balls	Tries to catch a wild Pokémon. Success rate i...
3	Poke-ball	[countable, consumable, usable-in-battle, hold...]	Standard-balls	Tries to catch a wild Pokémon.
4	Safari-ball	[countable, consumable, usable-in-battle, hold...]	Standard-balls	Tries to catch a wild Pokémon in the Great Mar...

Figura 28. Dataset Items API dopo la manipolazione

	Name	Type	Power	Accuracy	PP	Damage_class	Introduced_in	Learned_by
0	Pound	Normal	40.0	100.0	35.0	Physical	1	[{"name": "clefairy", "url": "https://pokeapi.co..."}]
1	Karate-chop	Fighting	50.0	100.0	25.0	Physical	1	[{"name": "mankey", "url": "https://pokeapi.co..."}]
2	Double-slap	Normal	15.0	85.0	10.0	Physical	1	[{"name": "clefairy", "url": "https://pokeapi.co..."}]
3	Comet-punch	Normal	18.0	85.0	15.0	Physical	1	[{"name": "hitmonchan", "url": "https://pokeapi.co..."}]
4	Mega-punch	Normal	80.0	85.0	20.0	Physical	1	[{"name": "charmander", "url": "https://pokeapi.co..."}]

Figura 29. Dataset Moves API dopo la manipolazione

	Name	Introduced_in	Damage_relations
0	Normal	1	{"double_damage_from": [{"name": "fighting", "ur..."}]}
1	Fighting	1	{"double_damage_from": [{"name": "flying", "ur..."}]}
2	Flying	1	{"double_damage_from": [{"name": "rock", "ur..."}]}
3	Poison	1	{"double_damage_from": [{"name": "ground", "ur..."}]}
4	Ground	1	{"double_damage_from": [{"name": "water", "ur..."}]}

Figura 30. Dataset Types API dopo la manipolazione

Come si può notare dalle figure precedenti, i valori degli attributi *Generation* e *Introduced_in* sono stati trasformati in valori numerici.

3.2.2 Relazioni

Buona parte delle informazioni relative alle relazioni tra le diverse entità sono state acquisite tramite scraping, portando all’ottenimento di opportune tabelle ponte. Pertanto, è stata presa la decisione di trasformare in tabelle ponte anche le relazioni racchiuse nei dataset acquisiti mediante API. Come già citato nella sezione 3.2.1, infatti, alcuni attributi di tipo lista dei dataset ottenuti tramite API rappresentano relazioni a noi utili. Nello specifico, le relazioni trattate in questo modo sono quella tra abilità e Pokémon che le posseggono e quella fra mosse e Pokémon in grado di apprenderle. Come si può vedere nel seguente esempio, tali relazioni sono riportate come liste di dizionari nei dataset:

	Name	Pokemon
0	Adaptability	[{"is_hidden": False, "pokemon": {"name": "eevee..."}]
1	Aerilate	[{"is_hidden": False, "pokemon": {"name": "pin..."}]
2	Aftermath	[{"is_hidden": True, "pokemon": {"name": "volt..."}]
3	Air Lock	[{"is_hidden": False, "pokemon": {"name": "ray..."}]
4	Analytic	[{"is_hidden": True, "pokemon": {"name": "magn..."}]

Figura 31. Relazione Abilities-Pokémon

	Name	Learned_by
0	Absorb	[{"name": "zubat", "url": "https://pokeapi.co..."}]
1	Accelerock	[{"name": "lycanroc-midday", "url": "https://p..."}]
2	Acid	[{"name": "ekans", "url": "https://pokeapi.co..."}]
3	Acid Armor	[{"name": "tentacool", "url": "https://pokeapi.co..."}]
4	Acid Spray	[{"name": "ekans", "url": "https://pokeapi.co..."}]

Figura 32. Relazione Moves-Pokémon

Per trasformare le precedenti tabelle in tabelle ponte è stato usato il metodo `pandas.DataFrame.explode()`, che genera una riga per ciascun elemento della lista. Successivamente, ciascuna chiave del dizionario è stata trasformata in una colonna della tabella, mantenendo poi solo quelle utili. Otteniamo, quindi, le seguenti tabelle ponte:

	Ability	Pokemon	Hidden
0	Adaptability	eevee	False
1	Adaptability	corphish	True
2	Adaptability	crawdaunt	True
3	Adaptability	feebas	True
4	Adaptability	porygon-z	False

Figura 33. Tabella ponte Abilities-Pokémon

	Move	Pokemon
0	Absorb	zubat
1	Absorb	golbat
2	Absorb	oddish
3	Absorb	gloom
4	Absorb	vileplume

Figura 34. Tabella ponte Moves-Pokémon

Per quanto riguarda la tabella ponte Abilities-Pokémon è stato mantenuto anche l’attributo *Hidden* che rappresenta una proprietà della relazione. In particolare, indica se un’abilità è nascosta o meno per lo specifico Pokémon.

Su queste tabelle ponte è stato poi necessario intervenire ulteriormente con degli aggiustamenti, come verrà illustrato in sezione 3.5.2.

Completata la pulizia dei dataset ottenuti tramite API, si è passati alla fase di integrazione. Come prima cosa sono stati integrati i dataset ottenuti da API con quelli ottenuti dallo scraping di *Pokémon Database* (sezione 2.2.2), che fanno riferimento a buona parte delle entità principali presenti nel database finale: Pokémon, Items, Abilities e Moves. Per effettuare l’integrazione è stata utilizzata la libreria **Recordlinkage** di Python.

3.2.3 Pokémon

Dall’analisi dello schema dei dataset **Species API** e **Complete Pokémon Pokédex** si è notato come fosse possibile stabilire un collegamento fra di essi sulla base dell’ID del Pokédex nazionale (univoco per ogni specie ma uguale tra le diverse varianti) e sul nome della specie del Pokémon. Si è ricercato un match esatto sull’ID e un match con soglia molto alta sul nome (è stata impostata una threshold = 0.85). Per valutare il matching sul nome è stata utilizzata la **Levenshtein distance normalizzata**[15], che valuta la distanza fra due stringhe in termini del numero minimo di inserimenti, cancellazioni o sostituzioni di caratteri per trasformare una stringa nell’altra. Così facendo, sono stati individuati 1054 match completi, sia ID che nome, e 23 match singoli, che combaciavano o per ID o per nome. Queste 23 coppie sono state valutate singolarmente: è stato osservato che quando l’ID corrisponde si è di fronte a un match effettivo, dove invece corrisponde il nome siamo di fronte a Pokémons differenti, semplicemente con un nome molto simile. A questo punto, i due dataset sono stati uniti mantenendo tutte le informazioni. Considerando che il dataset **Species API** fornisce informazioni generali su una specie di Pokémon, mentre **Complete Pokémon Pokédex** include anche le varianti di una stessa specie, nel dataset risultante le informazioni ottenute tramite API risulteranno duplicate tra le diverse varianti di uno stesso Pokémon.

#	Name	Generation	Rarity	Evolves_from	Has_gender_diff	Type1	Type2	Total	HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	Variant	
0	1	Bulbasaur	1	Common	None	False	Grass	Poison	318	45	49	49	65	65	45	NaN
1	2	Ivysaur	1	Common	Bulbasaur	False	Grass	Poison	405	60	62	63	80	80	60	NaN
2	3	Venusaur	1	Common	Ivysaur	True	Grass	Poison	525	80	82	83	100	100	80	NaN
3	3	Venusaur	1	Common	Ivysaur	True	Grass	Poison	625	80	100	123	122	120	80	Mega Venusaur

Figura 35. Dataset Pokémon integrato

3.2.4 Items

Nel caso dei dataset degli strumenti (**Items API** e **Items**) si è deciso di intervenire sull’attributo *Name* sostituendo tutti i simboli “-” con degli spazi e rendendo maiuscola la prima lettera di ogni parola, al fine di facilitare il matching. Non avendo un identificativo univoco per ciascuno strumento, il collegamento fra i vari oggetti è stato effettuato solo sulla base del nome, sfruttando sempre la *Levenshtein distance*. Sono stati individuati 826 strumenti con il nome esattamente uguale e 42 match potenziali di strumenti con nome molto simile (è stata impostata una threshold=0.9). Anche in questo caso i match potenziali sono stati trattati manualmente, osservando che alcuni di essi differivano semplicemente per la presenza di un punto. Tali record sono quindi stati sistemati e aggiunti.

	Name	Effect_API	Effect_SCRAPING	Attributes	Category_API	Category_SCRAPING	sprite_url
0	Ability Capsule	Switches a Pokémon between its two possible (n...)	A capsule that allows a Pokémon with two Abili...	[]	Vitamins	Hold items	https://img.pokemondb.net/sprites/items/abilities/capsule.png
1	Ability Patch	None	Changes a Pokémon’s ability to its Hidden Abil...	[]	Vitamins	General items	https://img.pokemondb.net/sprites/items/abilities/patch.png
2	Ability Urge	Forcibly activates a friendly Pokémon’s ability.	When used, it activates the Ability of an ally...	[]	Miracle Shooter	Battle items	https://img.pokemondb.net/sprites/items/abilities/urge.png
3	Abomasite	Held: Allows Abomasnow to Mega Evolve into Meg...	Enables Abomasnow to Mega Evolve during battle.	[]	Mega Stones	Hold items	https://img.pokemondb.net/sprites/items/abomasite.png
4	Absolite	Held: Allows Absol to Mega Evolve into Mega Ab...	Enables Absol to Mega Evolve during battle.	[]	Mega Stones	Hold items	https://img.pokemondb.net/sprites/items/absolite.png

Figura 36. Dataset Items parzialmente integrato

Nell’effettuare l’unione fra i due dataset, è stato necessario trattare la corrispondenza fra gli attributi *Category* ed *Effect*, presenti in entrambi. Per quanto riguarda le descrizioni degli effetti, avendo notato che essi spesso fornivano informazioni

complementari, è stato deciso di mantenerli entrambi, in modo da garantire una maggiore completezza e coprire l'eventuale presenza di valori mancanti in uno dei due attributi (come nel caso della seconda riga in figura 36). Rispetto alla categoria, invece, sono stati mantenuti entrambi gli attributi, perché è stato riconosciuto che la categoria dello scraping era più generale, mentre quella dell'API molto più specifica. Le colonne sono state, quindi, rinominate, e la tabella assume la seguente forma:

Name	Effect (alt.)	Effect	Usage Attributes	Specific Category	General Category	Image_url
Ability Capsule	Switches a Pokémon between its two possible (non-Hidden) forms.	A capsule that allows a Pokémon to change its form.	Vitamins	Hold items	https://img.pokeapi.co/media/items/ability-capsule.png	
Ability Patch		Changes a Pokéén's ability.	Vitamins	General items	https://img.pokeapi.co/media/items/ability-patch.png	
Ability Urge	Forcibly activates a friendly Pokémon's ability.	When used, it activates a friendly Pokémon's ability.		Miracle Shooter	Battle items	https://img.pokeapi.co/media/items/ability-urge.png
Abomasite	Held: Allows Abomasnow to Mega Evolve into Mega Abomasnow.	Enables Abomasnow to Mega Evolve into Mega Abomasnow.		Mega Stones	Hold items	https://img.pokeapi.co/media/items/abomasite.png
Absolite	Held: Allows Absol to Mega Evolve into Mega Absol.	Enables Absol to Mega Evolve into Mega Absol.		Mega Stones	Hold items	https://img.pokeapi.co/media/items/absolite.png
Absorb Bulb	Held: Raises the holder's Special Attack by one stage when it is hit.	A consumable bulb that increases the user's Special Attack.		Held Items	Hold items	https://img.pokeapi.co/media/items/absorb-bulb.png
Adamant Mint		Changes the Pokéén's Adamant Tint.		Nature Mints	Battle items	https://img.pokeapi.co/media/items/adamant-mint.png
Adamant Orb	Boosts the damage from Dialga's Dragon-type and Steel-type moves.	Increases the power of Dragon-type and Steel-type moves.	[holdable]	Species Specific	Hold items	https://img.pokeapi.co/media/items/adamant-orb.png
Adrenaline Orb	Makes wild Pokémon more likely to summon allies. Held: Using it makes wild Pokémon more likely to summon allies.	Using it makes wild Pokémon more likely to summon allies.		Held Items	Hold items	https://img.pokeapi.co/media/items/adrenaline-orb.png
Aerodactylite	Held: Allows Aerodactyl to Mega Evolve into Mega Aerodactyl.	Enables Aerodactyl to Mega Evolve into Mega Aerodactyl.		Mega Stones	Hold items	https://img.pokeapi.co/media/items/aerodactylite.png

Figura 37. Dataset Items integrato

Infine, è stato effettuato un controllo sulle righe escluse dal matching sia nel caso dell'API che dello scraping. Nel caso del dataset **Items API**, gli strumenti scartati fanno tutti parte delle categorie relative alla trama di gioco e senza alcuna rilevanza dal punto di vista competitivo. Nel caso di **Items**, gli strumenti scartati sono *cristalli Z*, strumenti non utilizzabili in competitivo secondo il regolamento attuale, e strumenti introdotti in *Legende Pokémon Arceus*, ultimo gioco pubblicato e ancora non inserito nel regolamento competitivo. Quindi, data la loro irrilevanza per il nostro obiettivo, tutti questi strumenti sono stati definitivamente eliminati.

3.2.5 Abilities e Moves

Per l'integrazione dei dataset relativi ad abilità (**Abilities API e Abilities**) e mosse (**Moves API e Moves**) è stato seguito lo stesso procedimento usato per **Items**. I nomi delle diverse abilità e mosse è stato trasformato sostituendo i simboli “-” con degli spazi e il matching è stato effettuato sul nome. Nel caso delle mosse, per gli attributi *Type*, *Power*, *Accuracy* e *PP* sono state mantenute le colonne provenienti dallo scraping in quanto più aggiornate.

In seguito al matching sono state valutate manualmente le righe scartate:

- **Abilities:** solo due abilità erano escluse dal matching. Tuttavia, si è verificato che tali abilità erano effettivamente corrispondenti, e sono quindi state sistematicamente eliminate:

Name	Generation	Description		
0	Adaptability	4	Powers up moves of the same type.	
1	Aerilate	6	Turns Normal-type moves into Flying-type moves.	
2	Aftermath	4	Damages the attacker landing the finishing hit.	
3	Air Lock	3	Eliminates the effects of weather.	
4	Analytic	5	Boosts move power when the Pokémon moves last.	

Figura 38. Abilities Dataset

- **Abilities:** fra le mosse scartate dai due dataset sono state individuate alcune corrispondenze, che differivano per l'assenza di alcuni simboli all'interno del nome. Tali mosse sono state aggiunte manualmente:

Name	Introduced_in	Type	Power	Acc.	PP	Damage_class	Effect	Prob. (%)
0	Absorb	1 Grass	20	100	25	Special	User recovers half the HP inflicted on opponent.	<NA>
1	Accelerock	7 Rock	40	100	20	Physical	User attacks first.	<NA>
2	Acid	1 Poison	40	100	30	Special	May lower opponent's Special Defense.	10
3	Acid Armor	1 Poison	<NA>	<NA>	20	Status	Sharply raises user's Defense.	<NA>
4	Acid Spray	5 Poison	40	100	20	Special	Sharply lowers opponent's Special Defense.	100

Figura 39. Moves Dataset

- Inoltre, dal dataset dello scraping sono state estratte le cosiddette *G-Max Moves*, che sono state trattate a parte e salvate nel nuovo dataset **G-Max Moves PokémonDatabase**:

Name	Type	Power	Acc.	PP	Effect	Prob. (%)	Cat.
G Max Befuddle	Bug	—	∞	5	Butterfree-exclusive G-Max Mo	100	—
G Max Cannonad	Water	—	∞	10	Blastoise-exclusive G-Max Mov	—	—
G Max Centiferno	Fire	—	∞	5	Centiskorch-exclusive G-Max N	100	—
G Max Chi Strike	Fighting	—	∞	5	Machamp-exclusive G-Max Mo	—	—
G Max Cuddle	Normal	—	∞	5	Eevee-exclusive G-Max Move. I	100	—

Figura 40. G-Max Moves PokémonDatabase Dataset

Al termine di questa fase, dall'integrazione dei dati di API e dello scraping di *Pokémon Database*, sono quindi stati ottenuti i dataset: **Pokémon**, **Abilities**, **Items**, **Moves** e **G-Max Moves PokémonDatabase**.

3.3 G-Max Moves e Gigantamax Pokémon

In questa fase è stato effettuato un processo di cleansing sui dataset **G-Max Moves** e **Gigantamax Pokémon** acquisiti da *Bulbapedia* e introdotti in sezione 2.2.6, con successiva integrazione ai dati di **G-Max Moves PokémonDatabase**, introdotto nella sezione precedente, e concatenazione ai dataset principali **Pokémon** e **Moves**. Viene inoltre ottenuta la tabella `ponte_moves_pokemon_GMAX_MOVE`, che permette di legare ogni forma *Gigantamax* alla sua peculiare mossa.

3.3.1 Cleansing e integrazione di Gigantamax Pokémon e unione al dataset Pokémon

- Correzione manuale di alcuni nomi dei Pokémon per rendere più semplice il successivo matching.
- Estrazione delle colonne [*Gigantamax Pokémon*, *image_url*] da **G-Max Moves**. La colonna *image_url*, nonostante sia parte del dataset relativo alle mosse G-Max, contiene le immagini dei Pokémon *Gigantamax*. Questa colonna viene quindi integrata al dataset **Gigantamax Pokémon** mediante *inner join*. Seguono alcuni processi atti ad armonizzare il dataset prima di concatenarlo a **Pokémon**.
- Generazione della colonna *Variant*, contenente per tutti i record la stringa *Gigantamax*.
- Split della colonna *Type* in [*Type1*, *Type2*].
- Modifica dei nomi degli attributi.
- Creazione della colonna *Generation*, contenente per tutti i record l'intero 8 (la variante *Gigantamax* è stata introdotta nell'ottava generazione).
- Aggiunta della colonna # (ID della specie nel Pokédex nazionale) mediante *inner join* col dataset **Pokémon**, considerando un match esatto tra i nomi. Le varianti *Gigantamax* acquisiscono quindi l'ID assegnato in **Pokédex** a tutte le varianti della loro stessa specie.
- Concatenazione di **Gigantamax Pokémon** a **Pokémon**. Si ha quindi l'aggiunta dei Pokémon varianti *Gigantamax* al dataset completo dei Pokémon, in cui precedentemente non erano presenti. *Sorting* di **Pokémon** in base alla colonna #.

3.3.2 Integrazione di G-Max Moves e G-Max Moves PokémonDatabase, unione al dataset Moves e creazione della tabella ponte Moves-Pokémon "GMAX_MOVE"

- Integrazione al dataset **G-Max Moves** dei dati di **G-Max Moves PokémonDatabase**: non essendo possibile una *join* con match esatto tra i nomi delle mosse nei due dataset (rispettivamente contenuti nelle variabili *Max Move* e *Name*), l'unione è stata effettuata mediante un processo di *fuzzy matching* usando la libreria **Python Record Linkage Toolkit**, utilizzando l'algoritmo di *Levenshtein* per il calcolo della similarità tra le stringhe e una threshold pari a 0.9.
- Creazione della tabella ponte `ponte_moves_pokemon_GMAX_MOVE`: Questa fase vede la creazione di una tabella ponte tra il dataset **Pokémon** e **Moves**, che permette di collegare i record relativi ai Pokémon di variante *Gigantamax* con le peculiari mosse che li contraddistinguono, relazione poi opportunamente implementata in fase di Data Modeling. La tabella è stata generata semplicemente estraendo dal dataset **G-Max Moves** le colonne [*Max Move*, *Gigantamax Pokémon*].

	Max Move	Gigantamax Pokémon
0	G-Max Vine Lash	Venusaur Gigantamax
1	G-Max Wildfire	Charizard Gigantamax
2	G-Max Cannonade	Blastoise Gigantamax
3	G-Max Befuddle	Butterfree Gigantamax

Figura 41. Tabella ponte_moves_pokemon_GMAX_MOVE

3. Preparazione del dataset ottenuto per l'unione a **Moves**: eliminazione delle variabili [*image_url*, *Gigantamax Pokémon*]; modifica dei nomi delle colonne per armonizzarle a **Moves** e introduzione della colonna *Generation*, contenente per tutti i record l'intero 8.
4. Concatenazione di **G-Max Moves** a **G-Max Moves PokémonDatabase**, permettendo l'introduzione delle mosse speciali G-Max nel dataset completo delle mosse.

3.3.3 Cleaning e manipolazione del dataset Pokémon arricchito e creazione di una chiave primaria

In seguito all'introduzione dei Pokémon *Gigantamax* sono emerse alcune problematiche nel dataset principale **Pokémon**, che hanno richiesto una ulteriore fase di cleaning. Inoltre il dataset è stato manipolato al fine di ottenere una variabile che permetta di identificare univocamente ogni record:

1. Creazione dell'attributo *Name*, ottenuto, per ogni Pokémon, come somma della vecchia variabile *Name* (contenente il nome della specie del Pokémon) e di *Variant* (la variante specifica di una certa specie di Pokémon). La variabile che precedentemente era chiamata *Name* prende ora il nome di *Species*.
2. Cleaning: correzione errori nella variabile *Variant* (alcuni record presentavano erroneamente il nome della specie insieme a quello della variante nel campo *Variant*); correzione errori nella variabile *Generation* (generazione di introduzione errata per alcune varianti); correzione errori variabile *Evolves from*; eliminazione variabile *Varieties*, considerata irrilevante a fini competitivi e cambio nome della colonna *sprite_url* in *image_url* (al fine di permettere l'importazione delle immagini in Neo4J).

3.4 I Types: dataset e tabelle ponte relative

I tipi sono proprietà elementali che caratterizzano Pokémon e mosse. Nel corso di questa fase del processo è stato creato il dataset **Types**, contenente tutti i 20 tipi, di cui solo 18 utilizzabili in battaglia, introdotti finora nel mondo Pokémon. Sono inoltre state ottenute le tabelle ponte che permettono di definire le relazioni che legano un Pokémon con i suoi tipi, una mossa con il suo tipo e, in battaglia, il tipo della mossa attaccante con il tipo del Pokémon che la subisce, in termini di efficiacia.

3.4.1 Creazione dataset Types a partire da Types API

Il dataset è stato ottenuto semplicemente estraendo da **Types API** la colonna *Name* dei nomi dei tipi e generando manualmente la colonna *Generation*, contenente la generazione di introduzione per ognuno di essi.

	Name	Generation
0	normal	1
1	fighting	1
2	flying	1
3	poison	1
4	ground	1
5	rock	1
6	bug	1
7	- - -	1

Figura 42. Dataset Types

3.4.2 Creazione tabella ponte Type-Type "MOVE_EFFECTIVENESS_ON_POKEMON" da "Type Chart"

Type Chart permette di valutare l'efficacia di una mossa di un tipo x quando utilizzata per colpire un Pokémon di tipo y. Il dataset, introdotto nella sezione 2.2.2, è formato da 18 righe (una per ogni tipo, relativo alla mossa attaccante) e 18 colonne (una per ogni tipo, relativo al Pokémon che si difende). Il valore delle celle è il moltiplicatore che viene applicato alla mossa del

Pokémon attaccante. Obiettivo di questa parte del processo è stato trasformare il dataset in una tabella ponte, che permettesse di collegare ogni record (ogni tipo) del dataset **Types** con tutti gli altri record (incluso se stesso), permettendo di definire la relazione chiamata **MOVE_EFFECTIVENESS_ON_POKEMON**. Le fasi del processo di manipolazione e arricchimento sono state:

1. Utilizzo della funzione di Pandas `pandas.melt()`, passando come argomento `Atk.Type/Pokemon.Type`(indice esplicito, indica il tipo del Pokémon attaccante), per trasformare il dataset **Type Chart** (18x18) nella tabella **ponte_type_type_MOVE_EFFECTIVENESS_ON_POKEMON**. La tabella ottenuta presenta 324 righe (una per ogni possibile coppia tipo-tipo) e tre colonne [*Atk. Move Type, Def. Pokemon Type, Damage Multiplier*].
2. Creazione della colonna *Effectiveness*, contenenti valori di tipo stringa assegnati condizionatamente al valore di *Damage Multiplier* (valori che esplicitano a parole il rapporto descritto da *Damage Multiplier*).

	Atk. Move Type	Def. Pokemon Type	Damage Multiplier	Effectiveness
0	Normal	Normal	<NA>	Normal (100%)
1	Fire	Normal	<NA>	Normal (100%)
2	Water	Normal	<NA>	Normal (100%)
3	Electric	Normal	<NA>	Normal (100%)
4	Grass	Normal	<NA>	Normal (100%)

Figura 43. Tabella `ponte_type_type_MOVE_EFFECTIVENESS_ON_POKEMON`

3.4.3 Creazione tabelle ponte Pokémon-Type "IS_OF_TYPE" e Moves-Type "MOVES_IS_TYPE"

Viene creata la tabella ponte utile a collegare ogni record del dataset **Pokémon** a uno o due record di **Types** (in funzione del fatto che un Pokémon abbia tipo singolo o doppio), permettendo di definire la relazione **IS_OF_TYPE** che lega Pokémon e tipi. La generazione della tabella ha richiesto:

1. Estrazione delle colonne [*Name, Type1, Type2*] da **Pokémon**.
2. La tabella estratta presenta colonne ripetute (*Type1, Type2*), la trasformazione nella corrispettiva tabella in *prima forma normale* permette ottenere la tabella ponte richiesta.

	Name	Type
0	Bulbasaur	Grass
0	Bulbasaur	Poison
1	Ivysaur	Grass
1	Ivysaur	Poison
2	Venusaur	Grass

Figura 44. Tabella `ponte_pokemon_type_IS_OF_TYPE`

Analogamente, viene creata la tabella ponte utile a collegare ogni record del dataset **Moves** a un record di **Types**, definendo la relazione **MOVES_IS_TYPE** che lega le mosse al proprio tipo, estraendo dal dataset **Moves** le colonne [*Name, Type*].

	Name	Type
0	Absorb	Grass
1	Accelerock	Rock
2	Acid	Poison
3	Acid Armor	Poison
4	Acid Spray	Poison

Figura 45. Tabella `ponte_move_type_MOVES_IS_TYPE`

3.5 Altre relazioni che interessano i Pokémon: tabelle ponte

Sono state ottenute una serie di tabelle ponte ulteriori, che permettano di collegare ogni record di **Pokémon** rispettivamente: alla sua evoluzione precedente (tabella **ponte_pokemon_pokemon_EVOLVES_FROM**); ogni Pokémon non-variante a tutte le sue varianti (tabella **ponte_pokemon_pokemon_HAS_VARIANT**); alle abilità che possiede (tabella **ponte_pokemon_abilities_MAY_HAS**); alle mosse che è in grado di apprendere (tabella **ponte_pokemon_moves_MAY_LEARN**).

3.5.1 Creazione tabelle ponte Pokémon-Pokémon "EVOLVES_FROM" e "HAS_VARIANT"

Viene creata la tabella ponte utile a collegare ogni record del dataset **Pokémon** al record del dataset stesso relativo alla sua forma precedente nella catena evolutiva. La generazione della tabella ha richiesto semplicemente l'estrazione degli attributi *[Name, Evolves_from]* dal dataset **Pokémon**, non considerando i record per i quali l'attributo *Evolves_from* risulta nullo (forme base non evolute).

Name	Evolves_from
Ivysaur	Bulbasaur
Venusaur	Ivysaur
Mega Venusaur	Venusaur
Venusaur Gigant	Venusaur
Charmeleon	Charmander
Mega Charizard Y	Charizard
Charizard Gigant	Charizard
Mega Charizard X	Charizard

Figura 46. Tabella **ponte_pokemon_pokemon_EVOLVES_FROM**

La tabella **ponte_pokemon_pokemon_HAS_VARIANT** viene invece ottenuta mediante:

1. Estrazione delle colonne *[Name, Species]* da **Pokémon**, considerando solo i record relativi ai Pokémon non-varianti, per i quali l'attributo *Variant* risulta nullo.
2. Estrazione delle colonne *[Name, Species]* da **Pokémon**, considerando solo i record relativi ai Pokémon varianti, per i quali l'attributo *Variant* risulti non-nullo.
3. Combinazione mediante *inner join*, sull'attributo *Species*, dei due dataset ottenuti nei punti precedenti.
4. Eliminazione della colonna *Species*.

	Non-Variant Pkm Name	Variant Pkm Name
0	Venusaur	Mega Venusaur
1	Venusaur	Venusaur Gigantamax
2	Charizard	Mega Charizard Y
3	Charizard	Charizard Gigantamax
4	Charizard	Mega Charizard X

Figura 47. Tabella **ponte_pokemon_pokemon_HAS_VARIANT**

3.5.2 Creazione tabelle ponte Pokémon-Abilities "MAY_HAS" e Pokémon-Moves "MAY_LEARN"

Le tabelle ponte corrispondenti (**ponte Abilities-Pokémon** e **ponte Moves-Pokémon**) a quelle desiderate sono state ottenute precedentemente, a partire dai dati acquisiti mediate API, e introdotte nella sezione 3.2.2. Affinché le due tabelle possano essere utilizzate effettivamente come tabelle ponte è stato però necessario risolvere il mancato match dei nomi dei Pokémon nei due dataset (in entrambi i casi presenti nella variabile *Pokemon*) con i nomi degli stessi nel dataset **Pokémon**. Le tabelle sono quindi state processate come segue (le operazioni sono valide per entrambi i dataset):

1. Applicazione della funzione `recordlinkage.preprocessing.clean()` alla variabile *Pokémon* delle tabelle ponte, al fine di eliminare i '-' che separano le parole in alcuni nomi complessi e applicazione di `.str.title()` al

fine di rendere maiuscola la prima lettera di ogni parola (per armonizzare il formato a quello usato nel dataset **Pokémon** e semplificare il matching).

Segue il processo di *fuzzy matching*, che ha permesso di "tradurre" i nomi dei Pokémons nelle due tabelle ponte, armonizzandoli con i nomi utilizzati nel dataset **Pokémon**.

2. Al fine di non rendere il *fuzzy matching* computazionalmente troppo oneroso, si è optato per considerare i soli nomi unici dei Pokémons presenti nelle due tabelle ponte, per eseguire il processo di "traduzione" (riducendo drasticamente il numero di calcoli di similarità da eseguire, essendo molti dei Pokémons presenti più volte nelle tabelle ponte).
3. *Join*, effettuato mediante un processo di *fuzzy matching*, usando la libreria *Python Record Linkage Toolkit*, tra la lista dei nomi unici delle tabella ponte e la lista dei nomi dei Pokémons estratti dal dataset **Pokémon**, utilizzando come algoritmo **q-gram**^[16] per il calcolo della similarità tra le stringhe e un threshold uguale a 0.85. La scelta è ricaduta su *q-gram* perchè risultato più performante nel caso di stringhe che presentano parole invertite. Si ottiene la tabella di "traduzione" che assegna ad ogni nome unico delle tabelle ponte il corrispettivo nome del Pokémon nel dataset **Pokémon**.
4. *inner join* tra la tabella ponte e la tabella di "traduzione" e conseguente eliminazione della variabile contenente i nomi non armonizzati.

Ability	Hidden	Pokemon
Adaptability	False	Eevee
Anticipation	True	Eevee
Run Away	False	Eevee
Adaptability	True	Corphish
Hyper Cutter	False	Corphish
Shell Armor	False	Corphish

Figura 48. Tabella
ponte_pokemon_ability_MAY_HAS

Move	Pokemon
Absorb	Zubat
Acrobatics	Zubat
Aerial Ace	Zubat
Agility	Zubat
Air Cutter	Zubat
Air Slash	Zubat

Figura 49. Tabella
ponte_pokemon_moves_MAY_LEARN

3.6 Integrazione regolamento VGC 2022 Series 12 a Pokémons

Questa fase del processo di integrazione ha visto la generazione dell'attributo *VGC2022.rules*, nel dataset **Pokémon**, che permette di associare ad ogni Pokémon il suo status competitivo in accordo con il regolamento *VGC 2022 Series 12*. Nella sezione 2.2.4 è stato descritto il processo di acquisizione delle liste contenenti i riferimenti ai Pokémons dallo status *Permitted*, *Restricted* e *Banned*, e la lista dei Pokémons con trasformazione *Gigantamax* di cui è permesso l'utilizzo in battaglia (*Gigantamax Allowed*). Il primo ostacolo incontrato in questa fase è stato associato all'eterogeneità con cui, all'interno delle liste del regolamento, si fa riferimento ai Pokémons, impedendo di eseguire il data matching in modo immediato, infatti:

- Le liste **Banned Pokémons** e **Allowed Gigantamax Pokémons** fanno riferimento ai Pokémons mediate il nome della specie dello stesso, permettendo il matching con i record di **Pokémon** sull'attributo *Species*.
- La lista **Restricted Pokémons** fa riferimento ai Pokémons mediate l'ID nel Pokédex nazionale, permettendo il matching con i record di **Pokémon** sull'attributo *#*.
- Le 4 liste dei **Permitted Pokémons** fanno riferimento ai Pokémons con modalità differenti: **National Permitted** usa come riferimento l'ID nel Pokédex nazionale (permettendo il matching sull'attributo *#* di **Pokémon**), mentre **Galar Permitted**, **Isle of Armor Permitted** e **The Crown Tundra Permitted** usano l'ID del Pokémon nel corrispettivo Pokédex regionale (ottenuti in fase di acquisizione e introdotti nella sezione 2.2.5).

3.6.1 Cleaning e manipolazione delle liste del regolamento Series 12 VGC 2022

A causa della problematica spiegata, prima di proseguire con il processo di data matching e integrazione del regolamento in **Pokémon**, è stato necessario uniformare il formato di rappresentazione dei Pokémons nelle diverse liste, utilizzando come identificativo il nome del Pokémon, al fine di eseguire poi il matching sull'attributo *Name*. Il processo applicato è stato personalizzato in funzione dei riferimenti utilizzati nella singola lista:

- Le 4 liste relative ai Pokémons *Permitted* hanno subito la conversione del formato e successivamente sono state unite nell'unica lista **Permitted Pokémons**:

- Cleaning e manipolazione, con modalità simili, dei 3 dataset dei Pokédex regionali (dataset **Galar Pokédex, Isle of Armor Pokédex e The Crown Tundra Pokédex**): conversione attributo *No.* (ID del Pokémon) da stringa a intero, rimuovendo i caratteri non numerici; rimozione caratteri giapponesi dall'attributo *Name* e inserimento della parola *Galarian* prima dei nomi dei Pokémon che costituiscono varianti della regione di *Galar* di Pokémon già presenti in **Pokémon**, al fine di distinguerli dalle versioni non varianti.

No.	Name	No.	Name
#001	Slowpoke サ ドン	1	Galarian Slowpoke
#002	Slowbro サ ド ラン	2	Galarian Slowbro
#003	Slowking サ ド キン グ	3	Slowking
#004	Buneary シ ミ ロル	4	Buneary
#005	Lopunny シ ミ ロ ッ プ	5	Lopunny

Figura 50. Esempio processo di cleansing sul dataset **Isle of Armor Pokédex**

- Ottenimento delle 4 liste dei nomi dei Pokémon *Permitted*: estrazione, per ognuno dei 4 Pokédex, dei record ,e in particolare dell'attributo *Name*), per i quali il valore dell'ID (# nel Pokédex nazionale, *Pokédex* e *No.* in quelli regionali) risultasse presente nelle corrispettive liste del regolamento.
- Concatenazione delle 4 liste a formare la lista **Permitted Pokémon**.

- Analogamente, **Restricted Pokémon** è stato convertito da lista di ID a lista di nomi dei Pokémon. Essendo la lista unica e usando come riferimento gli ID del Pokédex Nazionale, il processo di conversione del formato è risultato più semplice.
- La lista dei Pokémon *banned* non ha richiesto la conversione di formato, essendo già una lista di nomi delle specie di Pokémon proibiti. Alla lista sono stati aggiunti i nomi dei Pokémon del dataset **Pokémon** con valore di *Variant* uguale a *Mega* (il regolamento *VGC 2022 Series 12* non ammette le varianti *Mega*).
- Analogamente, **Allowed Gigantamax Pokémon** non ha richiesto la conversione del formato di riferimento utilizzato. Però, al fine di rendere efficace il successivo matching con i record di **Pokémon**, i nomi sono stati adattati a quelli presenti in quest'ultimo.

Si ottengono quindi le liste finali: **permitted_VGC2022**, **restricted_VGC2022**, **banned_VGC2022** e **gigantamax_allowed_VGC2022**:

Name	Value
gigantamax_allowed_list	['Venusaur', 'Charizard', 'Blastoise', 'Butterfree', 'Pikachu', 'Meowth', 'Machamp', 'Gengar', 'K
banned	['Mew', 'Celebi', 'Jirachi', 'Victini', 'Keldeo (both forms)', 'Genesect (all forms)', 'Diancie', '<...>
crown_tundra_permitted	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2 <...> 0,
galar_permitted	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2 <...> 8,
isle_of_armor_permitted	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 2 <...> 1,
national_permitted	[243, 244, 245, 252, 253, 254, 255, 256, 257, 258, 259, 260, 380, 381, 480, 481, 482, 485, 488,
national_restricted	[150, 249, 250, 382, 383, 384, 483, 484, 487, 643, 644, 646, 716, 717, 718, 789, 790, 791, 792,

Name	Value
banned_VGC2022	['Mega Venusaur', 'Mega Charizard Y', 'Mega Chari
gigantamax_allowed_VGC2022	['Venusaur Gigantamax', 'Charizard Gigantamax', 'E
permitted_VGC2022	['Raikou', 'Entei', 'Suicune', 'Treecko', 'Grovyle', 'Sce
restricted_VGC2022	['Mewtwo', 'Lugia', 'Ho-oh', 'Kyogre', 'Groudon', 'R

Figura 51. Liste regolamento *VGC 2022 Series 12*: prima e dopo la manipolazione

3.6.2 Creazione della colonna VGC2022.rules, nel dataset Pokémon

Una volta ottenute le liste relative al regolamento del *VGC 2022 Series 12*, si è proseguito integrando le informazioni al dataset **Pokémon**. Viene generato il nuovo attributo *VGC2022_rules*, relativo allo status competitivo dei Pokémon. I valori possibili per il nuovo attributo sono [*Permitted*, *Restricted*, *Banned*, *Gigantamax Allowed*] e sono stati assegnati ad ogni record in funzione della presenza del valore dell’attributo *Name* in una delle 4 liste ottenute nella sezione precedente. Nel caso in cui il nome di un Pokémon non fosse presente in nessuna delle liste è stato assegnato il valore nullo pd .NA.

#	Name	VGC2022_rules
1	Bulbasaur	Permitted
2	Ivysaur	Permitted
3	Venusaur	Permitted
3	Mega Venusaur	Banned
3	Venusaur Gigantamax	Gigantamax Allowed
4	Charmander	Permitted
5	Charmeleon	Permitted
6	Mega Charizard Y	Banned
6	Charizard Gigantamax	Gigantamax Allowed
6	Mega Charizard X	Banned
6	Charizard	Permitted
7	Squirtle	Permitted

Figura 52. Primi record relativi agli attributi [#, Name, VGC2022_rules] di **Pokémon**

3.7 Lotte competitive su *Showdown!*: integrazione e tabelle ponte

L’ultima fase del processo di cleaning e integrazione ha visto il trattamento dei dati introdotti nella sezione 2.2.3, relativi ai match competitivi svolti a Marzo 2022, seguendo il regolamento *VGC Series 12* e disputati sulla piattaforma *Pokémon Showdown!*. I dataset e le tabelle ponte ottenute in fase di acquisizione e relative al competitivo, a cui si farà riferimento in questa sezione, sono: **Pokémon Usage Dataset**, **Teammates Usage**, **Items Usage**, **Moves Usage**, **Abilities Usage** e **Natures Usage**. Il processo si è svolto in fasi diverse con specifici obiettivi.

3.7.1 Fuzzy matching

Parte dei nomi utilizzati per indicare Pokémon e mosse nei 6 dataset relativi alle lotte competitive non danno match esatto con i nomi utilizzati per indicare gli stessi all’interno dei dataset integrati **Pokémon** e **Moves** ottenuti in precedenza. Quindi, applicando un processo di *fuzzy matching* in modo del tutto analogo a quanto mostrato in sezione 3.5.2, sono state ottenute delle tabelle di “traduzione”, che permettessero di associare ad ogni nome utilizzato nei dataset delle lotte competitive, un nome nei vari dataset integrati. Lo scopo di tali tabelle ponte di “traduzione” è duplice: il primo è quello di permettere di integrare i dati del dataset **Pokémon Usage** **Pokémon** tramite *join* e il secondo è quello di sostituire i nomi utilizzati nelle tabelle **Pokémon Usage Dataset**, **Teammates Usage** e **Moves Usage** con nomi che diano sempre match con quelli dei due dataset integrati menzionati poc’anzi.

Come fatto in precedenza, si è scelto di utilizzare *q-gram* come algoritmo per il calcolo della distanza tra le stringhe, risultando esso più performante della *Levenshtein distance*, in termine di numero di *match* corretti restituiti a parità di threshold, quando applicato al confronto tra stringhe che presentano alcune parole simili ma invertite. In particolare:

- Tabella “traduzione” **Nomi Pokémon**: la tabella che permette di associare i nomi dei Pokémon nei dataset del competitivo con il corrispettivo nome in **Pokémon** è stata ottenuta applicando un processo di *fuzzy matching* tra la lista dei nomi di Pokémon unici estratti dalla variabile *Pokemon* di **Pokémon Usage** e i valori della variabile *Name* di **Pokémon**.
 1. Operazioni preliminari di cleaning per facilitare il matching: rimozione del ‘-’ utilizzato per separare le parole nella variabile *Pokemon* di **Pokémon Usage** e applicazione di *str.title()*.
 2. Un primo matching parziale tra i nomi è stato valutato applicando una threshold di 0.75. Su un totale di 342 nomi di Pokémon unici, 323 hanno restituito un match potenziale, numero che è sceso a 316 dopo l’eliminazione di corrispondenze non corrette.
 3. Si applica nuovamente il matching parziale sui 26 nomi che non hanno restituito nessuna corrispondenza, questa volta con un threshold di 0.5. Si ottengono 28 potenziali match, numero che scende a 15 dopo l’eliminazione delle corrispondenze errate.
 4. Per gli ultimi 11 nomi è stata manualmente inserita una corrispondenza nella tabella di “traduzione”.
- Tabella “traduzione” **Nomi Mosse**: dall’attributo *Move* di **Moves Usage** sono estratti 421 nomi unici di mosse e 16 di questi non presentano una corrispondenza nell’attributo *Name* di **Moves**. In seguito a un processo di cleaning analogo a

quello applicato ai nomi dei Pokémons, su queste 16 stringhe si applica il matching parziale con threshold pari a 0.80 ottenendo 17 match parziali, di cui 3 risultano corrispondenze errate. Sono stati analizzati gli ultimi 2 *missing match*: essi corrispondono ai valori *Other* e *Nan*, per i quali è corretto che non siano trovate corrispondenze.

3.7.2 Integrazione statistiche di utilizzo/popolarità dei Pokémons al dataset Pokémons

Durante questo step del processo sono stati integrati al dataset **Pokémon** i dati di utilizzo/popolarità dei Pokémons nei match competitivi contenuti in **Pokémon Usage**, portando all'introduzione degli attributi *Monthly Usage (k)*, *Usage Percent (%)* e *Monthly Rank*. È stata effettuata una prima *left outer join*, su *Name* in **Pokémon** e su *Pokemon* di **Pokémon Usage**, utilizzando nel mentre la tabella di "traduzione" **Nomi Pokémons**, ottenuta nella sezione precedente, per risolvere il mancato match esatto tra i nomi. La scelta di utilizzare una *left outer join* è dovuta alla necessità di mantenere quei record di **Pokémon** relativi a Pokémons che non compaiono nelle statistiche di utilizzo competitivo (probabilmente perché mai utilizzati nei match). Si ottiene quindi la forma finale, completamente integrata, del dataset **Pokémon**:

#	Name	Species	Variant	Generation	Rarity	Evolves_from	Has_gender_diff	Type1	Type2	Total
1	Bulbasaur	Bulbasaur			1 Common		False	Grass	Poison	318
2	Ivysaur	Ivysaur			1 Common	Bulbasaur	False	Grass	Poison	405
3	Venusaur	Venusaur			1 Common	Ivysaur	True	Grass	Poison	525
3	Mega Venusaur	Venusaur	Mega		6 Common	Venusaur	True	Grass	Poison	625
3	Venusaur Gigantamax	Venusaur	Gigantamax		8	Venusaur		Grass	Poison	
4	Charmander	Charmander			1 Common		False	Fire		309
5	Charmeleon	Charmeleon			1 Common	Charmander	False	Fire		405
6	Mega Charizard Y	Charizard	Mega Y		6 Common	Charizard	False	Fire	Flying	634
6	Charizard Gigantamax	Charizard	Gigantamax		8	Charizard		Fire	Flying	
6	Mega Charizard X	Charizard	Mega X		6 Common	Charizard	False	Fire	Dragon	634
6	Charizard	Charizard			1 Common	Charmeleon	False	Fire	Flying	534
HP	Attack	Defense	Sp. Atk	Sp. Def	Speed	image_url	VGC2022_rules	Monthly Usage (k)	Usage Percent (%)	Monthly Rank
45	49	49	65	65	45	https://img.poker	Permitted			
60	62	63	80	80	60	https://img.poker	Permitted			
80	82	83	100	100	80	https://img.poker	Permitted	204	7	24
80	100	123	122	120	80	https://img.poker	Banned			
						https://archives.b	Gigantamax Allowed			
39	52	43	60	50	65	https://img.poker	Permitted			
58	64	58	80	65	80	https://img.poker	Permitted			
78	104	78	159	115	100	https://img.poker	Banned			
						https://archives.b	Gigantamax Allowed			
78	130	111	130	85	100	https://img.poker	Banned			
78	84	78	109	85	100	https://img.poker	Permitted	344	19	12

Figura 53. Dataset **Pokémon** completamente integrato

3.7.3 Correzione nomi di Pokémons e mosse nelle tabelle ponte **Pokémon Usage Dataset**, **Teammates Usage** e **Moves Usage**

L'ultimo step del processo di integrazione e cleaning ha visto la correzione dei nomi di Pokémons e mosse presenti in **Pokémon Usage Dataset**, **Teammates Usage** e **Moves Usage**, utilizzando le tabelle di "traduzione" precedentemente ottenute. Il fine è quello di ottenere delle tabelle ponte pronte per l'importazione in Neo4J, per le quali si abbia sempre match esatto tra i nomi dei Pokémons e delle mosse contenuti nelle 3 tabelle e i record del dataset **Pokémon** e **Moves**. Il processo è stato lo stesso per le tre tabelle ponte e ha previsto, per ogni colonna contenente i nomi da correggere, di effettuare un *inner join* tra la tabella ponte e la tabella di "traduzione" corrispondente, con conseguente eliminazione della vecchia colonna contenente i nomi non armonizzati.

4. Data Quality

Le dimensioni di qualità che sono state tenute in considerazione all'interno di questo lavoro sono state **currency**, **completeness** e **consistency**. Durante il progetto sono state adottate delle precise scelte volte a valutare la qualità dei dati rispetto a questi 3 aspetti.

4.1 Currency

Come già emerso in sezione 3, alcuni dei dataset ottenuti tramite API e scraping presentavano le medesime informazioni, come nel caso delle statistiche delle mosse in **Moves API** e **Moves**. Per cercare di garantire la currency più elevata possibile, è stato deciso di mantenere le informazioni dello scraping, dal momento che i siti utilizzati vengono aggiornati e corretti quotidianamente, mentre l'API, specialmente rispetto a tutte le entità introdotte nell'ultima generazione, risulta poco affidabile.

Tuttavia, va sottolineato che la currency dei nostri dati, specialmente per le informazioni legate al gioco competitivo estratte da *Pikalytics*, diminuisce mensilmente, cioè ogni volta che tali informazioni vengono aggiornate attraverso i dati delle competizioni del mese precedente.

Inoltre, la currency diminuisce anche all'uscita di nuovi giochi, dove si introducono nuovi Pokémon, mosse, strumenti e abilità e dove spesso si interviene anche modificando alcune caratteristiche di quelli già esistenti.

4.2 Completeness

La completezza è stata valutata a livello di singolo attributo per ciascun dataset.

Per il dataset **Pokémon**, su un totale di 1098 righe:

Attributo	Totale di NaN	% di NaN
#	0	0
Name	0	0
Species	0	0
Variant	861	78,42
Generation	0	0
Rarity	32	2,91
Evolves_from	543	49,45
Has_gender_diff	32	2,91
Type1	0	0
Type2	510	46,45
Total	32	2,91
HP	32	2,91
Attack	32	2,91
Defense	32	2,91
Sp. Atk	32	2,91
Sp. Def	32	2,91
Speed	32	2,91
image_url	0	0
VGC2022_rules	381	34,70
Monthly Usage (k)	756	68,85
Usage Percent (%)	756	68,85
Monthly Rank	756	68,85

Tabella 1. Completezza Pokémon

- gli attributi relativi alle statistiche, *Rarity* e *Has_gender_diff* presentano dei valori mancanti in corrispondenza delle varianti *Gigantamax*, di cui mancano tali informazioni;
- i valori mancanti per *VGC2022_rules* sono determinati dal fatto che alcuni Pokémon non sono ottenibili in alcun modo negli attuali giochi;
- l'assenza di informazioni per *Monthly Usage (k)*, *Usage Percent (%)* e *Monthly Rank*, ricavate dalle statistiche dei match competitivi, è giustificata dal fatto che molti Pokémon non sono stati utilizzati nel periodo di Marzo 2022;
- i molti valori nulli per gli attributi *Variant*, *Evolves_from* e *Type2* sono attribuibili al fatto che, rispettivamente, la maggior parte dei Pokémon del Pokédex sono forme base di una specie, alcuni Pokémon non evolvono da nessun'altra specie e solo alcuni Pokémon sono caratterizzati da un secondo tipo.

Per il dataset **Items**, su un totale di 840 righe:

Attributo	Totale di NaN	% di NaN
Name	0	0
Effect (alt.)	224	26,67
Effect	16	1,9
Usage Attributes	599	71,31
Specific Category	0	0
General Category	0	0
image_url	0	0

Tabella 2. Completezza Items

Si può notare che per 16 strumenti manca il dato dell'effetto, mentre per circa il 27% delle righe manca la descrizione alternativa. L'assenza di tali informazioni è stata giustificata in sezione 3.2.4. Inoltre, c'è un alto numero di valori mancanti per l'attributo *Usage Attributes*. Per tale attributo, particolarmente sparso sin dall'acquisizione dei dati, non è stato possibile individuare informazioni specifiche per migliorarne la completezza.

Per il dataset **Moves**, su 821 righe:

Attributo	Totale di NaN	% di NaN
Name	0	0
Introduced_in	0	0
Type	0	0
Power	346	42,14
Acc.	216	26,31
PP	20	2,44
Damage_class	33	4,02
Effect	31	3,78
Prob. (%)	591	71,99

Tabella 3. Completezza Moves

- l'assenza dell'effetto per 31 mosse è legata alla presenza di alcune mosse che non possiedono effetti secondari. Allo stesso modo, l'elevato numero di valori mancanti per *Prob. (%)* è dovuto al fatto che questo attributo non è nullo solo per quel numero limitato di mosse che hanno un effetto secondario che si attiva con una certa probabilità: pertanto, se l'effetto secondario di una mossa si attiva sempre questo attributo è nullo;
- i valori mancanti per l'attributo *Power* corrispondono alle mosse *Status*, che non infliggono danno, mentre per l'assenza del valore per l'attributo *Acc.* indica che la mossa non può mai fallire;
- i valori mancanti per *Damage_class* corrispondono alle mosse *G-Max Moves*, che non hanno una classe di danno fissa.

Per quanto riguarda il dataset **Abilities** è stato riscontrato un solo valore mancante per l'attributo *Description*. Infine, i dataset **Natures** e **Types** non hanno valori mancanti.

Riassumiamo quanto detto, mostrando in questo caso la completezza a livello di intero dataset:

Dataset	% di NaN	Totale di NaN	Totale valori
Pokémon	20,08	4851	24156
Items	14,27	839	5880
Moves	16,74	1237	7389
Abilities	0,12	1	801
Natures	0	0	0
Types	0	0	0

Tabella 4. Completezza interi dataset

4.3 Consistency

Uno degli ostacoli principali incontrati nella fase di arricchimento e integrazione è stata la scarsa coerenza tra i vari dataset acquisiti, in modo particolare rispetto ai nomi delle diverse entità. Per garantire coerenza nel prodotto finale, è stato necessario intervenire sui vari dataset per rendere uniforme il formato utilizzato.

Di seguito vengono elencate le principali operazioni eseguite a riguardo:

- **Consistency fra i nomi:** per diverse delle entità considerate è stato necessario intervenire per uniformare i valori dell'attributo *Name*. Nello specifico:
 - **Nomi dei Pokémon:** avendo un grande numero di dataset riferiti all'entità Pokémon, le operazioni necessarie sono state molteplici:
 - * **Nomi dei Pokémon Gigantamax:** nel dataset **Gigantamax Pokémon** alcuni nomi sono stati corretti manualmente, mentre è stato generato l'attributo *Variant*, contenente per ogni record la stringa *Gigantamax*.

- * **Nomi nel dataset Pokémon integrato:** come descritto in sezione 3.3.3, dopo aver aggiunto le varianti *Gigantamax* al dataset **Pokémon**, è stato generato un nuovo attributo *Name* come somma della vecchia variabile *Name* (che indicava solo il nome della specie, non univoco perché condiviso dalle varianti della stessa specie) e di *Variant*, se non nulla (nel caso in cui il Pokémon sia la forma base). Il formato dato al nome di ogni Pokémon, e mantenuto per il resto del progetto, è dato da ***Nome della specie + Nome Variante***.
- * **Nomi dei Pokémon in alcune tabelle ponte:** all'interno delle tabelle **ponte_pokemon_ability_MAY_HAS** e **ponte_pokemon_moves_MAY_LEARN** i valori dell'attributo *Pokémon* di entrambe le tabelle sono stati trattati rimuovendo i “-” e rendendo maiuscola la prima lettera di ogni parola. Per quei nomi non ancora armonizzati è stato inoltre necessario effettuare un'opportuna trasformazione mediante l'utilizzo delle apposite tabelle di “traduzione” ottenute mediante *fuzzy matching*.
- * **Nomi dei Pokémon nei Pokédex regionali:** nei 3 dataset dei Pokédex regionali sono stati rimossi dall'attributo *Name* i caratteri giapponesi. Inoltre, per le forme regionali è stata aggiunta la stringa *Galarian*.
- * **Trattamento dei riferimenti ai Pokémon nelle liste del regolamento Series 12 VGC 2022:** come spiegato nel dettaglio in sezione 3.6.1, durante la fase di generazione dell'attributo *VGC2022_rules* nel dataset **Pokémon** è stato necessario superare il problema dell'eterogeneità con cui, all'interno delle liste dei Pokémon *Permitted*, *Restricted* e *Banned*, era fatto riferimento ai Pokémon. In 3 delle liste era utilizzato come riferimento l'ID in uno dei rispettivi 3 Pokédex regionali, in due liste era usato l'ID nel Pokédex nazionale e nelle due restanti era utilizzato il nome della specie. I riferimenti delle liste sono stati uniformati, utilizzando il Nome completo (***Nome della specie + Nome Variante***) del Pokémon.
- * **Nomi nei dataset e nelle tabelle ponte di Pikalytics:** per gli attributi contenenti nomi di Pokémon nelle tabelle **Pokémon Usage Dataset**, **Teammates Usage** e **Moves Usage** sono state generate delle tabelle di “traduzione” per ricondurli al formato utilizzato nel dataset **Pokémon**, come trattato nel dettaglio in sezione 3.7.1.
- **Nomi di strumenti, abilità e mosse:** gli attributi *Name* dei dataset **Items API**, **Items**, **Abilities API**, **Abilities**, **Moves API** e **Moves** sono stati trasformati sostituendo i simboli “-” con degli spazi e rendendo maiuscola la prima lettera di ogni parola, al fine di facilitare la fase di integrazione, come descritto in sezione 3.2.4 e 3.2.5.
- **Nomi delle mosse da Pikalytics:** analogamente a quanto fatto per i nomi dei Pokémon, anche per l'attributo *Move* di **Moves Usage** i valori sono stati trasformati mediante le appositamente create tabelle di “traduzione”, al fine di uniformarli al formato usato per i nomi delle mosse del dataset **Moves**, come descritto in sezione 3.7.1.
- **Consistency degli ID:** nei dataset dei Pokédex regionali ottenuti da *serebii.net* è stato necessario trasformare i valori dell'attributo *No.*, relativo all'ID del Pokédex, da stringhe a interi rimuovendo i caratteri non numerici.

Al termine del lungo processo di pulizia, integrazione e arricchimento, quindi, le rappresentazioni dei nomi delle principali entità del database sono consistenti fra le diverse tabelle. In particolare, per i nomi dei Pokémon, che hanno richiesto il maggior numero di accorgimenti, il formato utilizzato è stato quello con ***Nome della specie + Nome Variante***, che permette di identificare univocamente ogni Pokémon.

5. Data Model and Storage

Una volta opportunamente puliti e integrati i dati, si è passati alla fase di data modelling. In questa fase viene definito il modello più adatto a rappresentare le nostre informazioni. Come già accennato in precedenza, la scelta è ricaduta su un database a grafo. Questa scelta è dettata principalmente:

- per la natura relazionale dei dati e dalla volontà di mettere in risalto le relazioni fra gli elementi più rilevanti del mondo competitivo Pokémon.
- dal rapporto tra operazioni di scrittura sul database e quelle di lettura: le statistiche di utilizzo provenienti dal gioco competitivo vengono aggiornate mensilmente, di conseguenza le operazioni di scrittura avvengono in numero decisamente minore rispetto a quelle di lettura. Proprio per questo motivo risulta più idoneo l'utilizzo di un database a grafo, per cui il costo computazionale delle operazioni di scrittura risulta maggiore rispetto a quelle di lettura.

La base di dati è stata implementata tramite Neo4J, uno dei più popolari database a grafo open source. La scelta è ricaduta su di esso principalmente perché:

- è un database a grafo nativo.
- è un database totalmente transazionale, integrabile nelle applicazioni permettendone quindi il funzionamento stand-alone.

Il database, che presenta la struttura tradizionale dei database a grafo, è caratterizzato da:

- **Nodi**: descrivono le entità del dominio di riferimento. I nodi possono avere zero o più **labels**, che ne definiscono il **tipo**.
- **Archi**: descrivono le connessioni fra un nodo di partenza e uno di arrivo, modellando le relazioni fra le entità del dominio di riferimento. Anche le relazioni vengono caratterizzate da un **tipo** specifico.
- **Proprietà**: coppie chiave-valore, che descrivono ulteriormente le caratteristiche sia dei nodi che degli archi.

5.1 Definizione dei nodi

Con i dati a nostra disposizione, la progettazione del DBMS ha visto la definizione dei seguenti 6 tipi di nodi:

- **Pokémon**: entità che descrive le principali caratteristiche di un Pokémon e le sue statistiche legate al gioco competitivo aggiornate a Marzo 2022. Per tale entità sono state considerate le seguenti proprietà:
 - *NationalPokedexNumber*: Intero. ID del Pokémon all'interno del Pokédex nazionale. Valore condiviso fra tutte le varianti della stessa specie.
 - *Name*: Stringa. Nome completo del Pokémon (*Nome della specie + Nome variante*).
 - *Species*: Stringa. Nome della specie del Pokémon.
 - *Variant*: Stringa. Nome della specifica variante (se nullo, il record specifico fa riferimento a un Pokémon base).
 - *Generation*: Intero. Generazione di giochi in cui il Pokémon è stato introdotto.
 - *Rarity*: Stringa. Livello di rarità del Pokémon.
 - *Has_gender_diff*: Booleano. Indica se esistono o meno variazioni di aspetto fra Pokémon di sesso maschile e femminile.
 - *Total/HP/Attack/Defense/Sp.Atk/Sp.Def/Speed*: Interi. Statistiche base del Pokémon.
 - *image_url*: Stringa. Link allo *sprite* del Pokémon.
 - *VGC2022_rules*: Stringa. Status competitivo in accordo al regolamento *VGC 2022 Series 12*.
 - *Monthly Usage (k)*: Intero. Numero di migliaia team che hanno visto schierato il Pokémon specifico.
 - *Usage Percent (%)*: Intero. Percentuale di team (sul totale dei team schierati) che hanno visto l'utilizzo del Pokémon specifico.
 - *Monthly Rank*: Intero. Posizionamento nella classifica dei Pokémon più utilizzati.
- **Moves**: entità che descrive le principali caratteristiche di una mossa, una *skill* che un Pokémon può usare, una volta per turno, in battaglia. Per tale entità sono state considerate le seguenti proprietà:
 - *Name*: Stringa. Nome completo della mossa.
 - *Introduced_in*: Intero. Generazione di giochi in cui è stata introdotta la mossa.
 - *Power*: Intero. Potenza di base della mossa.
 - *Acc.*: Intero. Precisione di base della mossa.
 - *PP*: Intero. Quantità di possibili utilizzi in battaglia della mossa.
 - *Damage_class*: Stringa. Indica la classe di danno della mossa.
 - *Effect*: Stringa. Descrizione dell'effetto secondario correlato all'utilizzo della mossa.
 - *Prob. (%)*: Intero. Probabilità che si attivi l'effetto secondario della mossa (se nullo, l'effetto secondario si attiva sempre).
- **Items**: entità che descrive le principali caratteristiche di uno strumento, elementi fondamentali del gioco competitivo che tramite i loro effetti possono contribuire al potenziamento di Pokémon e mosse. Per tale entità sono state considerate le seguenti proprietà:
 - *Name*: Stringa. Nome completo dello strumento.
 - *Effect*: Stringa. Descrizione dell'effetto dello strumento.
 - *Effect (alt.)*: Stringa. Descrizione alternativa dell'effetto dello strumento.

- *Usage Attributes*: Lista di stringhe. Lista degli attributi specifici dello strumento.
- *General Category*: Stringa. Categoria generale dell'oggetto.
- *Specific Category*: Stringa. Categoria specifica dell'oggetto.
- *image_url*: Stringa. Link allo *sprite* dello strumento.
- **Abilities**: entità che descrive le caratteristiche di una specifica abilità, attributi speciali di Pokémon, che possono agire come potenziamento, aumentando il danno di una mossa o una statistica base, introdurre effetti particolari come una condizione meteorologica e, in generale, risultare determinanti per l'esito di una battaglia. Per tale entità sono state considerate le seguenti proprietà:
 - *Name*: Stringa. Nome completo dell'abilità.
 - *Description*: Stringa. Descrizione dell'effetto dell'abilità.
 - *Generation*: Intero. Generazione di giochi in cui è stata introdotta l'abilità.
- **Natures**: entità che descrive le principali caratteristiche di una specifica natura, *feature* che indica la "personalità" di un Pokémon e che ne influenza alcune statistiche. Per tale entità sono state considerate le seguenti proprietà:
 - *Name*: Stringa. Nome della natura.
 - *Increases*: Stringa. Statistica aumentata del 10% dalla specifica natura.
 - *Decreases*: Stringa. Statistica diminuita del 10% dalla specifica natura.
 - *Likes_berrie*: Stringa. Bacca (tipo di strumento) apprezzata dal Pokémon che possiede la specifica natura.
 - *Dislikes_berrie*: Stringa. Bacca (tipo di strumento) disprezzata dal Pokémon che possiede la specifica natura.
- **Type**: entità che descrive le caratteristiche di un tipo, caratteristica peculiare di Pokémon e mosse che influenzano il calcolo dei danni inflitti da una mossa a un Pokémon. Per tale entità sono state considerate le seguenti proprietà:
 - *Name*: Stringa. Nome completo del tipo.
 - *Generation*: Intero. Generazione di giochi in cui è stato introdotto il tipo.

5.2 Definizione delle relazioni

Dopo aver specificato tutte le entità di interesse, si è passati a definire le relazioni fra di esse. Sono state definite le seguenti 13 relazioni:

- Relazione *Pokémon* —→ ***Pokemon HAS_VARIANT***: collega ciascun Pokémon con la sua (o le sue) variante.
- Relazione *Pokémon* —→ ***Pokemon EVOLVES_FROM***: collega ciascun Pokémon con il Pokémon da cui esso evolve.
- Relazione *Pokémon* —→ ***Type IS_OF_TYPE***: collega ciascun Pokémon con il suo (o i suoi, per un massimo di 2) tipo.
- Relazione *Pokémon* —→ ***Moves MAY LEARN***: collega ciascun Pokémon con **ogni** mossa che può apprendere.
- Relazione *Pokémon* —→ ***Moves G-MAX MOVE***: collega ciascun Pokémon *Gigantamax* con la sua peculiare *G-Max Move*.
- Relazione *Pokémon* —→ ***Abilities MAY HAS***: collega ciascun Pokémon con **ogni** abilità che può possedere. Possiede la proprietà:
 - *Is_hidden*: Booleano. Indica se, per il Pokémon specifico, l'abilità è nascosta o meno.
- Relazione *Moves* —→ ***Type MOVES_IS_TYPE***: collega ciascuna mossa con il suo tipo.
- Relazione *Type* —→ ***MOVE_EFFECTIVENESS_ON_POKEMON***: collega ogni tipo (riferito alla mossa effettuata) con tutti gli altri tipi (riferito al Pokémon che subisce la mossa), incluso sé stesso. Possiede le proprietà:
 - *Effectiveness*: Stringa. Indica l'effetto del tipo della mossa attaccante sul tipo del Pokémon difensore (*non efficace*, *poco efficace*, *normalmente efficace*, *superefficace*).
 - *DamageMultiplier*: Stringa. Indica il moltiplicatore da applicare al calcolo dei danni.

- Relazione *Pokémon* → *Pokémon USED_IN_TEAM_WITH*: collega ciascun Pokémon con i compagni con cui è utilizzato più frequentemente nei match competitivi. Possiede la proprietà:
 - ***UsagePercentage***: Float. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato insieme al compagno specifico sul totale delle volte in cui è stato schierato.
- Relazione *Pokémon* → *Items USED_WITH_ITEM*: collega ciascun Pokémon con gli strumenti con cui è utilizzato più frequentemente nei match competitivi. Possiede la proprietà:
 - ***UsagePercentage***: Float. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con assegnata lo specifico strumento sul totale delle volte in cui è stato schierato.
- Relazione *Pokémon* → *Moves USED_WITH_MOVE*: collega ciascun Pokémon con le mosse con cui è utilizzato più frequentemente nei match competitivi. Possiede la proprietà:
 - ***UsagePercentage***: Float. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con assegnata la specifica mossa sul totale delle volte in cui è stato schierato.
- Relazione *Pokémon* → *Abilities USED_WITH_ABILITY*: collega ciascun Pokémon con le abilità con cui è utilizzato più frequentemente nei match competitivi. Possiede la proprietà:
 - ***UsagePercentage***: Float. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con assegnata la specifica abilità sul totale delle volte in cui è stato schierato.
- Relazione *Pokémon* → *Nature COMMON_SPREAD*: collega ciascun Pokémon con le nature con cui è utilizzato più frequentemente nei match competitivi. Possiede la proprietà:
 - ***HP/Atk/Def/SpA/SpD/Spe***: Stringa. Distribuzione dei punti EVs nelle diverse statistiche.
 - ***UsagePercentage***: Float. Percentuale ottenuta dal rapporto tra il numero di volte in cui il Pokémon è stato schierato con assegnata la specifica coppia *Nature/EVs Spread* sul totale delle volte in cui è stato schierato.

5.3 Model Schema

Come già precedentemente stabilito, in un database a grafo non è necessario definire uno schema a priori. Nel nostro caso, al fine di mantenere un certo grado di qualità e consistenza, è stato deciso di introdurre comunque alcune restrizioni. Nello specifico, per ogni entità è stato applicato un **vincolo di unicità** per la proprietà *Name*: ciò significa che all'interno del DBMS non è possibile inserire due nodi dello stesso tipo aventi lo stesso valore per la proprietà *Name*, necessaria, per altro, per la successiva creazione delle singole relazioni tramite tabelle ponte.

Imponendo un vincolo su una proprietà di un nodo, viene creato anche un indice riferito a quella proprietà. Questo consente di migliorare le performance in fase di interrogazione del DBMS, dal che il sistema riesce ad individuare più rapidamente il punto di partenza del grafo.

I vincoli introdotti sono riassunti di seguito:

Constraints
ON (abilities:Abilities) ASSERT (abilities.Name) IS UNIQUE
ON (item:Item) ASSERT (item.Name) IS UNIQUE
ON (moves:Moves) ASSERT (moves.Name) IS UNIQUE
ON (nature:Nature) ASSERT (nature.Name) IS UNIQUE
ON (type>Type) ASSERT (type.Name) IS UNIQUE
ON (pokemon:Pokemon) ASSERT (pokemon.Name) IS UNIQUE

Figura 54. Vincoli imposti nel DBMS

Index Name	Type	Uniqueness	EntityType	LabelsOrTypes	Properties	State
AbilitiesNameKey	BTREE	UNIQUE	NODE	["Abilities"]	["Name"]	ONLINE
ItemNameKey	BTREE	UNIQUE	NODE	["Item"]	["Name"]	ONLINE
MovesNameKey	BTREE	UNIQUE	NODE	["Moves"]	["Name"]	ONLINE
NatureNameKey	BTREE	UNIQUE	NODE	["Nature"]	["Name"]	ONLINE
TypesNameKey	BTREE	UNIQUE	NODE	["Type"]	["Name"]	ONLINE
index_343aff4e	LOOKUP	NONUNIQUE	NODE	[]	[]	ONLINE
index_f7700477	LOOKUP	NONUNIQUE	RELATIONSHIP	[]	[]	ONLINE
pokemonNameKey	BTREE	UNIQUE	NODE	["Pokemon"]	["Name"]	ONLINE

Figura 55. Indici introdotti con l'imposizione dei vincoli

Al termine del processo di data modelling, lo schema assume, dunque, la seguente forma:

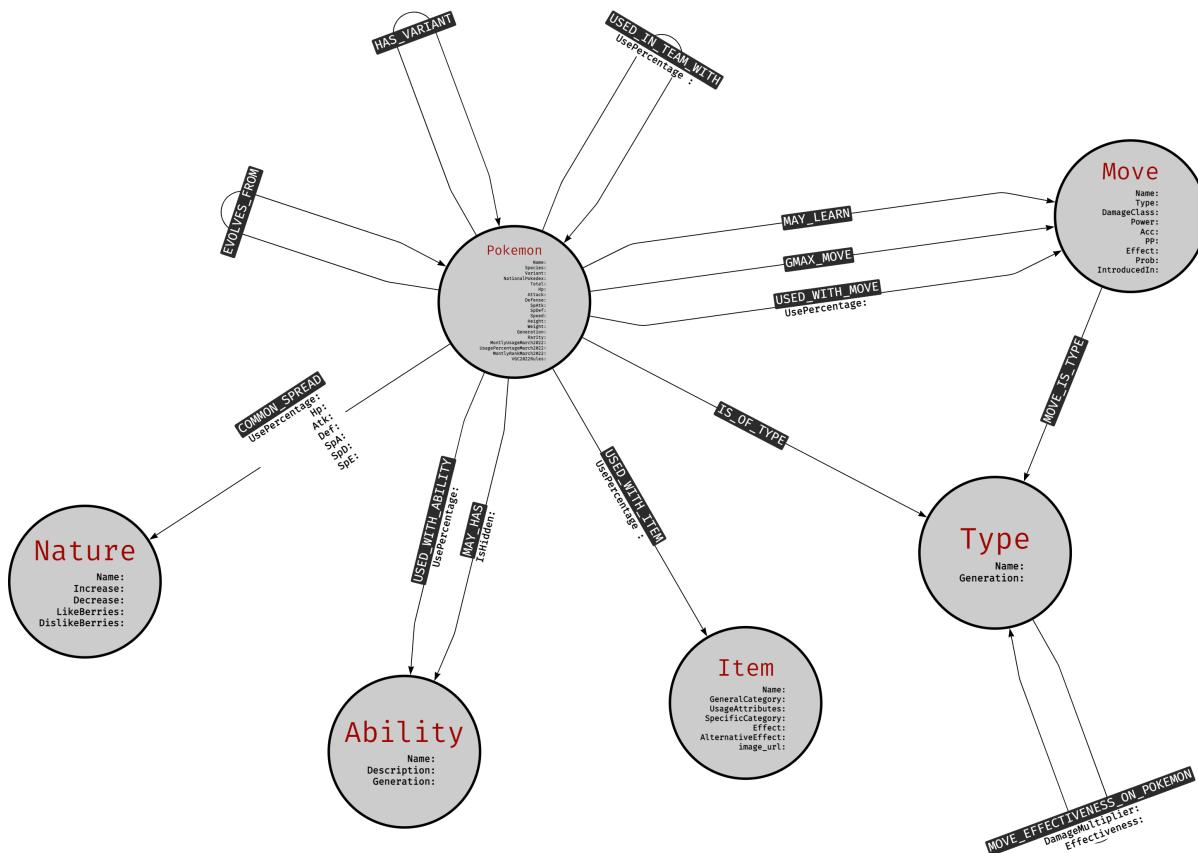


Figura 56. Pokémon Graph Model

5.4 Data Storage

La fase di data storage vede l'effettiva creazione del DBMS e la memorizzazione dei dati al suo interno. Per questo progetto è stata utilizzata la versione Community Server di Neo4J[17]. Le operazioni di **CREATE**, **UPDATE** e **DELETE** sono state realizzate tramite Python, attraverso il driver ufficiale *Neo4J*[18] che permette di stabilire una connessione al database e l'esecuzione delle varie operazioni direttamente da un notebook Python.

La creazione dei nodi e delle relazioni ha necessitato di una breve fase di pre-processing dei dati, al fine di uniformare i datasets con le regole e convezioni imposte da Neo4J. Le convenzioni sono riferite principalmente al modo in cui denominare le varie componenti del database, mentre le regole riguardano caratteri ammessi e non ammessi e i tipi di dati supportati.

L'intera fase di import ha prodotto un totale di **3071 nodi e 88.869 relazioni**.

5.4.1 Import nodi

L'import dei nodi è stato effettuato tramite la procedura **LOAD CSV** di Neo4J[19]. Inoltre, dal momento che, come mostrato precedentemente nello schema, non tutti i nodi presentano le stesse proprietà, è stata implementata una procedura che, per ogni record del file .csv:

- crea il nodo con la proprietà *Name* come chiave;
- per ciascuna proprietà controlla se nel file è presente il dato in input;
- se presente, aggiunge la proprietà e inserisce il relativo valore. In caso contrario, non inserisce la proprietà.

In questo modo, all'interno di un nodo non viene inserita una proprietà nodo qualora il dato fosse mancante, dal momento che non è garantito che il dato di una specifica proprietà sia disponibile per tutti i nodi. Come esempio, viene di seguito riportata la procedura utile alla creazione dei nodi di tipo *Moves*.

```
// Cypher Language
LOAD CSV WITH HEADERS FROM 'https://raw.githubusercontent.com/df\_moves\_import.csv' AS row
MERGE (m:Moves {Name: row.Name}) # Property Key
FOREACH(ignoreMe IN CASE WHEN trim(row.IntroducedIn) <> "" THEN [1] ELSE [] END | SET
    m.IntroducedIn = toInteger(row.IntroducedIn))
FOREACH(ignoreMe IN CASE WHEN trim(row.Type) <> "" THEN [1] ELSE [] END | SET m.Type =
    row.Type)
FOREACH(ignoreMe IN CASE WHEN trim(row.Power) <> "" THEN [1] ELSE [] END | SET m.Power =
    toInteger(row.Power))
FOREACH(ignoreMe IN CASE WHEN trim(row.Acc) <> "" THEN [1] ELSE [] END | SET m.Acc =
    toInteger(row.Acc))
FOREACH(ignoreMe IN CASE WHEN trim(row.PP) <> "" THEN [1] ELSE [] END | SET m.PP =
    toInteger(row.PP))
FOREACH(ignoreMe IN CASE WHEN trim(row.DamageClass) <> "" THEN [1] ELSE [] END | SET
    m.DamageClass = row.DamageClass)
FOREACH(ignoreMe IN CASE WHEN trim(row.Effect) <> "" THEN [1] ELSE [] END | SET m.Effect =
    row.Effect)
FOREACH(ignoreMe IN CASE WHEN trim(row.ProbPercentage) <> "" THEN [1] ELSE [] END | SET
    m.ProbPercentage = toInteger(row.ProbPercentage))
RETURN m
```

5.4.2 Import relazioni

L'operazione di import delle relazioni avviene sfruttando le tabelle ponte create in fase di cleansing e integrazione, sempre attraverso la procedura **LOAD CSV** di Neo4J. Ciò è possibile in quanto, per ogni tipo di nodo, è stato imposto un vincolo di unicità per la proprietà *Name*: questo garantisce un *MATCH* preciso per ogni nodo. Come esempio, viene di seguito viene riportata la procedura utile all'import della relazione **IS_OF_TYPE** tra i nodi di tipo *Pokémon* e *Type*:

```
// Cypher Language
LOAD CSV WITH HEADERS FROM
    "https://raw.githubusercontent.com/ponte\_pokemon\_type\_IS\_OF\_TYPE\_import.csv" AS row
MATCH (p:Pokemon {Name: row.Name}), (t:Type {Name: row.Type})
CREATE (p)-[:IS_OF_TYPE]->(t)
```

6. Graph Exploration

6.1 Query

L'interrogazione di un database Neo4J avviene tramite **Cypher**[20], linguaggio di tipo dichiarativo e *pattern-matching*. Anche in questo caso, tutte le interrogazioni sono state effettuate direttamente da un notebook Python, tramite la libreria **py2neo**[21] e **pprint**[22], utile per la stampa del risultato della query.

Come esempio, vengono riportate delle possibili interrogazioni che possono essere effettuate sul database:

- **Query1:** Quali sono le varianti del Pokemon *Charizard*?

```
// Cypher Language

MATCH (p:Pokemon) -[r:HAS_VARIANT]-> (v:Pokemon)
WHERE p.Name='Charizard'
RETURN v.Name AS Variante
```

Le varianti di *Charizard* sono *Mega Charizard X*, *Mega Charizard Y* e *Charizard Gigantamax*.

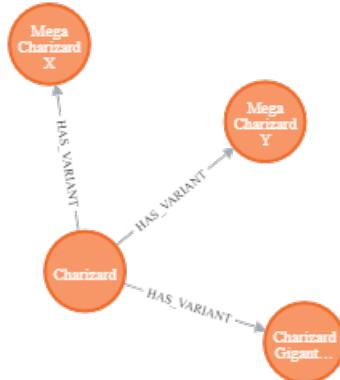


Figura 57. Visualizzazione query 1

- **Query2:** Durante il mese di marzo 2022, quali sono i compagni di squadra più frequenti del Pokemon *Charizard*? (È stata considerata una soglia del 30% per la proprietà *UsagePercentage* della relazione considerata)

```
// Cypher Language

MATCH (p:Pokemon)-[r:USED_IN_TEAM_WITH]->(t:Pokemon)
WHERE p.Name='Charizard' AND r.UsePercentage > 30
RETURN t.Name AS TeamMate, r.UsePercentage AS Percentual
ORDER BY r.UsePercentage DESC
```

I compagni più frequenti di *Charizard* sono *Groudon* (86.83%), *Incineroar* (70.40%), *Zacian Crowned Sword* (58.2%), *Grimmsnarl* (57.68%) e *Gastrodon* (47.79%).

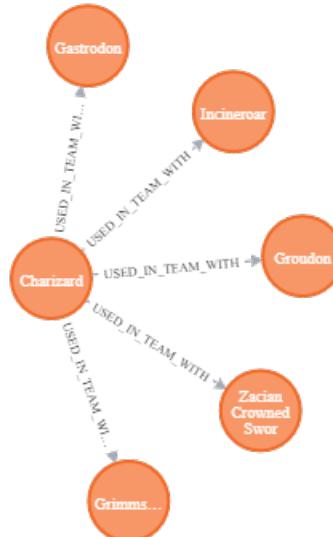


Figura 58. Visualizzazione query 2

- **Query3:** Quali, tra le mosse più utilizzate dal Pokémon *Charizard*, sono superefficaci sul Pokémon *Arcanine*?

- **Step 1:** Cerco il tipo del Pokémon *Arcanine*

```
// Cypher Language

MATCH (p:Pokemon{Name:'Arcanine'})-[rt:IS_OF_TYPE]-> (t:Type)
RETURN t.Type
```

Arcanine risulta essere di tipo *Fire*.

- **Step 2:** Cerco, fra le mosse più utilizzate da *Charizard*, quali sono quelle superefficaci su Pokémon di tipo *Fire*

```
// Cypher Language

MATCH (p:Pokemon {Name:'Charizard'})-[r:USED_WITH_MOVE]-> (m:Moves)
WHERE r.UsePercentage > 30
MATCH (p)-[rt:IS_OF_TYPE]-> (tp:Type)
MATCH (m)-[rm:MOVES_IS_TYPE]-> (tm)
CALL {
    WITH tm
    MATCH (t)-[eff:MOVE_EFFECTIVENESS_ON_POKEMON]-> (teff:Type)
    WHERE teff.Name='fire' AND eff.Effectiveness='Super-effective (200%)'
    RETURN eff, teff
}
RETURN p,r,m,tm,rm,eff
```

Tra le mosse più utilizzate da *Charizard* (*UsagePercentage* maggiore del 30%) la mossa che risulta essere superefficace su *Arcanine* è *Ancient Power* (utilizzo del 48%).

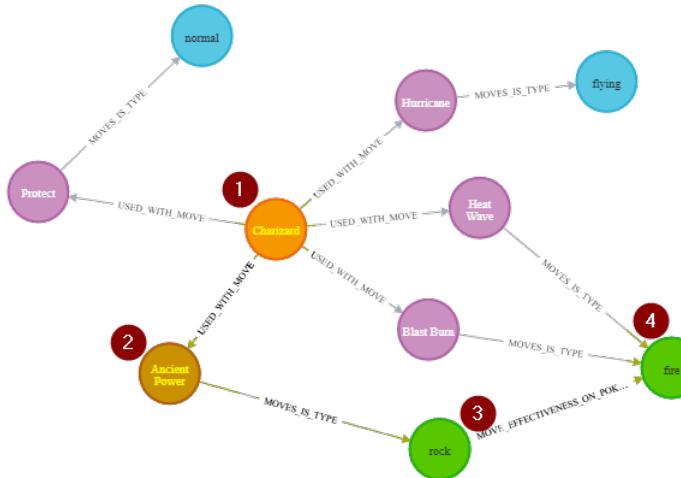


Figura 59. Visualizzazione query 3

- **Query 4:** Quali sono i Pokèmon di tipo *Fire* che vengono utilizzati con l'abilità *Solar Power*?

```
// Cypher Language

MATCH (p:Pokemon)-[r:IS_OF_TYPE]-> (t:Type {Name:'fire'})
CALL {
    WITH p
    MATCH (p)-[u:USED_WITH_ABILITY]-> (a:Abilities {Name:'Solar Power'})
    RETURN u,a
}
RETURN p.Name
```

I Pokèmon che rispondono alla query effettuata risultano essere *Mega Houndoom*, *Charmander*, *Charmeleon* e *Charizard*.

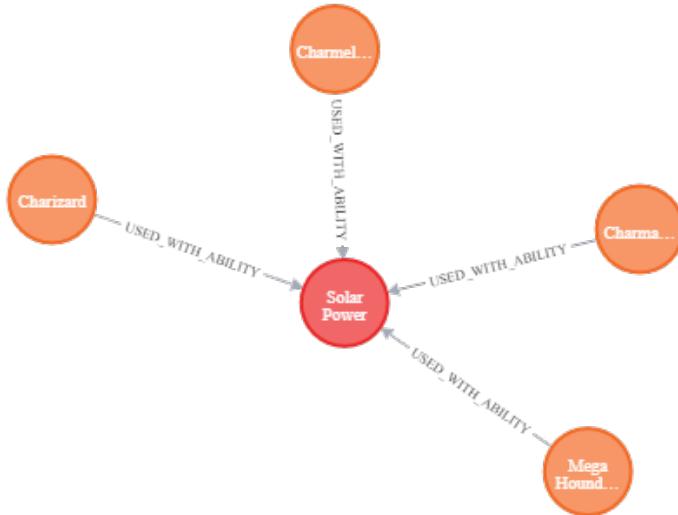


Figura 60. Visualizzazione query 4

6.2 Competitive Data Graph Visualization

Tramite i dati acquisiti in sezione 2.2.3, riferiti ai dati dei match competitivi svolti a Marzo 2022 sulla piattaforma **Pokémon Showdown!**[13], è possibile svolgere una serie di analisi per rendere più immediata la comprensione del comportamento dei giocatori competitivi in termini di:

- Pokèmon compagni di squadra più utilizzati;
- Mosse più utilizzate;
- Abilità più utilizzate;
- Strumenti più utilizzati.

Le analisi implementabili sono molteplici: in questa sezione ci limiteremo soltanto a produrre alcune visualizzazioni che aiutino a comprendere il comportamento del network. Le visualizzazioni sono state prodotte tramite il tool open-source **Gephi**[23], che consente di visualizzare e interagire con molteplici tipologie di grafi. Al netto di alcune leggere modifiche necessarie per lo storage dei dati in *Gephi*, sono state utilizzate le stesse tabelle usate per l'importazione in Neo4J.

6.2.1 Compagni di squadra più utilizzati

Si vanno a considerare i seguenti dati:

- Nodi di tipo *Pokémon*;
- Relazione *Pokémon* → *Pokémon* **USED_IN_TEAM_WITH**, che collega ciascun Pokémon con i compagni con cui è utilizzato più frequentemente nei match competitivi.

La forma del grafo è stata ottenuta con l'algoritmo *ForceAtlas 2*. La dimensione delle etichette è proporzionale al *grado entrante*, ovvero al numero di relazioni che un nodo ha in ingresso: più è alto il numero di relazioni che un Pokémon ha in ingresso, più la dimensione della relativa label sarà grande. Ciò significa che, se il grado entrante è alto tale Pokémon è spesso usato in squadra con altri Pokémon:

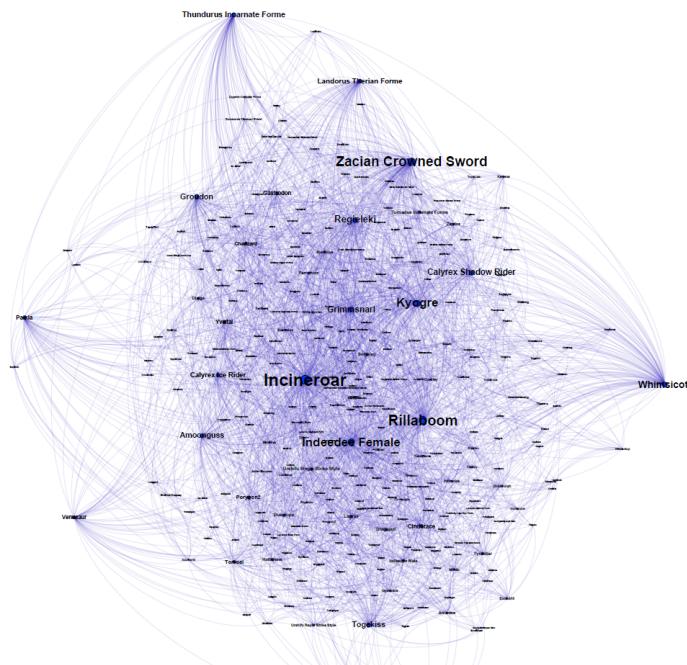


Figura 61. Compagni di squadra maggiormente utilizzati

6.2.2 Mosse maggiormente utilizzate

In questo caso vengono considerati i seguenti dati:

- Nodi di tipo *Pokémon*;
- Nodi di tipo *Moves*;
- Relazione *Pokémon* → *Moves USED WITH MOVE*, che collega ciascun Pokémon con le mosse con cui è utilizzato più frequentemente nei match competitivi.

È stato utilizzato sempre l'algoritmo *ForceAtlas 2*. Come in precedenza, la dimensione delle label è proporzionale al *grado entrante*: in questo caso se il grado entrante è alto significa che tale mossa è spesso utilizzata nei match competitivi:

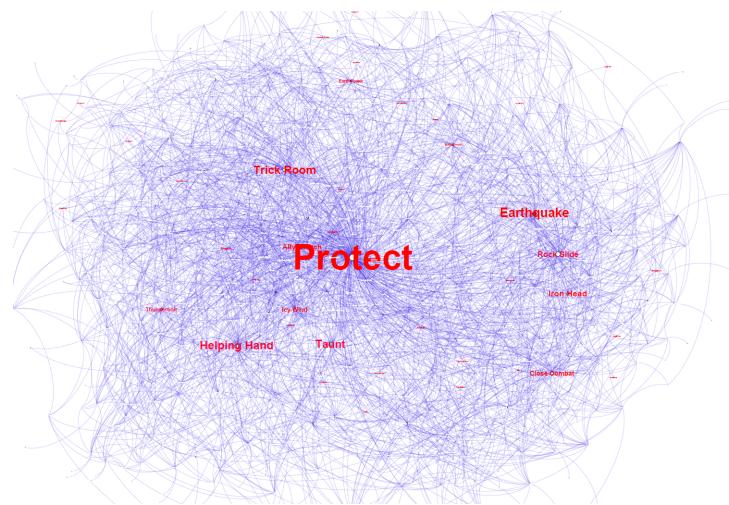


Figura 62. Mosse maggiormente utilizzate

Sitografia

- [1] *PokéAPI: The RESTful Pokémon API.* URL: <https://pokeapi.co/>.
- [2] *Requests: HTTP for Humans.* URL: <https://requests.readthedocs.io/en/master/>.
- [3] *Pokémon Database: the fastest way to get your Pokémon information.* URL: <https://pokemondb.net/>.
- [4] *Pikalytics: VGC 2022 Series 12 Stats Pokédex.* URL: <https://www.pikalytics.com/>.
- [5] *Serebii.net - Where Legends Come To Life.* URL: <https://www.serebii.net/>.
- [6] *Victory Road: 2022 Play! Pokémon Season Structure.* URL: <https://victoryroadvgc.com/2020-season/>.
- [7] *Bulbapedia: the community-driven Pokémon encyclopedia.* URL: https://bulbapedia.bulbagarden.net/wiki/Main_Page.
- [8] *Beautiful Soup. Libreria per web scraping.* URL: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>.
- [9] *html.parser — Simple HTML and XHTML parser.* URL: <https://docs.python.org/3/library/html.parser.html>.
- [10] *Pandas: flexible and easy to use open source data analysis and manipulation tool.* URL: <https://requests.readthedocs.io/en/master/>.
- [11] *Selenium with Python.* URL: <https://selenium-python.readthedocs.io/>.
- [12] *Microsoft Edge WebDriver.* URL: <https://developer.microsoft.com/en-us/microsoft-edge/tools/webdriver/>.
- [13] *Pokémon Showdown!. A web-based open-source Pokémon battle simulator.* URL: <https://play.pokemonshowdown.com/>.
- [14] *Python Record Linkage Toolkit: a library to link records in or between data sources.* URL: <https://recordlinkage.readthedocs.io/en/latest/about.html>.
- [15] *Levenshtein distance.* URL: https://en.wikipedia.org/wiki/Levenshtein_distance.
- [16] *N-gram similarity and distance.* URL: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.67.9369&rep=rep1&type=pdf>.
- [17] *Neo4j Community Server.* URL: <https://neo4j.com/download-center/#community>.
- [18] *Neo4j Driver.* URL: <https://neo4j.com/docs/api/python-driver/current/>.
- [19] *LOAD CSV Procedure.* URL: <https://neo4j.com/docs/cypher-manual/current/clauses/load-csv/>.
- [20] *The Neo4j Cypher Manual.* URL: <https://neo4j.com/docs/cypher-manual/current/>.
- [21] *The Py2neo Handbook.* URL: <https://py2neo.org/2021.1/>.
- [22] *Data pretty printer.* URL: <https://docs.python.org/3/library/pprint.html>.
- [23] *The Open Graph Viz Platform.* URL: <https://gephi.org/>.